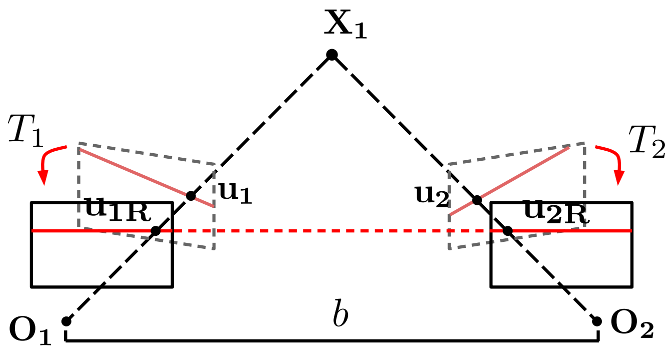
# Computer Vision



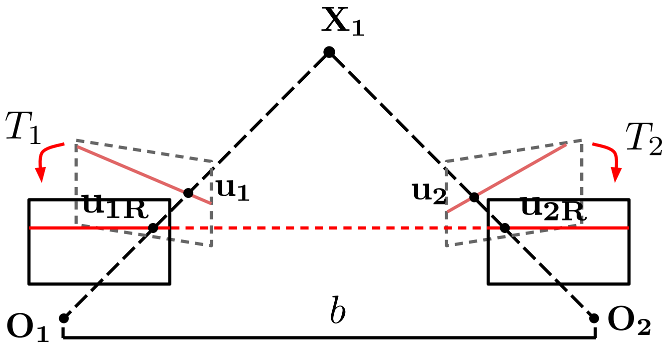

Class 11

Raquel Frizera Vassallo

# Summary

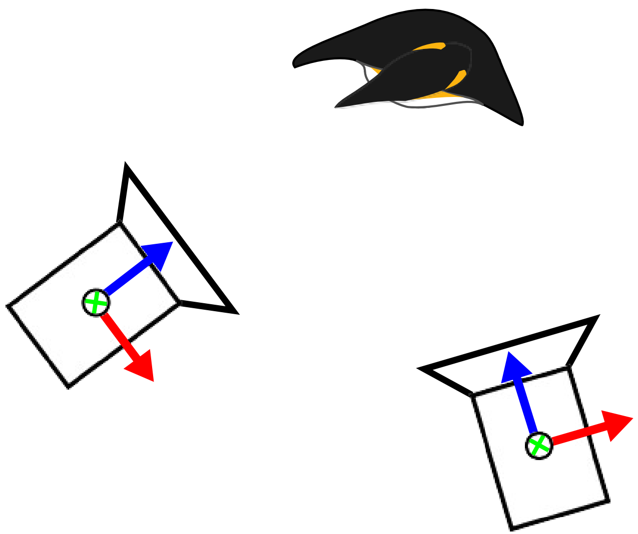
- Image Rectification
- 3D reconstruction from general stereo system through rectification

# Image Rectification



# General case

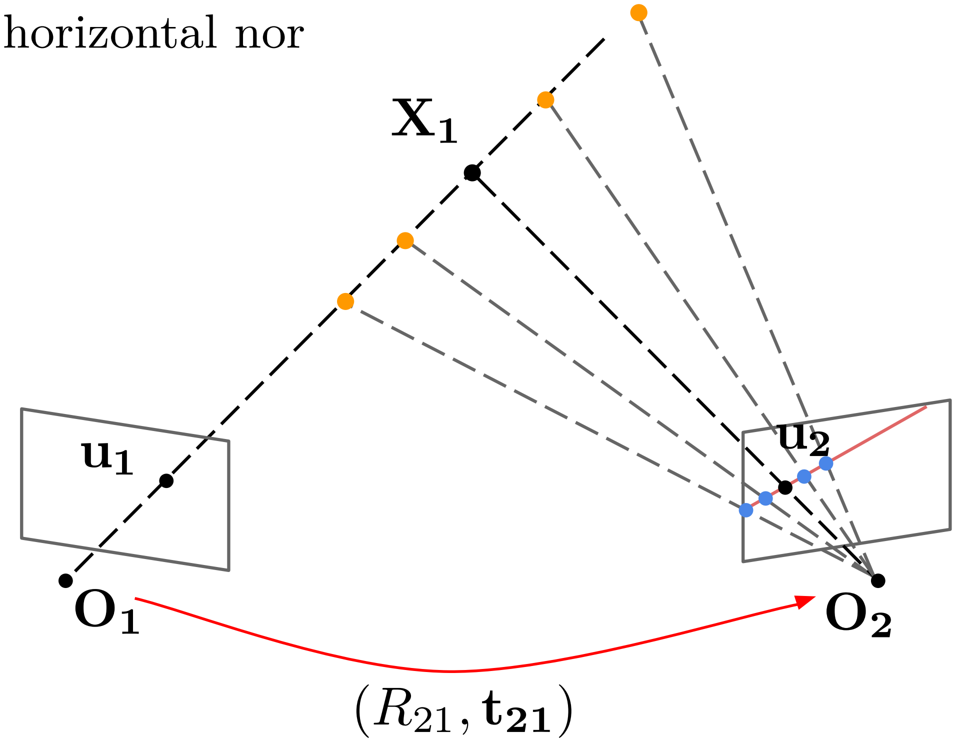
Usually the cameras are not disposed like a Rectified System.



# General case

In such systems the epipolar lines are neither horizontal nor parallel.

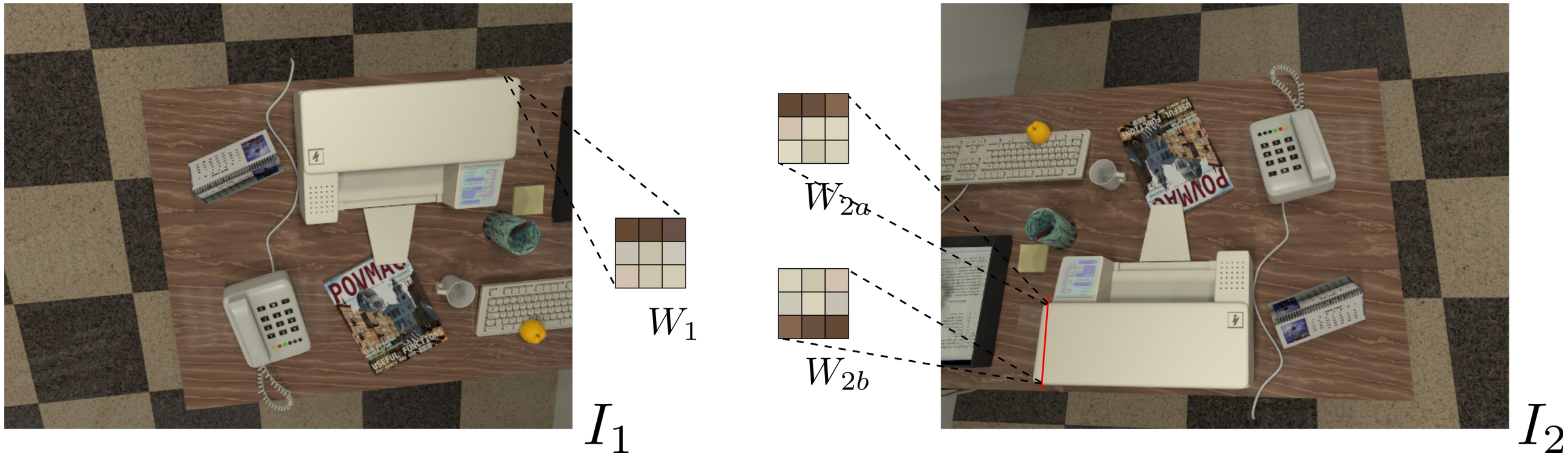
$$z_2\mathbf{u_2} = K_2(R_{21}(z_1K_1^{-1}\mathbf{u_1}) + \mathbf{t_{21}})$$



# General case

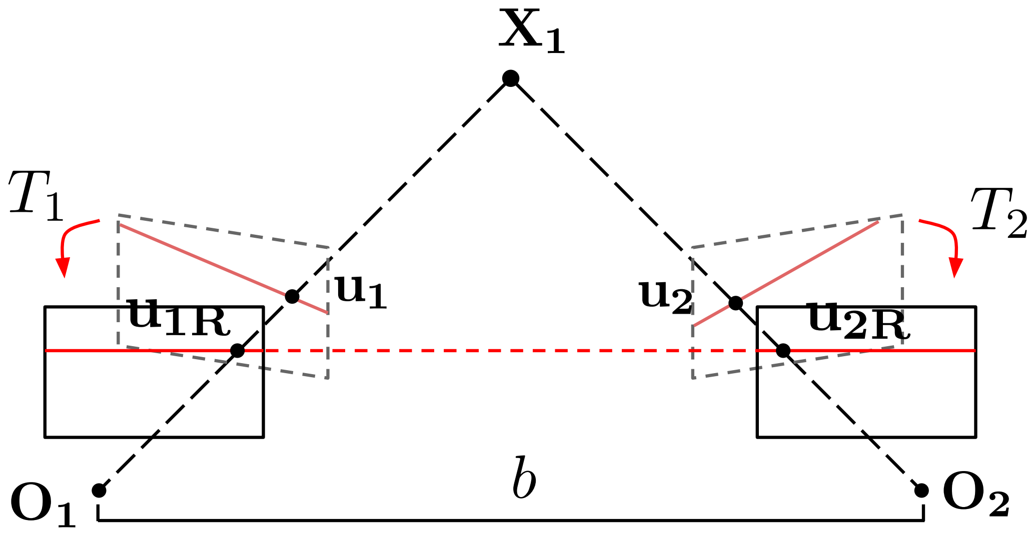
We have seen that we can find correspondences by traversing the epipolar line.

- Lines are diagonal $\rightarrow$ not ideal to compare pixel windows
- Images with different orientation also make it difficult



# Solution - Rectification

The solution to such problem is the image rectification.



# Solution - Rectification

The solution to such problem is the image rectification.

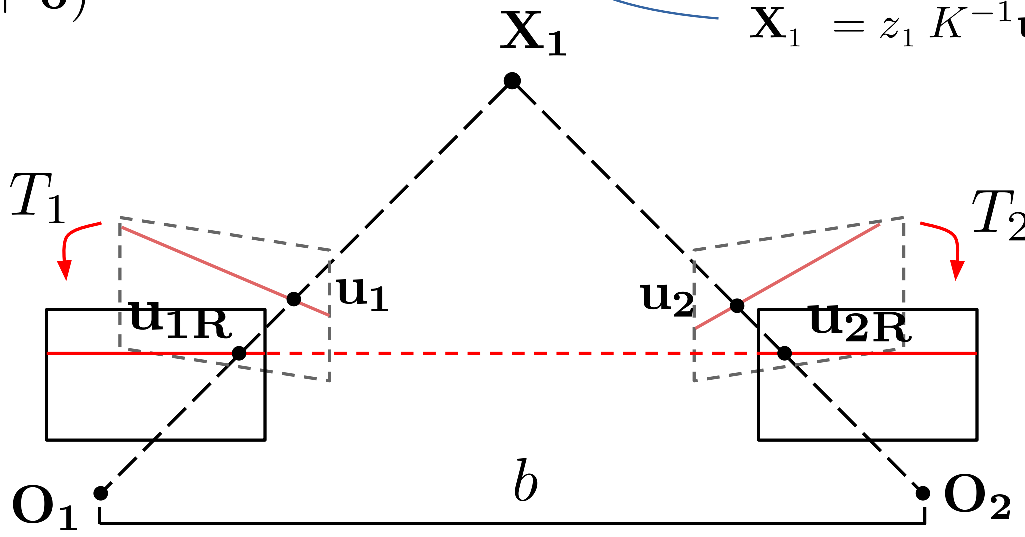
$$z_R \mathbf{u_{1R}} = K(R_{R1}(z_1 K^{-1}\mathbf{u_1}) + \mathbf{0})$$

$$z_R \mathbf{u_{1R}} = z_1 K R_{R1} K^{-1}\mathbf{u_1}$$

$$z_R \mathbf{u_{1R}} = z_1 T_1 \mathbf{u_1}$$

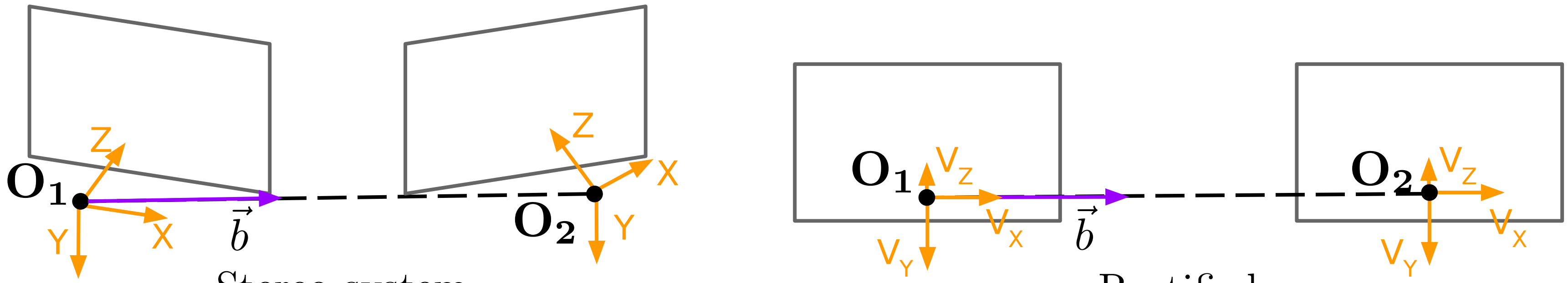
$$z_1 \mathbf{u}_1 = K\mathbf{X}_1$$

$$\mathbf{X}_1 = z_1 K^{-1}\mathbf{u}_1$$



# Solution - Rectification

But, how to obtain the matrices $T_1$ and $T_2$?




Stereo system

Rectified stereo system

# Solution - Rectification

1: $\vec{b} = \mathbf{O_2} - \mathbf{O_1}$

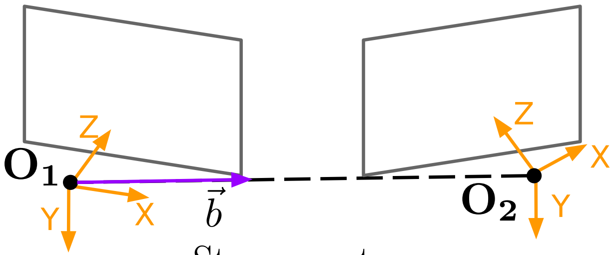
2: $\vec{\mathrm{v}}_x = \dfrac{\vec{b}}{\|\vec{b}\|_2}$

3: $\vec{\mathrm{v}}_y = \dfrac{\vec{z} \times \vec{\mathrm{v}}_x}{\|\vec{z} \times \vec{\mathrm{v}}_x\|_2}$

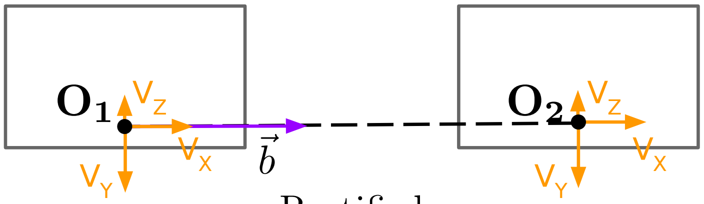
4: $\vec{\mathrm{v}}_z = \dfrac{\vec{\mathrm{v}}_x \times \vec{\mathrm{v}}_y}{\|\vec{\mathrm{v}}_x \times \vec{\mathrm{v}}_y\|_2}$

5: $R_{R1} = \begin{bmatrix} \vec{\mathrm{v}}_x^{\mathrm{T}} \\ \vec{\mathrm{v}}_y^{\mathrm{T}} \\ \vec{\mathrm{v}}_z^{\mathrm{T}} \end{bmatrix}$

$$\boxed{\begin{aligned} T_1 &= K R_{R1} K^{-1} \\ T_2 &= K R_{R1} R_{12} K^{-1} \end{aligned}}$$




Stereo system

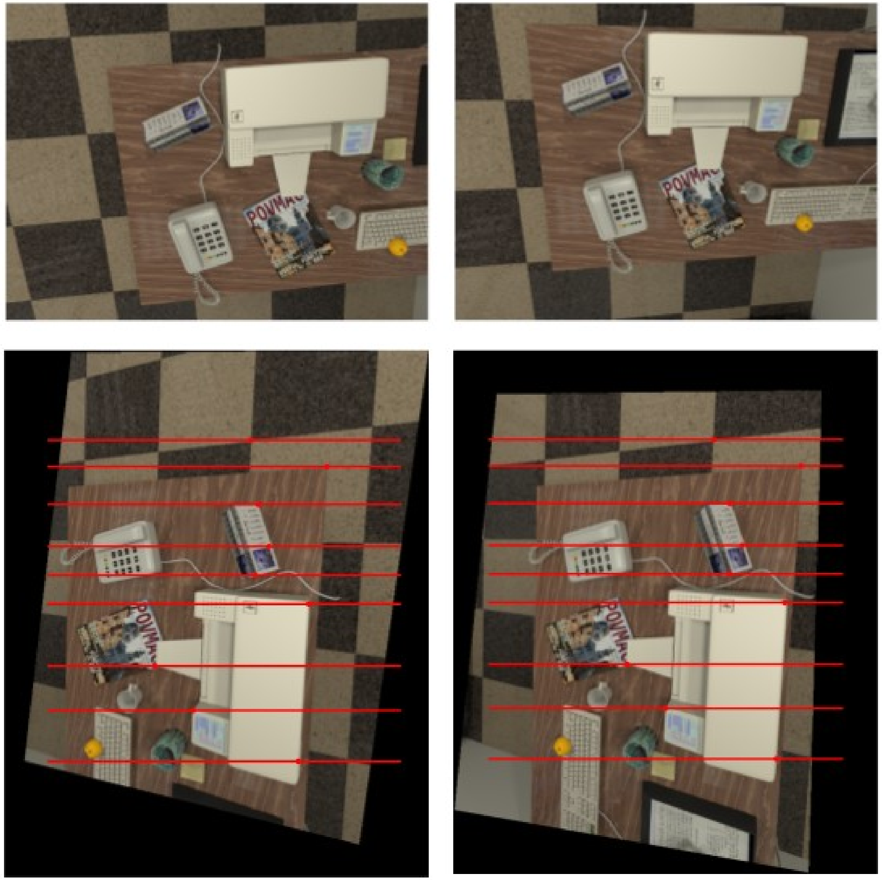



Rectified stereo system

# Rectification

$$z_R \mathbf{u_{1R}} = z_1 T_1 \mathbf{u_1}$$

$$z_R \mathbf{u_{2R}} = z_2 T_2 \mathbf{u_2}$$

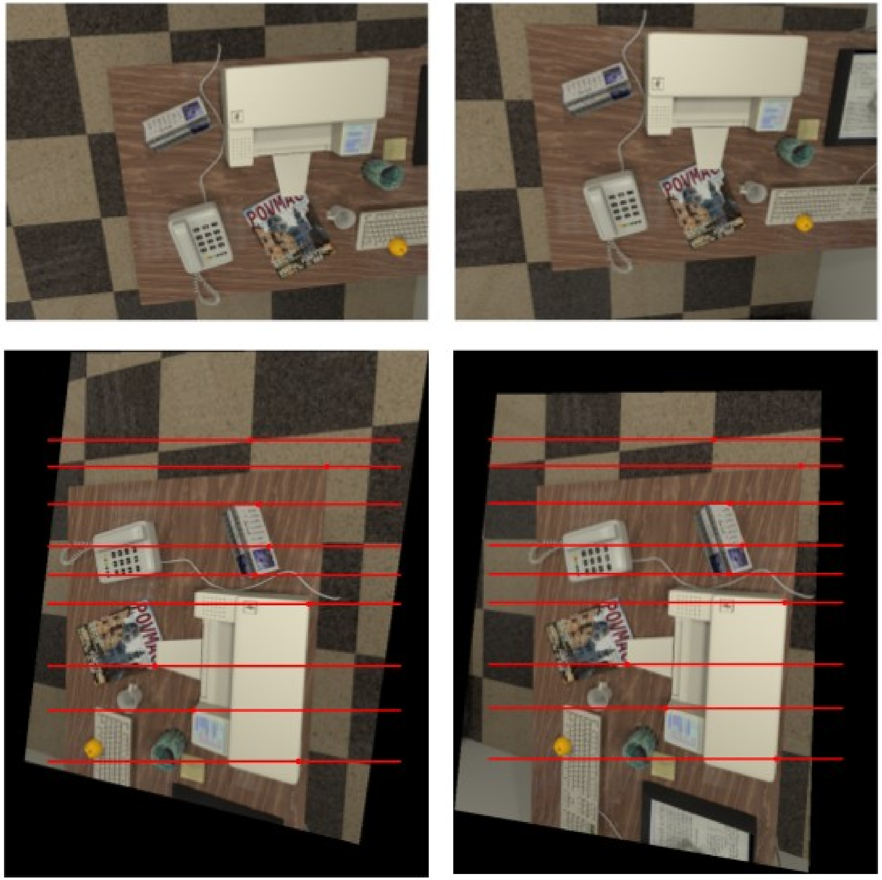


# Rectification

$$z_R \mathbf{u_{1R}} = z_1 \boxed{T_1 \mathbf{u_1}}$$

$$z_R \mathbf{u_{2R}} = z_2 T_2 \mathbf{u_2}$$

$$\boxed{\begin{bmatrix} \alpha_1 \\ \beta_1 \\ \gamma_1 \end{bmatrix} = T_1 \begin{bmatrix} u_1 \\ v_1 \\ 1 \end{bmatrix}}$$



# Rectification

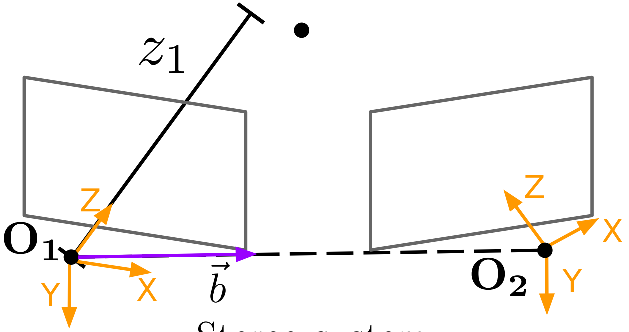
$$z_R \mathbf{u_{1R}} = z_1 \boxed{T_1 \mathbf{u_1}}$$

$$z_R \mathbf{u_{2R}} = z_2 T_2 \mathbf{u_2}$$

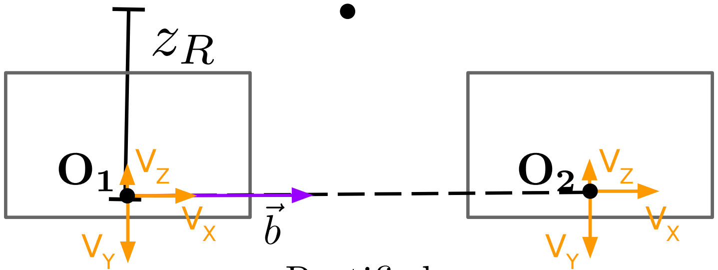
$$\boxed{\begin{bmatrix} \alpha_1 \\ \beta_1 \\ \gamma_1 \end{bmatrix} = T_1 \begin{bmatrix} u_1 \\ v_1 \\ 1 \end{bmatrix}}$$

$$\begin{bmatrix} z_R u_{1R} \\ z_R v_{1R} \\ z_R \end{bmatrix} = \begin{bmatrix} z_1 \alpha_1 \\ z_1 \beta_1 \\ z_1 \gamma_1 \end{bmatrix}$$

$$\boxed{z_R \neq z_1}$$




Stereo system




Rectified
stereo system

# Rectification

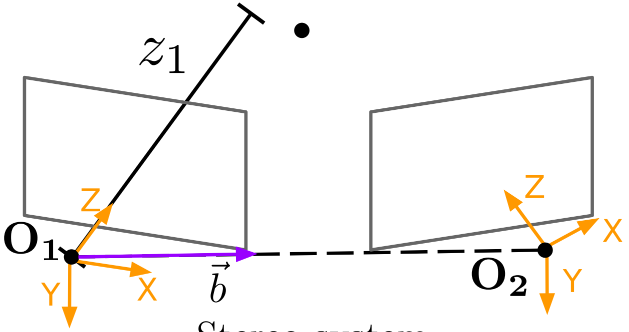
$$z_R \mathbf{u_{1R}} = z_1 \boxed{T_1 \mathbf{u_1}}$$

$$z_R \mathbf{u_{2R}} = z_2 T_2 \mathbf{u_2}$$

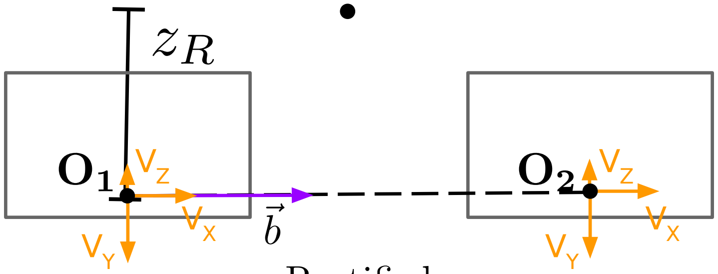
$$\boxed{\begin{bmatrix} \alpha_1 \\ \beta_1 \\ \gamma_1 \end{bmatrix} = T_1 \begin{bmatrix} u_1 \\ v_1 \\ 1 \end{bmatrix}}$$

$$\begin{bmatrix} z_R u_{1R} \\ z_R v_{1R} \\ z_R \end{bmatrix} = \begin{bmatrix} z_1 \alpha_1 \\ z_1 \beta_1 \\ z_1 \gamma_1 \end{bmatrix}$$

$$\boxed{z_R = \gamma_1 z_1}$$




Stereo system




Rectified
stereo system

# Rectification (advantages)

Advantages of image pair rectification are:

- fix the orientation of the windows;
- computational gain (less computational intensive).

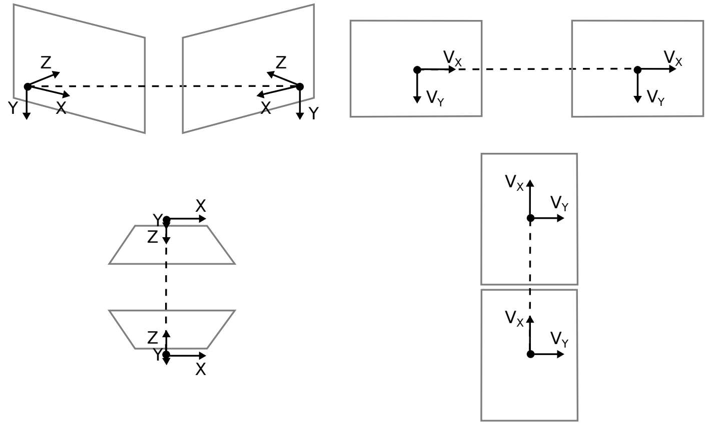
# Example 5

- Rectifying an image pair.

# Rectification (good cases)

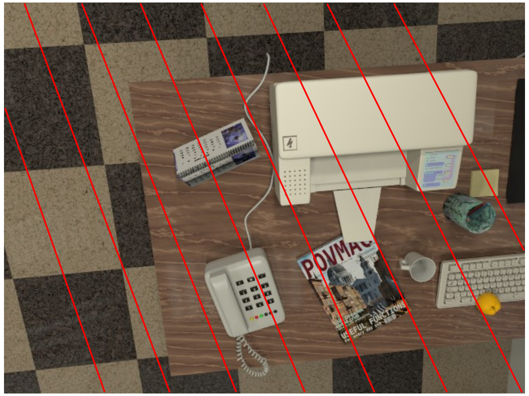
Rectification requires that the relative movement between the cameras is not predominantly foward.

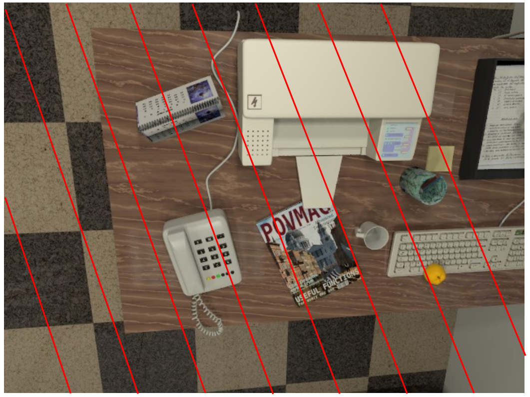
Stereo systems




Original

Rectified

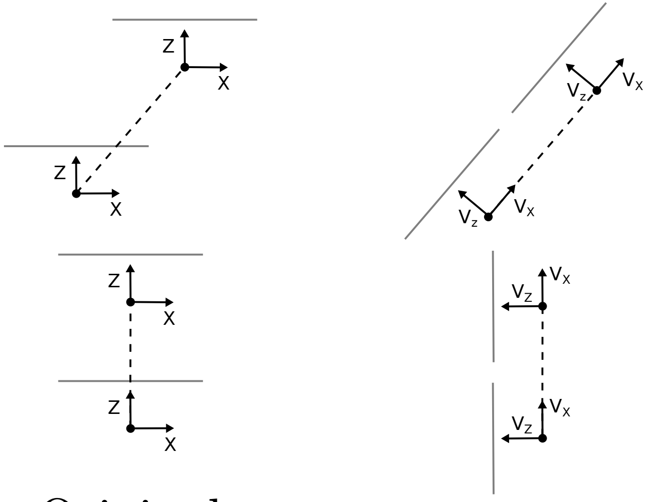




# Rectification (bad cases)

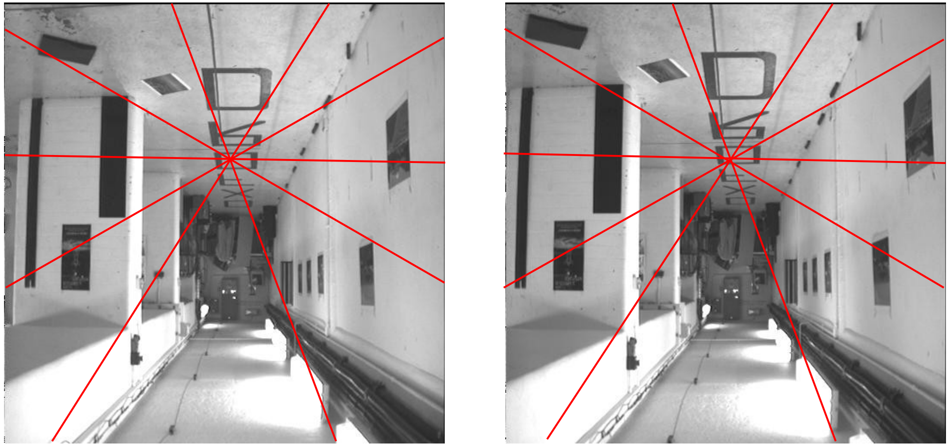
Rectification requires that the relative movement between the cameras is not predominantly foward.

Stereo systems




Original

Rectified



# Interesting things!

3D Cross eyed.

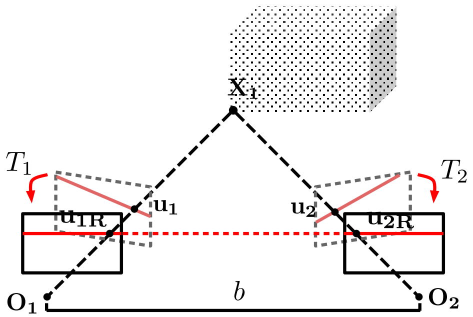
**3D crosseyed tutorial: http://www.starosta.com/3dshowcase/ihelp.html.**

# Interesting things!

3D Cross eyed for the "spot the seven errors" game.

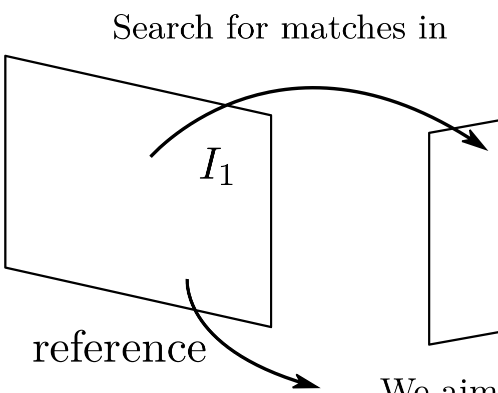
**Source: https://renklisheyler.files.wordpress.com/2012/05/jfira.jpg?w=700.**

# Rectifying images and Reconstruction



# 3D Reconstruction process

Goal: find $\boldsymbol{u_1}$ and $\boldsymbol{u_2}$, since $(\boldsymbol{u_1}, \boldsymbol{u_2}) \to z$.

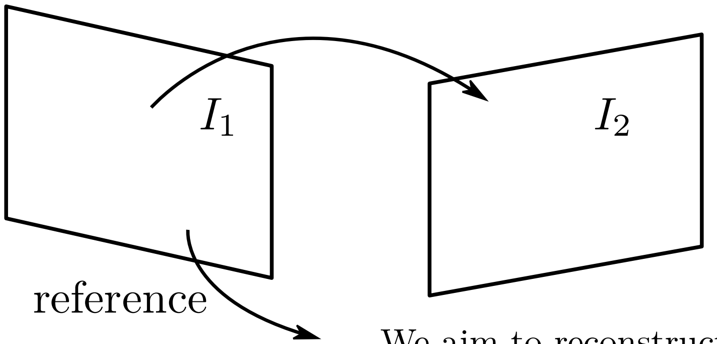



We aim to reconstruct all the pixels!
($480 \times 640 = 307200$ pixels)

# 3D Reconstruction process

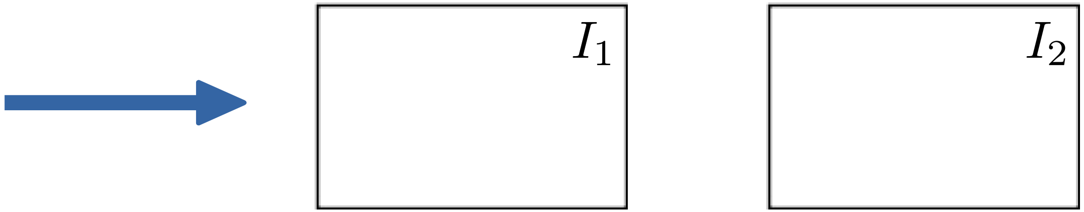
Goal: find $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$, since $(\boldsymbol{u}_1, \boldsymbol{u}_2) \to z$.

Search for matches in




We aim to reconstruct all the pixels!
$(480 \times 640 = 307200$ pixels$)$

**1st Step: Image rectification**

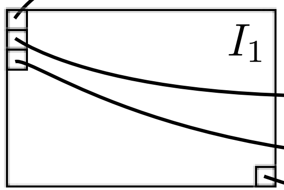


# 3D Reconstruction process

Goal: find $\boldsymbol{u_1}$ and $\boldsymbol{u_2}$, since $(\boldsymbol{u_1}, \boldsymbol{u_2}) \rightarrow z$.

**1st Step: Image rectification**

Stack each pixel coordinate horizontally:



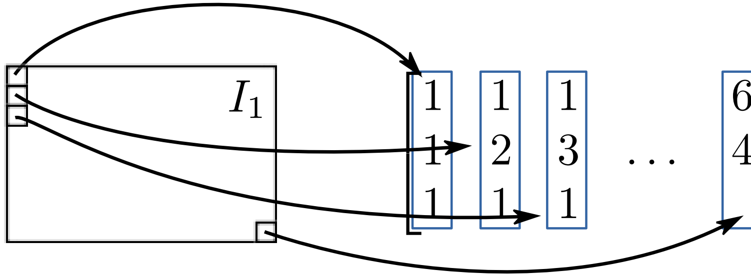

$$\begin{bmatrix} 1 & 1 & 1 & & 640 \\ 1 & 2 & 3 & \dots & 480 \\ 1 & 1 & 1 & & 1 \end{bmatrix} = \mathbf{U_1}$$

# 3D Reconstruction process

Goal: find $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$, since $(\boldsymbol{u}_1, \boldsymbol{u}_2) \rightarrow z$.

**1st Step: Image rectification**

Stack each pixel coordinate horizontally:




$$\begin{bmatrix} 1 & 1 & 1 & \dots & 640 \\ 1 & 2 & 3 & \dots & 480 \\ 1 & 1 & 1 & & 1 \end{bmatrix} = \mathbf{U_1}$$

- Compute $T_1$ and $T_2$, then rectify the image pair (warp)
- $\mathbf{U_{R1}} = T_1 \mathbf{U_1}$

$$\mathrm{z}_{R1} = \begin{bmatrix} \gamma_1 & \gamma_2 & \gamma_3 & \dots & \gamma_{307.200} \end{bmatrix} \mathrm{z}_1 = T_1(3,:)\, \mathbf{U_1}\, \mathrm{z}_1$$

$$\begin{bmatrix} \gamma_1 & \gamma_2 & \gamma_3 & \dots & \gamma_{307.200} \end{bmatrix} = T_1(3,:)\, \mathbf{U_1}$$

Remember the relation between $\mathrm{z}_1$ and $\mathrm{z}_{R1}$

# 3D Reconstruction process

Goal: find $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$, since $(\boldsymbol{u}_1, \boldsymbol{u}_2) \rightarrow z$.

**2nd Step: Find correspondences between $\boldsymbol{I}_1$ and $\boldsymbol{I}_2$**

For each point in $U_{R1}$ define the initial and final points in the epipolar lines of the second image to search for the correspondent points $U_{R2}$

$$U_{R2\,ini} = U_{R1} + \frac{b f s_x}{z_{ini}}$$

$$U_{R2\,final} = U_{R1} + \frac{b f s_x}{z_{final}}$$

- For each $u_{R\mathbf{1}}$ in $U_{R1}$ find:

$$u_{R2} = \text{matching}(u_{R2\text{ini}}, u_{R2\text{final}}, v_1, I_1, I_2)$$

- Then:

$$z_R = \frac{b f s_x}{(u_{R1} - u_{R2})}$$

$$z_1 = \frac{z_R}{\gamma 1}$$

# 3D Reconstruction process

Goal: find $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$, since $(\boldsymbol{u}_1, \boldsymbol{u}_2) \rightarrow z$.

**3rd Step: Recover the 3D points**

For each pair $u_{R1}$ and $u_{R2}$ find $z_R$:

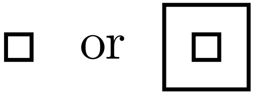
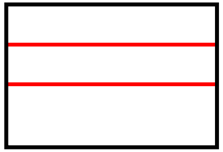
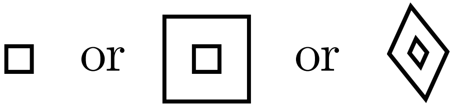
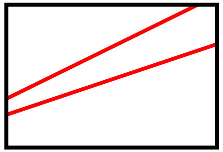
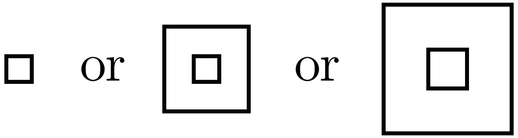
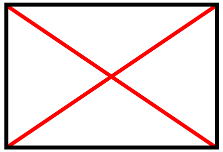
$$z_R = \frac{b f s_x}{(u_{R1} - u_{R2})}$$

- Then:

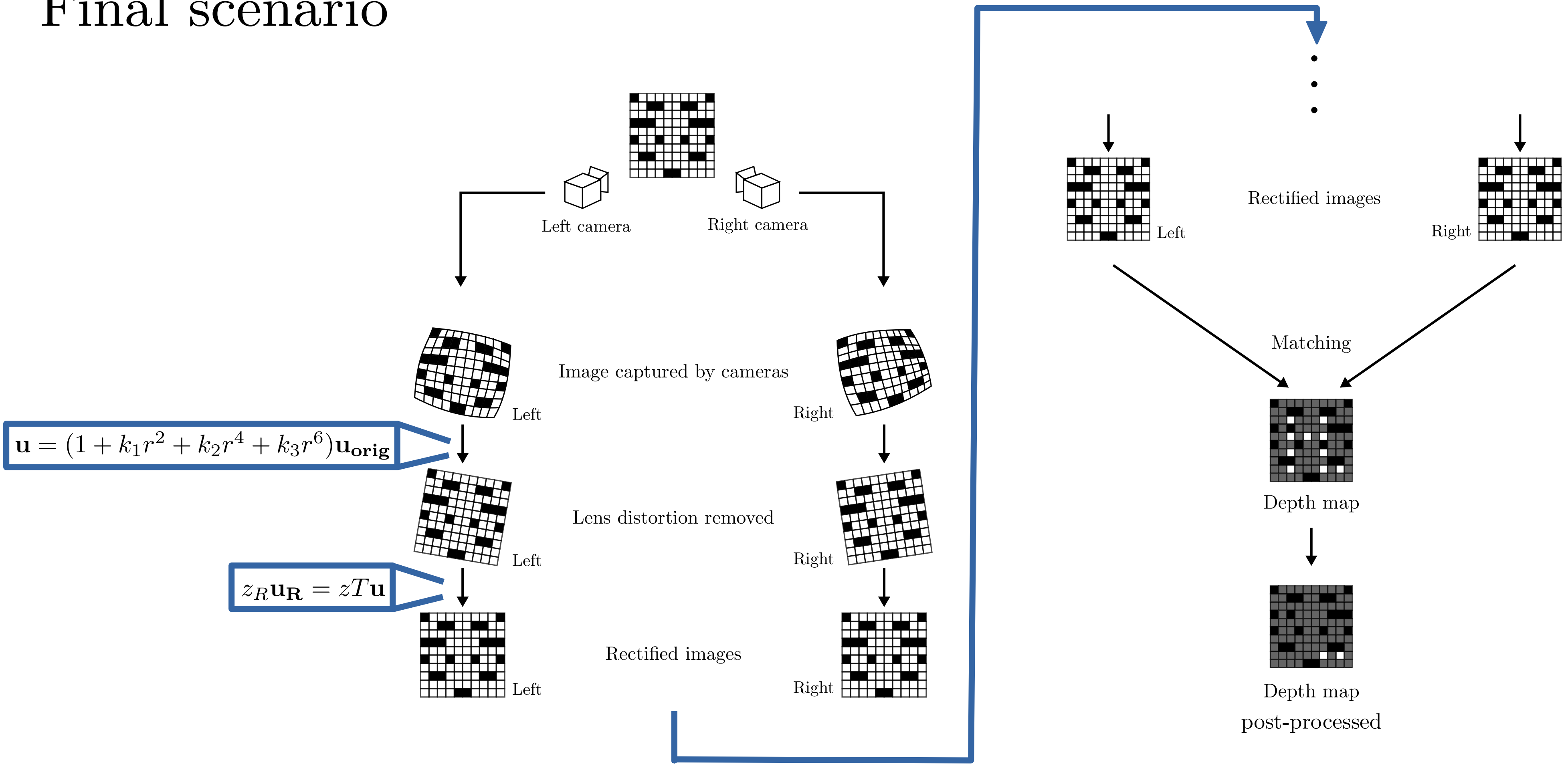
$$z_1 = \frac{z_R}{\gamma 1} \longrightarrow \mathbf{X}_1 = z_1 K^{-1} \mathbf{u}_1$$

Obtain original $z_1$ from $z_R$

# 3D Reconstruction process

| System | Matching | Epipolar line |
|---|---|---|
| Rectified |  $u_2^{k+1} = u_2^k + 1$ |  |
| XY movement (predominant) |  $z_2^{k+1}\mathbf{u_2^{k+1}} = K(z_1^{k+1}R_{21}K^{-1}\mathbf{u_1^k} + t_{21})$ |  |
| Forward movement (predominant) |  $z_2^{k+1}\mathbf{u_2^{k+1}} = K(z_1^{k+1}R_{21}K^{-1}\mathbf{u_1^k} + t_{21})$ |  |

# Final scenario



# Credits

- Andrea Fusiello. **Tutorial on rectification of stereo images.** http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/FUSIELLO/tutorial.html
- A. Fusiello, E. Trucco and A. Verri. **Epipolar rectification.** http://www.diegm.uniud.it/fusiello/demo/rect/