UNIVERSITY OF SOUTH CAROLINA

CSCE 590 INTRODUCTION TO IMAGE PROCESSING

Image Acquisition and Projective Geometry

Ioannis Rekleitis

Review: Linear Algebra

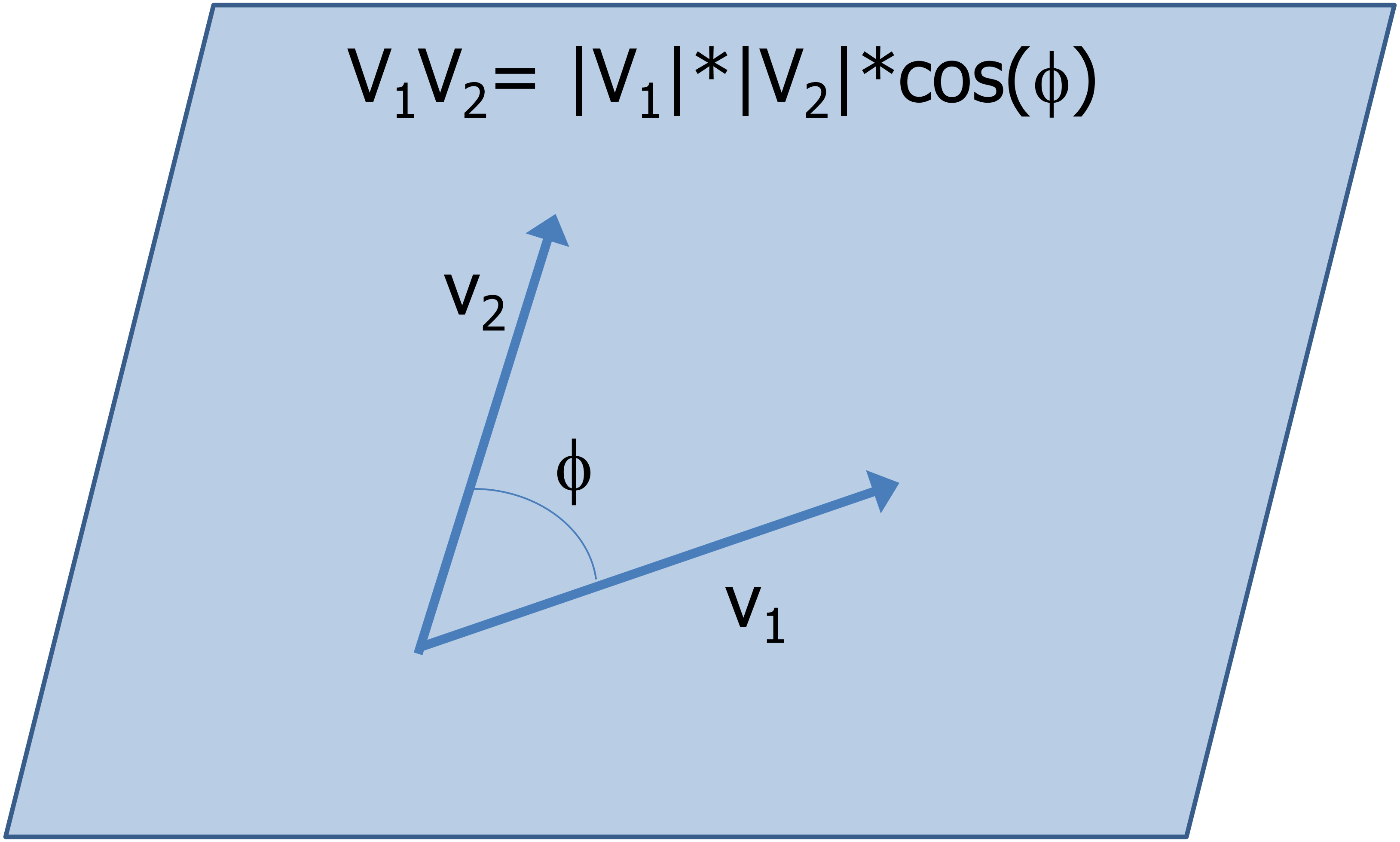
- Matrix Addition
- Matrix Multiplication **A*B**
- Matrix-Vector Multiplication **A***v
- Matrix Transpose $\mathbf{A}^T$, $(\mathbf{AB})^T = \mathbf{A}^T\mathbf{B}^T$
- Matrix Inverse $\mathbf{A}^{-1}$
- Identity Matrix $\mathbf{I}_3$=

1	**0**	**0**
0	**1**	**0**
0	**0**	**1**

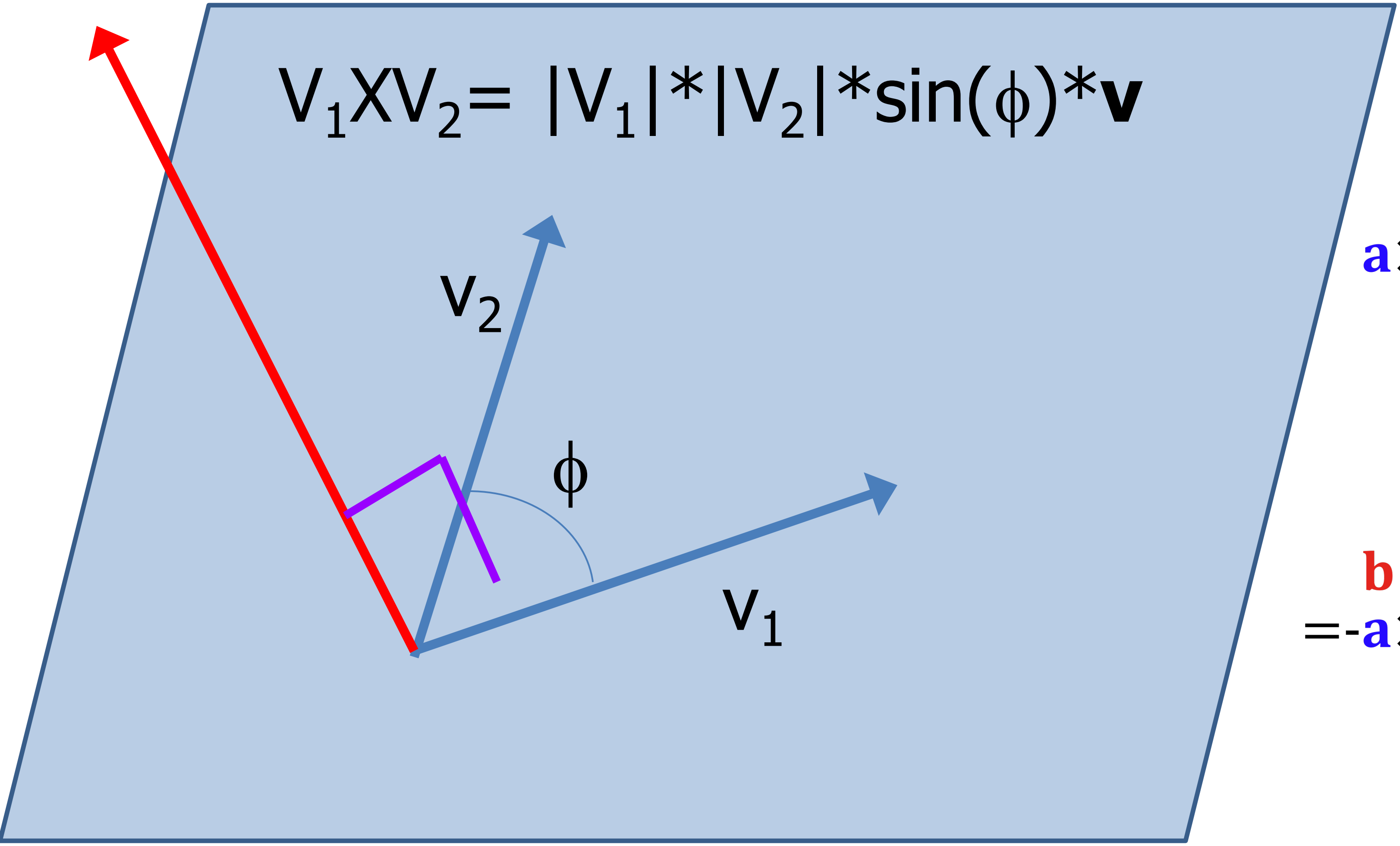
$$\begin{bmatrix}1 & 2\\ 3 & 4\\ 5 & 6\end{bmatrix}^{\mathrm{T}} = \begin{bmatrix}1 & 3 & 5\\ 2 & 4 & 6\end{bmatrix}$$

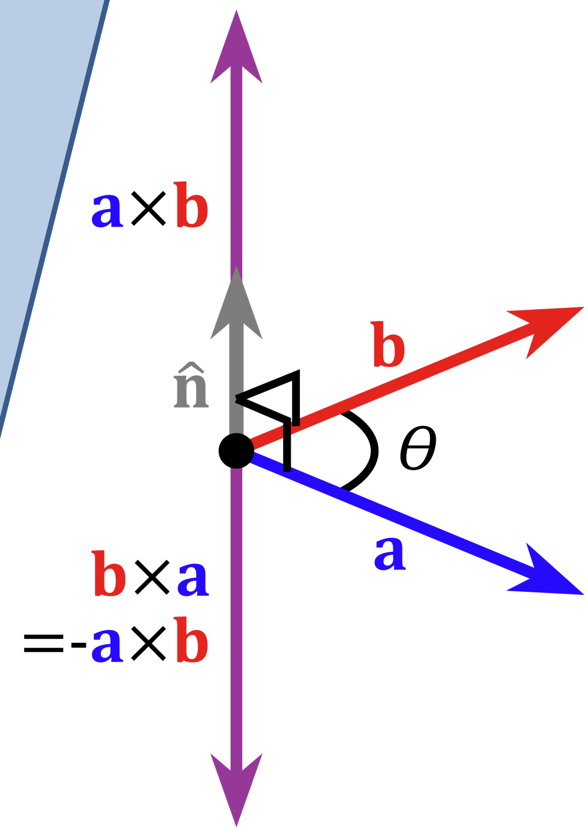
$$\mathbf{AB} = \begin{pmatrix} a & b & c \\ p & q & r \\ u & v & w \end{pmatrix} \begin{pmatrix} \alpha & \beta & \gamma \\ \lambda & \mu & \nu \\ \rho & \sigma & \tau \end{pmatrix} = \begin{pmatrix} a\alpha + b\lambda + c\rho & a\beta + b\mu + c\sigma & a\gamma + b\nu + c\tau \\ p\alpha + q\lambda + r\rho & p\beta + q\mu + r\sigma & p\gamma + q\nu + r\tau \\ u\alpha + v\lambda + w\rho & u\beta + v\mu + w\sigma & u\gamma + v\nu + w\tau \end{pmatrix},$$

Dot Product



Cross Product



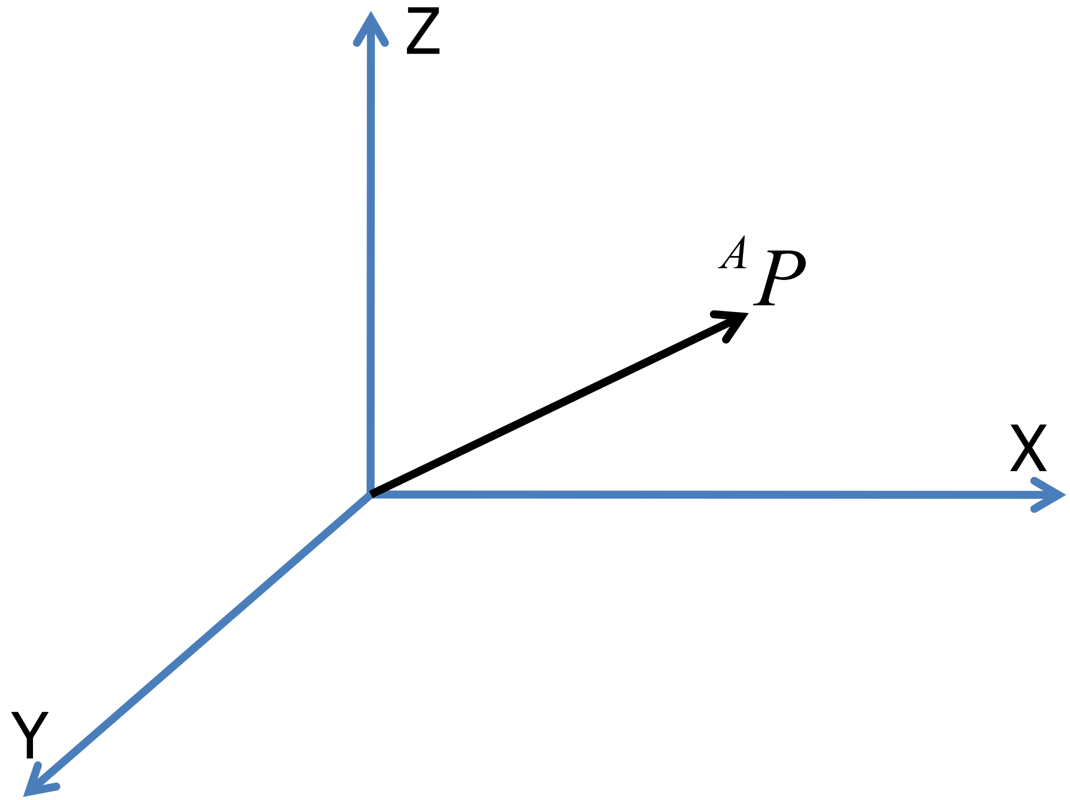


Review: Coordinate Systems and Orientation Representations

Position Representation

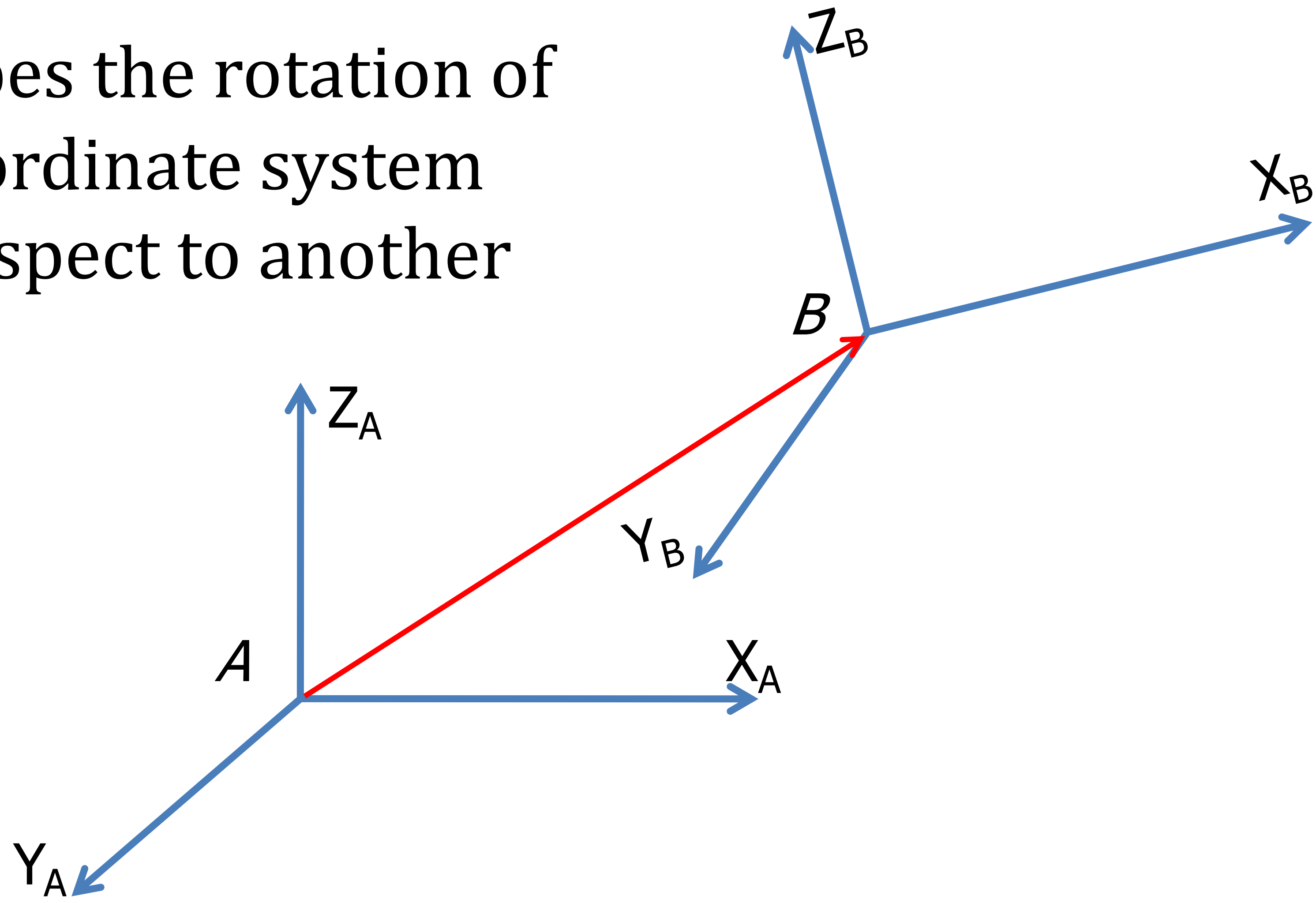
- Position representation is:

$$^{A}P = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$$



Orientation Representations

- Describes the rotation of one coordinate system with respect to another

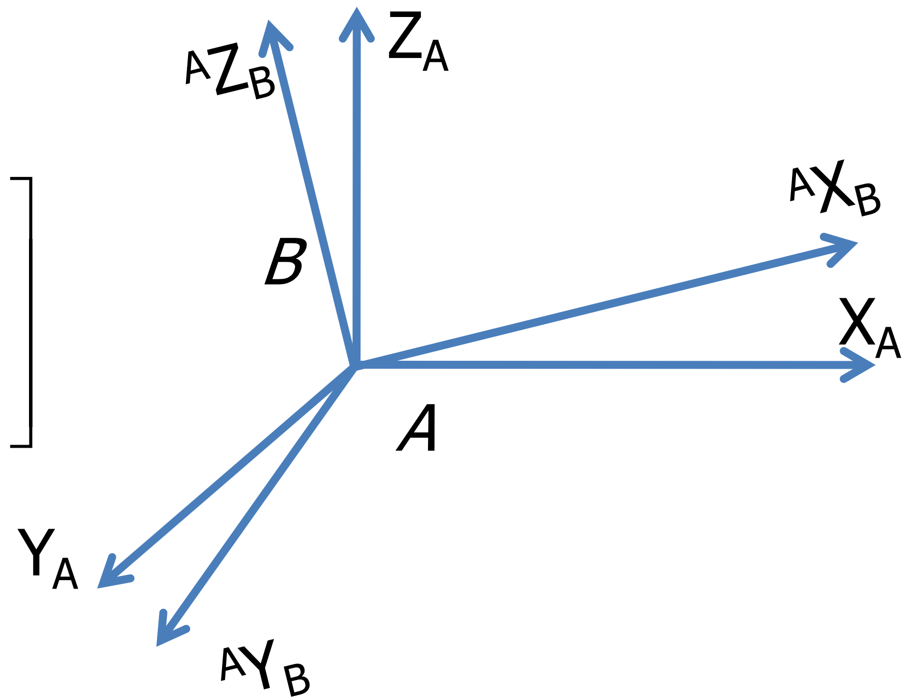


Rotation Matrix

- Write the unit vectors of B in the coordinate system of A.
- Rotation Matrix:

$$
{}^{A}_{B}R = \begin{bmatrix} {}^{A}\hat{X}_B & {}^{A}\hat{Y}_B & {}^{A}\hat{Z}_B \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}
$$

$$
= \begin{bmatrix} \hat{X}_B \cdot \hat{X}_A & \hat{Y}_B \cdot \hat{X}_A & \hat{Z}_B \cdot \hat{X}_A \\ \hat{X}_B \cdot \hat{Y}_A & \hat{Y}_B \cdot \hat{Y}_A & \hat{Z}_B \cdot \hat{Y}_A \\ \hat{X}_B \cdot \hat{Z}_A & \hat{Y}_B \cdot \hat{Z}_A & \hat{Z}_B \cdot \hat{Z}_A \end{bmatrix}
$$



Properties of Rotation Matrix

$$ {}_A^B R = {}_B^A R^T $$

$$ {}_B^A R^T \, {}_B^A R = I_3 $$

$$ {}_B^A R = {}_A^B R^{-1} = {}_A^B R^T $$

Coordinate System Transformation

$$M = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} R & T \\ 0_{3\times1} & 1 \end{bmatrix}$$

where R is the rotation matrix and T is the translation vector

Hierarchy of Transformations 2D

Transformation	Matrix	# DoF	Preserves
Translation	$[\mathbf{I}\|\mathbf{t}]_{2x3}$	2	orientation
Rigid	$[\mathbf{R}\|\mathbf{t}]_{2x3}$	3	lengths
Similarity	$[s\mathbf{R}\|\mathbf{t}]_{2x3}$	4	angles
Affine	$[\mathbf{A}]_{2x3}$	6	parallelism
Projective	$[\mathbf{H}]_{3x3}$	8	Straight lines

The above transformations apply to a vector $x=[x,y,1]^T$

Hierarchy of Transformations 2D

Transformation	**Matrix**	**T*x**
Translation	$[\mathbf{I}\|\mathbf{t}]_{2x3}$	$x=x+t_x$ $y=y+t_y$
Rigid	$[\mathbf{R}\|\mathbf{t}]_{2x3}$	$x=\cos(\phi)x-\sin(\phi)y+t_x$ $y=\sin(\phi)x+\cos(\phi)y+t_y$
Similarity	$[s\mathbf{R}\|\mathbf{t}]_{2x3}$	$x=ax-by+t_x$ $y=bx+ay+t_y$
Affine	$[\mathbf{A}]_{2x3}$	**A** is arbitrary
Projective	$[\mathbf{H}]_{3x3}$	**H** is 3 by 3

3D Transformations

Transformation	**Matrix**	**# DoF**	**Preserves**
Translation	$[\mathbf{I}\vert\mathbf{t}]_{3x4}$	3	orientation
Rigid	$[\mathbf{R}\vert\mathbf{t}]_{3x4}$	6	lengths
Similarity	$[s\mathbf{R}\vert\mathbf{t}]_{3x4}$	7	angles
Affine	$[\mathbf{A}]_{3x4}$	12	parallelism
Projective	$[\mathbf{H}]_{4x4}$	15	Straight lines

The above transformations apply to a vector $x=[x,y,z,1]^T$

Rotation Matrix

- The rotation matrix consists of 9 variables, but there are many constraints. The minimum number of variables needed to describe a rotation is three.

Rotation Matrix-Single Axis

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix}$$

$$R_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix}$$

$$R_z(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Fixed Angles

- One simple method is to perform three rotations about the axis of the original coordinate frame:
  - X-Y-Z fixed angles

$$
{}^{A}_{B}R(\theta,\phi,\psi) = R_z(\psi)R_y(\phi)R_x(\theta)
$$

$$
= \begin{bmatrix} \cos(\psi)\cos(\phi) & \cos(\psi)\sin(\phi)\sin(\theta) - \sin(\psi)\cos(\theta) & \cos(\psi)\sin(\phi)\cos(\theta) + \sin(\psi)\sin(\theta) \\ \sin(\psi)\cos(\phi) & \sin(\psi)\sin(\phi)\sin(\theta) + \cos(\psi)\cos(\theta) & \sin(\psi)\sin(\phi)\cos(\theta) + \cos(\psi)\sin(\theta) \\ -\sin(\phi) & \cos(\phi)\sin(\theta) & \cos(\theta)\cos(\psi) \end{bmatrix}
$$

- There are 12 different combinations

Inverse Problem

- From a Rotation matrix find the fixed angle rotations:

$$ {}^{A}_{B}R(\theta,\phi,\psi) = {}^{A}_{B}R \Rightarrow $$

$$\begin{bmatrix} \cos(\psi)\cos(\phi) & \cos(\psi)\sin(\phi)\sin(\theta)-\sin(\psi)\cos(\theta) & \cos(\psi)\sin(\phi)\cos(\theta)+\sin(\psi)\sin(\theta) \\ \sin(\psi)\cos(\phi) & \sin(\psi)\sin(\phi)\sin(\theta)+\cos(\psi)\cos(\theta) & \sin(\psi)\sin(\phi)\cos(\theta)+\cos(\psi)\sin(\theta) \\ -\sin(\phi) & \cos(\phi)\sin(\theta) & \cos(\theta)\cos(\psi) \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

thus :

$$\phi = A\tan 2\left(-r_{31}, \sqrt{\left(r_{11}^2 + r_{21}^2\right)}\right)$$

$$\psi = A\tan 2\left({r_{21}}/{\cos(\phi)}, {r_{11}}/{\cos(\phi)} \right)$$

$$\theta = A\tan 2\left({r_{32}}/{\cos(\phi)}, {r_{33}}/{\cos(\phi)} \right)$$

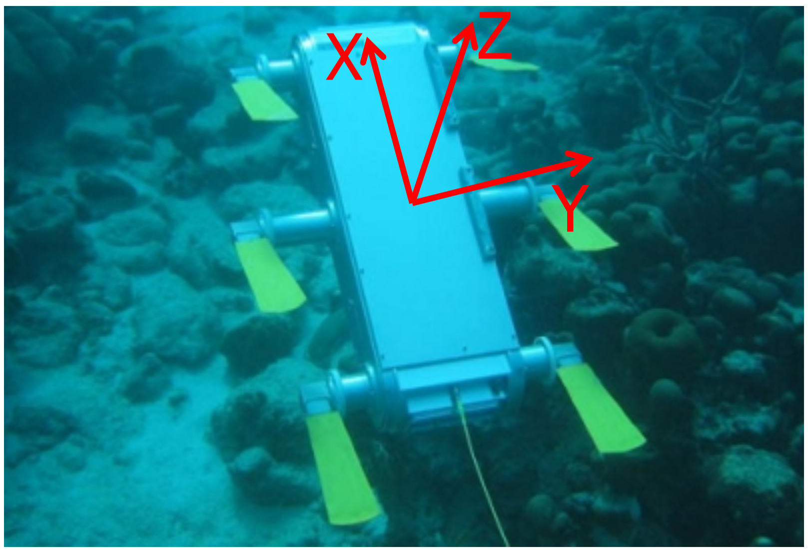
Euler Angles

- **ZYX**: Starting with the two frames aligned, first rotate about the Z_B axis, then by the Y_B axis and then by the X_B axis. The results are the same as with using XYZ fixed angle rotation.

- There are 12 different combination of Euler Angle representations

Euler Angles

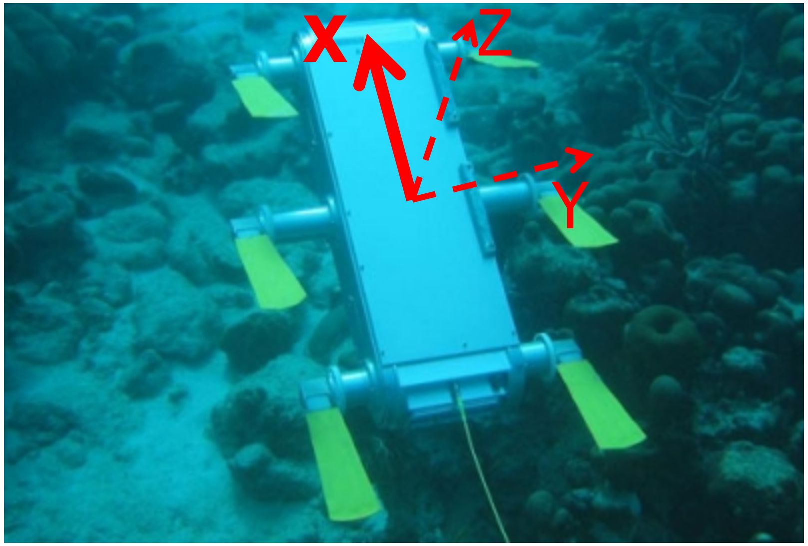
- Traditionally the three angles along the axis are called Roll, Pitch, and Yaw



Euler Angles

- Traditionally the three angles along the axis are called Roll, Pitch, and Yaw

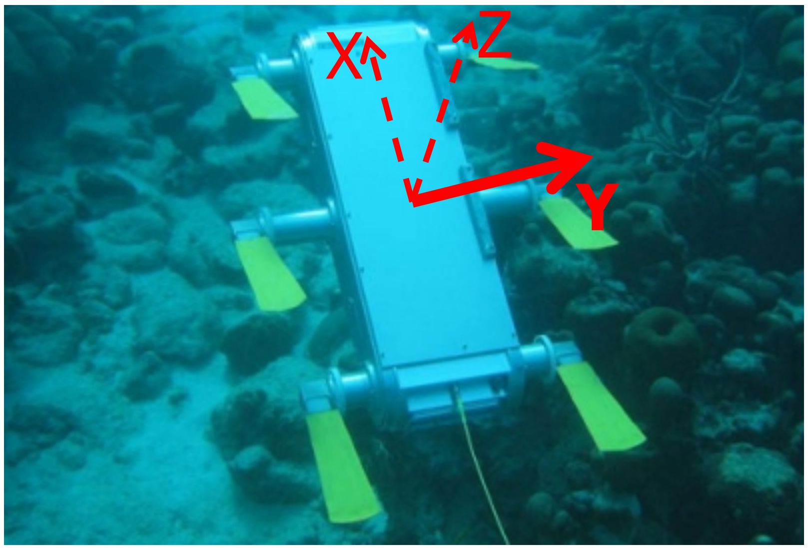
Roll



Euler Angles

- Traditionally the three angles along the axis are called Roll, Pitch, and Yaw

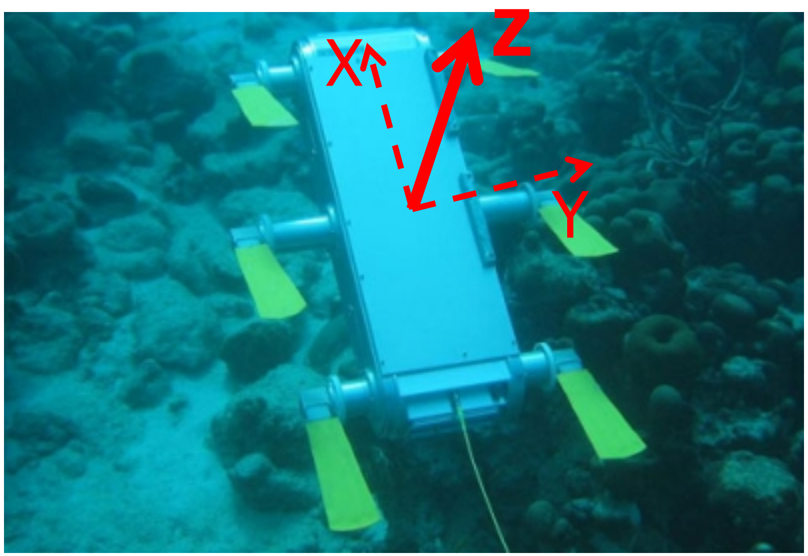
Pitch



Euler Angles

- Traditionally the three angles along the axis are called Roll, Pitch, and Yaw

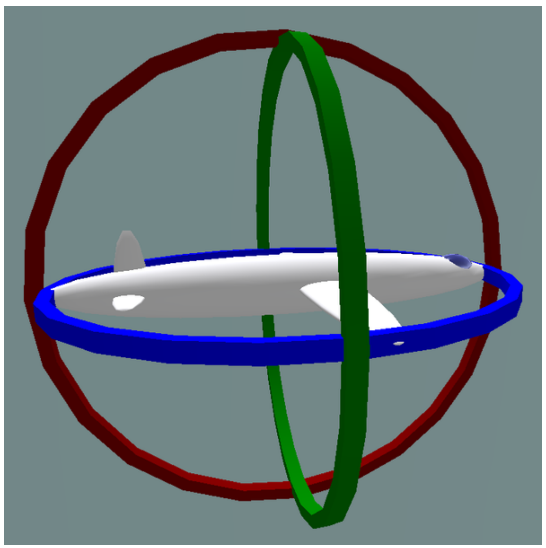
Yaw

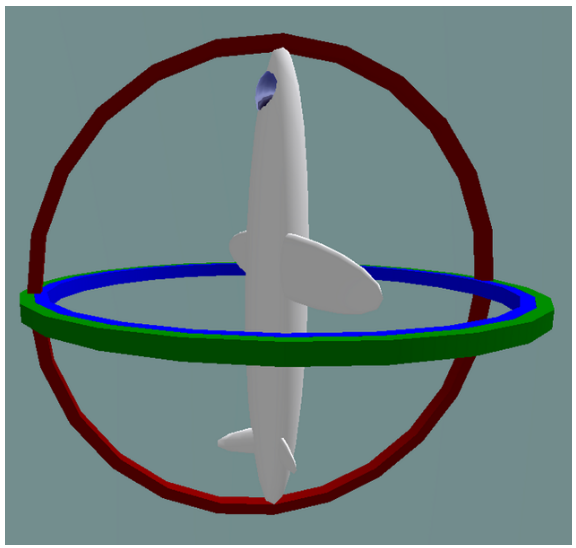


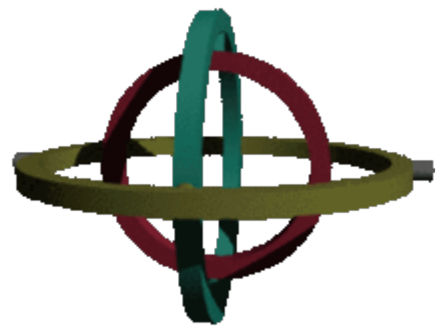
Euler Angle concerns: Gimbal Lock

Using the **ZYZ** convention

- (90°, 45°, −105°) ≡ (−270°, −315°, 255°) multiples of 360°
- (72°, 0°, 0°) ≡ (40°, 0°, 32°) singular alignment (Gimbal lock)
- (45°, 60°, −30°) ≡ (−135°, −60°, 150°) bistable flip

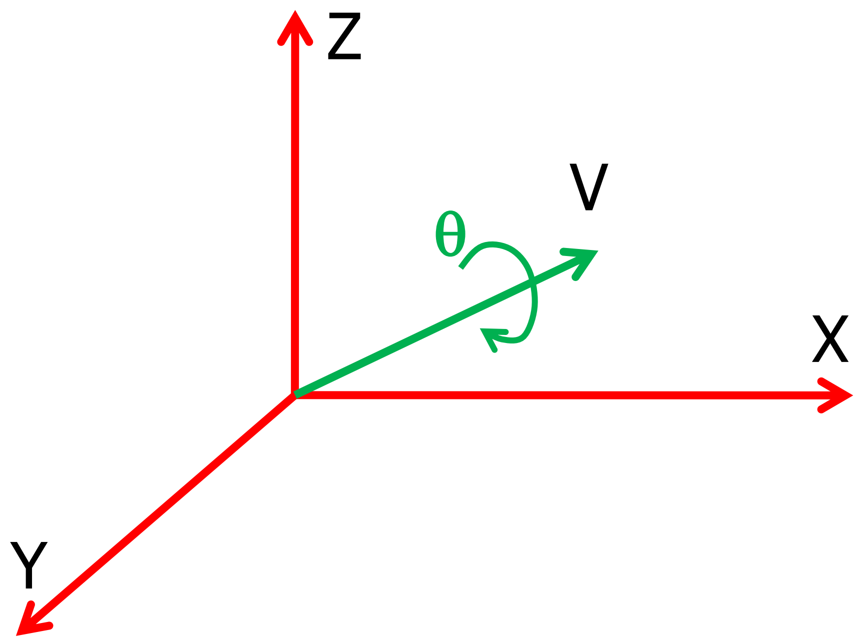






Axis-Angle Representation

- Represent an arbitrary rotation as a combination of a vector and an angle



Quaternions

- Are similar to axis-angle representation
- Two formulations
  - Classical
  - Based on JPL's standards

    W. G. Breckenridge, "Quaternions - Proposed Standard Conventions," JPL, Tech. Rep. INTEROFFICE MEMORANDUM IOM 343-79-1199, 1999.
- Avoids Gimbal lock

- See also: M. D. Shuster, "A survey of attitude representations," Journal of the Astronautical Sciences, vol. 41, no. 4, pp. 439–517, Oct.–Dec. 1993.

Quaternions

	Classic notation	JPL-based
	$\bar{q} = q_4 + q_1 i + q_2 j + q_3 k$	$\bar{q} = q_4 + q_1 i + q_2 j + q_3 k$
	$i^2 = j^2 = k^2 = ijk = -1$ $ij = -ji = k,\ jk = -kj = i,\ ki = -ik = j$	$i^2 = j^2 = k^2 = -1$ $-ij = ji = k, -jk = kj = i, -ki = ik = j$
Vector Notation	$\bar{q} = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix}, q_0 = \cos\left(\theta/2\right), \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} \sin\left(\theta/2\right)\cos(\beta_x) \\ \sin\left(\theta/2\right)\cos(\beta_y) \\ \sin\left(\theta/2\right)\cos(\beta_z) \end{bmatrix}$	$\bar{q} = \begin{bmatrix} \mathbf{q} \\ q_4 \end{bmatrix}, \mathbf{q} = \begin{bmatrix} k_x \sin\left(\theta/2\right) \\ k_y \sin\left(\theta/2\right) \\ k_z \sin\left(\theta/2\right) \end{bmatrix}, q_4 = \cos\left(\theta/2\right)$
		$\lVert\bar{q}\rVert = 1, \bar{q} \otimes \bar{p}, \mathbf{q} \times \mathbf{p}, \bar{q}_I, \lfloor \mathbf{q} \times \rfloor$

See also: N. Trawny and S. I. Roumeliotis, “Indirect Kalman Filter for 3D Attitude Estimation,” University of Minnesota, Dept. of Comp. Sci. & Eng., Tech. Rep. 2005-002, March 2005.

Coordinate frames on PR2

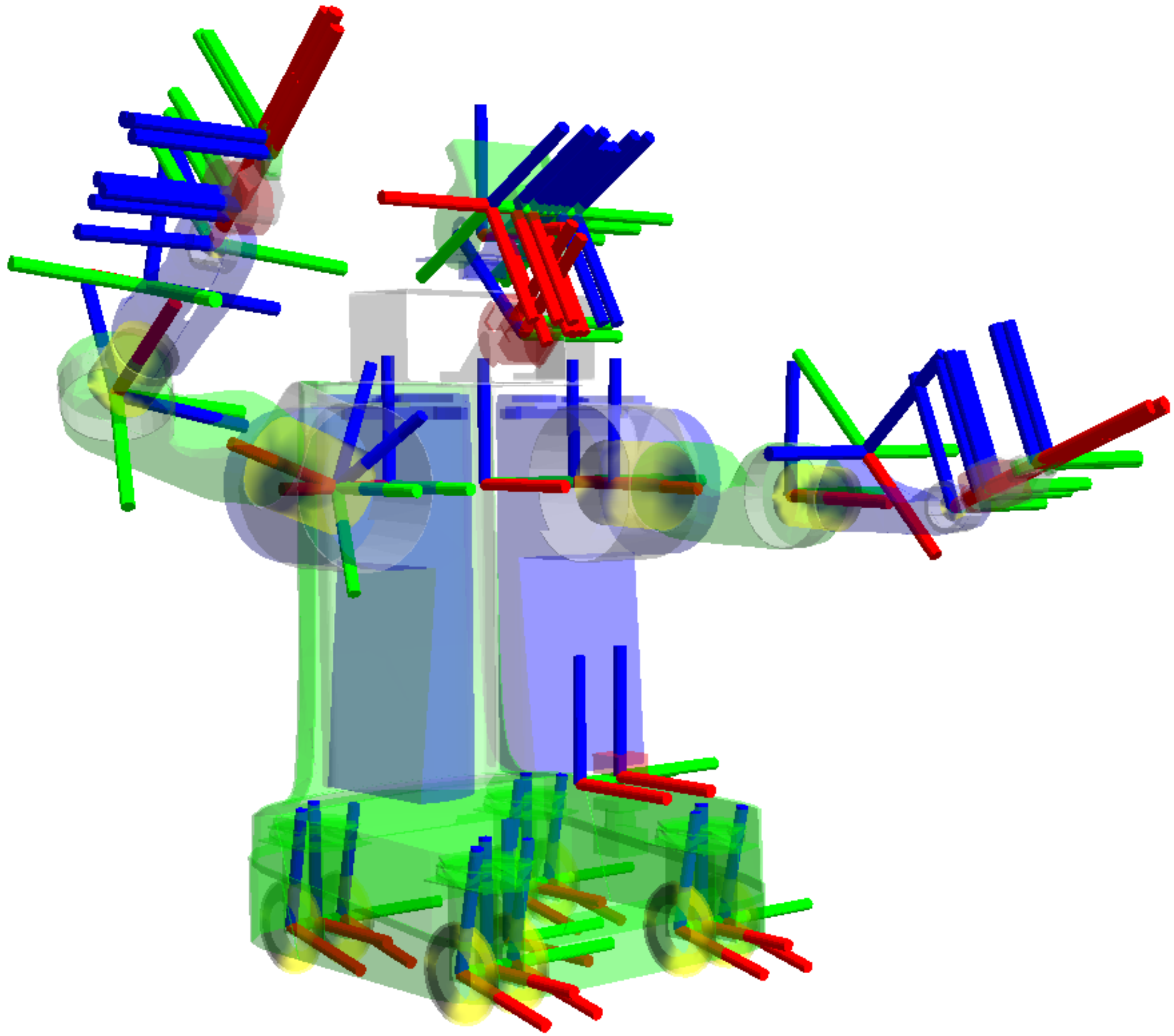


Image Formation in the Eye

Image is upside down in the retina/imaging plane!

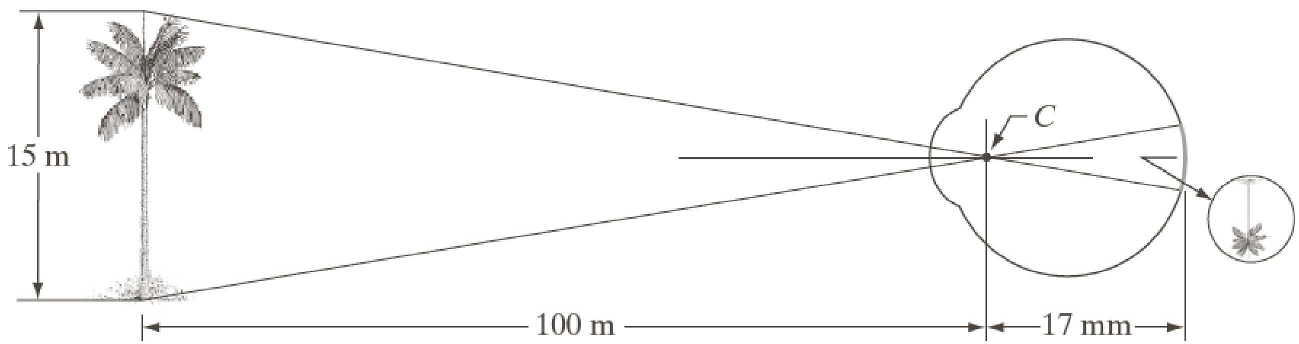


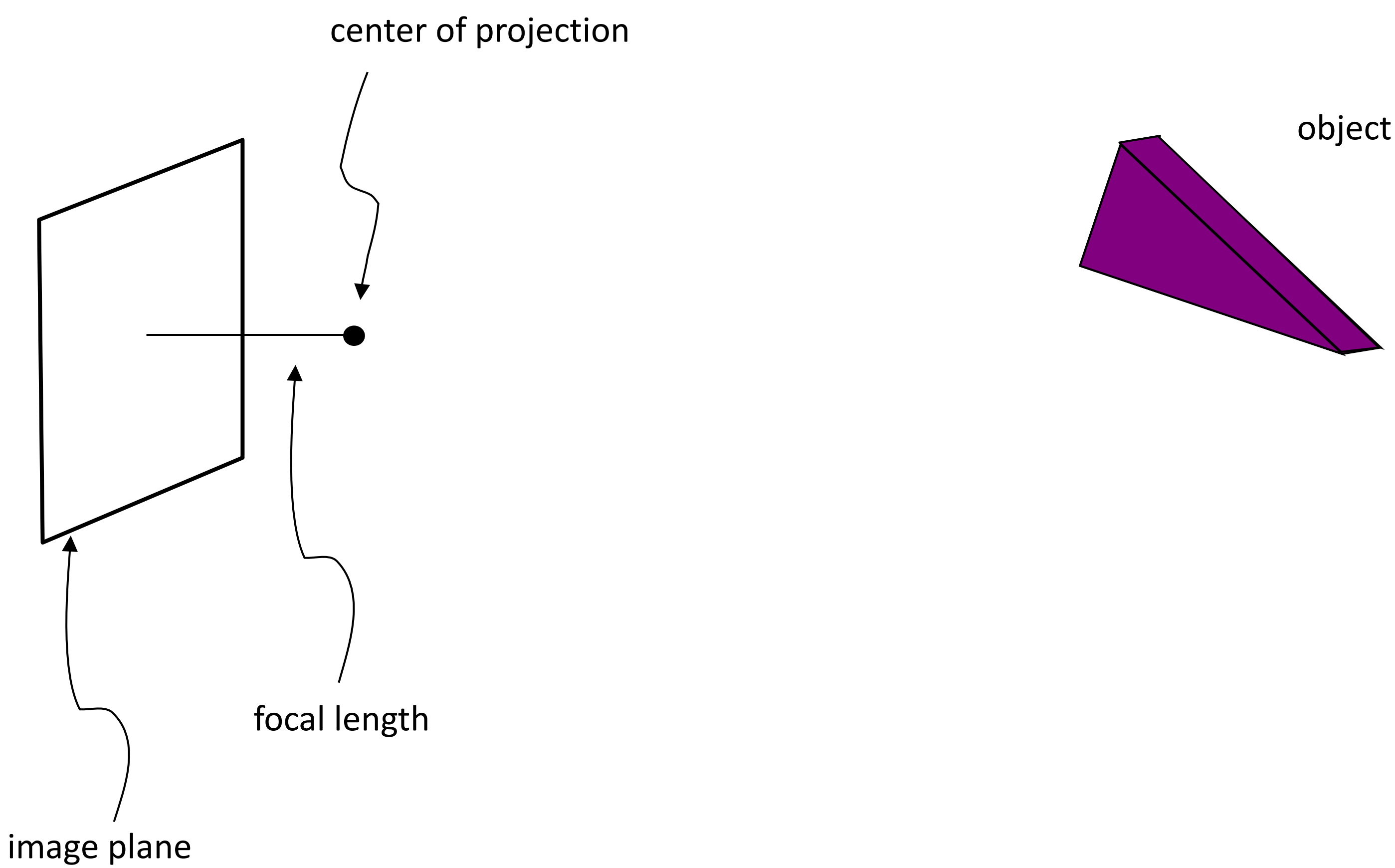
FIGURE 2.3 Graphical representation of the eye looking at a palm tree. Point *C* is the optical center of the lens.

Adjust focus length

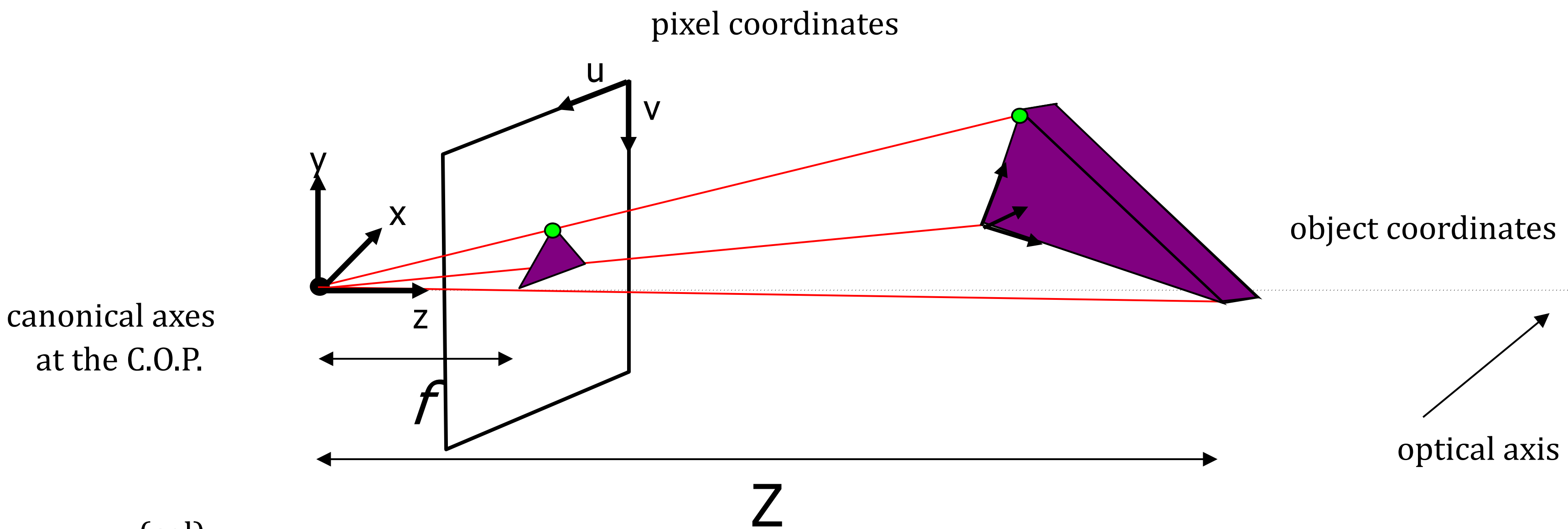
- Camera
- Human eye

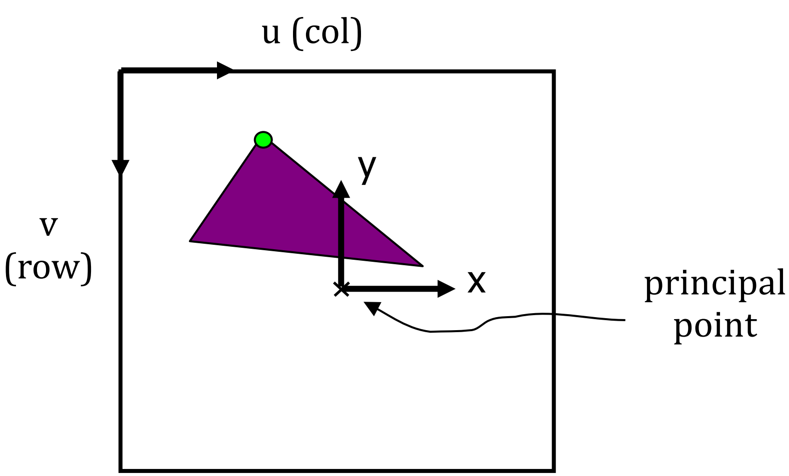
Camera Geometry

3D→2D transformation: perspective projection



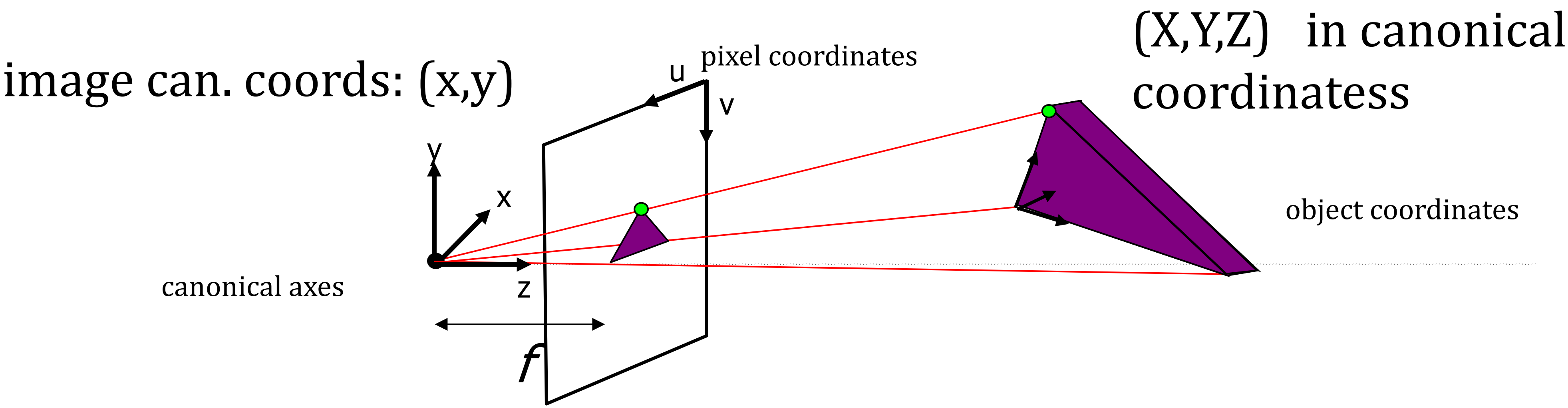
Coordinate Systems



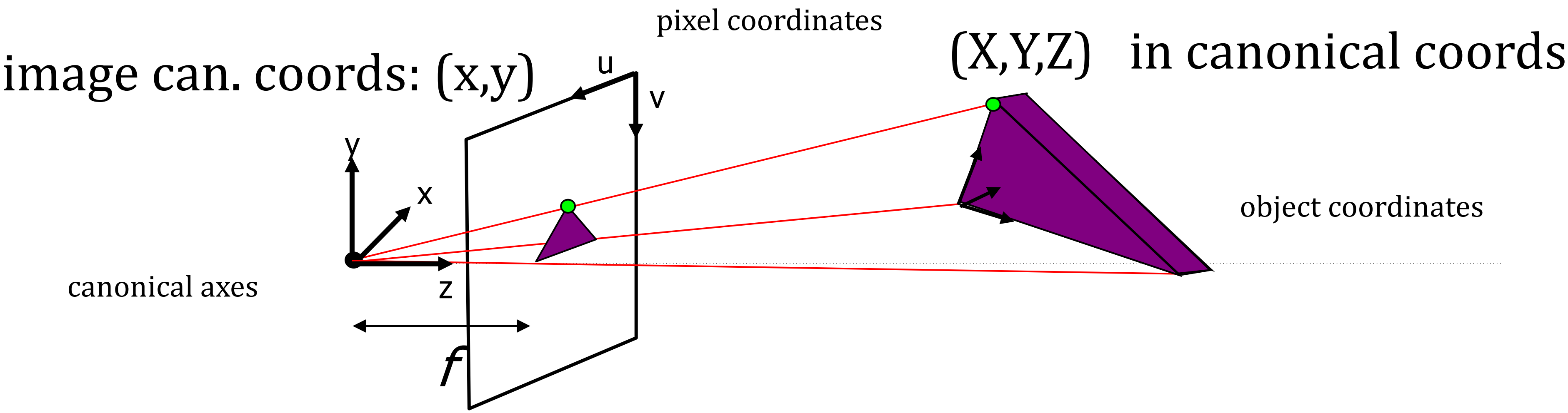


Add coordinate systems in order to describe feature points...

Coordinate Systems



From 3d to 2d

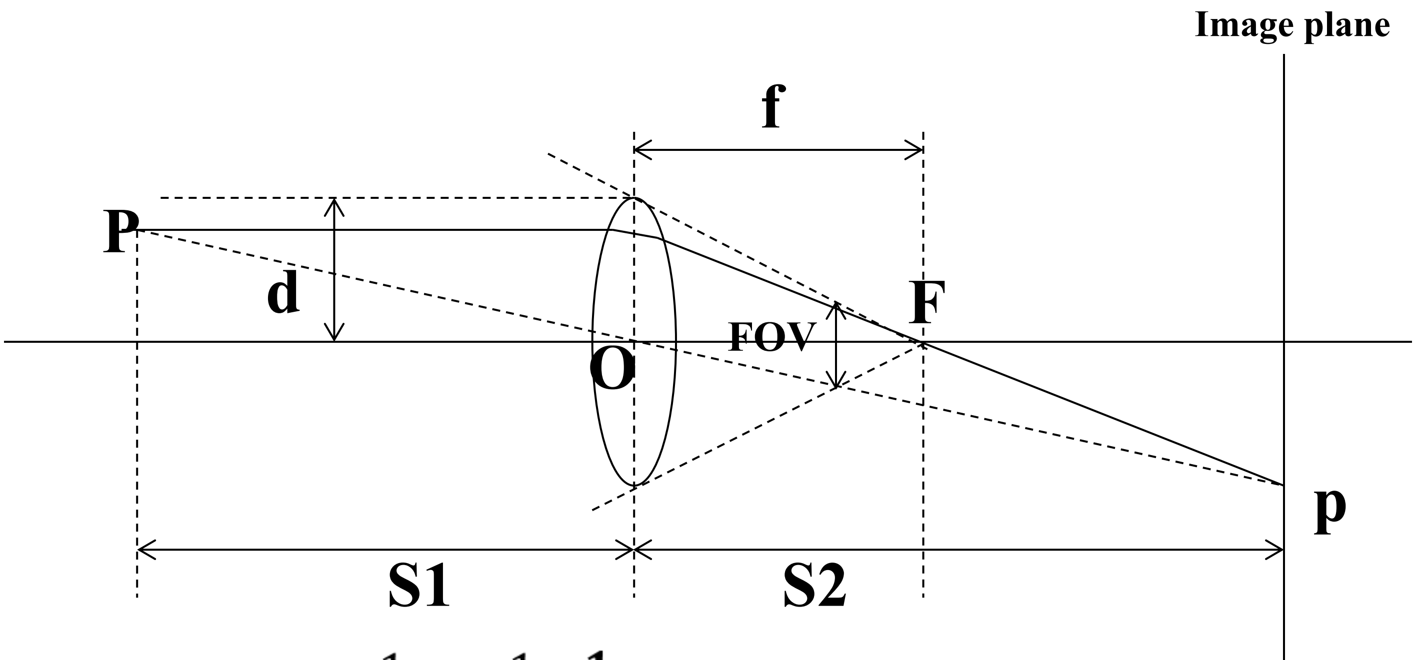


$$x = \frac{fX}{Z} \qquad y = \frac{fY}{Z}$$

a <u>nonlinear</u> transformation

goal: to recover information about (X,Y,Z) from (x,y)

Lens Parameters



Thin lens theory: $\frac{1}{S1} + \frac{1}{S2} = \frac{1}{f}$

Field of View: $\omega = 2arctan\frac{d}{f}$

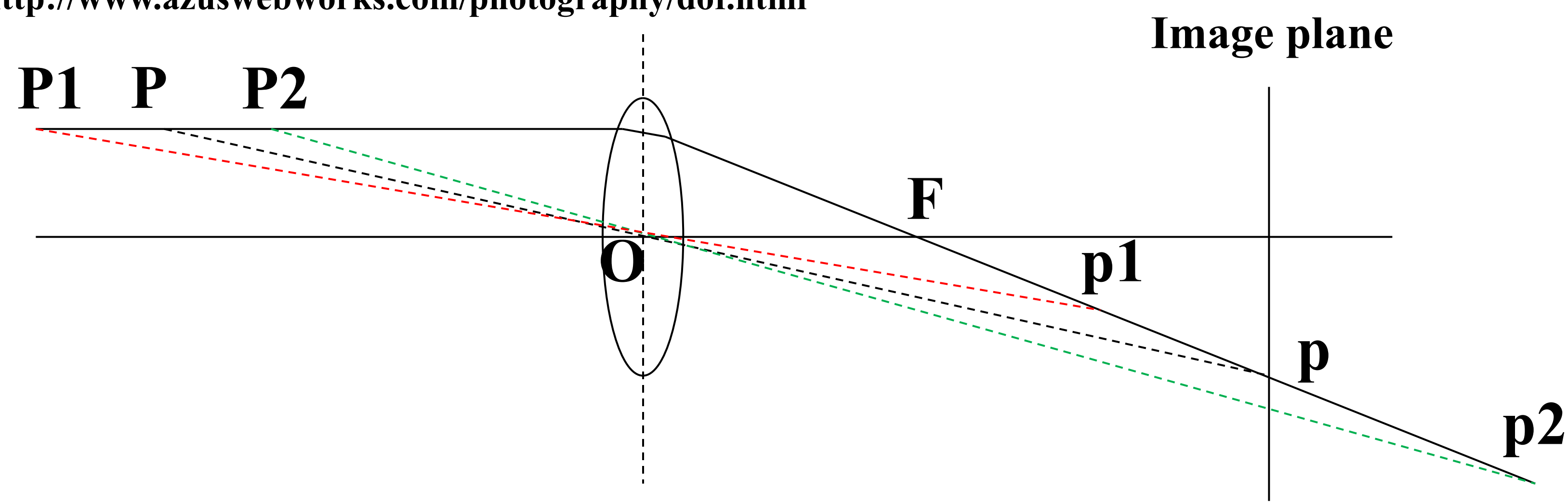
FOV

- **Increasing the distance from the object to the lens will reduce the size of image**
- **Large focus length will give a small FOV**

Depth of Field & Out of Focus

http://www.azuswebworks.com/photography/dof.html

- DOF is inversely proportional to the focus length
- DOF is proportional to S1



Camera Calibration

- Camera Model
  - [*u v 1*] Pixel coords
  - $[x_w \quad y_w \quad z_w \quad 1]^T$ World coords

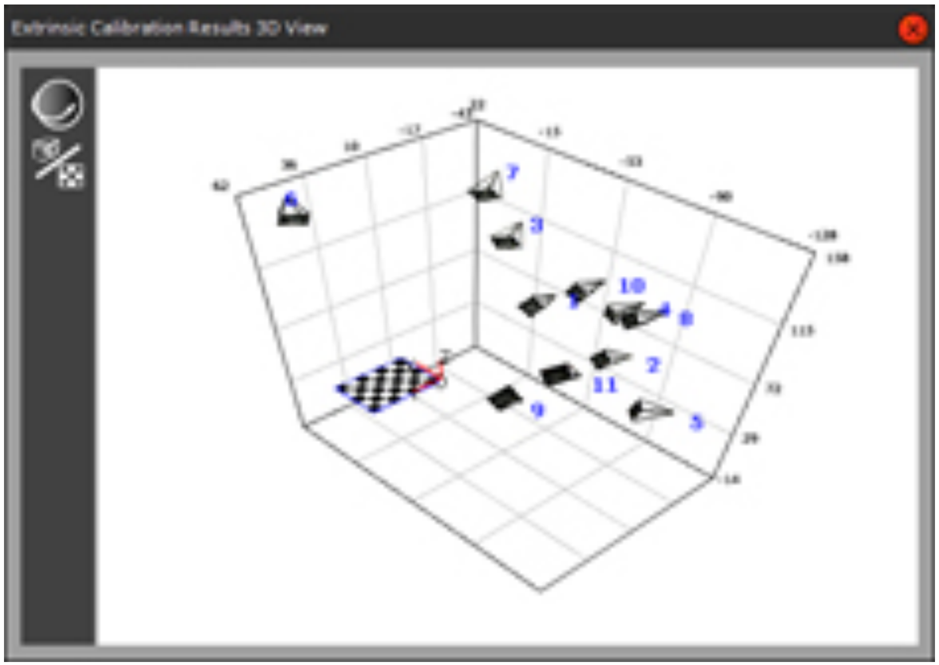
$$z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = A[R \quad T] \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

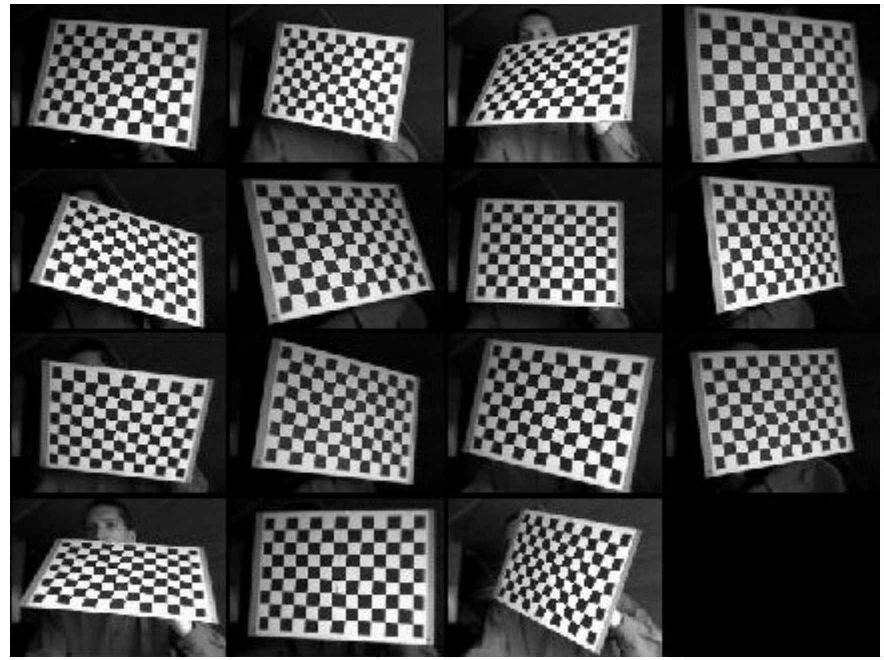
- Intrinsic Parameters
  - $\alpha_x = f \cdot m_x, \alpha_y = f \cdot m_y$ focal lengths in pixels
  - γ skew coefficient
  - u_0, v_o focal point

$$A = \begin{bmatrix} \alpha_x & \gamma & u_0 \\ 0 & \alpha_y & v_o \\ 0 & 0 & 1 \end{bmatrix}$$

- Extrinsic Parameters
  - $[R \quad T]$ Rotation and Translation

Camera Calibration





Existing packages in MATLAB, OpenCV, etc

Rectified Image Sample

Unrectified

From Clearpath Husky Axis M1013 camera

Rectified

Rectified Image Sample

Unrectified

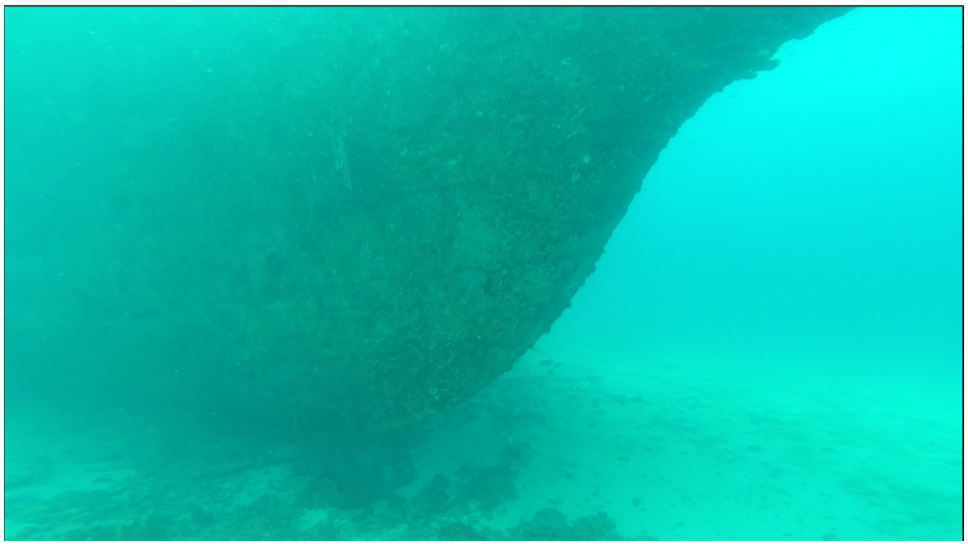
Rectified

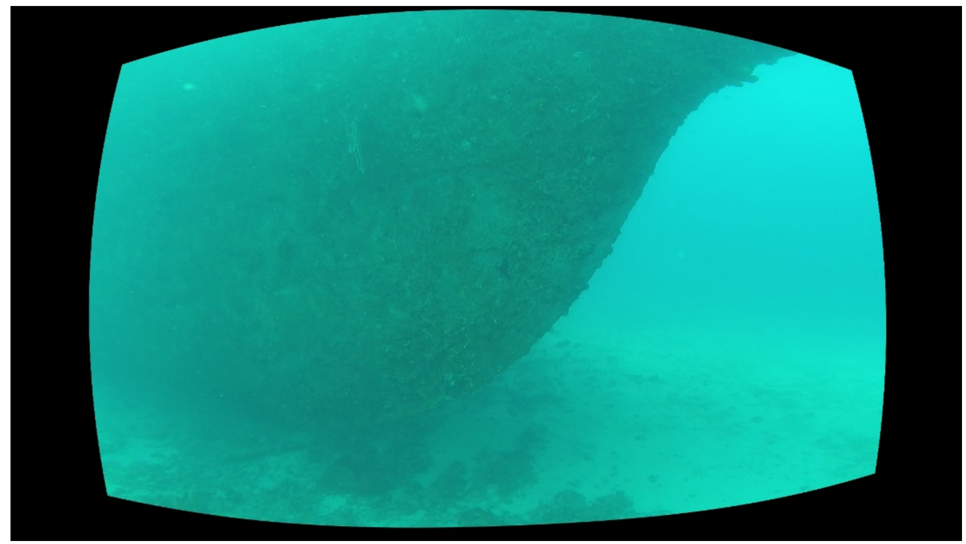
From Parrot ARDrone 2.0 front camera

Rectified Image Sample

Unrectified

Rectified



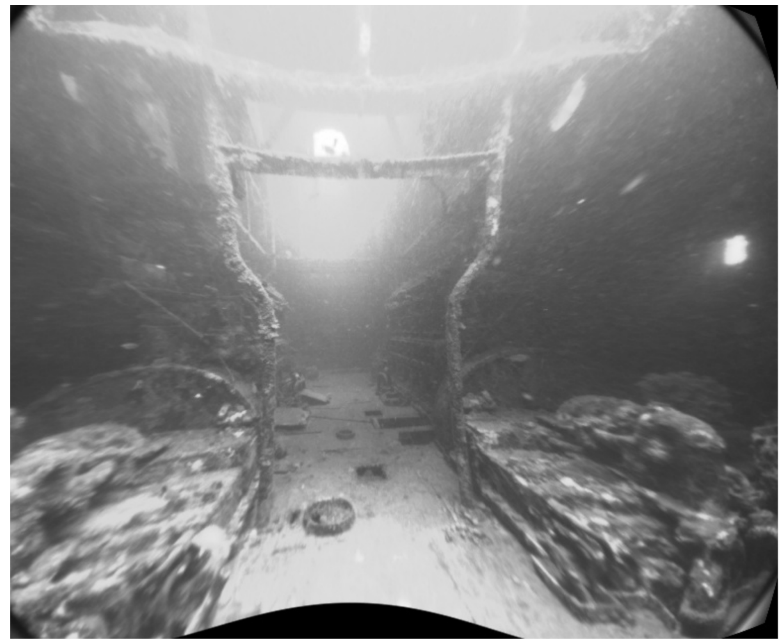


From GoPro HERO3+ at Barbados 2015 Field Trials

ReRectified Image Sample

Rectified

ReRectified



From Aqua front camera at Barbados 2013 Field Trials