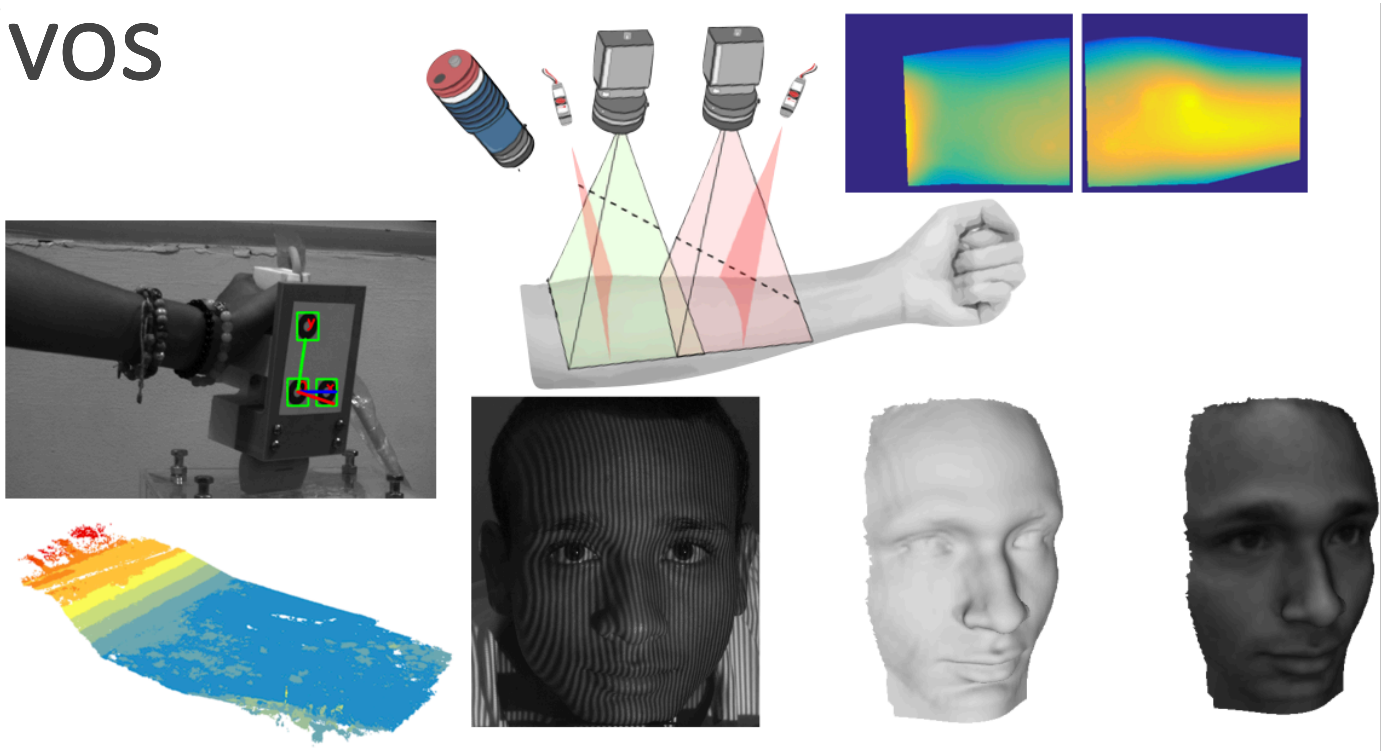
https://github.com/opi-lab/stsiva-workshop

Fundamentos de reconstrucción 3D mediante sistemas activos y pasivos



Jhacson Meza and Andrés G. Marrugo

Grupo de Física Aplicada y Procesamiento de Imágenes y Señales

opilab.utb.edu.co

17 de septiembre de 2021

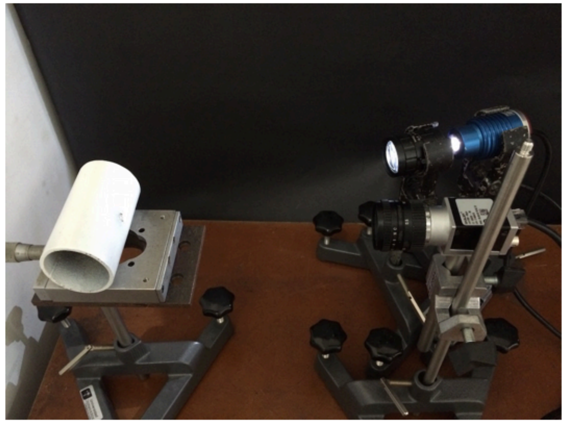
opilab.utb.edu.co

OPI·Lab

Publicaciones Personal Proyectos Seminario Docencia Wiki English

OPI·Lab - Laboratorio de Óptica y Procesamiento de Imágenes.

Somos un laboratorio de óptica aplicada y procesamiento de imágenes enfocados en metrología óptica, imagenología médica, reconstrucción 3D, microscopía y procesamiendo digital de imágenes.

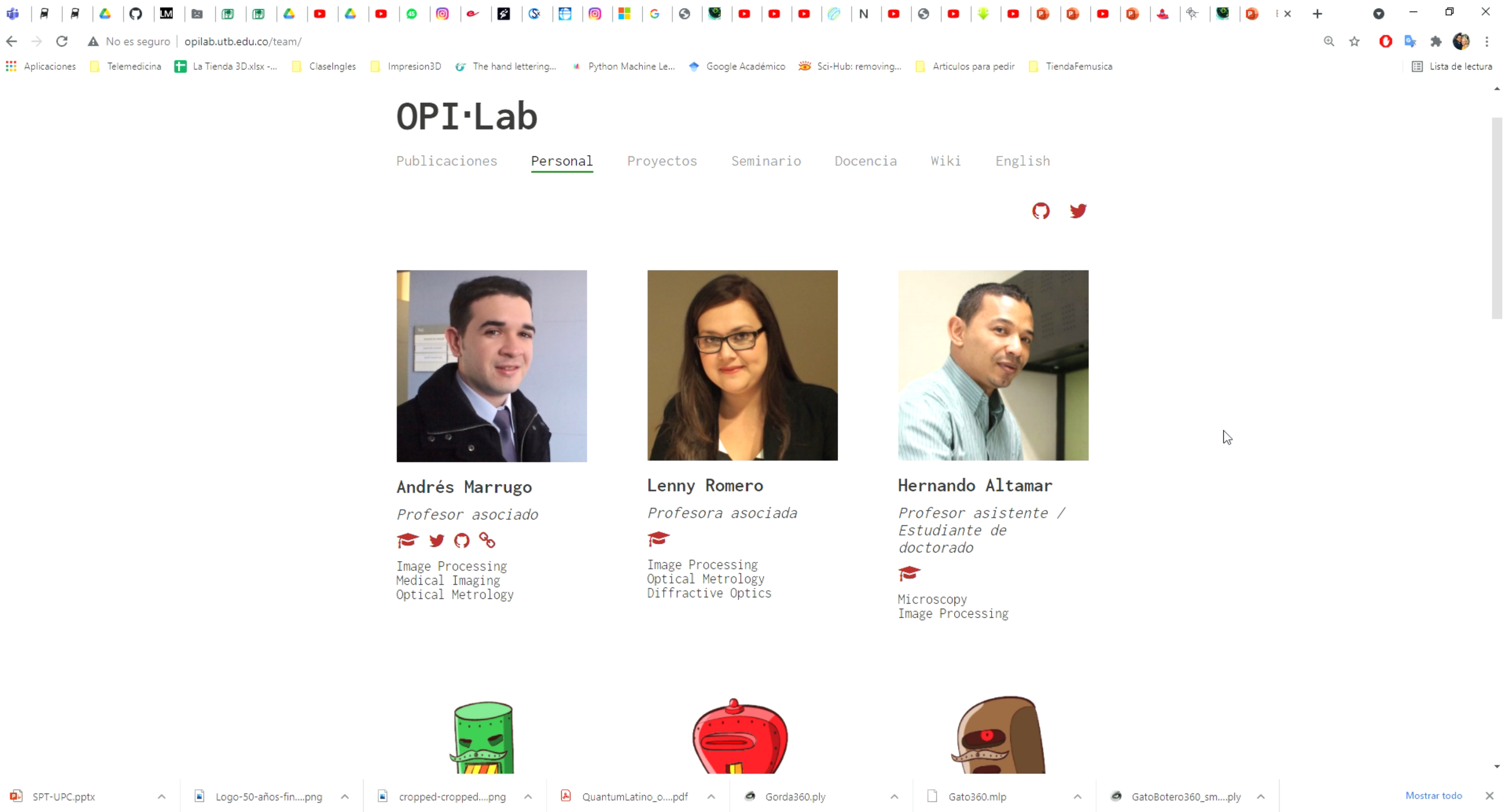


Noticias

10. Octubre 2018

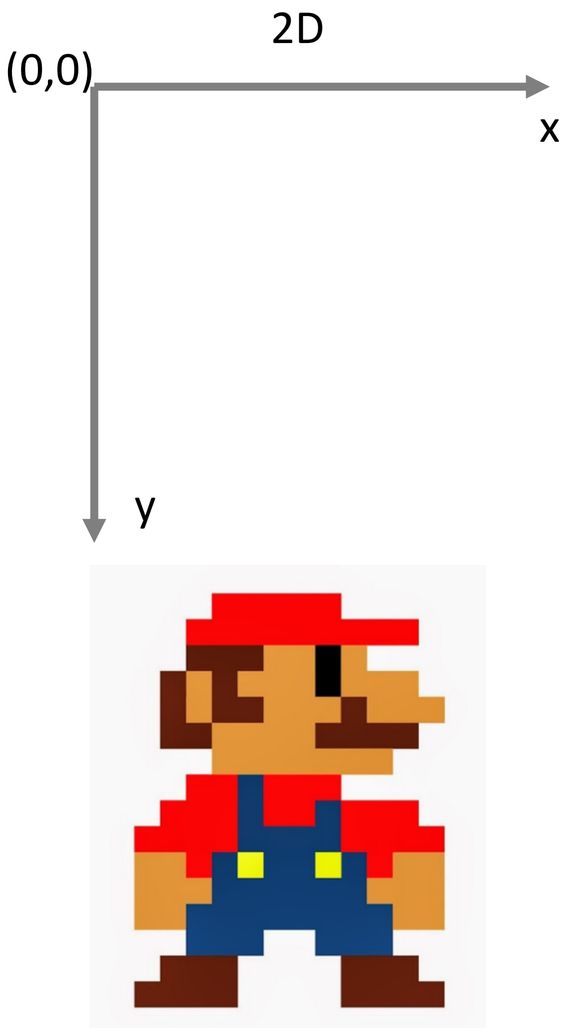
Co-edición de libro **New Uses of Micro and Nanomaterials** por el Prof. Marrugo y publicación del capítulo de libro **Magnetic Materials by Melt Spinning Method, Structural Characterization, and Numerical Modeling**

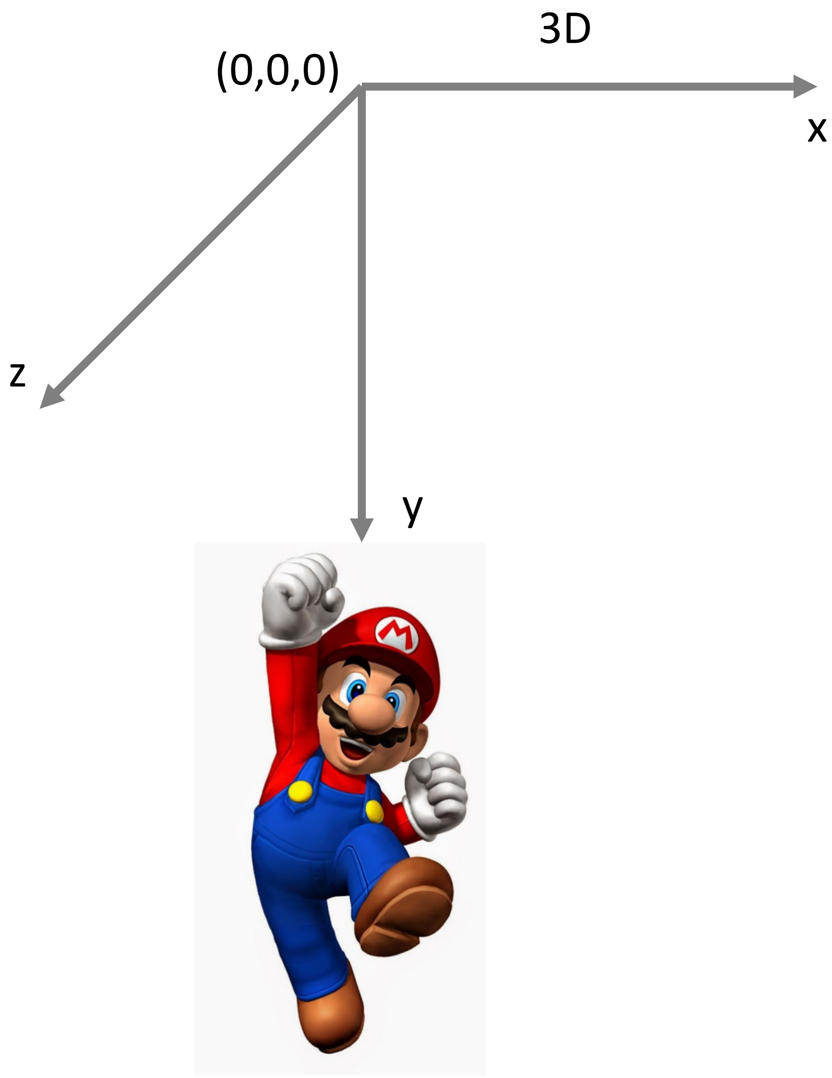
Team



No es seguro opilab.utb.edu.co/team/
OPI·Lab
Publicaciones
Personal
Proyectos
Seminario
Docencia
Wiki
English
Andrés Marrugo
Profesor asociado
Image Processing
Medical Imaging
Optical Metrology
Lenny Romero
Profesora asociada
Image Processing
Optical Metrology
Diffractive Optics
Hernando Altamar
Profesor asistente / Estudiante de doctorado
Microscopy
Image Processing

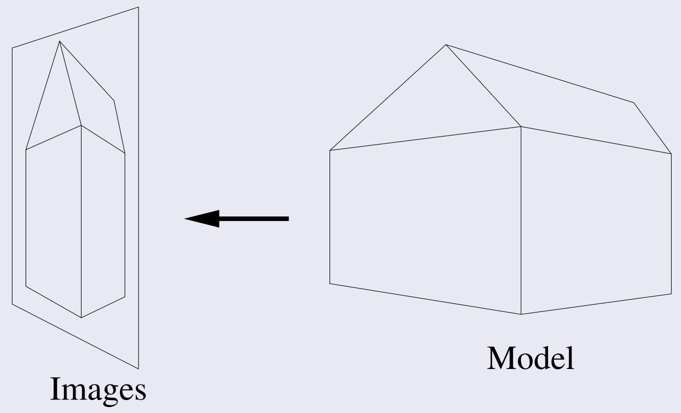
What is 3D?





However, we have to distinguish …

Computer Graphics

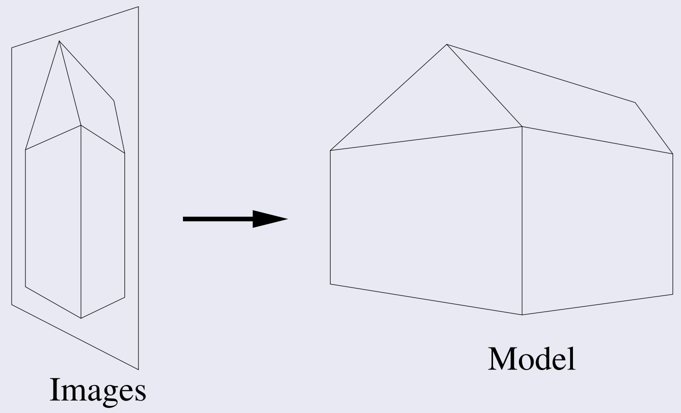


Generate images from a 3D model.

Solved

$\neq$

Computer Vision



The inverse problem: Generate 3D model from images.

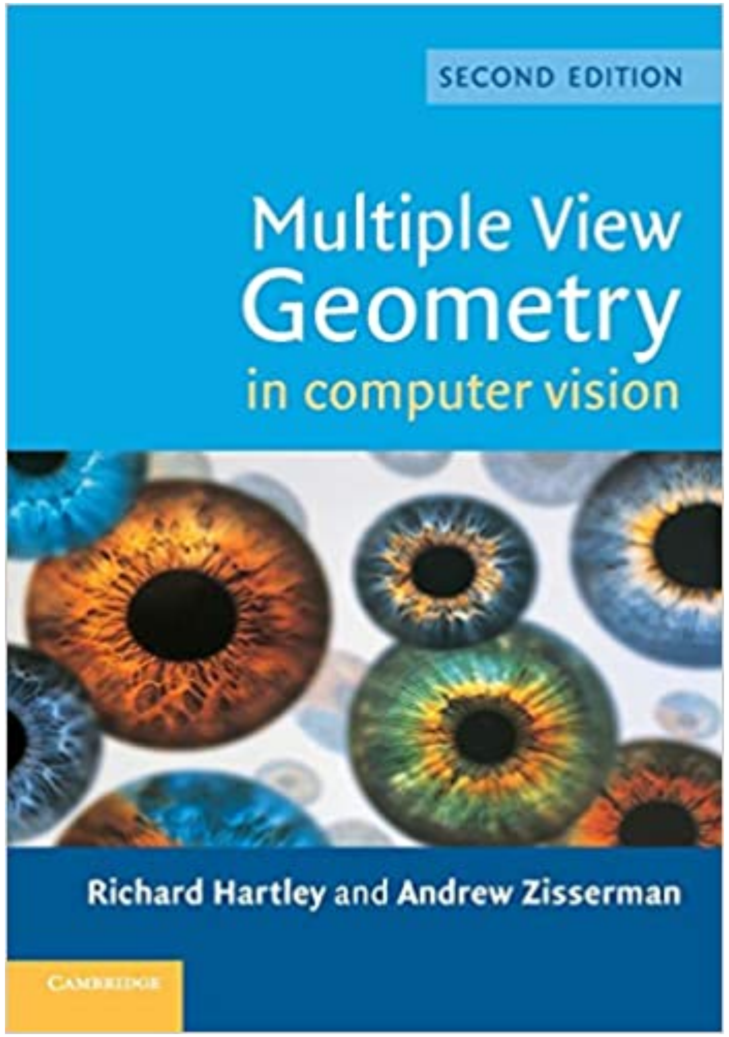
Partly solved

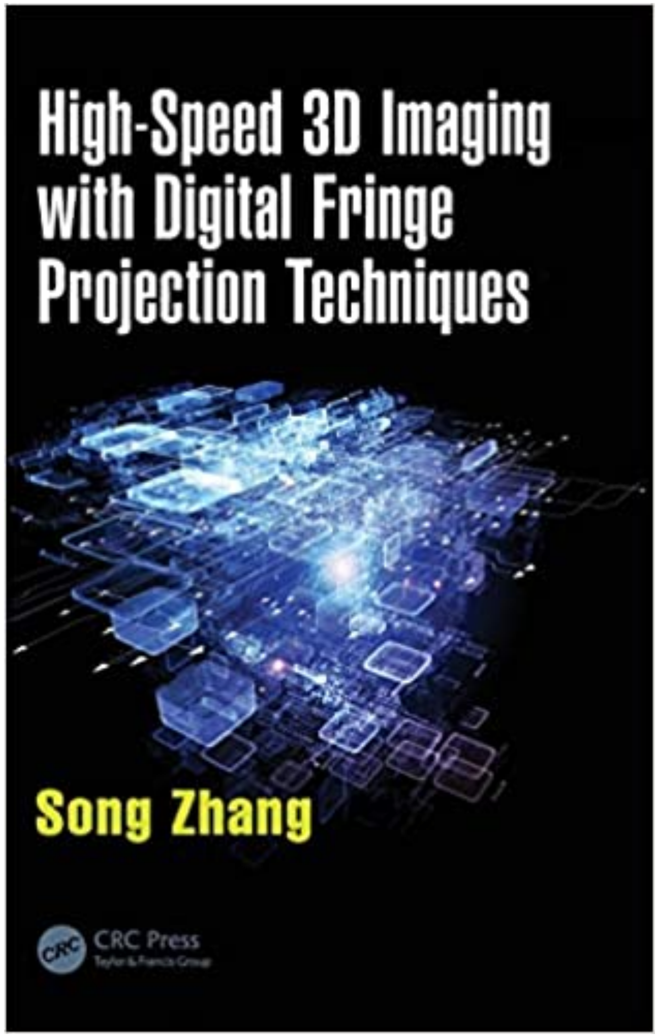
Source: M. Oskarson

In this workshop we'll cover…

1. Pinhole camera model, projective geometry, and camera calibration.
2. 3D from passive stereo vision system.
3. 3D from active stereo vision system (aka structured light).
4. Applications.

Useful references





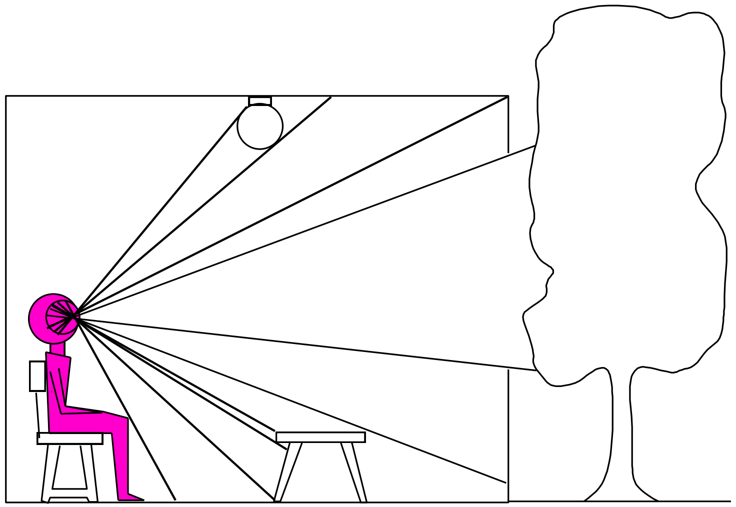
OpenCV

Documentation and tutorials

1. Pinhole camera model, projective geometry, and camera calibration

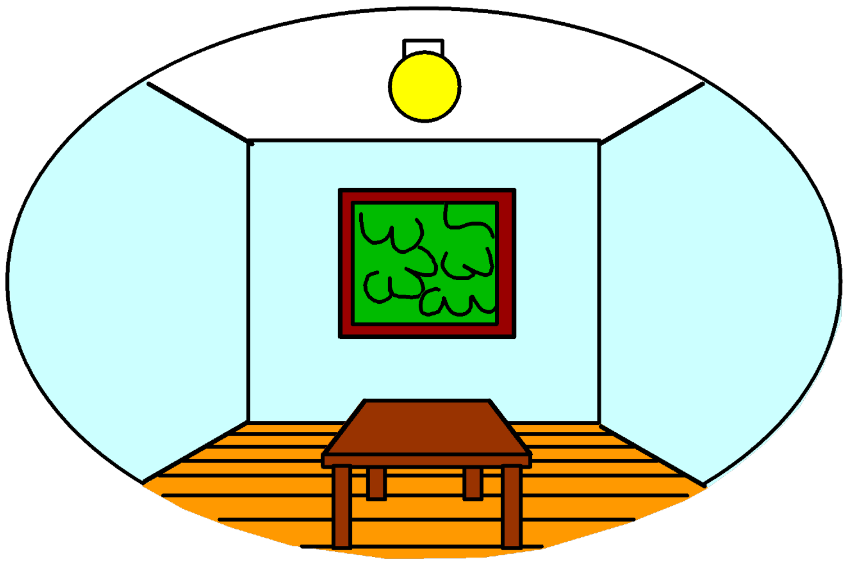
How does a camera "see" the world?

3D world



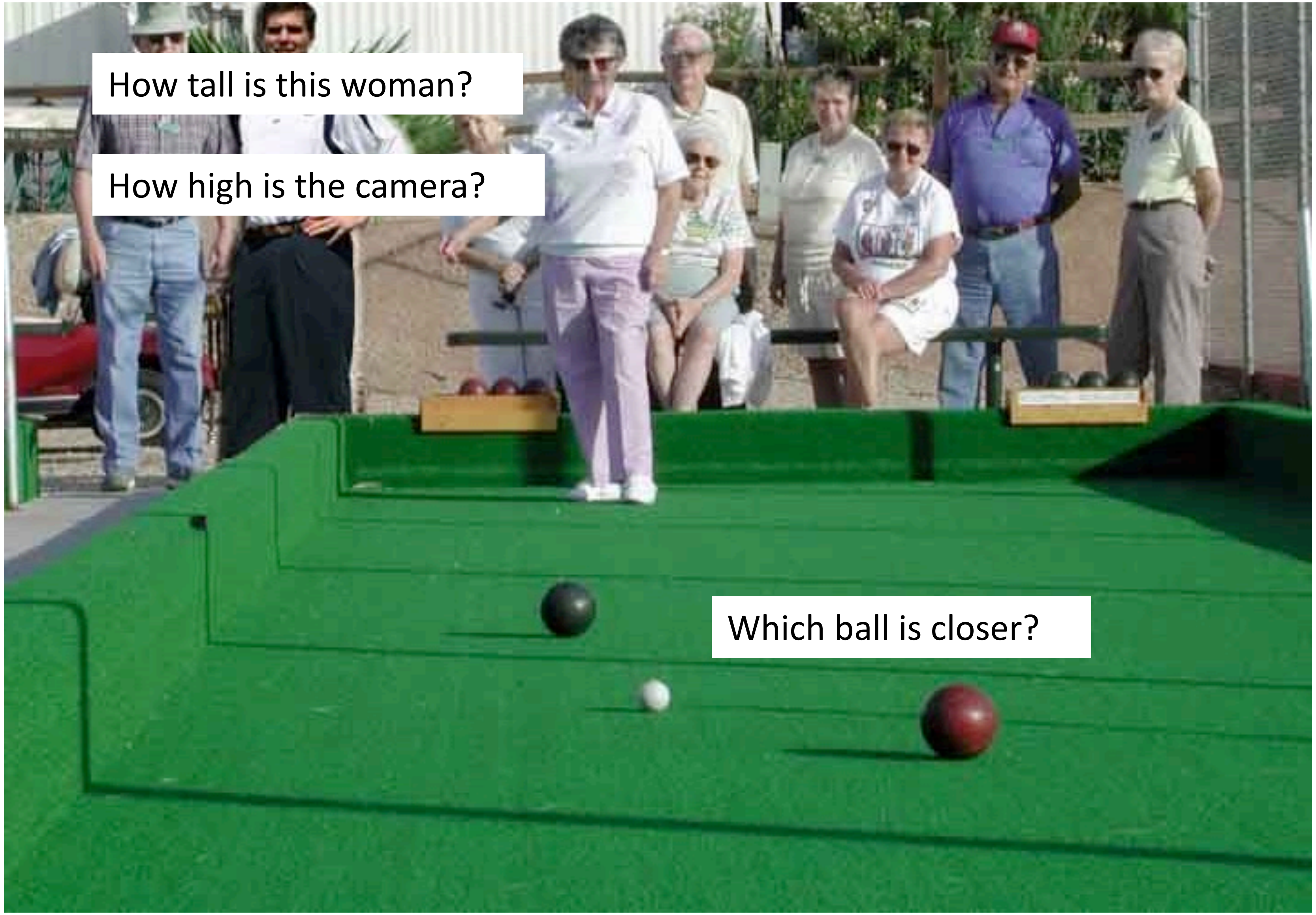
Point of observation

2D image



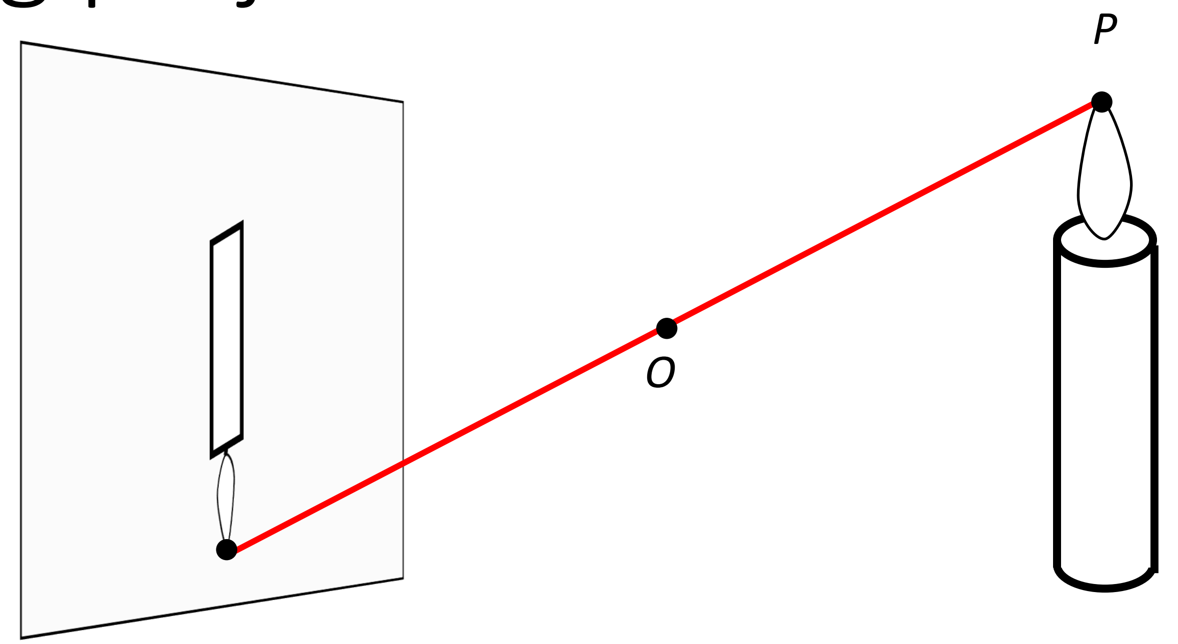
Figures © Stephen E. Palmer, 2002

How does a camera “see” the world?



Source: S. Lazebnik

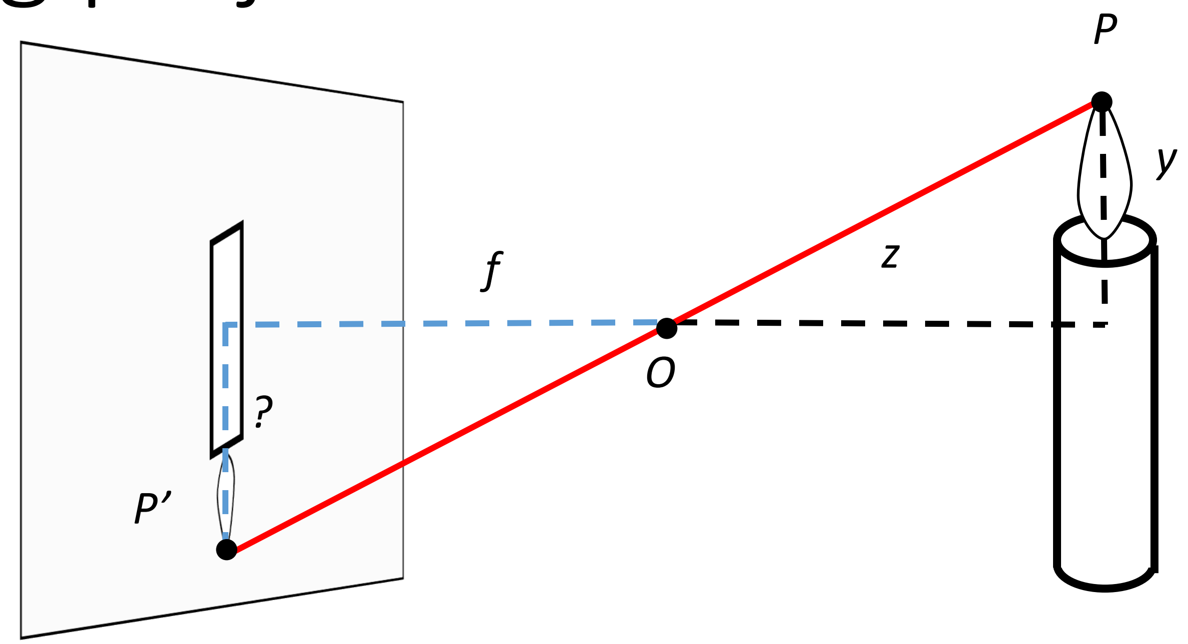
Modeling projection



- To compute the projection *P’* of a scene point *P*, form the **visual ray** connecting *P* to the camera center *O* and find where it intersects the image plane
  - All scene points that lie on this visual ray have the same projection in the image

Source: S. Lazebnik

Modeling projection



The coordinate system

- The optical center (***O***) is at the origin
- The image plane is parallel to xy-plane or perpendicular to the z-axis, which is the *optical axis*

Projection equations

- Derived using similar triangles

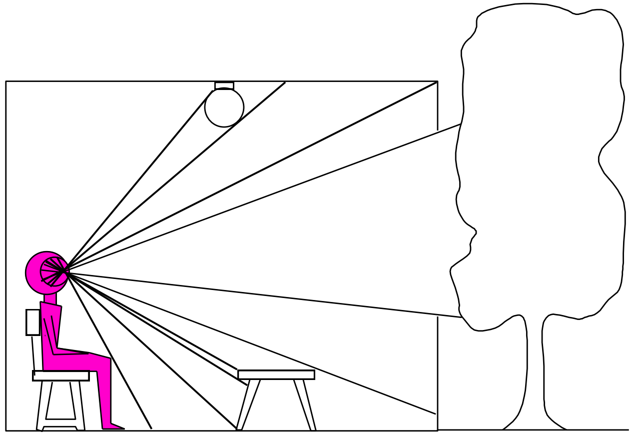
$$(x, y, z) \rightarrow (f\frac{x}{z}, f\frac{y}{z})$$

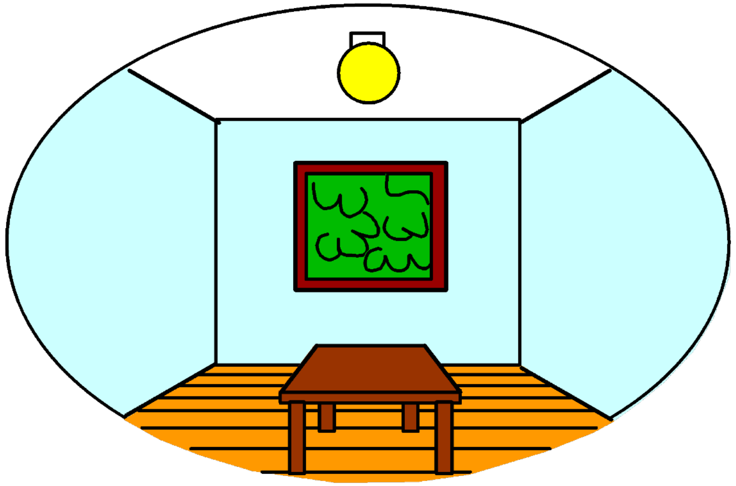
Source: S. Lazebnik

Fronto-parallel planes

What happens to the projection of a pattern on a plane parallel to the image plane?

- All points on that plane are at a fixed *depth z*
- The pattern gets scaled by a factor of *f / z*, but angles and ratios of lengths/areas are preserved





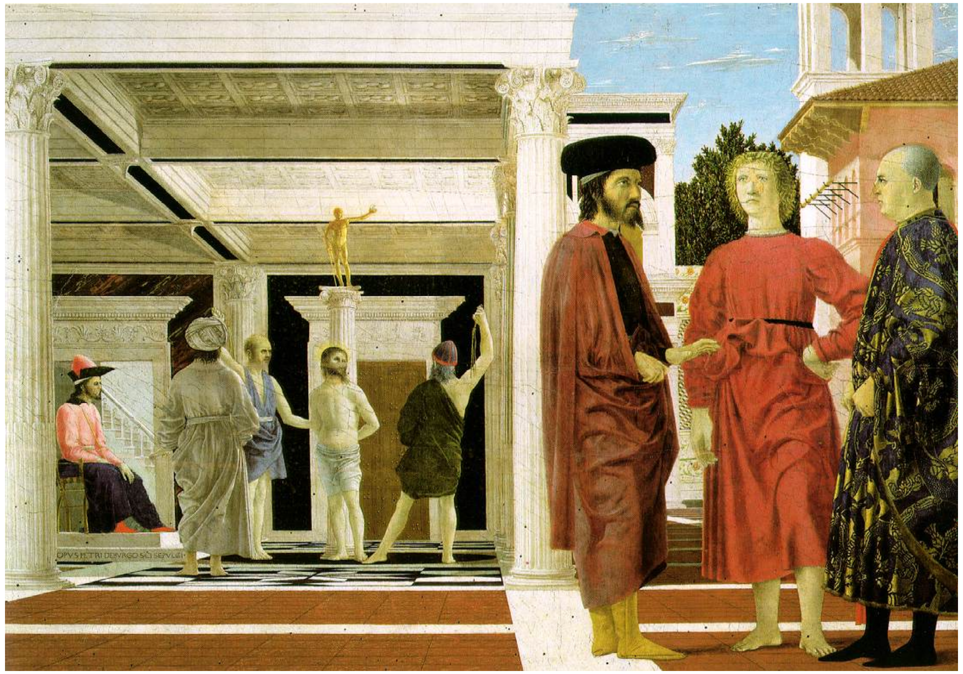
$$(x, y, z) \rightarrow (f\frac{x}{z}, f\frac{y}{z})$$

Source: S. Lazebnik

Fronto-parallel planes

What happens to the projection of a pattern on a plane parallel to the image plane?

- All points on that plane are at a fixed *depth z*
- The pattern gets scaled by a factor of *f / z*, but angles and ratios of lengths/areas are preserved

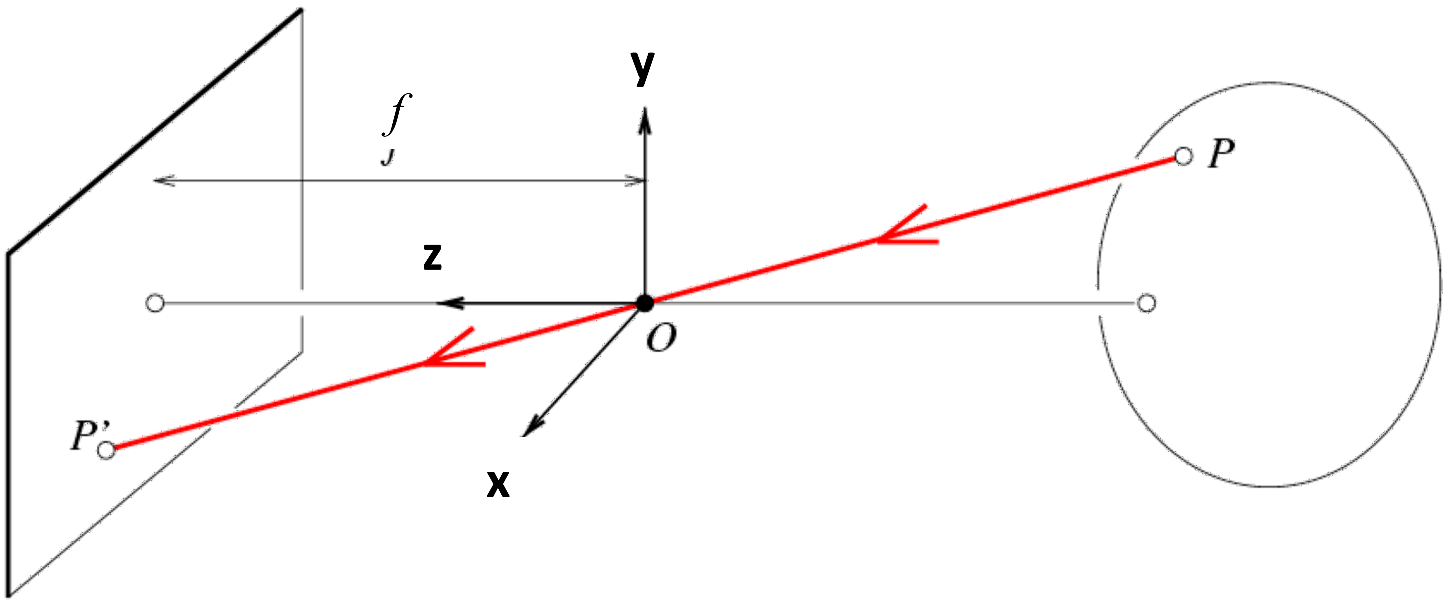


Piero della Francesca, *Flagellation of Christ*, 1455-1460

Jan Vermeer, *The Music Lesson*, 1662-1665

Source: S. Lazebnik

Modeling projection



•Projection equation: $$(x, y, z) \rightarrow (f\frac{x}{z}, f\frac{y}{z})$$

Source: J. Ponce, S. Seitz

Homogeneous coordinates

$$(x, y, z) \to (f\frac{x}{z}, f\frac{y}{z})$$

- Is this a linear transformation?
  - no—division by z is nonlinear

Trick: add one more coordinate:

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

homogeneous image coordinates

$$(x, y, z) \Rightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

homogeneous scene coordinates

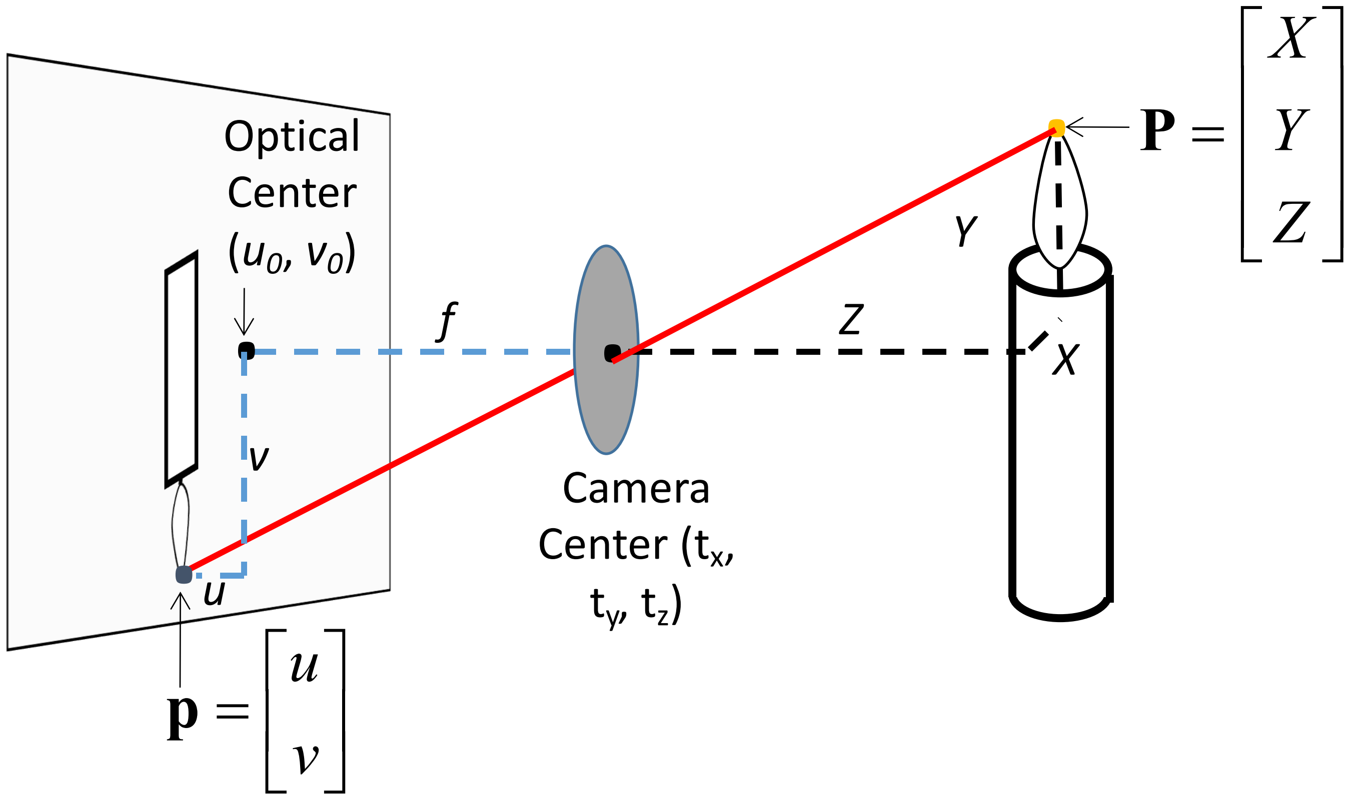
Converting *from* homogeneous coordinates

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w)$$

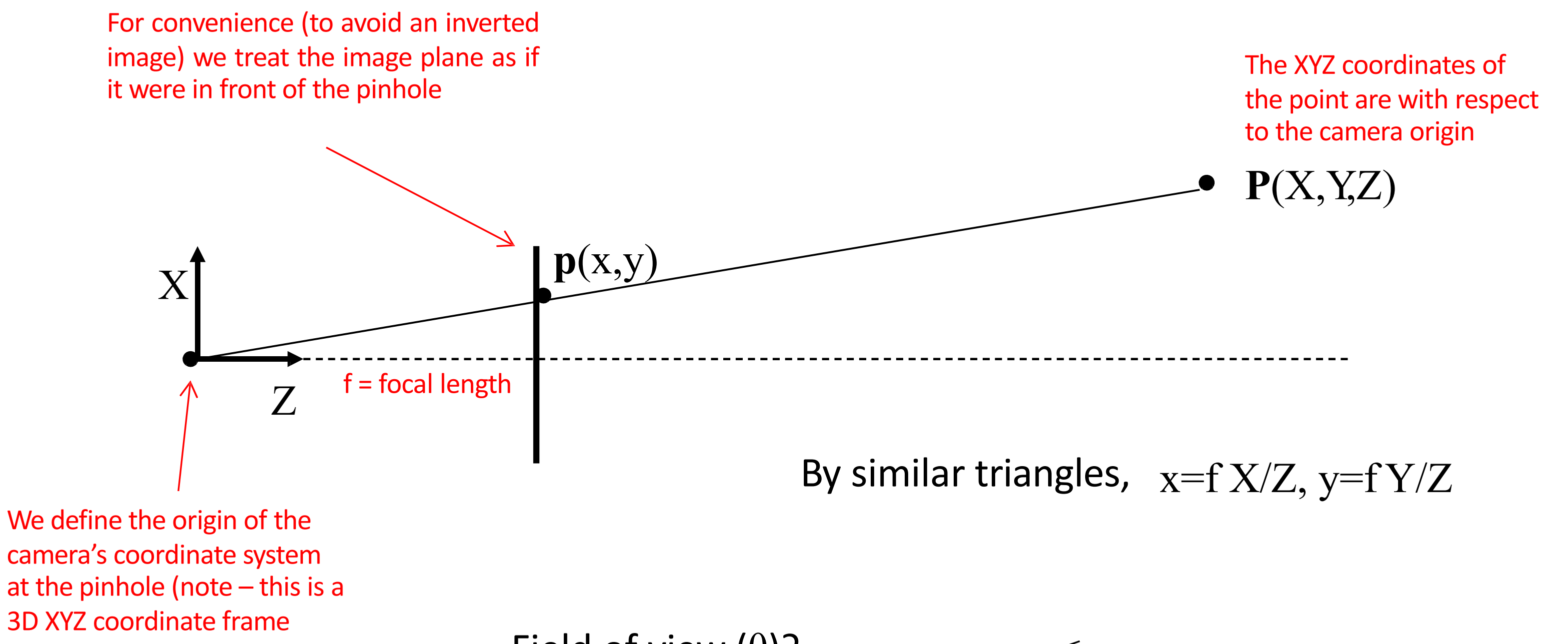
$$\begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \Rightarrow (x/w, y/w, z/w)$$

Slide by Steve Seitz

Projection: world coordinates → image coordinates

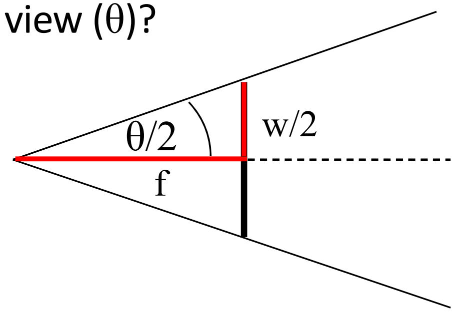


Perspective Projection Equations



By similar triangles, $x = f\,X/Z,\ y = f\,Y/Z$

Field of view (θ)?

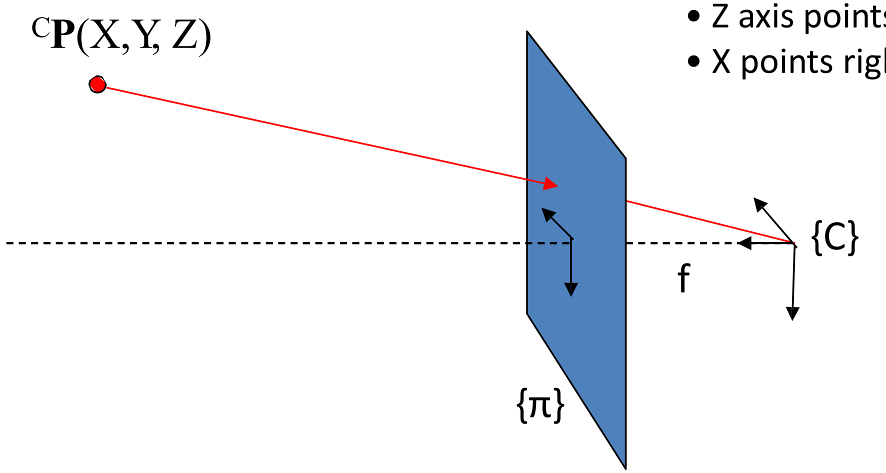


$\tan(\theta / 2) = (w/2)/f$

Camera vs Image Plane Coords

Camera coordinate system {C}

- A 3D coordinate system (X,Y,Z) – units say, in meters
- Origin at the center of projection
- Z axis points outward along optical axis
- X points right, Y points down



$x = f\,X/Z$

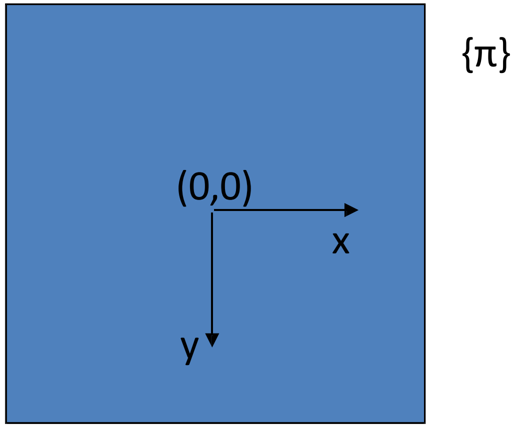
$y = f\,Y/Z$

Image plane coordinate system {π}

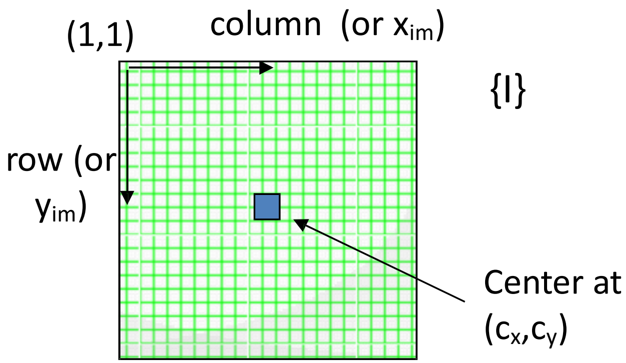
- A 2D coordinate system (x,y) – units in mm
- Origin at the intersection of the optical axis with the image plane
- In real systems, this is where the CCD or CMOS plane is

Image Buffer

- Image plane
  - The real image is formed on the CCD plane
  - (x,y) units in mm
  - Origin in center (principal point)



- Image buffer
  - Digital (or pixel) image
  - (row, col) indices
  - We can also use (x_{im}, y_{im})
  - Origin in upper left



Homogeneous coordinates

Invariant to scaling

$$k\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} kx \\ ky \\ kw \end{bmatrix} \Rightarrow \begin{bmatrix} \frac{kx}{kw} \\ \frac{ky}{kw} \end{bmatrix} = \begin{bmatrix} \frac{x}{w} \\ \frac{y}{w} \end{bmatrix}$$

Homogeneous Coordinates

Cartesian Coordinates

Point in Cartesian is ray in Homogeneous

Basic geometry in homogeneous coordinates

- Line equation: ax + by + c = 0

$$line_i = \begin{bmatrix} a_i \\ b_i \\ c_i \end{bmatrix}$$

- Append 1 to pixel coordinate to get homogeneous coordinate

$$p_i = \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix}$$

- Line given by cross product of two points

$$line_{ij} = p_i \times p_j$$

- Intersection of two lines given by cross product of the lines

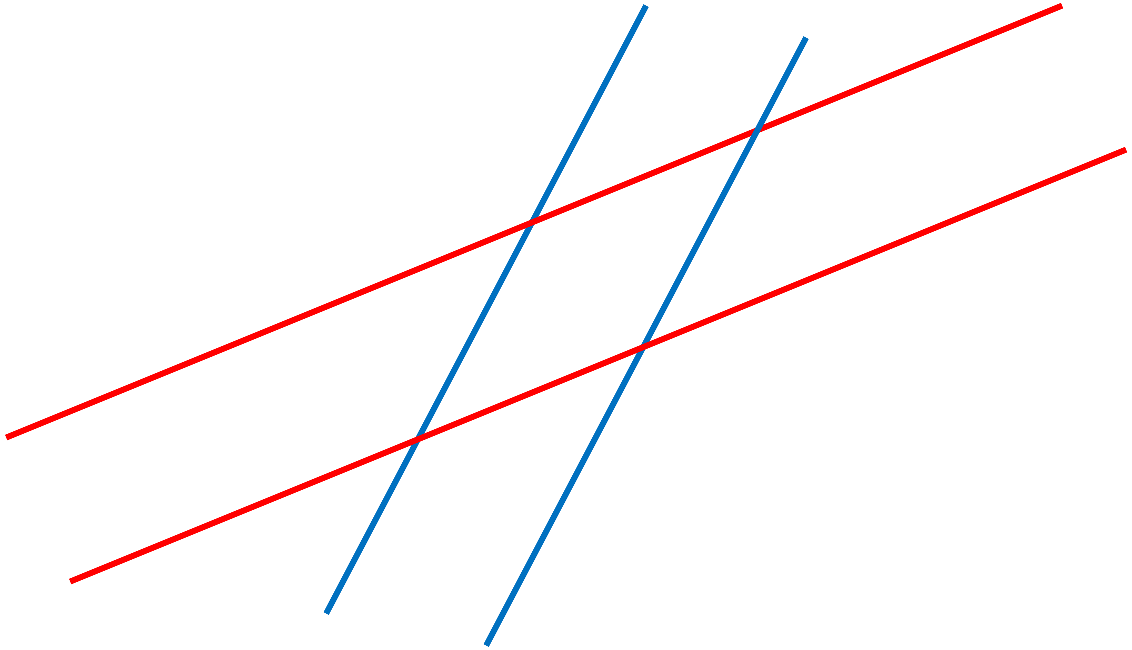
$$q_{ij} = line_i \times line_j$$

Another problem solved by homogeneous coordinates

Intersection of parallel lines

Cartesian: (Inf, Inf)
Homogeneous: (1, 1, 0)

Cartesian: (Inf, Inf)
Homogeneous: (1, 2, 0)



Perspective Projection Matrix

- Projection is a matrix multiplication using homogeneous coordinates

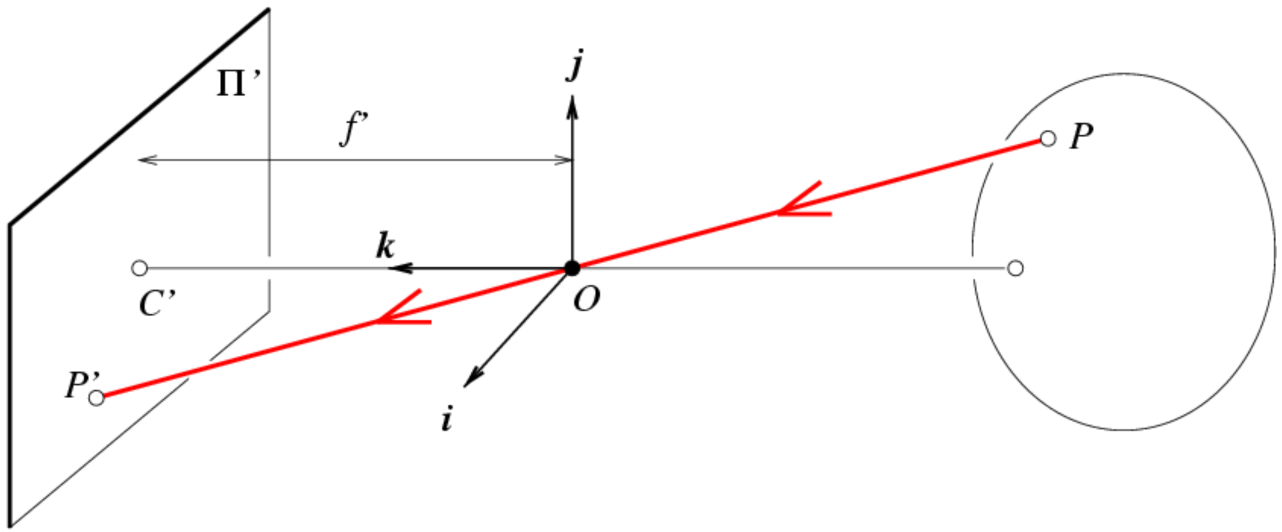
$$\begin{bmatrix} & & & \\ & & & \\ & & & \end{bmatrix}\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} \\ \\ \\ \end{bmatrix} \Rightarrow (f\frac{x}{z}, f\frac{y}{z})$$

divide by the third coordinate

In practice: lots of coordinate transformations…

2D point (3x1)	=	Camera to pixel coord. trans. matrix (3x3)	Perspective projection matrix (3x4)	World to camera coord. trans. matrix (4x4)	3D point (4x1)

Projection matrix



Intrinsic Assumptions

- Unit aspect ratio
- Optical center at (0,0)
- No skew

Extrinsic Assumptions

- No rotation
- Camera at (0,0,0)

K

$$\mathbf{x} = \mathbf{K}\begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\mathbf{X} \Rightarrow w\begin{bmatrix}u\\v\\1\end{bmatrix} = \begin{bmatrix}f & 0 & 0 & 0\\0 & f & 0 & 0\\0 & 0 & 1 & 0\end{bmatrix}\begin{bmatrix}x\\y\\z\\1\end{bmatrix}$$

Slide Credit: Saverese

Remove assumption: known optical center

Intrinsic Assumptions

- Unit aspect ratio
- No skew

Extrinsic Assumptions

- No rotation
- Camera at (0,0,0)

$$\mathbf{x} = \mathbf{K}\begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\mathbf{X} \quad \Rightarrow \quad w\begin{bmatrix}u \\ v \\ 1\end{bmatrix} = \begin{bmatrix}f & 0 & u_0 & 0 \\ 0 & f & v_0 & 0 \\ 0 & 0 & 1 & 0\end{bmatrix}\begin{bmatrix}x \\ y \\ z \\ 1\end{bmatrix}$$

Remove assumption: square pixels

Intrinsic Assumptions

- No skew

Extrinsic Assumptions

- No rotation
- Camera at (0,0,0)

$$\mathbf{x} = \mathbf{K}\begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\mathbf{X} \Rightarrow w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & 0 & u_0 & 0 \\ 0 & \beta & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Remove assumption: non-skewed pixels

Intrinsic Assumptions

Extrinsic Assumptions

- No rotation
- Camera at (0,0,0)

$$\mathbf{x} = \mathbf{K}\begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\mathbf{X} \Rightarrow w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & s & u_0 & 0 \\ 0 & \beta & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Note: different books use different notation for parameters

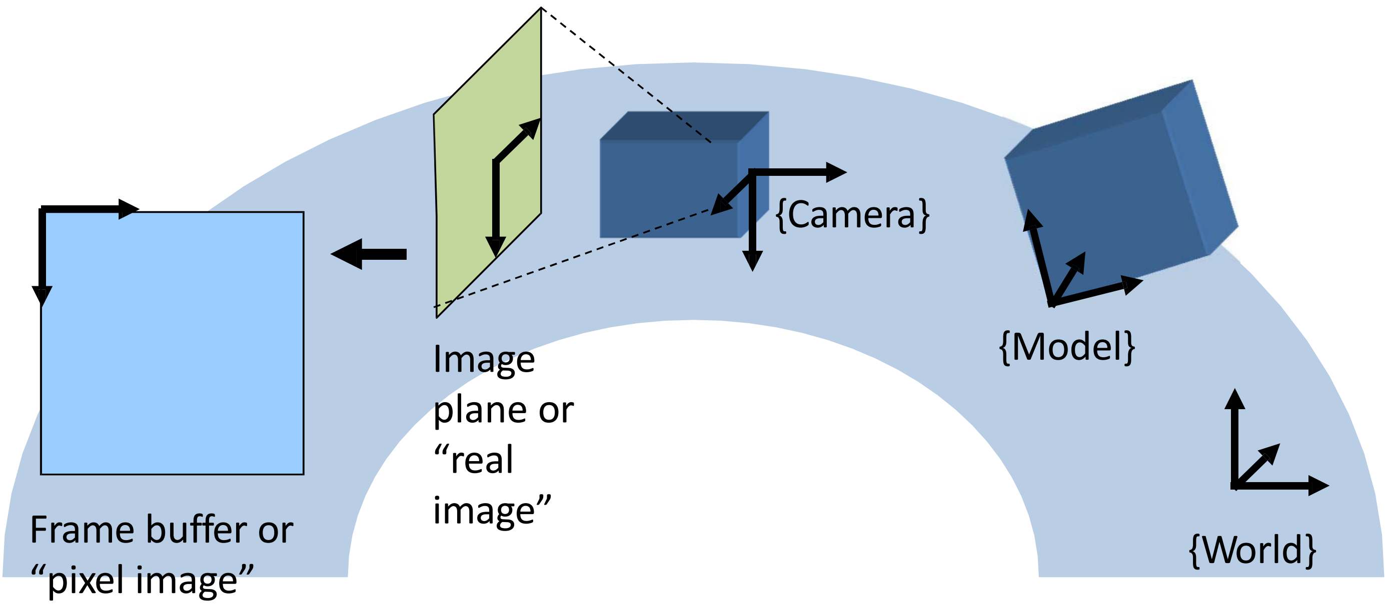
Degrees of freedom

$$\mathbf{x} = \mathbf{K}\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix}\mathbf{X}$$

$$w\begin{bmatrix}u\\ v\\ 1\end{bmatrix} = \overset{5}{\begin{bmatrix}\alpha & s & u_0\\ 0 & \beta & v_0\\ 0 & 0 & 1\end{bmatrix}}\overset{6}{\begin{bmatrix}r_{11} & r_{12} & r_{13} & t_x\\ r_{21} & r_{22} & r_{23} & t_y\\ r_{31} & r_{32} & r_{33} & t_z\end{bmatrix}}\begin{bmatrix}x\\ y\\ z\\ 1\end{bmatrix}$$

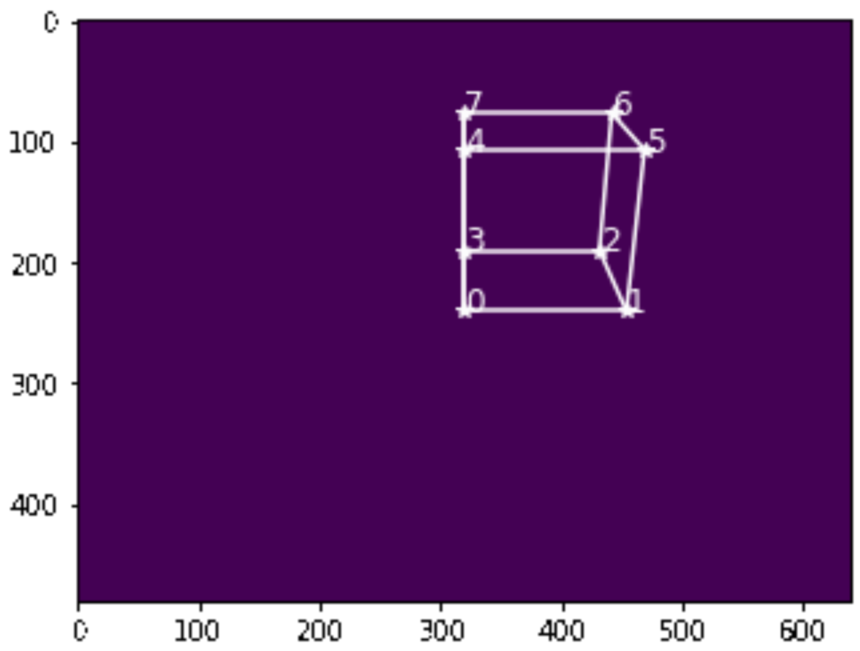
Frames of Reference

- Image frames are 2D; others are 3D
- The “pose” (position and orientation) of a 3D rigid body has 6 degrees of freedom



Example 1: the wireframe image of a cube

- In this example, we will be exploring how to estimate the camera matrix and project world points from a known object.



https://github.com/opi-lab/stsiva-workshop/blob/main/notebooks/stsiva_workshop_notebook01.ipynb

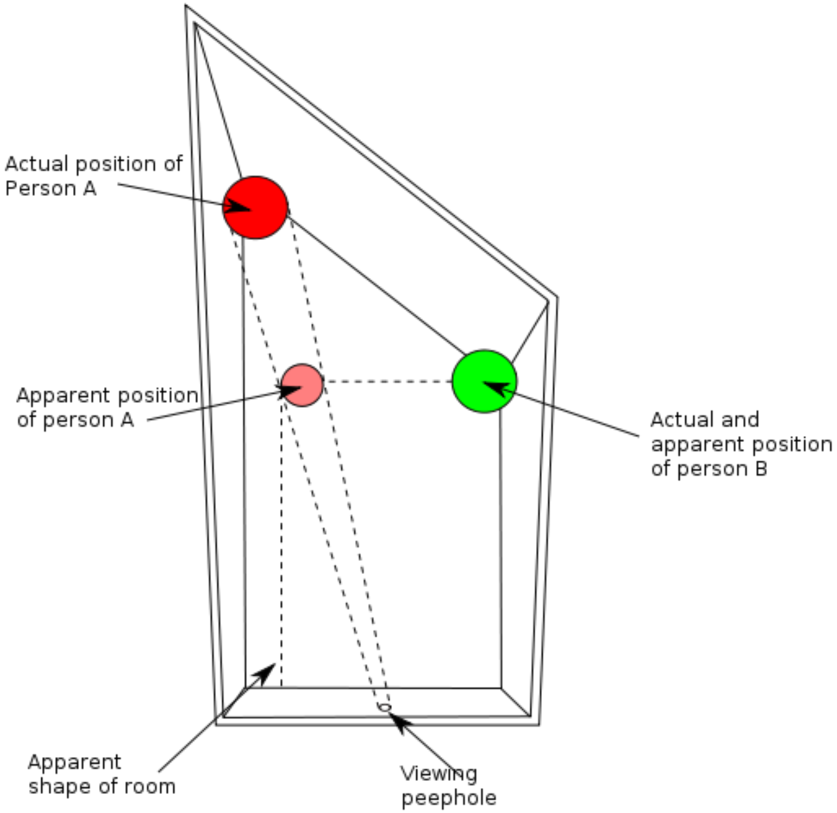
Our goal: Recovery of 3D structure

J. Vermeer, *Music Lesson*, 1662

A. Criminisi, M. Kemp, and A. Zisserman, Bringing Pictorial Space to Life: computer techniques for the analysis of paintings, *Proc. Computers and the History of Art*, 2002

Source: S. Lazebnik

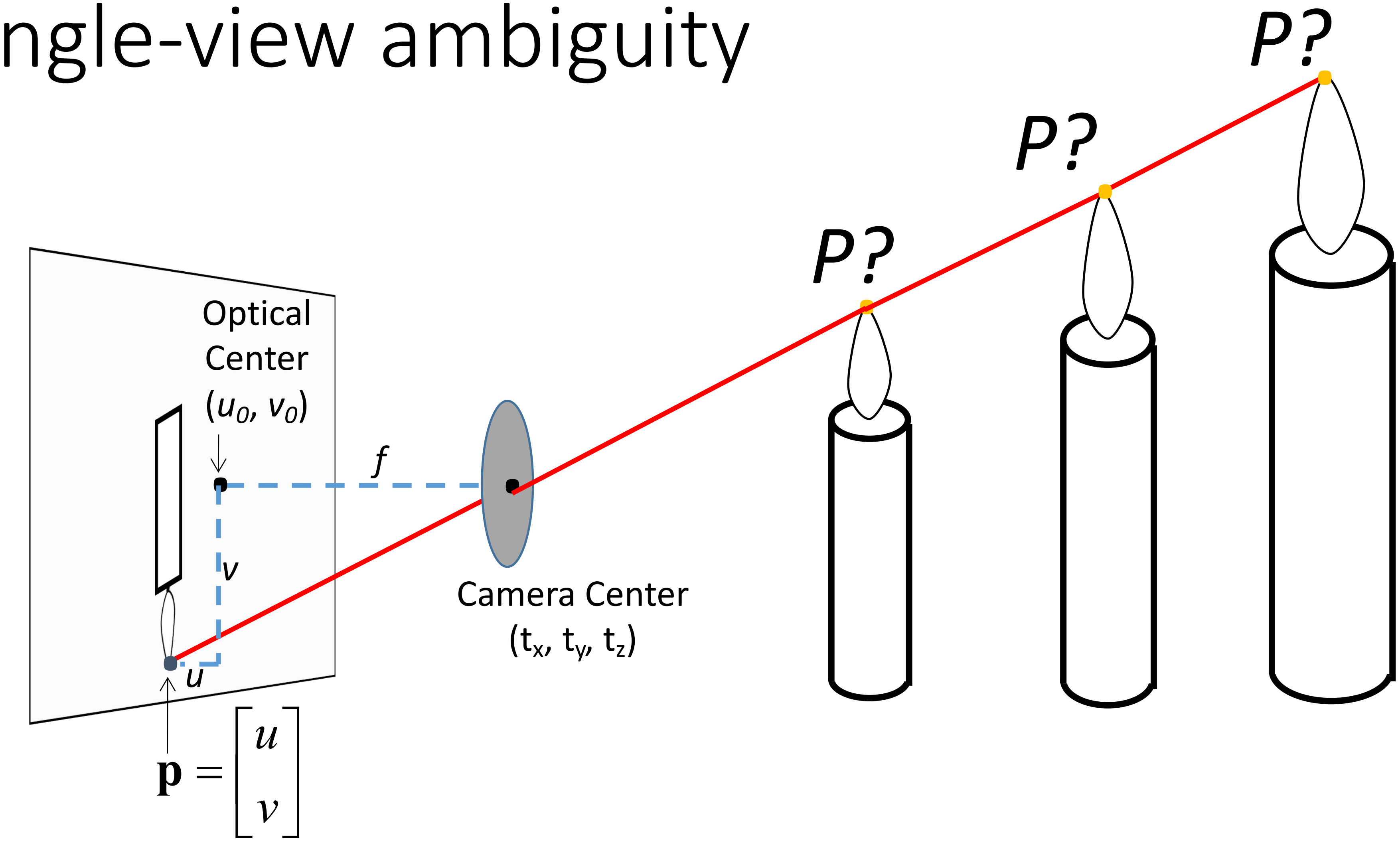
Things aren't always as they appear…



http://en.wikipedia.org/wiki/Ames_room

Source: S. Lazebnik

Single-view ambiguity



Single-view ambiguity

Source: S. Lazebnik

Our goal: Recovery of 3D structure

- When certain assumptions hold, we can recover structure from a single view

- In general, we need *multi-view geometry*

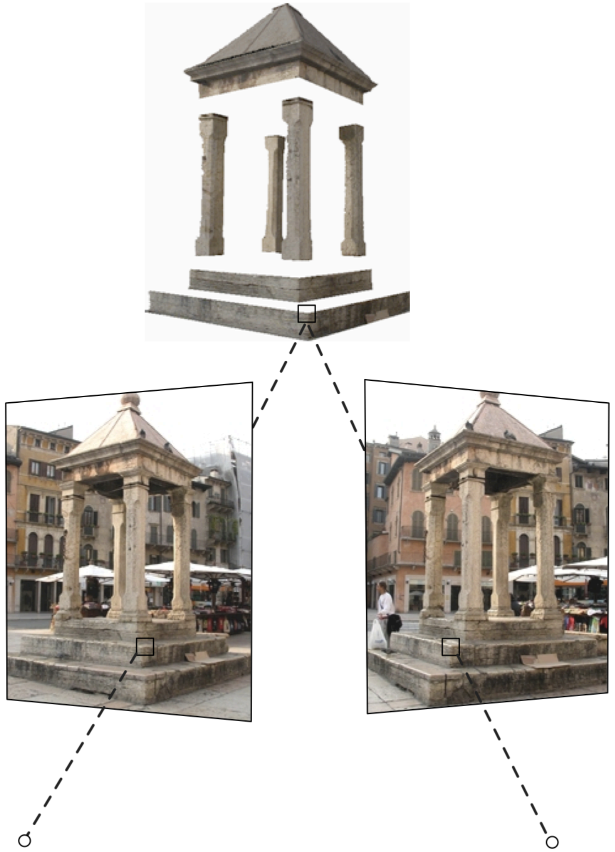
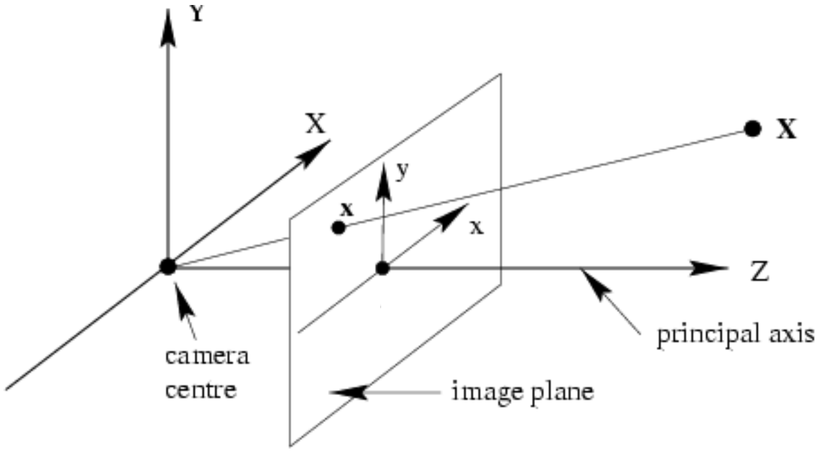


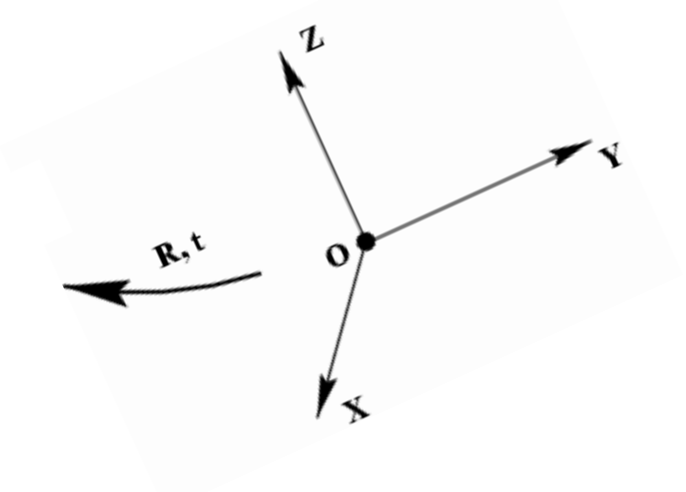
Image source

But first, we need to understand the geometry of a single camera…

Camera calibration



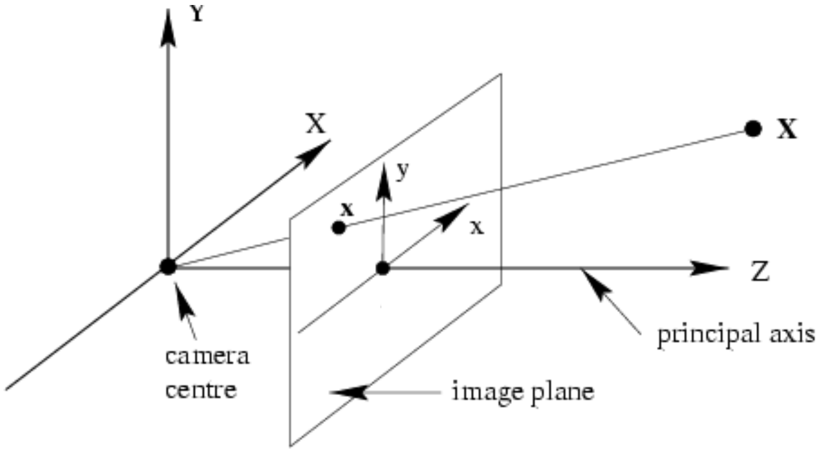
world coordinate system

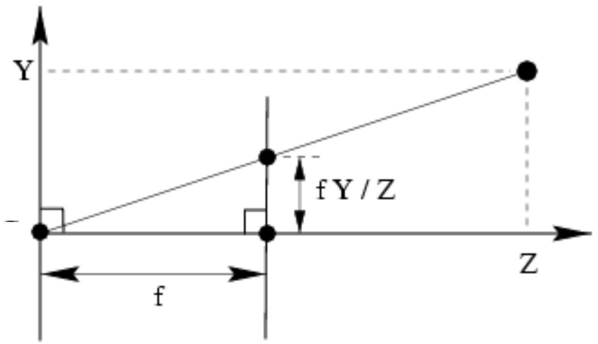


- **Normalized (camera) coordinate system:** camera center is at the origin, the *principal axis* is the z-axis,
  x and y axes of the image plane are parallel to x and y axes of the world

- **Camera calibration:** figuring out transformation from *world* coordinate system to *image* coordinate system

Pinhole camera model

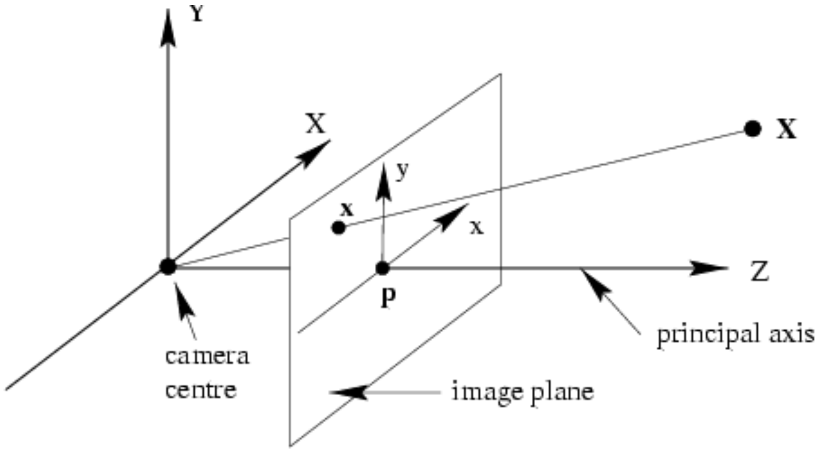


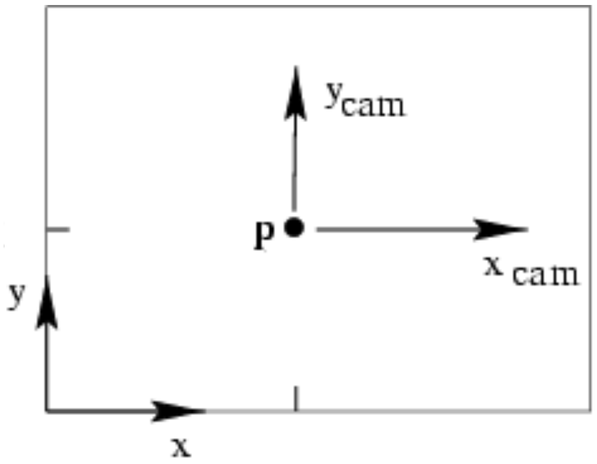


$$(X,Y,Z) \mapsto (f\,X/Z, f\,Y/Z)$$

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} f\,X \\ f\,Y \\ Z \end{pmatrix} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \qquad \lambda \mathrm{x} = \mathrm{PX}$$

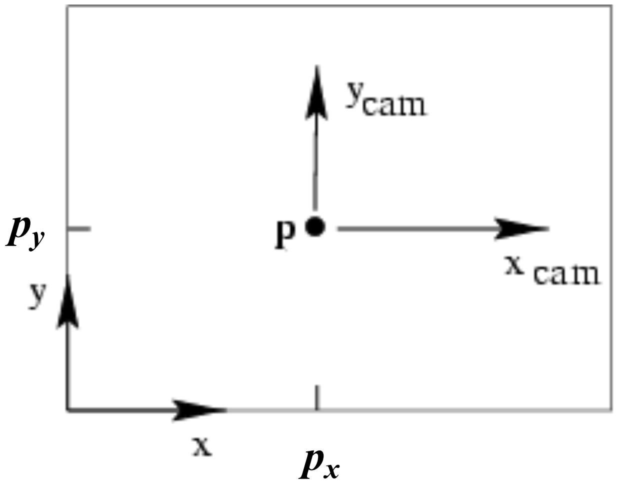
Principal point





- **Principal point (p):** point where principal axis intersects the image plane
- Normalized coordinate system: origin of the image is at the principal point
- Image coordinate system: origin is in the corner

Principal point offset

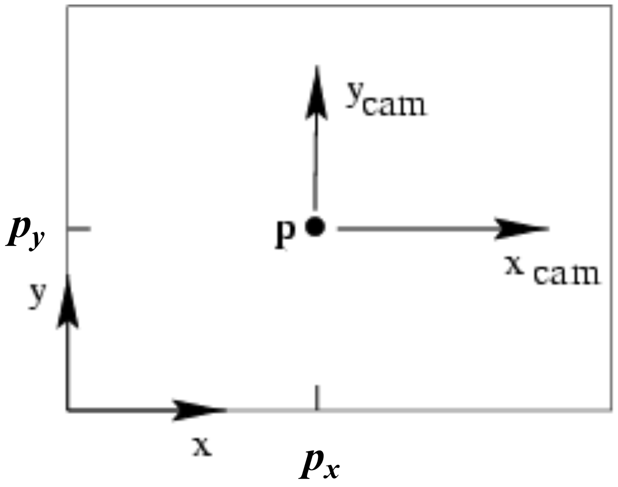


We want the principal point to map to (p_x, p_y) instead of (0,0)

$$(X,Y,Z) \mapsto (fX/Z \boxed{+ p_x}, fY/Z \boxed{+ p_y})$$

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX \boxed{+Zp_x} \\ fY \boxed{+Zp_y} \\ Z \end{pmatrix} = \begin{bmatrix} f & & \boxed{p_x} & 0 \\ & f & \boxed{p_y} & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

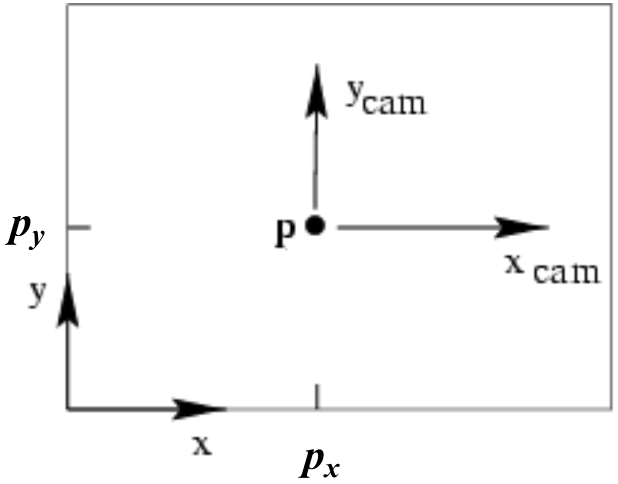
Principal point offset



principal point: (p_x, p_y)

$$\begin{bmatrix} f & & p_x & 0 \\ & f & p_y & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

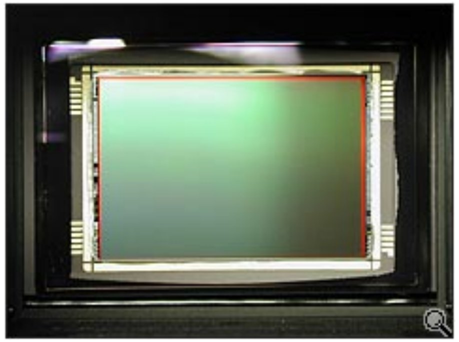
Principal point offset



principal point: (p_x, p_y)

$$\underbrace{\underset{\substack{\text{calibration matrix}\\ \mathrm{K}}}{\begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix}}\underset{\substack{\text{projection matrix}\\ [\mathrm{I} \mid 0]}}{\begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix}}}_{\mathrm{P} = \mathrm{K}[\mathrm{I} \mid 0]}\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{bmatrix} f & & p_x & 0 \\ & f & p_y & 0 \\ & & 1 & 0 \end{bmatrix}\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Pixel coordinates

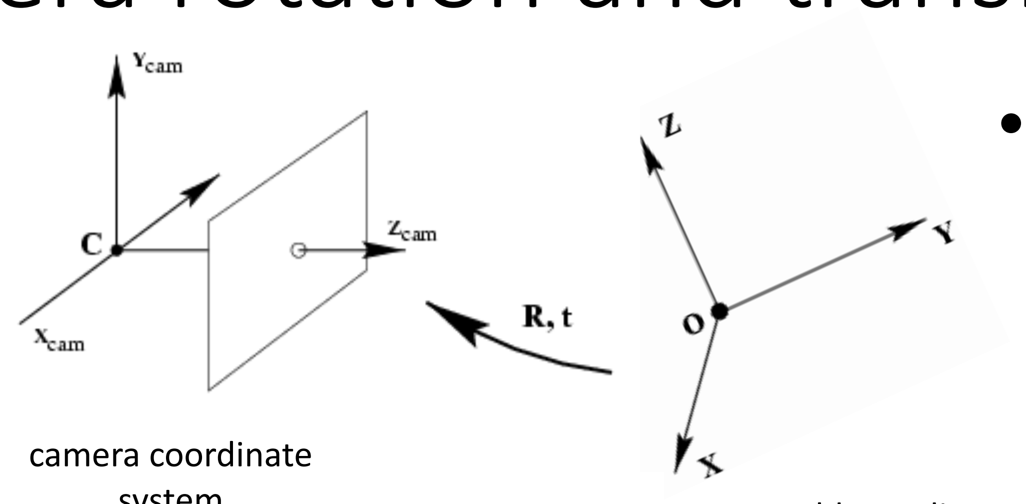


Pixel size: $$\frac{1}{m_x} \times \frac{1}{m_y}$$

- m_x pixels per meter in horizontal direction,
  m_y pixels per meter in vertical direction

$$K = \underbrace{\begin{bmatrix} m_x & & \\ & m_y & \\ & & 1 \end{bmatrix}}_{\text{pixels/m}} \underbrace{\begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix}}_{\text{m}} = \underbrace{\begin{bmatrix} \alpha_x & & \beta_x \\ & \alpha_y & \beta_y \\ & & 1 \end{bmatrix}}_{\text{pixels}}$$

Camera rotation and translation



camera coordinate system

world coordinate system

- In general, the *camera* coordinate frame will be related to the *world* coordinate frame by a rotation and a translation

- Conversion from world to camera coordinate system (in non-homogeneous coordinates):

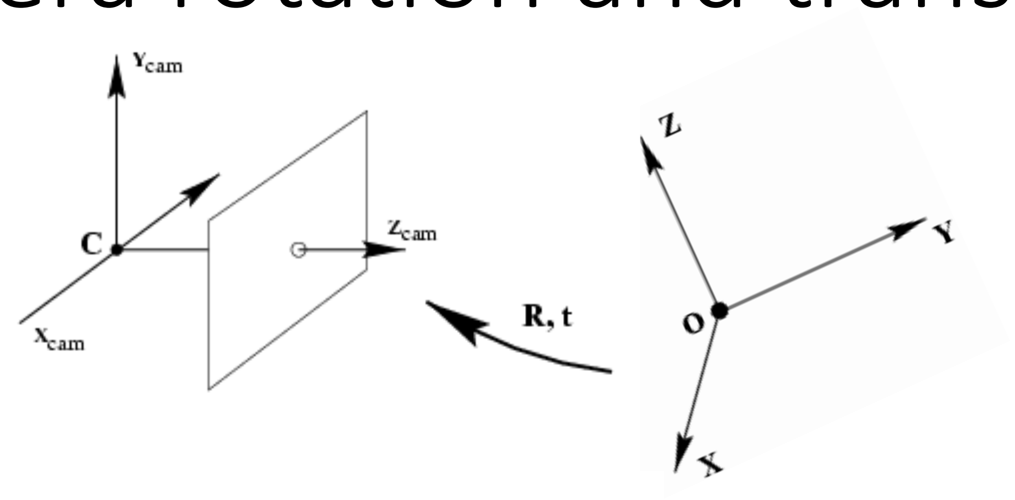
$$\tilde{X}_{cam} = R\left(\tilde{X} - \tilde{C}\right)$$

coords. of point in camera frame

coords. of a point in world frame

coords. of camera center in world frame

Camera rotation and translation

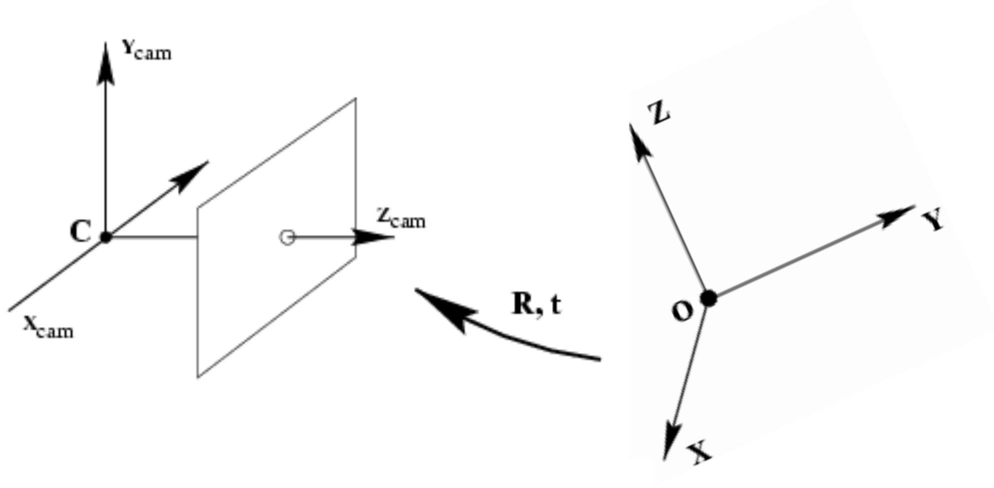


$$\tilde{X}_{cam} = R\left(\tilde{X} - \tilde{C}\right)$$

$$\begin{pmatrix} \tilde{X}_{cam} \\ 1 \end{pmatrix} = \begin{bmatrix} R & -R\tilde{C} \\ 0 & 1 \end{bmatrix} \begin{pmatrix} \tilde{X} \\ 1 \end{pmatrix}$$

3D transformation
matrix (4 x 4)

Camera rotation and translation

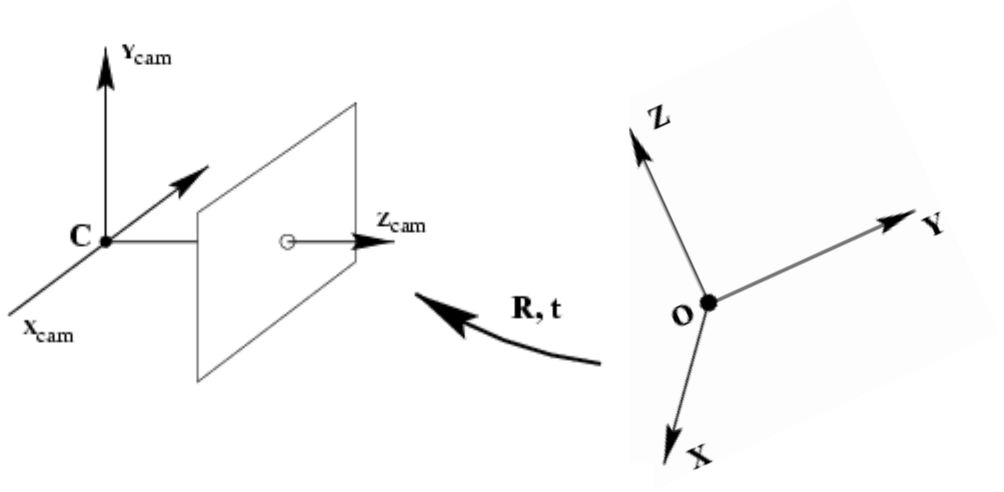


$$\widetilde{X}_{cam} = R\left(\widetilde{X} - \widetilde{C}\right)$$

$$X_{cam} = \begin{bmatrix} R & -R\widetilde{C} \\ 0 & 1 \end{bmatrix} X$$

3D transformation
matrix (4 x 4)

Camera rotation and translation



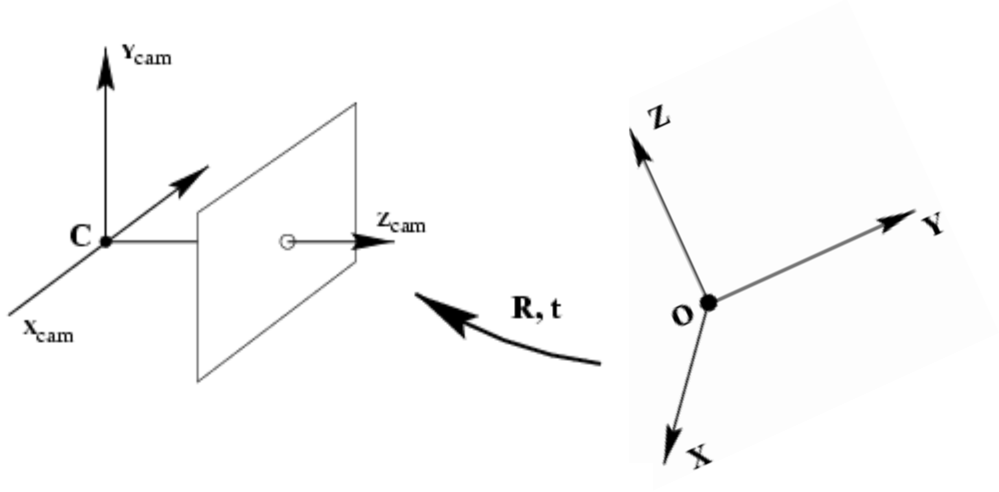
$$\mathbf{x} = \mathbf{K}\left[\mathbf{I} \mid \mathbf{0}\right]\begin{bmatrix} \mathbf{R} & -\mathbf{R}\widetilde{\mathbf{C}} \\ \mathbf{0} & 1 \end{bmatrix}\mathbf{X}$$

2D transformation matrix (3 x 3)

perspective projection matrix (3 x 4)

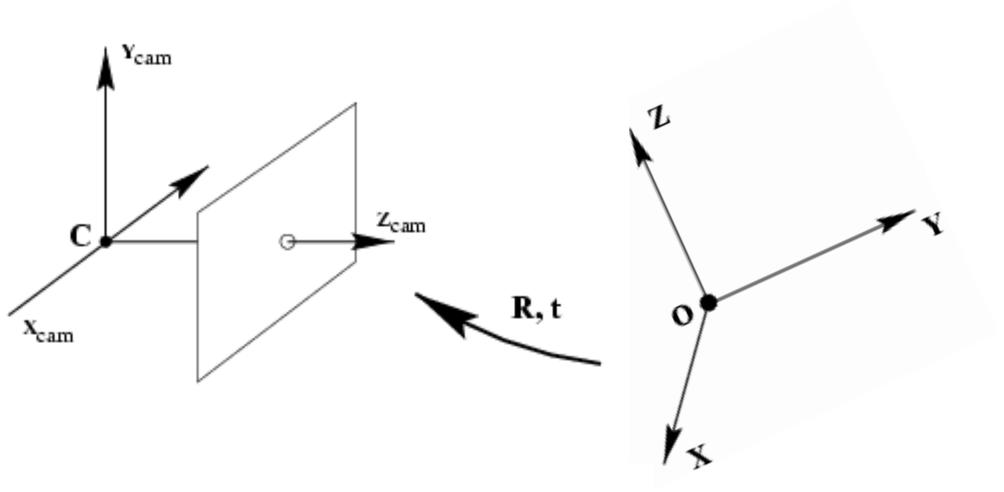
3D transformation matrix (4 x 4)

Camera rotation and translation



$$\mathbf{x} = \mathbf{K}\left[\mathbf{R} \mid -\mathbf{R}\tilde{\mathbf{C}}\right]\mathbf{X}$$

Camera rotation and translation

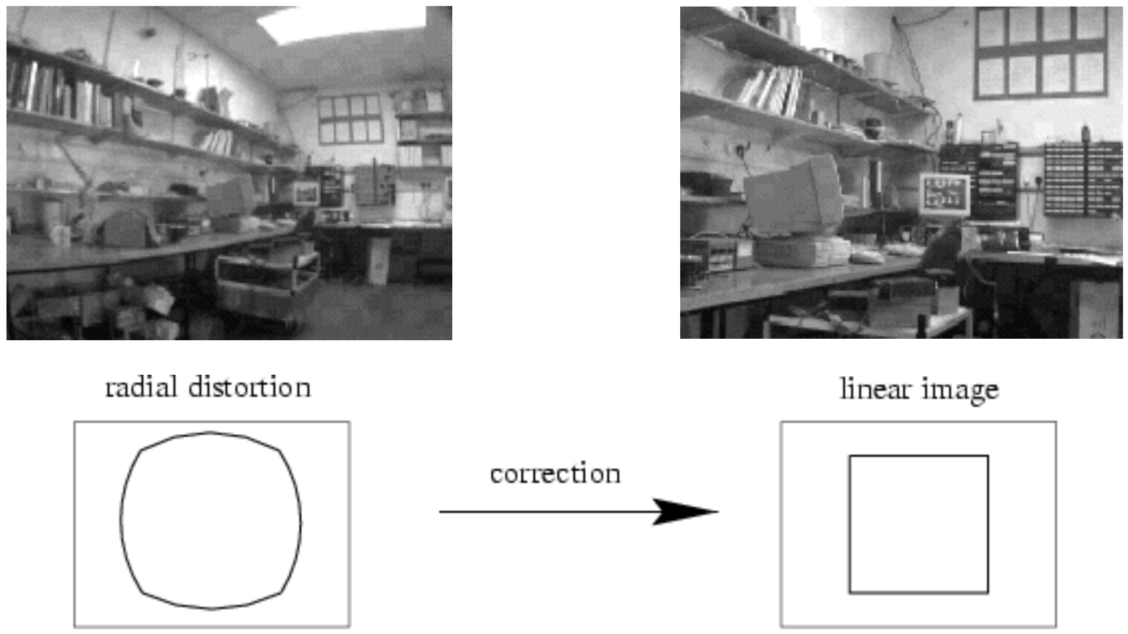


$$x = K[R \mid t]X \qquad t = -R\tilde{C}$$

Camera parameters $\quad \mathbf{P} = \mathbf{K}\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix}$

- Intrinsic parameters
  - Principal point coordinates
  - Focal length
  - Pixel magnification factors
  - *Skew (non-rectangular pixels)*
  - *Radial distortion*

$$\mathbf{K} = \begin{bmatrix} m_x & & \\ & m_y & \\ & & 1 \end{bmatrix} \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix} = \begin{bmatrix} \alpha_x & & \beta_x \\ & \alpha_y & \beta_y \\ & & 1 \end{bmatrix}$$



Camera parameters $\mathbf{P} = \mathbf{K}[\mathbf{R} \quad \mathbf{t}]$

- Intrinsic parameters
  - Principal point coordinates
  - Focal length
  - Pixel magnification factors
  - *Skew (non-rectangular pixels)*
  - *Radial distortion*
- Extrinsic parameters
  - Rotation and translation relative to world coordinate system

$$\mathbf{P} = \mathbf{K}[\mathbf{R} \quad -\mathbf{R}\underbrace{\tilde{\mathbf{C}}}_{\text{coords. of camera center in world frame}}]$$

  - What is the projection of the camera center?

$$\mathbf{PC} = \mathbf{K}[\mathbf{R} \quad -\mathbf{R}\tilde{\mathbf{C}}]\begin{bmatrix} \tilde{\mathbf{C}} \\ 1 \end{bmatrix} = 0$$

The camera center is the *null space* of the projection matrix!

Camera calibration

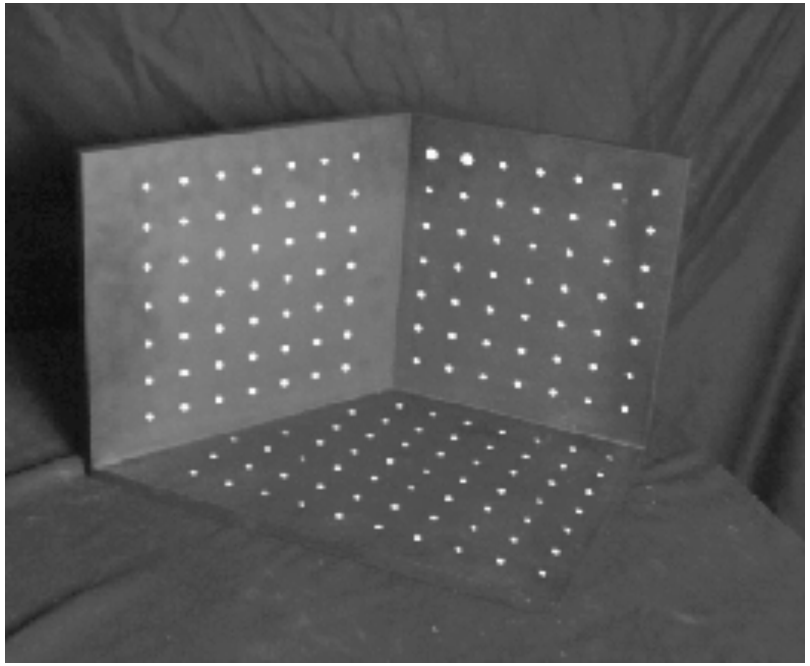
$$\lambda \mathbf{x} = \mathbf{K}\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix}\mathbf{X}$$

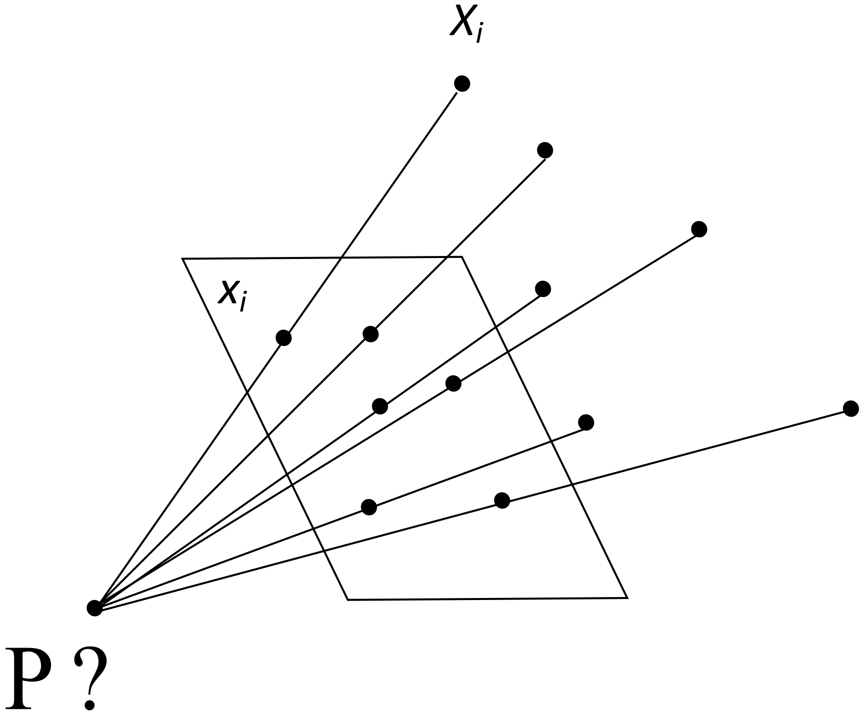
$$\begin{bmatrix} \lambda x \\ \lambda y \\ \lambda \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Source: D. Hoiem

Camera calibration

- Given *n* points with known 3D coordinates $\boldsymbol{X}_i$ and known image projections $\boldsymbol{x}_i$, estimate the camera parameters





Camera calibration: Linear method

$$\lambda \mathbf{x}_i = \mathbf{P}\mathbf{X}_i \qquad \mathbf{x}_i \times \mathbf{P}\mathbf{X}_i = 0 \qquad \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \times \begin{bmatrix} \mathbf{P}_1^T \mathbf{X}_i \\ \mathbf{P}_2^T \mathbf{X}_i \\ \mathbf{P}_3^T \mathbf{X}_i \end{bmatrix} = 0$$

$$\begin{bmatrix} 0 & -\mathbf{X}_i^T & y_i \mathbf{X}_i^T \\ \mathbf{X}_i^T & 0 & -x_i \mathbf{X}_i^T \\ -y_i \mathbf{X}_i^T & x_i \mathbf{X}_i^T & 0 \end{bmatrix} \begin{pmatrix} \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{pmatrix} = 0$$

Two linearly independent equations

Camera calibration: Linear method

$$\begin{bmatrix} 0^T & \mathbf{X}_1^T & -y_1\mathbf{X}_1^T \\ \mathbf{X}_1^T & 0^T & -x_1\mathbf{X}_1^T \\ \cdots & \cdots & \cdots \\ 0^T & \mathbf{X}_n^T & -y_n\mathbf{X}_n^T \\ \mathbf{X}_n^T & 0^T & -x_n\mathbf{X}_n^T \end{bmatrix} \begin{pmatrix} \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{pmatrix} = 0 \qquad \mathbf{Ap} = \mathbf{0}$$

- P has 11 degrees of freedom
- One 2D/3D correspondence gives us two linearly independent equations
  - 6 correspondences needed for a minimal solution
- Homogeneous least squares: find p minimizing $\|\mathrm{Ap}\|^2$
  - Solution given by eigenvector of $\mathrm{A}^\mathrm{T}\mathrm{A}$ with smallest eigenvalue

Camera calibration: Linear method

$$\begin{bmatrix} 0^T & \mathbf{X}_1^T & -y_1\mathbf{X}_1^T \\ \mathbf{X}_1^T & 0^T & -x_1\mathbf{X}_1^T \\ \cdots & \cdots & \cdots \\ 0^T & \mathbf{X}_n^T & -y_n\mathbf{X}_n^T \\ \mathbf{X}_n^T & 0^T & -x_n\mathbf{X}_n^T \end{bmatrix} \begin{pmatrix} \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{pmatrix} = 0 \qquad \mathbf{Ap} = \mathbf{0}$$

- Note: for coplanar points that satisfy $\mathbf{\Pi}^T\mathbf{X}=0$, we will get degenerate solutions $(\mathbf{\Pi},\mathbf{0},\mathbf{0})$, $(\mathbf{0},\mathbf{\Pi},\mathbf{0})$, or $(\mathbf{0},\mathbf{0},\mathbf{\Pi})$

Camera calibration: Linear vs. nonlinear

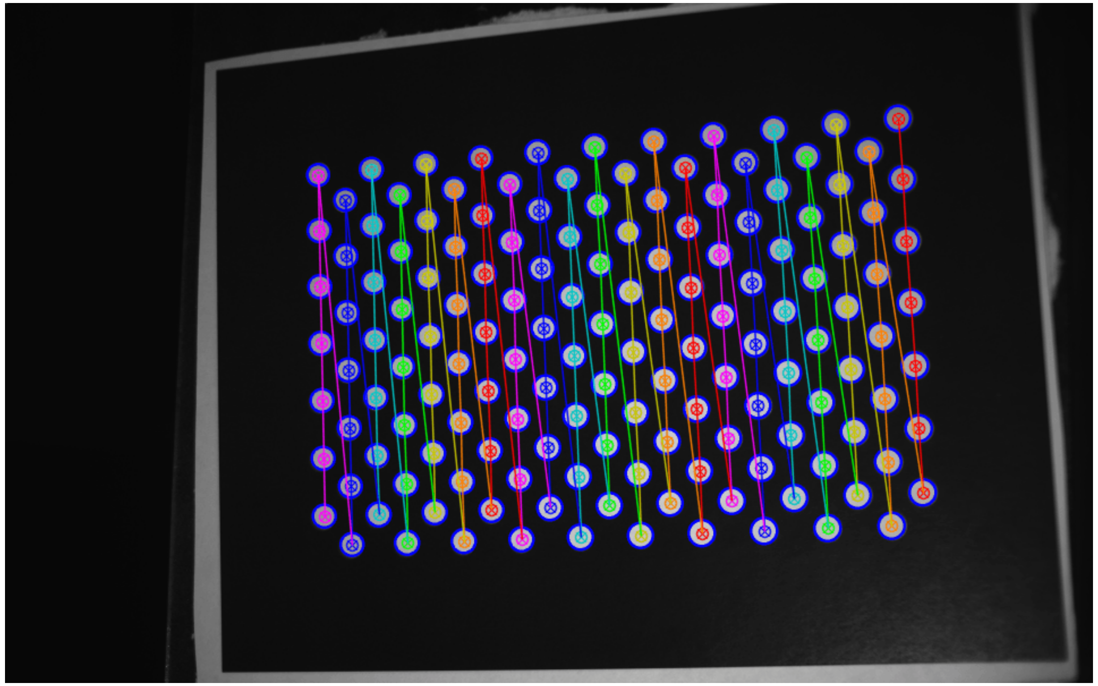
- Linear calibration is easy to formulate and solve, but it doesn't directly tell us the camera parameters

$$\begin{bmatrix} \lambda x \\ \lambda y \\ \lambda \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad \text{vs.} \quad \mathbf{x} = \mathbf{K}\begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix}\mathbf{X}$$

- In practice, non-linear methods are preferred
  - Write down objective function in terms of intrinsic and extrinsic parameters
  - Define error as sum of squared distances between measured 2D points and estimated projections of 3D points
  - Minimize error using Newton's method or another non-linear optimization
  - Can model radial distortion and impose constraints such as known focal length and orthogonality

Example 2: camera calibration

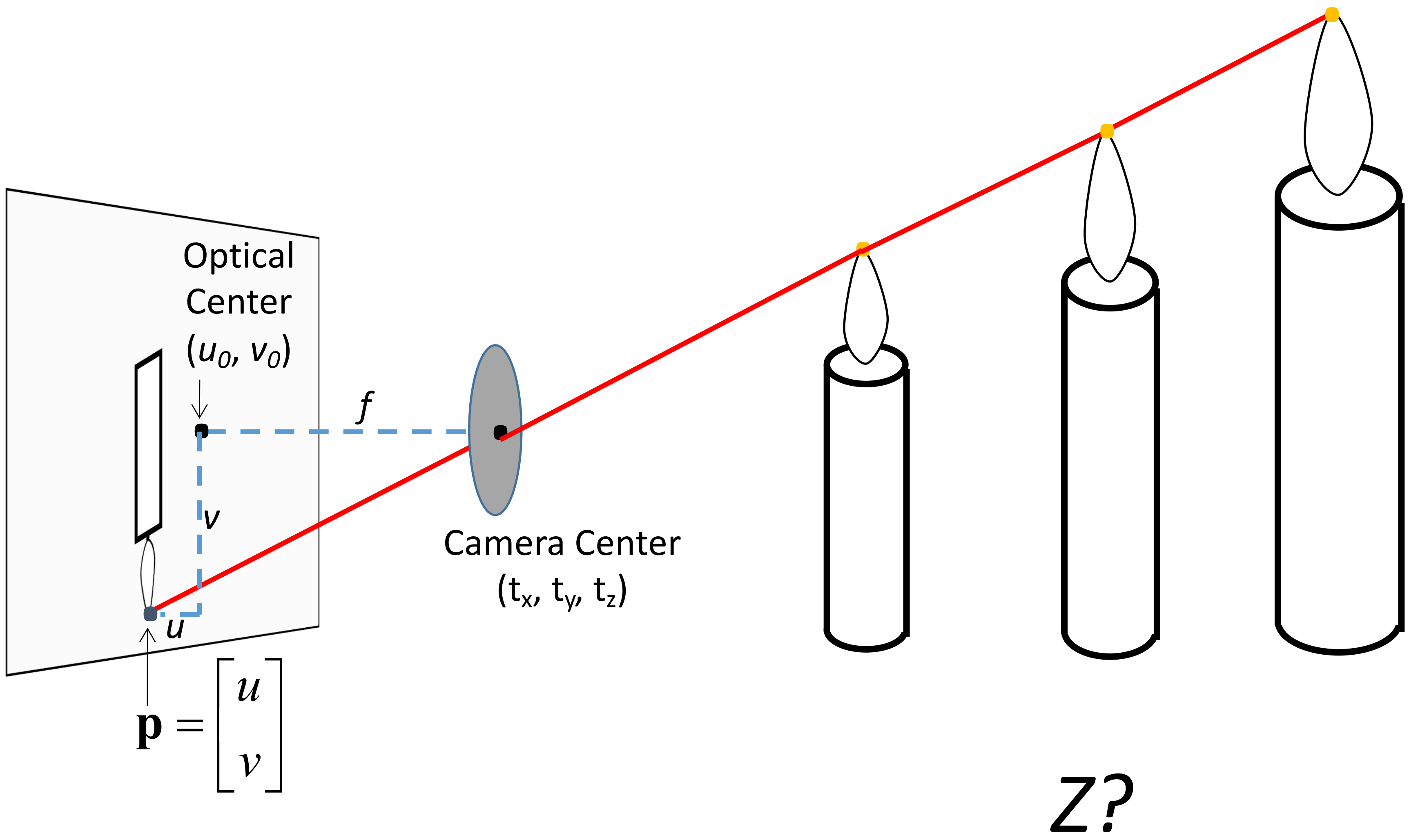
- In this example, we will calibrate a camera with the asymmetric circle grid pattern.



https://github.com/opi-lab/stsiva-workshop/blob/main/notebooks/stsiva_workshop_notebook02.ipynb

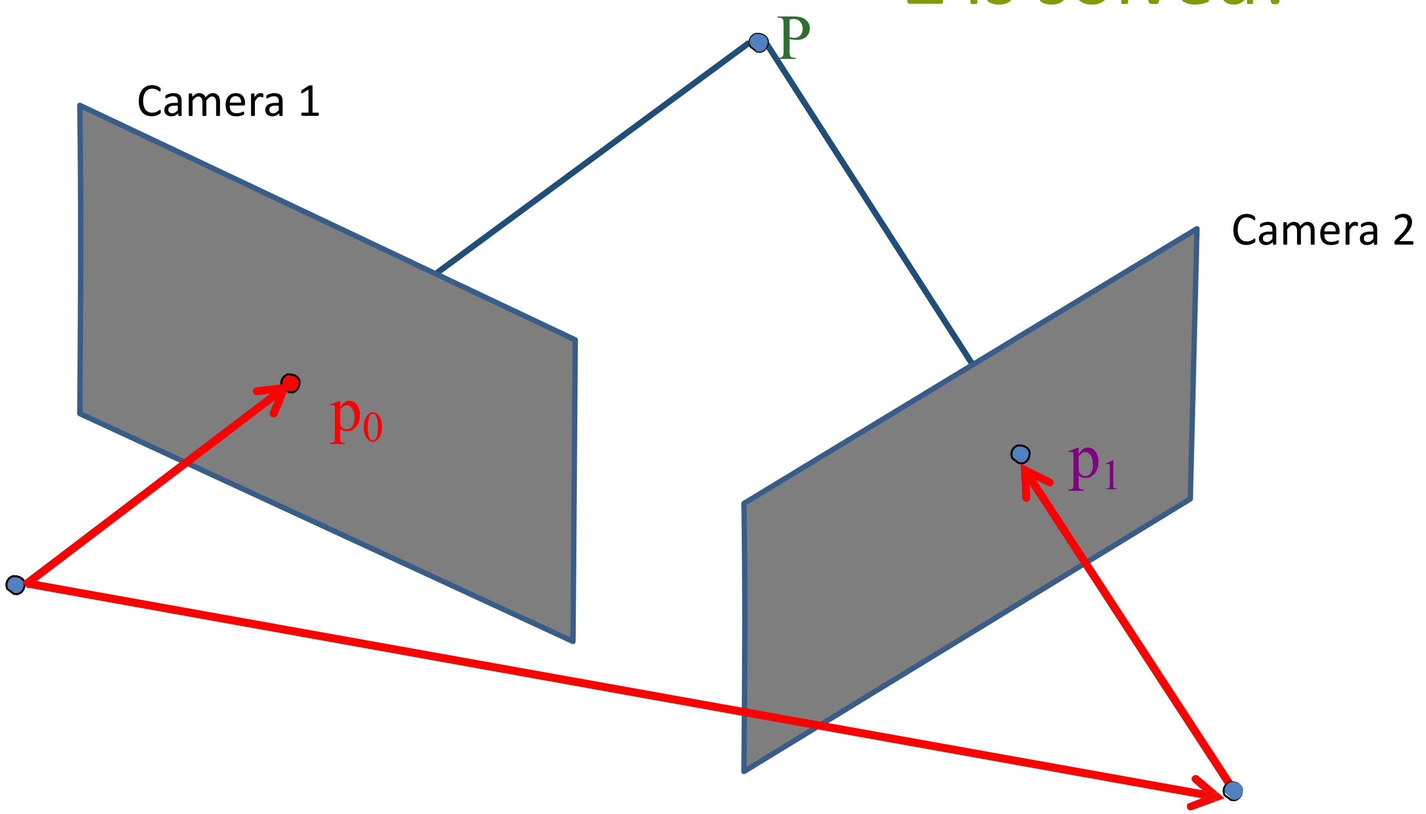
2. 3D from passive stereo vision system.

How does a camera “see” the world?



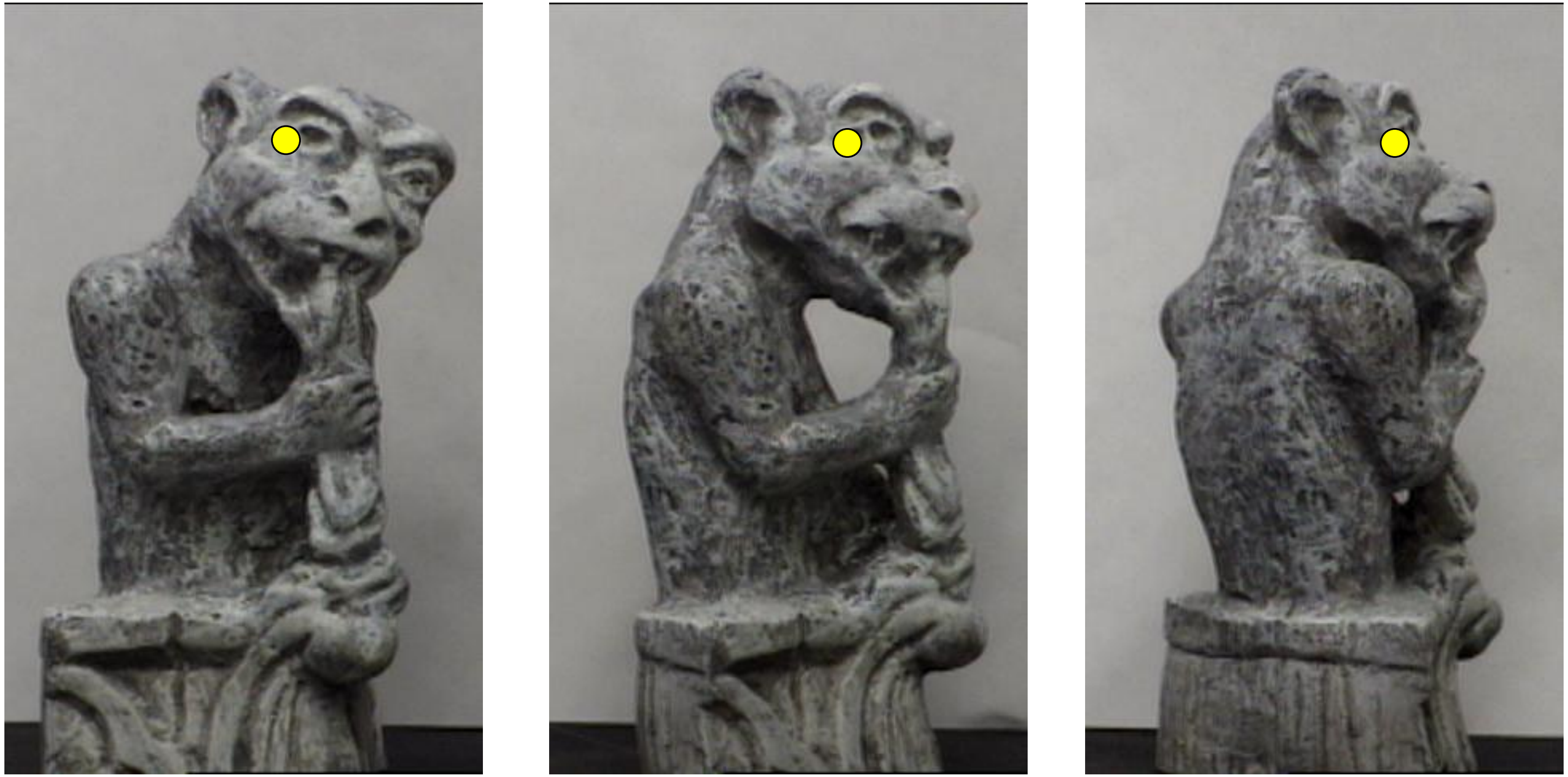
Solution: add another camera

Z is solved!



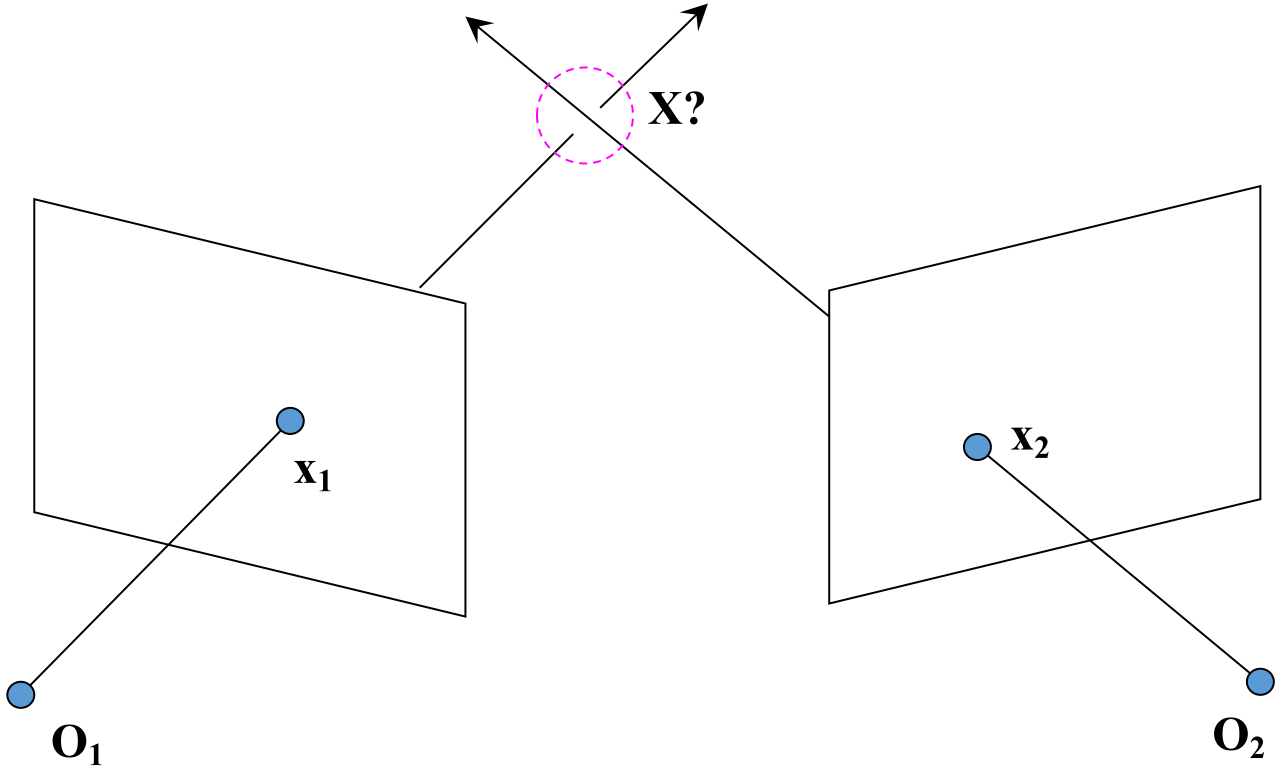
A taste of multi-view geometry: Triangulation

Given projections of a 3D point in two or more images (with known camera matrices), find the coordinates of the point



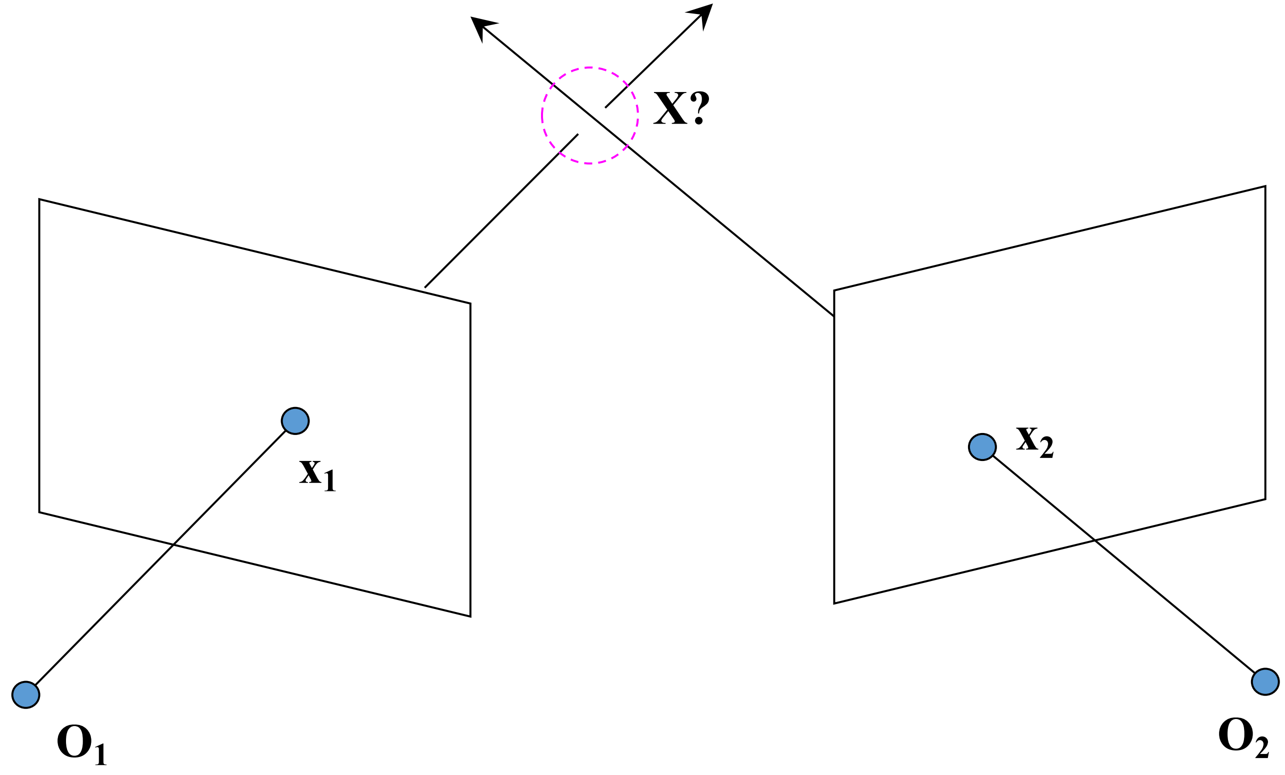
Triangulation

- Given projections of a 3D point in two or more images (with known camera matrices), find the coordinates of the point



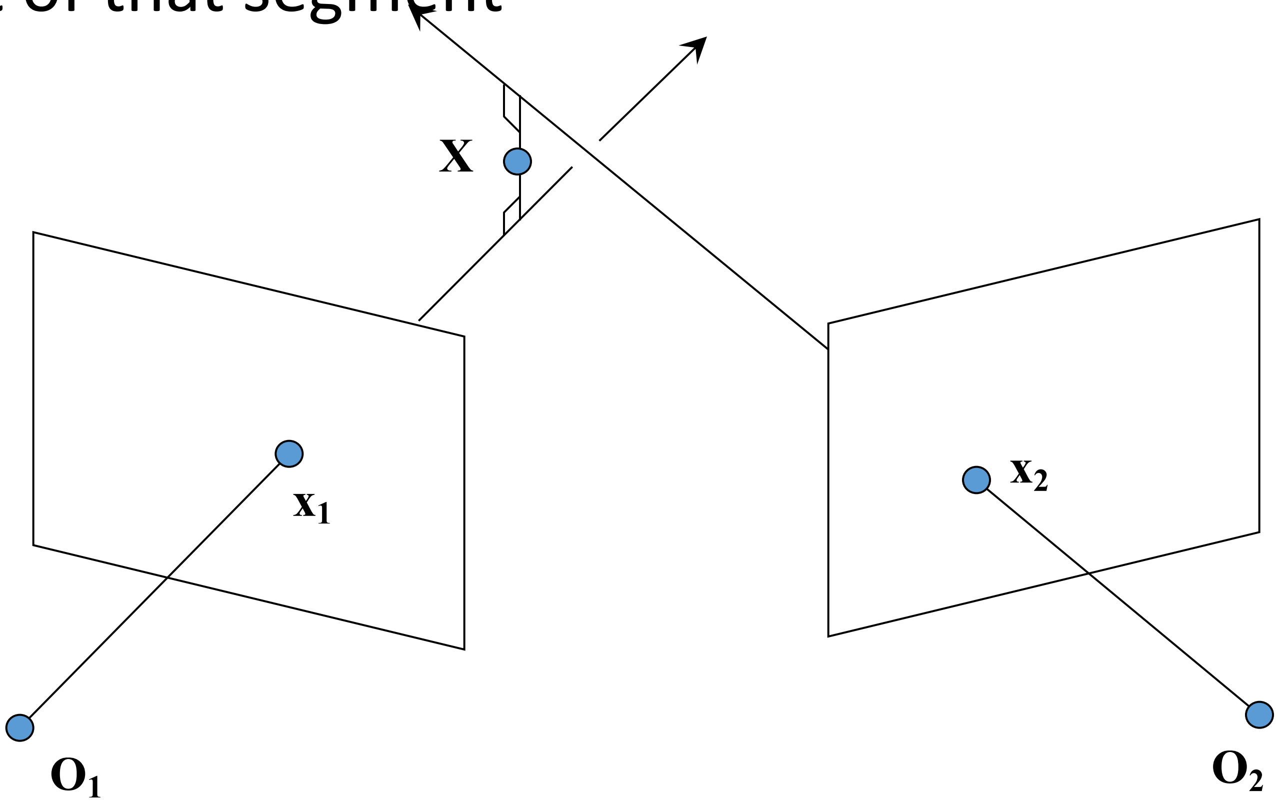
Triangulation

- We want to intersect the two visual rays corresponding to $\mathbf{x}_1$ and $\mathbf{x}_2$, but because of noise and numerical errors, they don't meet exactly



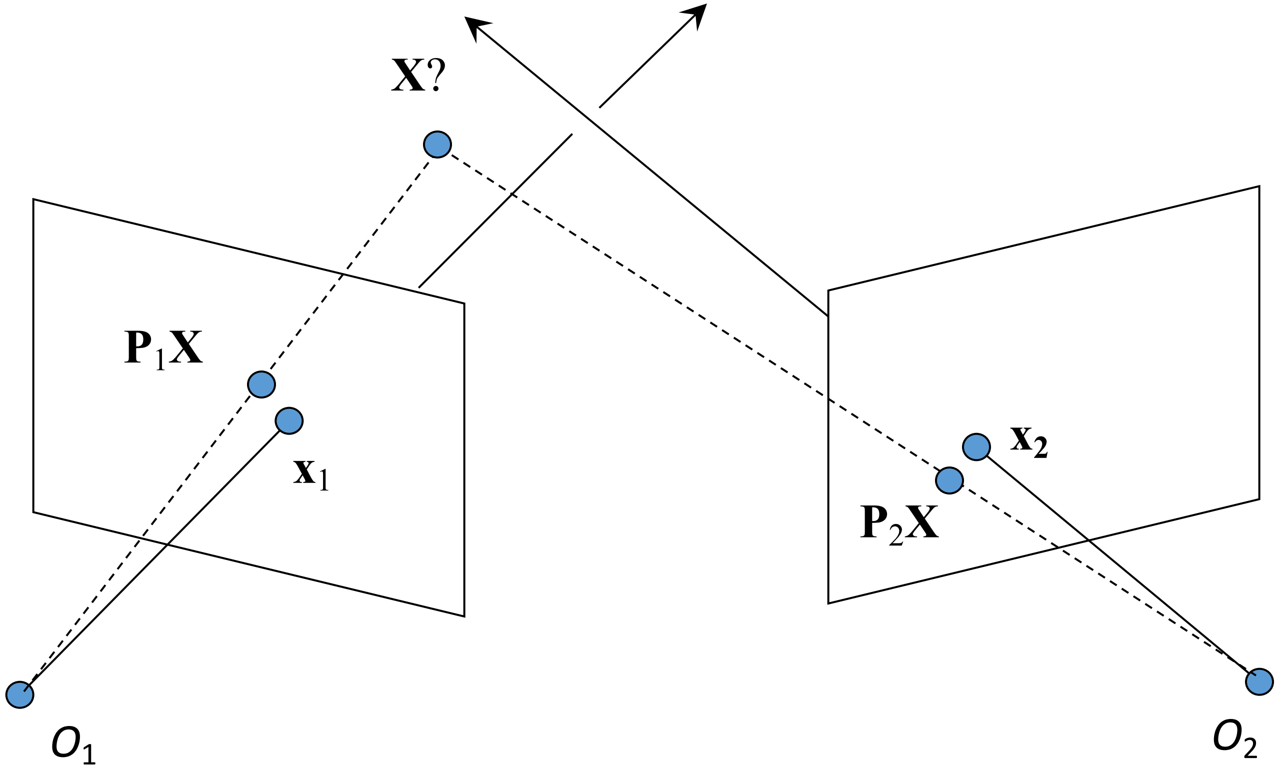
Triangulation: Geometric approach

- Find shortest segment connecting the two viewing rays and let **X** be the midpoint of that segment



Triangulation: Nonlinear approach

- Find X that minimizes $d^2(\mathbf{x_1}, \mathbf{P_1}\mathbf{X}) + d^2(\mathbf{x_2}, \mathbf{P_2}\mathbf{X})$



Triangulation: Linear approach

$$\lambda_1 \mathbf{x_1} = \mathbf{P_1 X} \qquad \mathbf{x_1} \times \mathbf{P_1 X} = \mathbf{0} \qquad [\mathbf{x_{1\times}}]\mathbf{P_1 X} = \mathbf{0}$$

$$\lambda_2 \mathbf{x_2} = \mathbf{P_2 X} \qquad \mathbf{x_2} \times \mathbf{P_2 X} = \mathbf{0} \qquad [\mathbf{x_{2\times}}]\mathbf{P_2 X} = \mathbf{0}$$

Cross product as matrix multiplication:

$$\mathbf{a} \times \mathbf{b} = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} = [\mathbf{a}_\times]\mathbf{b}$$

Triangulation: Linear approach

$$\lambda_1 \mathbf{x_1} = \mathbf{P_1 X} \qquad \mathbf{x_1} \times \mathbf{P_1 X} = \mathbf{0} \qquad [\mathbf{x}_{\mathbf{1}\times}]\mathbf{P_1 X} = \mathbf{0}$$

$$\lambda_2 \mathbf{x_2} = \mathbf{P_2 X} \qquad \mathbf{x_2} \times \mathbf{P_2 X} = \mathbf{0} \qquad [\mathbf{x}_{\mathbf{2}\times}]\mathbf{P_2 X} = \mathbf{0}$$

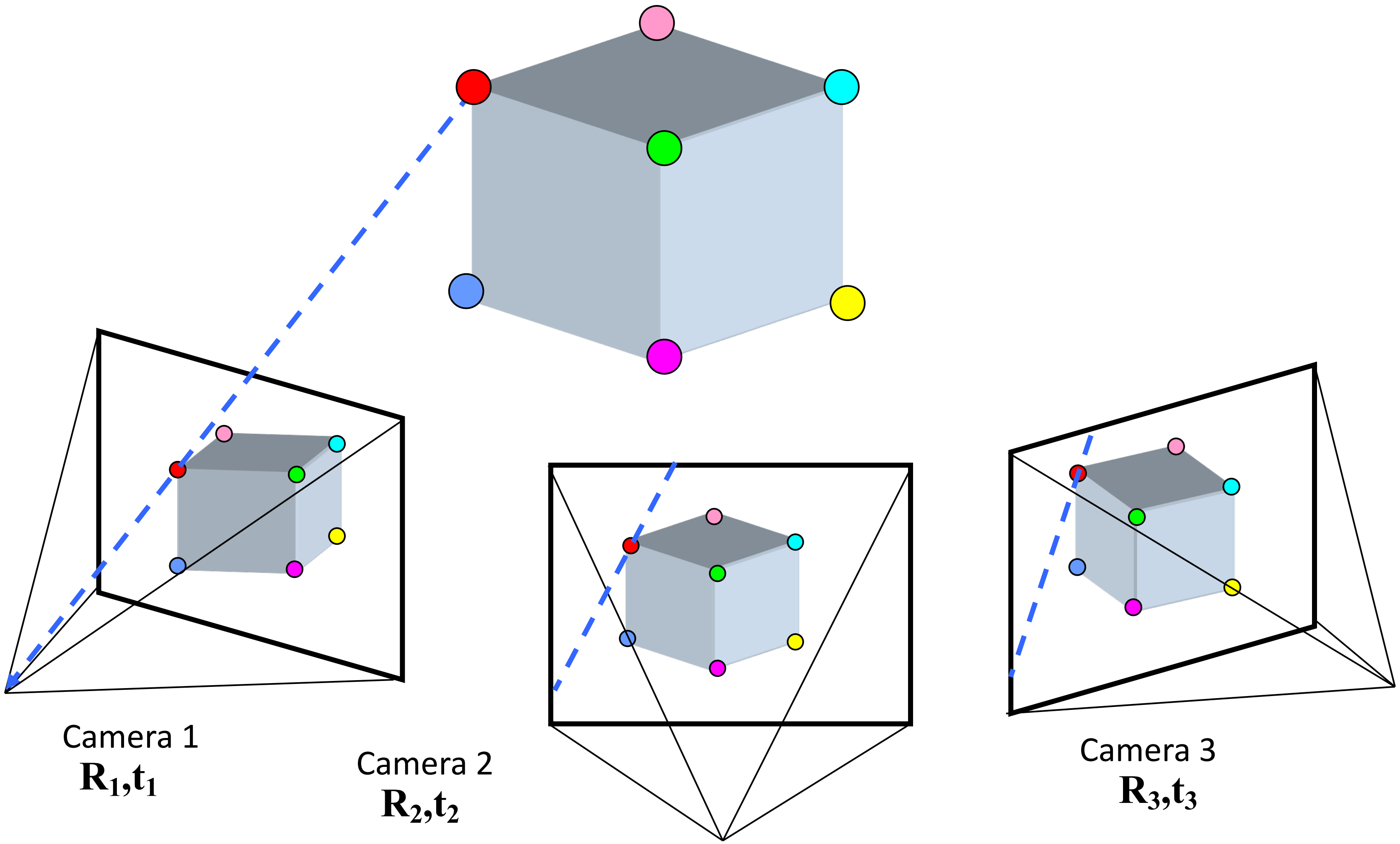
Two independent equations each in terms of three unknown entries of **X**

Problems?

Stereo correspondence: Given a point in one of the images, where could its corresponding points be in the other images?

Problems?

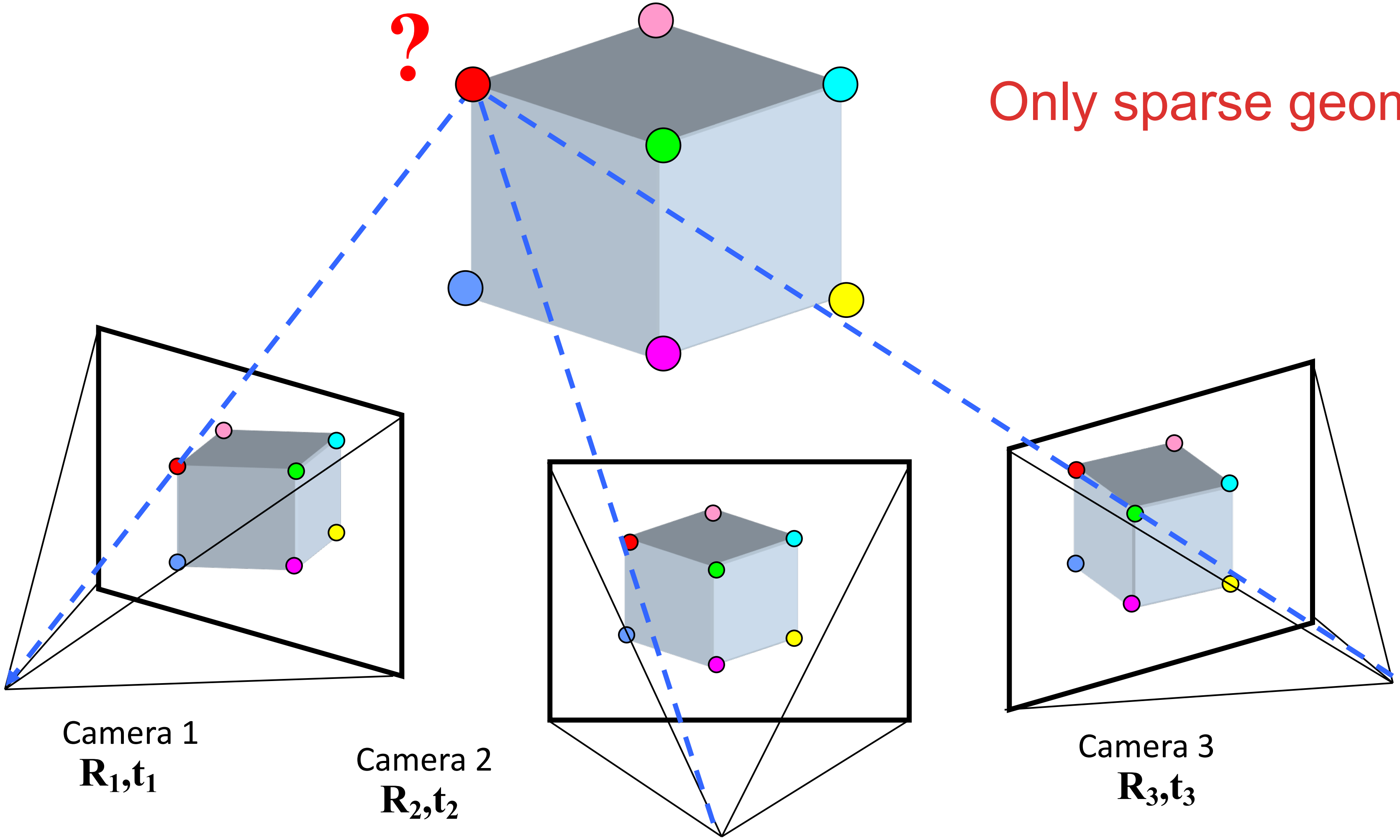
Stereo correspondence: Given a point in one of the images, where could its corresponding points be in the other images?



Slide credit: Noah Snavely

Problems?

Structure: Given projections of the same 3D point in two or more images, compute the 3D coordinates of that point



Only sparse geometry

Slide credit: Noah Snavely

Problems?

2D images Sparse reconstruction

Only sparse geometry

Problems?

Object

Feature based 3D
reconstruction

Only sparse geometry

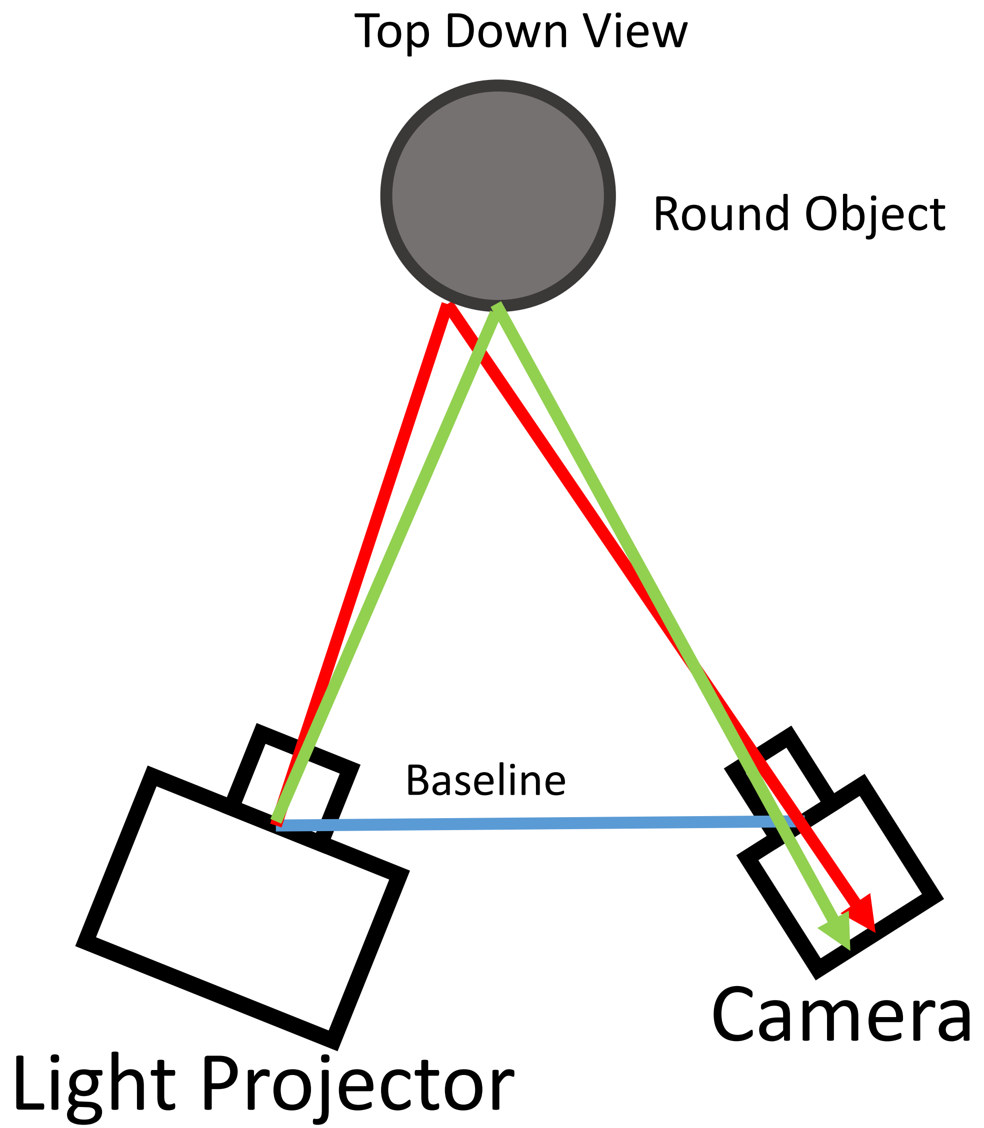
Solution?

Idea:

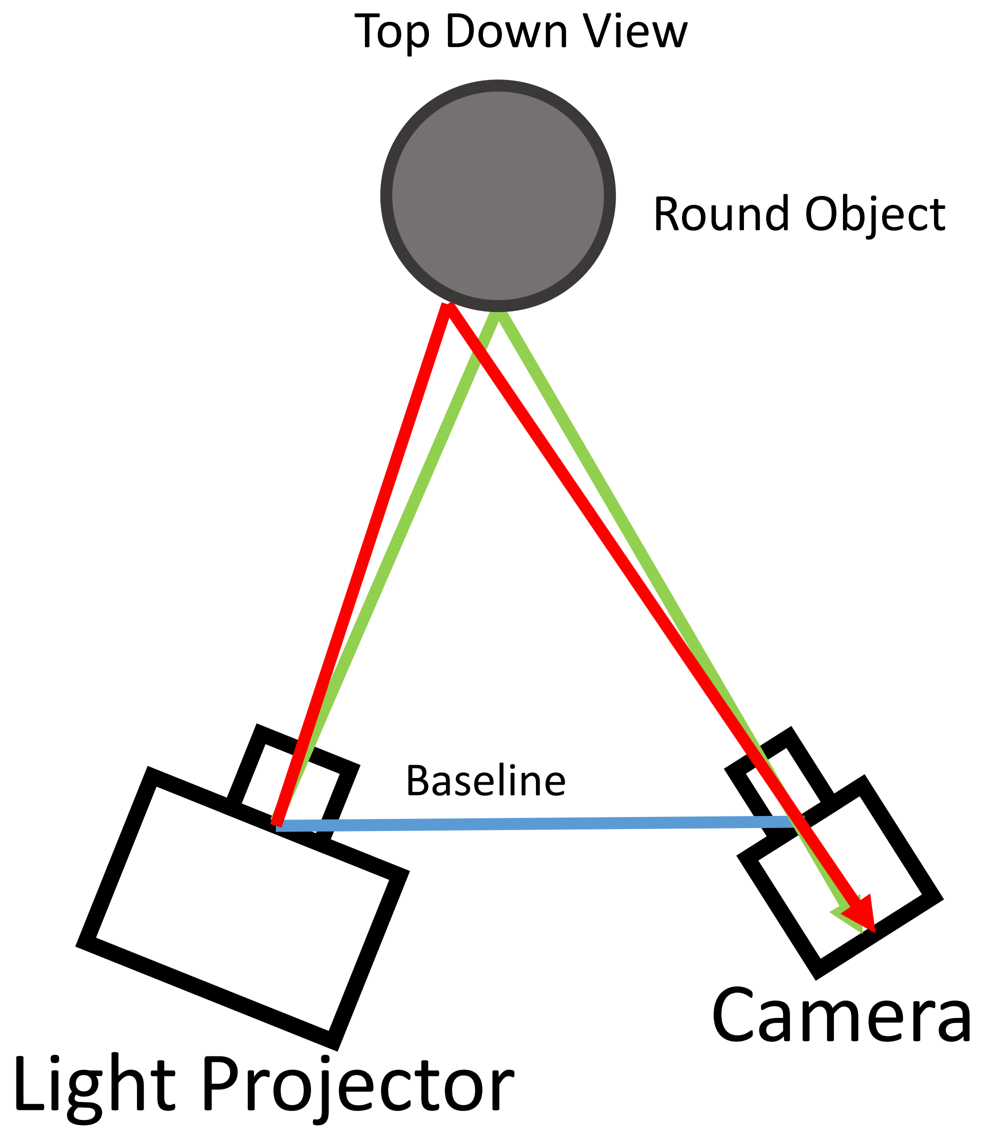
- If objects don't have a **distinctive texture**, give them one with e.g.
  - Point/line lasers
  - **Video projectors**

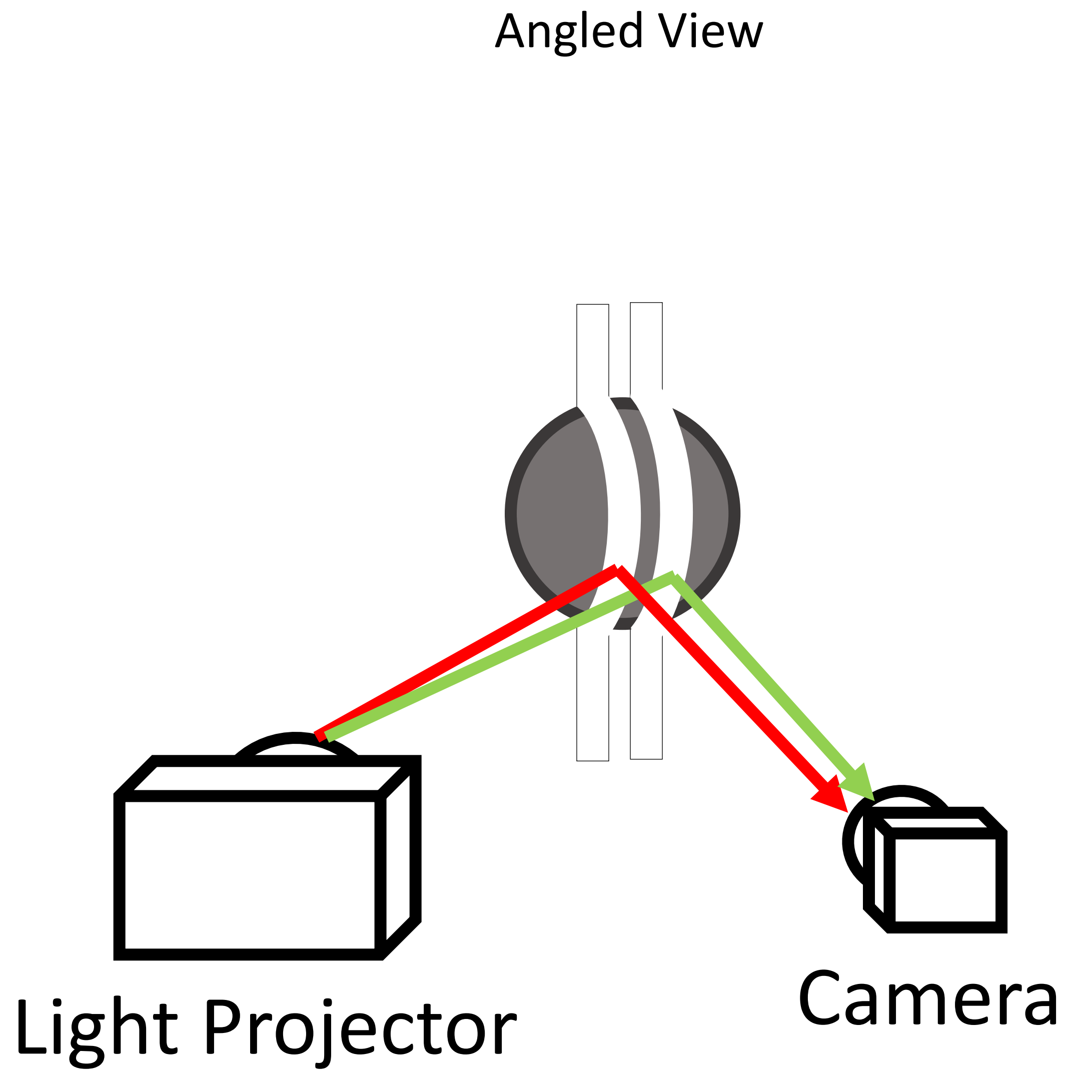
3. 3D from active stereo vision system (aka structured light).

Structured light system



Structured light system

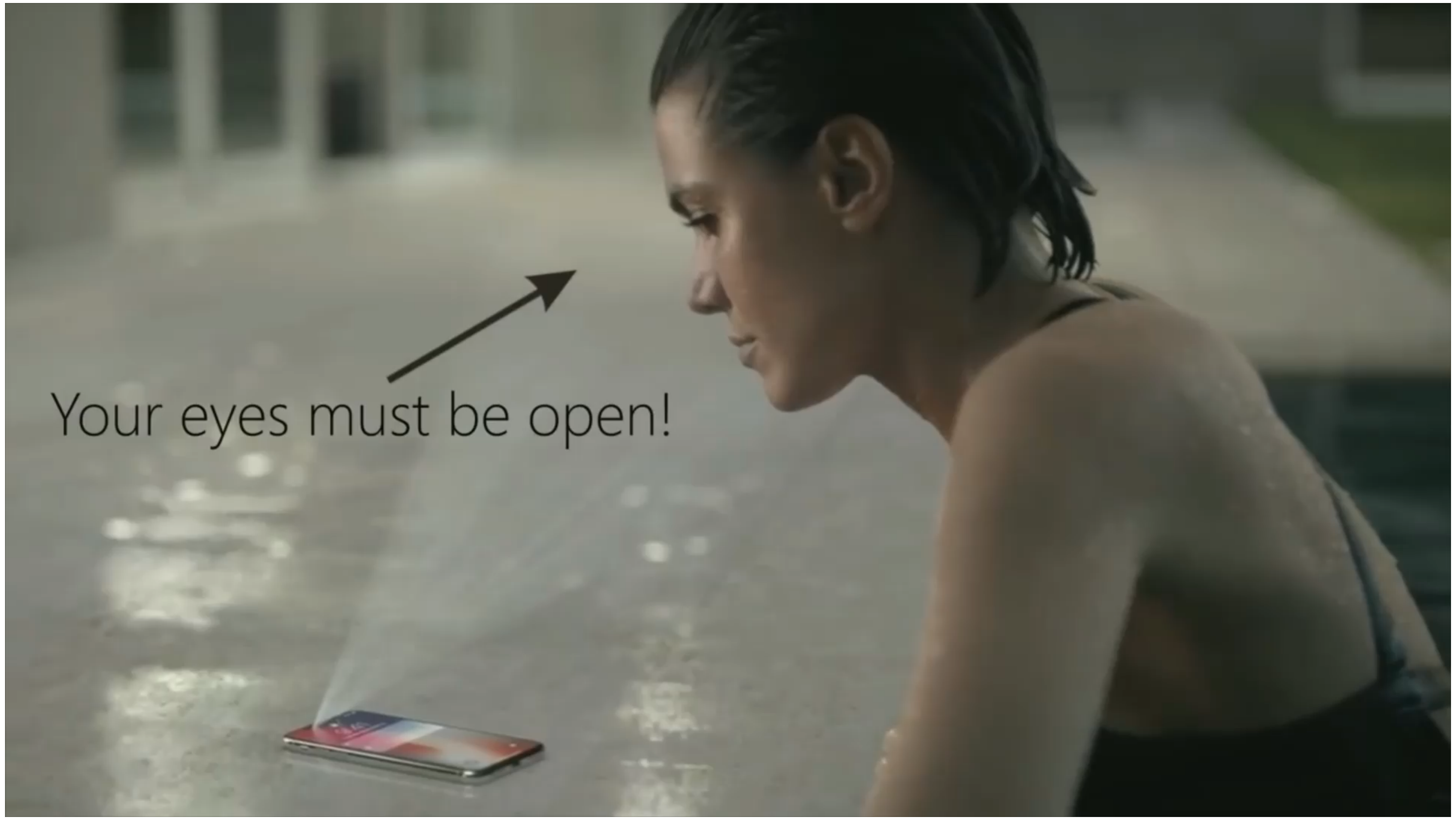




Structured light system

Many commercial systems are based on structured light, e.g.,

- FaceID iPhone X
- Kinect V1
- GOM
- Intel Realsense
- Orbbec

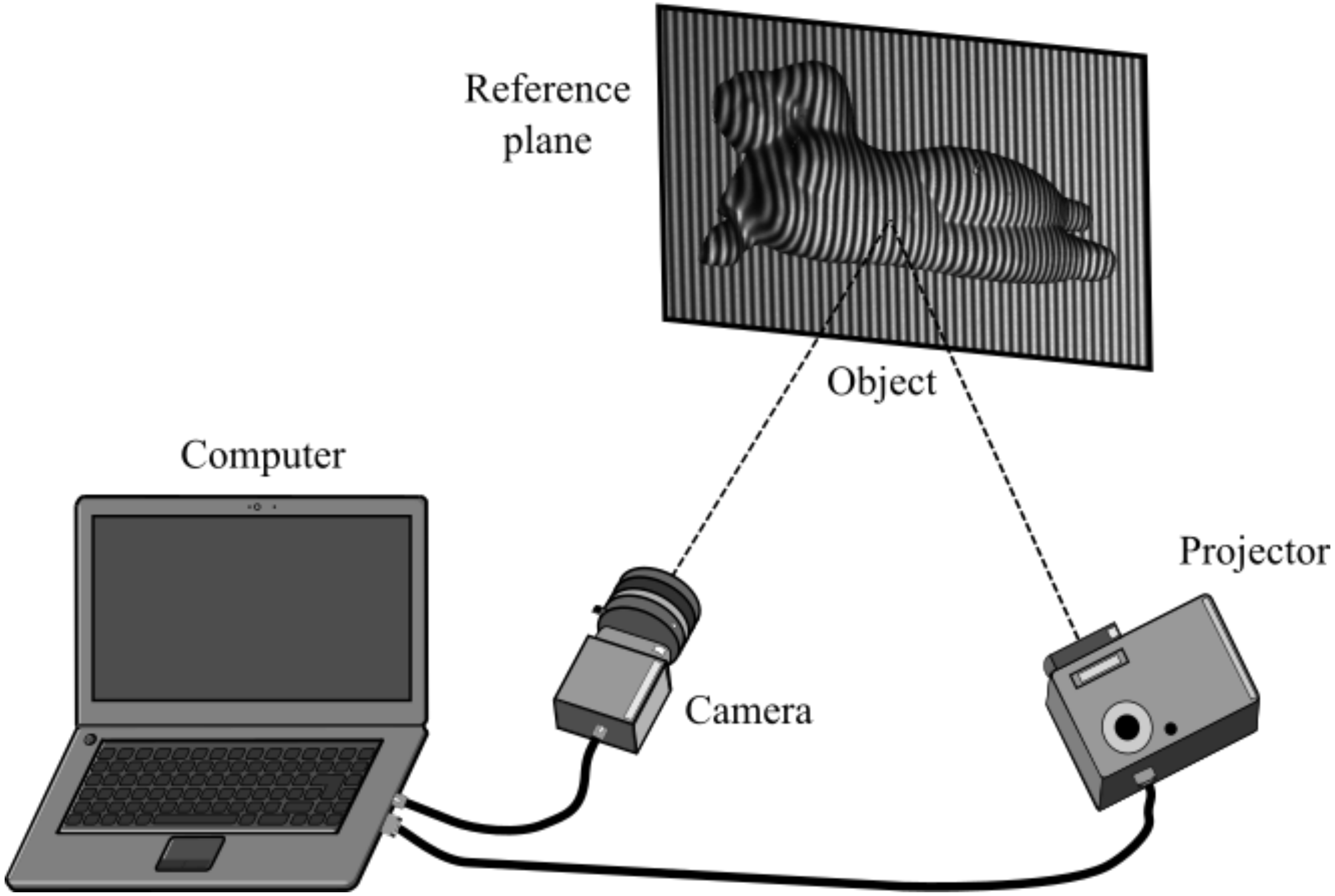


Structured light system

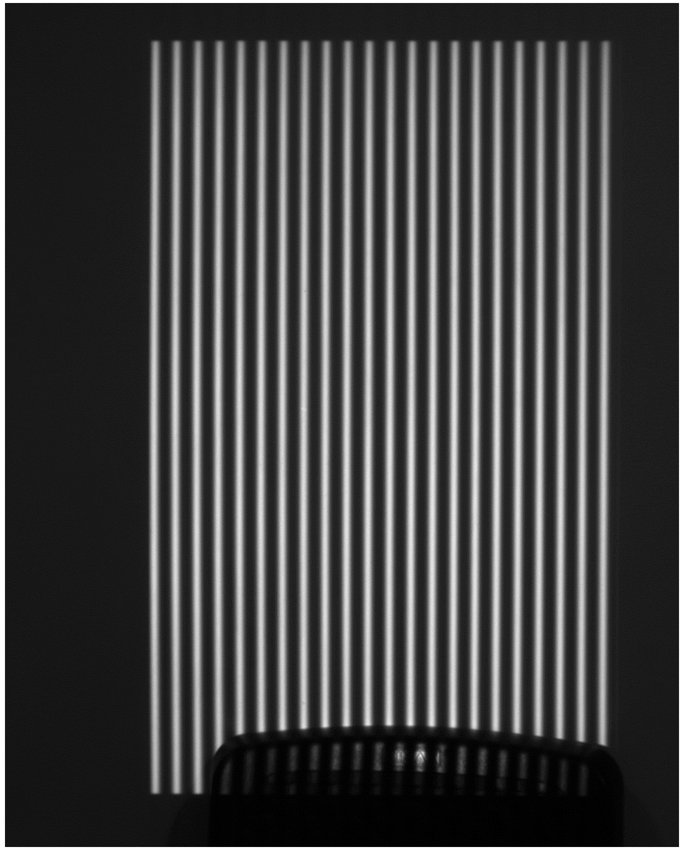
Many commercial systems are based on structured light, e.g.,

- FaceID iPhone X
- Kinect V1
- GOM
- Intel Realsense
- Orbbec

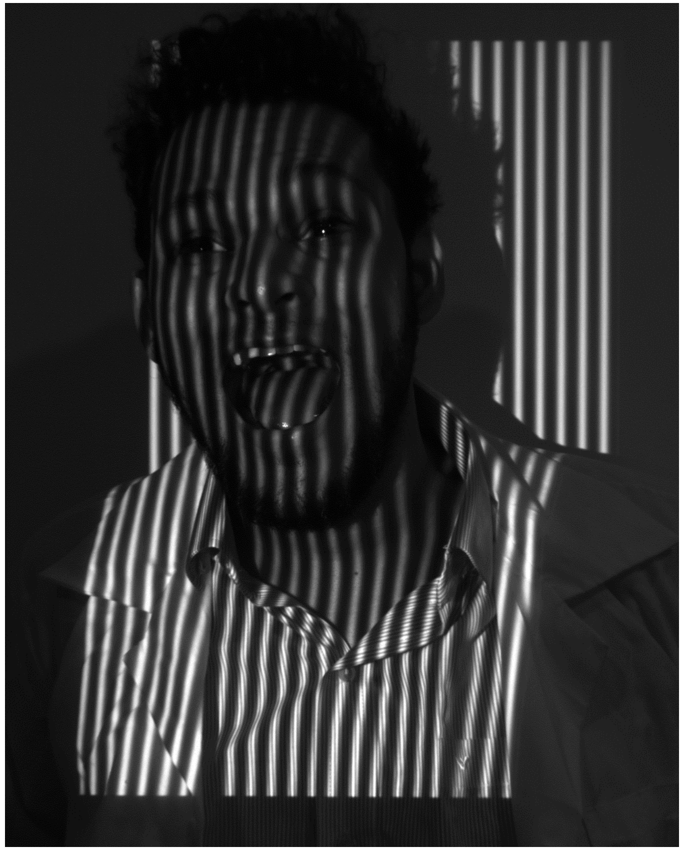
Structured light system



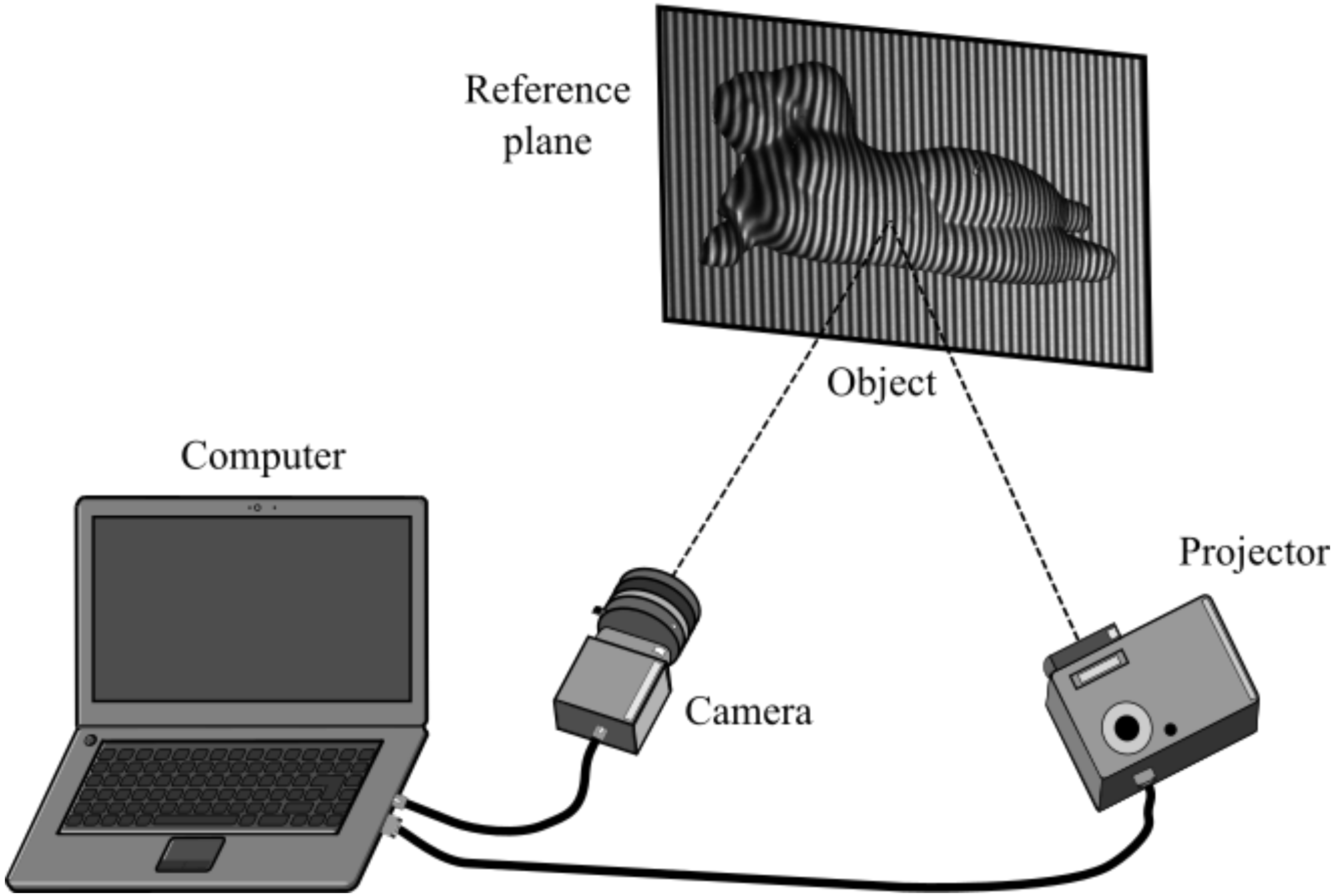
Flat surface

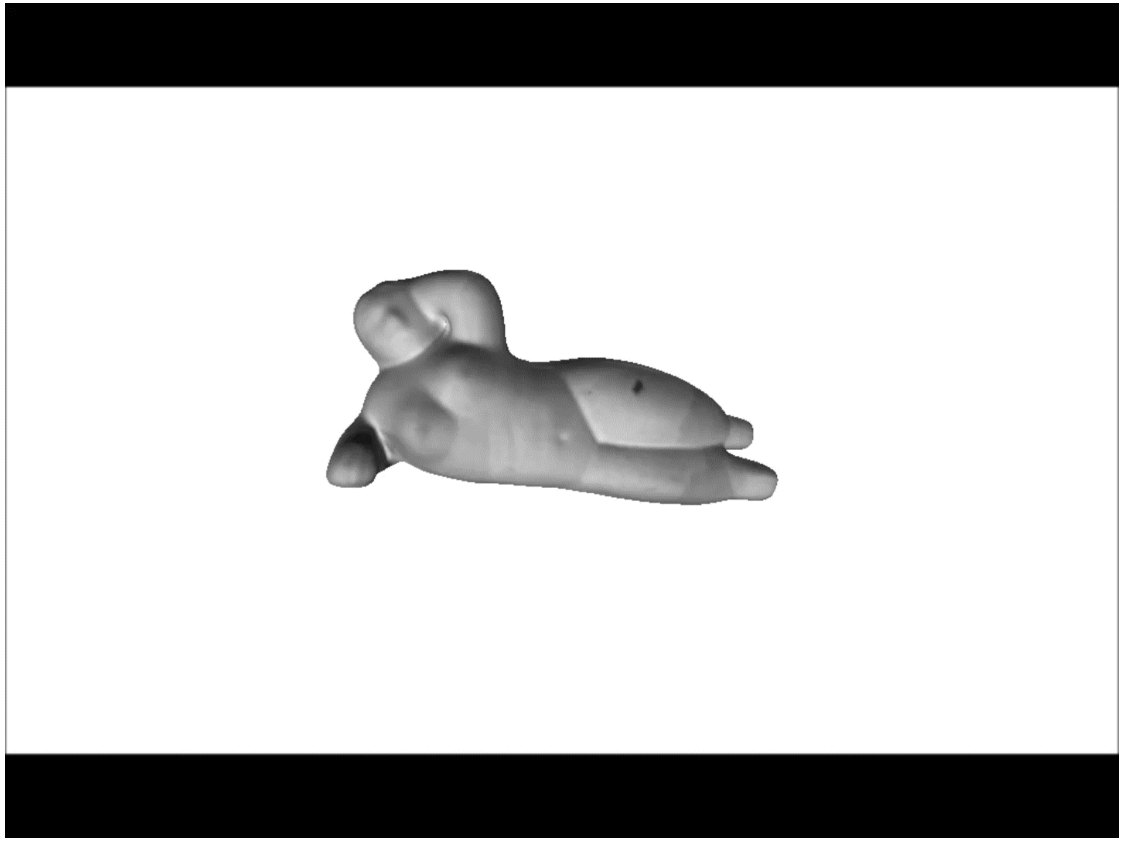


Surface with topography

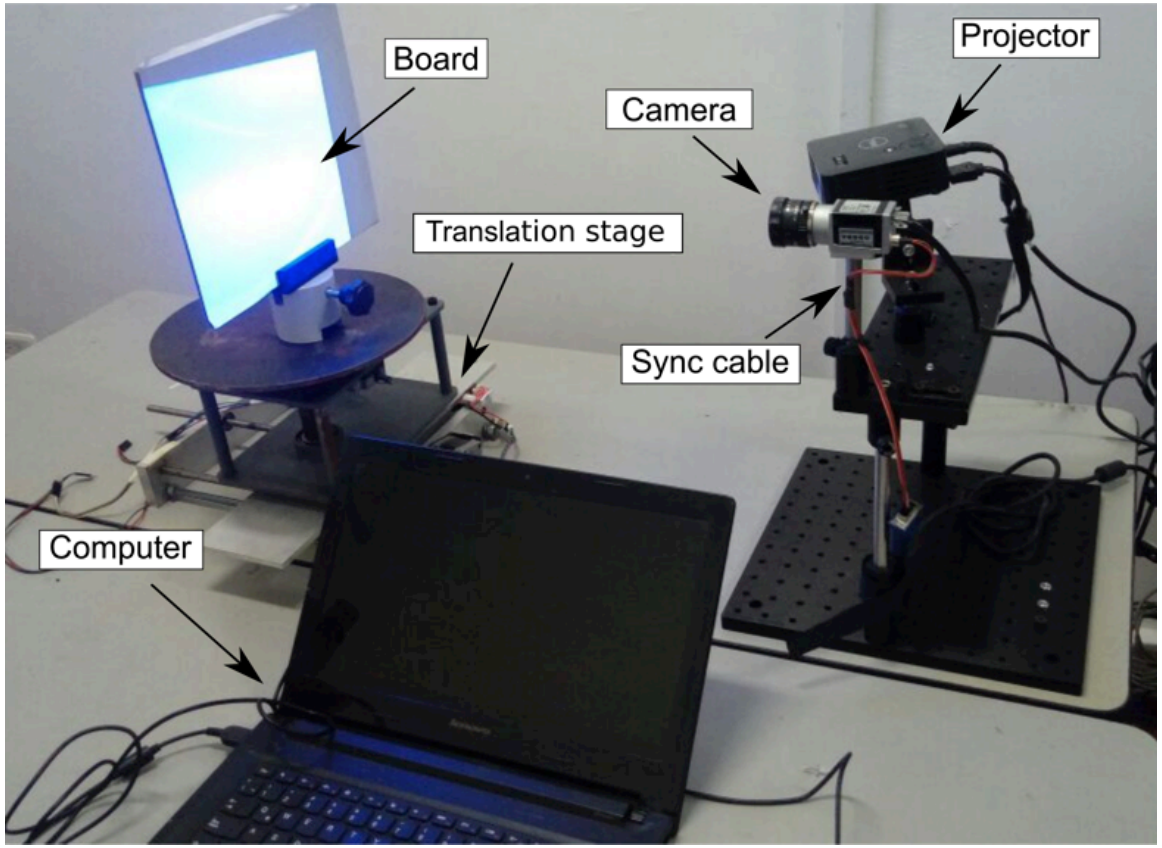


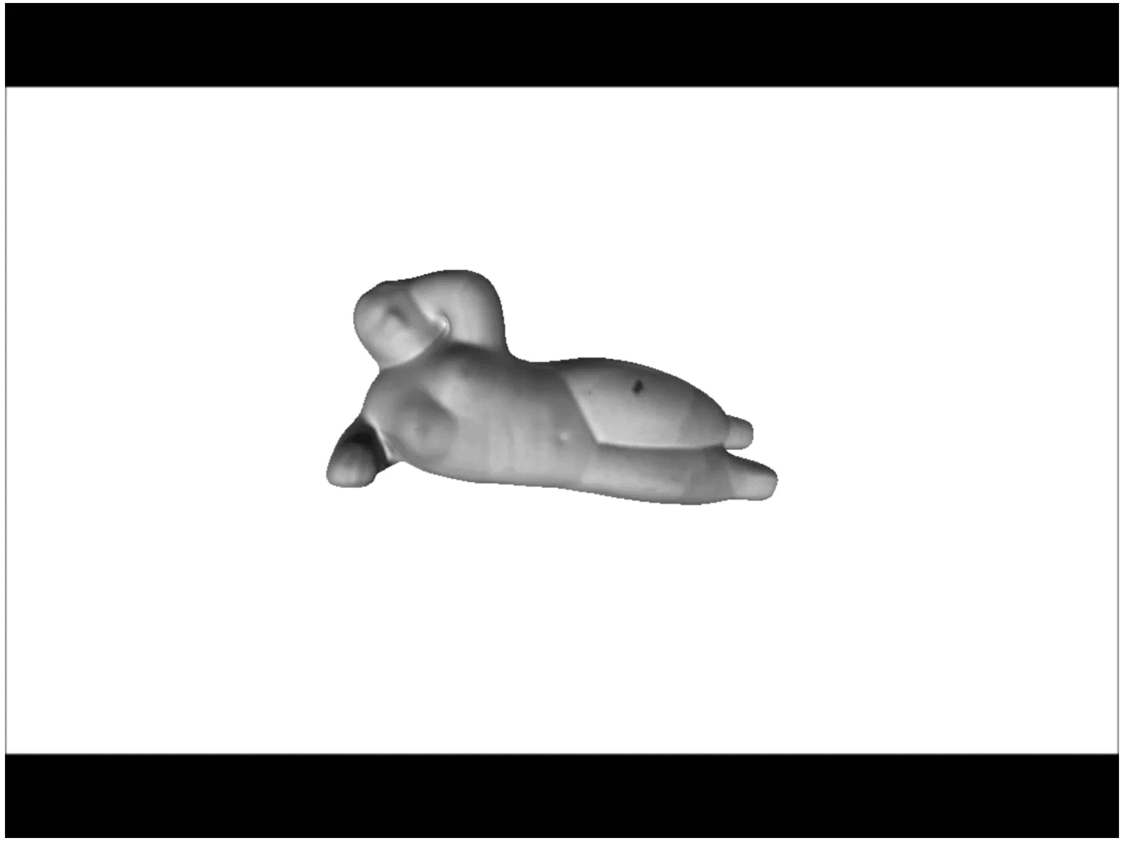
Structured light system





Structured light system

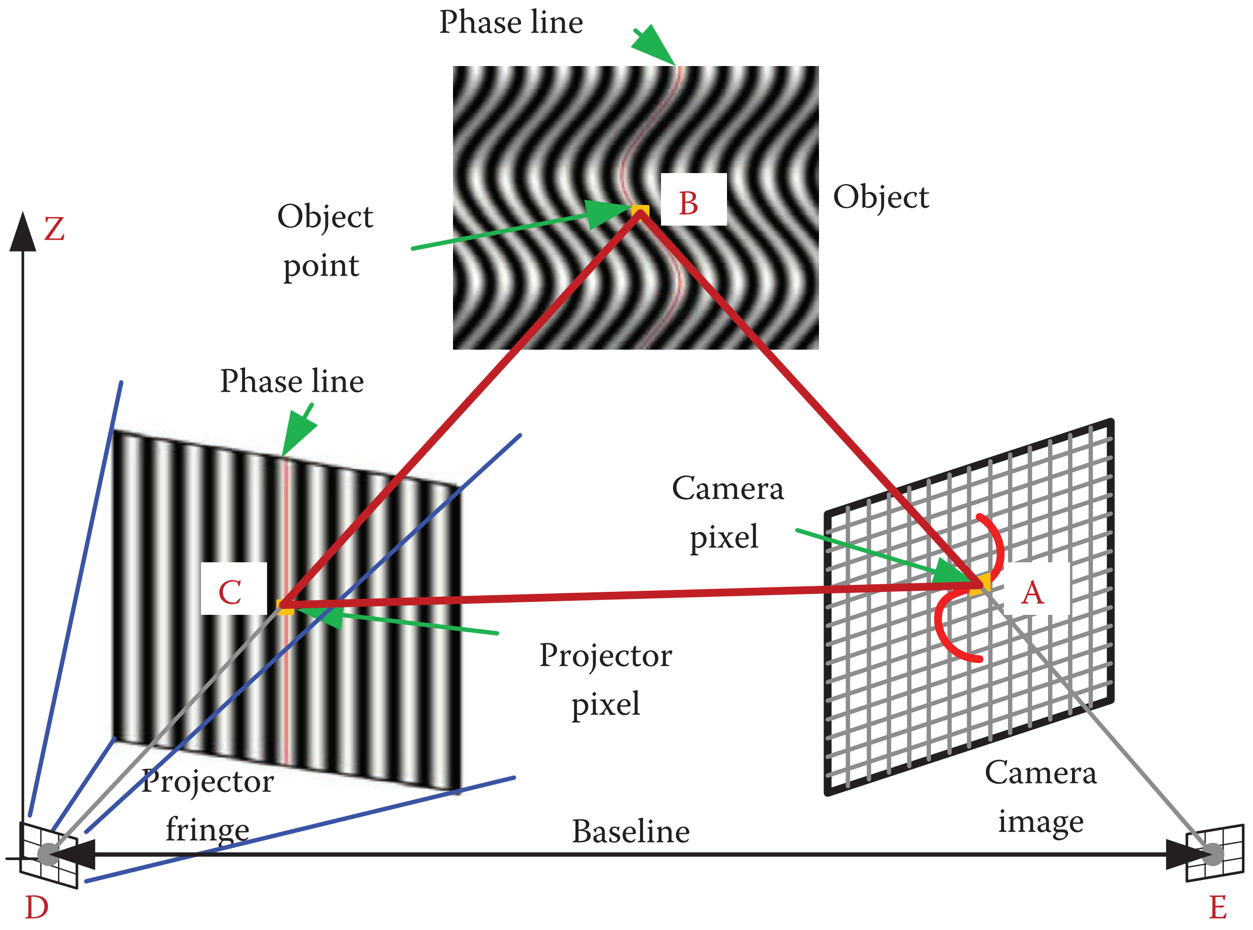




The stereo-vision model

The most used calibration method for a **structured light** system is the **stereo-vision method** with **pinhole camera model.**

The projector is regarded as an **inverse camera**.



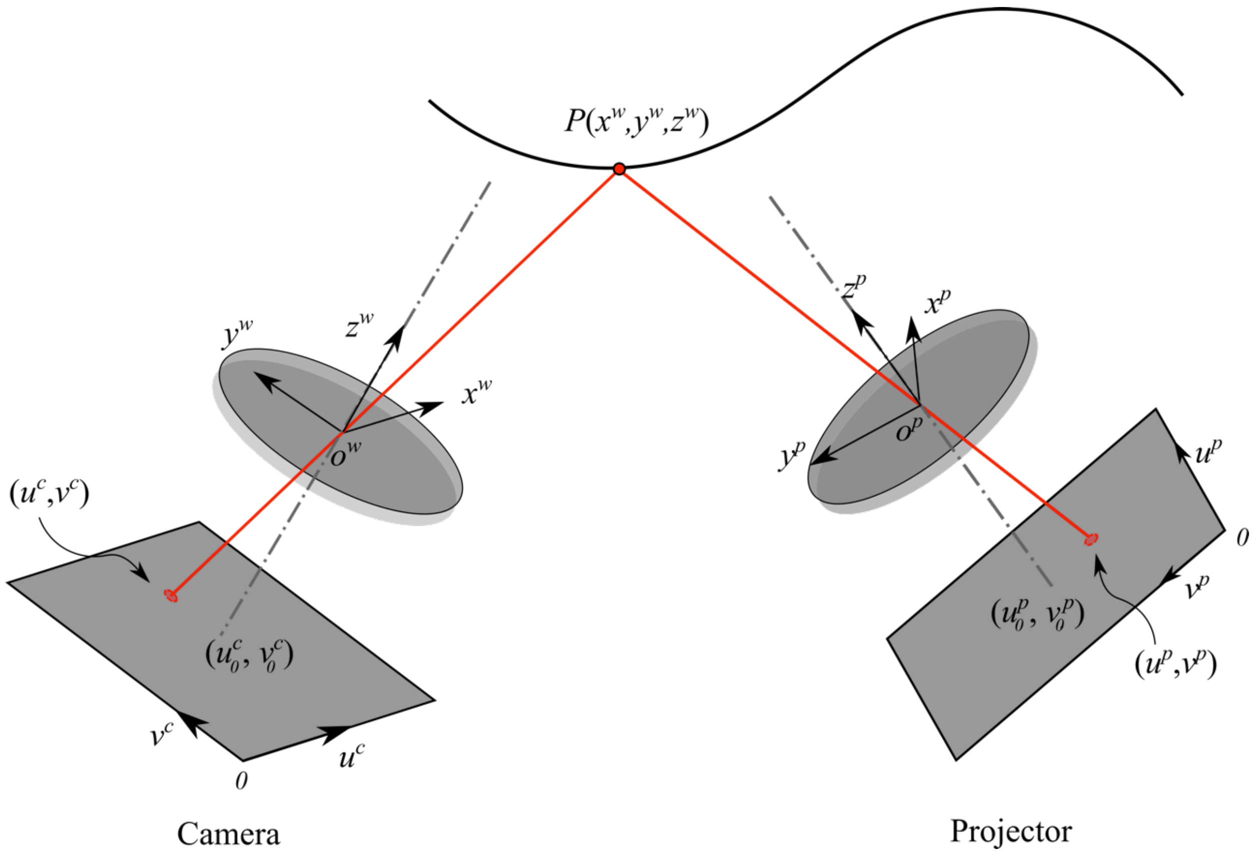
S. Zhang, 2016

The stereo-vision model

In this model, the **3D reconstruction** of a point $P(x^w, y^w, z^w)$ is carried out by solving the equations

$$s^c \mathbf{x}^c = \mathbf{P}^c \mathbf{X}^\mathrm{w} \quad ,$$

$$s^p \mathbf{x}^p = \mathbf{P}^p \mathbf{X}^\mathrm{w} \quad .$$

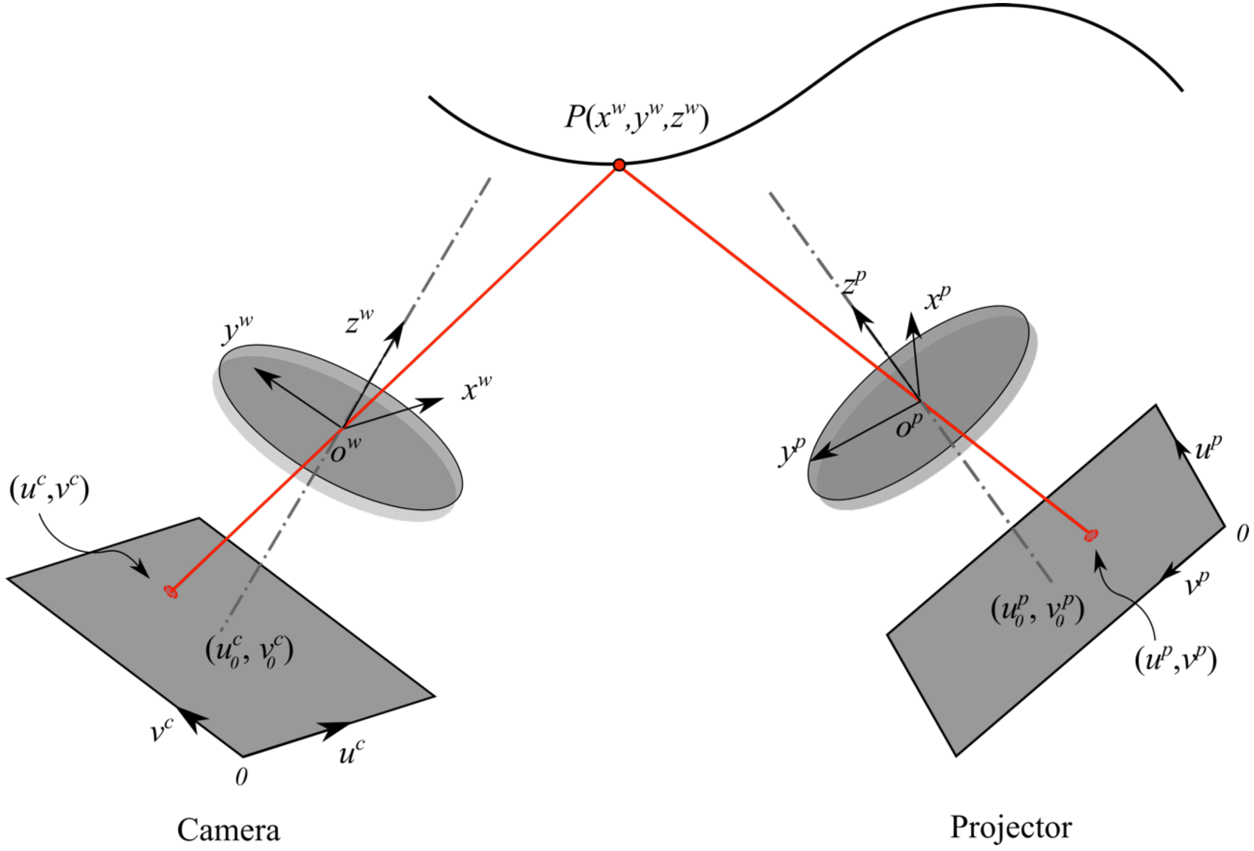


The stereo-vision model

In this model, the **3D reconstruction** of a point $P(x^w, y^w, z^w)$ is carried out by solving the equations

$$s^c \begin{bmatrix} u^c \\ v^c \\ 1 \end{bmatrix} = \mathbf{P}^c \begin{bmatrix} x^w \\ y^w \\ z^w \\ 1 \end{bmatrix} \quad ,$$

$$s^p \begin{bmatrix} u^p \\ v^p \\ 1 \end{bmatrix} = \mathbf{P}^p \begin{bmatrix} x^w \\ y^w \\ z^w \\ 1 \end{bmatrix} \quad ,$$

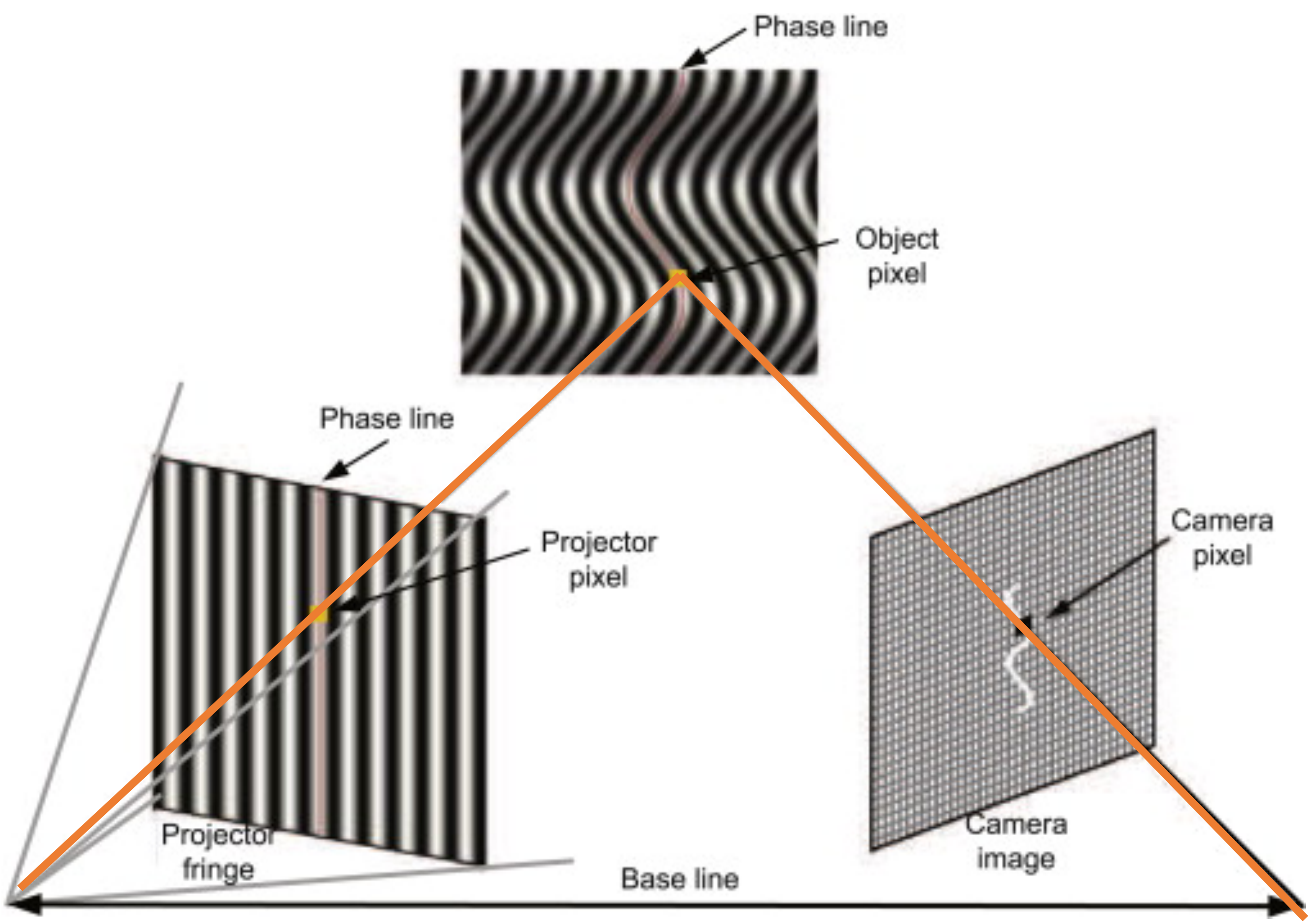


The stereo-vision model

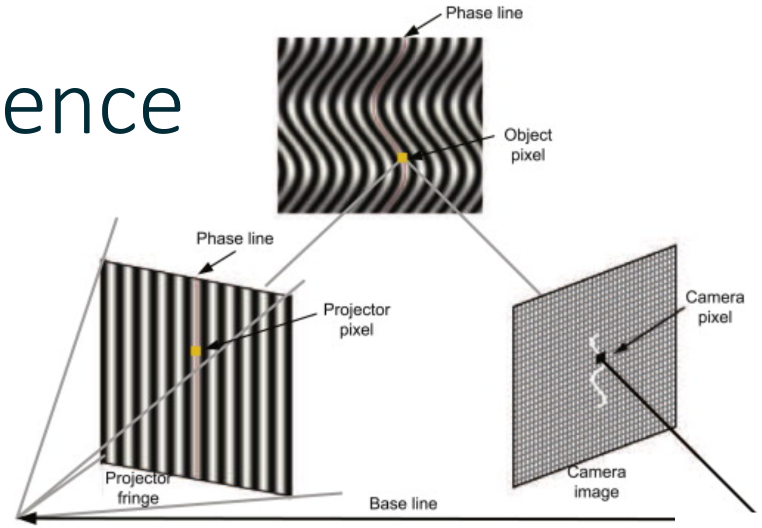
In this model, the **3D reconstruction** of a point $P(x^w, y^w, z^w)$ is carried out by solving the equations

$$s^c \begin{bmatrix} u^c \\ v^c \\ 1 \end{bmatrix} = \mathbf{P}^c \begin{bmatrix} x^w \\ y^w \\ z^w \\ 1 \end{bmatrix} \quad ,$$

$$s^p \begin{bmatrix} u^p \\ v^p \\ 1 \end{bmatrix} = \mathbf{P}^p \begin{bmatrix} x^w \\ y^w \\ z^w \\ 1 \end{bmatrix} \quad ,$$



Projector correspondence

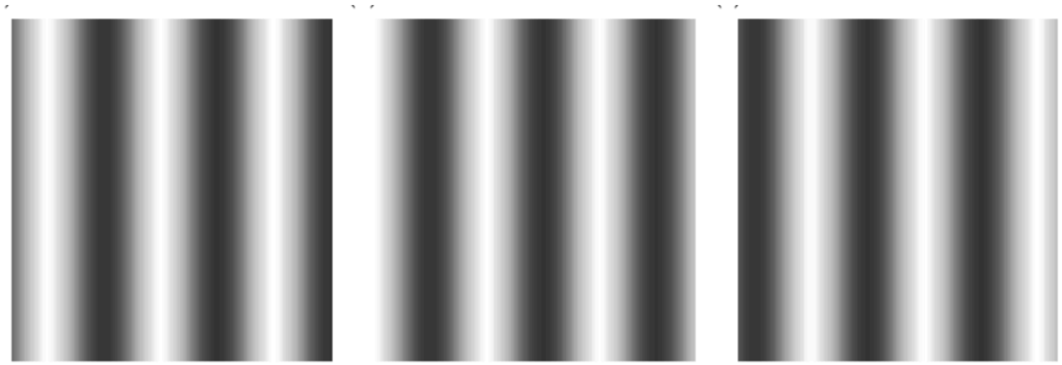


Projector

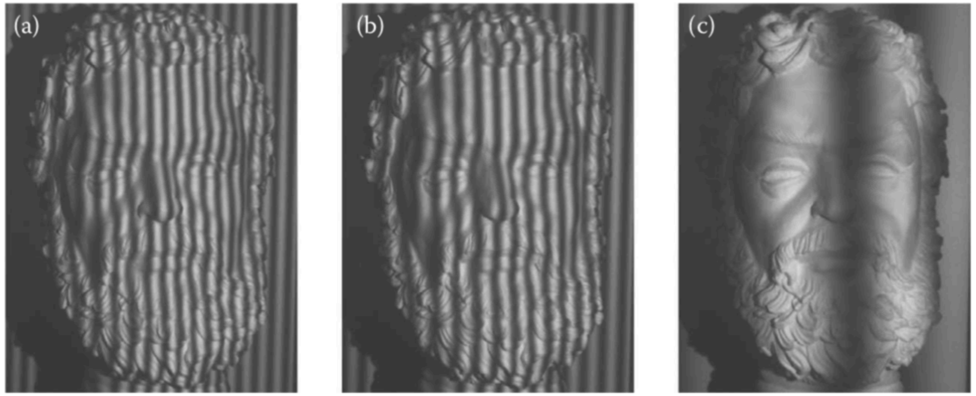
Camera

$$I_1(i,j) = 255/2[1 + \cos(2\pi j/P - 2\pi/3)],$$
$$I_2(i,j) = 255/2[1 + \cos(2\pi j/P)],$$
$$I_3(i,j) = 255/2[1 + \cos(2\pi j/P + 2\pi/3)].$$

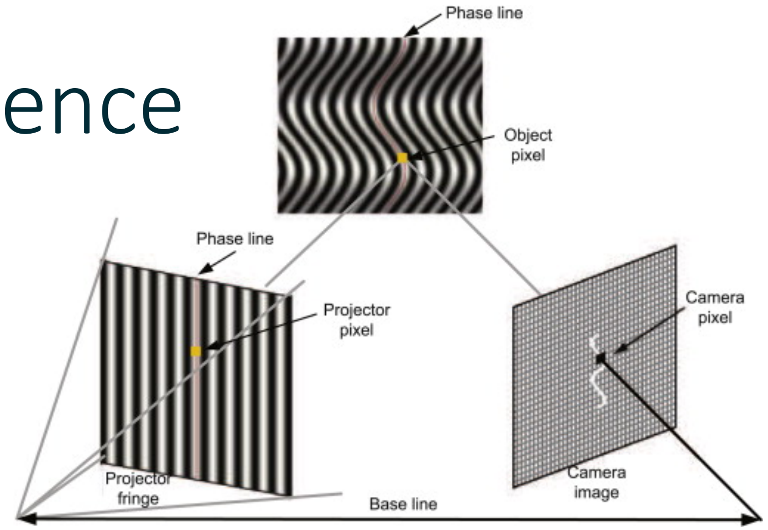
$$I_1(x,y) = I'(x,y) + I''(x,y)\cos[\phi(x,y) - \alpha],$$
$$I_2(x,y) = I'(x,y) + I''(x,y)\cos[\phi(x,y)],$$
$$I_3(x,y) = I'(x,y) + I''(x,y)\cos[\phi(x,y) + \alpha].$$



Phase-shifted fringe patterns

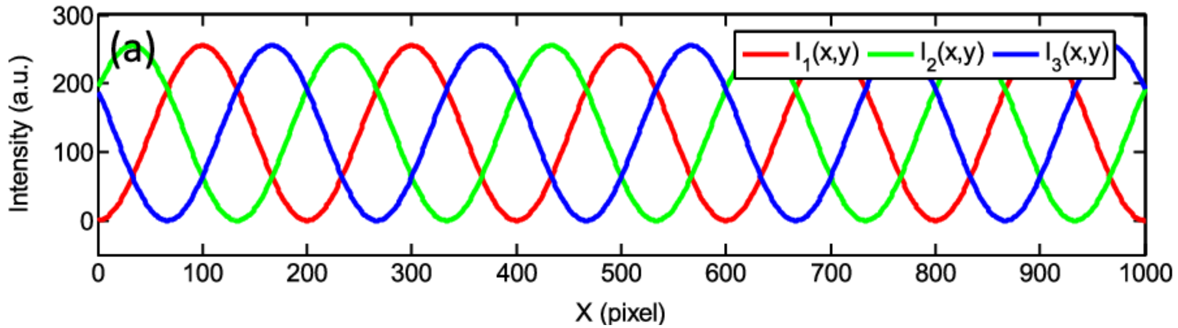


Projector correspondence



Projector

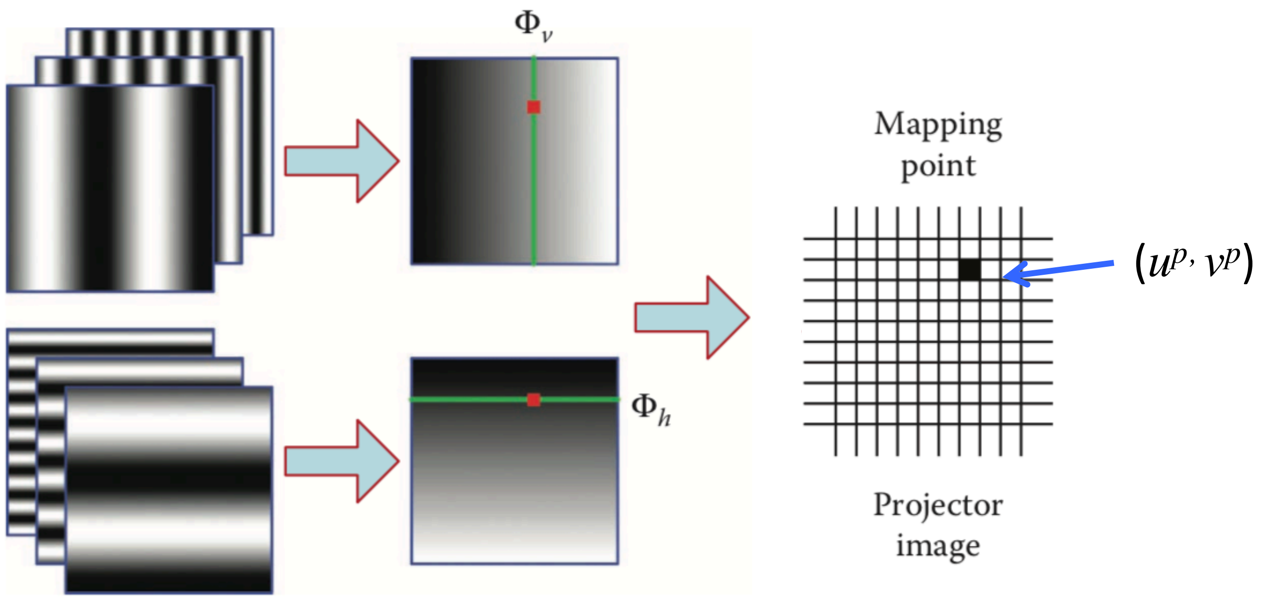
Camera



$$\phi(x,y) = \tan^{-1}\left[\frac{\sqrt{3}(I_1 - I_3)}{2I_2 - I_1 - I_3}\right]$$

Projector correspondence

Project along $\boldsymbol{u}$ and $\boldsymbol{v}$

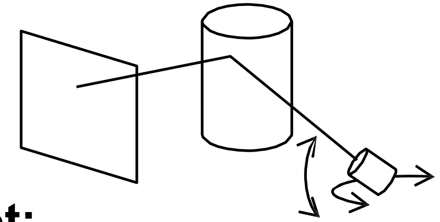


$$\Phi_a(u^c, v^c) = \Phi_a(u^p)$$

$$u^p = \Phi_v \times P/2\pi$$

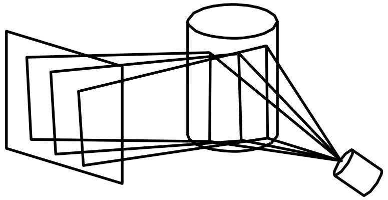
$$v^p = \Phi_h \times P/2\pi$$

Correspondence problem



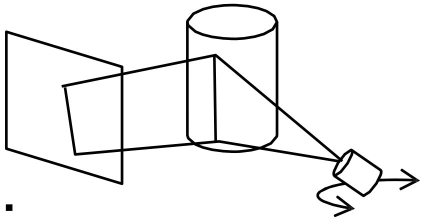
Single dot:

- No correspondence problem.
- Scanning both axis



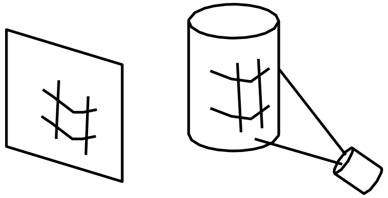
Stripe patterns:

- Correspondence problem among slits
- No scanning



Single stripe:

- Correspondence problem among points of the same slit
- Scanning the axis orthogonal to the stripe

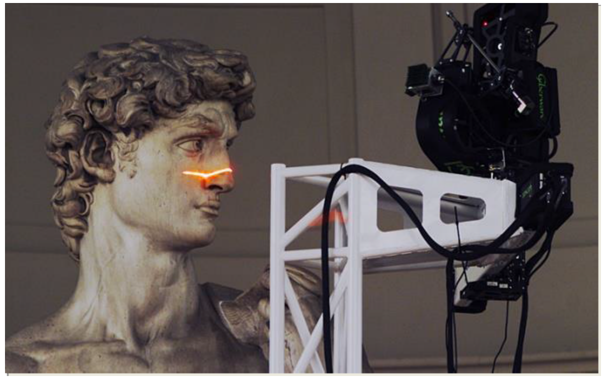


Grid, multiple dots:

- Correspondence problem among all the imaged segments
- No scanning

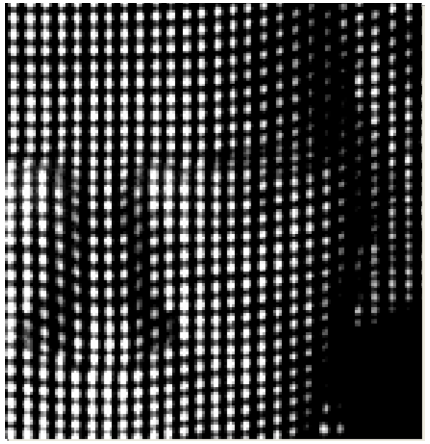
Slide from UdG

Correspondence methods



Single-stripe

Multi-stripe
Multi-frame



Single-frame

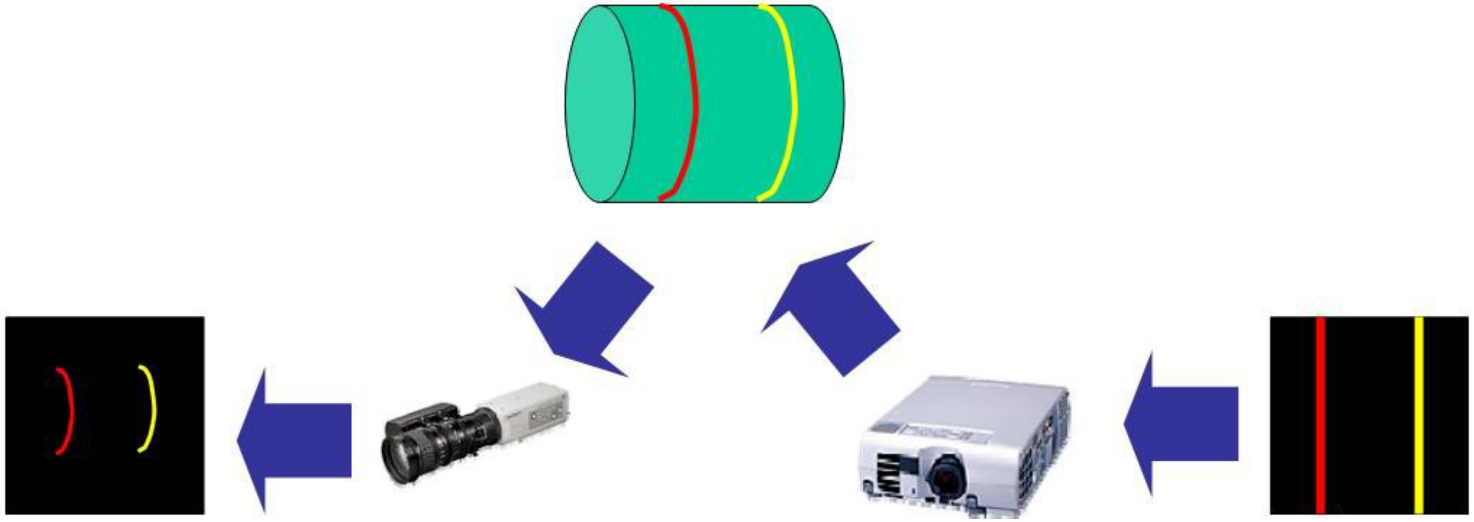
Slow, robust

Fast, fragile

Slide from UdG

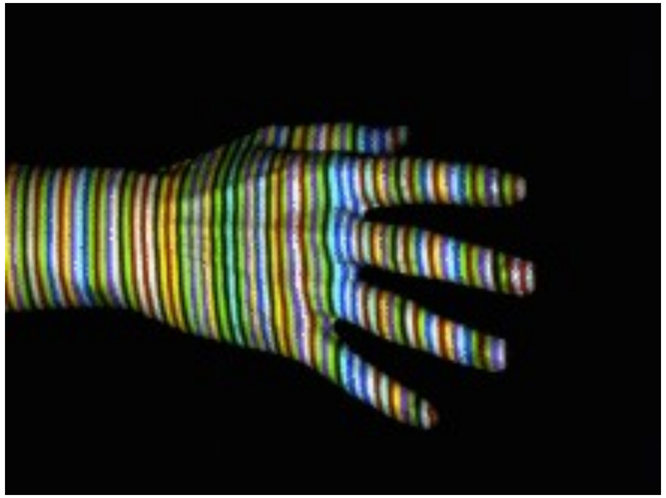
Correspondence methods: encoding

A pattern is called encoded, when after projecting it onto a surface, a set of regions of the observed projection can be easily matched with the original pattern.
Example: encoding by color



Correspondence methods: encoding

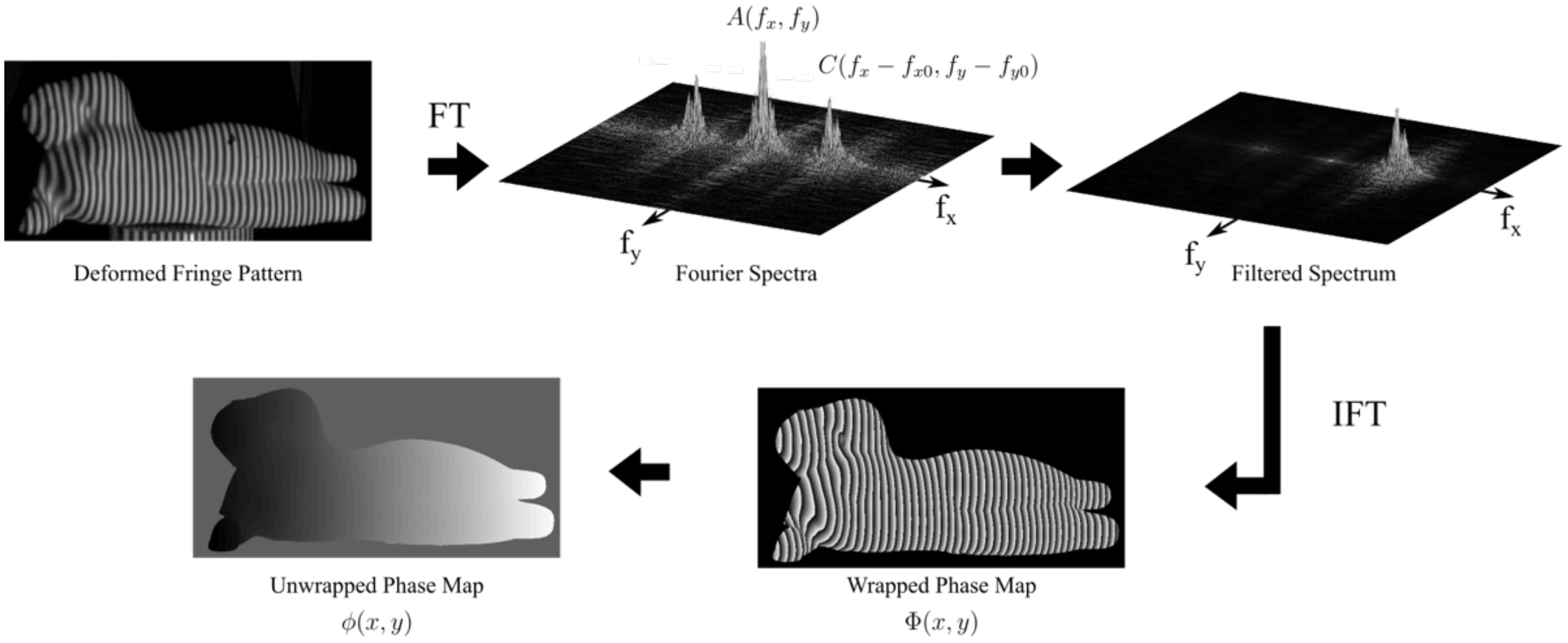
A pattern is called encoded, when after projecting it onto a surface, a set of regions of the observed projection can be easily matched with the original pattern.
Example: encoding by color



Correspondence methods: encoding

A popular codification method is called Fourier Transform Profilometry.

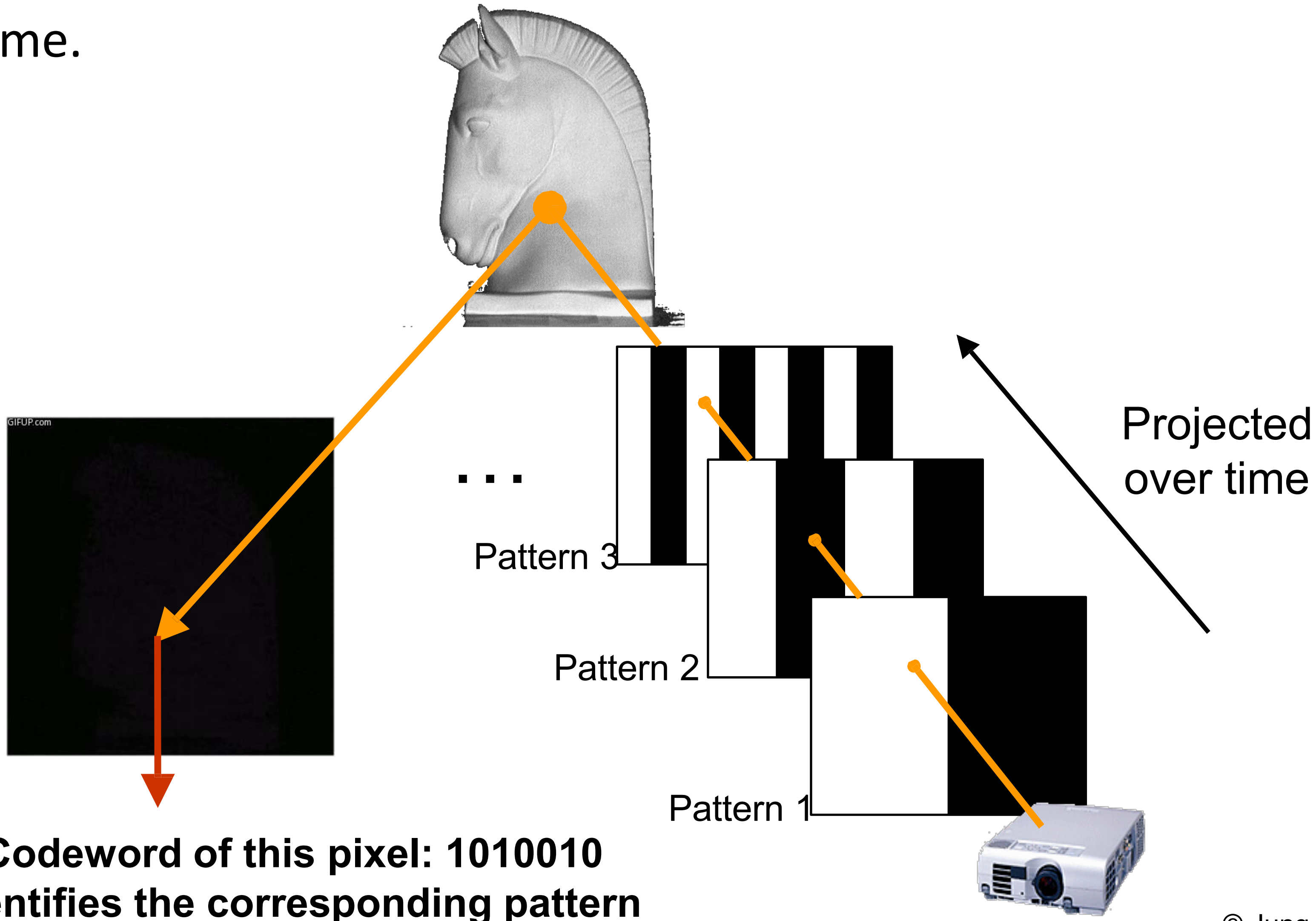
$$g(x,y) = a(x,y) + b(x,y)\cos\left[\omega_0 x + \phi(x,y)\right]$$



A. G. Marrugo et al, Fourier Transform Profilometry in Labview, 2018

Correspondence methods: encoding

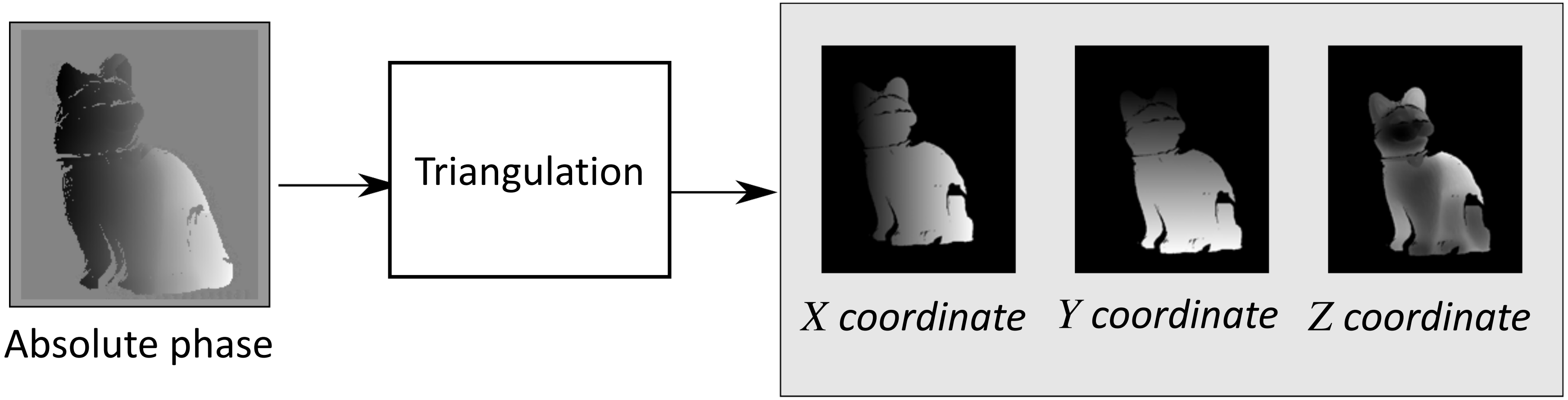
Binary coding in time.



Codeword of this pixel: 1010010 identifies the corresponding pattern stripe

Triangulation method

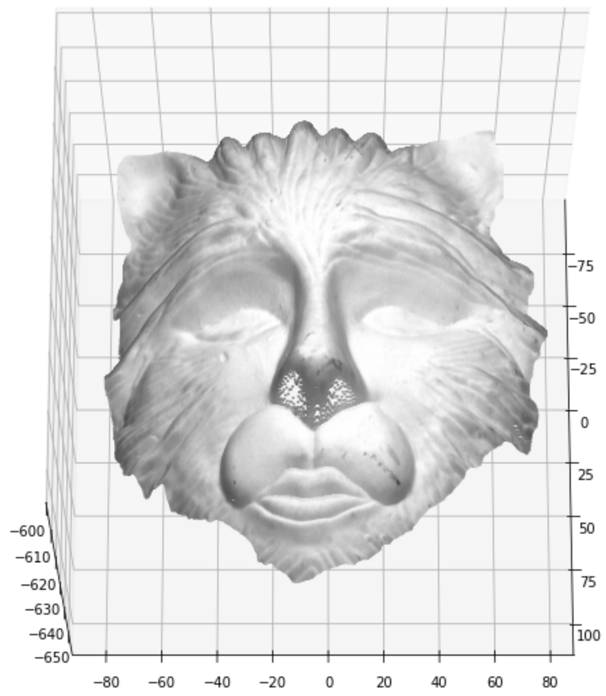
In general, after solving the correspondence problem the 3D reconstruction can be easily obtained.



A. G. Marrugo et al., "State-of-the-art active optical techniques for three-dimensional surface metrology: a review [Invited]." *JOSA A* 37.9 (2020): B60-B77.

Example 3: structured light 3D reconstruction

- In this example, we carry out a 3D reconstruction from a calibrated structured light system.



https://github.com/opi-lab/stsiva-workshop/blob/main/notebooks/stsiva_workshop_notebook03.ipynb

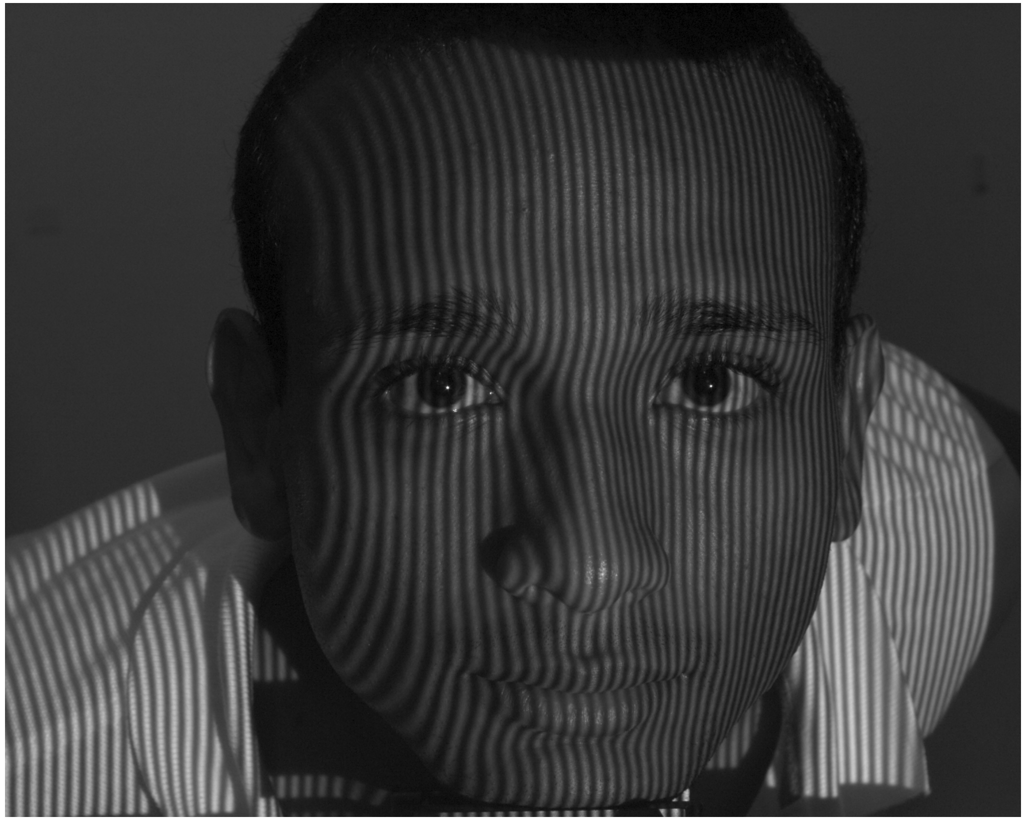
4. Applications

Biomedical applications

The **human skin** does not have significant texture.

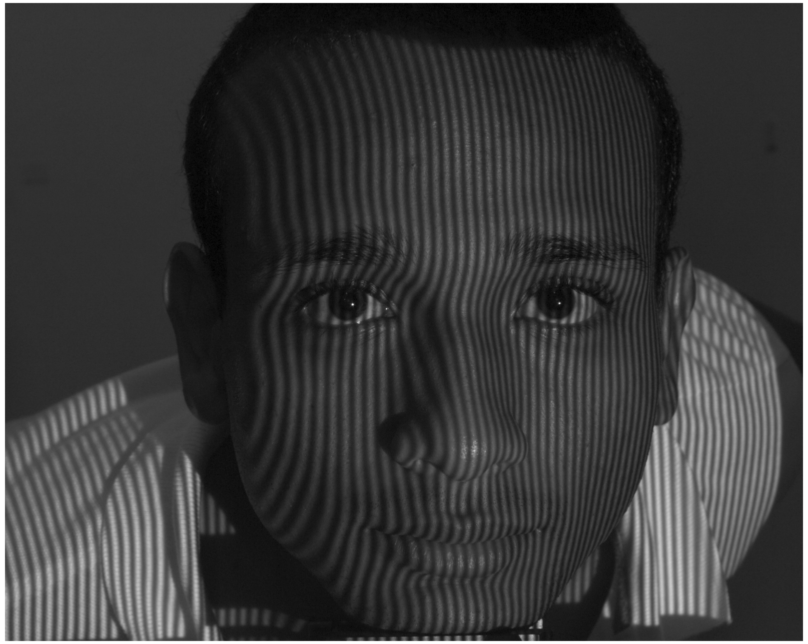
Structured light provides **fast acquisition** (people move).

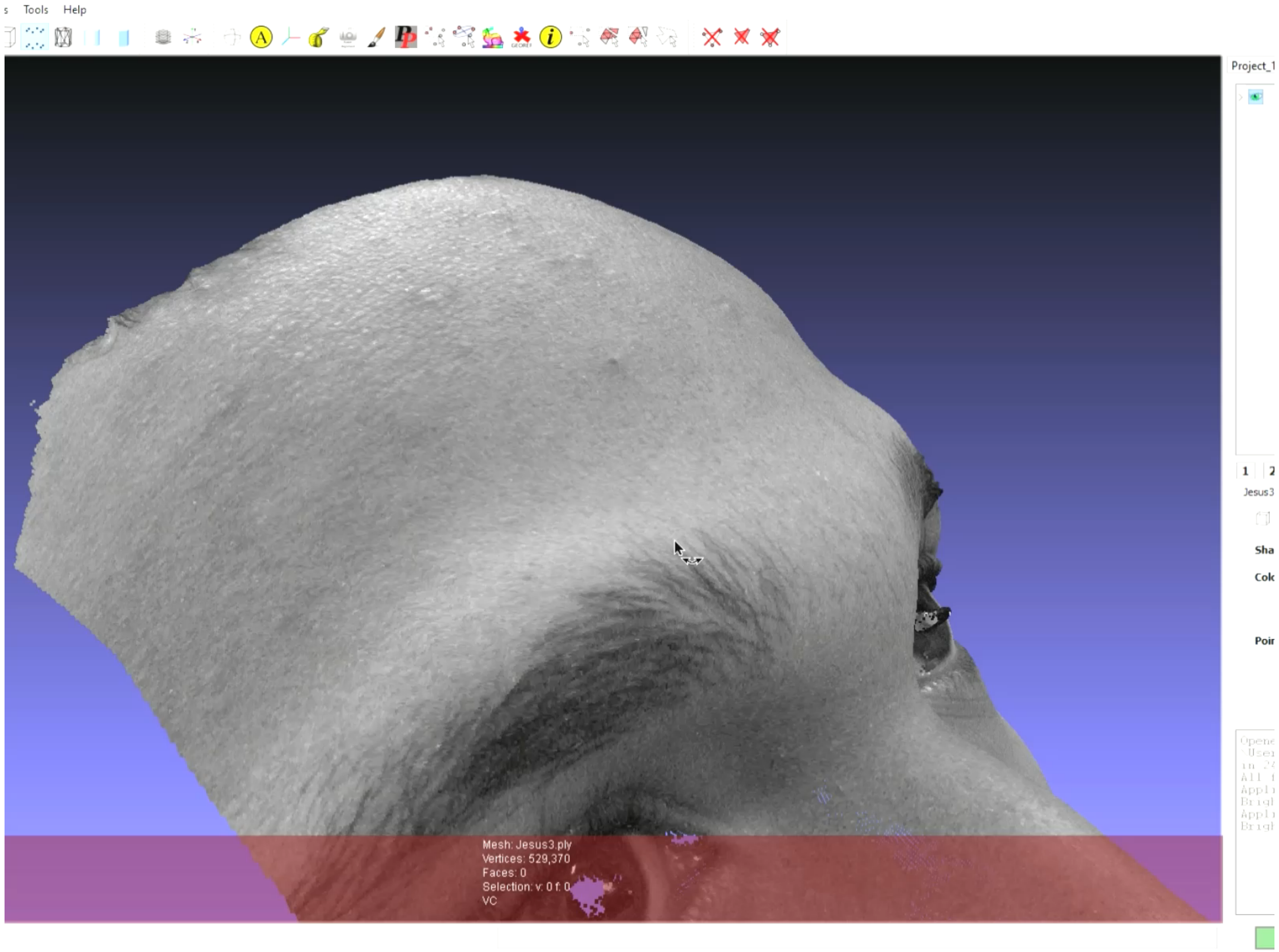
Structured light provides **reliable** measurements with **high accuracy**.



Facial metrology

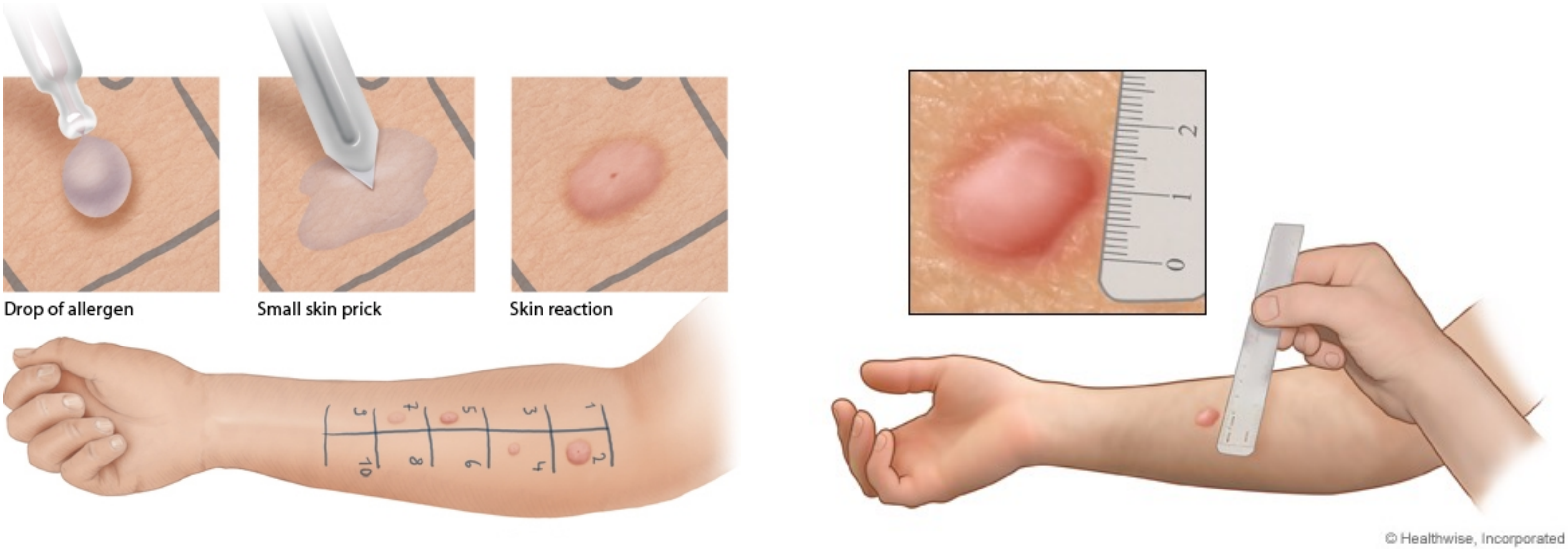
Surgical planning, cosmetic product testing, etc.





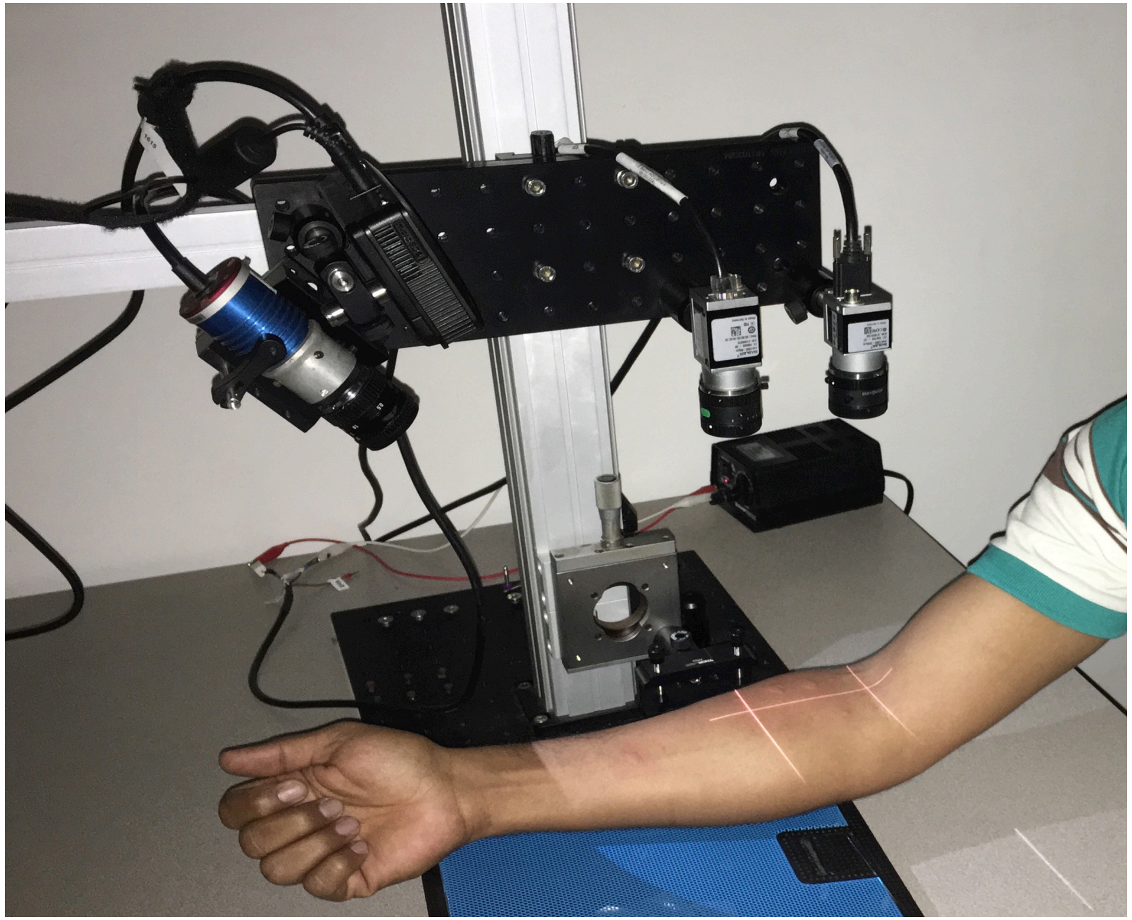
Skin prick test in allergy diagnosis

The local swealing of the skin is a change in 3D!



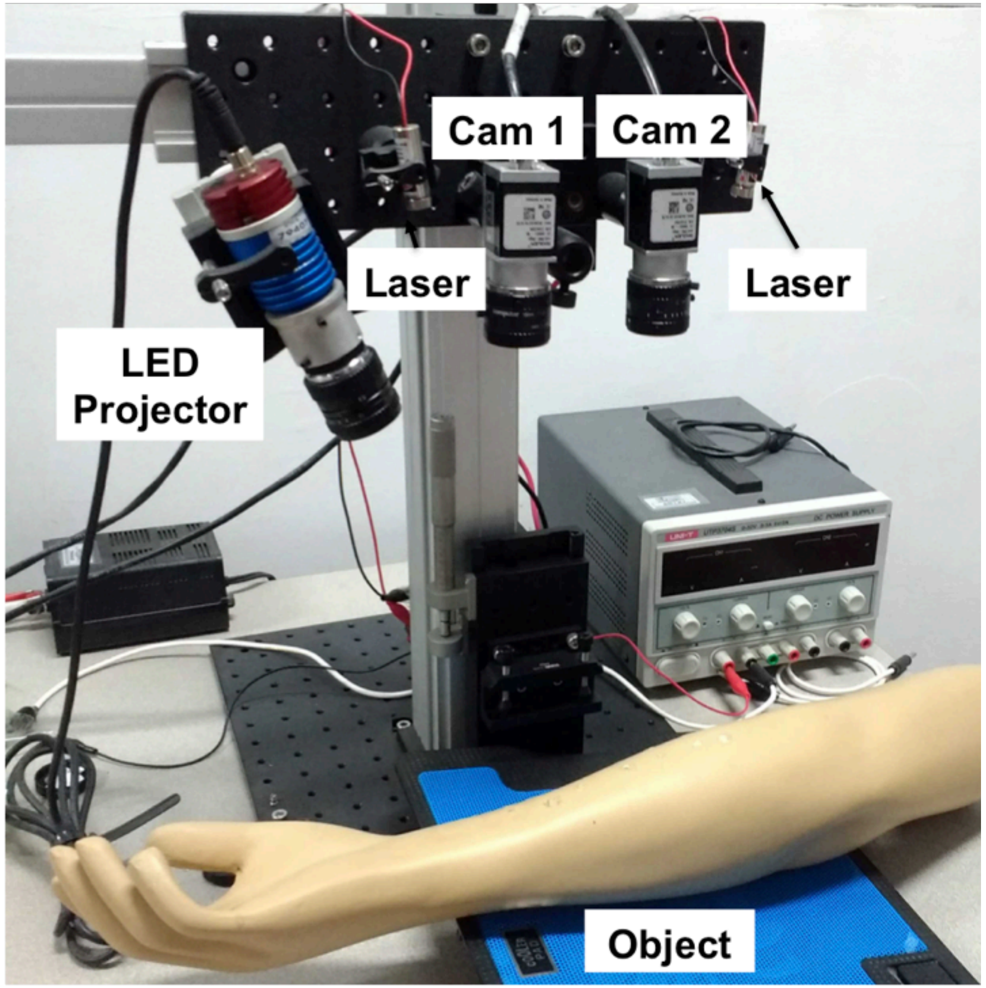
Skin prick test in allergy diagnosis

The 3D structured light system for automated measurement



Pineda et al., PLOS One, 2019

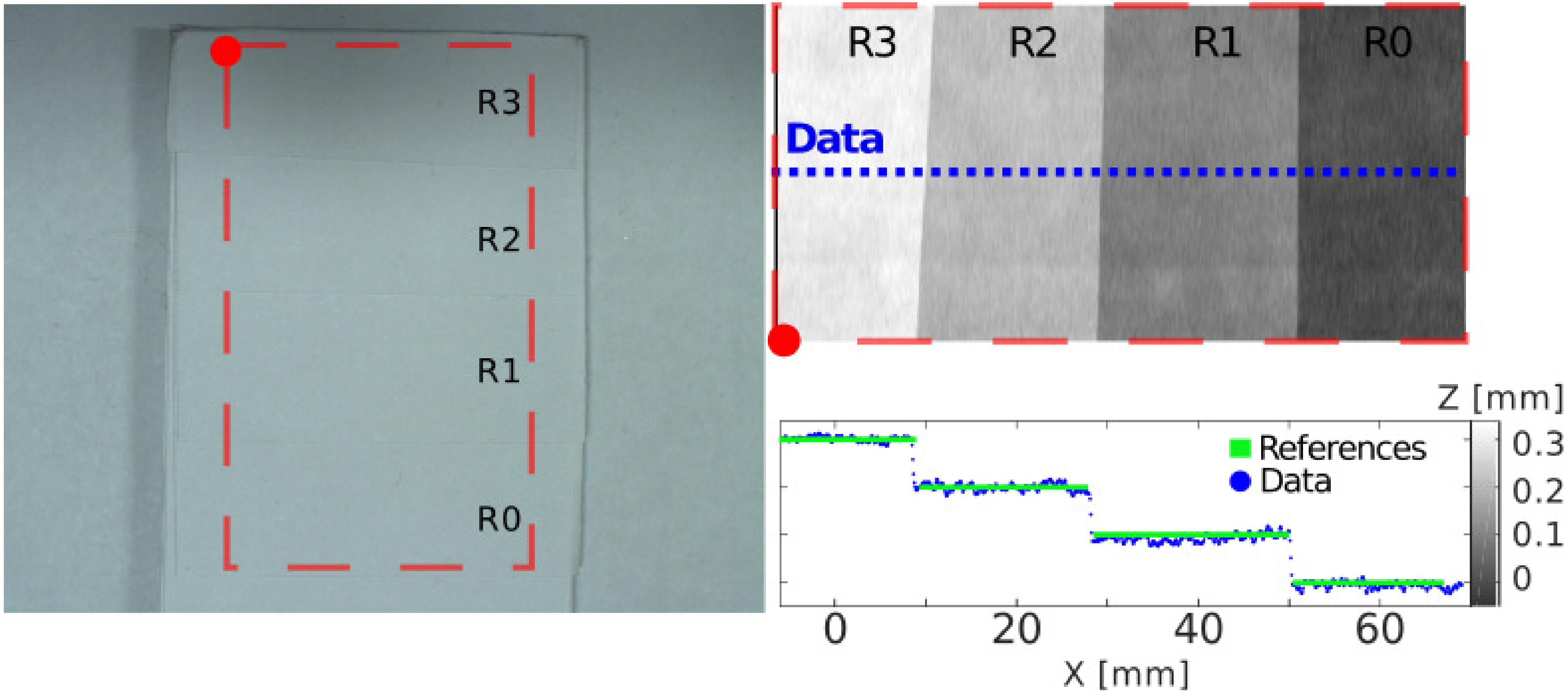
Skin prick test in allergy diagnosis



Pineda et al., PLOS One, 2019

Skin prick test in allergy diagnosis

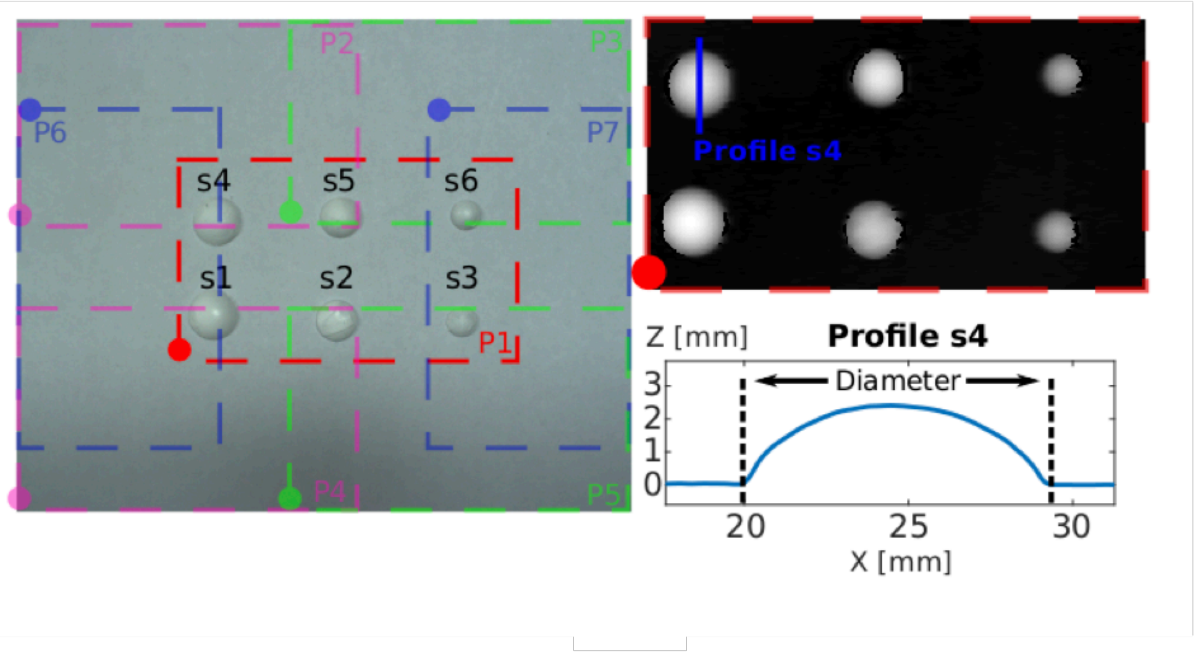
Validation measurements



Pineda et al., PLOS One, 2019

Skin prick test in allergy diagnosis

Validation measurements



Spherical cap	Reference measure	Camera 1 $\mu \pm \sigma$	Camera 2 $\mu \pm \sigma$
s1	9.08 ± 0.01	9.04 ± 0.05	9.02 ± 0.08
s2	7.42 ± 0.01	7.42 ± 0.03	7.53 ± 0.10
s3	5.65 ± 0.01	5.65 ± 0.11	5.65 ± 0.08
s4	8.88 ± 0.01	8.87 ± 0.07	8.85 ± 0.09
s5	7.42 ± 0.01	7.42 ± 0.09	7.48 ± 0.05
s6	5.50 ± 0.01	5.52 ± 0.09	5.53 ± 0.09

Pineda et al., PLOS One, 2019