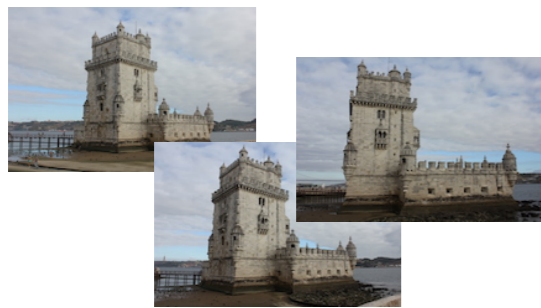


3D from Photographs: Camera Calibration

Francesco Banterle, Ph.D.

francesco.banterle@isti.cnr.it

3D from Photographs



Photographs



Automatic
Matching of
Images



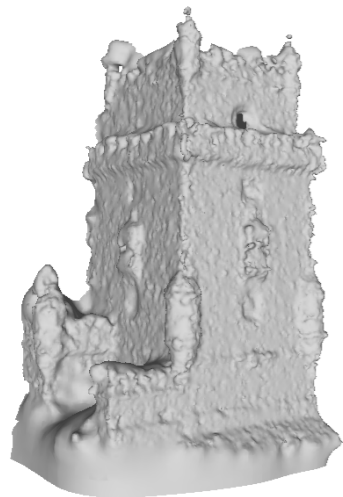
Camera
Calibration



Dense
Matching

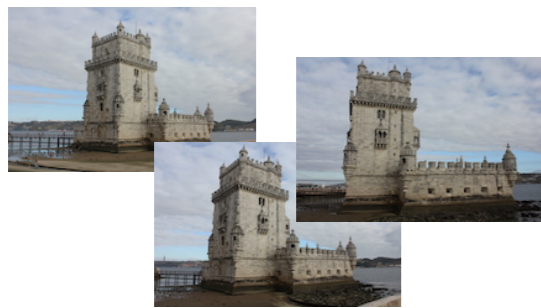


Surface
Reconstruction



3D model

3D from Photographs



Photographs



Automatic
Matching of
Images



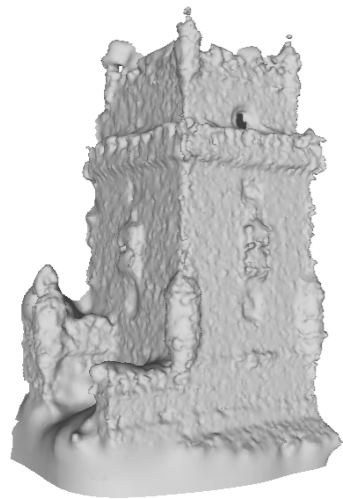
Camera
Calibration



Dense
Matching



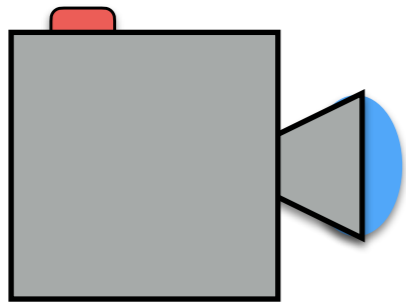
Surface
Reconstruction



3D model

Back to the Camera Model

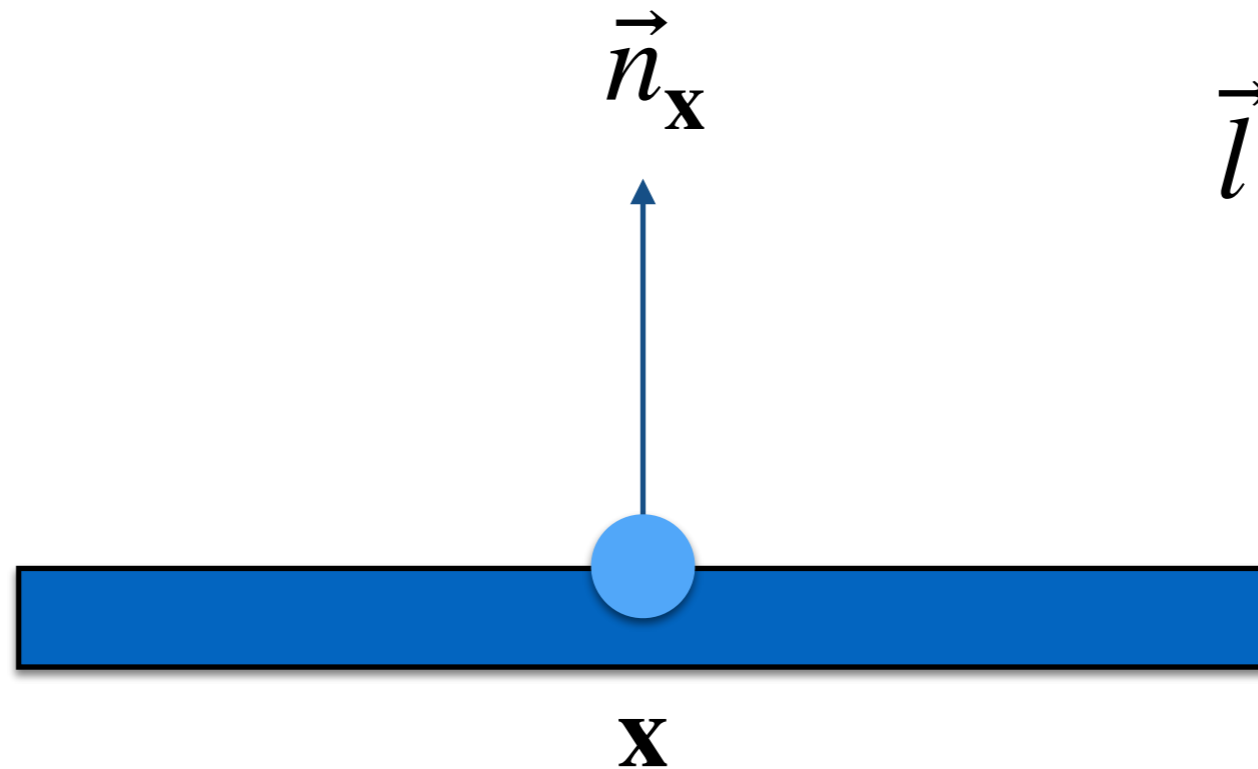
Camera Model: Image Formation



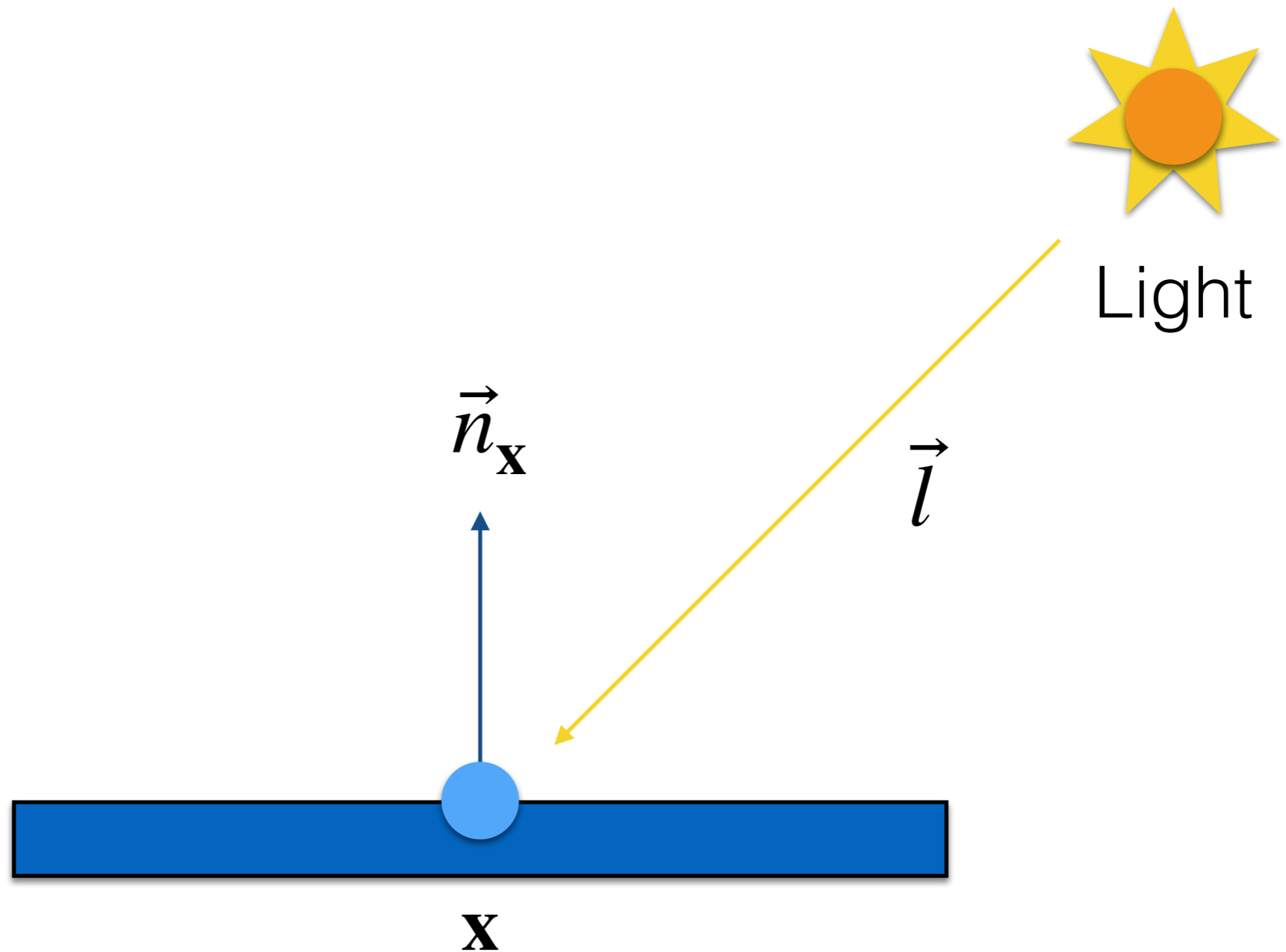
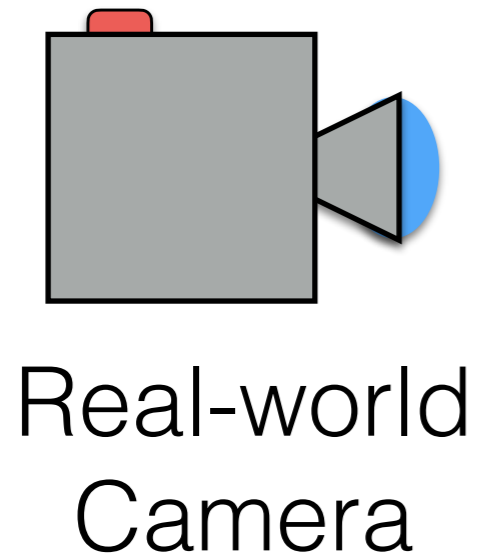
Real-world
Camera



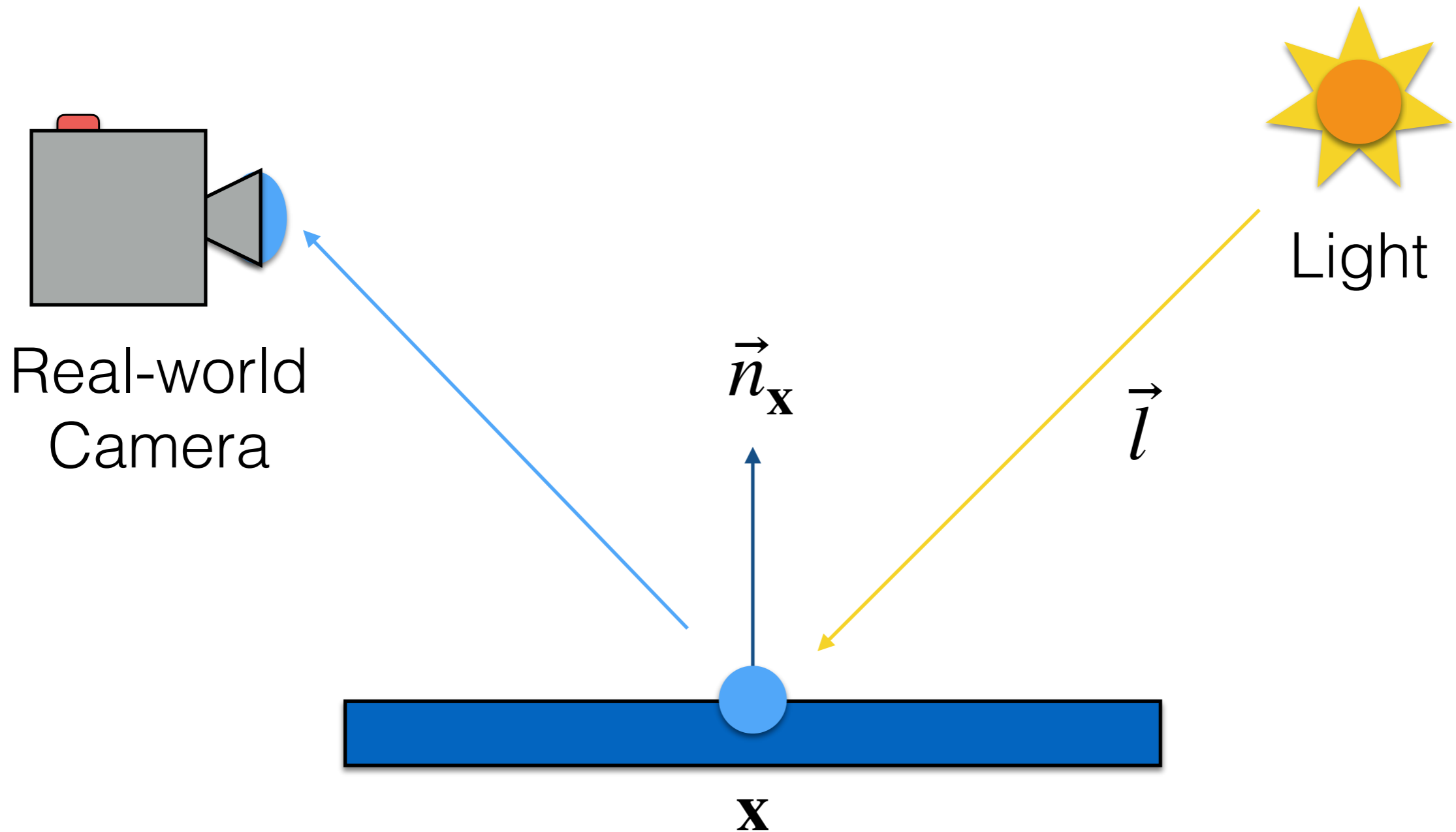
Light



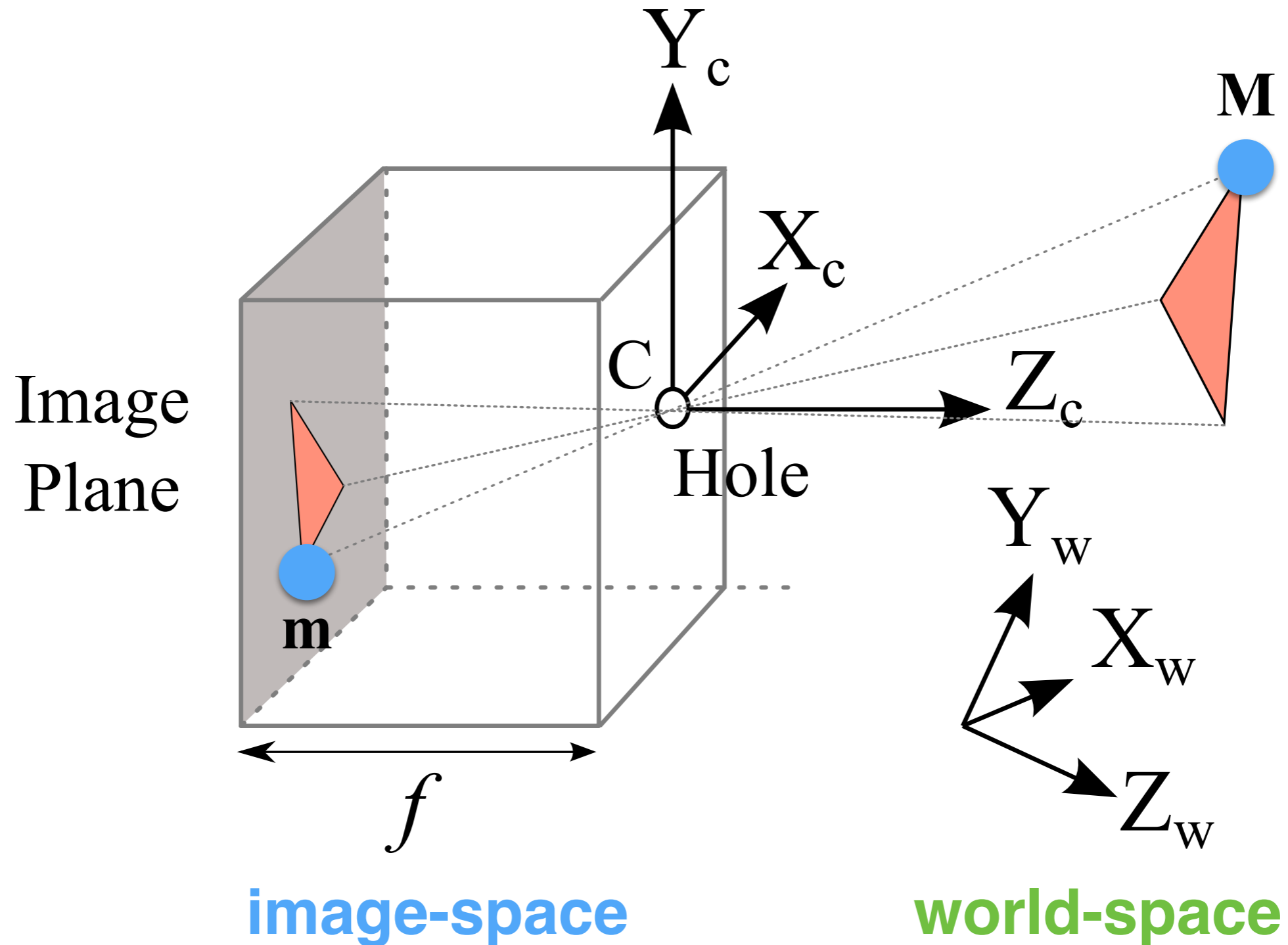
Camera Model: Image Formation



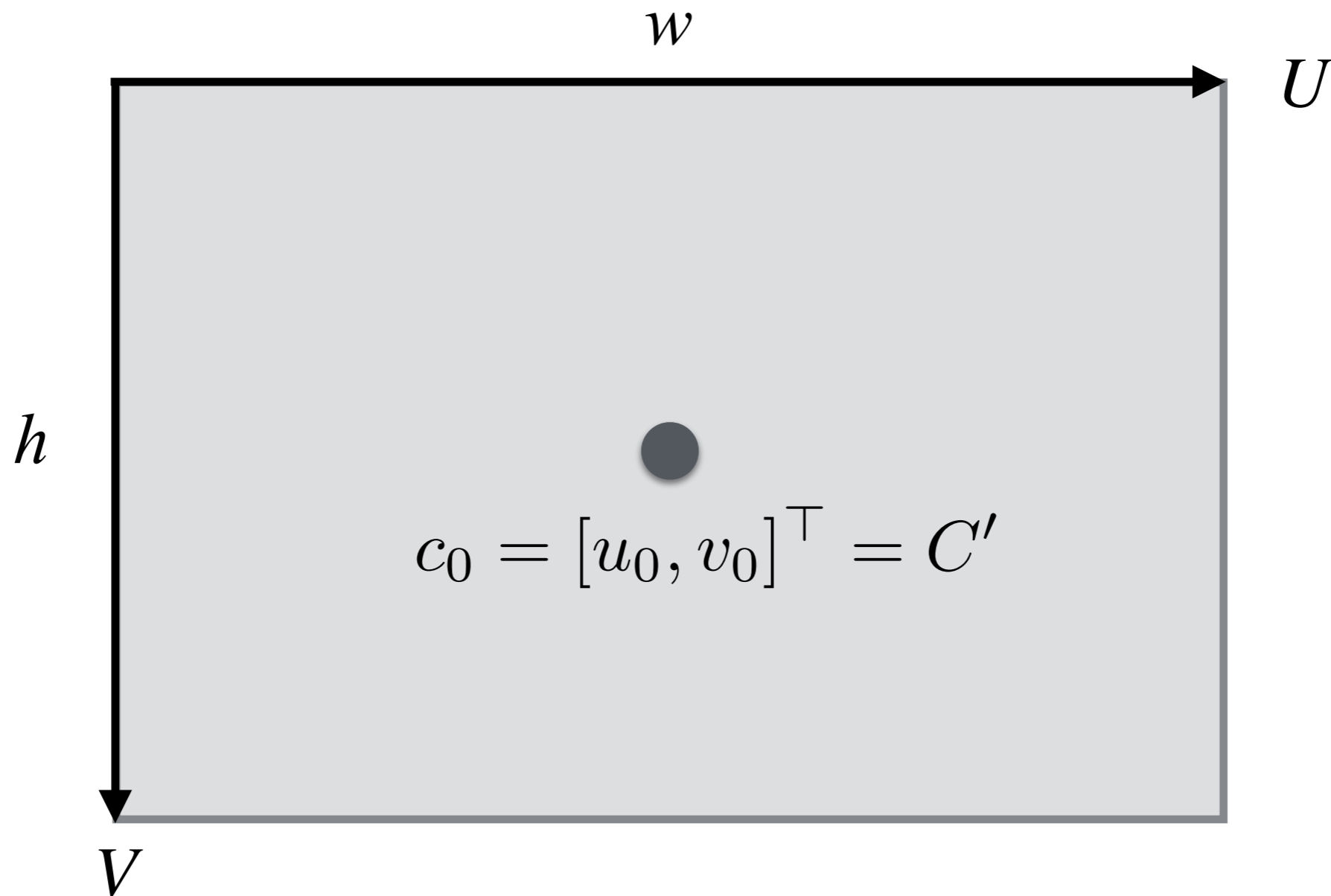
Camera Model: Image Formation



Camera Model: Pinhole Camera



Camera Model: Image Plane



- Pixels have different height and width; i.e., (k_u, k_v) .
- c_0 is called the principal point.
- The image plane has a finite size: w (width) and h (height)

Camera Model

- \mathbf{M} is a point in the 3D world, and it is defined as:

$$\mathbf{M} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

- \mathbf{m} is a 2D point, the projection of \mathbf{M} , and it lives in the image plane UV :

$$\mathbf{m} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

Camera Model

- By analyzing the two triangles (real-world and projected one), the following relationship emerges:

$$\frac{f}{z} = -\frac{u}{x} = -\frac{v}{y}$$

- This means that:

$$\begin{cases} u = -f \cdot \frac{x}{z} \\ v = -f \cdot \frac{y}{z} \end{cases}$$

Camera Model: Intrinsic Parameters

- If we take all into account of the optical center, and pixel size we obtain:

$$\begin{cases} u = -f \cdot \frac{x}{z} k_u + u_0 \\ v = -f \cdot \frac{y}{z} k_v + v_0 \end{cases}$$

- If we put this in matrix form, we obtain:

$$P = \begin{bmatrix} -fk_u & 0 & u_0 & 0 \\ 0 & -fk_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = K[I | \mathbf{0}] \quad K = \begin{bmatrix} -fk_u & 0 & u_0 \\ 0 & -fk_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{m}_z = P \cdot \mathbf{M} \quad \rightarrow \quad \mathbf{m} \sim P \cdot \mathbf{M} \quad \mathbf{m} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

Camera Model: Pinhole Camera

- The perspective projection is defined as:

$$\mathbf{m} \sim P \cdot \mathbf{M}$$

$$P = K \cdot \mathbf{G} = K \cdot [R | \mathbf{t}]$$

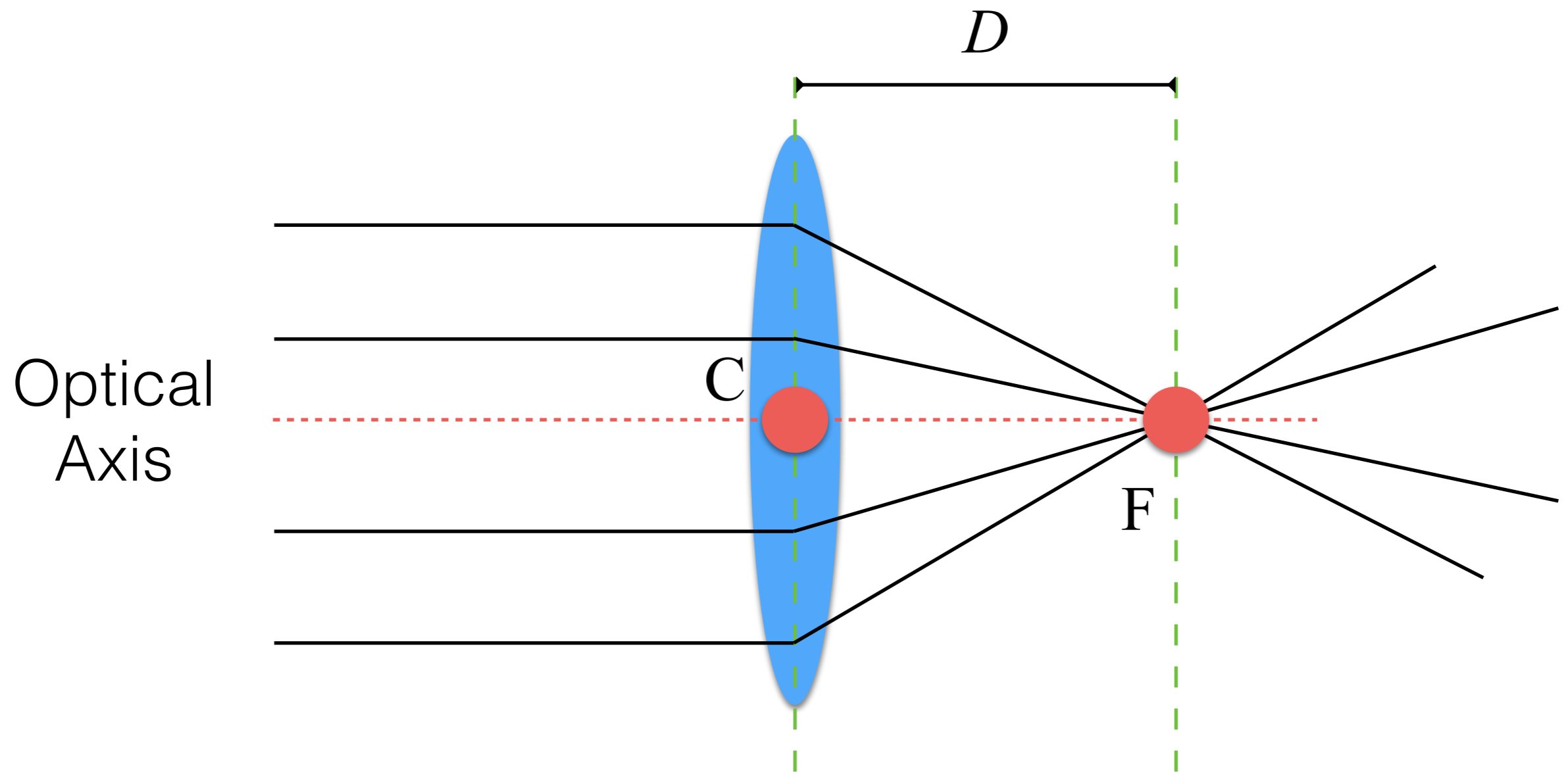
$$K = \begin{bmatrix} -fk_u & 0 & u_0 \\ 0 & -fk_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

Intrinsic Matrix

$$\mathbf{t} = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix} \quad R = \begin{bmatrix} \mathbf{r}_1^\top \\ \mathbf{r}_2^\top \\ \mathbf{r}_3^\top \end{bmatrix}$$

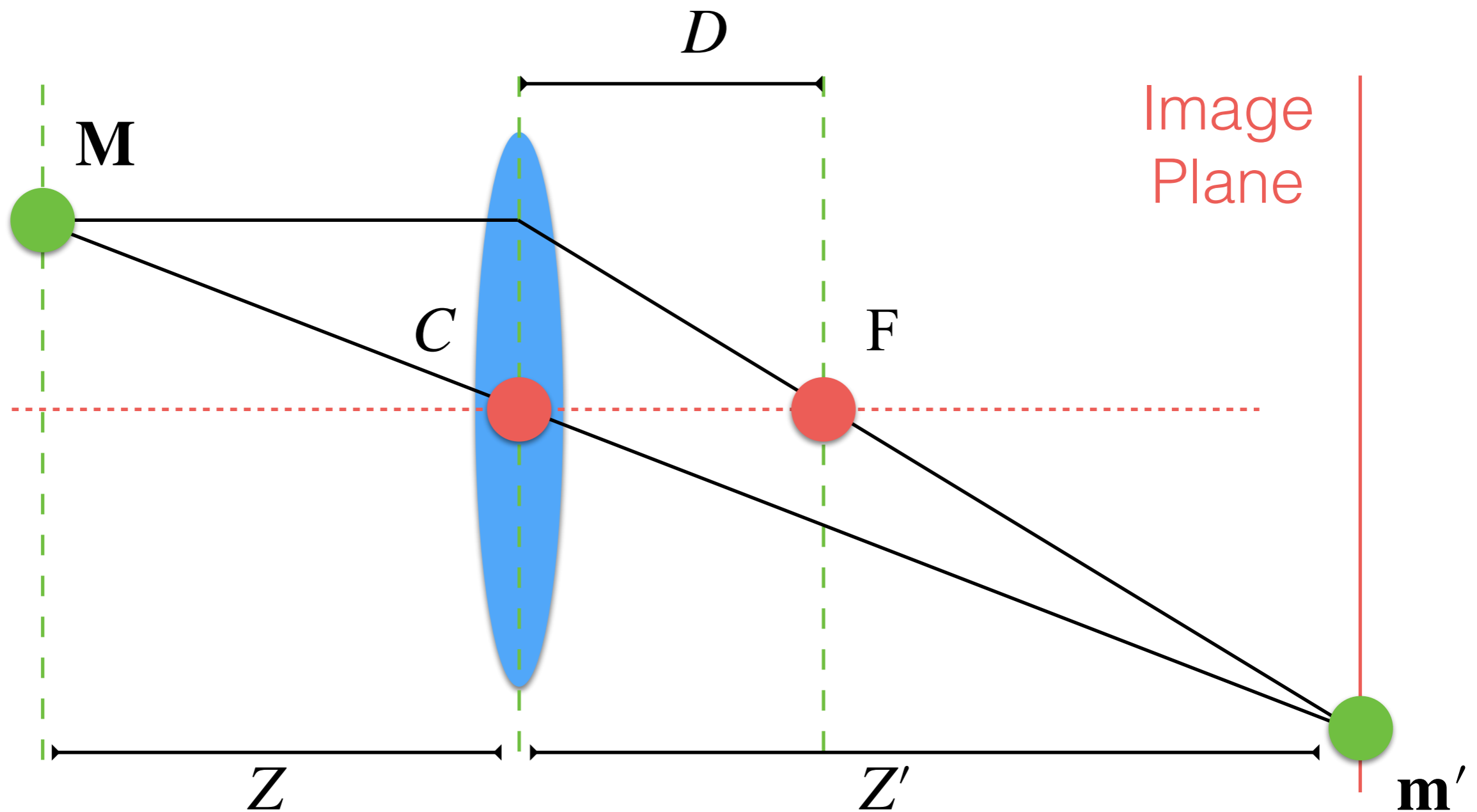
Extrinsic Matrix

Camera Model: Thin Lens



Camera Model: Thin Lens

$$\frac{1}{Z} + \frac{1}{Z'} = \frac{1}{D}$$



Camera Model: Thin Lens

- Typically, we model a thin lens system after projection:

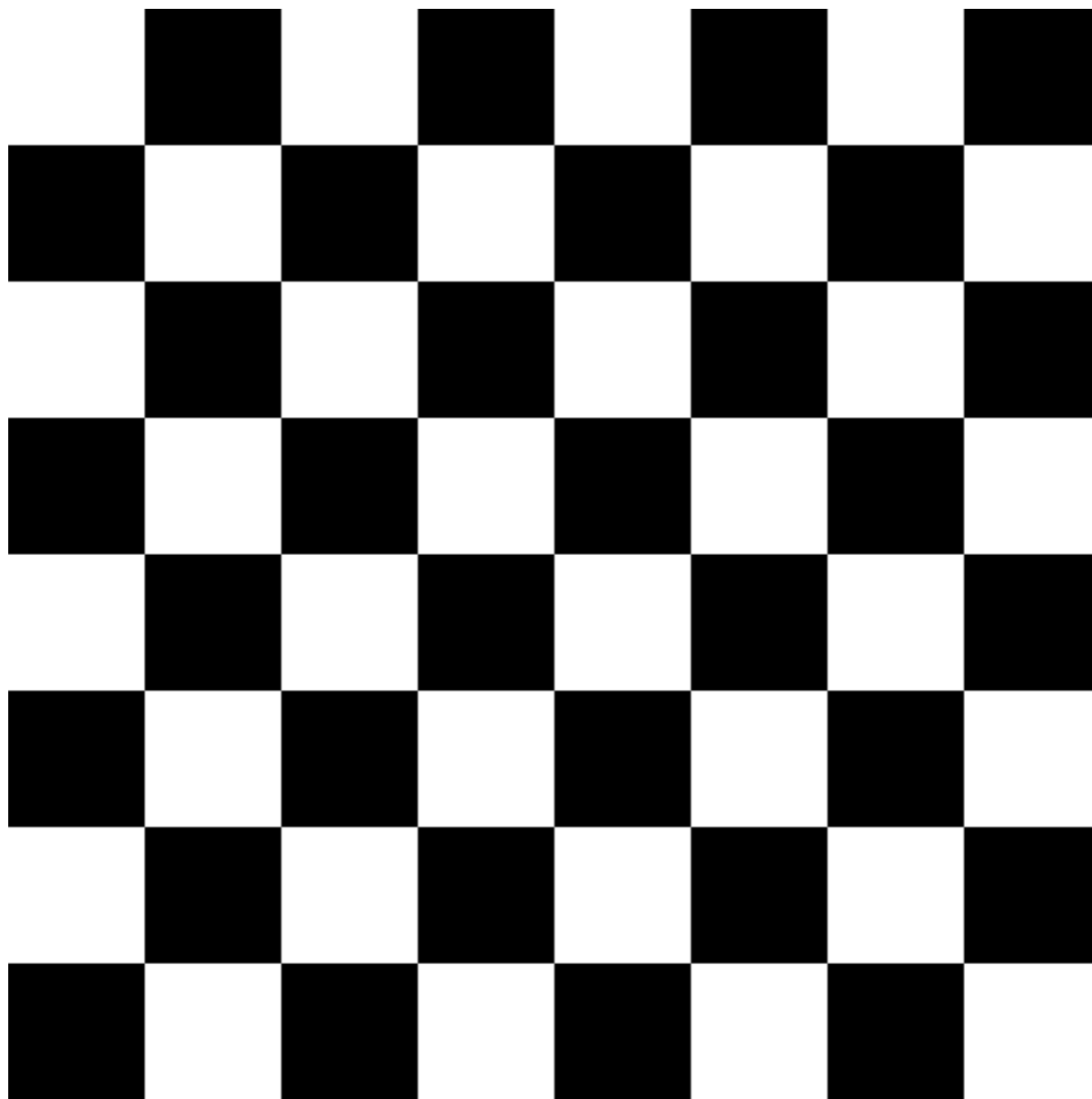
$$\mathbf{m}' = \begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} = \begin{cases} u' = (u - u_0)(1 + k_1 r_d^2 + \dots + k_n r_n^{2n}) + u_0 \\ v' = (v - v_0)(1 + k_1 r_d^2 + \dots + k_n r_n^{2n}) + v_0 \end{cases}$$

$$\mathbf{m} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim P \cdot \mathbf{M}$$

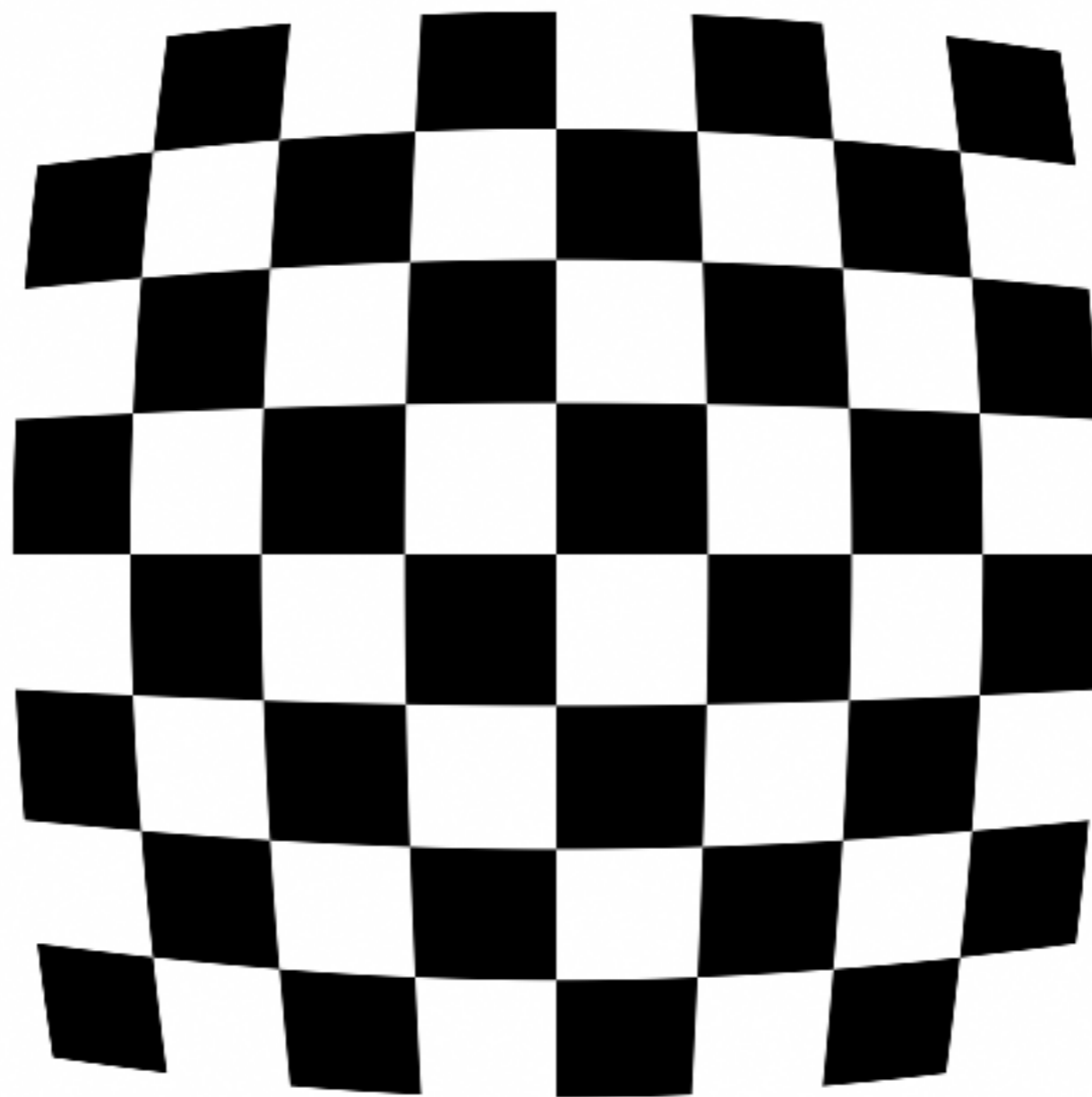
where n is usually set to 3, and r_d is defined as:

$$r_d^2 = \left(\frac{(u - u_o)^2}{\alpha_u^2} + \frac{(v - v_o)^2}{\alpha_v^2} \right)^2 \quad \alpha_u = -fk_u \quad \alpha_v = -fk_v$$

Camera Model: Thin Lens

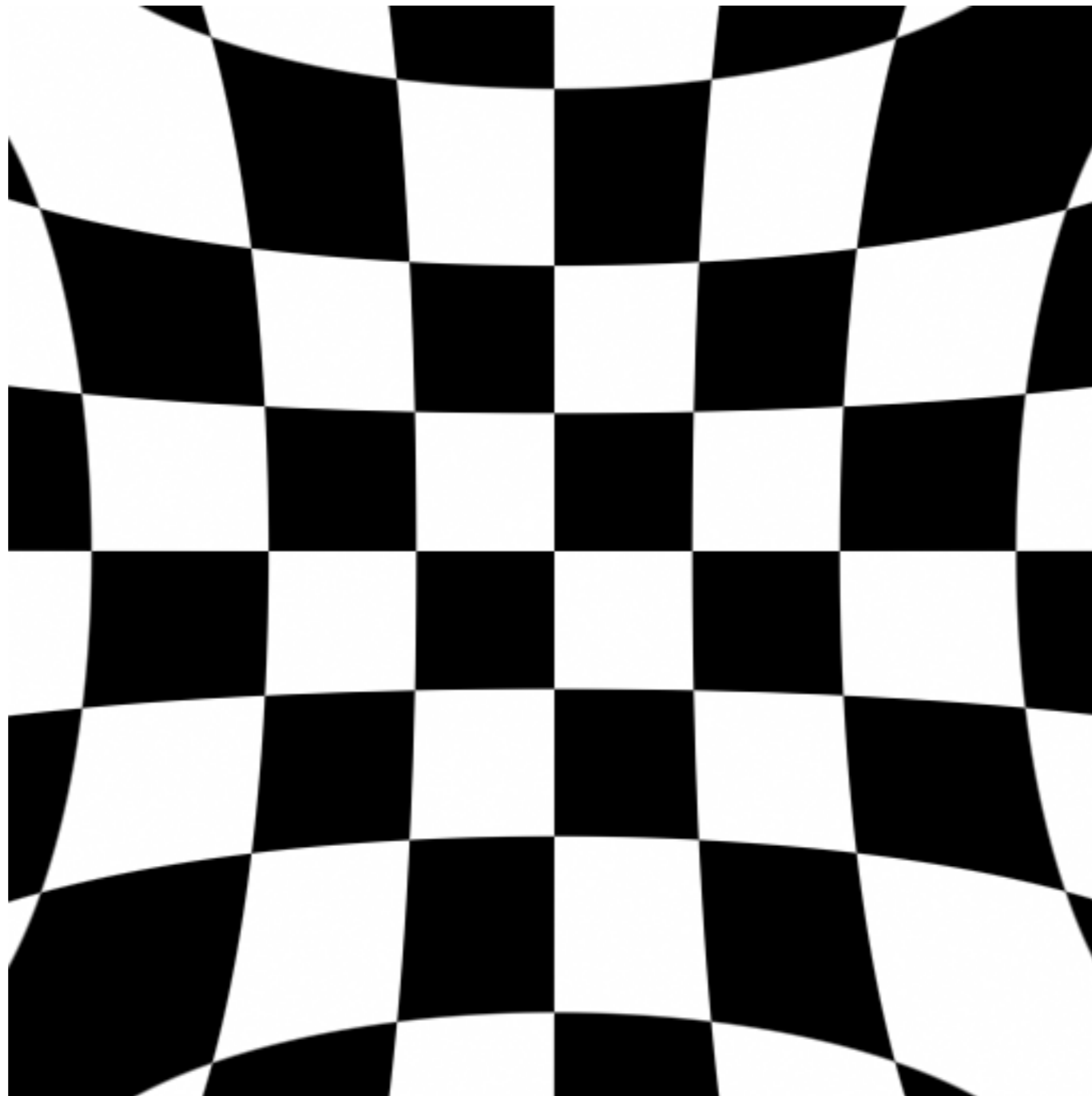


Camera Model: Thin Lens



Barrel distortion

Camera Model: Thin Lens



Pincushion distortion

Camera Pre-Calibration

Pre-Calibration: Why?

- In some cases, when we know the camera, it is useful to avoid intrinsics matrix estimation:
 - It is more precise.
 - We reduce computations.

Pre-Calibration: Parameters Estimation

- If we can have an “*estimation*” of K from camera parameters that are available in the camera specifications:

$$K = \begin{bmatrix} a & 0 & u_o \\ 0 & b & v_o \\ 0 & 0 & 1 \end{bmatrix}$$

- What do we need?
 - Focal length of the camera in mm (f), we can obtain it from the EXIF file of the JPEG/RAW file.
 - Resolution of the picture in pixels (w, h), we can find it in the manual of the camera or from the manufacturer specifications.
 - CCD/CMOS sensor size in mm (w_s, h_s), we can find it in the manual of the camera or from the manufacturer specifications.

Pre-Calibration: Parameters Estimation

- $a = f \cdot w/w_s$
- $b = f \cdot h/h_s$
- $u_0 = w/2$
- $v_0 = h/2$

Pre-Calibration: Parameters Estimation

- $a = f \cdot w/w_s$

- $b = f \cdot h/h_s$

- $u_0 = w/2$

- $v_0 = h/2$

Assuming it in the center!

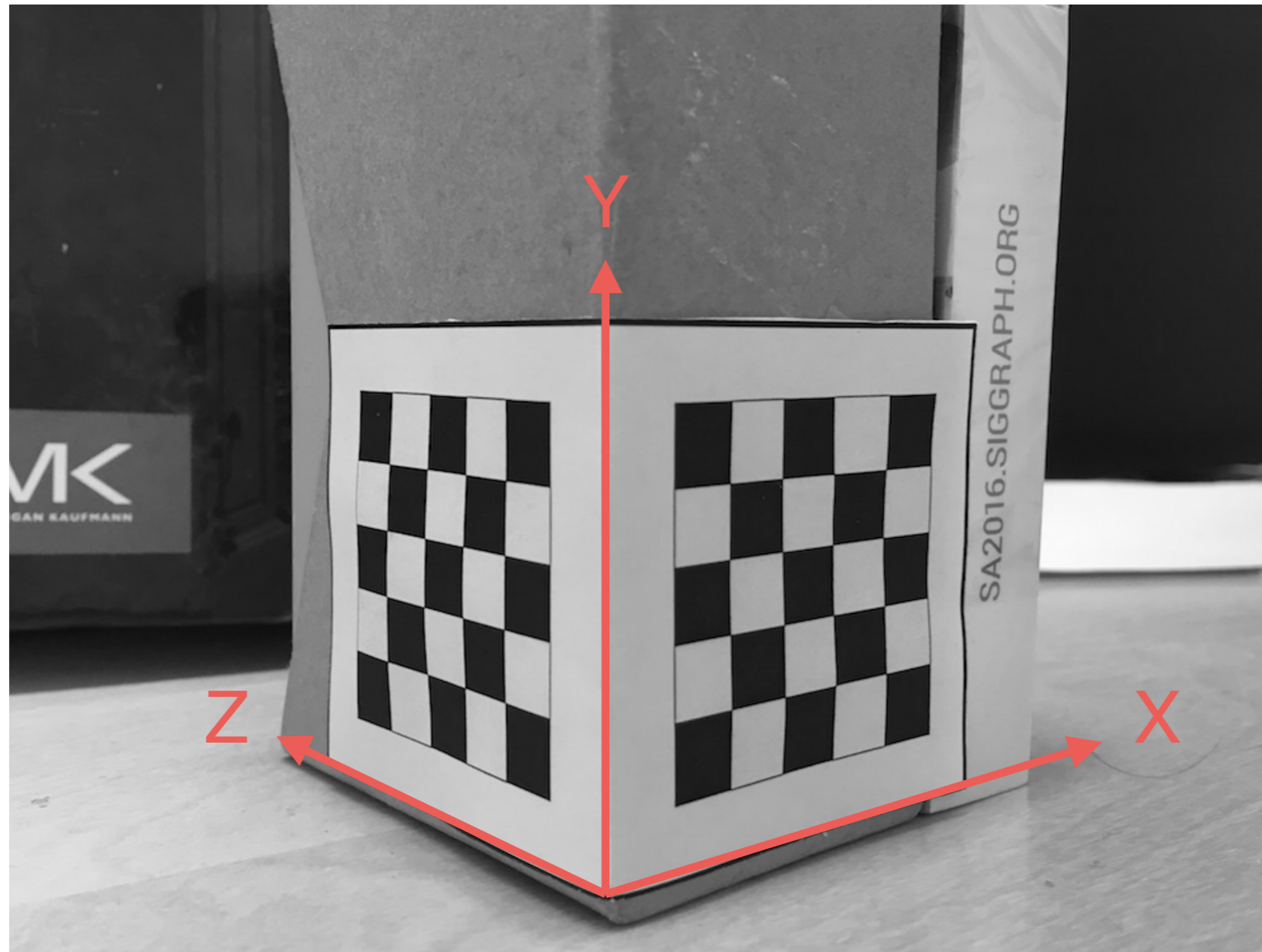
DLT:

Direct Linear Transform

DLT: Direct Linear Transform

- **Input:** a photograph of a non-coplanar calibration **object** (e.g., a checkerboard) with n 2D points (extracted manually or automatically) with known 3D coordinates (we know them because we built the calibration **object!**).
- **Output:** K of the camera. We can optionally recover $[R | \mathbf{t}]$.

DLT: Direct Linear Transform



DLT: Idea

$$\mathbf{m}_i = [u_i, v_i, 1]^\top \Leftrightarrow \mathbf{M}_i = [x, y, z, 1]^\top$$

2D-3D matches

DLT: Idea

- At this point, if we get the projection equation back, we can notice that we know something:

$$P = \begin{bmatrix} \mathbf{p}_1^\top \\ \mathbf{p}_2^\top \\ \mathbf{p}_3^\top \end{bmatrix} \begin{cases} u_i = \frac{\mathbf{p}_1^\top \cdot \mathbf{M}_i}{\mathbf{p}_3^\top \cdot \mathbf{M}_i} \\ v_i = \frac{\mathbf{p}_2^\top \cdot \mathbf{M}_i}{\mathbf{p}_3^\top \cdot \mathbf{M}_i} \end{cases}$$

DLT: Idea

$$\begin{cases} \mathbf{p}_1^\top \cdot \mathbf{M}_i - u_i \mathbf{p}_3^\top \cdot \mathbf{M}_i = 0 \\ \mathbf{p}_2^\top \cdot \mathbf{M}_i - v_i \mathbf{p}_3^\top \cdot \mathbf{M}_i = 0 \end{cases}$$

$$\mathbf{m}_i = [u_i, v_i, 1]^\top \Leftrightarrow \mathbf{M}_i = [x, y, z, 1]^\top$$

2D-3D matches

DLT: Linear System

- This leads to:

$$\begin{bmatrix} \mathbf{M}_i^\top & \mathbf{0} & -u_i \mathbf{M}_i^\top \\ \mathbf{0} & -\mathbf{M}_i^\top & v_i \mathbf{M}_i^\top \end{bmatrix} \cdot \begin{bmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \end{bmatrix} = \mathbf{0}$$

- For each point, we need to stack this equations obtaining a matrix A .
- We obtain a $2n \times 12$ linear system to solve.
- The minimum number of points to solve is 6, but more points are required to have robust and stable solutions.

What's the problem
with this method?

DLT: Direct Linear Transform

- DLT minimizes an algebraic error:
 - It does not have **geometric meaning**!!!
- Hang on, is it all wrong?
 - Nope, we can use it as input for a non-linear method.

DLT: Non-linear Refinement

- The non-linear refinement minimizes (at least squares) the distance between 2D points of the image (\mathbf{m}_i) and projected 3D points (\mathbf{M}_i):

$$\arg \min_P \sum_{i=1}^n \left(\frac{\mathbf{p}_1^\top \cdot \mathbf{M}_i}{\mathbf{p}_3^\top \cdot \mathbf{M}_i} - u_i \right)^2 + \left(\frac{\mathbf{p}_2^\top \cdot \mathbf{M}_i}{\mathbf{p}_3^\top \cdot \mathbf{M}_i} - v_i \right)^2$$

- Different methods for solving it such as Gradient Descent (*we need gradients*), Nelder-Mead's method (MATLAB's **fminsearch**), etc.

Now we have a nice
matrix $P\dots$

DLT: Direct Linear Transform

- Let's recap:

- K has to be upper-triangular. $K = \begin{bmatrix} -fk_u & 0 & u_0 \\ 0 & -fk_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}$

- R is orthogonal.

- $P = K[R|\mathbf{t}] = [K \cdot R|K \cdot \mathbf{t}] = [P'|\mathbf{p}_4]$

DLT: Direct Linear Transform

- QR decomposition of a matrix A :

$$A = O \cdot T$$

where:

- O is **orthogonal**.
 - T is **upper-triangular**.
- In our case, we have:

$$P' = K \cdot R \rightarrow (P')^{-1} = R^{-1} \cdot K^{-1}$$

DLT: Direct Linear Transform

- We apply the QR decomposition to P' as:

$$[(P')^{-1}]_{\text{QR}} = O \cdot T$$

- In our case, we have:

$$R = O^{-1} = O^{\top} \quad K = T^{-1}$$

- We compute \mathbf{t} as:

$$\mathbf{t} = K^{-1} \cdot \mathbf{p}_4 = T \cdot \mathbf{p}_4$$

and what's about the
radial distortion?

Estimating Radial Distortion

- Let's start with simple radial distortion; i.e., only a coefficient:

$$\begin{cases} u' = (u - u_0) \cdot (1 + k_1 r_d^2) + u_0 \\ v' = (v - v_0) \cdot (1 + k_1 r_d^2) + v_0 \end{cases}$$

$$r_d^2 = \left(\frac{(u - u_0)}{\alpha_u} \right)^2 + \left(\frac{(v - v_0)}{\alpha_v} \right)^2 \quad \alpha_u = -f \cdot k_u \quad \alpha_v = -f \cdot k_v$$

- Can we solve it?

Estimating Radial Distortion

- We have only one unknown, which is linear; i.e., k_1 :

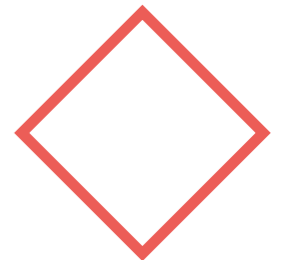
$$\begin{cases} k_1 = \frac{u' - u}{(u - u_0)r_d^2} \\ k_1 = \frac{v' - v}{(v - v_0)r_d^2} \end{cases}$$

- In theory, ***a single point is enough***, but it is better to use more points to get a more robust solution.

Homography

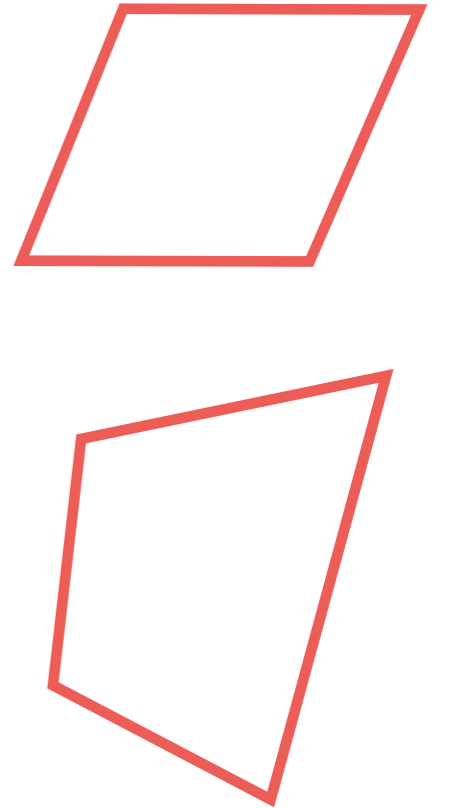
2D Transformations

- We can have different type of transformation (defined by a matrix) of 2D points:
 - Translation (2 degree of freedom [DoF]):
 - It preserves orientation.
 - Rigid/Euclidian (3 DoF); translation, and rotation:
 - It preserves lengths.
 - Similarity (4DoF); translation, rotation, and scaling:
 - It preserves angles.

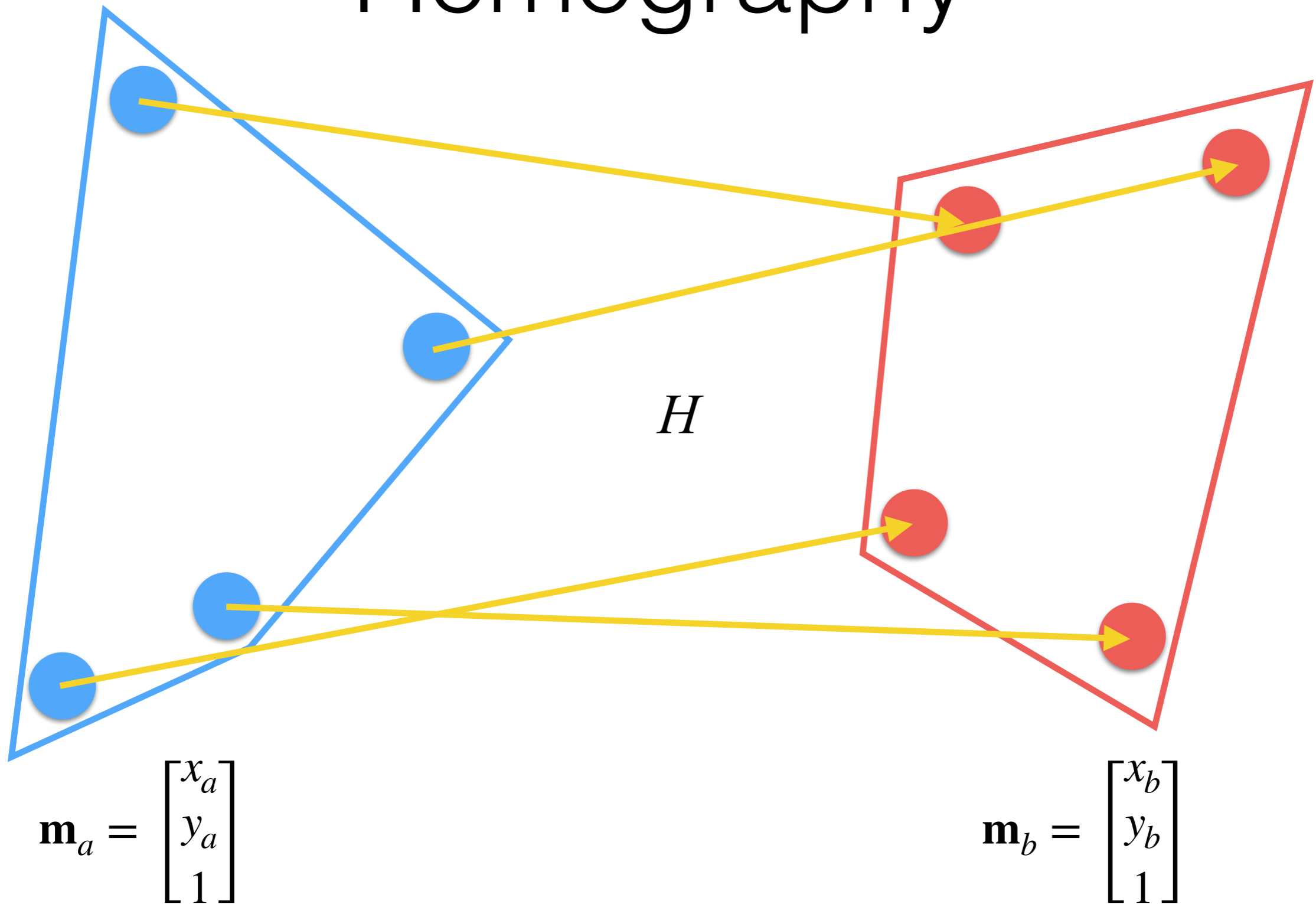


2D Transformations

- Affine (6 degree of freedom [DoF]):
 - It reserves parallelism.
- Projective (8 DoF):
 - It preserves straight lines.



2D Transformations: Homography



2D Transformations: Homography

- Homography is defined as

$$\hat{\mathbf{m}}_b = H \cdot \mathbf{m}_a \quad \rightarrow \quad \mathbf{m}_b = \begin{bmatrix} x_b \\ y_b \\ 1 \end{bmatrix} = \hat{\mathbf{m}}_b \frac{1}{\hat{m}_{b,3}}$$

- This is typically expressed as

$$\mathbf{m}_b \sim H \cdot \mathbf{m}_a$$

where H is a 3×3 non-singular matrix with 8 DoF.

Homography Estimation

$$\mathbf{m}_b \sim H \cdot \mathbf{m}_a \quad \mathbf{m}_a = \begin{bmatrix} x_a \\ y_a \\ 1 \end{bmatrix} \quad \mathbf{m}_b = \begin{bmatrix} x_b \\ y_b \\ 1 \end{bmatrix}$$

$$\mathbf{m}_b = \begin{cases} x_b = \frac{h_{11}x_a + h_{12}y_a + h_{13}}{h_{31}x_a + h_{32}y_a + h_{33}} \\ y_b = \frac{h_{21}x_a + h_{22}y_a + h_{23}}{h_{31}x_a + h_{32}y_a + h_{33}} \end{cases}$$

Homography Estimation

$$\begin{cases} x_b(h_{31}x_a + h_{32}y_a + h_{33}) - h_{11}x_a + h_{12}y_a + h_{13} = 0 \\ y_b(h_{31}x_a + h_{32}y_a + h_{33}) - h_{21}x_a + h_{22}y_a + h_{23} = 0 \end{cases}$$



Stacking multiple equations;
one for each match (at least 5!)

$$A \cdot \mathbf{vec}(H) = \mathbf{0}$$

Homography Estimation

- Again, we have minimized an algebraic error!!
- Technically speaking, we should run a non-linear optimization:

$$\arg \min_H \sum_{i=1}^m \left(x_{b,i} - \frac{\mathbf{h}_1^\top \cdot \mathbf{m}_{a,i}}{\mathbf{h}_3^\top \cdot \mathbf{m}_{a,i}} \right)^2 + \left(y_{b,i} - \frac{\mathbf{h}_2^\top \cdot \mathbf{m}_{a,i}}{\mathbf{h}_3^\top \cdot \mathbf{m}_{a,i}} \right)^2$$

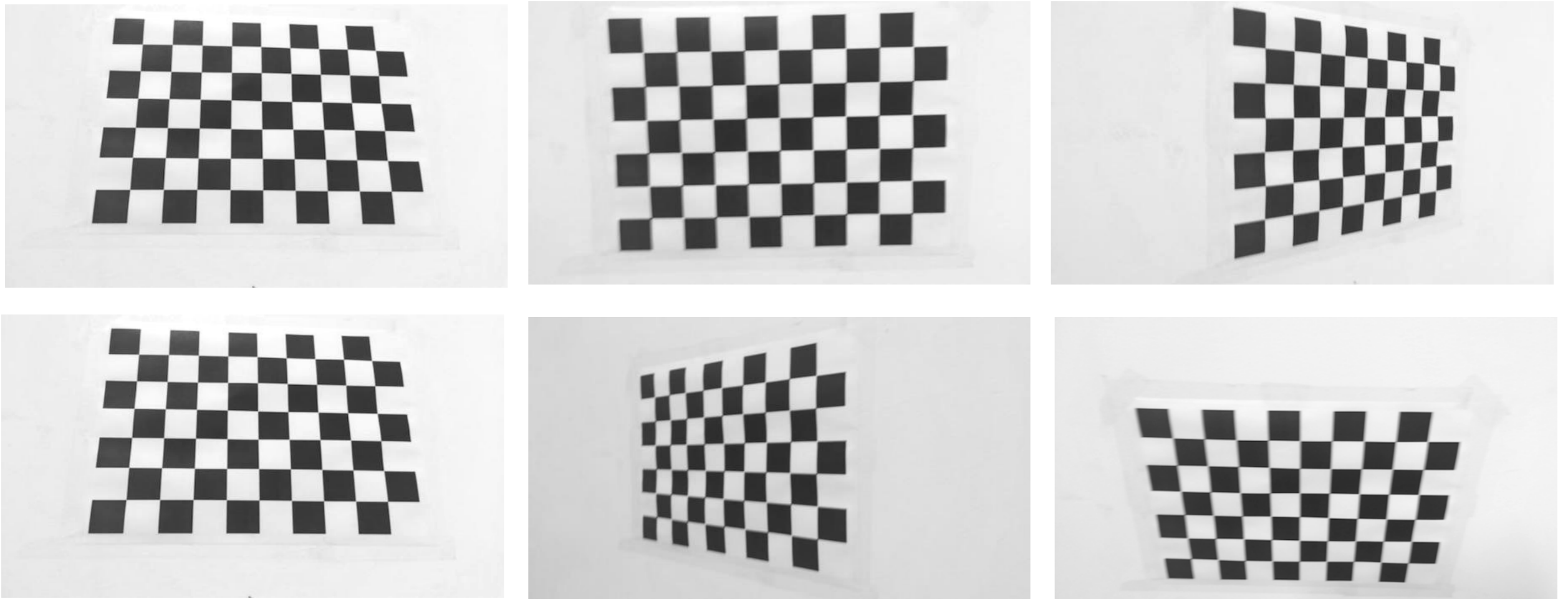
where $H = [\mathbf{h}_1 \quad \mathbf{h}_2 \quad \mathbf{h}_3]$, and m is the number of matched point.

Zhang's Method

Zhang's Method

- **Input:** a set of n photographs capturing a checkerboard (fully visible) or other patterns. From these, we have to extract m points (i.e., all corners of a checker!) in each photograph.
- **Output:** K . We can optionally compute $G = [R | \mathbf{t}]$ for each photograph.

Zhang's Method



A set of input images

Zhang's Method

$$K = \begin{bmatrix} -fk_u & 0 & u_0 \\ 0 & -fk_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

Zhang's Method

$$K = \begin{bmatrix} \alpha & c & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

Note: c is a function of the angle between the u -axis and v -axis in the image plane.

Zhang's Method

$$K = \begin{bmatrix} \alpha & c & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

Note: c is a function of the angle between the u -axis and v -axis in the image plane.

Zhang's Method

- **Assumption:**

- We have a set of photographs of a plane so Z is equal 0.
- So we have 3D points defined as

$$\mathbf{M} = \begin{bmatrix} x \\ y \\ 0 \\ 1 \end{bmatrix}$$

Zhang's Method

- This means that we have: $\mathbf{m} \sim P \cdot \mathbf{M}$

$$\begin{aligned} P \cdot \mathbf{M} &= K \cdot [R|\mathbf{t}] \cdot \begin{bmatrix} x \\ y \\ 0 \\ 1 \end{bmatrix} = \\ &= K \cdot [\mathbf{r}_1 \mathbf{r}_2 \mathbf{r}_3 | \mathbf{t}] \cdot \begin{bmatrix} x \\ y \\ 0 \\ 1 \end{bmatrix} = \\ &= K \cdot [\mathbf{r}_1 \mathbf{r}_2 | \mathbf{t}] \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \end{aligned}$$

Zhang's Method

$$= K \cdot [\mathbf{r}_1 \mathbf{r}_2 | \mathbf{t}] \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Zhang's Method

$$= K \cdot [\mathbf{r}_1 \mathbf{r}_2 | \mathbf{t}] \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Zhang's Method

$$=K \cdot [\mathbf{r}_1 \mathbf{r}_2 | \mathbf{t}] \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

This is a homography!

Zhang's Method

$$=K \cdot [\mathbf{r}_1 \mathbf{r}_2 | \mathbf{t}] \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

This is a homography!



Zhang's Method

$$= K \cdot [\mathbf{r}_1 \mathbf{r}_2 | \mathbf{t}] \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

This is a homography!



$$H = K \cdot [\mathbf{r}_1 \mathbf{r}_2 | \mathbf{t}]$$

Zhang's Method

$$= K \cdot [\mathbf{r}_1 \mathbf{r}_2 | \mathbf{t}] \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

This is a homography!

$$H = K \cdot [\mathbf{r}_1 \mathbf{r}_2 | \mathbf{t}]$$

Zhang's Method

$$= K \cdot [\mathbf{r}_1 \mathbf{r}_2 | \mathbf{t}] \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

This is a homography!

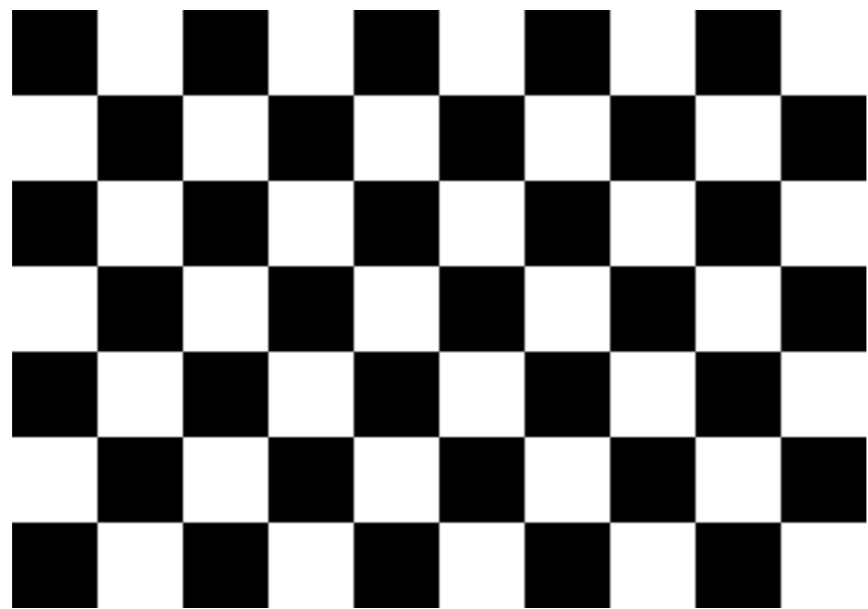
$$H = K \cdot [\mathbf{r}_1 \mathbf{r}_2 | \mathbf{t}]$$

$$H = [\mathbf{h}_1 \quad \mathbf{h}_2 \quad \mathbf{h}_3]$$

Zhang's Method

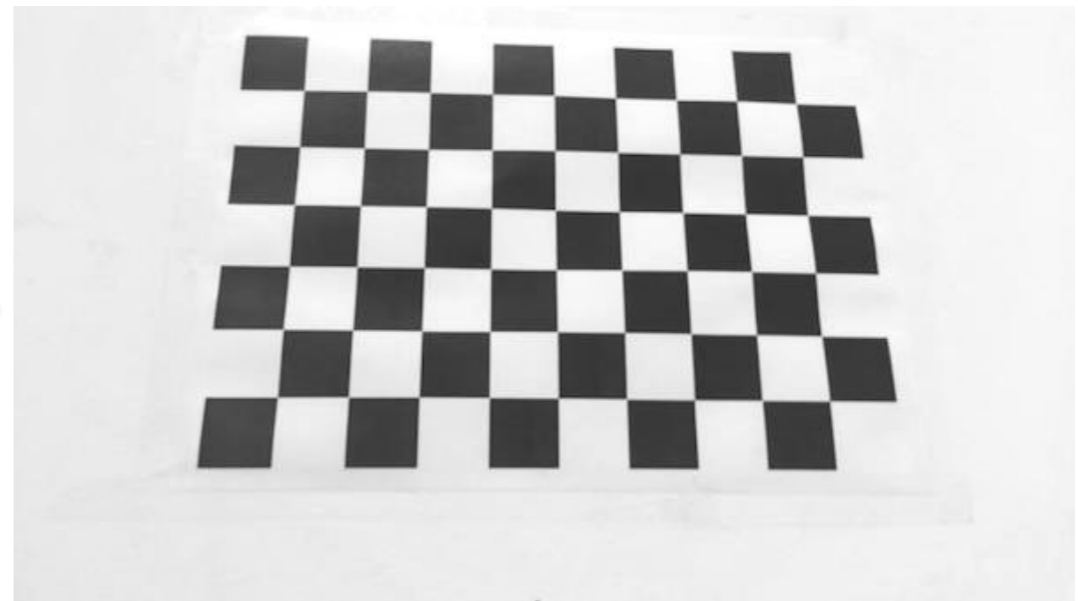
- What to do?
- For each photograph:
 - We compute the homography H between photographed checkerboard corners and its model.

Zhang's Method



Model

H



Photograph

Zhang's Method

- At this point, starting from H for each photograph, we need to compute K , R , and \mathbf{t} .
- \mathbf{r}_1 and \mathbf{r}_2 are orthonormal, so we have:

1.
$$\mathbf{h}_1^\top K^{-\top} K^{-1} \mathbf{h}_2 = 0$$

2.
$$\mathbf{h}_1^\top K^{-\top} K^{-1} \mathbf{h}_1 = \mathbf{h}_2^\top K^{-\top} K^{-1} \mathbf{h}_2$$

Zhang's Method

- Note that all K parameters can be compressed into:

$$B = K^{-\top} K^{-1} = \begin{bmatrix} \frac{1}{\alpha^2} & -\frac{c}{\alpha^2 \beta} & \frac{cv_0 - u_0 \beta}{\alpha^2 \beta} \\ -\frac{c}{\alpha^2 \beta} & \frac{c^2}{\alpha^2 \beta^2} + \frac{1}{\beta^2} & -\frac{c(cv_0 - u_0 \beta)}{\alpha^2 \beta^2} - \frac{v_0}{\beta^2} \\ \frac{cv_0 - u_0 \beta}{\alpha^2 \beta} & -\frac{c(cv_0 - u_0 \beta)}{\alpha^2 \beta^2} - \frac{v_0}{\beta^2} & \frac{(cv_0 - u_0 \beta)^2}{\alpha^2 \beta^2} + \frac{v_0^2}{\beta^2} + 1 \end{bmatrix}$$

- B is symmetric \rightarrow defined only by six values:

$$\mathbf{vec}(B) = [b_{11}, b_{12}, b_{22}, b_{13}, b_{23}, b_{33}]^{\top}$$

Zhang's Method

- Given that:

$$\mathbf{h}_1^\top K^{-\top} K^{-1} \mathbf{h}_1 = \mathbf{h}_2^\top K^{-\top} K^{-1} \mathbf{h}_2$$

- We have so:

$$\mathbf{h}_i^\top \cdot B \cdot \mathbf{h}_j = \mathbf{v}_{ij}^\top \cdot \mathbf{vec}(B)$$

- where:

$$H = [\mathbf{h}_1 \quad \mathbf{h}_2 \quad \mathbf{h}_3]$$

$$\mathbf{h}_i = [h_{i1} \quad h_{i2} \quad h_{i3}]^\top$$

$$\mathbf{v}_{ij} = \begin{bmatrix} h_{i1}h_{j1} \\ h_{i1}h_{j2} + h_{i2}h_{j1} \\ h_{i2}h_{j2} \\ h_{i3}h_{j1} + h_{i1}h_{j3} \\ h_{i3}h_{j2} + h_{i2}h_{j3} \\ h_{i3}h_{j3} \end{bmatrix}$$

Zhang's Method

- Given that \mathbf{r}_1 and \mathbf{r}_2 are orthonormal, and that:

$$\mathbf{h}_1^\top K^{-\top} K^{-1} \mathbf{h}_2 = 0$$

- We obtain:

$$\begin{bmatrix} \mathbf{v}_{12}^\top \\ (\mathbf{v}_{11} - \mathbf{v}_{12})^\top \end{bmatrix} \cdot \mathbf{vec}(B) = \mathbf{0}$$

Zhang's Method

- If n images of the model plane are observed, we obtain the following by stacking n of such equations:

$$\begin{bmatrix} \mathbf{v}_{12}^T \\ (\mathbf{v}_{11} - \mathbf{v}_{12})^T \end{bmatrix} \cdot \mathbf{vec}(B) = \mathbf{0}$$

- This leads to:

$$V \cdot \mathbf{vec}(B) = \mathbf{0}$$

- V is $2n \times 6$ matrix, so we need $n > 2$.

Zhang's Method

- At this point, we can compute elements of K as

$$u_0 = (b_{12}b_{13} - b_{11}b_{23}) / (b_{11}b_{22} - b_{12}^2)$$

$$\lambda = b_3^3 - (b_{13}^2 + v_0(b_{12}b_{13} - b_{11} - b_{23})) / b_{11}$$

$$\alpha = \sqrt{\lambda - b_{11}}$$

$$\beta = \sqrt{\lambda b_{11} / (b_{11}b_{22} - b_{12}^2)}$$

$$c = -b_{12}\alpha^2\beta / \lambda$$

$$u_0 = cv_0 / \alpha - b_{13}\alpha^2 / \lambda$$

Zhang's Method

- Furthermore, we can extract the pose as

$$\mathbf{r}_1 = \lambda \cdot K^{-1} \mathbf{h}_1$$

$$\mathbf{r}_2 = \lambda \cdot K^{-1} \mathbf{h}_2$$

$$\mathbf{r}_3 = \mathbf{r}_1 \times \mathbf{r}_2$$

$$\mathbf{t} = \lambda K^{-1} \mathbf{h}_3$$

Zhang's Method: Non-Linear Refinement

- So far, we have obtained a solution through minimizing an algebraic distance that is not physically meaningful!
- From that solution, we can use a non-linear method for minimizing the following error:

$$\sum_{i=1}^n \sum_{j=1}^m \left\| \mathbf{m}_{i,j} - \tilde{\mathbf{m}}(K, R_i, \mathbf{t}_i, \mathbf{M}_j) \right\|^2$$

- where n is the number of photographs, and m is the number of matched points.

Zhang's Method: Non-Linear Refinement

- So far, we have obtained a solution through minimizing an algebraic distance that is not physically meaningful!
- From that solution, we can use a non-linear method for minimizing the following error:

$$\sum_{i=1}^n \sum_{j=1}^m \left\| \mathbf{m}_{i,j} - \tilde{\mathbf{m}}(K, R_i, \mathbf{t}_i, \mathbf{M}_j) \right\|^2$$

This function projects \mathbf{M}_j points (3D) given K and $G_i = [R_i | \mathbf{t}_i]$.

- where n is the number of photographs, and m is the number of matched points.

Zhang's Method: Optical Distortion

- What's about the parameters for modeling the radial distortion?

$$\begin{bmatrix} (u - u_0)r_d^2 & (u - u_0)r_d^4 \\ (v - v_0)r_d^2 & (v - v_0)r_d^4 \end{bmatrix} \cdot \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} = \begin{bmatrix} u' - u \\ v' - v \end{bmatrix}$$

$$r_d^2 = \left(\frac{(u - u_0)}{\alpha_u} \right)^2 + \left(\frac{(v - v_0)}{\alpha_v} \right)^2 \quad \alpha_u = -f \cdot k_u \quad \alpha_v = -f \cdot k_v$$

Zhang's Method: Optical Distortion

- What's about the parameters for modeling the radial distortion?

$$\begin{bmatrix} (u - u_0)r_d^2 & (u - u_0)r_d^4 \\ (v - v_0)r_d^2 & (v - v_0)r_d^4 \end{bmatrix} \cdot \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} = \begin{bmatrix} u' - u \\ v' - v \end{bmatrix}$$

$$r_d^2 = \left(\frac{(u - u_0)}{\alpha_u} \right)^2 + \left(\frac{(v - v_0)}{\alpha_v} \right)^2 \quad \alpha_u = -f \cdot k_u \quad \alpha_v = -f \cdot k_v$$



Zhang's Method: Optical Distortion

- What's about the parameters for modeling the radial distortion?

$$\begin{bmatrix} (u - u_0)r_d^2 & (u - u_0)r_d^4 \\ (v - v_0)r_d^2 & (v - v_0)r_d^4 \end{bmatrix} \cdot \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} = \begin{bmatrix} u' - u \\ v' - v \end{bmatrix}$$

$$r_d^2 = \left(\frac{(u - u_0)}{\alpha_u} \right)^2 + \left(\frac{(v - v_0)}{\alpha_v} \right)^2 \quad \alpha_u = -f \cdot k_u \quad \alpha_v = -f \cdot k_v$$


$$D \cdot \mathbf{k} = \mathbf{d}$$

Zhang's Method: Optical Distortion

- What's about the parameters for modeling the radial distortion?

$$\begin{bmatrix} (u - u_0)r_d^2 & (u - u_0)r_d^4 \\ (v - v_0)r_d^2 & (v - v_0)r_d^4 \end{bmatrix} \cdot \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} = \begin{bmatrix} u' - u \\ v' - v \end{bmatrix}$$

$$r_d^2 = \left(\frac{(u - u_0)}{\alpha_u} \right)^2 + \left(\frac{(v - v_0)}{\alpha_v} \right)^2 \quad \alpha_u = -f \cdot k_u \quad \alpha_v = -f \cdot k_v$$


$$D \cdot \mathbf{k} = \mathbf{d}$$



Zhang's Method: Optical Distortion

- What's about the parameters for modeling the radial distortion?

$$\begin{bmatrix} (u - u_0)r_d^2 & (u - u_0)r_d^4 \\ (v - v_0)r_d^2 & (v - v_0)r_d^4 \end{bmatrix} \cdot \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} = \begin{bmatrix} u' - u \\ v' - v \end{bmatrix}$$

$$r_d^2 = \left(\frac{(u - u_0)}{\alpha_u} \right)^2 + \left(\frac{(v - v_0)}{\alpha_v} \right)^2 \quad \alpha_u = -f \cdot k_u \quad \alpha_v = -f \cdot k_v$$


$$D \cdot \mathbf{k} = \mathbf{d}$$


$$\mathbf{k} = (D^\top \cdot D)^{-1} \cdot D^\top \cdot \mathbf{d}$$

Zhang's Method: Non-Linear Refinement

- As before, first algebraic solution, and then a non-linear solution.
- We extend the previous non-linear model to include optical distortion:

$$\bullet \sum_{i=1}^n \sum_{j=1}^m \left\| \mathbf{m}_{i,j} - \tilde{\mathbf{m}}(K, R_i, \mathbf{t}_i, \mathbf{M}_j, \mathbf{k}) \right\|^2$$

- where n is the number of photographs, and m is the number of matched points.

Zhang's Method: Non-Linear Refinement

- As before, first algebraic solution, and then a non-linear solution.
- We extend the previous non-linear model to include optical distortion:

$$\bullet \sum_{i=1}^n \sum_{j=1}^m \left\| \mathbf{m}_{i,j} - \tilde{\mathbf{m}}(K, R_i, \mathbf{t}_i, \mathbf{M}_j, \mathbf{k}) \right\|^2$$

This function projects \mathbf{M}_j points (3D) given K and $G_i = [R_i | \mathbf{t}_i]$ and radial parameters \mathbf{k} .

- where n is the number of photographs, and m is the number of matched points.

that's all folks!