



# Camera Calibration

Camera Geometry and The Pinhole Model

# Camera calibration

Camera calibration estimates the **parameters of a pinhole camera model** given photograph.

The pinhole camera parameters are represented in a  $3 \times 4$  matrix called the camera matrix.

These parameters are used to **estimate the actual size of an object** or **determine the location of the camera in the world**

# The Pinhole Model

The pinhole camera model explains the relationship between a point in the world and the projection on the image plane (image sensor).

If we use a wide-open camera sensor, we will end up with blurry images, because the imaging sensor collects light rays from multiple points on the object at the same location on the sensor.

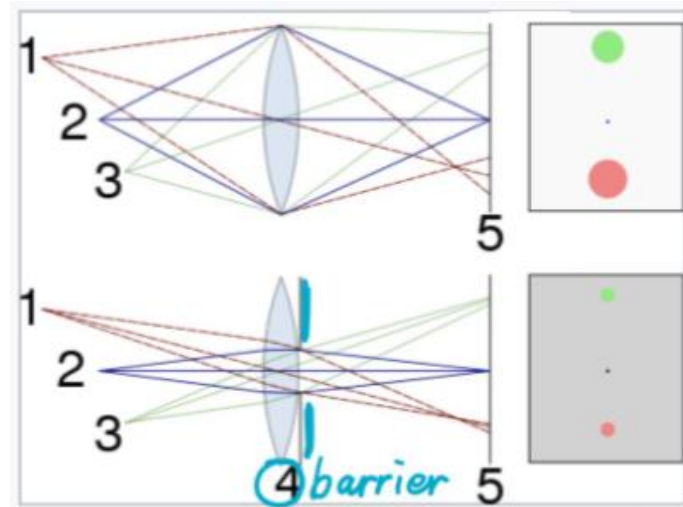
The solution to this problem is to put a barrier in front of the imaging sensor with a tiny hole.


The barrier allows only a limited number of light rays to pass through the hole, and reduces the blurriness of the image.

# DOF – Depth of Field

1

2



Effect of aperture on blur and DOF. 

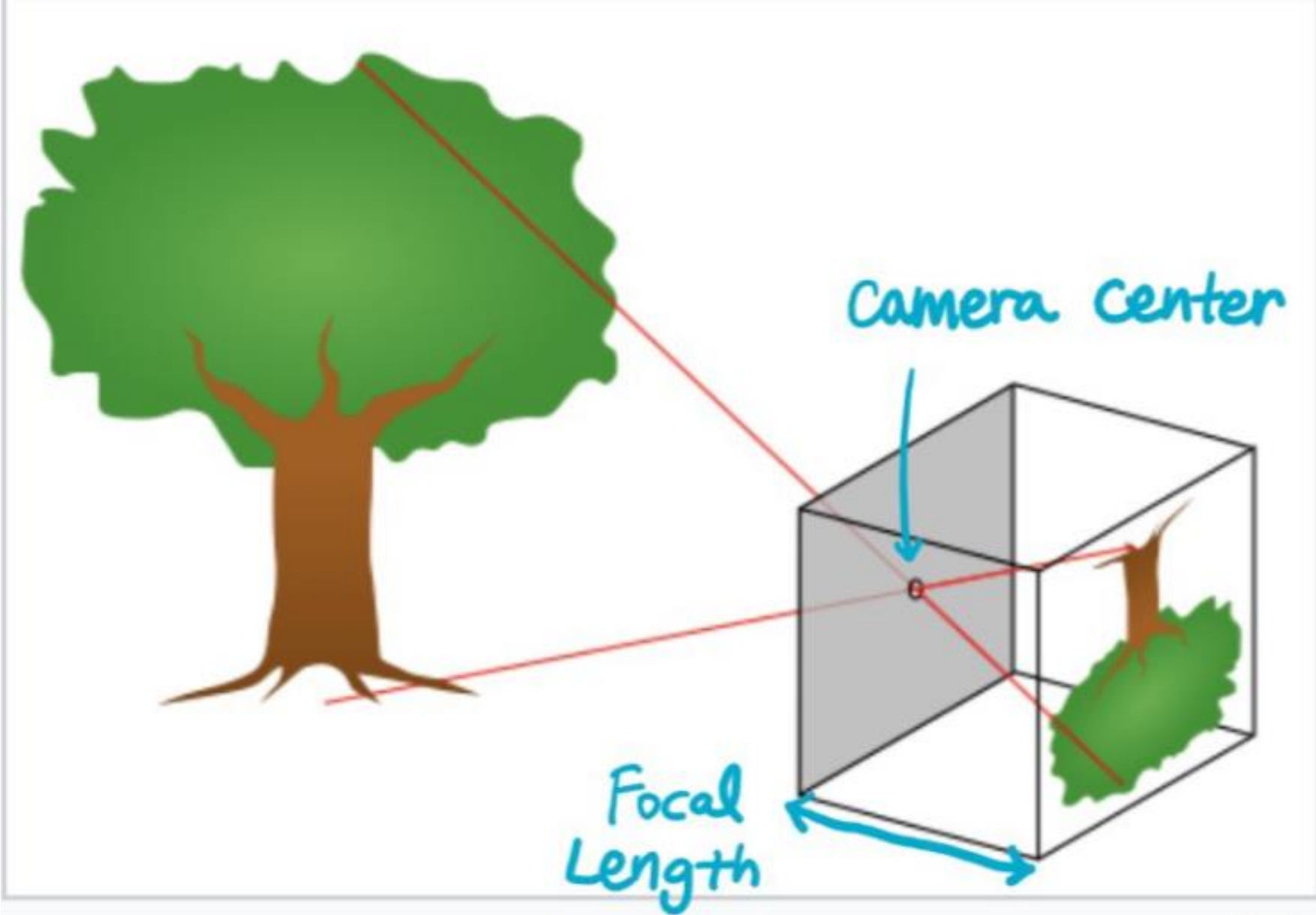
The points in focus (2) project points onto the image plane (5), but points at different distances (1 and 3) project blurred images, or **circles of confusion**. Decreasing the aperture size (4) reduces the size of the blur spots for points not in the focused plane, so that the blurring is imperceptible, and all points are within the DOF.

# The two most important parameters in a pinhole camera model

**1. Focal length:** the distance between the pinhole and the image plane

- It affects **the size of the projected image**.
- It affects the camera **focus** when using lenses.


**2. Camera center:** The coordinates of the center of the **pinhole**.



A diagram of a **pinhole camera**.



The pinhole camera model is very straightforward. By knowing the focal length and camera's center, we can mathematically calculate the location where a ray of light that is reflected from an object will strike the image plane.



The focal length and the camera center are the camera **intrinsic parameters**,  $K$ . ( $K$  is an industry norm to express the intrinsic matrix.)

## Intrinsic parameters

(Aka, the camera matrix.)

$(C_x, C_y)$  : camera center in pixels.

$(f_x, f_y)$  : Focal length in pixels.

$$f_x = F/p_x$$

$$f_y = F/p_y$$

$F$  : Focal length in world units (e.g. millimeters.)

$$\begin{bmatrix} f_x & 0 & 0 \\ s & f_y & 0 \\ c_x & c_y & 1 \end{bmatrix}$$

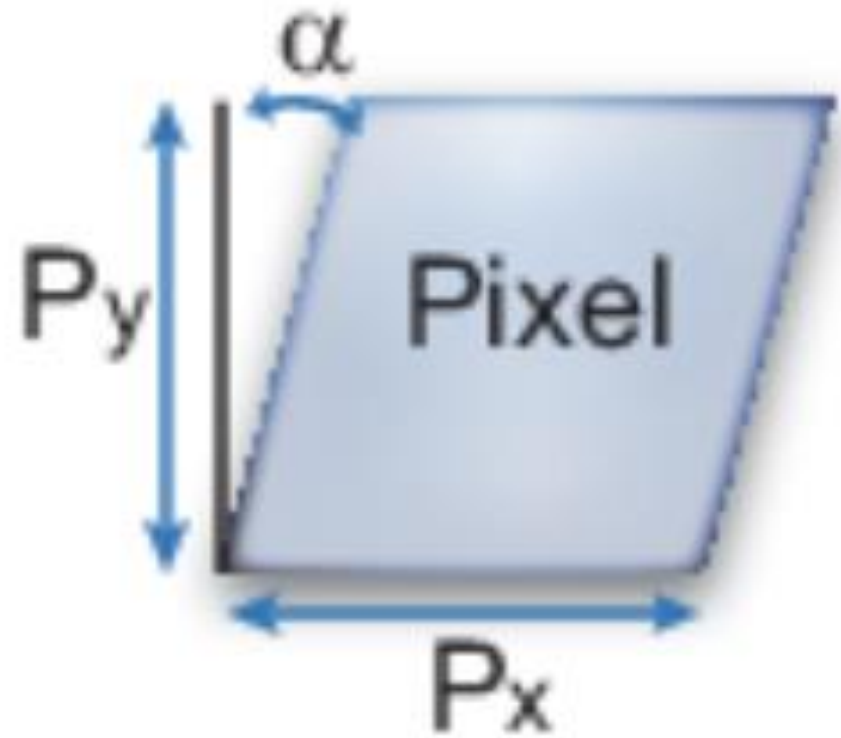
intrinsic parameters,  $K$ .



$(P_x, P_y)$  : Size of the pixel in world units.

$s$  : Skew coefficient, which is non-zero if the image axes are not perpendicular.

$$s = f_x \tan(\alpha)$$



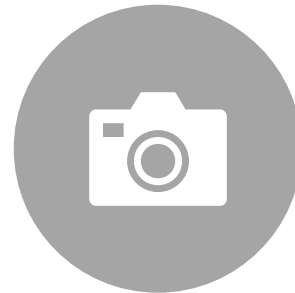
Skew

Pixel Skew

# Coordinate System Transformation (via Matrix Algebra!)



Why do we want this?



**In order to project the point in the world frame to the camera image plane!**



To do what?

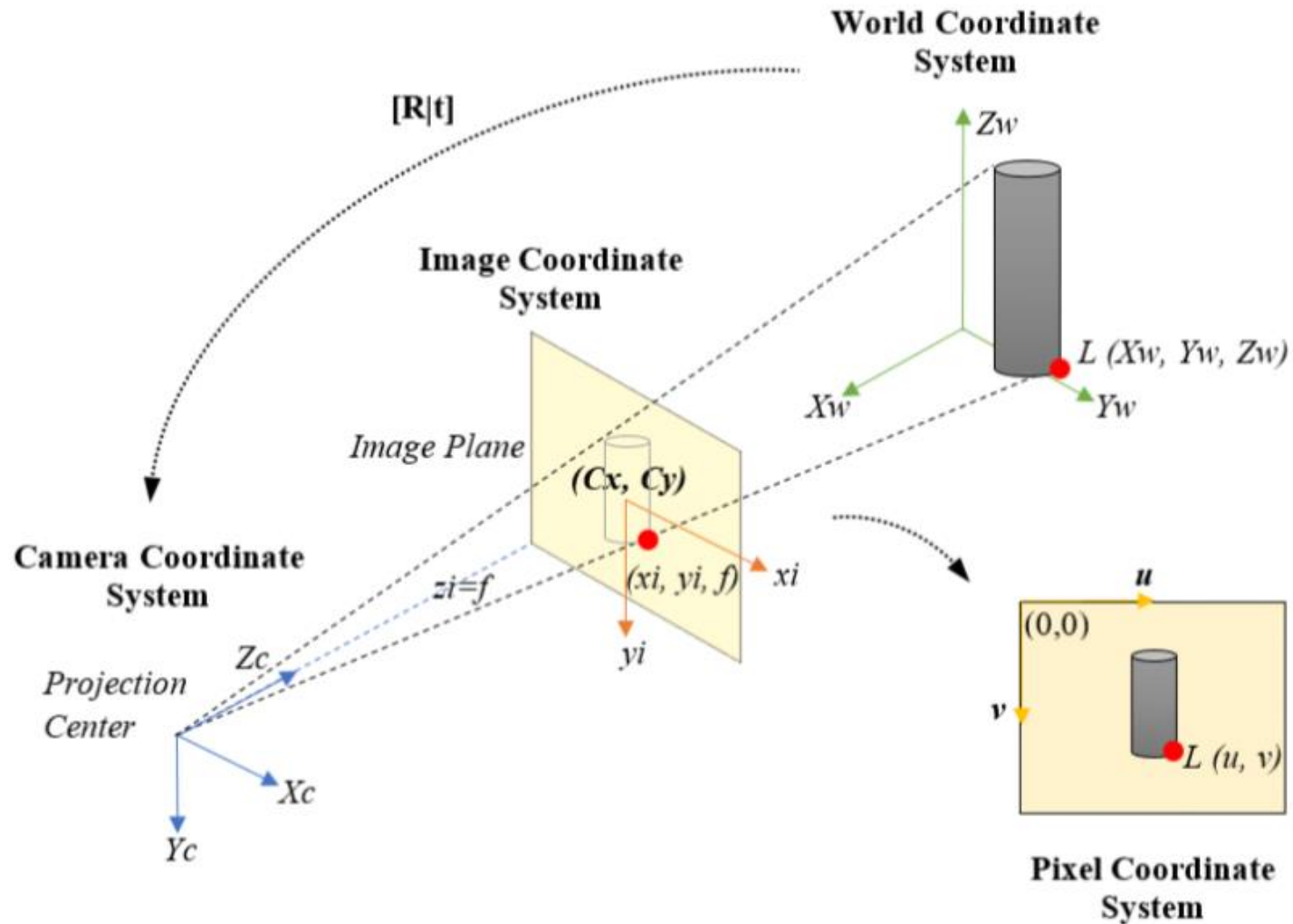


**(If we are talking about self-driving cars) To localize self-driving cars!**

# Coordinate System Transformation

- **Light (reflected from the object)** travels from the world through the camera aperture (pinhole) to the sensor surface.
- The projection onto the sensor surface through the aperture results in flipped images.
- To avoid the flipping confusion, we define a **virtual image plane** (yellow plane) in front of the camera center.

# Camera model projection



```
# World Coordinate System
Oworld = [Xw, Yw, Zw]

# Camera Coordinate System
Ocamera = [Xc, Yc, Zc]

# Pixel Coordinate System
Oimage = [u,v]
```

- We define a 3 by 3 **rotation matrix (R)** and a 3 by 1 **translation vector (t)** in order to model ANY transformation between a world coordinate system and another.
- Now we can frame the projection problem (World Coordinates → Image Coordinates) as

**1. World coordinates → Camera coordinates**

**2. Camera coordinates → Image coordinate**

Oworld [Xw,Yw,Zw] → Oimage [u,v]

# How? by using Linear Algebra!

1. World coordinates  $\rightarrow$  Camera coordinates

$$O_{\text{camera}} = [R|t] * O_{\text{world}}$$

2. Camera coordinates  $\rightarrow$  Image coordinate

$$O_{\text{image}} = K * O_{\text{camera}}$$

Remind me what  $K$  (camera intrinsic parameter) was?

$$\begin{bmatrix} f_x & 0 & 0 \\ s & f_y & 0 \\ c_x & c_y & 1 \end{bmatrix}$$

intrinsic parameters,  $K$ :  $f$  for focal length,  $c$  for camera center, which are camera specific params

# Both steps 1 and 2 are just matrix multiplications.

$$O_{\text{image}} = P * O_{\text{world}} = K[R|t] * O_{\text{world}}$$

Let  $P = K[R|t]$   
P as Projection

Wait,  $K$  is (3,3) matrix.  $[R|t]$  is (3,4). (| means you are concatenating matrix  $R$  with vector  $t$ .)  $O_{\text{world}}$   $[X_w, Y_w, Z_w]$  is (3,1).

Then you can't multiply  $K[R|t]$  (3,4) with  $O_{\text{world}}$   $[X_w, Y_w, Z_w]$  (3,1)!

🕶 We can resolve this by adding one at the end the  $O_{\text{world}}$  vector  $[X_w, Y_w, Z_w, 1]$ , called **homogeneous coordinate (or projective coordinate)**.

- This simple projection principle will be used in every 3d visual perception algorithm, from object detection to 3d scene reconstruction.
- In real life, there will be more complex scenarios, e.g. non-square pixels, camera access skew, distortion, non-unit aspect ratio, etc. However, **they only change the camera matrix  $K$** , and the equations will still be the same.



# A few things to note:

- a) The **rotation matrix** ( $\mathbf{R}$ ) and the **translation vector** ( $\mathbf{t}$ ) are called **extrinsic parameters** because they are "external" to the camera.
- The translation vector  $\mathbf{t}$  can be interpreted as the position of the world origin **in camera coordinates**, and the columns of the rotation matrix  $\mathbf{R}$  represent the directions of the world-axes **in camera coordinates**. This can be a little confusing to interpret because we are used to thinking in terms of the world coordinates.
- b) Usually, multiple sensors (e.g. camera, lidar, radar, etc.) are used for perception in self-driving vehicles. Each sensor comes with its own extrinsic parameters that define the transform from the sensor frame to the vehicle frame.
- c) Image coordinate (virtual image plane)  $[\mathbf{u}, \mathbf{v}]$  starts from the top left corner of the virtual image plane. That's why we adjust the pixel locations to the image coordinate frame.