



University of Trento

Part V: Camera Geometry and Multi-View

Nicola Conci

From the beginning, in 2D



- Typically we represent a point (pixel) as:

$$P = [x, y]^t = \begin{bmatrix} x \\ y \end{bmatrix}$$

- Often convenient to use homogeneous coordinates:

$$P = [x, y]^t = [sx, sy, s]$$

- s is a scaling factor, commonly 1.0 $P = [x, y]^t = [x, y, 1]$
- For simplicity we can omit “ t ”

Affine transformations



- Spatial transformations represented as:
 - Multiplication of a matrix and a homogeneous point
- Different types
 - Scaling
 - Rotation
 - Translation
 - Combination of them

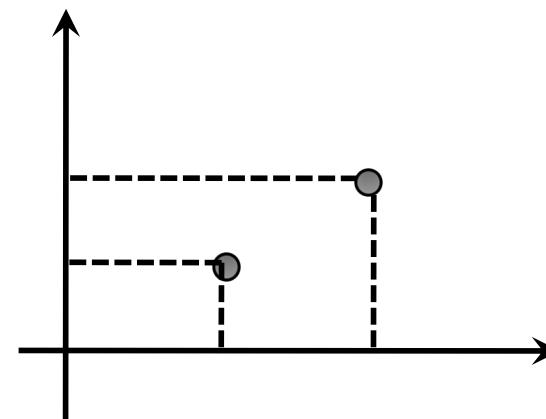
Scaling



- Linear transformation applied to all points. It can be uniform or non uniform:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} c & 0 \\ 0 & c \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} c_x & 0 \\ 0 & c_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

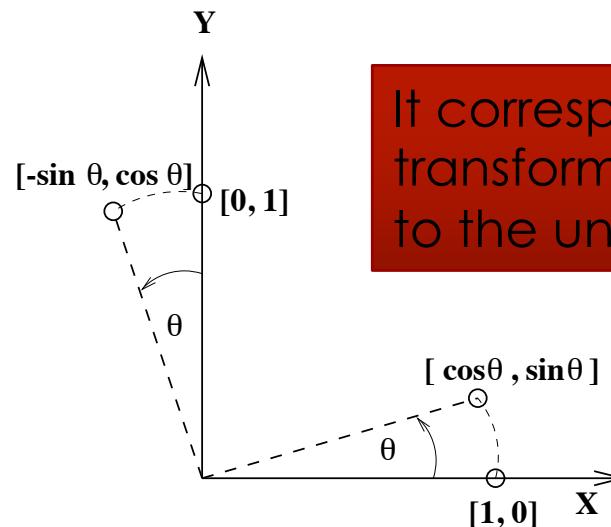
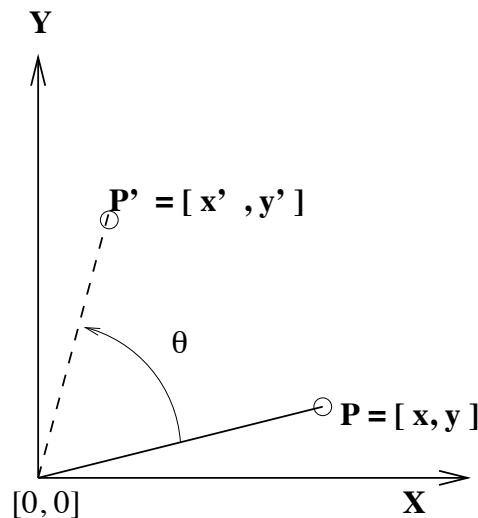


Rotation



- $P=[x,y]$ rotated by ϑ

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\vartheta & -\sin\vartheta \\ \sin\vartheta & \cos\vartheta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x\cos\vartheta - y\sin\vartheta \\ x\sin\vartheta + y\cos\vartheta \end{bmatrix}$$



It corresponds to the transformation applied to the unit vectors



Translation

- Uniform shift of the object points
- Equivalent to changing the origin
- The displacement $D([x,y]) = [x+x_0, y+y_0]$
- Using the homogeneous coordinates:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x + x_0 \\ y + y_0 \\ 1 \end{bmatrix}$$

Rotation, scaling, and translation

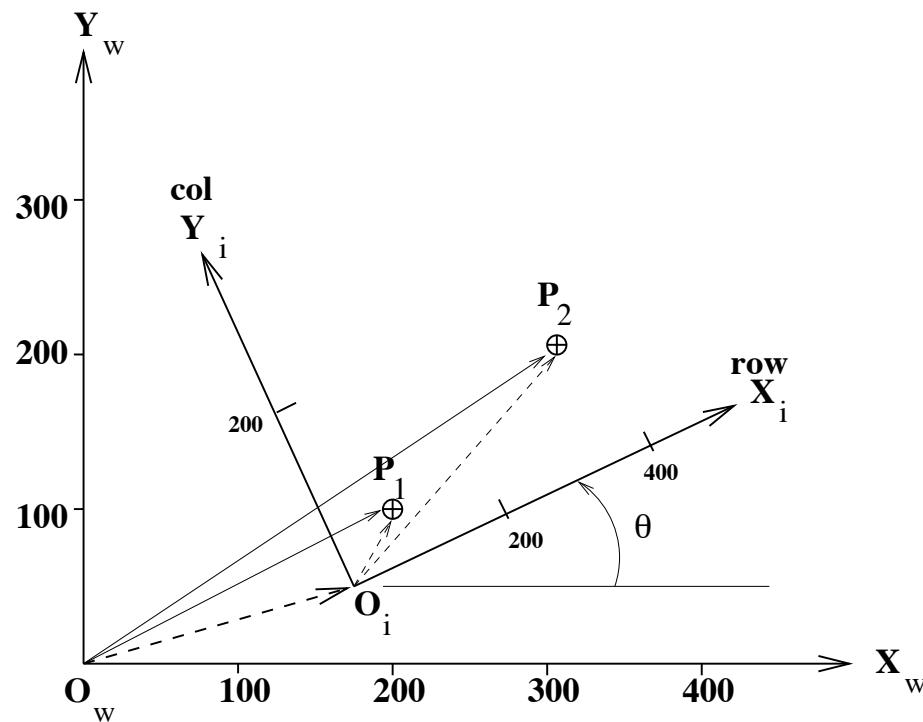


- Image $I(r,c)$
- Planar surface $W(x,y)$
- Need to map: real world \leftrightarrow image plane
- Combination of rotation R, scaling S, translation D
- 4 parameters:
 - Rotation angle
 - Scale factor
 - Translation vector

Rotation, scaling, and translation



- ${}^wP_j = D_{x0,y0}S_sR_\theta {}^iP_j$
- $w \rightarrow world, i \rightarrow image, j \rightarrow generic point$



Rotation, scaling, and translation



- To obtain the transformation matrix
 - Take two points, called control points
 - Control points must be clearly visible
 - Determine the position vector in the world and in the image plane
- Now solve the system...

$$\begin{bmatrix} x_w \\ y_w \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\vartheta & -\sin\vartheta & 0 \\ \sin\vartheta & \cos\vartheta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

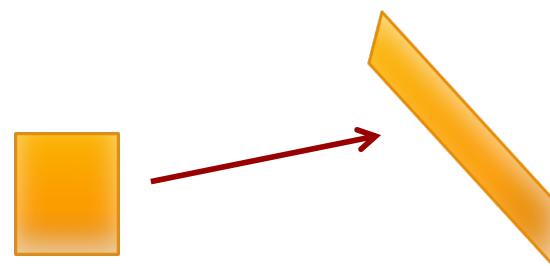
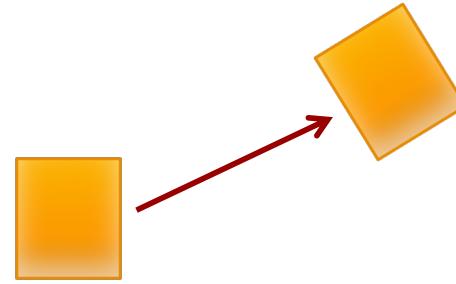
For P_1
Then the same for P_2

Transformations



- Rigid (easy)
 - Distance between points after transformation is the same as before the transformation
 - Composition of rotation and translation
 - Scaling does not fit this model...

- Deformable (more difficult)
 - The properties above do not hold anymore
 - Objects deform and the geometrical relationships between points are not known a priori



General affine transformation



- It handles parameters in a single matrix

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- 6 params determined using 3 matching pairs
- Errors may occur
- The higher the number of points, the smaller the error
- This is defined as the **CAMERA MATRIX**

General affine transformation



- Error (deviation) using least-squares method:

$$\varepsilon(a_{11}, a_{12}, a_{13}, a_{21}, a_{22}, a_{23}) = \sum_{j=1}^n ((a_{11}x_j + a_{12}y_j + a_{13} - u_j)^2 + (a_{21}x_j + a_{22}y_j + a_{23} - v_j)^2)$$

- To determine the matrix coefficients, take partial derivatives of the error function and set them to zero.
- The resulting equation system is:

$$\begin{bmatrix} \sum x_j^2 & \sum x_j y_j & \sum x_j & 0 & 0 & 0 \\ \sum x_j y_j & \sum y_j^2 & \sum y_j & 0 & 0 & 0 \\ \sum x_j & \sum y_j & \sum 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sum x_j^2 & \sum x_j y_j & \sum x_j \\ 0 & 0 & 0 & \sum x_j y_j & \sum y_j^2 & \sum y_j \\ 0 & 0 & 0 & \sum x_j & \sum y_j & \sum 1 \end{bmatrix} \begin{bmatrix} a_{11} \\ a_{12} \\ a_{13} \\ a_{21} \\ a_{22} \\ a_{23} \end{bmatrix} = \begin{bmatrix} \sum u_j x_j \\ \sum u_j y_j \\ \sum u_j \\ \sum v_j x_j \\ \sum v_j y_j \\ \sum v_j \end{bmatrix}$$

Least square method



- Developed to solve overdetermined equation systems
- Over-determined = more equations than unknowns
- Goal: minimize the sum of the squares of the errors
- **Error** is the distance between the observed value (u and v in our case) and the value obtained by applying the transformation (x and y after transformation)
- A simple case: $y = f(x) = Ax + B \rightarrow \varepsilon = \sum_{j=i}^N (Ax_j + B - y_j)^2$
- The solution is found by computing the minimum of the error = compute the partial derivative with respect to each unknown and set them to zero $\frac{\partial \varepsilon}{\partial A}, \frac{\partial \varepsilon}{\partial B}$

Going 3D



- One view in some cases is not sufficient
 - Depth of a point cannot be perceived
- Need to acquire images from different perspectives for:
 - 3D mesh reconstruction
 - Point Cloud acquisition
 - Position estimation
 - Structure from motion
 - Mosaicking
 - Etc...

Going 3D

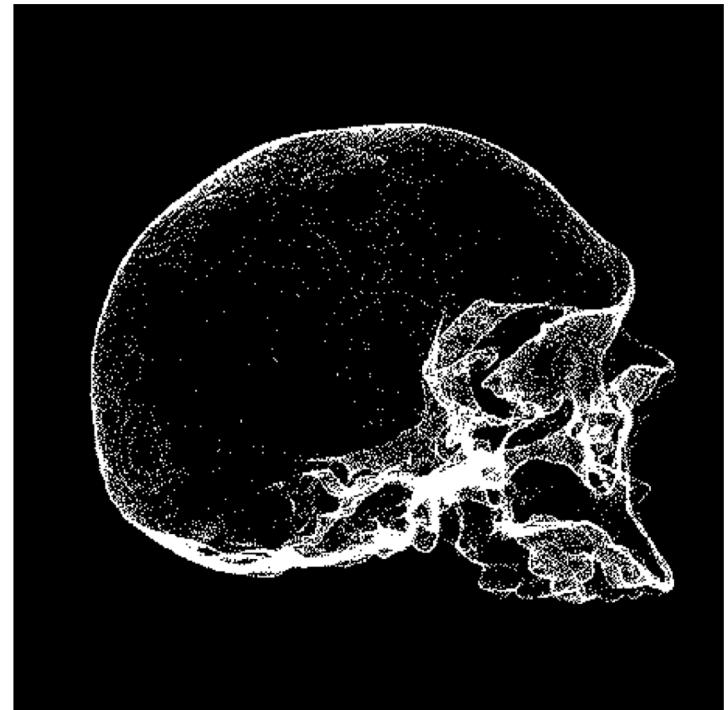
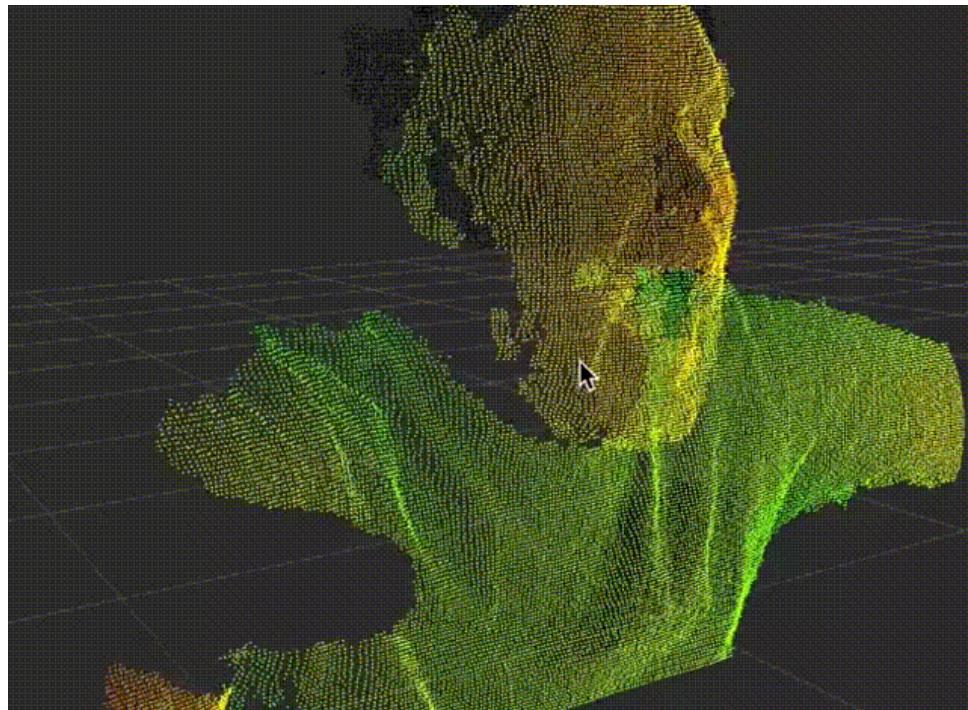


- 3D analysis handles the structure of objects and their real motion in space
 - More robust than 2D estimation
- 3D computation requires **at least two 2D projections**
- **Calibration** is needed
 - **Intrinsic** parameters
 - **Extrinsic** parameters

Examples



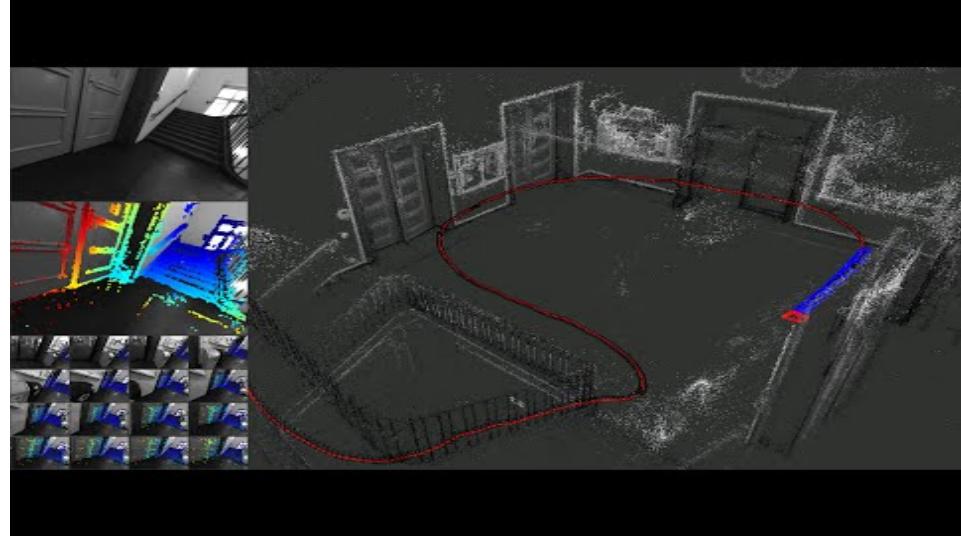
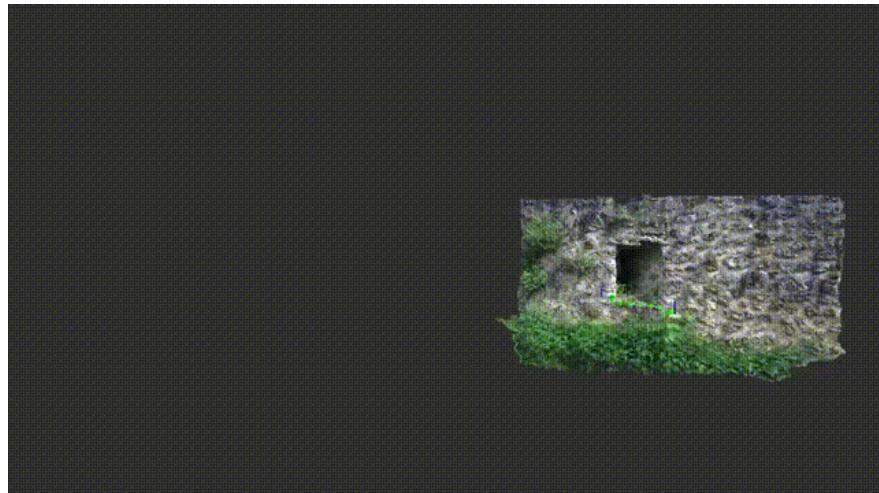
Point Clouds



Examples



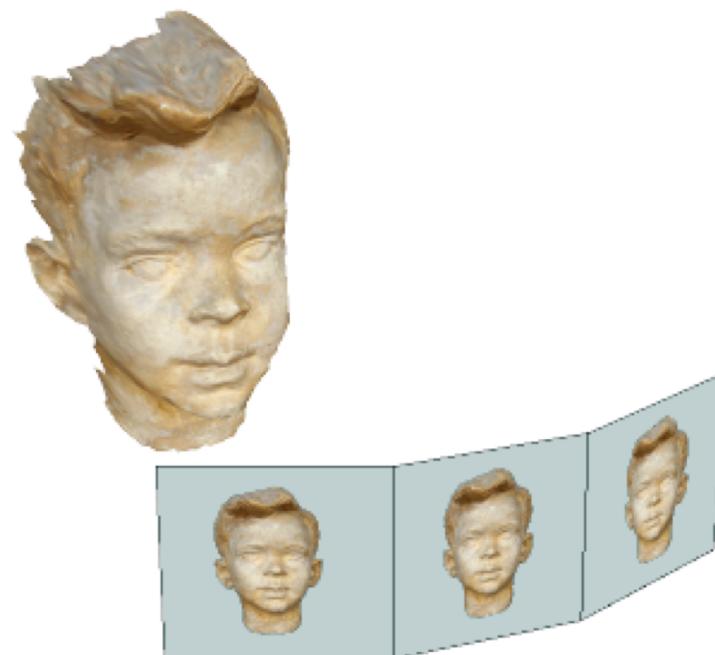
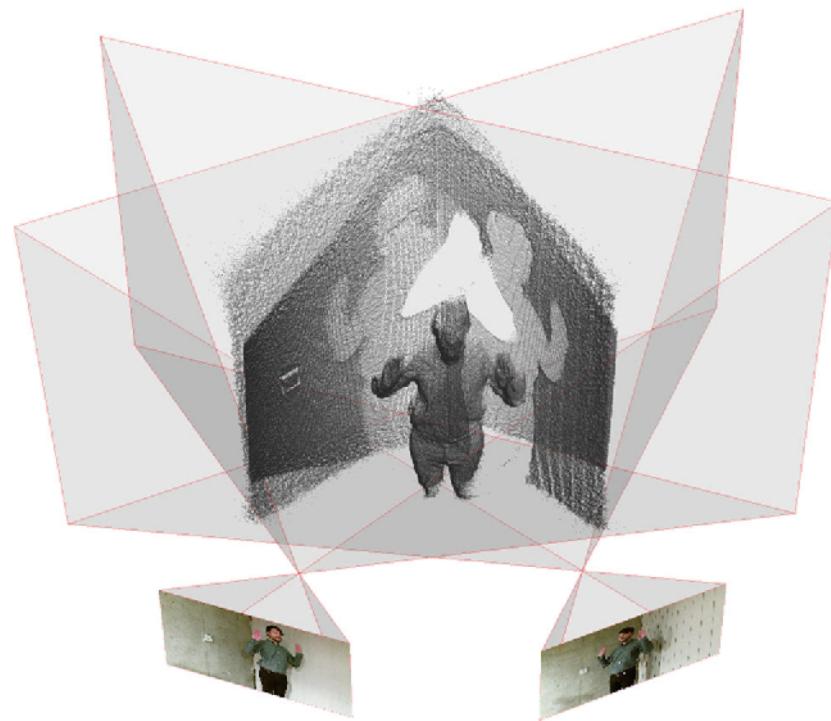
Structure From Motion



Examples



Mesh Reconstruction



Intrinsic and extrinsic parameters



- Intrinsic parameters
 - Specific for the sensor
 - Link pixel coordinates with the corresponding coordinates in the camera reference system
 - Optical, geometric, digital features
 - Focal length
 - Image distortion
 - Size of the pixel
- Extrinsic parameters:
 - Position of the camera in the world coordinates
 - Translation vector
 - Rotation matrix

3D affine transformations

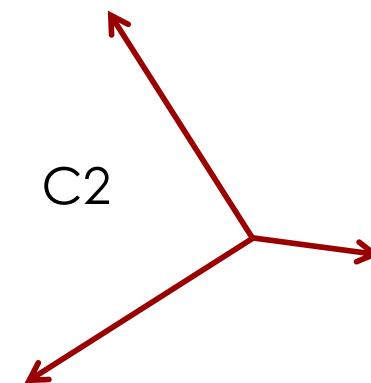
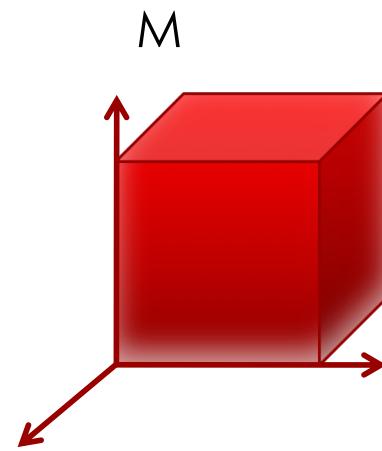
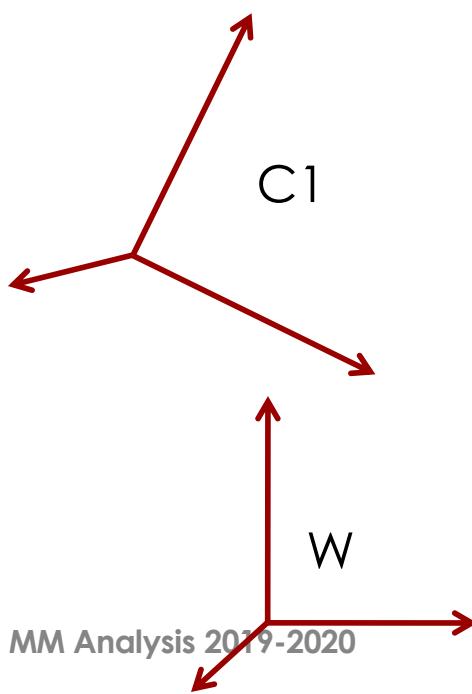


- Extension to the 3D of what we have seen for 2D
- Rotation, translation, scaling
- In homogeneous coordinates, go from
 - $[P_x, P_y, P_z]$ to $[sP_x, sP_y, sP_z, s]$
- Each point has in general 4 coordinate systems
 - Model
 - World
 - C1
 - C2

3D affine transformations



- $wP = \mathbf{T}\mathbf{R}^M P$, same for 1P and 2P
- ${}^1P \neq {}^2P$
- C1 and C2 can have opposite views, thus there is in general no one-to-one mapping of all points



Translation



$${}^2P = T(x_0, y_0, z_0) {}^1P$$

$$\begin{bmatrix} {}^2P_x \\ {}^2P_y \\ {}^2P_z \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & x_0 \\ 0 & 1 & 0 & y_0 \\ 0 & 0 & 1 & z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^1P_x \\ {}^1P_y \\ {}^1P_z \\ 1 \end{bmatrix}$$

- Add a translation vector to map point 1P in coordinate of C1 to 2P in coordinate C2

Scaling



$$^2P = S(s_x, s_y, s_z) ^1P$$

$$\begin{bmatrix} {}^2P_x \\ {}^2P_y \\ {}^2P_z \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^1P_x \\ {}^1P_y \\ {}^1P_z \\ 1 \end{bmatrix}$$

- Scaling can be applied differently on the image coordinates

Rotation



- Rotation x , y , or z axis
- In case of rotation about z , it is the same as in 2D

$${}^2P = R(X, \vartheta) {}^1P$$

$$\begin{bmatrix} {}^2P_x \\ {}^2P_y \\ {}^2P_z \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\vartheta & -\sin\vartheta & 0 \\ 0 & \sin\vartheta & \cos\vartheta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^1P_x \\ {}^1P_y \\ {}^1P_z \\ 1 \end{bmatrix}$$

$${}^2P = R(Y, \vartheta) {}^1P$$

$$\begin{bmatrix} {}^2P_x \\ {}^2P_y \\ {}^2P_z \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\vartheta & 0 & \sin\vartheta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\vartheta & 0 & \cos\vartheta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^1P_x \\ {}^1P_y \\ {}^1P_z \\ 1 \end{bmatrix}$$

$${}^2P = R(Z, \vartheta) {}^1P$$

$$\begin{bmatrix} {}^2P_x \\ {}^2P_y \\ {}^2P_z \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\vartheta & \sin\vartheta & 0 & 0 \\ -\sin\vartheta & \cos\vartheta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^1P_x \\ {}^1P_y \\ {}^1P_z \\ 1 \end{bmatrix}$$

General configuration



- The generic 3D affine transformation can be expressed through

$$\begin{bmatrix} {}^2P_x \\ {}^2P_y \\ {}^2P_z \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^1P_x \\ {}^1P_y \\ {}^1P_z \\ 1 \end{bmatrix}$$

Camera model



- It consists of the transformation that maps the 3D point into the image plane, so:

$${}^I P = {}_W^I C {}^W P$$

- According to our knowledge:

$$\begin{bmatrix} s^I P_r \\ s^I P_c \\ s \end{bmatrix} = {}_W^I C \begin{bmatrix} {}^W P_x \\ {}^W P_y \\ {}^W P_z \\ 1 \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ c_{31} & c_{32} & c_{33} & 1 \end{bmatrix} \begin{bmatrix} {}^W P_x \\ {}^W P_y \\ {}^W P_z \\ 1 \end{bmatrix}$$

- Using the 3x4 camera matrix we can handle rotation, translation and scaling

Nice, but



- In general camera coordinates differ from world coordinates
- Need to apply a roto-translation transformation to go from ${}^W P$ to ${}^C P$

$$\begin{bmatrix} {}^C P_x \\ {}^C P_y \\ {}^C P_z \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^W P_x \\ {}^W P_y \\ {}^W P_z \\ 1 \end{bmatrix}$$

$${}^C P = {}_W^C TR(\alpha, \beta, \gamma, t_x, t_y, t_z) {}^W P$$

- Furthermore, ${}^C P$ is about the camera coordinates, and not the image coordinates ${}^F P$

From world to image coordinates



- From world to camera, from camera to image

$${}^F P = {}_C^F \Pi(f) {}^C P$$

$${}^F P = {}_C^F \Pi(f) {}_W^C TR(\alpha, \beta, \gamma, t_x, t_y, t_z) {}^W P$$

$$\begin{bmatrix} s^F P_r \\ s^F P_c \\ s \end{bmatrix} = \begin{bmatrix} d_{11} & d_{12} & d_{13} & d_{14} \\ d_{21} & d_{22} & d_{23} & d_{24} \\ d_{31} & d_{32} & d_{33} & 1 \end{bmatrix} \begin{bmatrix} {}^W P_x \\ {}^W P_y \\ {}^W P_z \\ 1 \end{bmatrix}$$

From mm to pixels



- It consists of a scaling factor that is related to the real size of the pixel
 - d_x is the horizontal size
 - d_y is the vertical size
- Not only that, origin is usually bottom-left
- In the image, origin is top-left

$$\begin{aligned} {}^I P &= {}^I_F S^F P \\ {}^I_F S &= \begin{bmatrix} 0 & -1/d_y & 0 \\ 1/d_x & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{aligned}$$

In the end



- The final expression becomes then

$$[p_r, p_c]^T = {}^I P = {}_F^I S_C^F \Pi(f) {}_W^C T R(\alpha, \beta, \gamma, t_x, t_y, t_z) {}^W P$$

$$\begin{bmatrix} {}^I s P_r \\ {}^I s P_c \\ s \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ c_{31} & c_{32} & c_{33} & 1 \end{bmatrix} \begin{bmatrix} {}^W P_x \\ {}^W P_y \\ {}^W P_z \\ 1 \end{bmatrix}$$

Calibration



- To make a real measurement in 3D
 - The camera matrix we derive is appropriate but the coefficients of the matrix have to be determined
 - 11 parameters using least squares
1. Typically an object of known size is used
 2. A set of points in the image/world are taken
 3. 6 pairs are ok, but a higher number is preferable

Calibration



- From the camera matrix
 - Given a 3D point $[{}^W P_x, {}^W P_y, {}^W P_z] = [x, y, z]$
 - Given the 2D projection $[{}^I P_r, {}^I P_c] = [u, v]$
 - For each point in the calibration process we have:

$$\begin{bmatrix} x_j & y_j & z_j & 1 & 0 & 0 & 0 & -x_j u_j & -y_j u_j & -z_j u_j \\ 0 & 0 & 0 & 0 & x_j & y_j & z_j & 1 & -x_j v_j & -y_j v_j & -z_j v_j \end{bmatrix} \begin{bmatrix} c_{11} \\ c_{12} \\ c_{13} \\ c_{14} \\ c_{21} \\ c_{22} \\ c_{23} \\ c_{24} \\ c_{31} \\ c_{32} \\ c_{33} \end{bmatrix} = \begin{bmatrix} u_j \\ v_j \end{bmatrix}$$

Calibration



- Overdetermined system: 11 unknowns and 12 or more equations
- No vector can satisfy all the equations
- Need to use a least squares approach
- Objective: minimize the sum of all the square differences

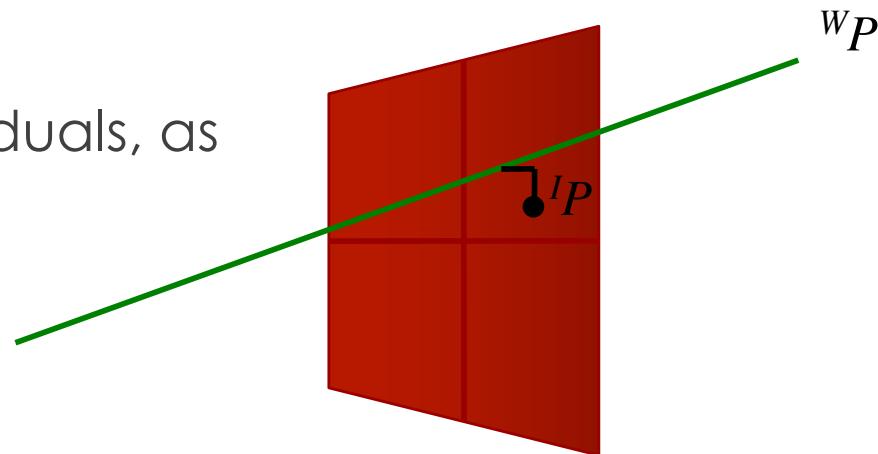
Calibration



- The error between the actual image point measurements and the world points comes from

$${}^I P = {}_W^I C {}^W P$$

- Need to minimize the residuals, as in the example:



Calibration



- The least squares problem tries to find x that minimizes r :

$$Ax = b$$

$$r = b - Ax$$

- It can be demonstrated that the residual is orthogonal to A , therefore:

$$A^t r = 0$$

$$A^t A x = A^t b$$

- $A^t A$ is symmetric and positive definite, so there exists an inverse that solves:

$$x = (A^t A)^{-1} A^t b$$

- Once x is calculated, also r can be computed

Compute 3D position of a point



- If the cameras are calibrated we can determine the 3D position of a generic point $[x, y, z]$ given two projections $[r_1, c_1]$ & $[r_2, c_2]$

$$\begin{bmatrix} sr_1 \\ sc_1 \\ s \end{bmatrix} = \begin{bmatrix} b_{11} & b_{12} & b_{13} & b_{14} \\ b_{21} & b_{22} & b_{23} & b_{24} \\ b_{31} & b_{32} & b_{33} & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} tr_2 \\ tc_2 \\ t \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ c_{31} & c_{32} & c_{33} & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Compute 3D position of a point



- If we drop the scaling factors we obtain four equations to compute r_1 , r_2 , c_1 , c_2

$$r_1 = (b_{11} - b_{31}r_1)x + (b_{12} - b_{32}r_1)y + (b_{13} - b_{33}r_1)z + b_{14}$$

$$c_1 = (b_{21} - b_{31}c_1)x + (b_{22} - b_{32}c_1)y + (b_{23} - b_{33}c_1)z + b_{24}$$

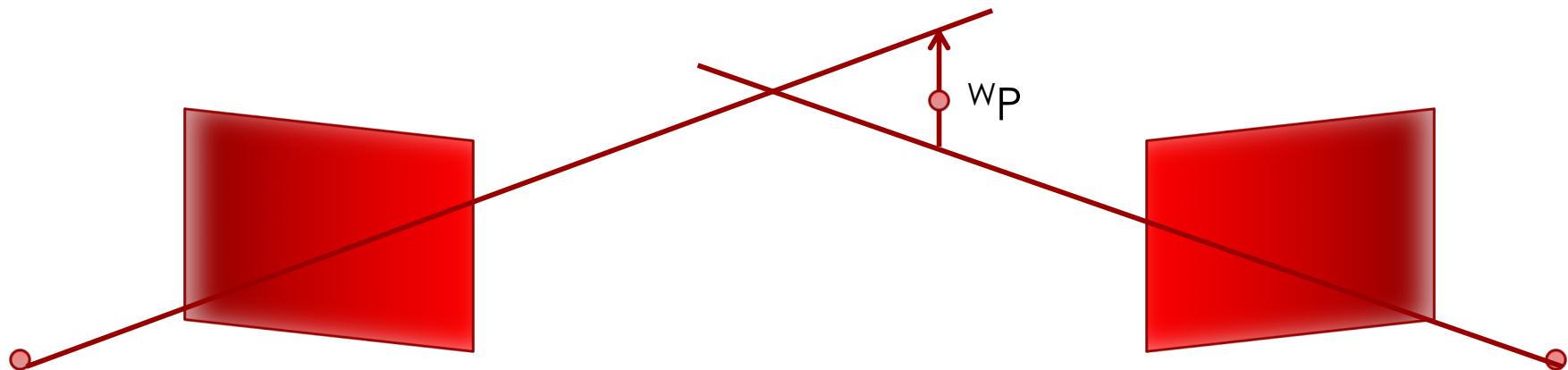
$$r_2 = (c_{11} - c_{31}r_2)x + (c_{12} - c_{32}r_2)y + (c_{13} - c_{33}r_2)z + c_{14}$$

$$c_2 = (c_{21} - c_{31}c_2)x + (c_{22} - c_{32}c_2)y + (c_{23} - c_{33}r_2)z + c_{24}$$

- We have three unknowns
- From any three of these four eq. we can find a solution
- The solution is subject to errors though

Approximation errors

- If we took the four equations all together they would be inconsistent
 - Approximation in image points
 - Approximation in camera model
 - In practice the rays do not intersect where they should in the real world



Binocular stereo

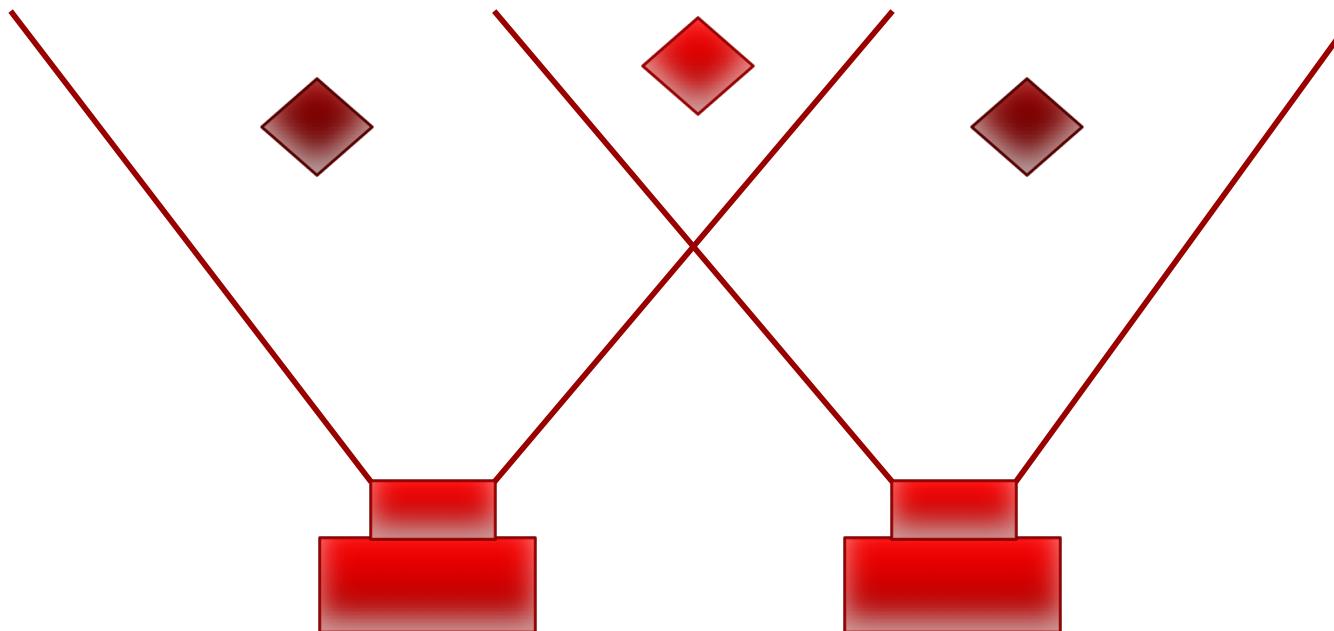


- Two images grab a scene from different positions
- Human Visual System is stereo
- Identification of two main problems
 - Compute correspondences
 - Reconstruction

Two co-planar cameras



- Only for the points “seen” by both cameras, the depth information can be extracted



Two co-planar cameras



- Offset in the view, due to the camera shift
- Same point is projected in a different place
- Depending on the depth of the point, shift is different
- The closer the object, the higher the shift
- Parallax: Apparent motion → closer objects move faster



Compute correspondences



- Find points in the images that correspond to the projection of the same point in the real scene
- Image coupling is possible because the two images do not differ much between each other.
- False correspondences may be present → introduction of other constraints
- **Epipolar constraint:**
 - Correspondences can be met along a horizontal line called “epipolar line”

Reconstruction

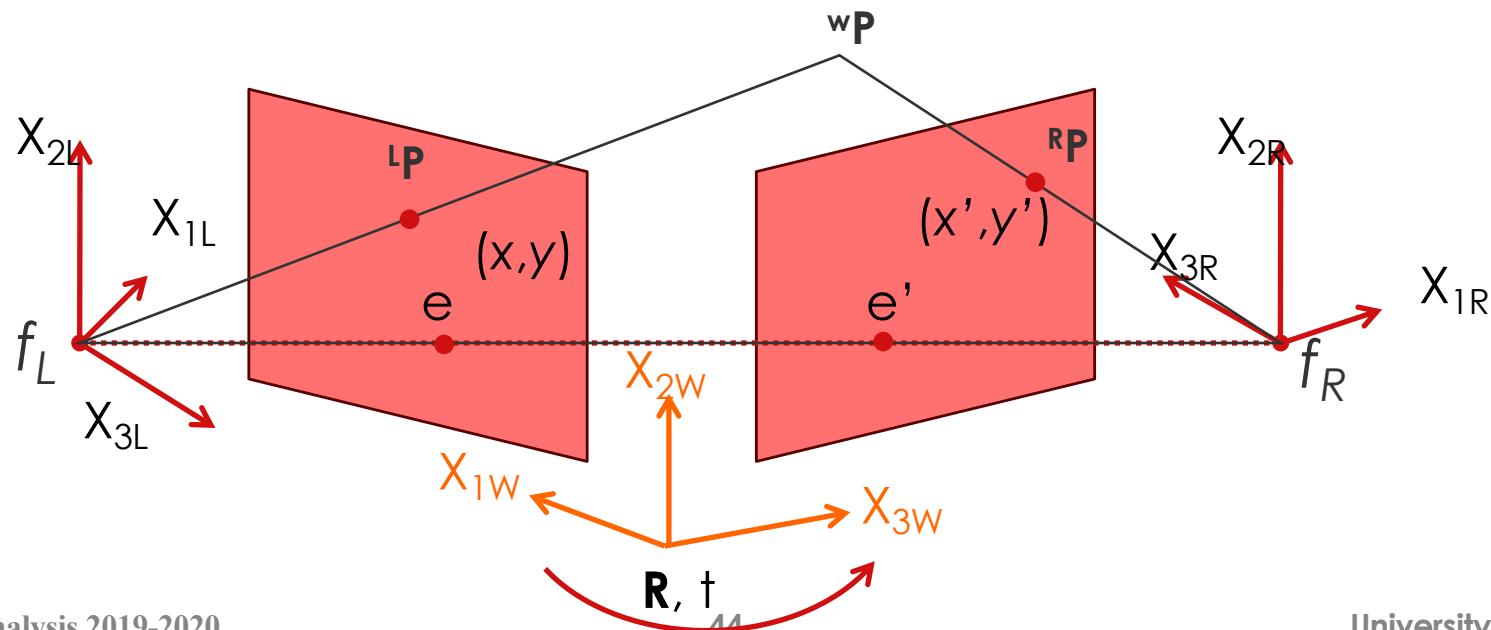


- Once the points in the two images are matched, reconstruction is possible.
- Before computing correspondences it is necessary to **calibrate** the cameras to determine:
 - Extrinsic parameters
 - Intrinsic parameters

Stereo vision



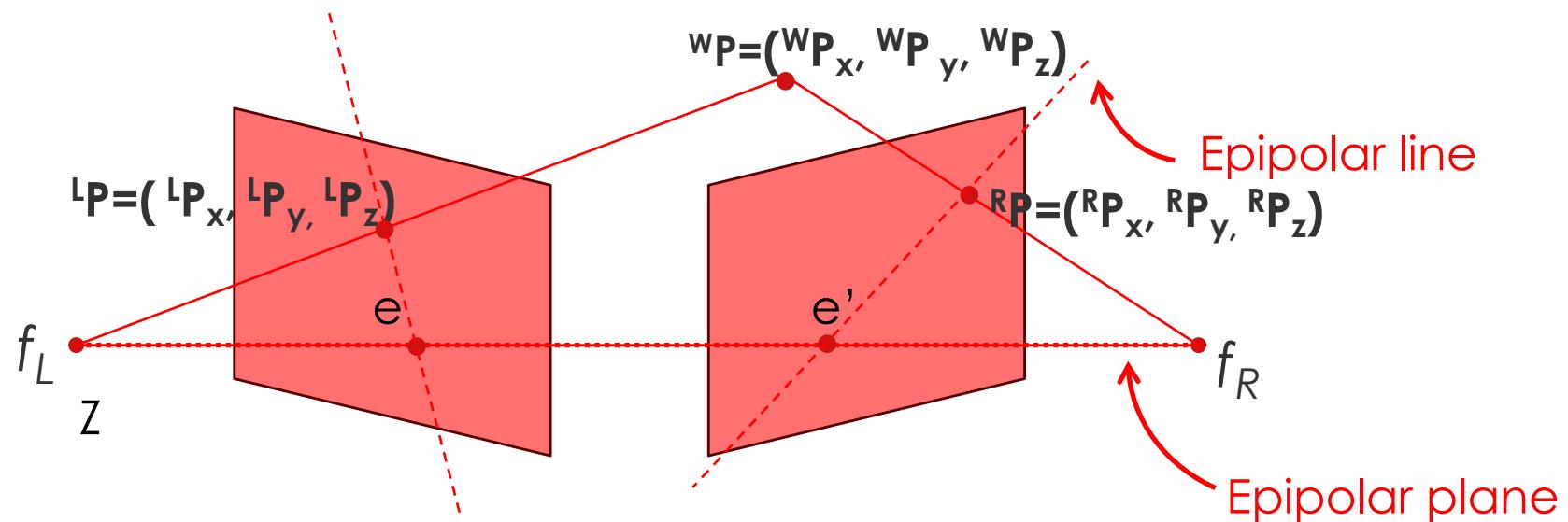
- Based on the so-called epipolar geometry
- Estimation of the depth of a point using the horizontal disparity of the projections
- The problem consists in computing:
 - Correspondences of intensity points (correlation)
 - Correspondences of features (edges, contours)



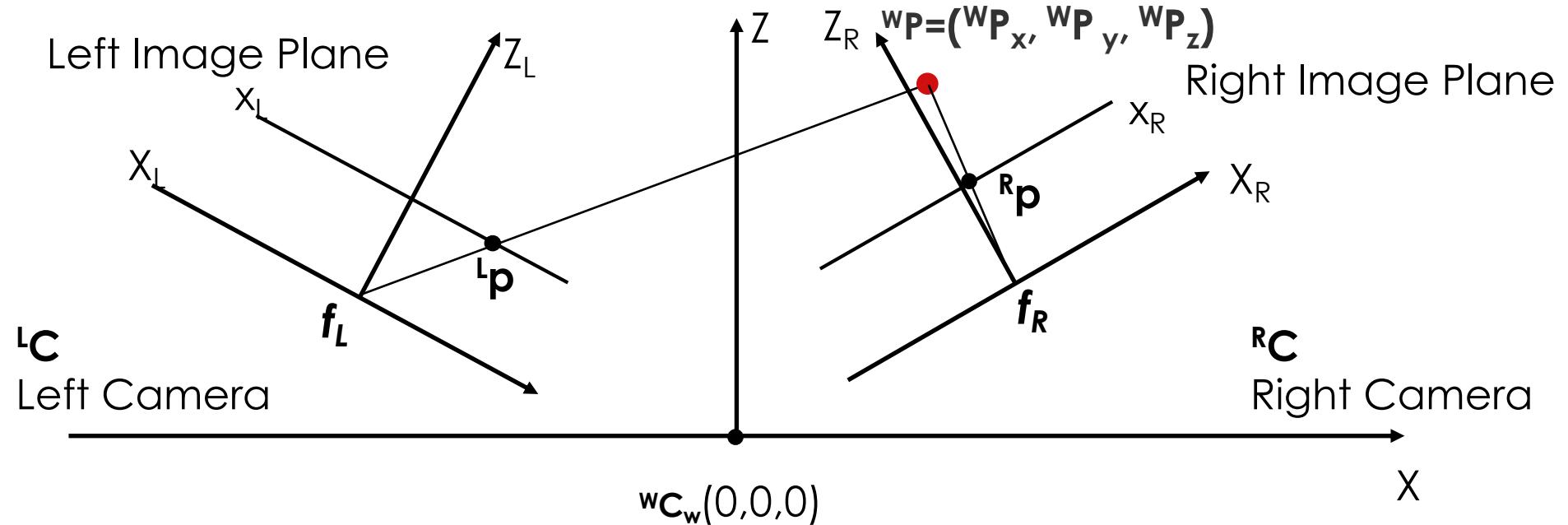
Epipolar geometry



- Epipolar plane given by: P , f_L and f_R
- The intersection between the epipolar plane with the image planes defines the **epipolar lines**
- All epipolar lines pass through a point called **epipole**.
- The projection of each point in the epipolar plane falls onto the epipolar line.
- Epipolar planes are a set of planes that share the line (baseline) connecting f_L and f_R .



Reconstruction from two static images



- Hypothesis: $f_L = f_R = f$
- We obtain:

$${}^R \mathbf{P} = ({}^R P_x, {}^R P_y, {}^R P_z), \quad {}^L \mathbf{P} = ({}^L P_x, {}^L P_y, {}^L P_z), \quad {}^W \mathbf{P} = ({}^W P_x, {}^W P_y, {}^W P_z)$$

$${}^R \mathbf{P} = {}^R \mathbf{R} {}^W \mathbf{P} + {}^R \mathbf{T}, \quad {}^L \mathbf{P} = {}^L \mathbf{R} {}^W \mathbf{P} + {}^L \mathbf{T}$$
- where ${}^R \mathbf{R}$, ${}^R \mathbf{T}$, ${}^L \mathbf{R}$, and ${}^L \mathbf{T}$ are the extrinsic parameters of the cameras that indicate the positioning of ${}^R \mathbf{C}$ and ${}^L \mathbf{C}$ with respect to ${}^W \mathbf{C}$



Reconstruction from two static images

- From the previous eq:

$$\begin{aligned} {}^L \mathbf{P} &= {}^L \mathbf{R} {}^R \mathbf{R}^{-1} {}^R \mathbf{P} - {}^L \mathbf{R} {}^R \mathbf{R}^{-1} {}^R \mathbf{T} + {}^L \mathbf{T} \\ &= \mathbf{M} {}^R \mathbf{P} + \mathbf{B} \end{aligned}$$

- Using the simplified perspective projections ($Z \gg f$) we obtain the coordinates on the two image planes

$${}^L p_x = \frac{f {}^L P_x}{{}^L P_z}, \quad {}^L p_y = \frac{f {}^L P_y}{{}^L P_z} \quad (1)$$

$${}^R p_x = \frac{f {}^R P_x}{{}^R P_z}, \quad {}^R p_y = \frac{f {}^R P_y}{{}^R P_z}$$

- From which we obtain:
$$\frac{{}^L P_z}{f} \begin{bmatrix} {}^L p_x \\ {}^L p_y \\ f \end{bmatrix} = \frac{{}^R P_z}{f} \mathbf{M} \begin{bmatrix} {}^R p_x \\ {}^R p_y \\ f \end{bmatrix} + \mathbf{B} \quad (2)$$



Estimation of the 3D position

- **Stereo Matching Problem :**

Estimate the coordinates of ${}^W P$, given $({}^L p_x, {}^L p_y)$ and $({}^R p_x, {}^R p_y)$, and the extrinsic parameters of the cameras.

- **Algorithm:**

1. From (2) determine the depth of ${}^W P$ with respect to ${}^R P_z$ e ${}^L P_z$
2. From (1) calculate the other coordinates for ${}^L P$ e ${}^R P$
3. Ordinary Least Squares to estimate ${}^W P$ in ${}^W C$

- If ${}^L C$ and ${}^R C$ are parallel and aligned:

$${}^W P_z = \frac{fb}{{}^L p_x - {}^R p_x}$$

Where b is the distance between the two cameras

From points to images



- ${}^L p_x - {}^R p_x$ is called disparity
- The disparity image is the difference between the left and right image
- To compute the disparity image, need to find a match pixel by pixel
- Issues:
 - Regions can be occluded
 - Regions can be disoccluded
 - Uniform areas do not allow finding correspondences

From points to images



- To find a match, an error function must be evaluated
 - Intensity difference
 - Evaluate edges, windows, segmented known areas

Window-based analysis



- Take a window in the left (right) image
- Along the epipolar line find the window that best matches in the right (left) image
- Compute an error function (MSE, SAD, SSD)
- Find the minimum
- Winner-take-all → that's the disparity!

Winner-take-all with SAD



- Moving the window of analysis, compute:

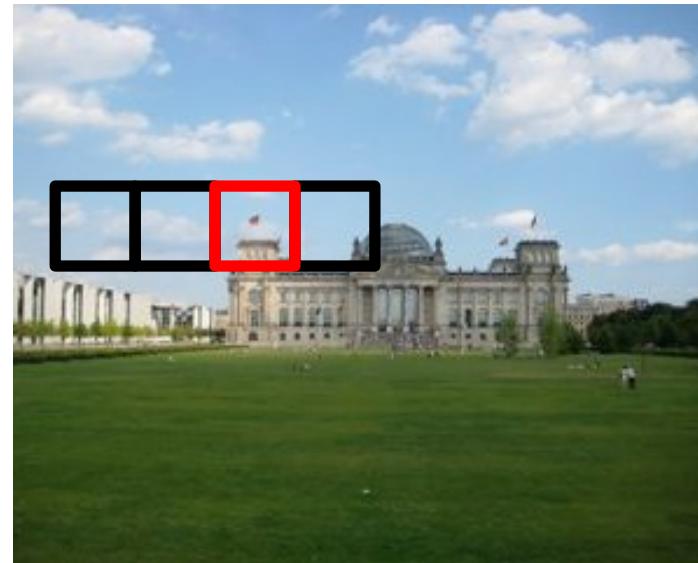
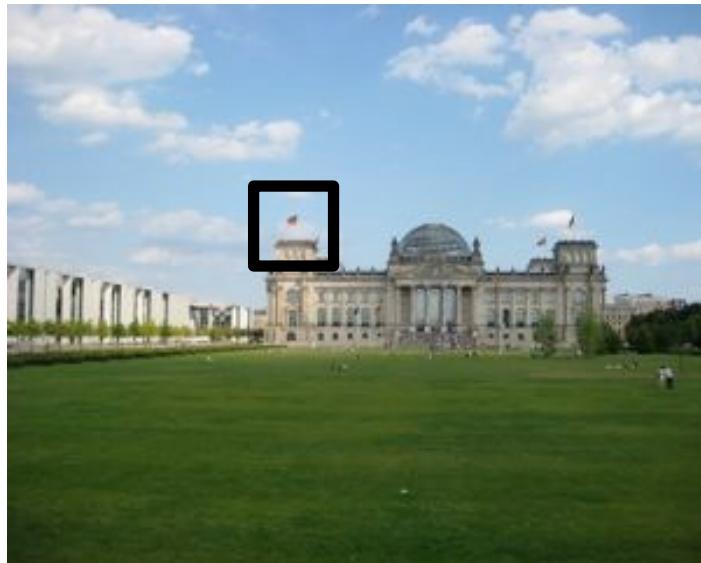


Image Normalization



- Snapshots may be taken under different conditions
- Reflectance of surfaces could be not ideal (not Lambertian)
- It is preferable to work with normalized values
- A window-based analysis helps preventing acquisition issues

$$\bar{I} = \frac{1}{|W(x,y)|} \sum_{(u,v) \in W(x,y)} I(u,v)$$

Compute the average pixel in the window

$$\|I\|_{W(x,y)} = \sqrt{\sum_{(u,v) \in W(x,y)} [I(u,v)]^2}$$

Compute the window magnitude

$$\hat{I}(x,y) = \frac{I(x,y) - \bar{I}}{\|I - \bar{I}\|_{W(x,y)}}$$

Calculate the normalized value

Normalization and SSD



- Normalization makes windows comparable
- Now comparison can be computed, using for example, SAD or the Sum of Squared Differences (SSD)

$$SSD(x, y, d) = \sum_{(u,v) \in W(x,y)} [\hat{I}_L(u, v) - \hat{I}_R(u - d, v)]^2$$

Correlation



- Compare the intensity in a neighborhood
- Take a window around point (u,v)
 - Arrange the values in a vector $w \in \Re^p$ by scanning the window in the first image row by row
 - Same for w' in the second image
 - Evaluate the correlation using

$$C(d) = \frac{1}{|w - \bar{w}|} \frac{1}{|w' - \bar{w}'|} (w - \bar{w})(w' - \bar{w}')$$

- Maximizing the correlation means minimizing the SSD

Correlation



- If the window is 3x3, p=9
- It's a comparison between vectors, and the correlation measure is the angle between

$$w - \bar{w} \quad \text{and} \quad w' - \bar{w}'$$

$$C(d) = \sum \hat{I}(u,v)\hat{I}'(u-d,v) = w \cdot w' = \cos \theta$$

- In the normalized case, the correlation is maximum if the original brightness of the two windows is shifted by an offset and a scale factor

$$I' = \lambda I + \mu$$

Comments



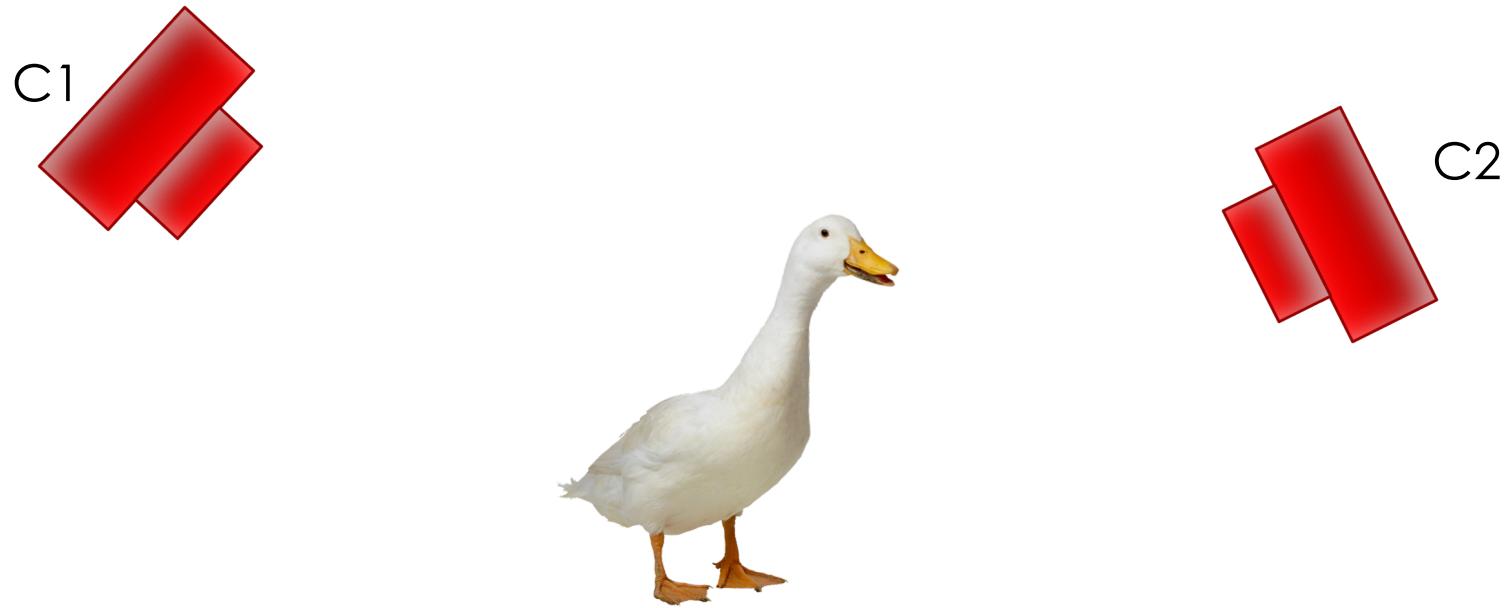
- Computing the correlation at each frame for the whole image can be expensive
- Luckily window shifts overlap
- Keep the information about windows in memory
- Usually the computation is carried out considering a range of the disparity
- In computing the correlation, it is assumed that the scene is parallel to the image plane

General stereo configuration



- Stereo rig we have seen:
 - Two cameras
 - Aligned
 - Parallel views
- In general
 - Two cameras with arbitrary positioning in 3D space

General stereo configuration



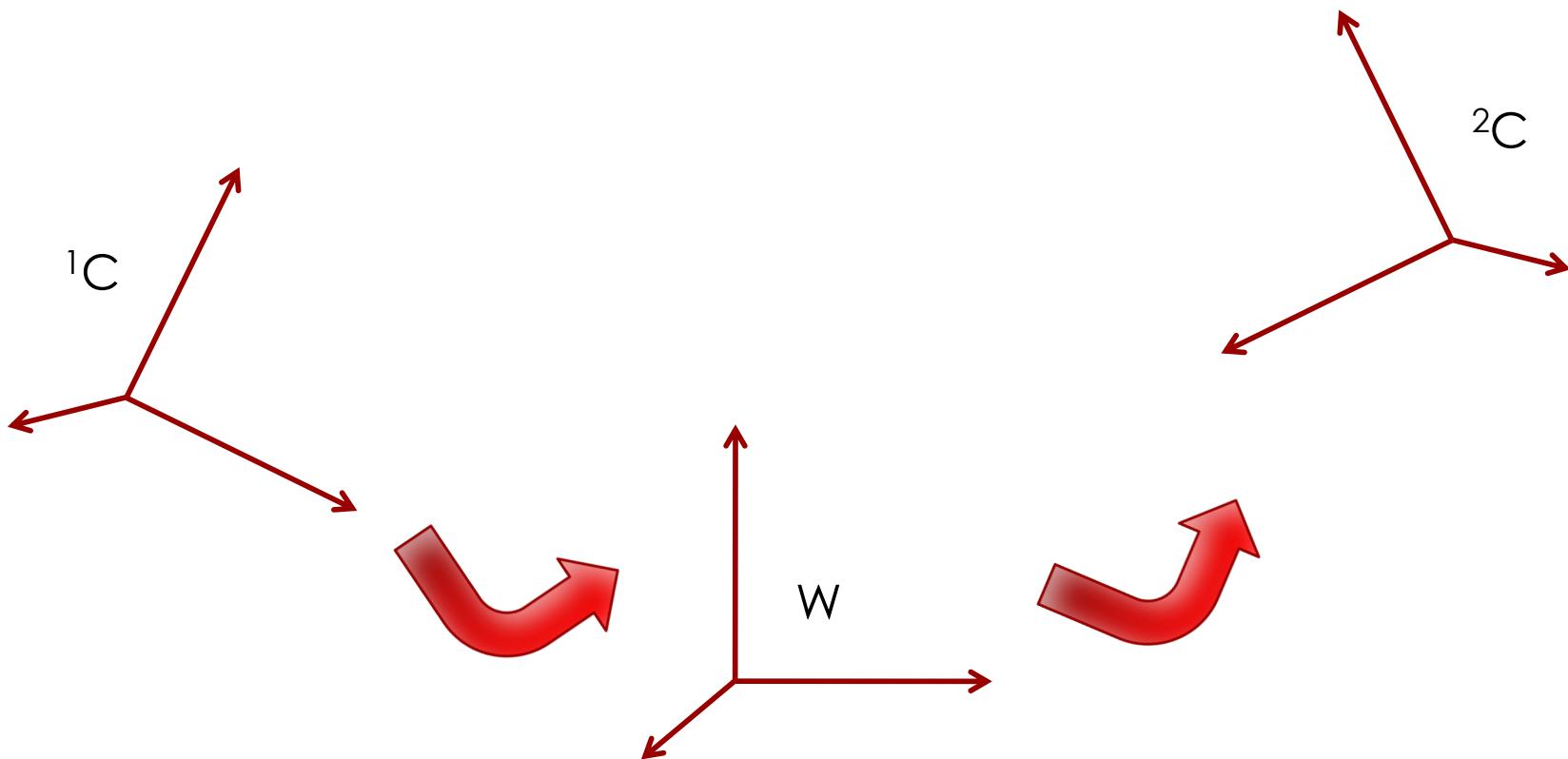
- The object “lives” in the world coordinates
 - ${}^wP = [{}^wP_x, {}^wP_y, {}^wP_z]^T$
- As in the previous case we have to match ${}^1P \leftrightarrow {}^2P$
- Intersection of ${}^wP^1O$ and ${}^wP^2O$

What do we need?



- Position of C1 and some internal parameters such as the focal length
 - This information is embedded in the *camera matrix*
 - The camera matrix defines a ray for each point wP mapped on 1P
- Same thing for C2
- Correspondences: ${}^1P \leftrightarrow {}^2P$ of wP
- Compute wP from 1P and 2P

The world is the shared information



Fundamental matrix



- Represents the epipolar geometry in case two generic views
- F is a 3×3 matrix that maps p into p'

$$p'^T F p = 0$$

- Independent from the scene structure
- Can be computed using correspondences
- No need to know the intrinsic parameters

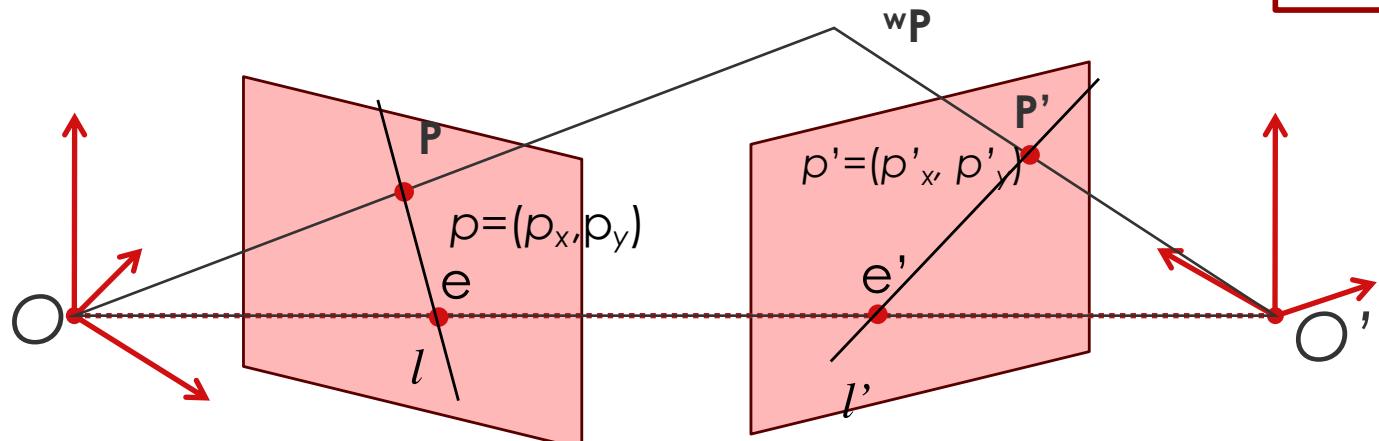
Fundamental matrix



- What does the fundamental matrix represent?
- Say we know the intrinsic parameters of the cameras
- The two cameras are shifted by a certain translation vector t and rotated by a matrix R

$$Op \cdot [OO' \times O'p'] = 0 \quad p \cdot [t \times Rp'] = 0 \quad p = (u, v, 1)^T \quad p' = (u', v', 1)^T \quad p^T [t_x R] p' = 0 \quad p^T E p' = 0$$

Essential Matrix



Fundamental matrix



- The Essential matrix assumes intrinsic parameters are known
- Intrinsic parameters can be modeled through a matrix as:

$$p = K\hat{p} \quad p' = K'\hat{p}' \quad \rightarrow F = K^{-T} E K'^{-1}$$

- We say p and p' correspond, being different projections of the same point
- So, for each point p in one view, there's a corresponding epipolar line l' in the other image
- p' lies on the epipolar line l'

Fundamental matrix



- F represents the fundamental matrix for the cameras 1C and 2C '
→ F^T is for $({}^2C, {}^1C)$
- Similarly:
 - $l' = Fp$
 - $l = F^Tp'$



The camera matrix

- Starting from the fundamental matrix it is possible to derive the camera matrices (1M and 2M) of the two views
- Conversely to F , camera matrices relate 3D space coordinates with the image
 - Depend on image coordinates
 - Depend on world coordinates
- A change of the world coordinates affects 1M and 2M , not F
- This tells us:
 - 1M and 2M determine a unique F
 - F determines 1M and 2M up to a multiplication by a certain matrix H

Ambiguity



- Given F , for an object, impossible to determine
 - The absolute position (in the world)
 - Orientation (NSWE)
 - Scale
- However, up to a projective transformation, the ambiguity in reconstruction can be solved
- In fact the projected points don't change if:

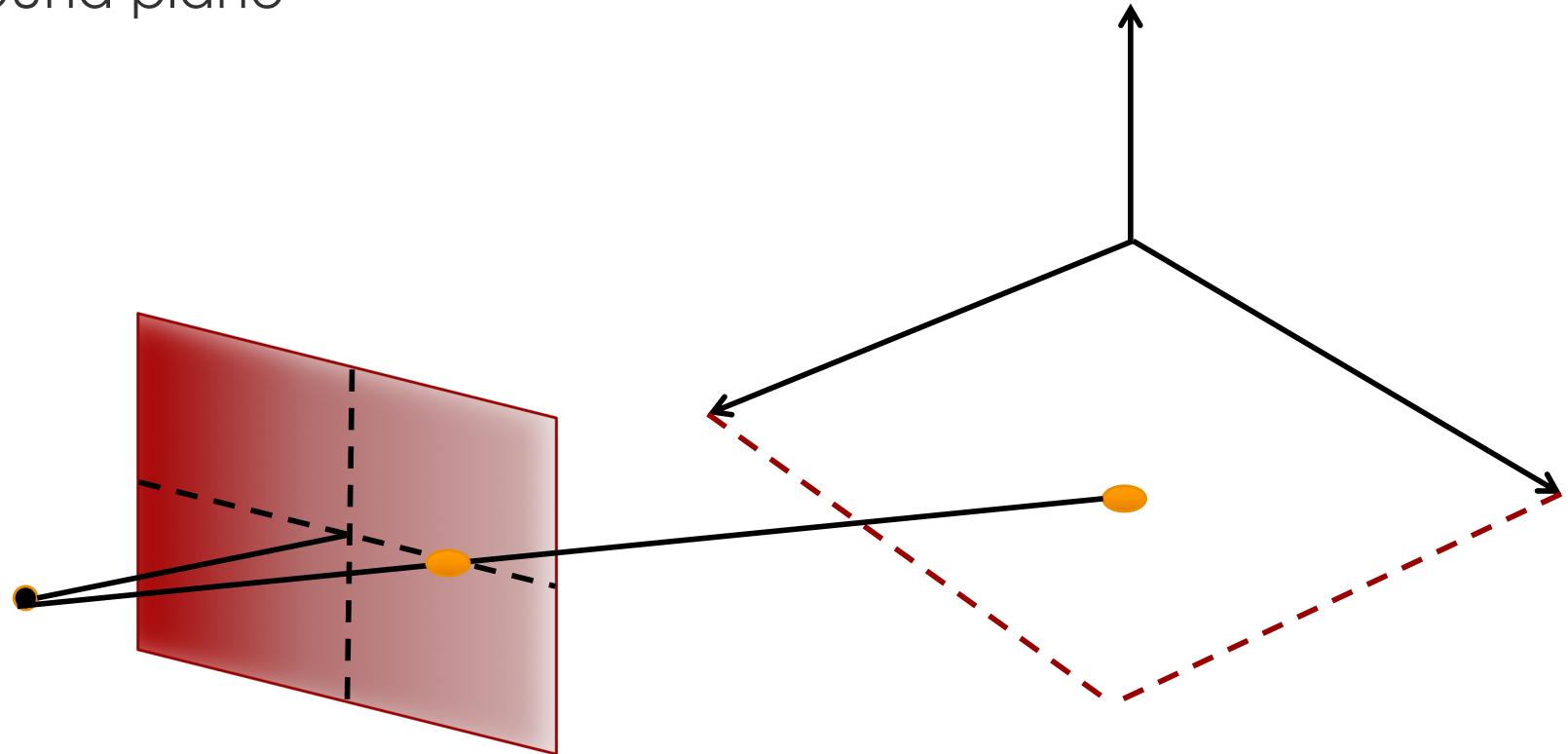
$$MP = (MH^{-1})(HP)$$

- H is a projective transformation, that does not affect the projection of wP onto the image plane

Localization on a Plane: 2D Homography



- In many applications, the mapping between the cameras can be limited to the computation of correspondences using a common 2D ground plane



2D Homography



- It's an invertible transformation between two planes
 - The image plane
 - The ground plane
- The homography matrix is defined as H that satisfies $p' = Hp$
 - Where p is the point in the world ground plane and p' is the point in the image plane
 - Since any vector crossed with itself gives 0, we can rewrite it as:

$$p' \times (Hp) = 0$$

See Hartley – Zissermann, p.90

Computation of the Matrix



- Given $p' = (x', y', w')^T$ in homogeneous coordinates the cross product becomes

$$p_i \times H p_i = \begin{pmatrix} y_i \bar{h}^{3T} p_i - w_i \bar{h}^{2T} p_i \\ w_i \bar{h}^{1T} p_i - x_i \bar{h}^{3T} p_i \\ x_i \bar{h}^{2T} p_i - y_i \bar{h}^{1T} p_i \end{pmatrix}$$

- This can be rewritten (since $h_j^T p_i = p_i^T h_j$) as:

$$\begin{bmatrix} 0^T & -w_i p_i^T & y_i p_i^T \\ w_i p_i^T & 0^T & -x_i p_i^T \\ -y_i p_i^T & x_i p_i^T & 0^T \end{bmatrix} \begin{bmatrix} \bar{h}^1 \\ \bar{h}^2 \\ \bar{h}^3 \end{bmatrix} = 0$$

Computation of the matrix



- Only two of the equations are independent, so the third equation can be discarded since it results as the multiplication of (x' times the first row) + (y' times the second row) up to scale.

$$A_i h = 0$$

$$\begin{bmatrix} 0^T & -w_i' p_i^T & -y_i' p_i^T \\ w_i' p_i^T & 0^T & -x_i' p_i^T \end{bmatrix} \begin{bmatrix} \bar{h}^1 \\ \bar{h}^2 \\ \bar{h}^3 \end{bmatrix} = 0$$

- A_i is a 2×9 matrix
- h^i are the lines of matrix $H = \{h_{11}, \dots, h_{33}\}$
- For each point we have two equations. For simplicity we can assume $w' = 1$

Solving for H



- Each correspondence gives two independent equations
- We then need 4 corresponding points, in order to get $Ah=0$, where A is made up by four A_i contributed from each correspondence
- We want to obtain the coefficients of H, where A is 8×9
- H is determined up to scale
- Scale can be arbitrarily chosen inserting a requirement on the norm, such as $\text{norm}(h)=1$

Overdetermined system



- As in calibration, using more points can be useful.
- If $n > 4$, the system is overdetermined
- Due to measurement errors, the solution $Ah = 0$ is only satisfied by the trivial case of $h_{ij}=0$
- The non-trivial solution is to minimize the norm $\|A\bar{h}\|$
- Subject to the constraint $\|\bar{h}\|=1$
- This corresponds to finding the minimum of $\|A\bar{h}\|/\|h\|$
- The solution is found applying the DLT (Direct Linear Transformation)

The inhomogeneous solution



- The matrix can be solved also by imposing the condition $h_j=1$ for some entry of the matrix H , since the matrix is defined up to scale.
- By setting for example $h_{33}=1$ the resulting system becomes:

$$\begin{bmatrix} 0 & 0 & 0 & -x_i w_i & -y_i w_i & -w_i w_i & -x_i y_i & y_i y_i \\ x_i w_i & y_i w_i & w_i w_i & 0 & 0 & 0 & -x_i x_i & -y_i x_i \end{bmatrix} h = \begin{bmatrix} -w_i y_i \\ w_i x_i \end{bmatrix}$$

- In this case we obtain 2 equations per point in the form of $Mh=b$
- h has 8 unknowns, b is a 8-vector
- Standard solution for 4 points or least squares for more points
- Not recommended if h_j are close to zero

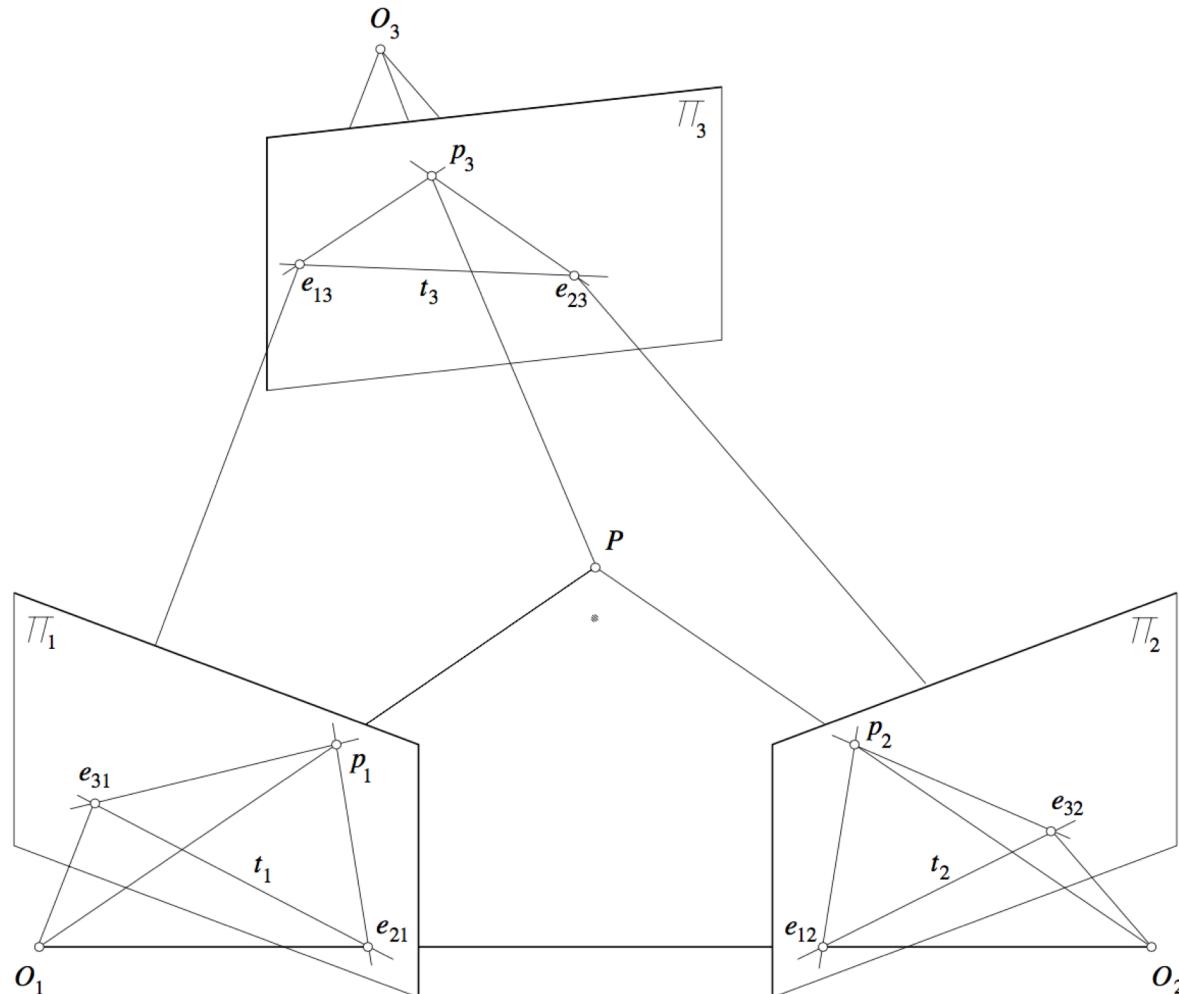
Multiple view geometry



- So far two examples:
 - Stereo
 - Generic two-view
- Extension to multiple views is possible (not trivial) for applications like 3D reconstruction
 - Need to acquire the object from multiple perspectives
 - Different views may not overlap significantly
 - Need to register the pictures
 - Instead of pixels, we talk about voxels

See Forsyth- Ponce, ch.11

What about three views?



What about three views?!



- Let us assume we know the internal parameters
- Having 3 cameras means having 1p , 2p , and 3p observing the same point P
- The 3 origins, 1O , 2O , and 3O intersect forming a trifocal plane that creates three trifocal lines when intersecting the image planes
- Each line passes through the epipoles determined by the two other cameras
- Each pair of cameras defines an epipolar constraint, as:

$$\left\{ \begin{array}{l} p_1^T E_{12} p_2 = 0 \\ p_2^T E_{23} p_3 = 0 \\ p_3^T E_{31} p_1 = 0 \end{array} \right.$$

What about three views?



- The three equations are not independent since

$$e_{31}^T E_{12} e_{32} = e_{12}^T E_{23} e_{13} = e_{23}^T E_{31} e_{21} = 0$$

- In fact, e_{31} and e_{32} are the first and second images of the optical center 3O of the third camera, and are therefore in epipolar correspondence
- Any of the two eq are independent, which means that it is possible to compute the position of 1p given the position of 2p and 3p , if the essential matrices are known.
- Using eq 1 and 3 from the previous slide, we have a system in the unknown coordinates of 1p

Problem of Transfer



- To understand the position of a point in W we only need two projections
- Knowing the essential/fundamental matrices it is possible to predict the position of a point in a picture given the other two (or more) projections