

Camera Models

CS4243 Computer Vision and Pattern Recognition

Leow Wee Kheng

Department of Computer Science
School of Computing
National University of Singapore



Outline

- 1 Introduction
- 2 Pinhole Camera Model
- 3 Orthographic Projection
- 4 Scaled Orthographic Projection
- 5 Perspective Projection
- 6 Camera Calibration
- 7 Summary
- 8 Exercises
- 9 Further Reading
- 10 Reference

Introduction

3D scene projects to 2D images through the camera's lens.

Camera captures **geometric** and **photometric** info about 3D scene.

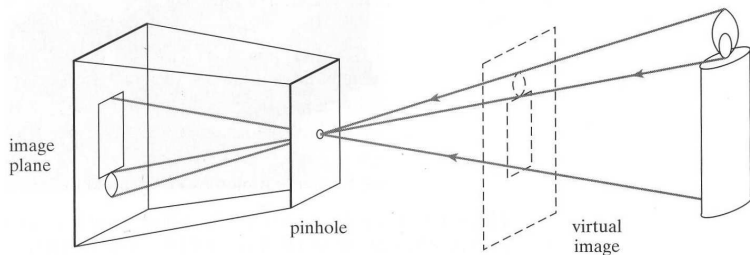
- Geometric: points, lines, curves, etc.
- Photometric: intensity, colour.

Describe relationships between 3D world and 2D images using **models**.

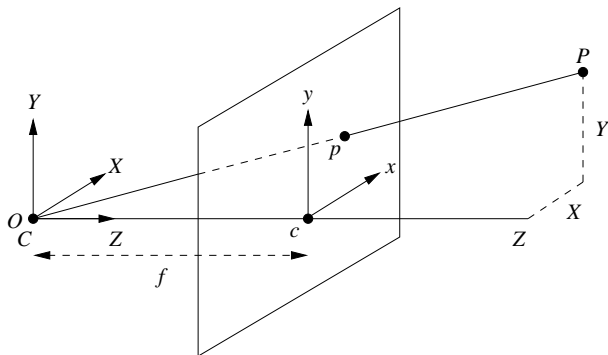
- Actual relationships are very complex.
- Models are approximation of the actual relationships.
- Different models are appropriate for different applications.
- Typically, choose simplest model that offers sufficiently small error.

Pinhole Camera Model

Most common camera model.

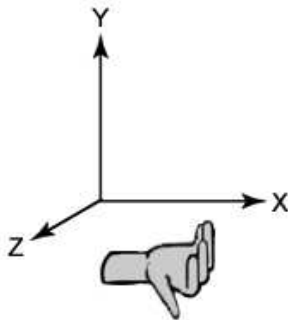
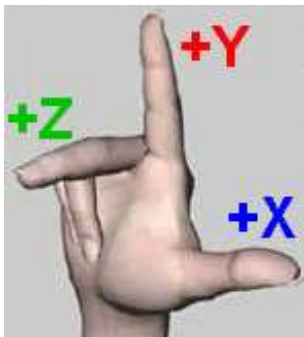


- **Real image plane** lies behind the pinhole.
- Image in real image plane is **inverted**.
- Convenient to consider **virtual image plane** in front of pinhole.
- Virtual image is equivalent to real image and is not inverted.
- Distance between pinhole and image plane is **focal length**.



- **Optical axis** is perpendicular to **image plane** and passes through **optical center C** and **image center c** .
- Origin O of world coordinate frame is at optical center C .
- World coordinate frame's Z -axis is aligned with optical axis.
- Image frame is parallel and aligned to world coordinate frame.
- Origin of image frame is at image center c .
- All projection lines pass through optical center C .

By default, use **right-handed coordinate system**.



3D scene point $P = (X, Y, Z)^\top$ projects to 2D image point $p = (x, y)^\top$.

By symmetry,

$$\frac{X}{Z} = \frac{x}{f}, \quad \frac{Y}{Z} = \frac{y}{f} \quad (1)$$

i.e.,

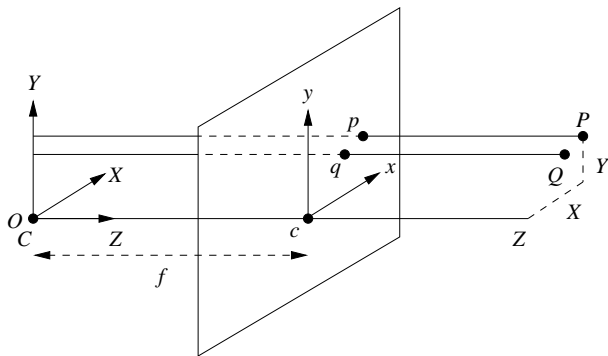
$$x = f \frac{X}{Z}, \quad y = f \frac{Y}{Z}. \quad (2)$$

This is the simplest form of **perspective projection**.

There are four kinds of projection models, from least to most accurate:

- orthographic projection
- scaled orthographic projection or weak perspective projection
- paraperspective projection
- perspective projection

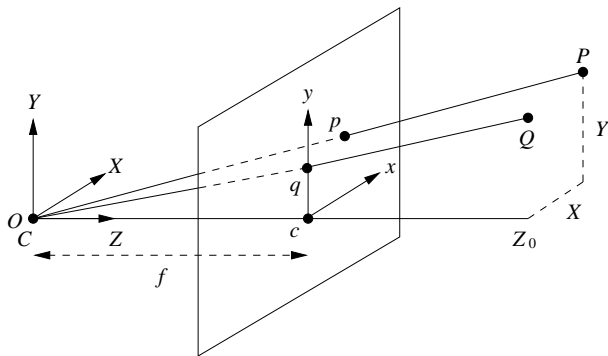
Orthographic Projection



- Camera is at infinite distance from 3D scene.
- All projection lines are parallel to optical axis.

$$x = X, \quad y = Y. \quad (3)$$

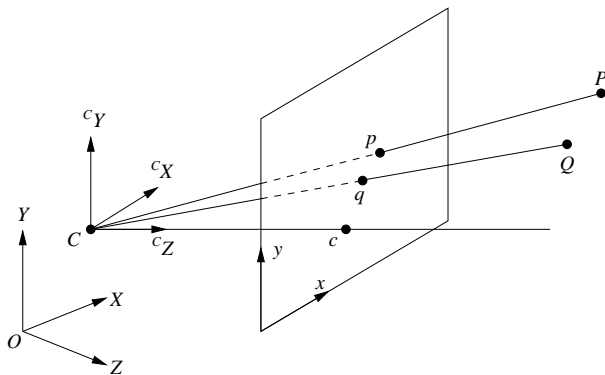
Scaled Orthographic Projection



- Scene depth is small compared to distance to camera.
- Z is approximately the same for all scene points, say Z_0 .

$$x = sX, \quad y = sY, \quad s = \frac{f}{Z_0} \text{ for all scene points.} \quad (4)$$

Perspective Projection



- In general, camera frame is not aligned with world frame.
- Image frame is not at image center.
- Image frame is still parallel to camera frame.

Intrinsic Parameters

Let's begin from the pinhole camera model:

$$x = f \frac{X}{Z}, \quad y = f \frac{Y}{Z}. \quad (5)$$

In real cameras, the pixels may not be exactly square.

So, the scales k and l along the x - and y -axes may be different:

$$x = kf \frac{X}{Z}, \quad y = lf \frac{Y}{Z}. \quad (6)$$

- x and y are expressed in units of **pixels**.
- Focal length f is a distance, expressed in **meters** (or cm, mm).
- Scale parameters k and l are expressed in **pixels/meter**.
- k , l , f are not independent.

Can be replaced by $f_x = kf$ and $f_y = lf$, expressed in **pixels**.

So, we have

$$x = f_x \frac{X}{Z}, \quad y = f_y \frac{Y}{Z}. \quad (7)$$

Optical axis intersects image frame at image center or **principal point** c .

In general, the origin of the image frame is not located at c . Use c_x and c_y to denote the location of c in image frame.

$$x = f_x \frac{X}{Z} + c_x, \quad y = f_y \frac{Y}{Z} + c_y. \quad (8)$$

Along optical axis, $X = Y = 0$.

Then, $x = c_x, y = c_y$, the principal point.

The image frame may not be exactly 90° .

Let θ denote the skew angle between x - and y -axis. Then,

$$x = f_x \frac{X}{Z} - f_x \cot \theta \frac{Y}{Z} + c_x, \quad y = \frac{f_y}{\sin \theta} \frac{Y}{Z} + c_y. \quad (9)$$

Combining all the parameters into a matrix yields

$$\tilde{\mathbf{x}} = \frac{1}{Z} \mathbf{K} \mathbf{X}, \quad \mathbf{K} = \begin{bmatrix} f_x & -f_x \cot \theta & c_x \\ 0 & \frac{f_y}{\sin \theta} & c_y \\ 0 & 0 & 1 \end{bmatrix} \quad (10)$$

or

$$\rho \tilde{\mathbf{x}} = \mathbf{K} \mathbf{X} \quad (11)$$

$\tilde{\mathbf{x}}$ is homogeneous coordinate vector of \mathbf{x} : $\tilde{\mathbf{x}} = [x, y, 1]^\top$, $\rho = Z$.

\mathbf{K} is called the **intrinsic parameter matrix** of the camera.

In some books and papers, $1/Z$ is absorbed into $\tilde{\mathbf{x}}$:
With $\tilde{\mathbf{x}} = [\rho x, \rho y, \rho]^\top$, $\rho = Z$, we have

$$\tilde{\mathbf{x}} = \mathbf{K}\mathbf{X}. \quad (12)$$

So, be careful of which $\tilde{\mathbf{x}}$ is used.

A simpler form of \mathbf{K} is to introduce the skew parameter s :

$$\mathbf{K} = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}. \quad (13)$$

If $s = 0$, then

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}. \quad (14)$$

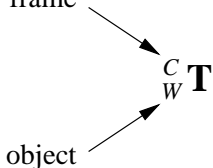
Extrinsic Parameters

In general, camera frame is not aligned with world coordinate frame. There is a rigid transformation between the two frames:

$${}^C\mathbf{X} = {}^C_W\mathbf{R} {}^W\mathbf{X} + {}^C_W\mathbf{T}. \quad (15)$$

- ${}^C\mathbf{X}$: 3D coordinates of scene point P measured in camera frame.
- ${}^W\mathbf{X}$: 3D coordinates of scene point P measured in world frame.
- ${}^C_W\mathbf{R}$: Rotation matrix describing world frame in camera frame.
- ${}^C_W\mathbf{T}$: Position of world frame's origin (${}^W\mathbf{X} = \mathbf{0}$) in camera frame.

Notation: frame



Translation \mathbf{T} :

$$\mathbf{X}' = \mathbf{X} + \mathbf{T}, \quad \mathbf{T} = \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix} \quad (16)$$

Rotation about x -axis $\mathbf{R}_x(\omega)$:

$$\mathbf{X}' = \mathbf{R}_x(\omega) \mathbf{X}, \quad \mathbf{R}_x(\omega) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \omega & -\sin \omega \\ 0 & \sin \omega & \cos \omega \end{bmatrix} \quad (17)$$

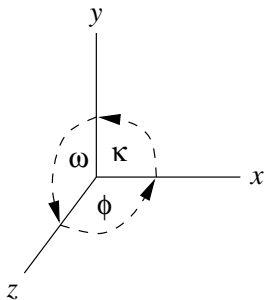
Rotation about y -axis $\mathbf{R}_y(\phi)$:

$$\mathbf{X}' = \mathbf{R}_y(\phi) \mathbf{X}, \quad \mathbf{R}_y(\phi) = \begin{bmatrix} \cos \phi & 0 & \sin \phi \\ 0 & 1 & 0 \\ -\sin \phi & 0 & \cos \phi \end{bmatrix} \quad (18)$$

Rotation about z -axis $\mathbf{R}_z(\kappa)$:

$$\mathbf{X}' = \mathbf{R}_z(\kappa) \mathbf{X}, \quad \mathbf{R}_z(\kappa) = \begin{bmatrix} \cos \kappa & -\sin \kappa & 0 \\ \sin \kappa & \cos \kappa & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (19)$$

Rotation angles in right-handed coordinate system:



- ω : about x -axis, from positive y -axis to positive z -axis
- ϕ : about y -axis, from positive z -axis to positive x -axis
- κ : about z -axis, from positive x -axis to positive y -axis

Combining the three rotations:

$$\mathbf{R} = \mathbf{R}_z(\kappa) \mathbf{R}_y(\phi) \mathbf{R}_x(\omega) = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \quad (20)$$

where

$$\begin{aligned} r_{11} &= \cos \phi \cos \kappa \\ r_{12} &= \sin \omega \sin \phi \cos \kappa - \cos \omega \sin \kappa \\ r_{13} &= \cos \omega \sin \phi \cos \kappa + \sin \omega \sin \kappa \\ r_{21} &= \cos \phi \sin \kappa \\ r_{22} &= \sin \omega \sin \phi \sin \kappa + \cos \omega \cos \kappa \\ r_{23} &= \cos \omega \sin \phi \sin \kappa - \sin \omega \cos \kappa \\ r_{31} &= -\sin \phi \\ r_{32} &= \sin \omega \cos \phi \\ r_{33} &= \cos \omega \cos \phi. \end{aligned} \quad (21)$$

The rotation matrix is an orthonormal matrix:

$$\mathbf{R}^\top \mathbf{R} = \mathbf{I}. \quad (22)$$

So, $\mathbf{R}^{-1} = \mathbf{R}^\top$. In program, should use \mathbf{R}^{-1} for accuracy.

In particular, let

$$\mathbf{R} = \begin{bmatrix} \mathbf{R}_1^\top \\ \mathbf{R}_2^\top \\ \mathbf{R}_3^\top \end{bmatrix}. \quad (23)$$

Then (Exercise),

$$\mathbf{R}_i^\top \mathbf{R}_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases} \quad (24)$$

Now, let's look at it from the view point of the world frame:

$${}^W\mathbf{X} = {}^W\mathbf{R} {}^C\mathbf{X} + {}^W\mathbf{T} \quad (25)$$

- ${}^W\mathbf{R}$: Rotation matrix describing camera frame in world frame.
- ${}^W\mathbf{T}$: Position of camera frame's origin (${}^C\mathbf{X} = \mathbf{0}$) in world frame.

Rearranging Eq. 15 yields

$${}^W\mathbf{X} = {}^C\mathbf{R}^{-1} {}^W\mathbf{X} - {}^C\mathbf{R}^{-1} {}^W\mathbf{T}. \quad (26)$$

So, ${}^C\mathbf{R} = {}^W\mathbf{R}^{-1}$, ${}^C\mathbf{T} = -{}^W\mathbf{R}^{-1} {}^W\mathbf{T}$.

Let ${}^W\mathbf{C}$ denote position of camera center in world frame. Then,

$${}^W\mathbf{C} = -{}^C\mathbf{R}^{-1} {}^C\mathbf{T}. \quad (27)$$

Geometric Model

Combining intrinsic and extrinsic parameters yield the general perspective projection model:

$$\rho \tilde{\mathbf{x}} = \mathbf{K}^C \mathbf{X} = \mathbf{K} \left({}^C_W \mathbf{R}^W \mathbf{X} + {}^C_W \mathbf{T} \right). \quad (28)$$

Simpler notation:

$$\rho \tilde{\mathbf{x}} = \mathbf{K}(\mathbf{R}\mathbf{X} + \mathbf{T}) = \mathbf{K} [\mathbf{R} \mid \mathbf{T}] \tilde{\mathbf{X}} = \mathbf{K} \mathbf{R}(\mathbf{X} - \mathbf{C}) \quad (29)$$

where $\tilde{\mathbf{X}} = [X, Y, Z, 1]^T$.

Be careful of the meanings of the symbols in the absence of the superscripts and subscripts.

Lens Distortion

Lens can have distortion, especially at short focal length.
The most common type is **radial distortion**.



(a)



(b)



(c)

- (a) barrel distortion
- (b) pincushion distortion
- (c) fisheye distortion

We examine the case in which $s = 0$.

Let (x, y) denote the undistorted image coordinates before scaling by f_x, f_y and shifted by (c_x, c_y) . Then (Exercise),

$$\begin{aligned}x &= \frac{\mathbf{R}_1^\top \mathbf{X} + T_X}{\mathbf{R}_3^\top \mathbf{X} + T_Z} \\y &= \frac{\mathbf{R}_2^\top \mathbf{X} + T_Y}{\mathbf{R}_3^\top \mathbf{X} + T_Z}.\end{aligned}\tag{30}$$

Radial distortion can be modeled as

$$\begin{aligned}x_d &= x(1 + \kappa_1 r^2 + \kappa_2 r^4 + \kappa_3 r^6) \\y_d &= y(1 + \kappa_1 r^2 + \kappa_2 r^4 + \kappa_3 r^6)\end{aligned}\tag{31}$$

where $r^2 = x^2 + y^2$, and $\kappa_1, \kappa_2, \kappa_3$ are the **radial distortion parameters**.

Then, the actual image coordinates (x_a, y_a) are given by

$$\begin{aligned}x_a &= f_x x_d + c_x \\y_a &= f_y y_d + c_y.\end{aligned}\tag{32}$$

Sometime, Eq. 31 is expressed the other way around (e.g., OpenCV):

$$\begin{aligned}x &= x_d(1 + \kappa_1 r_d^2 + \kappa_2 r_d^4 + \kappa_3 r_d^6) \\y &= y_d(1 + \kappa_1 r_d^2 + \kappa_2 r_d^4 + \kappa_3 r_d^6)\end{aligned}\tag{33}$$

where $r_d^2 = x_d^2 + y_d^2$.

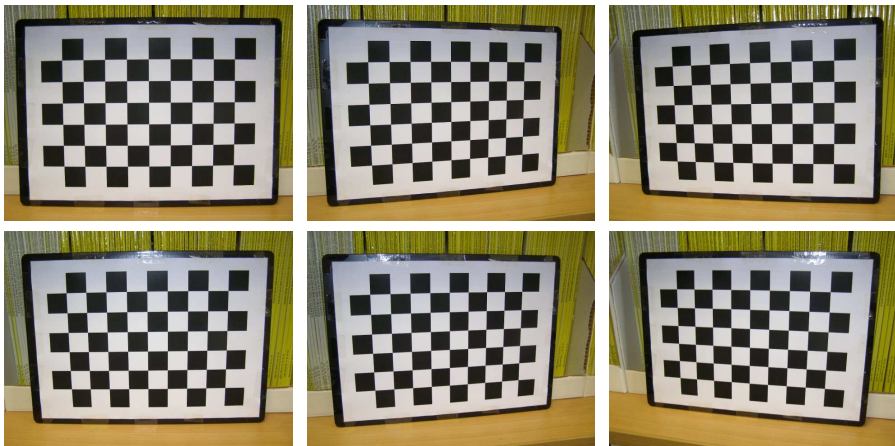
Tangential distortion is modeled as

$$\begin{aligned}x &= x_d + (2p_1 y_d + p_2(r_d^2 + 2x_d^2)) \\y &= y_d + (p_1(r_d^2 + 2y_d^2) + 2p_2 x_d)\end{aligned}\tag{34}$$

where p_1, p_2 are **tangential distortion parameters**.

Camera Calibration

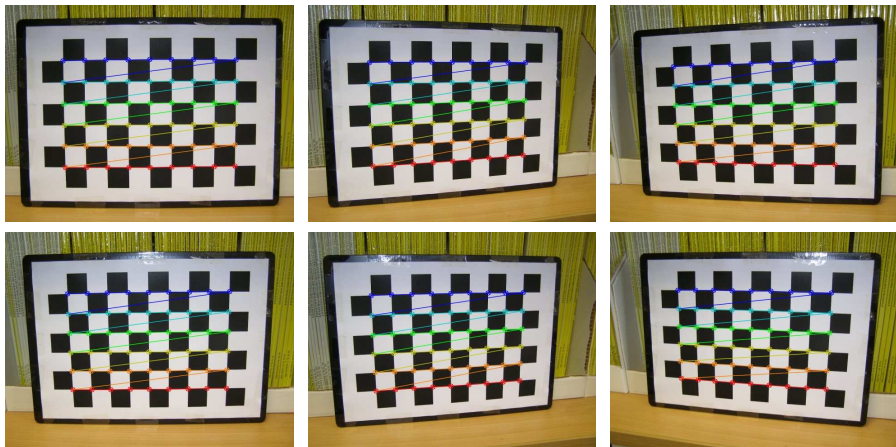
Compute intrinsic parameters and lens distortion parameters.
Capture calibration pattern from various view points:



Notes on using OpenCV calibration function:

- OpenCV camera calibration algorithm is based on **homography**.
- When camera motion is pure rotation, no translation, then two image views are related by a homography.
- So, should try to just rotate image or camera when taking various views.
- Keep center of view fixed at same point in calibration pattern. Slight translation is ok.

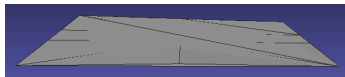
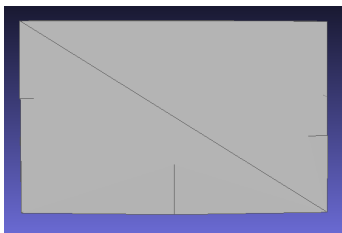
Then, detect inner corners in the calibration images:



Then, run calibration program to obtain intrinsic parameters and distortion parameters.

If calibration is accurate, recovered 3D points of calibration images should

- fall on a plane,
- form a rectangle.



For additional accuracy, can apply **undistortion** to get undistorted coordinates.

Summary

- Simplest camera model: pinhole camera model.
- Most commonly used camera model: perspective camera model.
- Intrinsic parameters: focal length, principal point.
- Extrinsic parameters: rotation, translation.
- Lens distortion: radial, tangential.
- Camera calibration: compute extrinsic and distortion parameters.

Exercises

(1) Show that the rotation matrix \mathbf{R} is orthonormal:

$$\mathbf{R}_i^\top \mathbf{R}_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases} \quad (35)$$

(2) Derive Eq. 30 for the case $s = 0$.

Software Available

Camera calibration tool

- OpenCV: Python version available in course website.
- Matlab toolbox: www.vision.caltech.edu/bouguetj/calib_doc/
- GML C++ toolbox:
graphics.cs.msu.ru/en/science/research/calibration/cpp





Image undistortion

- OpenCV

Further Reading

- Camera models: [Sze10] Section 2.1.5, [FP03] Section 1.1, 1.2, 2.2, 2.3.
- Lens distortion: [Sze10] Section 2.1.6, [BK08] Chapter 11.
- Camera calibration: [BK08] Chapter 11, [Zha00], [Bou].
- Image undistortion: [BK08] Chapter 11.

Reference I

-  Bradski and Kaehler.
Learning OpenCV: Computer Vision with the OpenCV Library.
O'Reilly, 2008.
-  J.-Y. Bouguet.
Camera calibration toolbox for Matlab.
-  D. A. Forsyth and J. Ponce.
Computer Vision: A Modern Approach.
Pearson Education, 2003.
-  R. Szeliski.
Computer Vision: Algorithms and Applications.
Springer, 2010.

Reference II



Z. Zhang.

A flexible new technique for camera calibration.

IEEE Trans. PAMI, 22(11):1330–1334, 2000.