Content

- Segmenting Multiple Blobs
- 3D Pose Estimation for Known Objects
  - Introduction
  - Camera Intrinsic Parameters
  - Camera Extrinsic Parameters
  - Camera Calibration
  - 3D Pose Estimation
- Depth Perception for Arbitrary Objects
  - Introduction
  - Stereo Disparity
  - Correspondence Problem
  - Non-coplanar Cameras

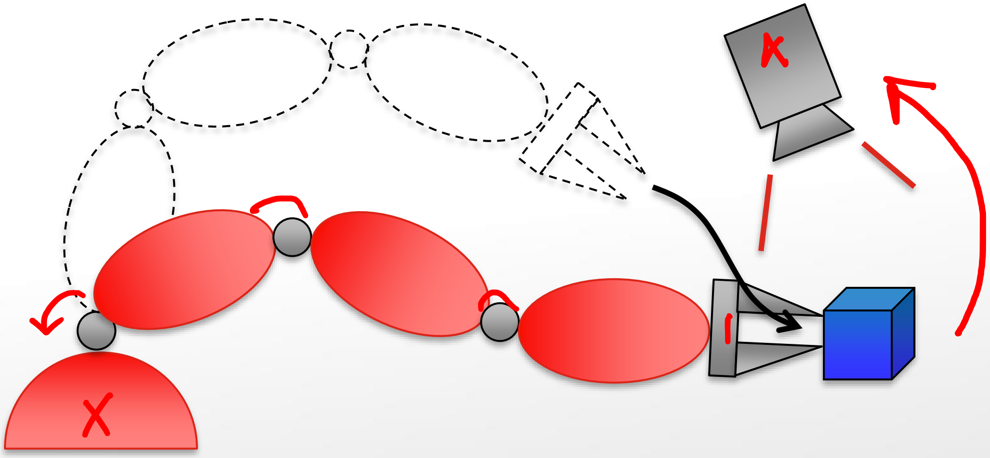
Introduction

- Last week, we have learnt a few techniques in robot vision or image processing to perform:
  - Feature extraction – e.g. detect edges, corners
  - Part identification – e.g. selecting conical shaped parts out of many different parts.
- Today, we will learn about:
  - Pose estimation – obtaining the 3D pose (translation and orientation) of parts, to allow robotic handling.

Robot identifying parts and esimating the 3D pose
https://i.ytimg.com/vi/mQpVCSM8Vgc/maxresdefault.jpg

Introduction

- The idea behind 3D pose estimation is to estimate the position and orientation of the object, with respect to a camera (location known to robot).
- Once these are known, we can command the robot to manipulate the object.



Introduction

- Estimation of the position/orientation of camera can be captured under the topic "Camera Calibration".
- The goal of camera calibration is to find out:
  - The intrinsic parameters of the camera: Resolution
    - Focal length
    - Scaling factor
    - Distortion
    - Etc.
  - The extrinsic parameters of the camera:
    - Translation to world coordinate frame
    - Rotation to world coordinate frame

This is what we were looking for

- We will obtain both the intrinsic and extrinsic parameters through the process of calibration, the latter representing the 3D pose of the camera.

Content

- Segmenting Multiple Blobs
- 3D Pose Estimation for Known Objects
  - Introduction
  - Camera Intrinsic Parameters
  - Camera Extrinsic Parameters
  - Camera Calibration
  - 3D Pose Estimation
- Depth Perception for Arbitrary Objects
  - Introduction
  - Stereo Disparity
  - Correspondence Problem
  - Non-coplanar Cameras

Image Formation

- Pinhole Projection Model:
  - Light ray comes through the pinhole (camera center), and is projected onto the film or CCD, which is at focal length, f, distance away from pinhole.

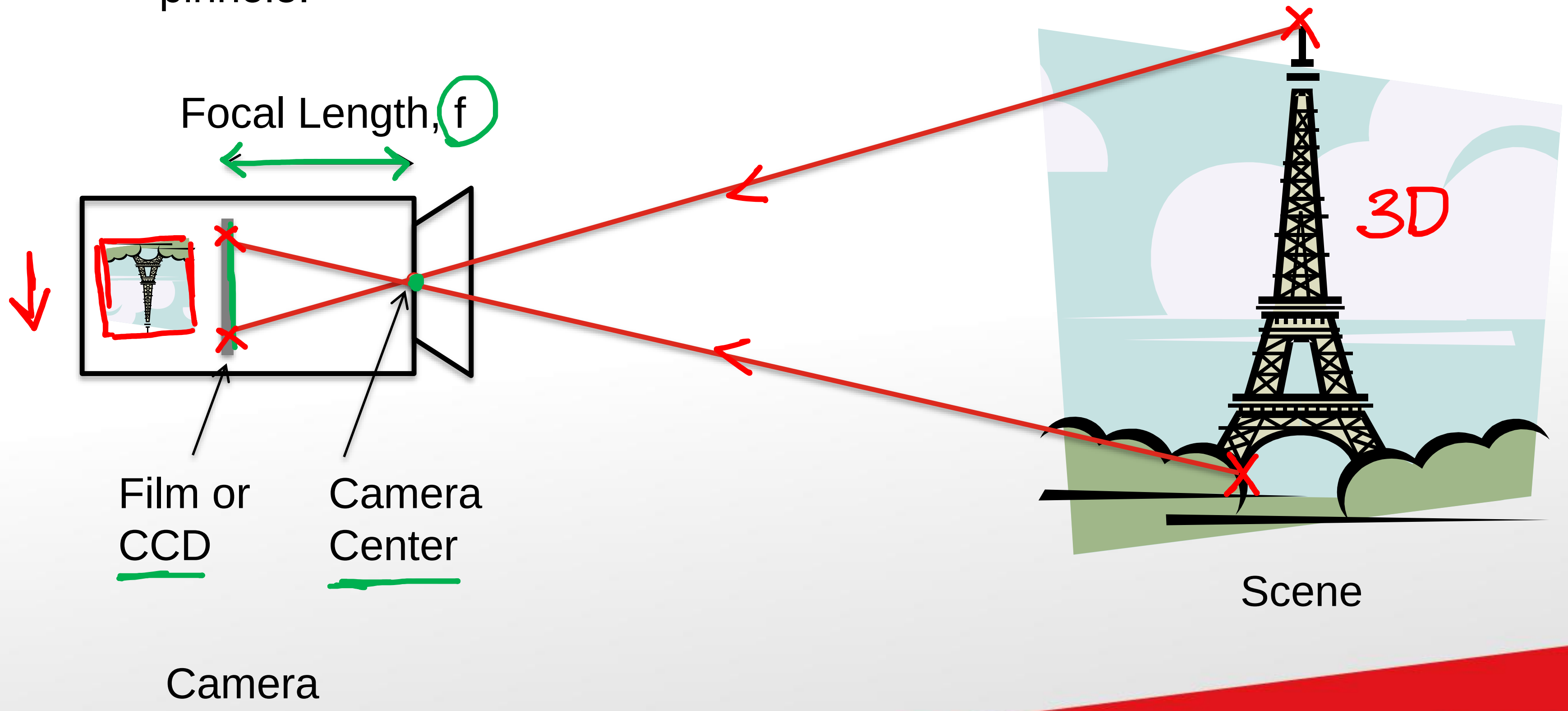


Image Formation

- It is obvious that the image will become upside down.
- To simplify calculation, it is proposed to have a "virtual" image plane at distance f in front of the camera instead, so that the image is not rotated.

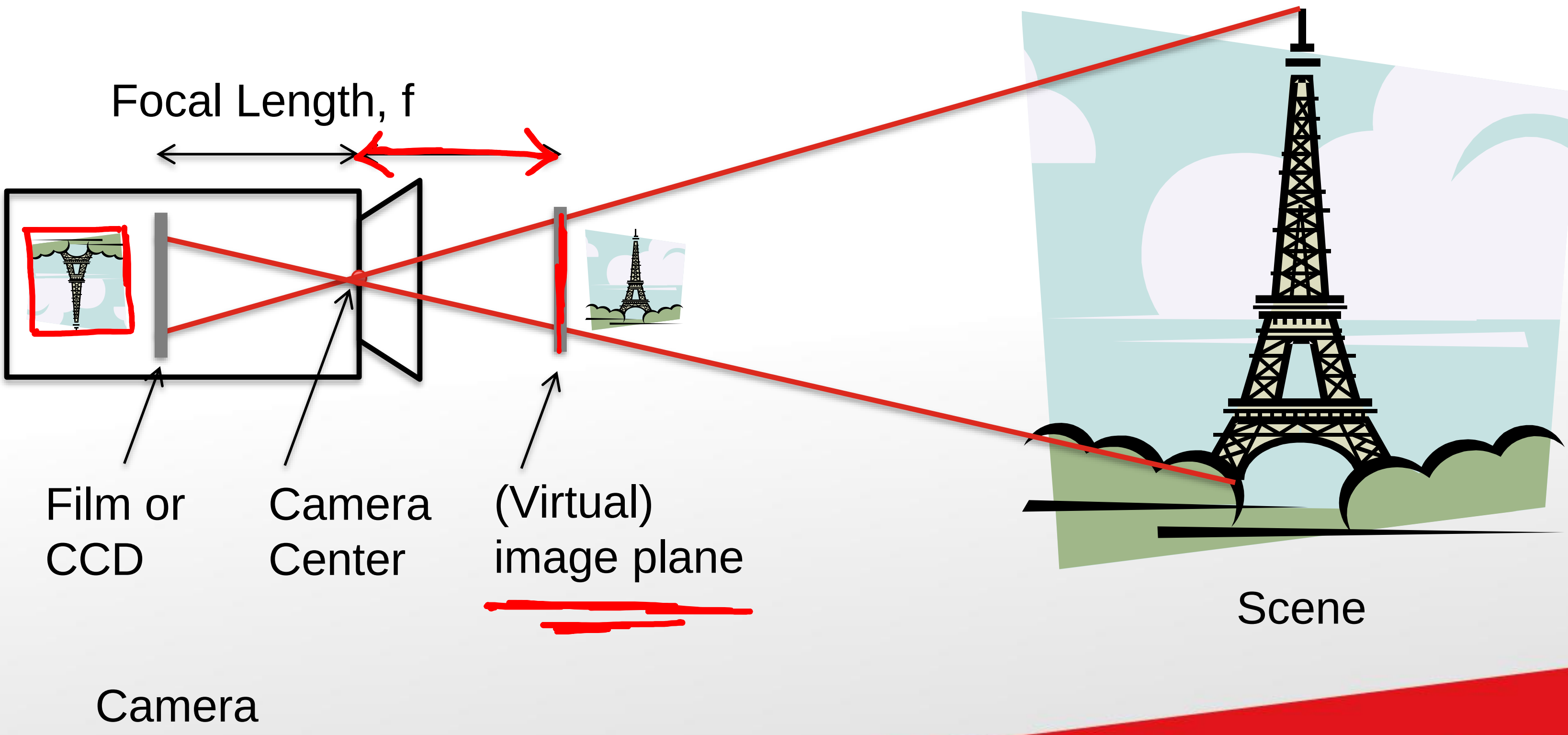


Image Formation

- The scenario is thus as follows:

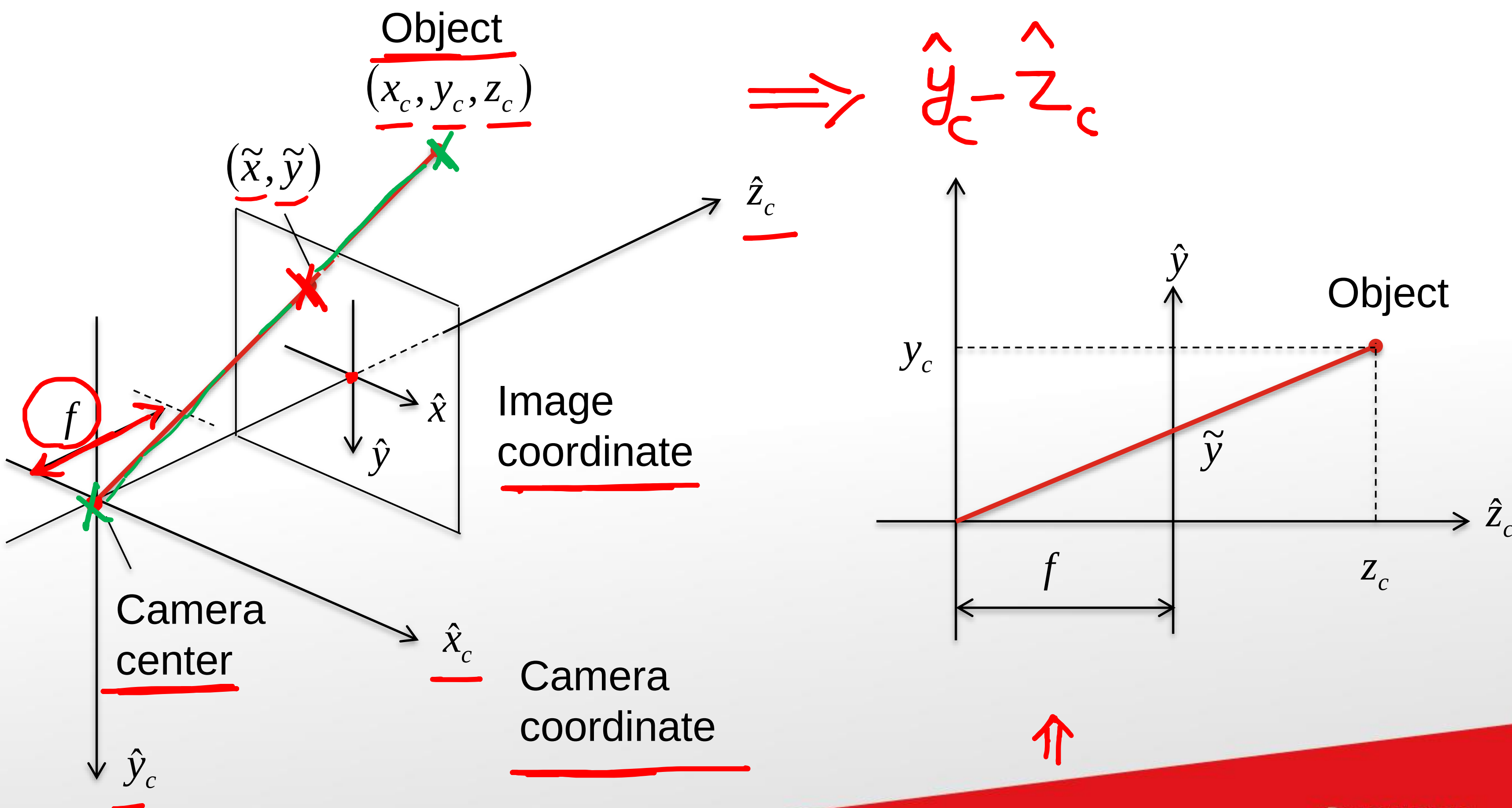
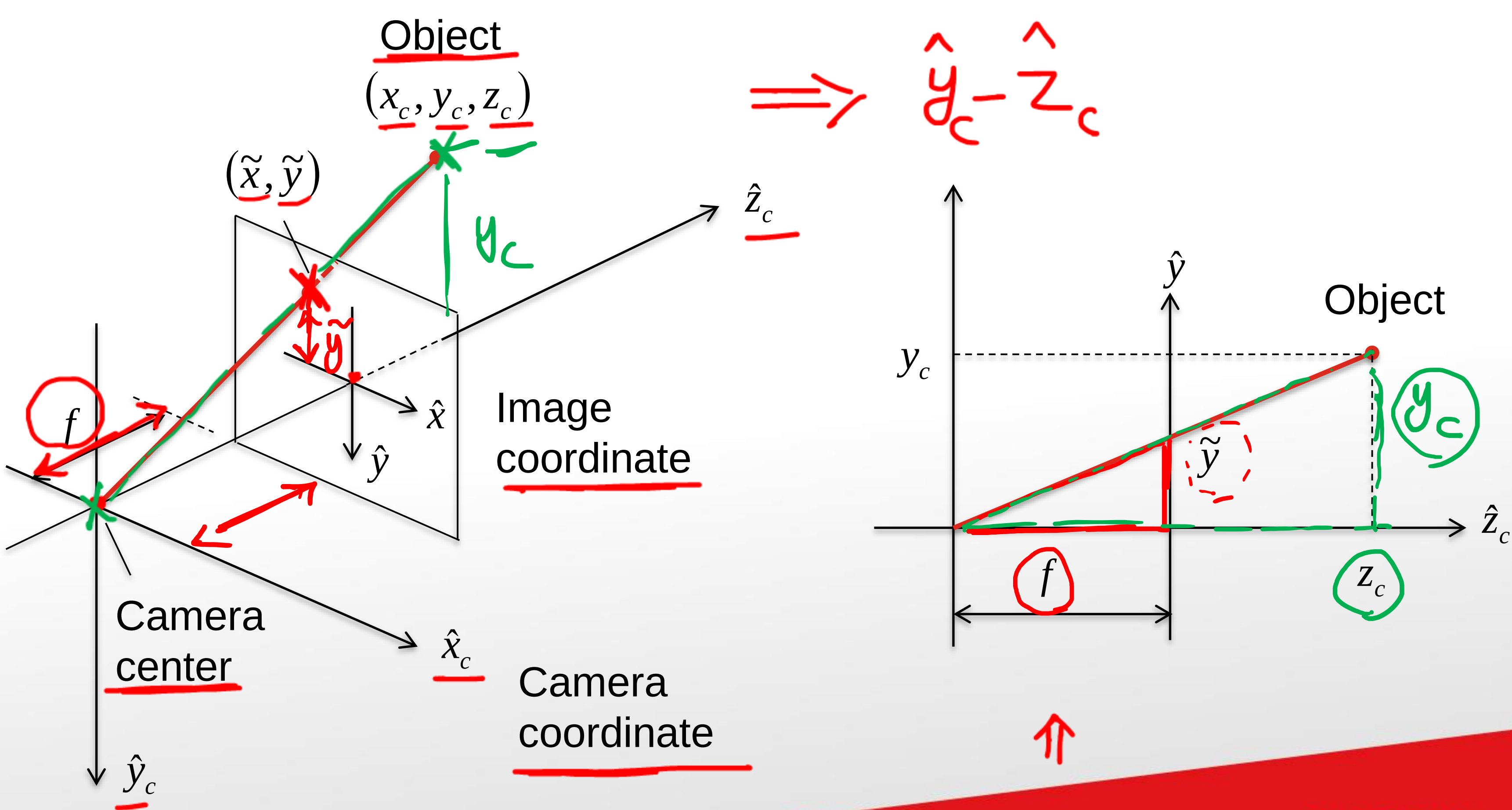


Image Formation

- The scenario is thus as follows:



Pinhole Projection Equation

- From the 2-dimensional sketch, it is easy to see that (due to similar triangles):

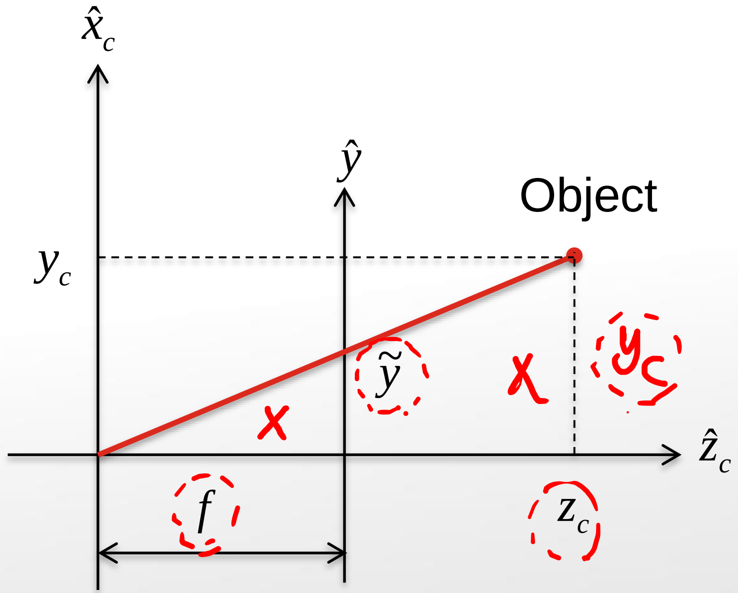
$$\Rightarrow \quad \frac{\tilde{y}}{f} = \frac{y_c}{z_c}$$

- This gives:

$$\Rightarrow \quad \tilde{y} = f\,\frac{y_c}{z_c}$$

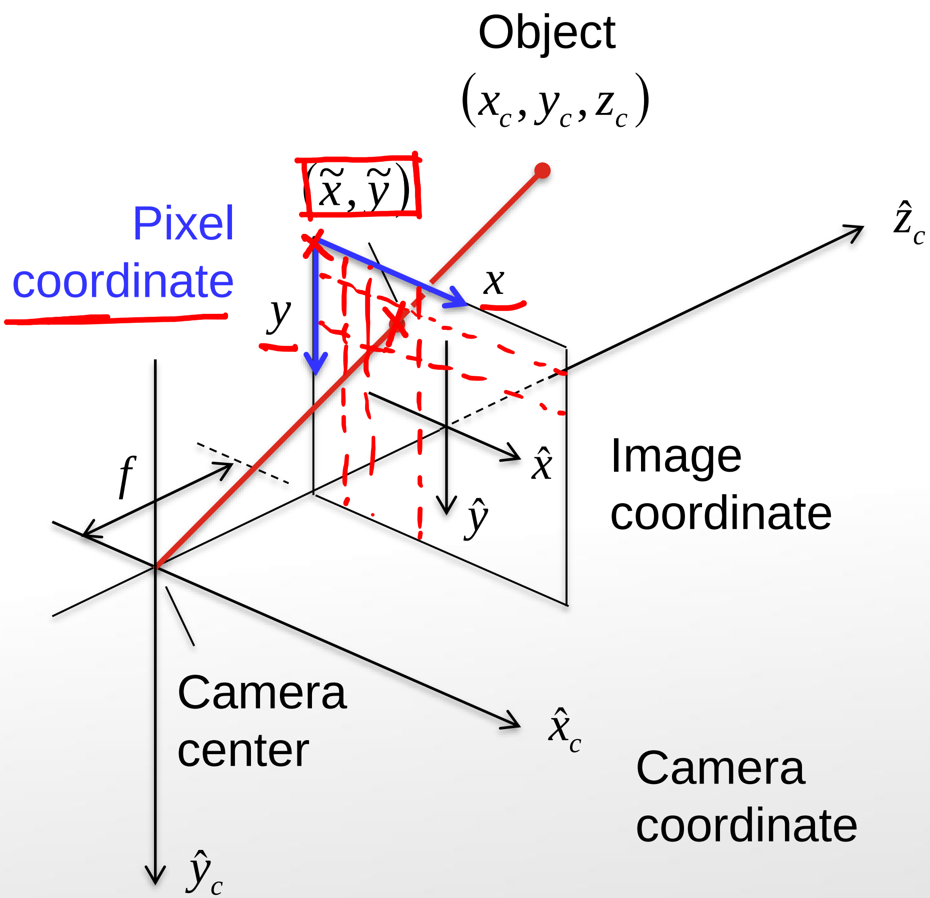
- Similarly, we will have:

$$\Rightarrow \quad \tilde{x} = f\,\frac{x_c}{z_c}$$



Pixel Value

- The point location in the image coordinate will then need to be given in terms of the pixels.



- With reference to the pixel coordinate system, the point $(\tilde{x}, \tilde{y})$ has the value:

Location in image plane

Shift the center (0,0) of image to a corner

$$\Rightarrow \quad x = \frac{\tilde{x}}{dx} + x_0 \quad \Rightarrow \quad y = \frac{\tilde{y}}{dy} + y_0$$

Location in terms of pixels

Scale by physical dimension of pixel

Pixel Value

- The point location in the image coordinate will then need to be given in terms of the pixels.

- With reference to the pixel coordinate system, the point $(\tilde{x}, \tilde{y})$ has the value:

Location in image plane

Shift the center (0,0) of image to a corner

$$x = \frac{\tilde{x}}{dx} + x_0 \qquad y = \frac{\tilde{y}}{dy} + y_0$$

Location in terms of pixels

Scale by physical dimension of pixel

- For example:
  - If the x-location of a point in image plane is $\underline{\tilde{x} = 3\mu m}$,
  - And if the dimension of a pixel is $\underline{dx = 1.5\mu m}$,
  - Then the pixel value (ignoring the translation) is 2.

$$\frac{3}{1.5} = \underbrace{2 + x_0}$$

Camera Calibration Matrix

- Combining all equations we have so far, i.e.
  - From camera coordinate system to image coordinate system:

$$\tilde{x} = f\frac{x_c}{z_c} \qquad \tilde{y} = f\frac{y_c}{z_c}$$

3D

  - From image coordinate system to pixel coordinate system:

$$x = \frac{\tilde{x}}{dx} + x_0 \qquad y = \frac{\tilde{y}}{dy} + y_0$$

  - We can write:

$$\boxed{x = \frac{f}{dx}\frac{x_c}{z_c} + x_0 \qquad y = \frac{f}{dy}\frac{y_c}{z_c} + y_0}$$

Camera Calibration Matrix

- The final equations,

$$\Rightarrow \quad x = \frac{f}{dx}\frac{x_c}{z_c} + x_0 \qquad y = \frac{f}{dy}\frac{y_c}{z_c} + y_0$$

- Can be expressed in a matrix form (homogeneous form, i.e. adds a component to a 2D vector to make it a 3D vector) :

$x \sim \alpha_x x_c + x_0 z_c$

$y \sim \alpha_y y_c + y_0 z_c$

$1 \sim z_c$

$$\boxed{\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \underbrace{\begin{bmatrix} \alpha_x & 0 & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}}_{\substack{3\times3 \\ K}} \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}}$$

or

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim K \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}$$

Note: This is proportional sign, NOT equal sign.

- Where:

$$\boxed{K = \begin{bmatrix} \alpha_x & 0 & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}} \qquad \alpha_x = \frac{f}{dx} \qquad \alpha_y = \frac{f}{dy}$$

is called the Camera Calibration Matrix.

Camera Calibration Matrix

- How does the equation work?

$$\Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \overset{\rho}{\sim} \begin{bmatrix} \alpha_x & 0 & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} \qquad \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = S \begin{bmatrix} \alpha_x & 0 & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}$$

- The proportional sign means "Equal up to Scale".
- The equation gives:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} \alpha_x x_c + x_0 z_c \\ \alpha_y y_c + y_0 z_c \\ z_c \end{bmatrix} \qquad 1 = 1$$

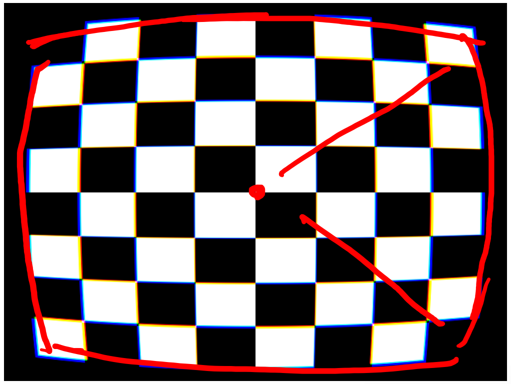
- It is clear that the row should be 1 = 1. Therefore, we divide the right hand by z_c and get:

$$\Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha_x \dfrac{x_c}{z_c} + x_0 \\ \alpha_y \dfrac{y_c}{z_c} + y_0 \\ 1 \end{bmatrix} = \begin{bmatrix} \dfrac{f}{d_x} \dfrac{x_c}{z_c} + x_0 \\ \dfrac{f}{d_y} \dfrac{y_c}{z_c} + y_0 \\ 1 \end{bmatrix}$$

same eq. from the previous

Distortion

- The pinhole camera model is not necessarily valid for all camera.
- Most images suffer from lens distortion:
- Barrel Distortion:



  - A type of "radial distortion".
  - The amount of "bulging out" depends on how far a point is from the center.

Distortion

- The relationship between undistorted and distorted point (in image coordinate system) is:

$$\Rightarrow \quad \begin{bmatrix} \tilde{x}_{dist} \\ \tilde{y}_{dist} \end{bmatrix} = \left(1 + K_1 r^2 + K_2 r^4\right) \begin{bmatrix} \tilde{x}_{un} \\ \tilde{y}_{un} \end{bmatrix}$$
$$= \left(1 + K_1\left(\tilde{x}_{un}^2 + \tilde{y}_{un}^2\right) + K_2\left(\tilde{x}_{un}^2 + \tilde{y}_{un}^2\right)^2\right) \begin{bmatrix} \tilde{x}_{un} \\ \tilde{y}_{un} \end{bmatrix}$$

  - We can stop at r^2 if the distortion not serious, or we can go up to higher degree if distortion is serious.

- We can estimate K1 and K2 using checkerboard, for e.g. using Least Squares Algorithm.
- Then, to undo the distortion, we can use the inverse relationship between distorted and undistorted point.
- For the remainder of this lecture, we will not consider this distortion effect.

RMIT UNIVERSITY

Content

- Segmenting Multiple Blobs
- 3D Pose Estimation for Known Objects
  - Introduction
  - Camera Intrinsic Parameters
  - Camera Extrinsic Parameters
  - Camera Calibration
  - 3D Pose Estimation
- Depth Perception for Arbitrary Objects
  - Introduction
  - Stereo Disparity
  - Correspondence Problem
  - Non-coplanar Cameras

$$K = \begin{bmatrix} \alpha_x & 0 & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}$$

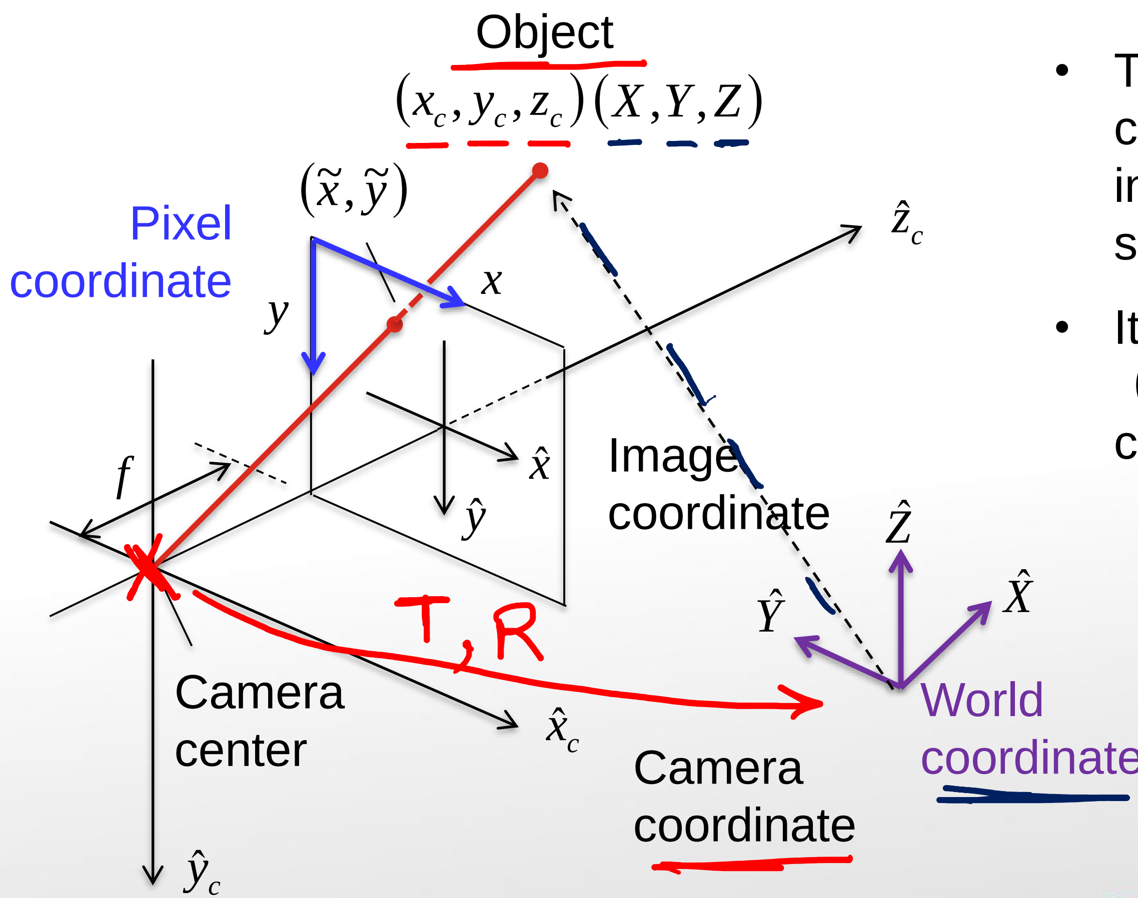
$\alpha_x, \alpha_y, x_0, y_0$

$T = \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix}$ & R betw. CCS & WCS

Extrinsic Parameters

$$ {}^{B}P,\ {}^{A}_{B}\underbrace{\begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix}}_{T} \rightarrow {}^{A}P $$

- The extrinsic parameters give the relationship between the World Coordinate System and the Camera Coordinate System.



- The object point has coordinates (x_c, y_c, z_c) in Camera coordinate system.
- It also has coordinates (X, Y, Z) in World coordinate system.

Extrinsic Parameters

- We can convert the point from World Coordinate System to Camera Coordinate System by a rotation and translation:

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

  - R = Orientation of World Coordinate System wrt. Camera Coordinate System.
  - T = Position of the origin of World Coordinate System expressed in Camera Coordinate System.
- The values of the rotation matrix and translation vector are what we call the Extrinsic Parameters of a camera.

Content

- Segmenting Multiple Blobs
- 3D Pose Estimation for Known Objects
  - Introduction
  - Camera Intrinsic Parameters
  - Camera Extrinsic Parameters
  - Camera Calibration
  - 3D Pose Estimation
- Depth Perception for Arbitrary Objects
  - Introduction
  - Stereo Disparity
  - Correspondence Problem
  - Non-coplanar Cameras

Camera Matrix

- Summary:
- The extrinsic parameters give relationship between World Coordinate System (X, Y, Z) and Camera Coordinate System (x_c, y_c, z_c):

$$\Rightarrow \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

- The intrinsic parameters give relationship between Camera Coordinate System (x_c, y_c, z_c) and Pixel Coordinate System (x, y, z):

$$\Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim K \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}$$

Camera Matrix

- We can combine the both to get:

$$\Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim K \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = K\left(R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \right) = K\begin{bmatrix} R & T \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = P \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

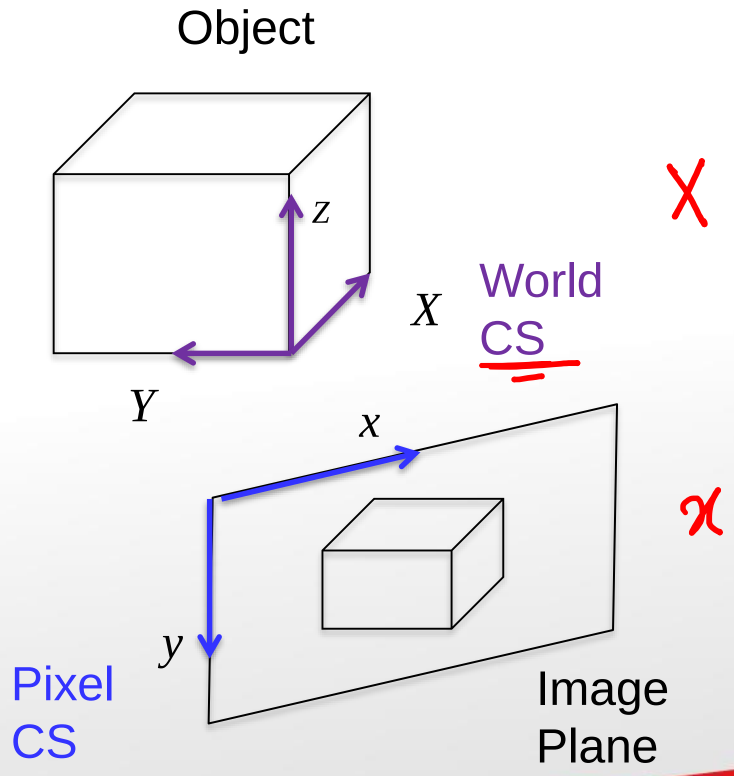
- i.e.:

$$\boxed{\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim P \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}}$$

- Where $P = K\begin{bmatrix} R & T \end{bmatrix}$ is called the Camera Matrix. (Not to be confused with Camera Calibration Matrix K).

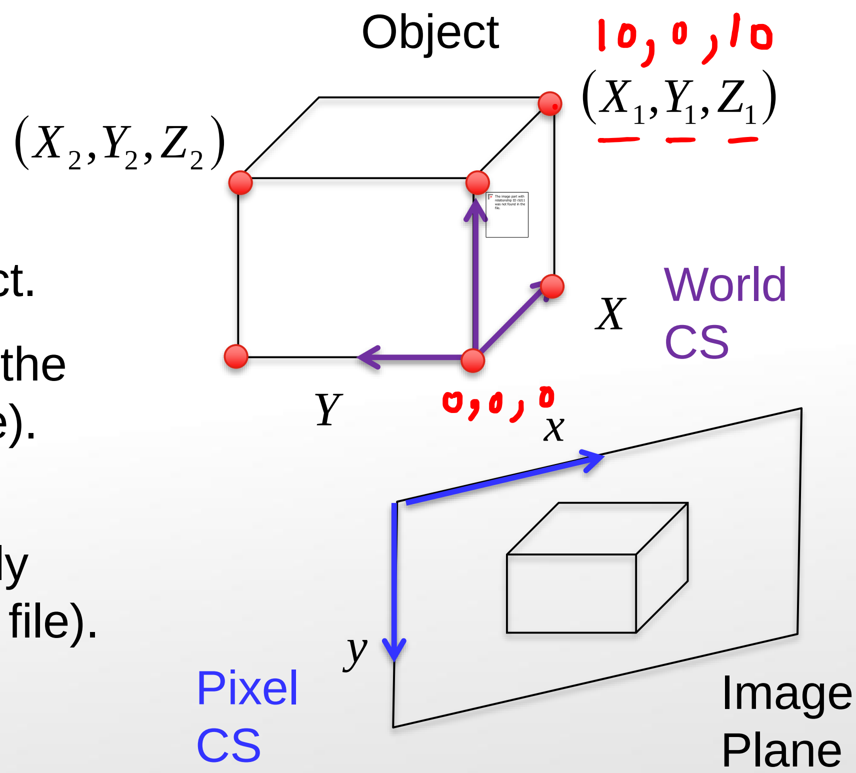
Camera Calibration

- But how do we get P?
- This is the goal of camera calibration (also called resectioning) → To estimate P from known x and X.
- Imagine the following scenario:
- Now, do the following:
  - Attach the World CS onto the object.



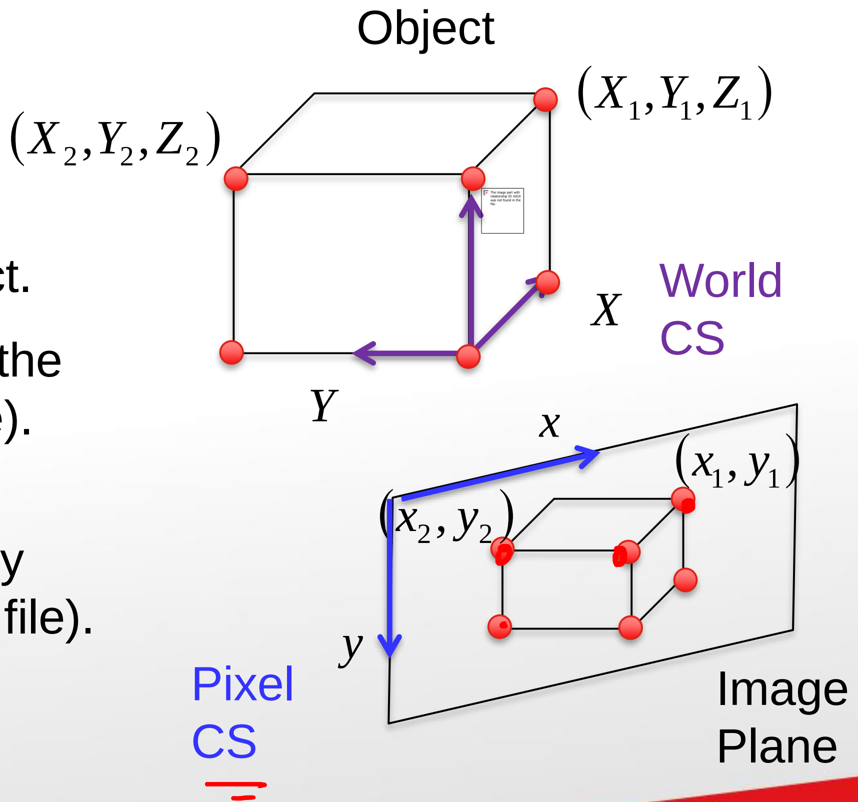
Camera Calibration

- But how do we get P?
- This is the goal of camera calibration (also called resectioning) → To estimate P from known x and X.
- Imagine the following scenario:
- Now, do the following:
  - Attach the World CS onto the object.
  - Then choose at least six points on the object (Not all on the same Z-plane).
  - The location of these points with reference to World CS can be easily determined (measurement or CAD file).



Camera Calibration

- But how do we get P?
- This is the goal of camera calibration (also called resectioning) → To estimate P from known x and X.
- Imagine the following scenario:
- Now, do the following:
  - Attach the World CS onto the object.
  - Then choose at least six points on the object (Not all on the same Z-plane).
  - The location of these points with reference to World CS can be easily determined (measurement or CAD file).
  - Determine the pixel value of the corresponding points on the image plane.



Camera Calibration

$$\underbrace{P}_{3\times4} = \underset{3\times3}{K}\,\overbrace{[\underset{3\times3}{R}\ \underset{3\times1}{T}]}^{3\times4}$$

- For each point, we have:

$$\Rightarrow \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \sim P \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \sim \underbrace{\begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix}}_{3\times4} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

$P_{11}, P_{12}, \ldots$

— — —

— — , P_{34}

- Remember, the relationship is only "proportional", not equal. How can we solve it?
- The proportionality means that $[x_i \quad y_i \quad 1]^T$ is a scalar multiple of $P[X_i \quad Y_i \quad Z_i \quad 1]^T$
- Therefore, their cross product is zero.

Camera Calibration

- In other words:

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \times \begin{bmatrix} p_{11}X_i + p_{12}Y_i + p_{13}Z_i + p_{14} \\ p_{21}X_i + p_{22}Y_i + p_{23}Z_i + p_{24} \\ p_{31}X_i + p_{32}Y_i + p_{33}Z_i + p_{34} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{vmatrix} i & j & k \\ x_i & y_i & 1 \\ \begin{pmatrix} p_{11}X_i + p_{12}Y_i \\ + p_{13}Z_i + p_{14} \end{pmatrix} & \begin{pmatrix} p_{21}X_i + p_{22}Y_i \\ + p_{23}Z_i + p_{24} \end{pmatrix} & \begin{pmatrix} p_{31}X_i + p_{32}Y_i \\ + p_{33}Z_i + p_{34} \end{pmatrix} \end{vmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{cases} y_i(p_{31}X_i + p_{32}Y_i + p_{33}Z_i + p_{34}) - (p_{21}X_i + p_{22}Y_i + p_{23}Z_i + p_{24}) = 0 \\ x_i(p_{31}X_i + p_{32}Y_i + p_{33}Z_i + p_{34}) - (p_{11}X_i + p_{12}Y_i + p_{13}Z_i + p_{14}) = 0 \end{cases}$$

- (Only two independent equations).

Camera Calibration

- From the last equation, we can write:

$$\overbrace{\begin{bmatrix} 0 & 0 & 0 & 0 & X_i & Y_i & Z_i & 1 & -y_iX_i & -y_iY_i & -y_iZ_i & -y_i \\ X_i & Y_i & Z_i & 1 & 0 & 0 & 0 & 0 & -x_iX_i & -x_iY_i & -x_iZ_i & -x_i \end{bmatrix}}^{\text{known values}} \overbrace{\begin{bmatrix} p_{11} \\ p_{12} \\ p_{13} \\ p_{14} \\ p_{21} \\ p_{22} \\ p_{23} \\ p_{24} \\ p_{31} \\ p_{32} \\ p_{33} \\ p_{34} \end{bmatrix}}^{\text{unknown P's (12)}} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

- There are 12 parameters but only 2 equations, for one point.
- Not solvable.

Camera Calibration

- If we now use 6 or more points, we can obtain:

$$
\begin{array}{c}
1 \\
\\
2 \\
\\
\vdots \\
6 \\
\\
\end{array}
\begin{bmatrix}
0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -y_1X_1 & -y_1Y_1 & -y_1Z_1 & -y_1 \\
X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & -x_1X_1 & -x_1Y_1 & -x_1Z_1 & -x_1 \\
0 & 0 & 0 & 0 & X_2 & Y_2 & Z_2 & 1 & -y_2X_2 & -y_2Y_2 & -y_2Z_2 & -y_2 \\
X_2 & Y_2 & Z_2 & 1 & 0 & 0 & 0 & 0 & -x_2X_2 & -x_2Y_2 & -x_2Z_2 & -x_2 \\
 & & & & & & & \vdots & & & & \\
0 & 0 & 0 & 0 & X_n & Y_n & Z_n & 1 & -y_nX_n & -y_nY_n & -y_nZ_n & -y_n \\
X_n & Y_n & Z_n & 1 & 0 & 0 & 0 & 0 & -x_nX_n & -x_nY_n & -x_nZ_n & -x_n
\end{bmatrix}
\begin{bmatrix}
p_{11} \\ p_{12} \\ p_{13} \\ p_{14} \\ p_{21} \\ p_{22} \\ p_{23} \\ p_{24} \\ p_{31} \\ p_{32} \\ p_{33} \\ p_{34}
\end{bmatrix}
=
\begin{bmatrix}
0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0
\end{bmatrix}
$$

12 eqs, 12 unknown ✓

Camera Calibration

- The equation is of the form:

$$\Rightarrow \quad \boxed{Ap = 0}$$

- Because it is a homogeneous equation (right hand side equals zero), the solution is not unique.
- There are a few ways to solve for p, for e.g.
  - If exactly six points measured: Find null-space of A. Then pick the one with $\|p\| = 1$.

    Matlab
    - If more than six points are measured, it is not possible to get null space of A due to measurement noise.
  - Minimize $\|Ap\|$ subject to $\|p\| = 1$.
  - Using Singular Value Decomposition of A $A = U\Sigma V^T$.
    - Then set p = last column of V.
  - One more method on the next slide…

Camera Calibration

- We know that $$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \sim P \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$
  - i.e. the equation is correct up to a scale.
- We can arbitrarily fix one element, e.g. $P_{34} = 1$, and then solve for the remaining ones.
- (Continue next slide)

Camera Calibration

- This means:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -y_1X_1 & -y_1Y_1 & -y_1Z_1 & -y_1 \\ X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & -x_1X_1 & -x_1Y_1 & -x_1Z_1 & -x_1 \\ 0 & 0 & 0 & 0 & X_2 & Y_2 & Z_2 & 1 & -y_2X_2 & -y_2Y_2 & -y_2Z_2 & -y_2 \\ X_2 & Y_2 & Z_2 & 1 & 0 & 0 & 0 & 0 & -x_2X_2 & -x_2Y_2 & -x_2Z_2 & -x_2 \\ & & & & & & & \vdots & & & & \\ 0 & 0 & 0 & 0 & X_n & Y_n & Z_n & 1 & -y_nX_n & -y_nY_n & -y_nZ_n & -y_n \\ X_n & Y_n & Z_n & 1 & 0 & 0 & 0 & 0 & -x_nX_n & -x_nY_n & -x_nZ_n & -x_n \end{bmatrix} \begin{bmatrix} p_{11} \\ p_{12} \\ p_{13} \\ p_{14} \\ p_{21} \\ p_{22} \\ p_{23} \\ p_{24} \\ p_{31} \\ p_{32} \\ p_{33} \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

- (Continue next slide)

Camera Calibration

P's (11)

- Or:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -y_1X_1 & -y_1Y_1 & -y_1Z_1 \\ X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & -x_1X_1 & -x_1Y_1 & -x_1Z_1 \\ 0 & 0 & 0 & 0 & X_2 & Y_2 & Z_2 & 1 & -y_2X_2 & -y_2Y_2 & -y_2Z_2 \\ X_2 & Y_2 & Z_2 & 1 & 0 & 0 & 0 & 0 & -x_2X_2 & -x_2Y_2 & -x_2Z_2 \\ & & & & & & \vdots & & & & \\ 0 & 0 & 0 & 0 & X_n & Y_n & Z_n & 1 & -y_nX_n & -y_nY_n & -y_nZ_n \\ X_n & Y_n & Z_n & 1 & 0 & 0 & 0 & 0 & -x_nX_n & -x_nY_n & -x_nZ_n \end{bmatrix} \begin{bmatrix} p_{11} \\ p_{12} \\ p_{13} \\ p_{14} \\ p_{21} \\ p_{22} \\ p_{23} \\ p_{24} \\ p_{31} \\ p_{32} \\ p_{33} \end{bmatrix} = \begin{bmatrix} y_1 \\ x_1 \\ y_2 \\ x_2 \\ \vdots \\ y_n \\ x_n \end{bmatrix}$$

$$\Rightarrow \quad \tilde{A}\tilde{P} = \theta$$

- With this, the vector p can be calculated using least squares method, i.e.

$$\Rightarrow \quad \tilde{P} = \left(\tilde{A}^T \tilde{A}\right)^{-1} \tilde{A}^T \theta$$

Content

- Segmenting Multiple Blobs
- 3D Pose Estimation for Known Objects
  - Introduction
  - Camera Intrinsic Parameters
  - Camera Extrinsic Parameters
  - Camera Calibration $\rightarrow$ solve $P_{11}, P_{12}, \ldots$
  - 3D Pose Estimation $\rightarrow$ $P \rightarrow$ Extract R, T
- Depth Perception for Arbitrary Objects
  - Introduction
  - Stereo Disparity
  - Correspondence Problem
  - Non-coplanar Cameras

$\left.\right\}$ $P = K[R\ T]$ $P_{11}, P_{12}, \ldots$ unknown

Recovering the Parameters

- In the last section, we have obtained the matrix P.
- We now need to recover all the individual parameters (intrinsic and extrinsic) from the matrix P.
- We split the (3 x 4) matrix P into: $P = [P_1 \quad P_2]$
- Also, recall that: $P = K[R \quad T]$
  - Therefore:

$K = \begin{bmatrix} \alpha_x & 0 & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}$ $\Rightarrow$ $\underbrace{P_1}_{3\times3} = K \cdot R$ $\Rightarrow$ $\underbrace{P_2}_{3\times1} = K \cdot T$ $\Rightarrow T = K^{-1} \cdot P_2$

- For P_1, K is an upper triangular matrix, and R is orthogonal (rotation matrix).
  - There is a standard algorithm, called RQ decomposition to solve it.
  - Thus, assume we have K and R now.
- With known K, we can then calculate T from: $T = K^{-1} \cdot P_2$

Some Details

- Note, in MATLAB we only have QR decomposition. (Q orthogonal and R upper triangular)
- However, what we need is RQ decomposition.
- Trick: use inverse, i.e.:
  - We know $\underbrace{P_1}_{3\times 3} = \underbrace{K}_{\substack{upper\\ triangle}} \cdot \underbrace{R}_{orthogonal}$
  - Then $\underbrace{P_1^{-1}}_{3\times 3} = \left(\underbrace{K}_{\substack{upper\\ triangle}} \cdot \underbrace{R}_{orthogonal} \right)^{-1} = \underbrace{R^{-1}}_{orthogonal} \cdot \underbrace{K^{-1}}_{\substack{upper\\ triangle}}$
  - This is suitable for QR decomposition. → Matlab $[Rinv, Kinv] = qr(P1inv)$
  - After decomposition, we then invert $Rinv$ and $Kinv$ to get R and K

Some Details

- Another issue with the RQ decomposition is that the answer is not unique!
  - Sometimes we might get negative diagonal elements of K, which is weird because if the camera looks in positive direction, f must be positive.

$$K = \begin{bmatrix} \alpha_x & 0 & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \alpha_x = \frac{f}{dx} \qquad \alpha_y = \frac{f}{dy}$$

- Solution:

  x(-1)

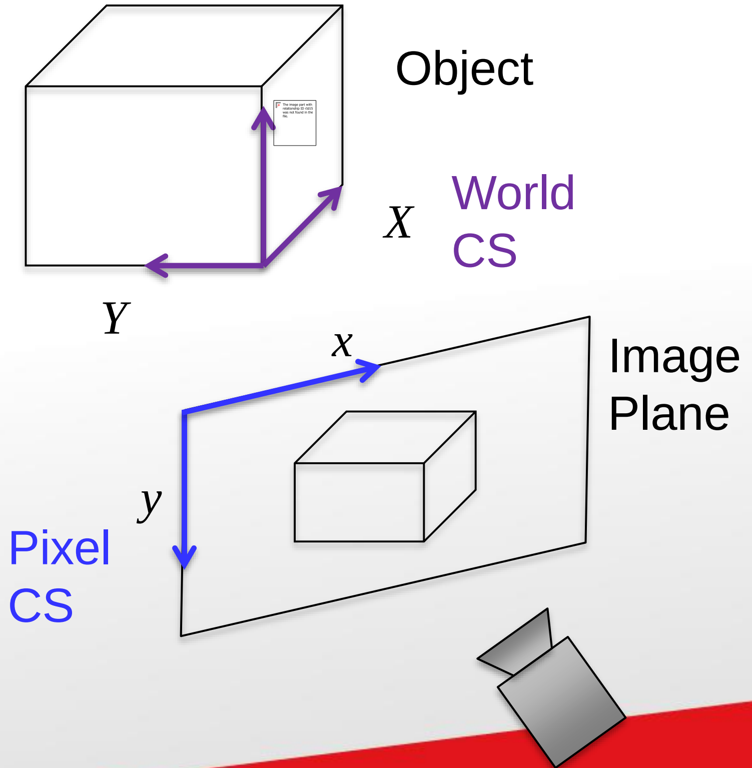
  - Notice that if any column of K is negated, and the corresponding row of R is also negated, then P_1 = KR is still the same.
  - Therefore, we can force the diagonal terms of K to be positive.

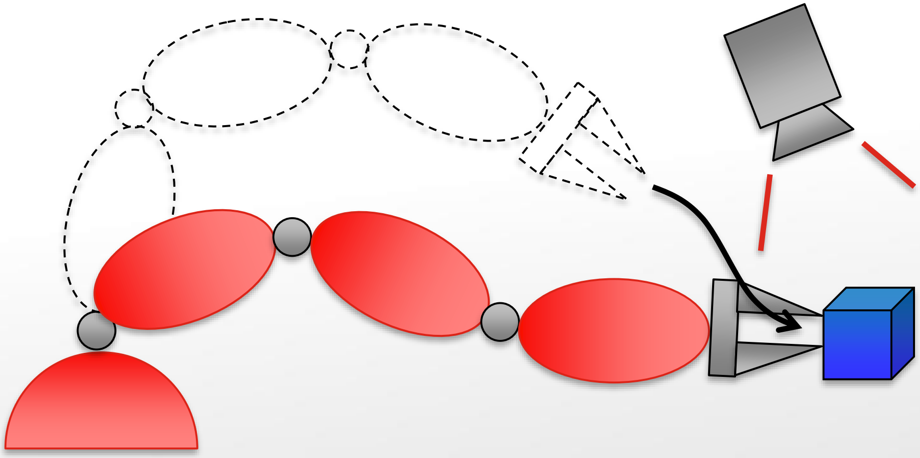
3D Pose Estimation

P

QR

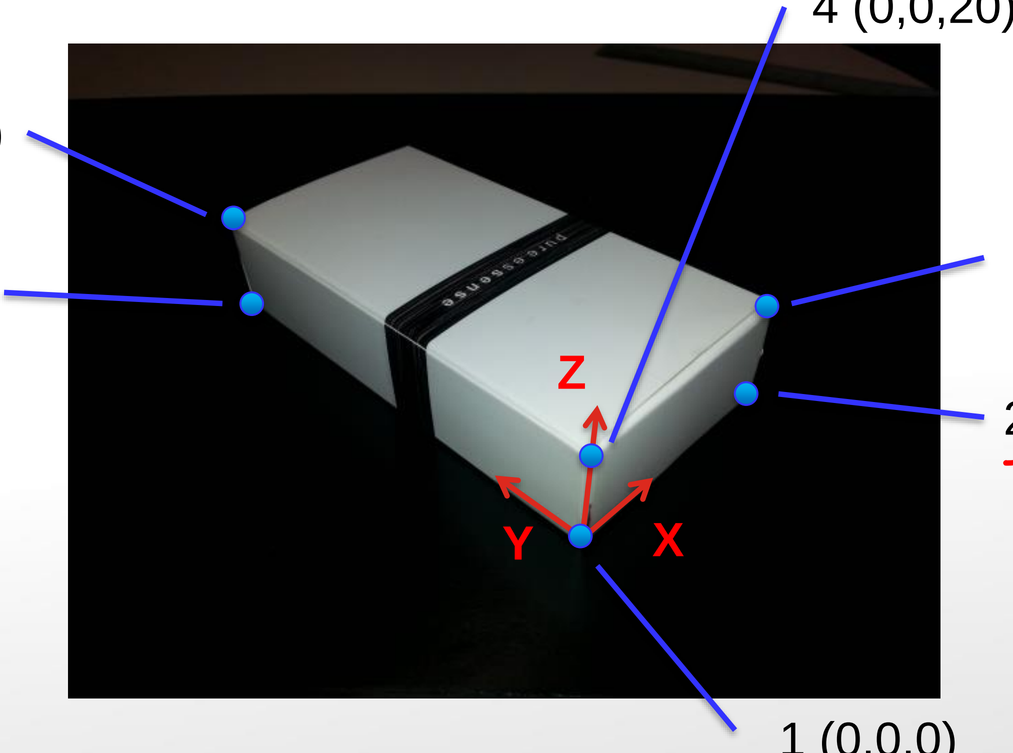
- Up to this stage, we have already calculated the R and T matrices.
- Thus, we have already estimated the 3D pose of the camera w.r.t. the world frame (also object, since we attach the world frame onto the object).
- Finally, we can command the robot manipulator to move towards the object and grasp it.





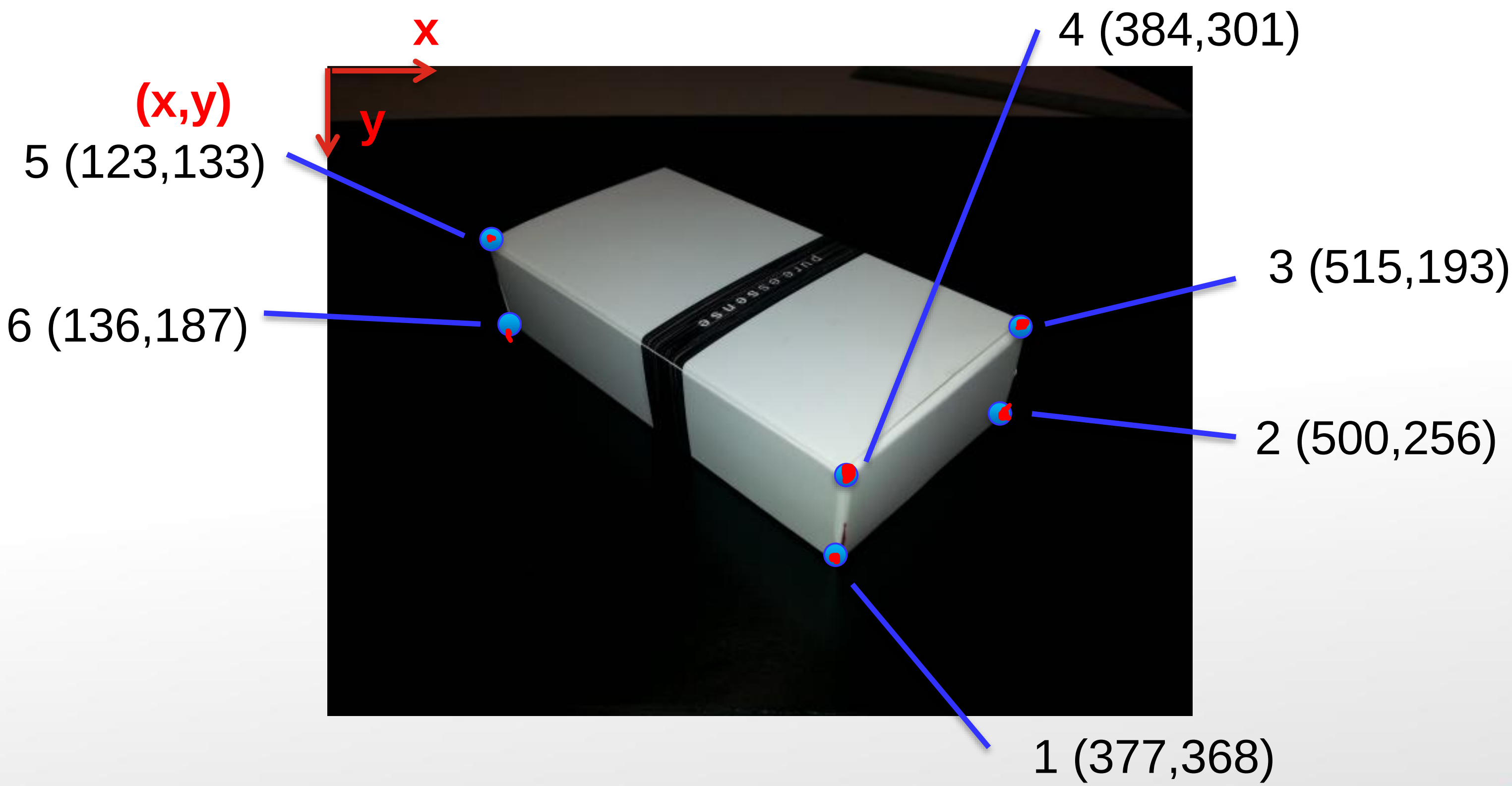
Complete Example

- Following is a box with known dimension.
- A frame is fixed at one of the vertices and the other points are given wrt. the frame.



Complete Example

- The pixel coordinates of the points are as follows:



Complete Example

- Thus in summary, we have:

WCS

$X_1 = 0$	$Y_1 = 0$	$Z_1 = 0$
$X_2 = 45$	$Y_2 = 0$	$Z_2 = 0$
$X_3 = 45$	$Y_3 = 0$	$Z_3 = 20$
$X_4 = 0$	$Y_4 = 0$	$Z_4 = 20$
$X_5 = 0$	$Y_5 = 90$	$Z_5 = 20$
$X_6 = 0$	$Y_6 = 90$	$Z_6 = 0$

PCS

$x_1 = 377$	$y_1 = 368$
$x_2 = 500$	$y_2 = 256$
$x_3 = 515$	$y_3 = 193$
$x_4 = 384$	$y_4 = 301$
$x_5 = 123$	$y_5 = 133$
$x_6 = 136$	$y_6 = 187$

Complete Example

- We can then set the matrix equation below using the numerical values from the previous page:

$$
\overbrace{\begin{bmatrix}
0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -y_1X_1 & -y_1Y_1 & -y_1Z_1 \\
X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & -x_1X_1 & -x_1Y_1 & -x_1Z_1 \\
0 & 0 & 0 & 0 & X_2 & Y_2 & Z_2 & 1 & -y_2X_2 & -y_2Y_2 & -y_2Z_2 \\
X_2 & Y_2 & Z_2 & 1 & 0 & 0 & 0 & 0 & -x_2X_2 & -x_2Y_2 & -x_2Z_2 \\
 & & & & & & \vdots & & & & \\
0 & 0 & 0 & 0 & X_n & Y_n & Z_n & 1 & -y_nX_n & -y_nY_n & -y_nZ_n \\
X_n & Y_n & Z_n & 1 & 0 & 0 & 0 & 0 & -x_nX_n & -x_nY_n & -x_nZ_n
\end{bmatrix}}^{\text{known}}
\overbrace{\begin{bmatrix} p_{11} \\ p_{12} \\ p_{13} \\ p_{14} \\ p_{21} \\ p_{22} \\ p_{23} \\ p_{24} \\ p_{31} \\ p_{32} \\ p_{33} \end{bmatrix}}^{\text{unknown (11)}}
=
\overbrace{\begin{bmatrix} y_1 \\ x_1 \\ y_2 \\ x_2 \\ \vdots \\ y_n \\ x_n \end{bmatrix}}^{\text{known}}
$$

Complete Example

declare $X_1 = --$
$Y_1 = --$
$Z_1 = --$
$1, = --$

- The MATLAB Code is as follows:

```
LHS = [0 0 0 0 X1 Y1 Z1 1 -y1*X1 -y1*Y1 -y1*Z1;
       X1 Y1 Z1 1 0 0 0 0 -x1*X1 -x1*Y1 -x1*Z1;
       0 0 0 0 X2 Y2 Z2 1 -y2*X2 -y2*Y2 -y2*Z2;
       X2 Y2 Z2 1 0 0 0 0 -x2*X2 -x2*Y2 -x2*Z2;
       0 0 0 0 X3 Y3 Z3 1 -y3*X3 -y3*Y3 -y3*Z3;
       X3 Y3 Z3 1 0 0 0 0 -x3*X3 -x3*Y3 -x3*Z3;
       0 0 0 0 X4 Y4 Z4 1 -y4*X4 -y4*Y4 -y4*Z4;
       X4 Y4 Z4 1 0 0 0 0 -x4*X4 -x4*Y4 -x4*Z4;
       0 0 0 0 X5 Y5 Z5 1 -y5*X5 -y5*Y5 -y5*Z5;
       X5 Y5 Z5 1 0 0 0 0 -x5*X5 -x5*Y5 -x5*Z5;
       0 0 0 0 X6 Y6 Z6 1 -y6*X6 -y6*Y6 -y6*Z6;
       X6 Y6 Z6 1 0 0 0 0 -x6*X6 -x6*Y6 -x6*Z6];

RHS = [y1 x1 y2 x2 y3 x3 y4 x4 y5 x5 y6 x6]';

P = LHS\RHS;
```

Complete Example

- The MATLAB Code continued…

P

$P = [P_1 \;\; P_2] \quad P_1 = KR$

```
%%%%%%%%%%%%%%%%%%%%%%%
% Getting K, R from P %
%%%%%%%%%%%%%%%%%%%%%%%

P1 = [P(1) P(2) P(3);
    P(5) P(6) P(7);
    P(9) P(10) P(11)];

P1inv = inv(P1);
[Rinv, Kinv] = qr(P1inv);
K = inv(Kinv);
R = inv(Rinv);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% make diagonal of K positive %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

SIGNS = diag(sign(diag(K)));

K = K * SIGNS
R = SIGNS * R % Orientation of world CS wrt. camera-centered CS
```

Complete Example

- The MATLAB Code continued…

```
%%%%%%%%%%%%%%%%%%%%
% Getting T from P %
%%%%%%%%%%%%%%%%%%%%

P2 = [P(4) P(8) 1]'; % Recall that P34 = P(12) = 1

T = inv(K)*P2 % Origin of the world CS expressed in camera-centered CS
```

$T = K^{-1} P_2$

Complete Example

- And the answer given by MATLAB is:

```
K =

    5.2722   -0.0534    2.6288
         0    4.8751    1.3524
         0         0    0.0095


R =

    0.7348   -0.6763   -0.0517
   -0.3881   -0.3567   -0.8498
    0.5563    0.6445   -0.5245


T =

   19.5326
   46.2685
  105.3472
```

Complete Example

- Let's interpret the results. We normalize K such that K(3,3) = 1:

```
>> K/K(3,3)

ans =

  555.4112   -5.6276  276.9332
         0  513.5750  142.4748
         0         0    1.0000
```

skewness

$$K = \begin{bmatrix} \alpha_x & 0 & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \alpha_x = \frac{f}{dx} \qquad \alpha_y = \frac{f}{dy}$$

- From camera data sheet, the sensor size is 4.54mm x 3.42mm.
- The image has 640 pixel x 480 pixel. → 320, 240
- Thus each pixel size is 0.07mm x 0.07mm. → dx = 0.07, dy = 0.07
- Focal length of camera is 3.7mm. → f = 3.7
- Therefore $\alpha_x = f / dx = 530$ $\alpha_y = f / dy = 530$
- Answer (555 and 513) quite close to actual values (530 and 530).
- Also, x0 = 277 pixel and y0 = 142 pixel from the pixel CS origin (somewhat off-centered).

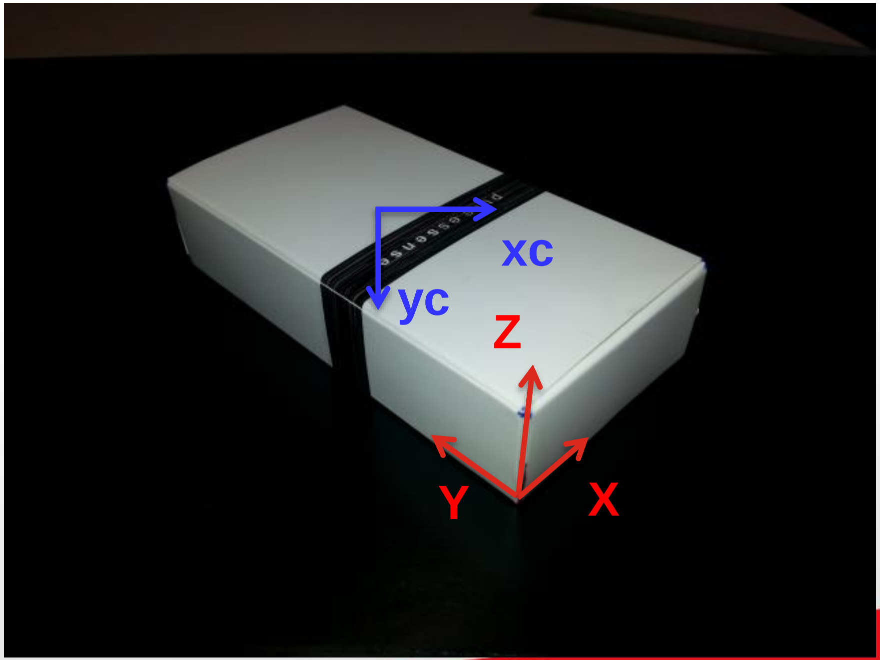
Complete Example

- The translation vector was:

```
T =

   19.5326
   46.2685
  105.3472
```

- The answer of T looks correct from the figure below. (Remember that camera CS is somewhat off-centered).



Complete Example

- The rotation matrix interpreted as Z-Y-X-Euler angles are:
  - Z: -27.8 degrees
  - Y: -33.8 degrees
  - X: 129.1 degrees
  - Which seems correct.