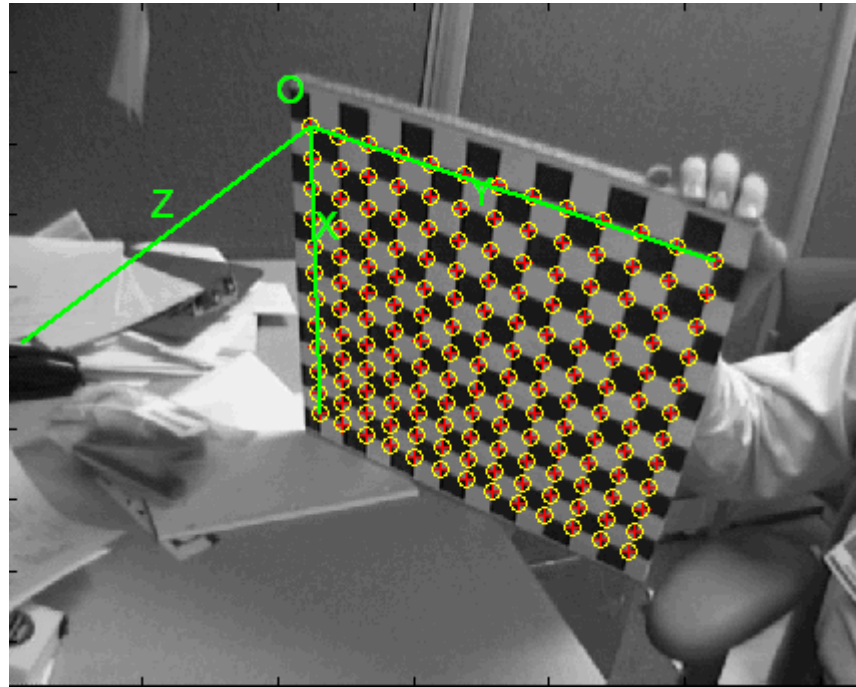


# Image Transformations & Camera Calibration



CSE398/498  
04 Oct 04

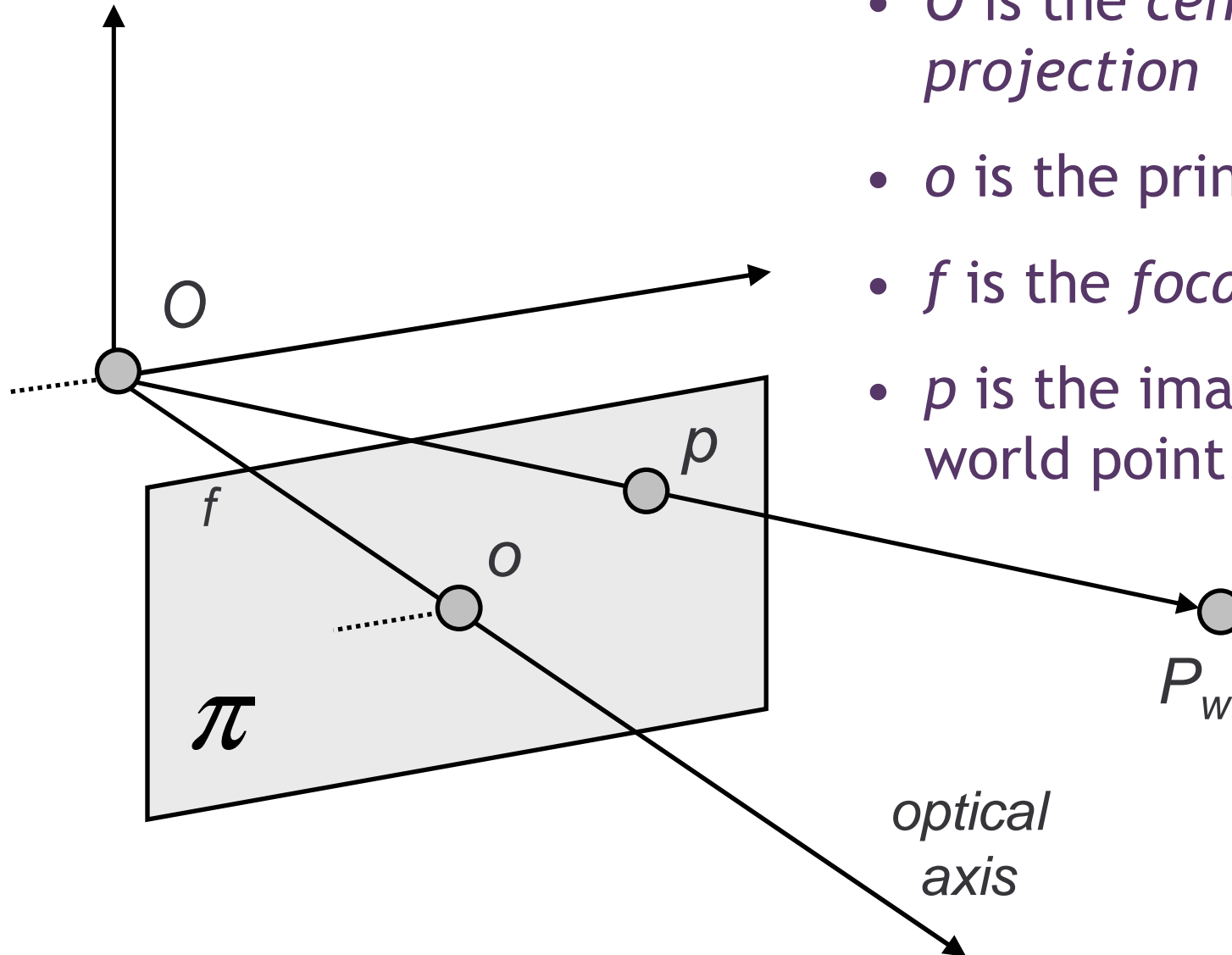
# Class Objectives

- Review transformations from the world frame to the image plane
- Demonstrate how unknown camera parameters can be recovered through camera calibration

# References

- Trucco & Verri Ch. 2 & Ch. 6
- Caltech Matlab Calibration Toolbox  
[http://www.vision.caltech.edu/bouguetj/calib\\_doc/](http://www.vision.caltech.edu/bouguetj/calib_doc/)

# The Perspective Camera Model



- $O$  is the *center of projection*
- $o$  is the *principal point*
- $f$  is the *focal length*
- $p$  is the *image of the world point  $P_w$*

# Perspective Camera

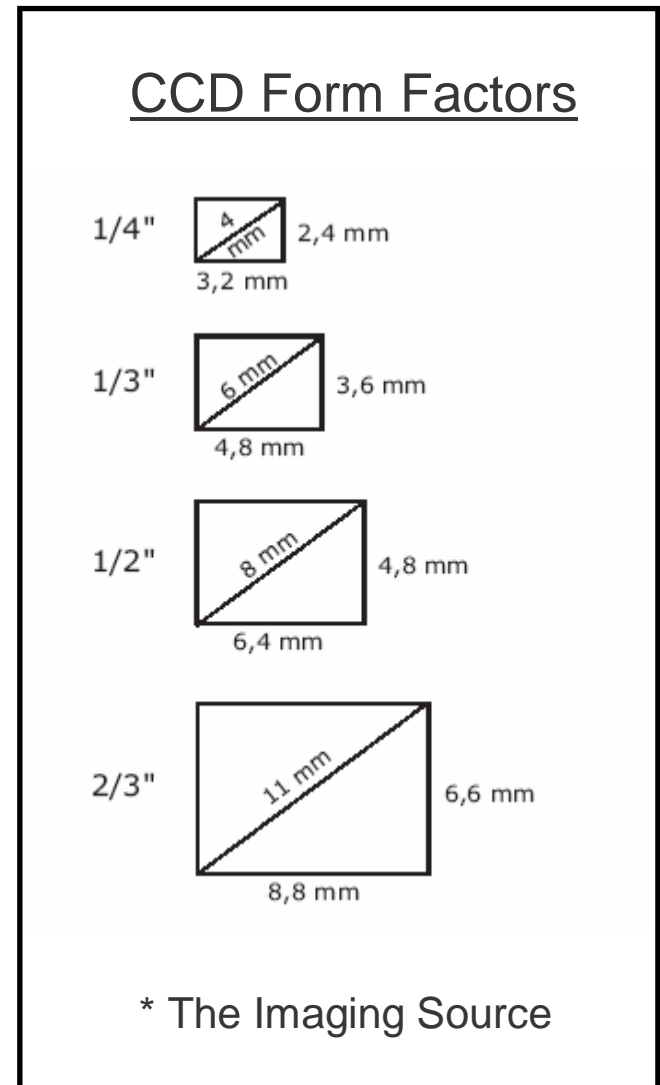
- The 3D reference frame with origin  $O$  is referred to as the camera frame
- We can then relate points in space with points on the image frame by the following fundamental equations of the perspective camera model

$$x = f \frac{x_w}{z_w}$$

$$y = f \frac{y_w}{z_w}$$

# In Class Exercise ☹️

- The focal length of the iSight camera is nominally 2.8 mm.
- It uses a 1/4" form factor CCD
- What are the horizontal and vertical fields of view (FOV) for the camera?
- You are tracking a tennis ball with a diameter of 64mm, and it's radius is 64 pixels in the image, what is its distance from the camera frame?



# Homogeneous Coordinates

- *Homogeneous Coordinates* provide a convenient means for representing and composing multiple rigid transformations
- The dimension of each coordinate is increased by 1:

$$\begin{bmatrix} x \\ y \end{bmatrix} \Rightarrow \begin{bmatrix} x \\ y \\ \lambda \end{bmatrix} \quad \begin{bmatrix} x \\ y \\ z \end{bmatrix} \Rightarrow \begin{bmatrix} x \\ y \\ z \\ \lambda \end{bmatrix}$$

# Homogeneous Coordinates

- *Homogeneous Coordinates* provide a convenient means for representing and composing multiple rigid transformations
- The dimension of each coordinate is increased by 1:

$$\begin{bmatrix} x \\ y \end{bmatrix} \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad \begin{bmatrix} x \\ y \\ z \end{bmatrix} \Rightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

- Last coordinate normally “normalized” to 1

# Perspective Camera

- Recall that for the perspective model, we have

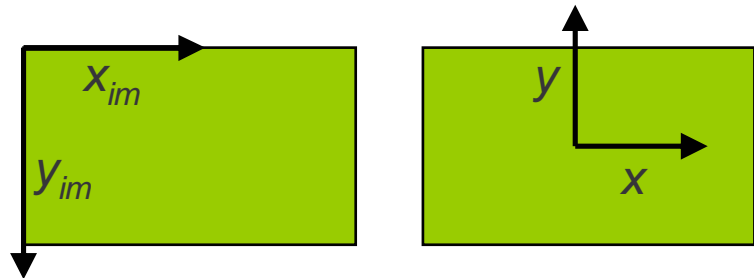
$$x = f \frac{x_w}{z_w}$$

$$y = f \frac{y_w}{z_w}$$

and to convert from pixels to camera coords

$$x = (x_{im} - o_x) s_x$$

$$y = -(y_{im} - o_y) s_y$$



and from the world frame to the camera frame

$$P_c = R(P_w - t)$$

# Perspective Camera (cont'd)

- Putting it all together, we have

$$\begin{aligned}(x_{im} - o_x)s_x &= f \frac{R_1^T (P_W - t)}{R_3^T (P_W - t)} \\ -(y_{im} - o_y)s_y &= f \frac{R_2^T (P_W - t)}{R_3^T (P_W - t)}\end{aligned}$$

where  $R_i$  is the vector formed from the  $i^{th}$  row of the rotation matrix  $R$

# Perspective Camera (cont'd)

- Neglecting radial distortion, we can rewrite this as a matrix product where

$$M_{\text{int}} = \begin{bmatrix} f/s_x & 0 & o_x \\ 0 & -f/s_y & o_y \\ 0 & 0 & 1 \end{bmatrix}$$

$$M_{\text{ext}} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & -R_1^T t \\ r_{21} & r_{22} & r_{23} & -R_2^T t \\ r_{31} & r_{32} & r_{33} & -R_3^T t \end{bmatrix}$$

and where now

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = M_{\text{int}} M_{\text{ext}} \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix}$$

# Perspective Camera (cont'd)

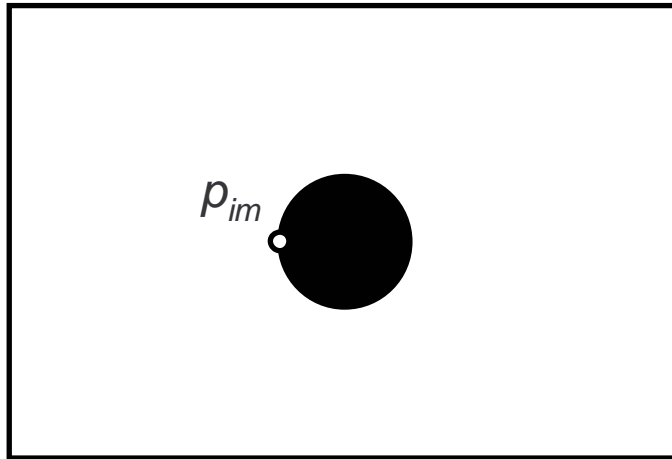
- Note that the ratios  $x_1/x_3$  and  $x_2/x_3$  are  $(x_{im}, y_{im})$

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = M_{int} M_{ext} \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix}$$

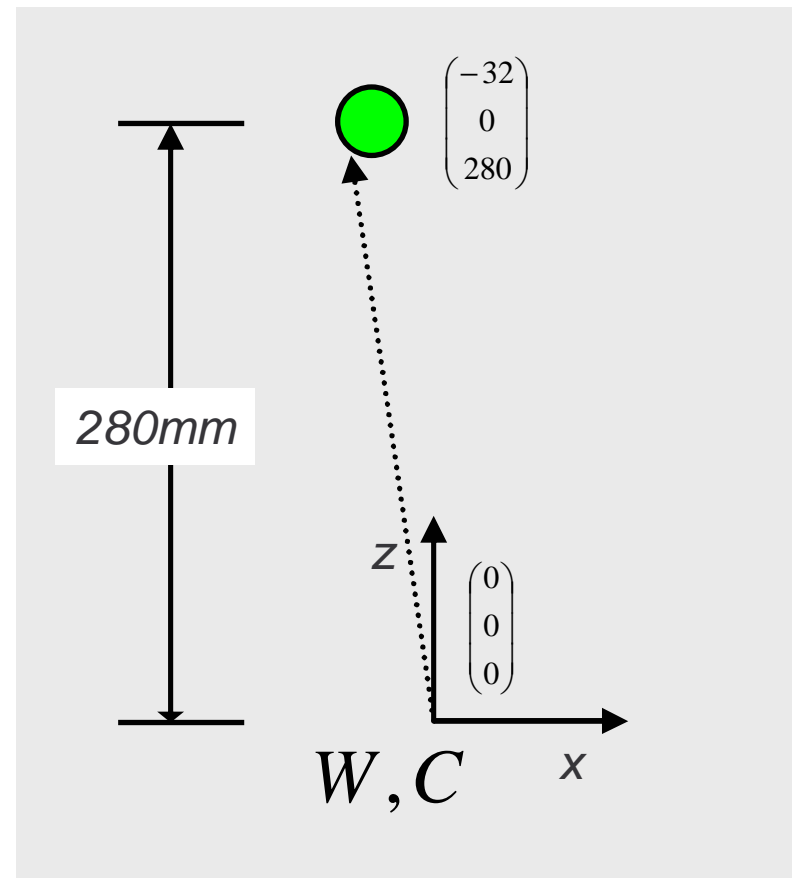
- This formulation is nice in that it separates the process from world coordinates to image coordinates into 2 steps
  - $M_{ext}$  transforms points in the world frame to points in the camera frame
  - $M_{int}$  transforms points from the camera frame to points on the image plane

# Example 1: Intrinsic Parameters Known & Camera Frame = World Frame

- Returning to our iSight Example:



- $P_w = [-32\text{mm}, 0, 280\text{mm}]^T$
- What is  $p_{im}$ ?



## Example 1 (cont'd)

- For  $f=2.8\text{mm}$  and  $s_x=s_y=5\mu\text{m}$ , and assuming that  $(o_x, o_y) = (320, 240)$

$$M_{\text{int}} = \begin{bmatrix} f/s_x & 0 & o_x \\ 0 & -f/s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 560 & 0 & 320 \\ 0 & -560 & 240 \\ 0 & 0 & 1 \end{bmatrix}$$

# Example 1 (cont'd)

- When the camera frame is coincident with the world frame, we have

$$M_{ext} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & -R_1^T t \\ r_{21} & r_{22} & r_{23} & -R_2^T t \\ r_{31} & r_{32} & r_{33} & -R_3^T t \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

# Example 1 (cont'd)

- From these we obtain

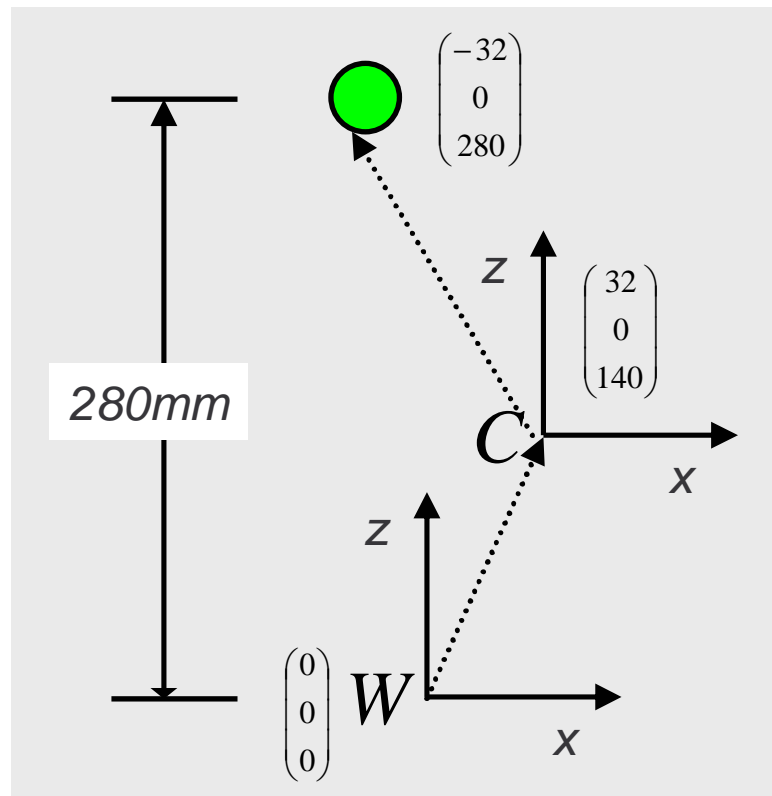
$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = M_{\text{int}} M_{\text{ext}} \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix} = \begin{bmatrix} 560 & 0 & 320 \\ 0 & -560 & 240 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} -32 \\ 0 \\ 280 \\ 1 \end{bmatrix}$$

and the result for  $p_{im}$  as

$$p_{im} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 71680 \\ 67200 \\ 280 \end{pmatrix} = \frac{1}{280} \begin{pmatrix} 256 \\ 240 \\ 1 \end{pmatrix} \equiv \begin{pmatrix} 256 \\ 240 \\ 1 \end{pmatrix}$$

# Example 2: Intrinsic Parameters Known & Camera Frame Translated

- Let us now assume that the camera frame  $C$  is translated from the world frame  $W$  by  $t = [32, 0, 100]^T$



## Example 2 (cont'd)

- From these we obtain

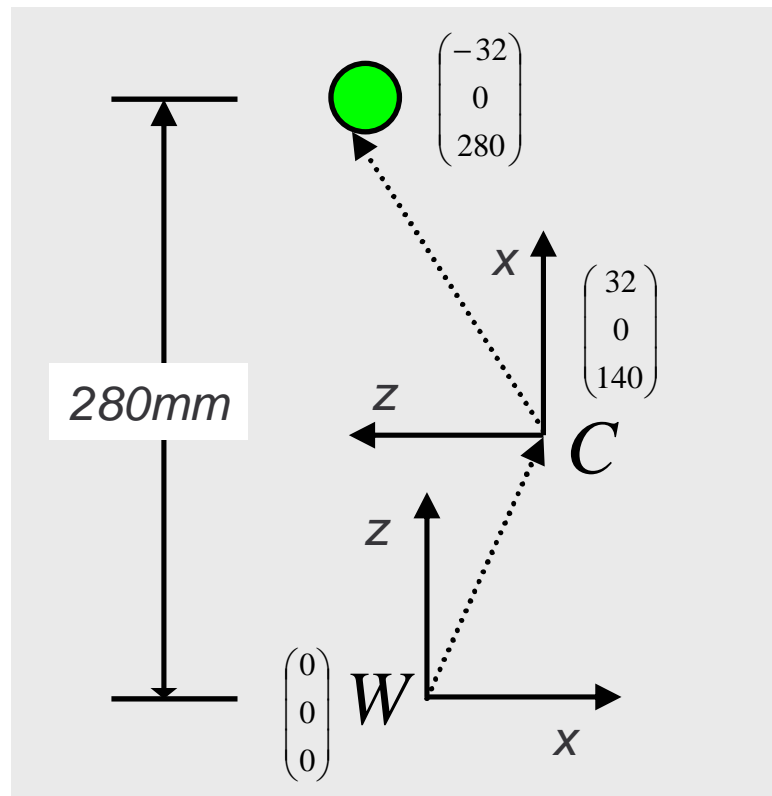
$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = M_{\text{int}} M_{\text{ext}} \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix} = \begin{bmatrix} 560 & 0 & 320 \\ 0 & -560 & 240 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -32 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -140 \end{bmatrix} \begin{bmatrix} -32 \\ 0 \\ 280 \\ 1 \end{bmatrix}$$

and the result for  $p_{im}$  as

$$p_{im} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 8960 \\ 33600 \\ 140 \end{pmatrix} = \frac{1}{140} \begin{pmatrix} 64 \\ 240 \\ 1 \end{pmatrix} \equiv \begin{pmatrix} 64 \\ 240 \\ 1 \end{pmatrix}$$

# Example 3: Intrinsic Parameters Known & Camera Frame Translated/Rotated

- Assume that the camera frame  $C$  is translated by  $t = [32, 0, 100]^T$  and then rotated by an angle of 90 degrees about the  $y$ -axis



# Properties of Rotation Matrices

- $\det(R) = 1, R^T=R^{-1}$
- $R=I$  denotes no rotation
- The rows/columns of  $R$  form an orthonormal basis
- In three dimensions:

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

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

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

## Example 3 (cont'd)

- From these we obtain

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = M_{int} M_{ext} \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix} = \begin{bmatrix} 560 & 0 & 320 \\ 0 & -560 & 240 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 & -140 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 32 \end{bmatrix} \begin{bmatrix} -32 \\ 0 \\ 280 \\ 1 \end{bmatrix}$$

and the result for  $p_{im}$  as

$$p_{im} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 98880 \\ 15360 \\ 64 \end{pmatrix} = \frac{1}{64} \begin{pmatrix} 1545 \\ 240 \\ 1 \end{pmatrix} \equiv \begin{pmatrix} 1545 \\ 240 \\ 1 \end{pmatrix}$$

# Camera Calibration

- Until this point, it was assumed that the intrinsic parameters ( $f, o_x, o_y$ ) were known *a priori*
- While these can be estimated from the components, for accurate stereo reconstruction, motion estimation, *etc.*, more accurate values are required
- As such, an explicit calibration procedure is required to estimate the intrinsic parameters
- We will also use this to recover the extrinsic parameters
- We will ignore geometric effects (lens distortions) in this calibration procedure

# The Projection Matrix

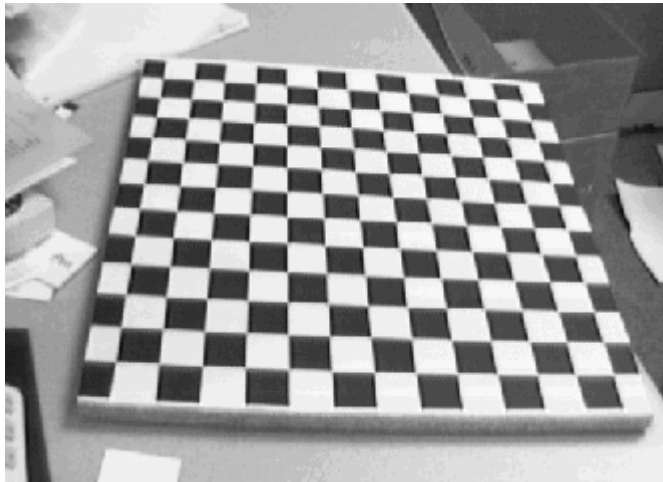
- We know that world and image coordinates are related by a projection Matrix  $M$

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = M_{\text{int}} M_{\text{ext}} \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix} = M \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix}$$

- The calibration procedure we will discuss is made up of 3 phases
  - *Obtain a set of world/image correspondences*
  - *Estimate  $M$  from the correspondences*
  - *Decompose  $M$  into its respective components*

# Step 1: Obtaining World/Image Correspondences

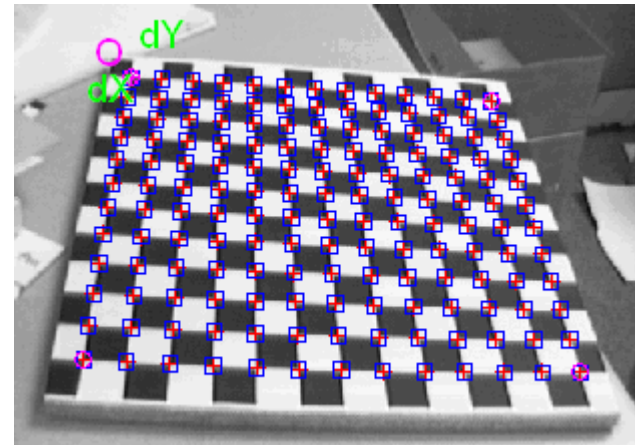
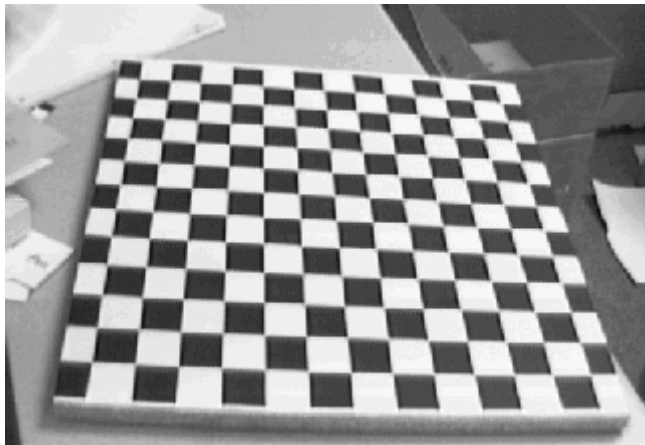
- This calibration assumes that we can obtain an accurate set of correspondences between known world and image points
- These are recovered from standard calibration patterns



• [http://www.vision.caltech.edu/bouguetj/calib\\_doc/](http://www.vision.caltech.edu/bouguetj/calib_doc/)

# Obtaining World/Image Correspondences

- By using such patterns, we can set  $Z_w=0$  for the board plane
- Knowing the pattern size, the position of each checkboard corner is well defined in the world
- Corner extraction techniques can then be used to automate most of the feature extraction process



# The Projection Matrix

- We now have the image points and their world correspondences

$$\begin{pmatrix} x_{im} \\ y_{im} \\ 1 \end{pmatrix} \propto \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = M \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix}$$

- We can use this to calculate  $M$  to *an unknown scale factor*
- $M$  then has only 11 independent entries

# The Projection Matrix (cont'd)

- We can write  $M$  as

$$M = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix}$$

- And then  $x_{im}$ ,  $y_{im}$  as

$$x = \frac{x_1}{x_3} = \frac{m_{11}X_W + m_{12}Y_W + m_{13}Z_W + m_{14}}{m_{31}X_W + m_{32}Y_W + m_{33}Z_W + m_{34}}$$

$$y = \frac{y_1}{x_3} = \frac{m_{21}X_W + m_{22}Y_W + m_{23}Z_W + m_{24}}{m_{31}X_W + m_{32}Y_W + m_{33}Z_W + m_{34}}$$

- These equations are both linear functions of our unknowns  $m_{ij}$

# Step 2: Solving for M

- From this we construct the linear system of equations

$$Am = 0$$

World Point

where A is defined by

Image Point

$$A = \begin{bmatrix} X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 & -x_1 X_1 & -x_1 Y_1 & -x_1 Z_1 & -x_1 \\ 0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 & -y_1 X_1 & -y_1 Y_1 & -y_1 Z_1 & -y_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ X_i & Y_i & Z_i & 1 & 0 & 0 & 0 & 0 & -x_i X_i & -x_i Y_i & -x_i Z_i & -x_i \\ 0 & 0 & 0 & 0 & X_i & Y_i & Z_i & 1 & -y_i X_i & -y_i Y_i & -y_i Z_i & -y_i \end{bmatrix}$$

and  $m$  by

$$m = [m_{11}, m_{12}, \dots, m_{34}]^T$$

$i$  pairs of correspondences

# Singular Value Decomposition (SVD)

- SVD provides an efficient means for solving rank-deficient systems of homogeneous linear equations
- Any  $m \times n$  matrix  $A$  can be decomposed into the product of three matrices

$$A = UDV^T$$

where

1. The columns of the  $m \times m$  matrix  $U$  and the  $n \times n$  matrix  $V$  are mutually orthogonal vectors
2. The  $m \times n$  matrix  $D$  is diagonal, and its elements  $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_n \geq 0$  are the *singular values* of the matrix

# Some Properties of SVD

1. If  $A$  is a rectangular matrix, the number of non-zero singular values equals the *rank* of  $A$
2. The columns of  $U$  corresponding to the non-zero singular values span the *range* of  $A$
3. The columns of  $V$  corresponding to the zero singular values span the *null space* of  $A$

## SVD (cont'd)

- SVD provides an efficient means for solving rank-deficient systems of homogeneous linear equations

$$Am = 0$$

- In our example,  $M$  has only 11 independent parameters and so  $A$  has a rank of 11
- By property 1,  $A$  will have exactly 1 zero singular value
- By property 3, *one* solution for  $m$  then corresponds to the corresponding column of  $V$

## Step 3: Decomposing $M$

- Recall that we have only recovered  $M$  TO AN UNKNOWN SCALE FACTOR
- If we expand  $M$ , we obtain

$$M = \begin{bmatrix} f_x r_{11} + o_x r_{31} & f_x r_{12} + o_x r_{32} & f_x r_{13} + o_x r_{33} & f_x t_x + o_x t_z \\ -f_y r_{21} + o_y r_{31} & -f_y r_{22} + o_y r_{32} & -f_y r_{23} + o_y r_{33} & -f_y t_y + o_y t_z \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix}$$

where  $f_x = f/s_x$  and  $f_y = f/s_y$

- From our solution vector,  $m$  we also construct 3 vectors where

$$q_i = [m_{i1} \quad m_{i2} \quad m_{i3}]^T, \quad i = 1, 2, 3$$

## Step 2: Decomposing $M$ (cont'd)

- From the last row of  $M$

$$M = \begin{bmatrix} f_x r_{11} + o_x r_{31} & f_x r_{12} + o_x r_{32} & f_x r_{13} + o_x r_{33} & f_x t_x + o_x t_z \\ -f_y r_{21} + o_y r_{31} & -f_y r_{22} + o_y r_{32} & -f_y r_{23} + o_y r_{33} & -f_y t_y + o_y t_z \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix}$$

we can infer the scale to a sign change since

$$\sqrt{m_{31}^2 + m_{32}^2 + m_{33}^2} = |\alpha| \sqrt{r_{31}^2 + r_{32}^2 + r_{33}^2} = |\alpha|$$

- By dividing the solution vector  $m$  by  $\alpha$  we now have  $M$  to a sign change

## Step 2: Decomposing $M$ (cont'd)

$$M = \begin{bmatrix} f_x r_{11} + o_x r_{31} & f_x r_{12} + o_x r_{32} & f_x r_{13} + o_x r_{33} & f_x t_x + o_x t_z \\ -f_y r_{21} + o_y r_{31} & -f_y r_{22} + o_y r_{32} & -f_y r_{23} + o_y r_{33} & -f_y t_y + o_y t_z \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix}$$

- We can infer the image center

$$o_x = q_1^T q_3$$

$$o_y = q_2^T q_3$$

since the rows of the rotation matrix are orthogonal and orthonormal

## Step 2: Decomposing $M$ (cont'd)

$$M = \begin{bmatrix} f_x r_{11} + o_x r_{31} & f_x r_{12} + o_x r_{32} & f_x r_{13} + o_x r_{33} & f_x t_x + o_x t_z \\ -f_y r_{21} + o_y r_{31} & -f_y r_{22} + o_y r_{32} & -f_y r_{23} + o_y r_{33} & -f_y t_y + o_y t_z \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix}$$

- We can then infer the focal length from

$$f_x = \sqrt{q_1^T q_1 - o_x^2}$$
$$f_y = \sqrt{q_2^T q_2 - o_y^2}$$

since the rows of the rotation matrix are orthogonal and orthonormal

## Step 2: Decomposing $M$ (cont'd)

$$M = \begin{bmatrix} f_x r_{11} + o_x r_{31} & f_x r_{12} + o_x r_{32} & f_x r_{13} + o_x r_{33} & f_x t_x + o_x t_z \\ -f_y r_{21} + o_y r_{31} & -f_y r_{22} + o_y r_{32} & -f_y r_{23} + o_y r_{33} & -f_y t_y + o_y t_z \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix}$$

- By substitution, we can then derive the remaining extrinsic parameters (where sigma is the sign):

$$r_{1i} = \sigma (m_{1i} - o_x m_{3i}) / f_x, \quad i = 1, 2, 3$$

$$r_{2i} = \sigma (o_y m_{3i} - m_{2i}) / f_y, \quad i = 1, 2, 3$$

$$t_z = \sigma m_{34}$$

$$t_x = \sigma (o_x t_z - m_{14}) / f_x$$

$$t_y = \sigma (o_y t_z - m_{24}) / f_y$$

## In Practice...

- While you could get away with 6 image/world correspondences (12 equations for 11 unknowns), it is better to use many more
- There will not be a zero singular value due to noise. As such, the smallest singular value is treated as the zero value
- Due to noise, the “rotation” matrix recovered will NOT be orthogonal. We can find the closest rotation matrix to it also through SVD techniques

# Summary

- The Projection Matrix  $M$  allows us to directly relate image and world coordinates
- We can recover the projection matrix through a proper calibration procedure
- The strength of the calibration required is strongly dependent upon the application
  - Strong calibration required for parts inspection, stereo, etc.
  - Weaker calibration may be acceptable for navigation
- We will employ a similar procedure to infer camera (robot) motion without knowing the world points *a priori*