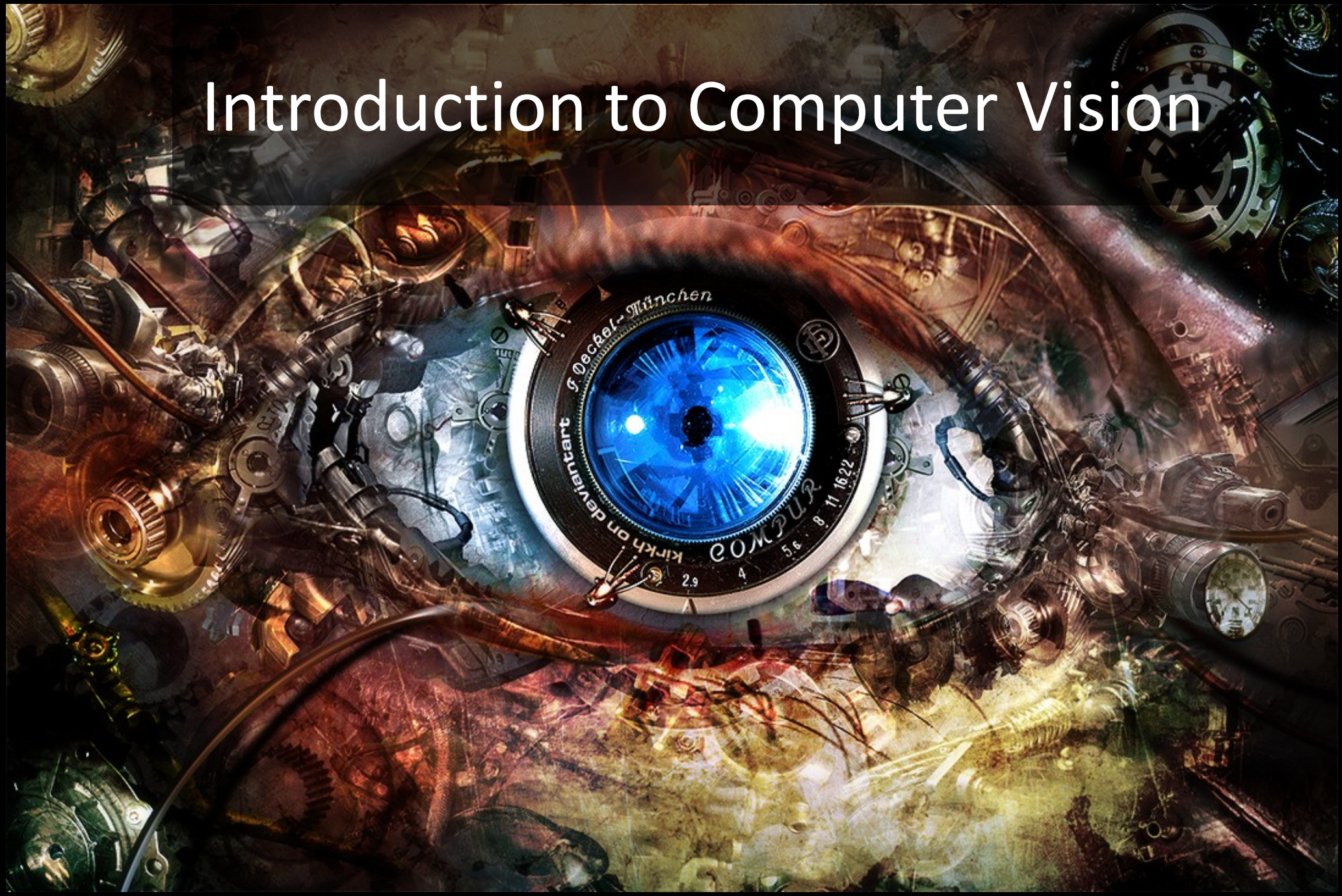
Introduction to Computer Vision



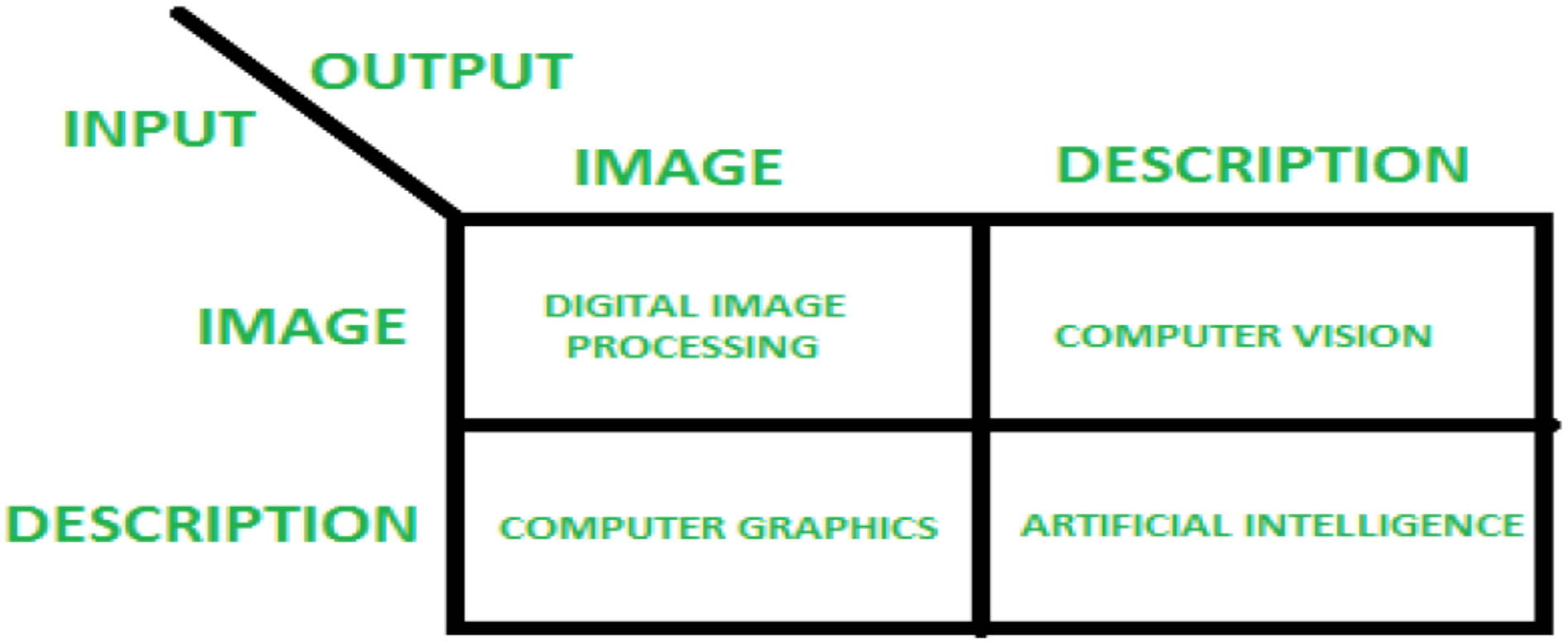
What is Computer Vision?

- Computer vision is a rapidly growing field of artificial intelligence (AI) that enables machines to understand and interpret visual information from real world.
- Computer Vision, often abbreviated as CV, is defined as a field of study that seeks to develop techniques to help computers see and understand the content of digital images and videos.
- The goal of computer vision is to understand the content of digital images. Typically, this involves developing methods that attempt to reproduce the capability of human vision.

What is Computer Vision

- Automatically identifying objects in images or video
- Extracting latent information from visual data
- Technology that interprets light stimuli
- Computers seeing/learning things that the programmers who made them didn't tell them
- Mimicking the human perception of sight with computational algorithms
- Train computers to understand the visual world
- The study of understanding the world through visual perception
- Converting images to more understandable things like distance, edges, directions etc.
- Computer getting information out of images/video
- Giving the computer "eyes" to see and identify as humans would

Prospective of Study



Computer Vision and Image Processing

- Computer vision is different from image processing.
- Image processing is the process of creating a new image from an existing image, typically simplifying or enhancing the content in some way.
- Computer vision is the automated extraction of information from images.
- A given computer vision system may require image processing to be applied to raw input, e.g. pre-processing images. The pre-processing includes normalizing brightness or color, cropping the boundary of the ROI, removing digital noise from an image.

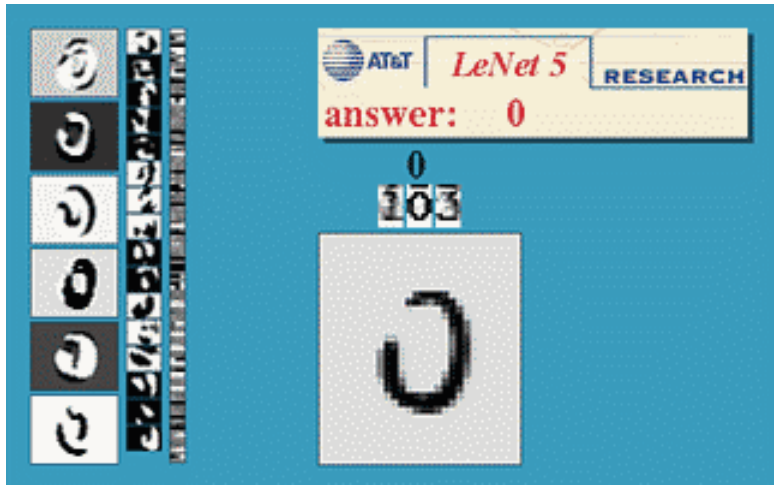
Tasks in Computer Vision

- A list of some high-level problems where we have seen success with computer vision.
  - Optical character recognition (OCR)
  - Machine inspection
  - Retail (e.g. automated checkouts)
  - 3D model building
  - Medical imaging
  - Automotive safety
  - Surveillance
  - Face recognition and biometrics
  - Autonomous Vehicles (Self-driving cars)
  - Robotics & Automation
  - Gesture & Action Recognition

Optical character recognition (OCR)

Technology to convert scanned docs to text

- If you have a scanner, it probably came with OCR software



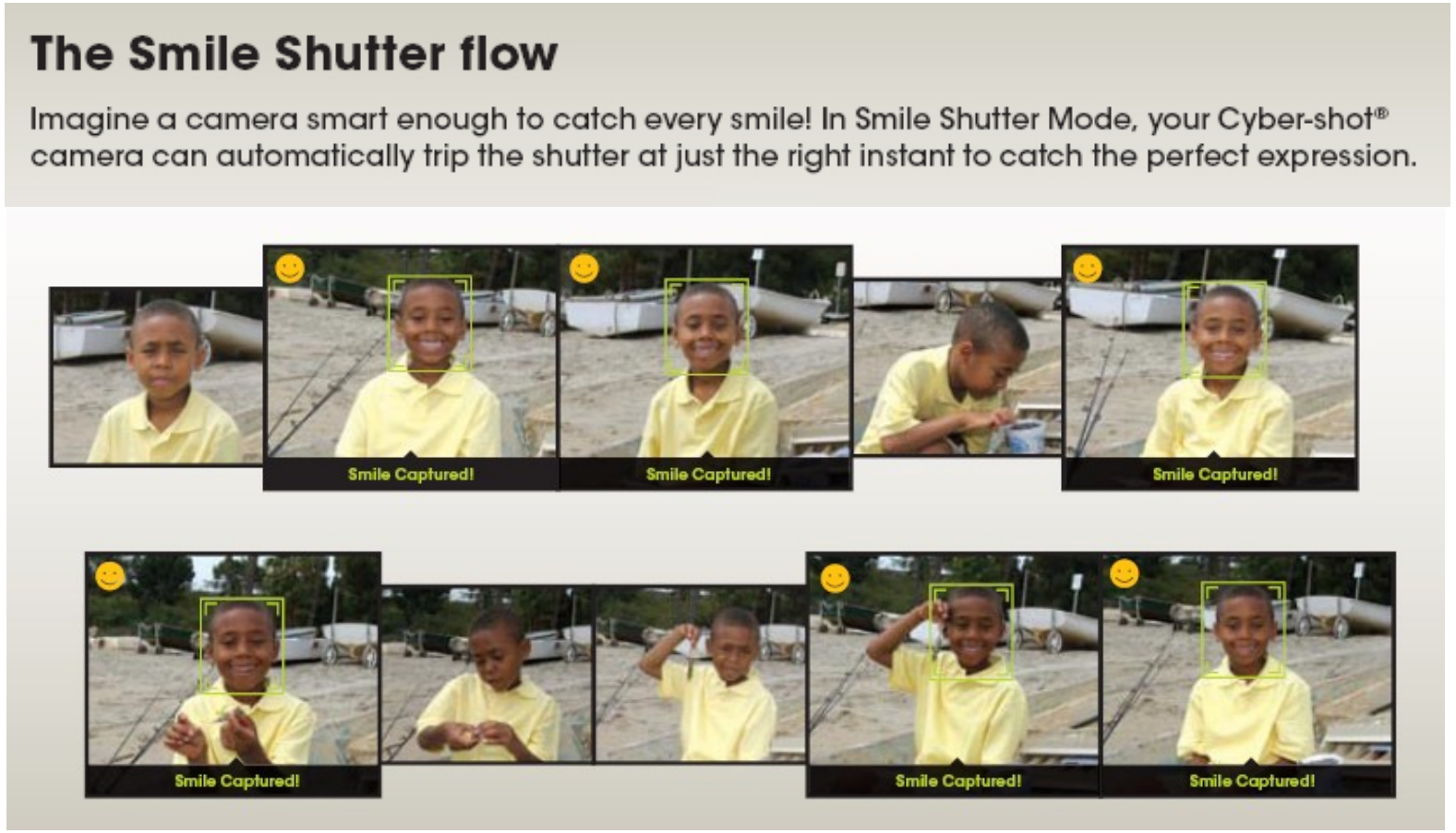
Digit recognition, AT&T labs
http://www.research.att.com/~yann/

License plate readers
http://en.wikipedia.org/wiki/Automatic_number_plate_recognition

Face detection

- Many new digital cameras now detect faces
  - Canon, Sony, Fuji, …

Smile detection



[Sony Cyber-shot® T70 Digital Still Camera]

3D from thousands of images

Building Rome in a Day: Agarwal et al. 2009

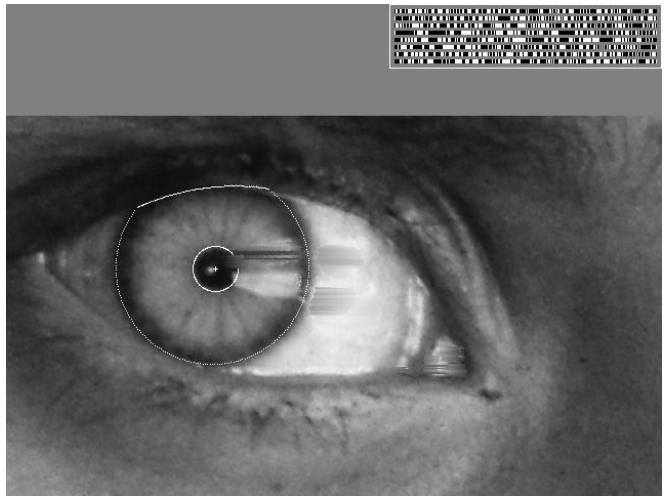
Object recognition (in supermarkets)

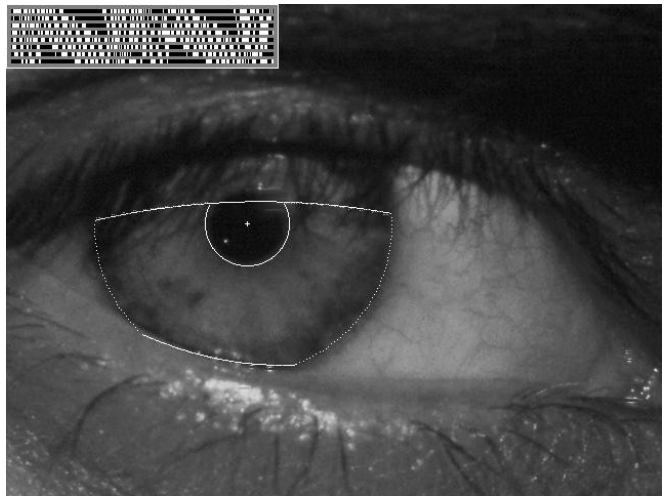
LaneHawk by EvolutionRobotics

“A smart camera is flush-mounted in the checkout lane, continuously watching for items. When an item is detected and recognized, the cashier verifies the quantity of items that were found under the basket, and continues to close the transaction. The item can remain under the basket, and with LaneHawk,you are assured to get paid for it… “

Vision-based biometrics

"*How the Afghan Girl was Identified by Her Iris Patterns*" Read the story wikipedia





Login without a password…

Fingerprint scanners on many new laptops, other devices

Face recognition systems now beginning to appear more widely
http://www.sensiblevision.com/

Object recognition (in mobile phones)

[Point & Find](), [Nokia]()

[Google Goggles]()

Special effects: shape capture

The Matrix movies, ESC Entertainment, XYZRGB, NRC

Special effects: motion capture

Pirates of the Carribean, Industrial Light and Magic

Sports

Sportvision first down line
Nice explanation on www.howstuffworks.com

http://www.sportvision.com/video.html

Smart cars

Slide content courtesy of Amnon Shashua

- Mobileye
  - Vision systems currently in high-end BMW, GM, Volvo models
  - By 2010: 70% of car manufacturers.

Google cars

http://www.nytimes.com/2010/10/10/science/10google.html?ref=artificialintelligence

Classification

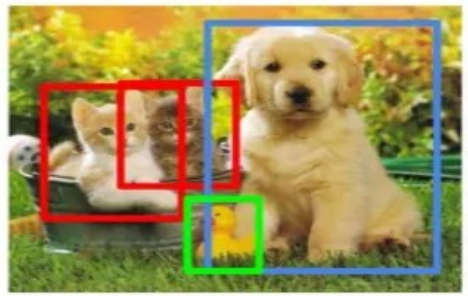
CAT

Classification + Localization

CAT

Single object

Object Detection



CAT, DOG, DUCK

Instance Segmentation

CAT, DOG, DUCK

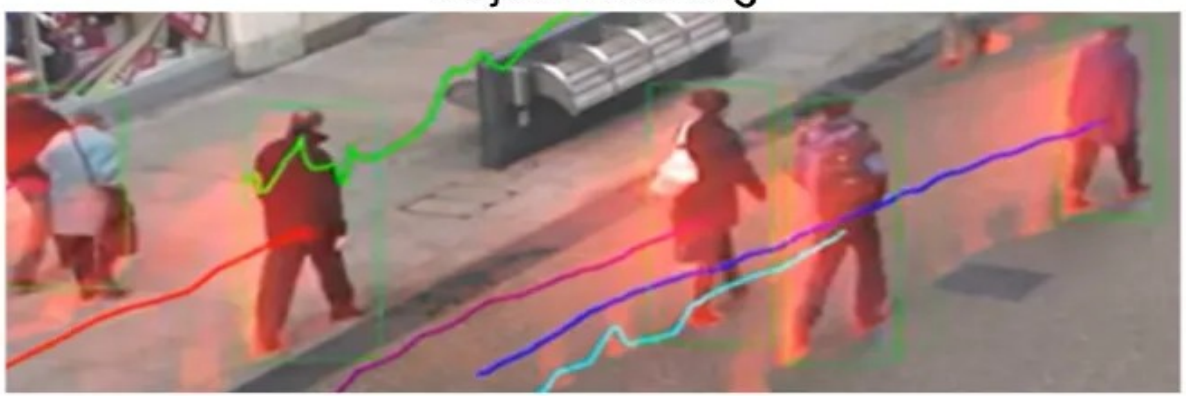
Multiple objects

Semantic Segmentation

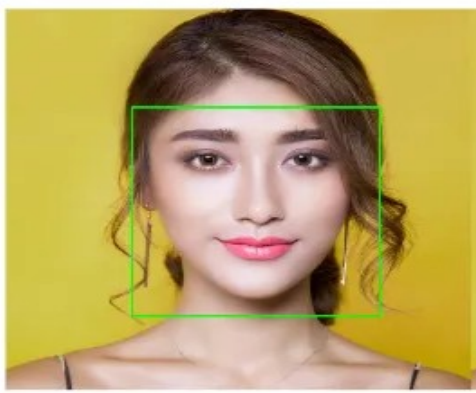
GRASS, CAT, TREE, SKY

No objects, just pixels

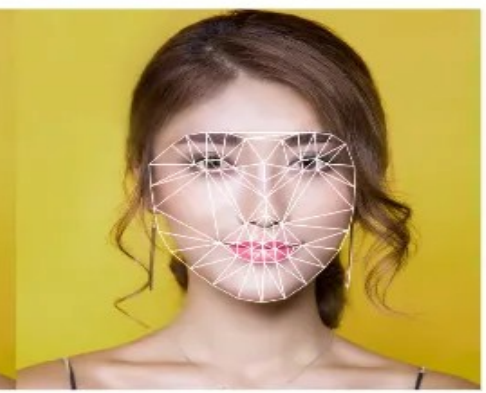
Object Tracking



Face Detection



Face Recognition

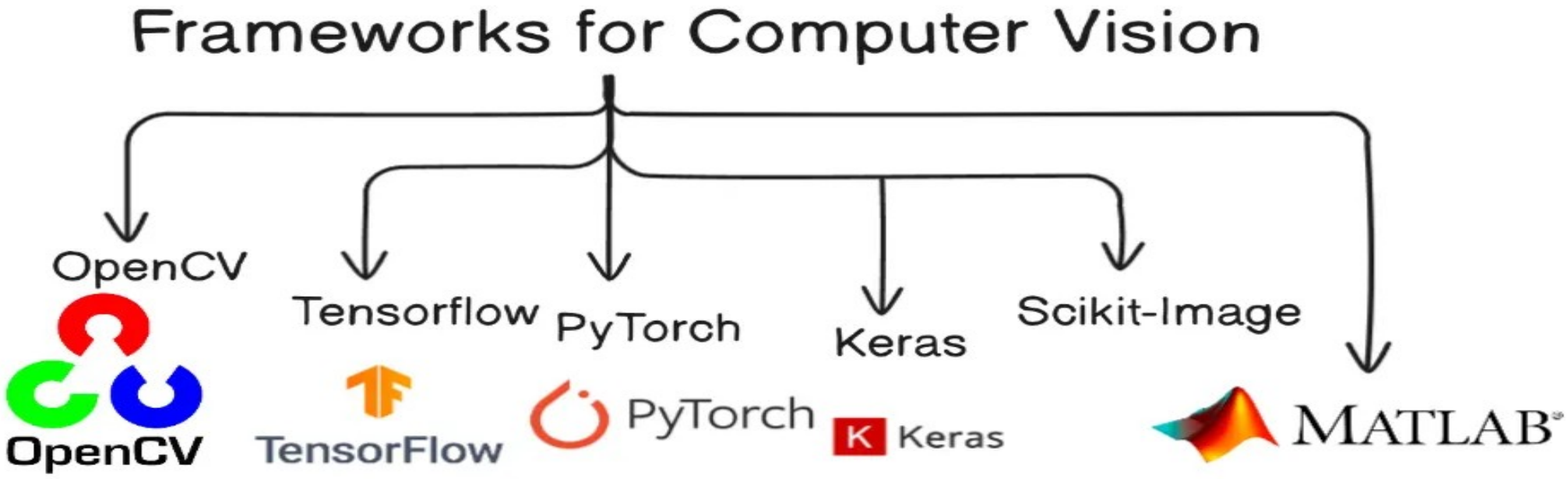


Feature Detection and Description

- SIFT (Scale-Invariant Feature Transform):
  - Detects and describes local features in images that are invariant to scale and rotation.
- SURF (Speeded-Up Robust Features):
  - Similar to SIFT but faster, it is used for object recognition and 3D reconstruction.
- HOG Feature Descriptor (Histogram Of Oriented Gradient): ---
  - Describes image features based on the distribution of intensity gradients.

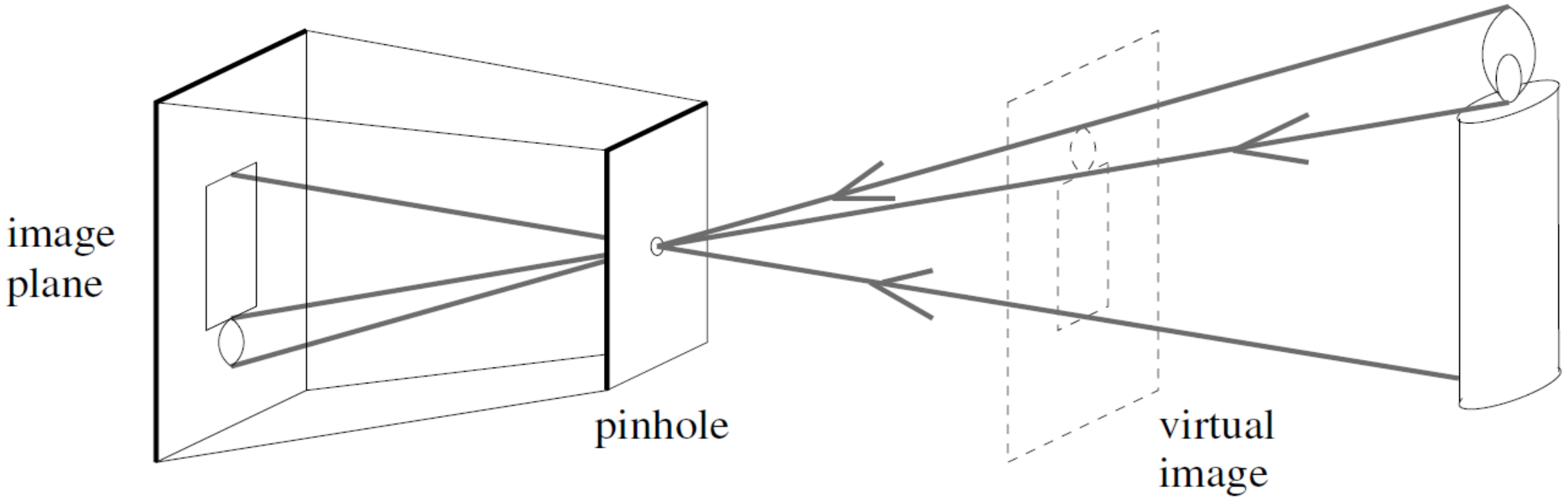
Frameworks for Computer Vision

- Computer vision development and deployment are facilitated by various tools and frameworks that provide pre-built functions, libraries, and environments.

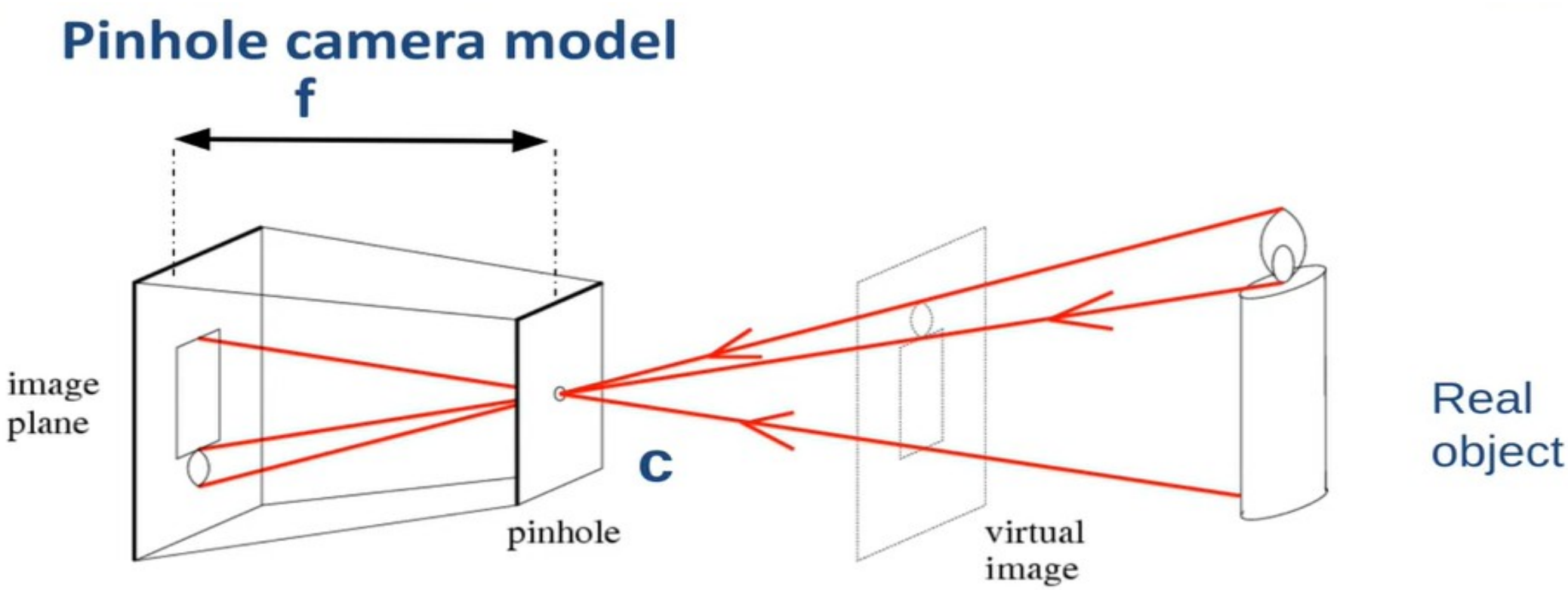


Pinhole Camera Model

Imagine taking a box, using a pin to prick a small hole in the center of one of its sides, and then replacing the opposite side with a translucent plate. If you hold that box with the pinhole facing some light source say a candle, an inverted image of the candle will appear on the translucent plate as shown in the figure.



Pinhole Camera Model



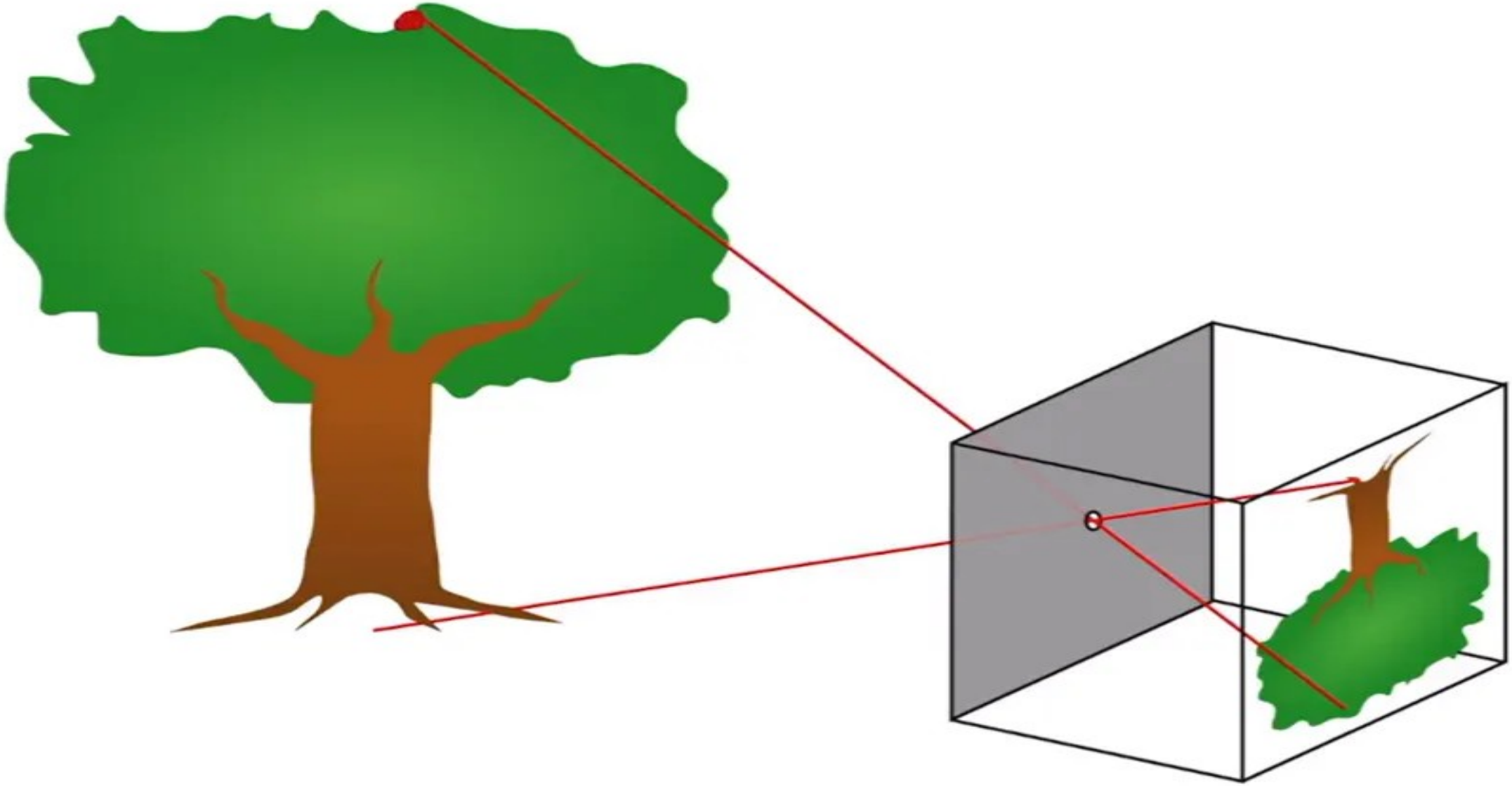
f = Focal length
C = Optical center of the camera

Pinhole imaging model

- In reality, a real pinhole is not infinitely small, light spreads slightly, causing blurring. So, the perfect pinhole camera model is only an approximation.
- The pinhole perspective model (also called perspective projection) is a basic way of understanding how cameras form images.
- The perspective model (also called perspective projection) is a mathematical model that describes how a 3D scene is projected onto a 2D image, like a photograph.

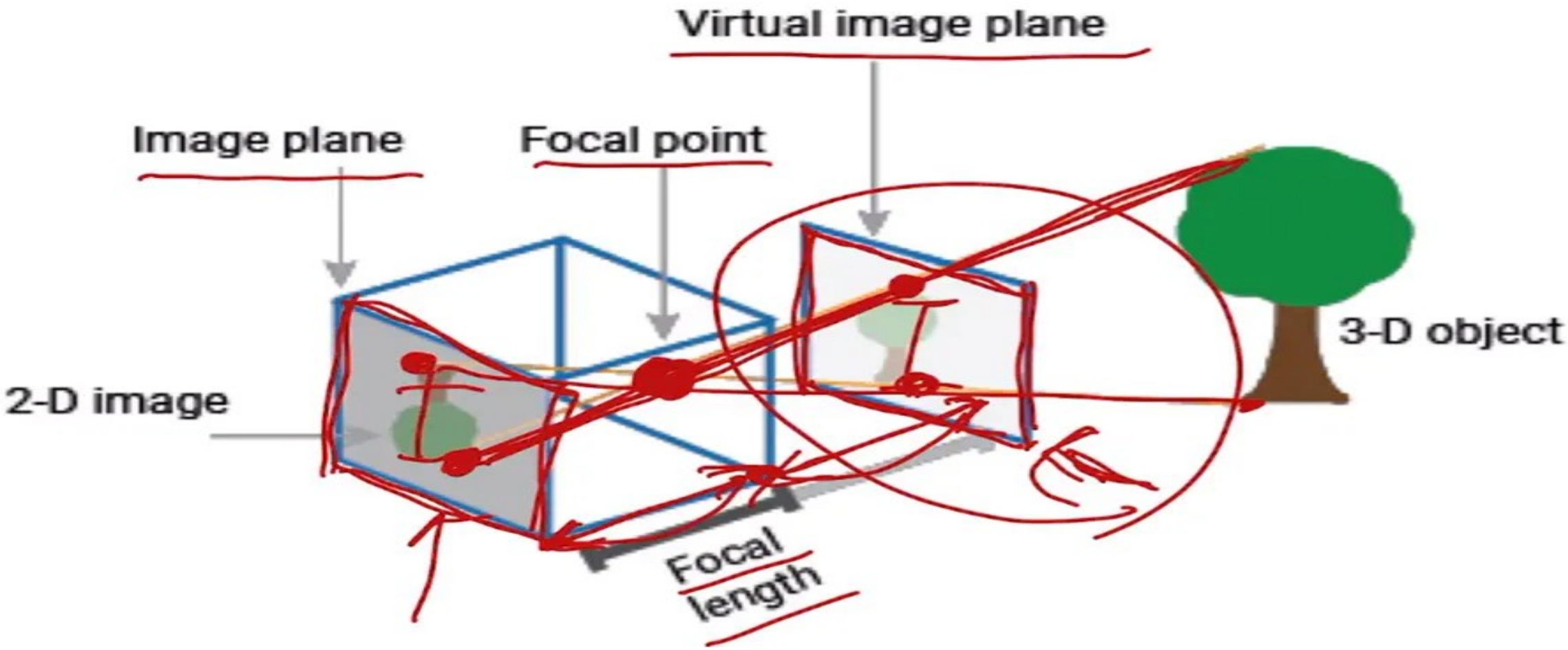
- It was first introduced by an architect named Brunelleschi in the early 1400s. Even though it is a simple model, it is very useful because the mathematics behind it is easy to work with.
- Real pinhole projection flips the image upside down (inverted image), so for easier mathematical modeling, we often use a virtual image plane in front of the pinhole where the image appears as per actual image at the same distance from it as the actual image plane, it is shown in the previous image.
- We imagine a virtual image plane in front of the pinhole so that images look upright instead of inverted, making perspective geometry simpler to work with.

Pinhole Camera Model



Virtual Image Plane and Focal length

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

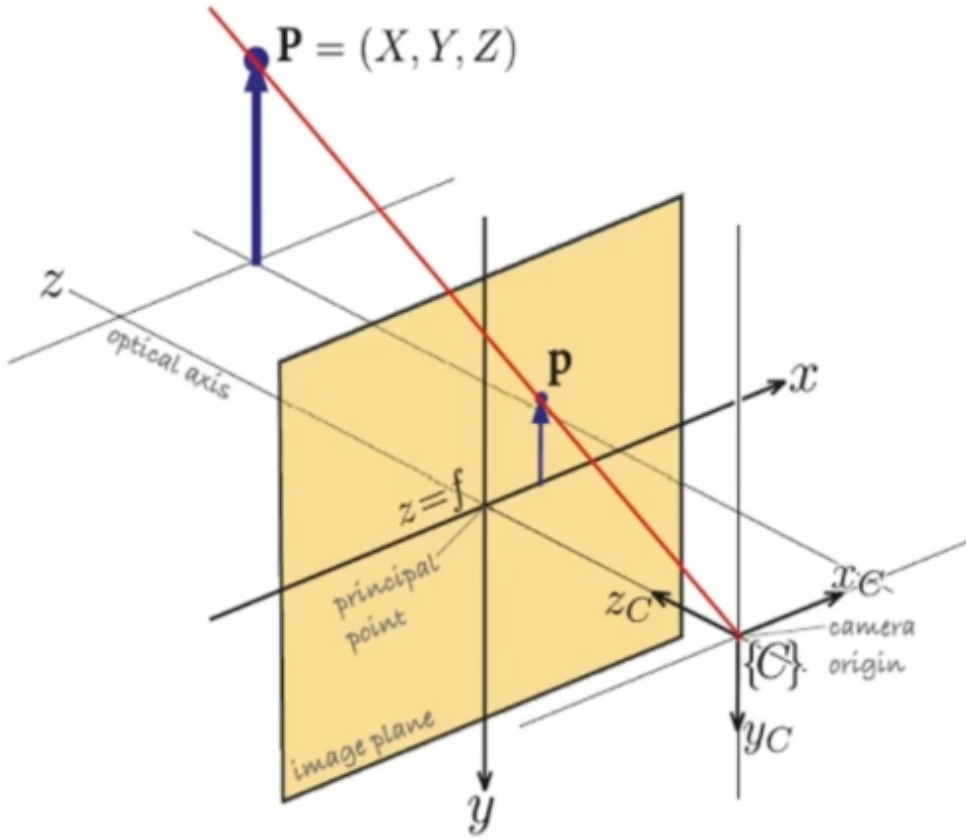


Camera 3D space to 2D Image plane

From camera coordinates (3D) to camera image plane coordinates (2D)

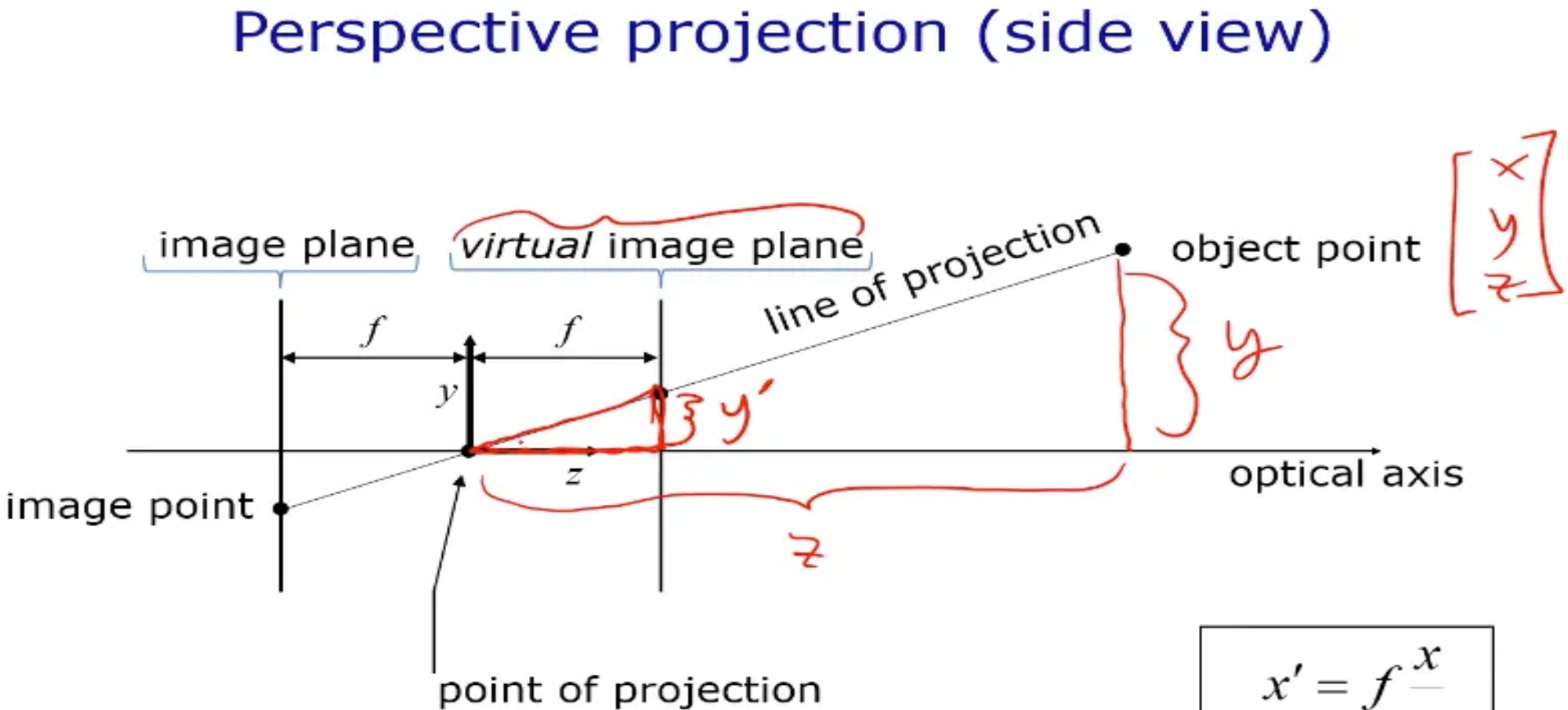
- The rays converge on the origin of the camera frame {C}.
- **A noninverted image is projected onto the image plane located at z = f.**
- The z-axis intersects the image plane at the principal point which is the origin of the 2D image coordinate frame.
- Using similar triangles we can show that
  - a point at the **camera coordinates P = (X, Y, Z)** is projected to the image point **p = (x, y)**

$$\mathbf{z = f} \quad x = f\frac{X}{Z},\ y = f\frac{Y}{Z}$$



projective transformation**, or more specifically a **perspective projection

Perspective Projection

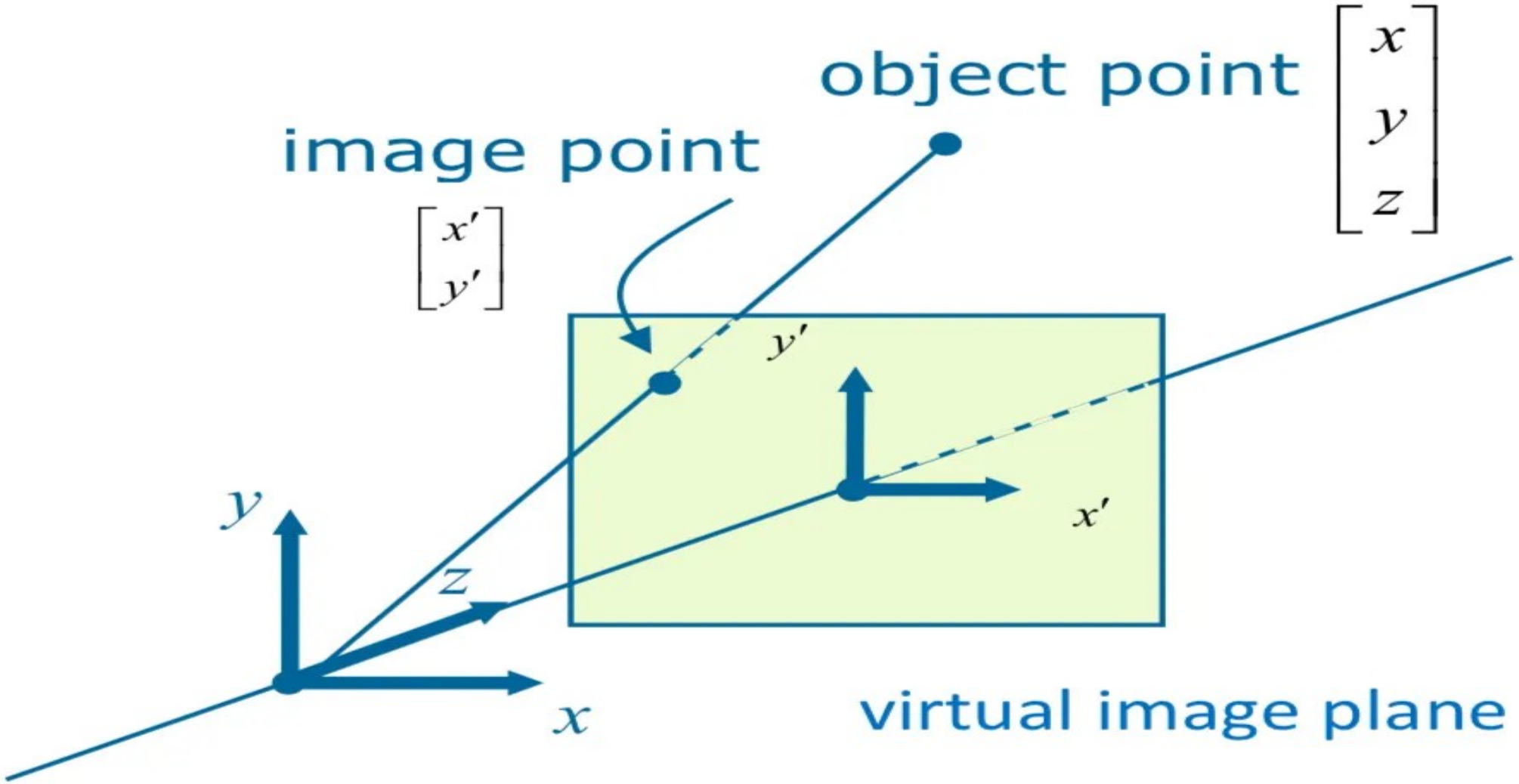


From similar triangles,
we can derive these important relations:

$$x' = f\frac{x}{z}$$

$$y' = f\frac{y}{z}$$

2D Virtual Image Plane



Perspective Projection 3D to 2D Steps

- Transformation from the world coordinate frame to the camera coordinate frame.

World Coordinates :

$$X_w = \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix}$$

Camera Coordinates :

$$\mathbf{X}_c = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}$$

- Perspective projection from the camera coordinate frame to the image plane.

Image Coordinates:

$$x_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix}$$

- Homogeneous Coordinates
- To linearize the camera model, we introduce homogeneous coordinates. Homogeneous coordinates allow us to represent a 2D u = (u, v) or 3D point using an additional fictitious coordinate. This enables us to express the perspective projection equation as a matrix multiplication, which is a linear operation.

Homogeneous coordinates

Converting to homogeneous coordinates

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

2D (image) coordinates

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

3D (scene) coordinates

Converting from homogeneous coordinates

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

2D (image) coordinates

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

3D (scene) coordinates

From camera coordinates (3D) to camera image plane coordinates (2D)

$$\mathbf{z = f} \quad x = f\frac{X}{Z},\ y = f\frac{Y}{Z}$$

- **Representing the projection using homogenous matrix**
  - We can write the image-plane point coordinates in homogeneous form p=(x~, y~, z~) where

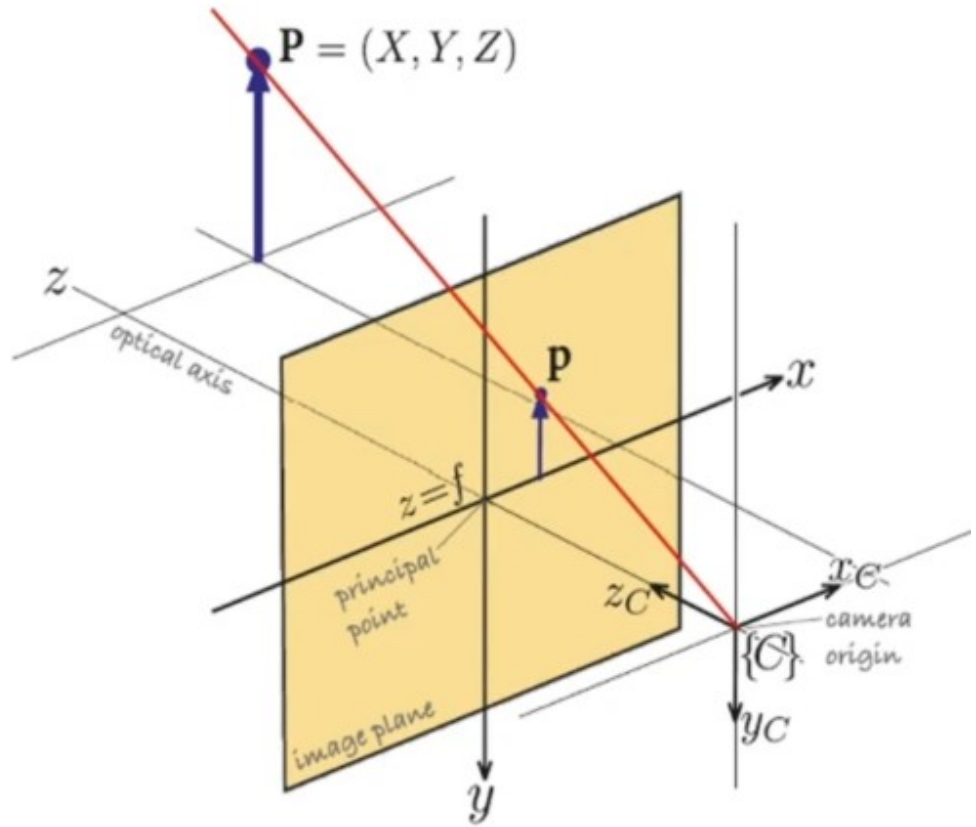
$$\tilde{x} = fX,\ \tilde{y} = fY,\ \tilde{z} = Z$$

$$\tilde{p} = \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$$

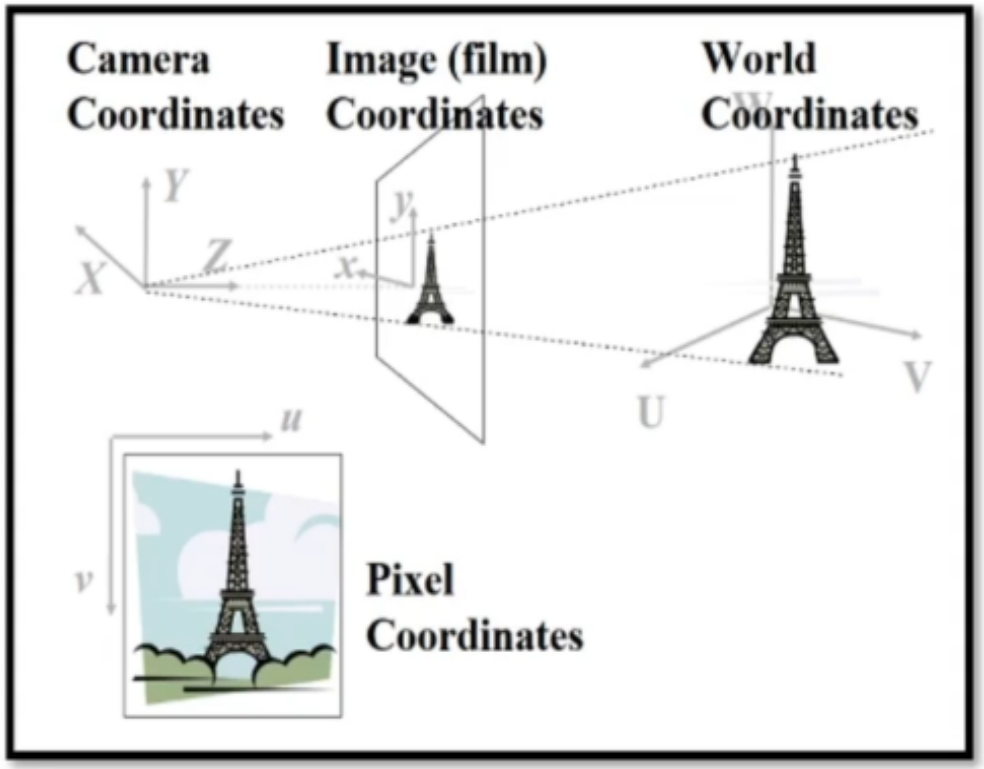
$$x = \frac{\tilde{x}}{\tilde{z}},\ y = \frac{\tilde{y}}{\tilde{z}}$$

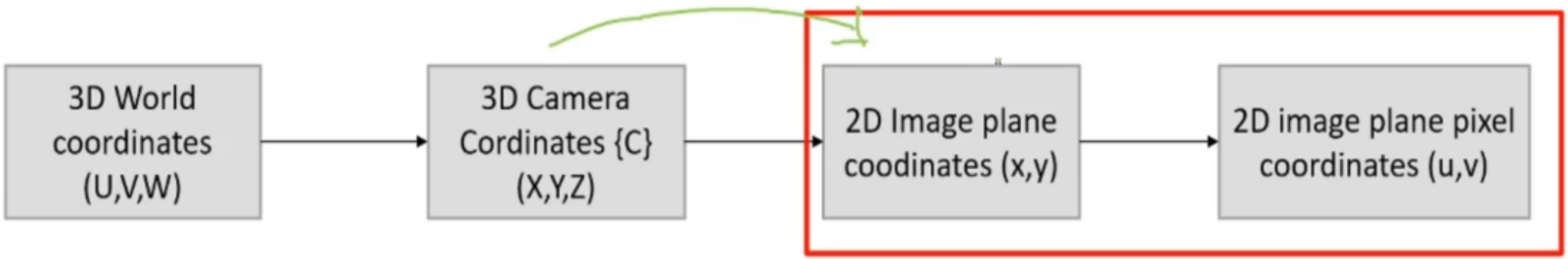
image-plane point coordinates

Camera coordinates

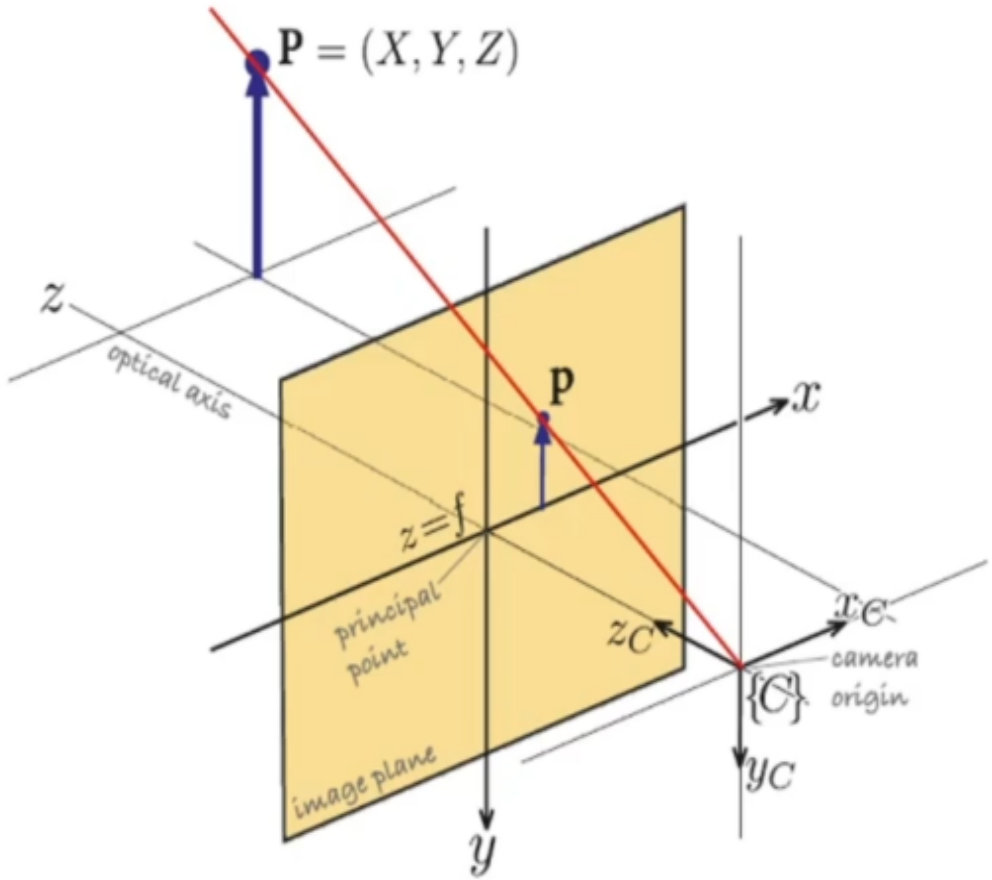


Forward Projection using camera projection model

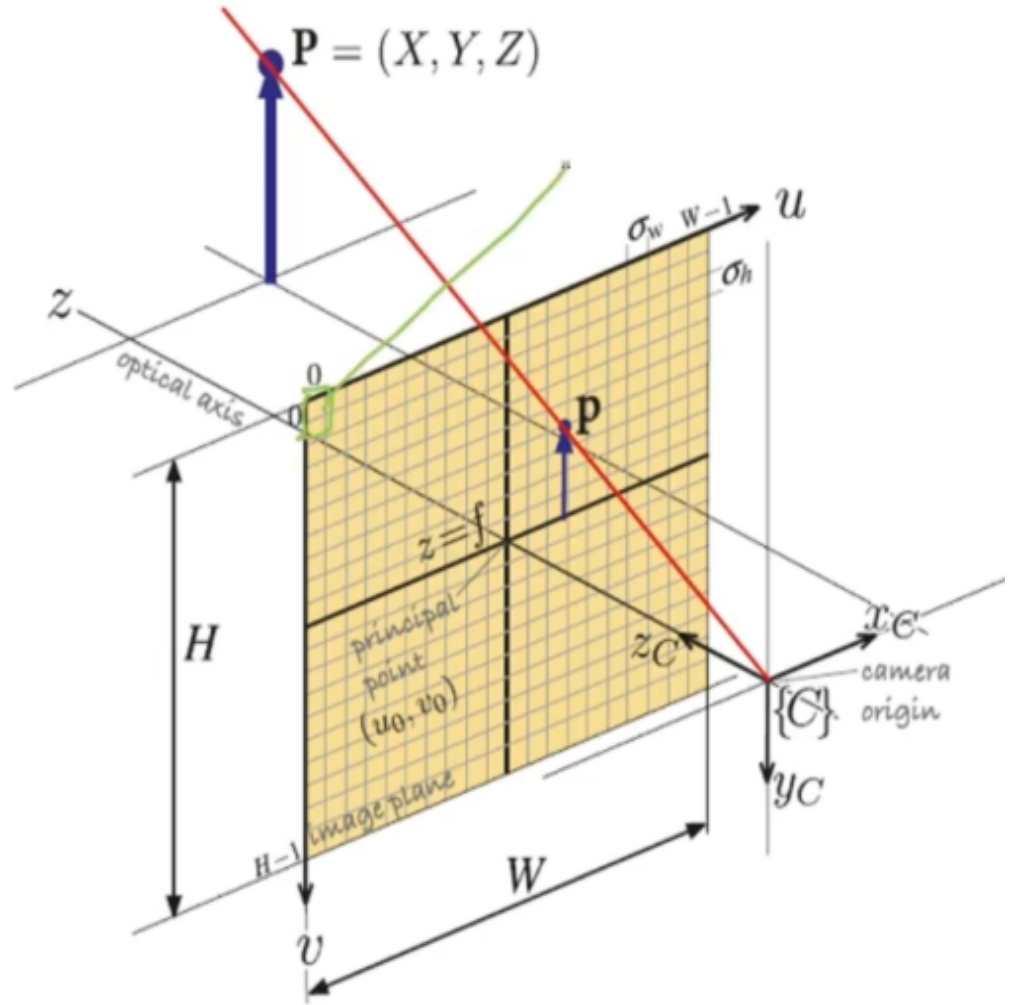




From 2D image coordinate to 2D image pixel coordinate



Normal Image Plane



Discrete Image Plane

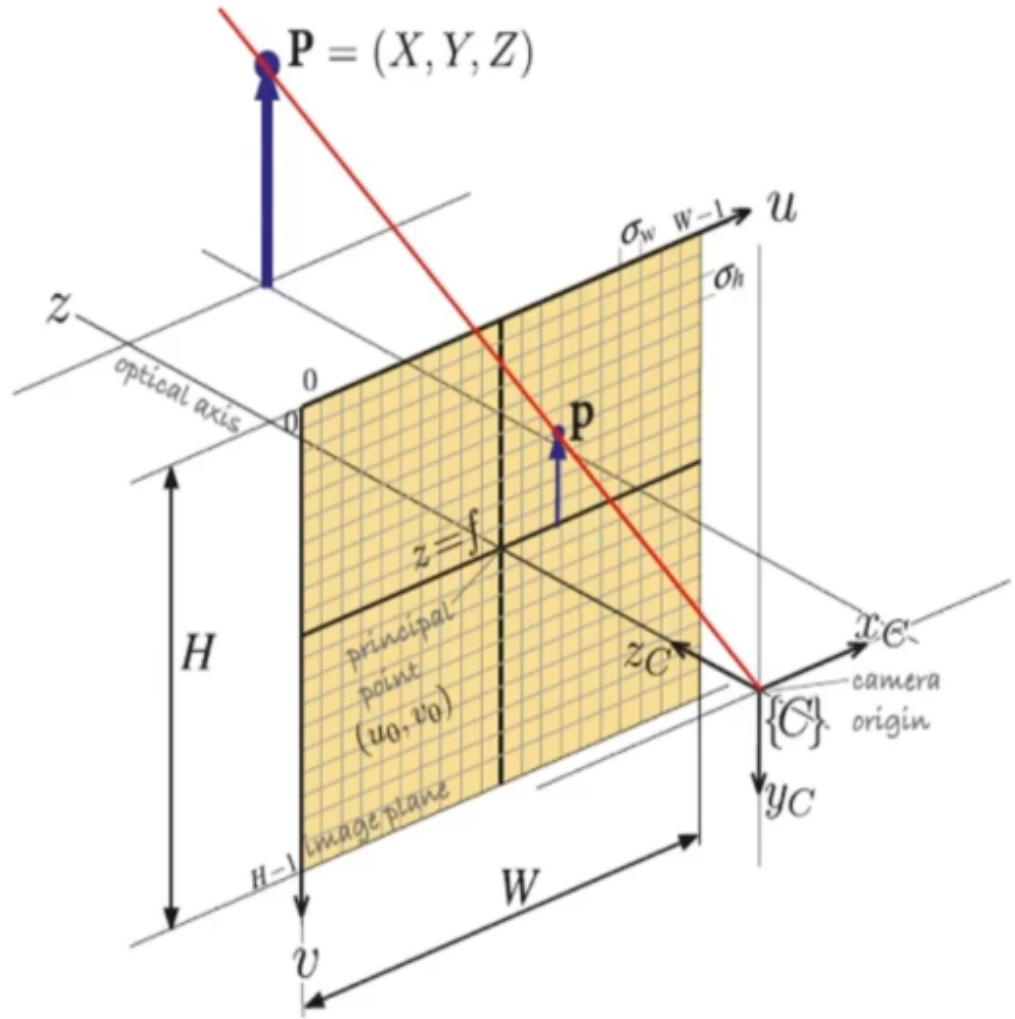
From 2D image coordinate to 2D image pixel coordinate

- The pixel coordinates are a 2-vector (u, v) of nonnegative integers and by convention the origin is at the top-left hand corner of the image plane.

- The pixels are uniform in size and centered on a regular grid so the pixel coordinate is related to the image-plane coordinate by

$$u = \frac{x}{\rho_w} + u_0, \; v = \frac{y}{\rho_h} + v_0$$

where **ρ_w and ρ_h are the width and height of each pixel respectively,** and **(u_0, v_0) is the principal point – the pixel coordinate of the point where the optical axis intersects the image plane with respect to the new origin**



Discrete Image Plane

Conversion of 2D physical image coordinates to digital pixel coordinates

- A point (x,y)(x,y) on the image plane (measured in mm or physical units) is converted to pixel coordinates (u,v)(u,v) using:

$$u = \frac{x}{\rho_w} + u_0 \qquad u = \frac{x}{\text{pixel width}} + \text{center x offset}$$

$$v = \frac{y}{\rho_h} + v_0 \qquad v = \frac{y}{\text{pixel height}} + \text{center y offset}$$

Symbol	Meaning
(x, y)	Coordinates on image plane (real metric values, mm)
(u, v)	Pixel coordinates in digital image (pixels)
ρ_w	Width of a pixel (mm/pixel)
ρ_h	Height of a pixel (mm/pixel)
(u_0, v_0)	Principal point (center of image where optical axis hits) in pixel units

Example

Suppose:

- Image plane point: $x = 2.4\ \text{mm}, y = -1.2\ \text{mm}$
- Pixel size: $\rho_w = 0.006\ \text{mm}, \rho_h = 0.006\ \text{mm}$
- Principal point: $u_0 = 320, v_0 = 240$

$$u = \frac{2.4}{0.006} + 320 = 720 + 320 = 1040$$

$$v = \frac{-1.2}{0.006} + 240 = -200 + 240 = 40$$

📌 **Pixel coordinate = (1040, 40)**

Homogeneous Coordinates

The homogenous representation of a 3D point $\mathbf{x} = (x, y, z) \in \mathcal{R}^3$ is a 4D point $\tilde{\mathbf{x}} = (\tilde{x}, \tilde{y}, \tilde{z}, \tilde{w}) \in \mathcal{R}^4$. The fourth coordinate $\tilde{w} \neq 0$ is fictitious such that:

$$x = \frac{\tilde{x}}{\tilde{w}} \qquad y = \frac{\tilde{y}}{\tilde{w}} \qquad z = \frac{\tilde{z}}{\tilde{w}}$$

$$\mathbf{x} \equiv \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \equiv \begin{bmatrix} \tilde{w}x \\ \tilde{w}y \\ \tilde{w}z \\ \tilde{w} \end{bmatrix} \equiv \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \\ \tilde{w} \end{bmatrix} = \tilde{\mathbf{x}}$$

The homogenous representation of a 2D point $\mathbf{u} = (u, v)$ is a 3D point $\tilde{\mathbf{u}} = (\tilde{u}, \tilde{v}, \tilde{w})$. The third coordinate $\tilde{w} \neq 0$ is fictitious such that:

$$u = \frac{\tilde{u}}{\tilde{w}} \qquad v = \frac{\tilde{v}}{\tilde{w}}$$

$$\mathbf{u} \equiv \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \equiv \begin{bmatrix} \tilde{w}u \\ \tilde{w}v \\ \tilde{w} \end{bmatrix} \equiv \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \tilde{\mathbf{u}}$$

From 2D image coordinate to 2D image pixel coordinate

$$u = \frac{x}{\rho_w} + u_0, \; v = \frac{y}{\rho_h} + v_0$$

- **Representing using homogenous matrix**
  - We can write the image-plane pixel coordinates in homogeneous form p=(u', v', w') where u = u'/w' and v = v'/w' and w' = $\tilde{z}$

$$\begin{bmatrix} u' \\ v' \\ w' \end{bmatrix} = \begin{pmatrix} 1/\rho_w & 0 & u_0 \\ 0 & 1/\rho_h & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \end{bmatrix}$$

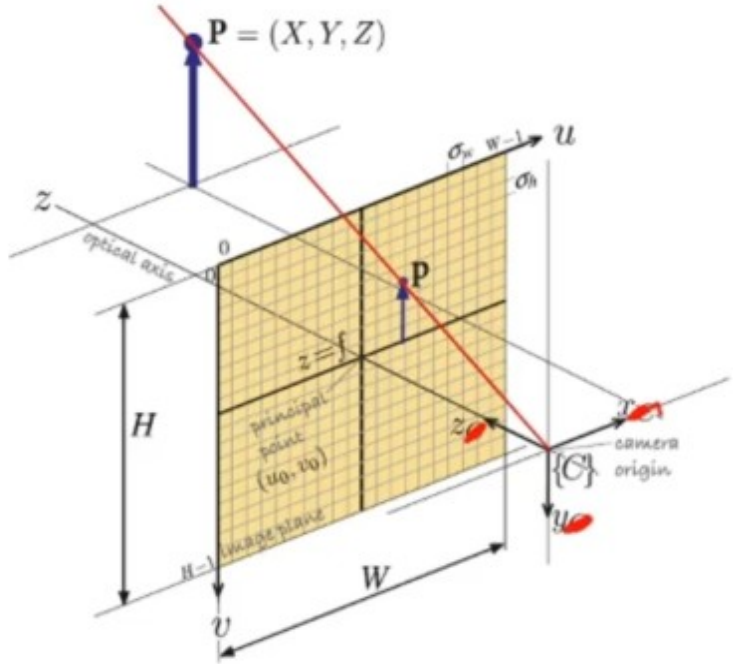
(1) $u' = \frac{1}{\rho_w}\tilde{x} + u_0\tilde{z}$

(2) $v' = \frac{1}{\rho_h}\tilde{y} + v_0\tilde{z}$

(3) $w' = \tilde{z}$

$$\therefore u' = \frac{x \cdot \tilde{z}}{\rho_w} + u_0\tilde{z}$$

$$u = \frac{u'}{w'} = \frac{u'}{\tilde{z}} = \frac{x}{\rho_w} + u_0$$

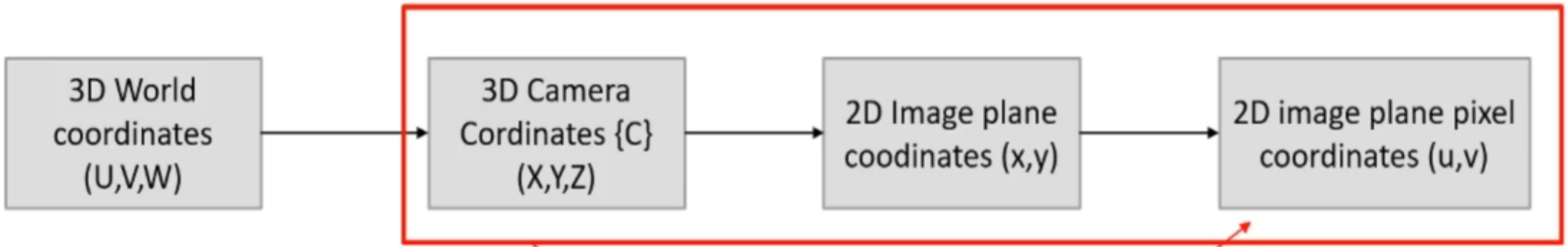


Discrete Image Plane

$$x = \frac{\tilde{x}}{\tilde{z}}, \; y = \frac{\tilde{y}}{\tilde{z}}$$

$$\tilde{\boldsymbol{p}} = \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$$

Forward Projection using camera projection model



$$f/p_w = f_x$$

$$f/p_h = f_y$$

$$\begin{bmatrix} f/p_w & 0 & u_0 \\ 0 & f/p_h & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

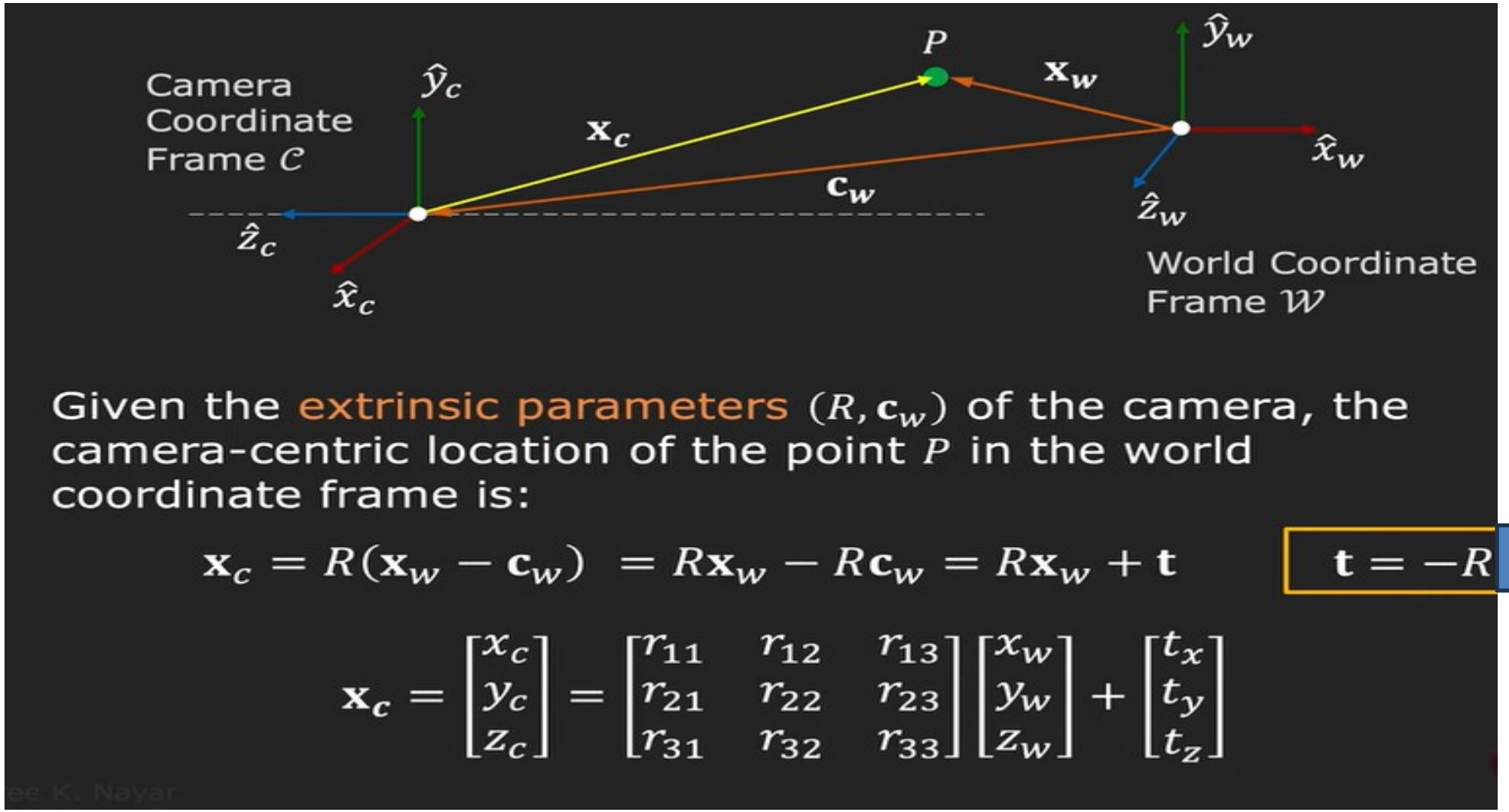
$u_0 = c_x$

$v_0 = c_y$

Intrinsic matrix of camera

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

World to Camera Coordinates Transformation



World coordinate to camera coordinate (3D to 3D)

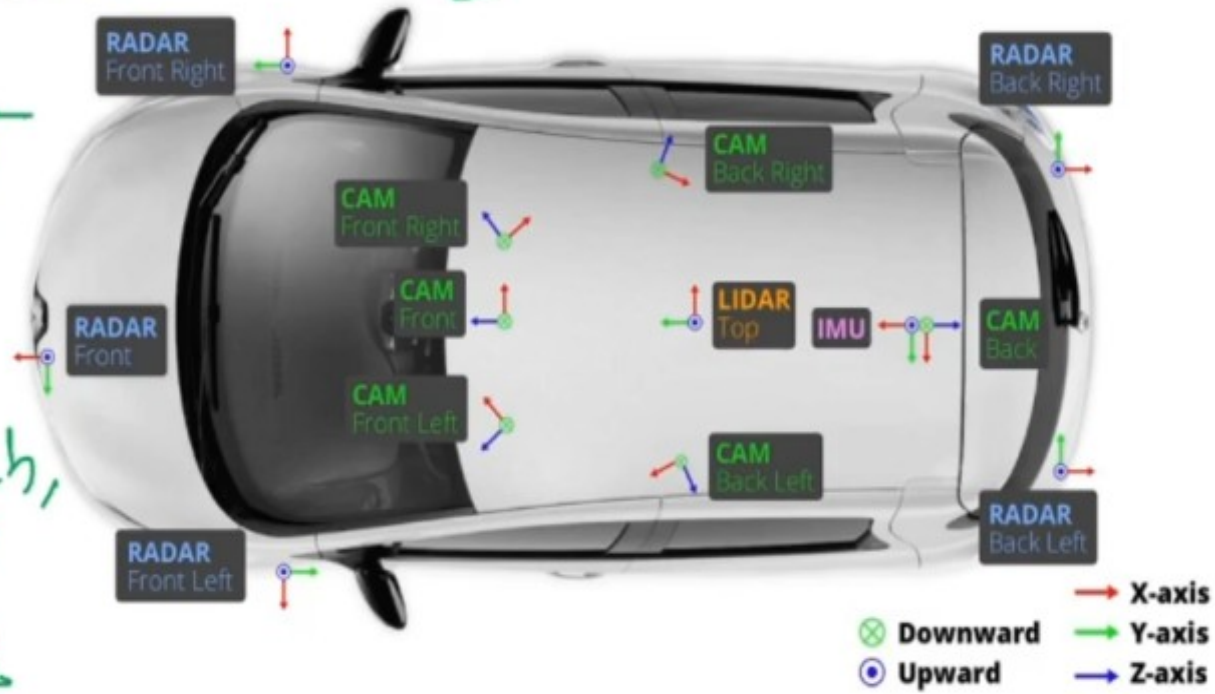
$$[R|t] = \left[\begin{array}{ccc|c} r_{1,1} & r_{1,2} & r_{1,3} & t_1 \\ r_{2,1} & r_{2,2} & r_{2,3} & t_2 \\ r_{3,1} & r_{3,2} & r_{3,3} & t_3 \end{array}\right] = T$$

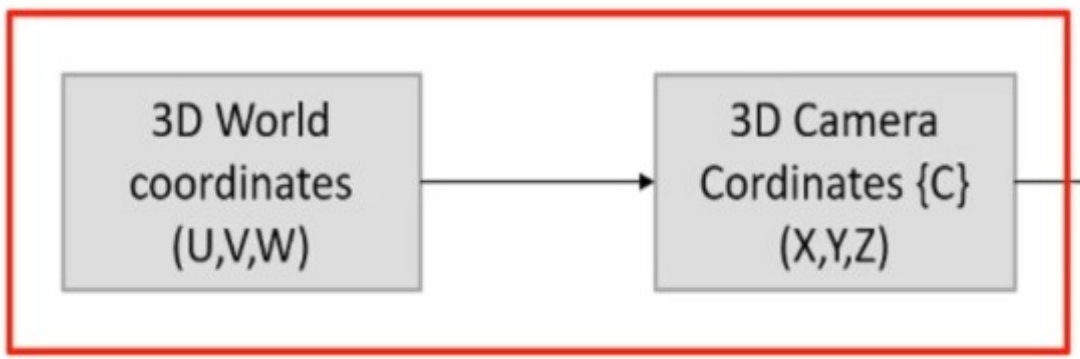
t_x
t_y
t_z

4x3

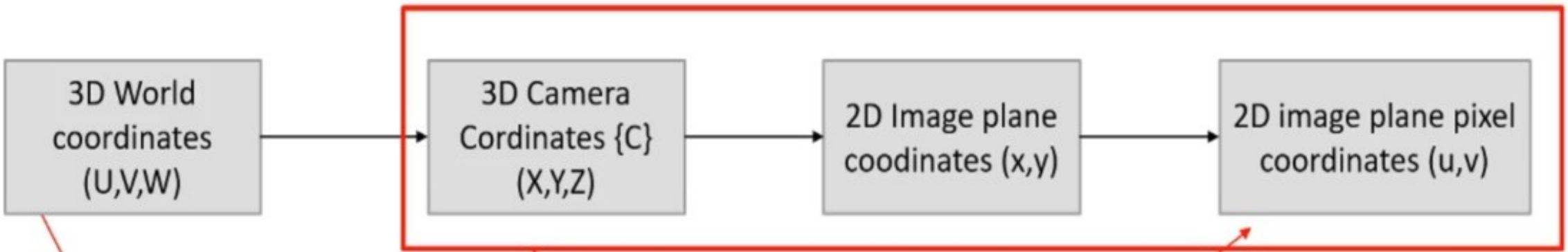
R

1. 3x3 = 9
2. Euler angles, α, β, γ = Roll, pitch, yaw
3. Quaternion → q_w, q_x, q_y, q_z





Forward Projection using camera projection model



$$[R|t] = \left[\begin{array}{ccc|c} r_{1,1} & r_{1,2} & r_{1,3} & t_1 \\ r_{2,1} & r_{2,2} & r_{2,3} & t_2 \\ r_{3,1} & r_{3,2} & r_{3,3} & t_3 \\ 0 & 0 & 0 & 1 \end{array}\right]$$

4×4

Extrinsic matrix of camera

$$\begin{bmatrix} f/p_w & 0 & u_0 \\ 0 & f/p_h & v_0 \\ 0 & 0 & 1 \end{bmatrix} \rightarrow \begin{bmatrix} fx & 0 & cx \\ 0 & fy & cy \\ 0 & 0 & 1 \end{bmatrix}$$

Instrinsic matrix of camera

Degree of freedom

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

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

Summary

- We convert a 3-D point in the world into a 2-D pixel (image) coordinate in three conceptual stages:

- 1. World → Camera (3D → 3D)
  - This places the 3D point in the camera's coordinate frame. The camera may be translated and rotated relative to the world.

  - This is a rigid transform described by a rotation matrix R and a translation vector t. Result: point coordinates (Xc, Yc, Zc) expressed relative to the camera.

- 2. Camera → Image plane (3D → 2D, perspective projection)
  - Using the pinhole (perspective) model, the 3D camera point projects onto the image plane at distance f (focal length) from the pinhole. This gives metric image-plane coordinates (x,y) (usually in mm). The projection is perspective: coordinates scale by $1/Zc$.

- 3. Image plane → Pixel (2D Space → 2D discrete pixels)
  - Sensors store images as pixels. Convert metric coordinates x, y into pixel coordinates u, v by scaling with pixel size and shifting by the principal point ($u0$, $v0$). Intrinsic camera parameters K collect focal lengths in pixels, skew, and principal point.

- Note: We use **homogeneous coordinates** to keep all these steps as linear matrix multiplications (so translation and perspective can be expressed compactly).

Notation summary

- World point (homogeneous): $P_w = [X_w,\, Y_w,\, Z_w,\, 1]^\top$.
- Camera (non-homog): $P_c = [X_c,\, Y_c,\, Z_c]^\top$.
- Image plane (metric): (x, y) with focal length f.
- Pixel coordinates: $p = [u,\, v]^\top$.
- Rotation: $R \in \mathbb{R}^{3\times 3}$.
- Translation: $t \in \mathbb{R}^3$.
- Intrinsic matrix:

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

where f_x, f_y = focal length in pixels, s = skew, $(c_x, c_y) = (u_0, v_0)$ principal point.

Step A — World → Camera (rigid transform)

Apply rotation and translation to move the world point into camera coordinates:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = R \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} + t.$$

In homogeneous form:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} R & t \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}.$$

Call the 3×4 matrix $[R \mid t]$.

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

Step B — Camera → Image plane (pinhole perspective)

Using the pinhole model, the metric image-plane coordinates (assuming image plane at distance f):

$$x = f\frac{X_c}{Z_c}, \qquad y = f\frac{Y_c}{Z_c}.$$

This is nonlinear due to division by Z_c. To linearize we switch to homogeneous image coordinates:

Let the homogeneous image vector be $[\tilde{u}, \tilde{v}, \tilde{w}]^\top = K[X_c, Y_c, Z_c]^\top$ (we'll connect K next). The actual pixel coordinates are obtained after dividing by $\tilde{w}$.

Step C — Image plane (metric) → Pixel coordinates (u,v)

Convert metric x, y to pixels:

$$u = \frac{x}{\rho_w} + u_0, \qquad v = \frac{y}{\rho_h} + v_0,$$

where ρ_w, ρ_h are pixel widths/heights (mm/pixel). Equivalently, combine the scale into $f_x = f/\rho_w$ and $f_y = f/\rho_h$ to get:

$$\begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \underbrace{\begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}}_{K} \begin{bmatrix} X_c/Z_c \\ Y_c/Z_c \\ 1 \end{bmatrix}.$$

Removing the explicit division by Z_c by working in homogeneous coordinates:

$$\begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = K \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}.$$

Finally normalize to get pixel coordinates:

$$u = \frac{\tilde{u}}{\tilde{w}}, \qquad v = \frac{\tilde{v}}{\tilde{w}}.$$

Combine steps A–C into one matrix equation

Substitute $[X_c\ Y_c\ Z_c]^\top = [R \mid t]P_w$ into the K multiplication:

$$\begin{bmatrix}\tilde{u}\\ \tilde{v}\\ \tilde{w}\end{bmatrix} = K\ [R \mid t]\ \begin{bmatrix}X_w\\ Y_w\\ Z_w\\ 1\end{bmatrix}. \qquad P_w = \begin{bmatrix}X_w\\ Y_w\\ Z_w\\ 1\end{bmatrix}$$

Define the full **projection matrix** P:

$$P = K\ [R \mid t] \quad (\text{a } 3 \times 4 \text{ matrix}).$$

Thus in homogeneous form:

$$\tilde{p} = P\,P_w, \qquad \text{where } \tilde{p} = [\tilde{u}, \tilde{v}, \tilde{w}]^\top.$$

Then recover pixel coordinates by normalization:

$$u = \frac{\tilde{u}}{\tilde{w}}, \qquad v = \frac{\tilde{v}}{\tilde{w}}.$$

This is the full pipeline: **world** → **camera** → **image** → **pixel** in one linear matrix and one division.

- Numerical
  - World point $P_w = [4, 3, 10, 1]^\top$.
  - Choose $R = I, t = [1, 2, 5]^\top$ so camera coords: $[X_c, Y_c, Z_c]^\top = [5, 5, 15]^\top$.
  - Let focal length $f = 20$ mm, pixel size $\rho = 0.005$ mm/pixel so $f_x = f_y = f/\rho = 4000$ px.
  - Principal point $(c_x, c_y) = (640, 480)$.
  - Then

$$K = \begin{bmatrix} 4000 & 0 & 640 \\ 0 & 4000 & 480 \\ 0 & 0 & 1 \end{bmatrix}, \quad [R|t] = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 5 \end{bmatrix}.$$

  - Compute homogeneous image: $\tilde{p} = K[R|t]P_w$.
    First $[R|t]P_w = [5, 5, 15]^\top$. Then

$$\tilde{p} = \begin{bmatrix} 4000 \cdot 5 + 0 \cdot 5 + 640 \cdot 15 \\ 0 \cdot 5 + 4000 \cdot 5 + 480 \cdot 15 \\ 0 \cdot 5 + 0 \cdot 5 + 1 \cdot 15 \end{bmatrix} = \begin{bmatrix} 20{,}000 + 9{,}600 \\ 20{,}000 + 7{,}200 \\ 15 \end{bmatrix} = \begin{bmatrix} 29{,}600 \\ 27{,}200 \\ 15 \end{bmatrix}.$$

  - Normalize:

$$u = 29{,}600/15 \approx 1973.33, \qquad v = 27{,}200/15 \approx 1813.33.$$

That yields the same pixel result as computing step-by-step: camera coords → metric projection → pixel scaling.

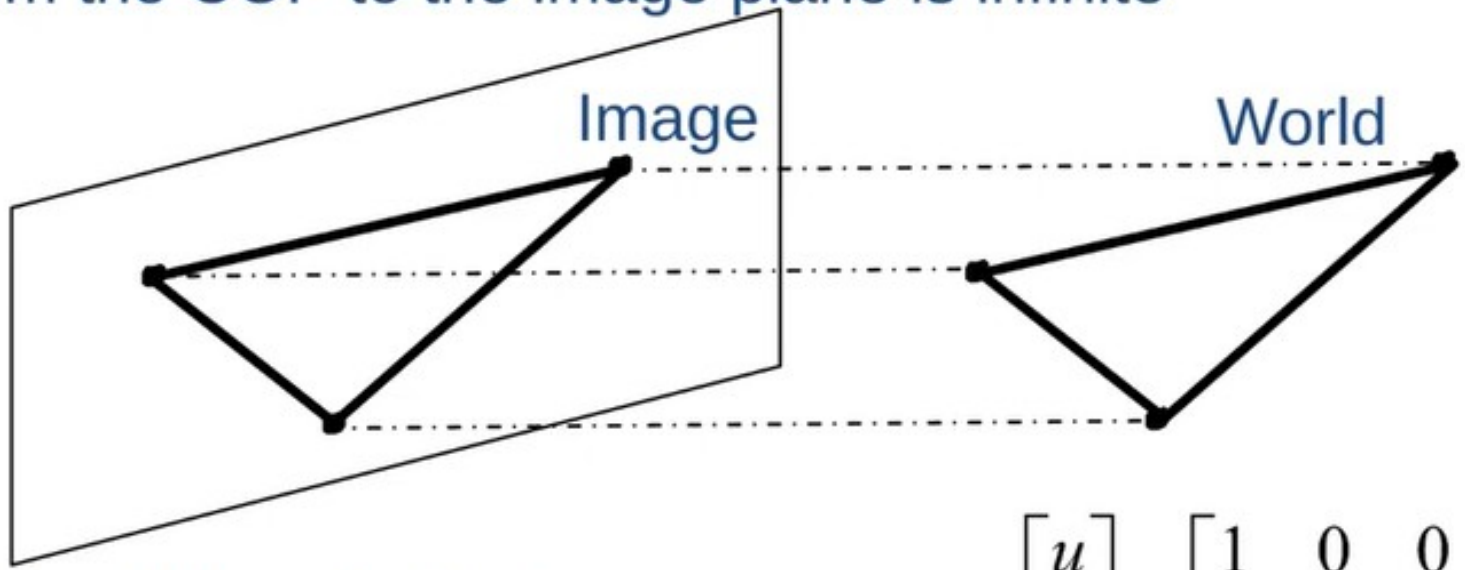
Orthographic Projection

- Orthographic projection is a way of projecting 3D points onto a 2D plane by simply dropping the depth (Z coordinate). All projection lines are parallel to each other and perpendicular to the image plane.
- Imagine photographing a large building from very far away using a zoom lens. All parts of the building appear almost the same size, even if some parts are farther.
- Key Characteristics
  - No depth perception (Z-value is ignored)
  - Objects do not shrink as they move farther away
  - Parallel lines remain parallel

Orthographic Projection

Special case of perspective projection

- Distance from the COP to the image plane is infinite



- Also called “parallel projection”
- What’s the projection matrix?

$$w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

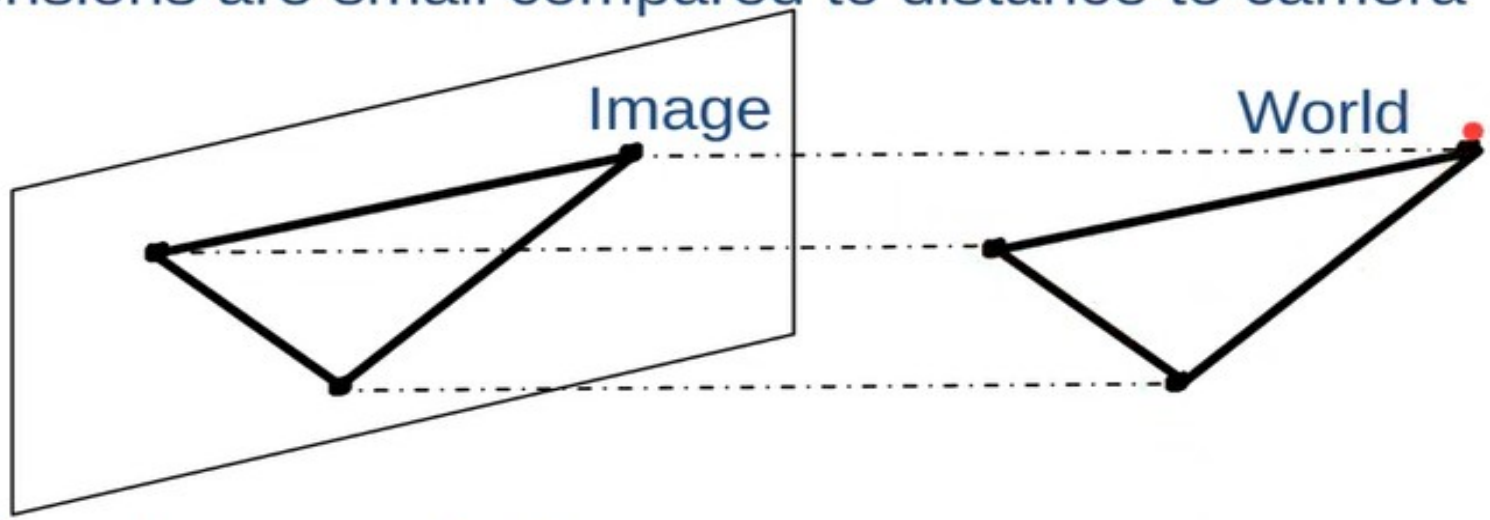
Weak Perspective Projection

- Weak perspective projection is an approximation between full perspective projection and orthographic projection.
- It assumes that the depth variation within an object is small compared to its average distance from the camera.
- Key Characteristics
  - Perspective scaling effect exists, but uniform across the object.
  - Depth differences inside the object are ignored.
  - Useful for object-level modeling, human face tracking, motion capture, 3D reconstruction when object is far away.

Scaled Orthographic Projection

Special case of perspective projection

- Object dimensions are small compared to distance to camera



- Also called “weak perspective”
- What’s the projection matrix?

$$w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 0 & s \end{bmatrix}\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Why weak perspective fails when depth varies a lot?

Weak perspective assumes:

$$Z \approx Z_{avg}$$

meaning **all points are at roughly the same depth**.

So projection becomes:

$$u = \frac{f}{Z_{avg}} X, \quad v = \frac{f}{Z_{avg}} Y$$

But if depths vary heavily (200 vs 1000), then scaling for each point should really be different:

$$u = \frac{fX}{Z_i}$$

Weak perspective **cannot scale each point differently**, so:

- Far points may appear too large or too small
- The object becomes distorted (incorrect shape)

Difference Between Perspective Projection and Weak Perspective Projection

Aspect	Perspective Projection	Weak Perspective Projection
Definition	A realistic projection where objects farther from the camera appear smaller.	A simplified approximation of perspective assuming depth variation is small across the object.
Dependency on depth	Each point is scaled by its **individual depth** Z.	All points are scaled by **average depth** Z_{avg}.
Formula	$u = \frac{fX}{Z}, \quad v = \frac{fY}{Z}$	$u = \frac{f}{Z_{avg}}X, \quad v = \frac{f}{Z_{avg}}Y$
Appearance	Converging lines, perspective distortion visible.	Nearly orthographic, minimal distortion.
When used?	Real camera imaging, VR/AR rendering, photogrammetry.	Face tracking, hand tracking, motion capture, objects far from camera.
Computational complexity	Higher (division by Z for each point).	Lower (one scaling value for all points).
Accuracy	High realism.	Approximate; good only when depth variation is small.
Parallel lines	Converge at vanishing point.	Remain almost parallel.