Examples
Example 1

- Project some 3D points (represented in the coordinate frame attached to the camera) onto an image.
- Assume the image is 350x350 pixels, and the intrinsic camera parameter matrix is \( K \).
- Render the image showing the projected points.

Here are the points (each column is \( X,Y,Z \)):

\[
\begin{align*}
8 & \quad 1 & \quad 8 & \quad 0 \\
6 & \quad 6 & \quad -2 & \quad 0 \\
20 & \quad 22 & \quad 24 & \quad 20 \\
\end{align*}
\]

Here is our intrinsic camera parameter matrix \( K \):

\[
\begin{bmatrix}
400 & 0 & 175 \\
0 & 400 & 175 \\
0 & 0 & 1 \\
\end{bmatrix}
\]
clear all
close all

% Here are some 3D points, in camera coordinates. Each column is an XYZ
% point, with a 1 added in the fourth row.
p_c = [
    8   1   8   0;
    6   6  -2  0;
   20  22  24  20;
    1   1   1   1 ];

% Here is our intrinsic camera parameter matrix.
K = [
    400 0  175;
     0 400  175;
     0   0    1];

% The extrinsic camera matrix describes the pose of the camera frame with
% respect to the object's frame. It is composed of the rotation matrix R
% and the translation vector t, in a 3x4 matrix: [R t].

% In this case, since the object is already in the camera's coordinate
% frame, R is identity and t is zero:
Mext = [ eye(3) zeros(3,1) ];

% The full camera projection matrix is K*Mext, a 3x4 matrix.

% To project the points, just multiply the points by K*Mext.
p_img = K*Mext*p_c; % Each column is an image point

% Divide through by the last element of each column, since these are
% homogeneous points.
p_img(1,:) = p_img(1,:) ./ p_img(3,:);
p_img(2,:) = p_img(2,:) ./ p_img(3,:);
p_img(3,:) = p_img(3,:) ./ p_img(3,:);
% Create an image showing those points.
HEIGHT = 350;
WIDTH = 350;
I = zeros(HEIGHT, WIDTH);
for i=1:size(p_img,2)
    r = p_img(2,i);     % row is y
    c = p_img(1,i);     % col is x
    r = round(r);       % round to nearest integer
    c = round(c);
% Make sure (r,c) is within image boundaries.
    if r>0 && r<=HEIGHT && c>0 && c<=WIDTH
        I(r,c) = 255;
    end
end
imshow(I);

% Can also use Matlab's plot function. This will draw on top of an already % displayed image.
hold on
plot(p_img(1,:), p_img(2,:), 'Color', 'w', 'Marker', 'd', 'LineStyle', 'none');
Example 2

• The pose of an object (or model) with respect to the world is given by

\[
H_{m\_w} =
\begin{bmatrix}
0.9254 & 0.0180 & 0.3785 & 0.5000 \\
0.1632 & 0.8826 & -0.4410 & -0.5000 \\
-0.3420 & 0.4698 & 0.8138 & 5.0000 \\
0 & 0 & 0 & 1.0000
\end{bmatrix}
\]

\text{Pose of “model to world”}

• A camera has pose with respect to the world:
  – Orientation of the camera is aligned with the world
  – The origin of the camera is at world point = [0;0;-2]

• Draw (in 3D) the coordinate axes of the world, the camera, and the model
clear all % good idea to do these at the beginning of each program

close all

% Draw a dot at the world origin.  This also opens a new figure.
plot3(0,0,0,'.');  
hold on

drawCoordinateAxes(eye(4), 'W');  % Draw world axes

% Camera pose.
H_c_w = ...  
    [ 1 0 0 0;  
      0 1 0 0; 
      0 0 1 -2;  
      0 0 0 1];
drawCoordinateAxes(H_c_w, 'C');  % Draw camera frame axes

% Model pose.
H_m_w = [  
    0.9254  0.0180  0.3785  0.5000;  
    0.1632  0.8826 -0.4410 -0.5000; 
    -0.3420  0.4698  0.8138  5.0000;  
         0  0  0  1.0000];
drawCoordinateAxes(H_m_w, 'M');  % Draw camera frame axes

xlabel('X'), ylabel('Y'), zlabel('Z');  % Label the axes

% These options make the 3D display look nicer.
axis equal
axis vis3d
Example 2 (continued)

• Assume the camera has image size 640x480, focal length = 600 pixels
• Create an image of the model as seen by the camera
• We’ll just draw the XYZ axes of the model onto the image
Example 2 (continued)

• Approach to project points
  – Create intrinsic camera matrix K
  – Create extrinsic matrix M_{ext}, representing the pose of the model with respect to camera
  – Project the model points using $p = K M_{ext} P$

• Approach to create image
  – Create a blank image
  – Write the points into (or plot onto) the image
  – Draw lines onto the image