ECE661 Computer Vision Homework 6

Camera Models and Image Reprojection

Rong Zhang

10/30/2008

1 Problem

In this homework, we consider camera projection. Given a camera matrix P, we are going to determine the camera parameters (camera model), e.g., camera center etc. We then use an old image as the object placed at a distance of two focal length plane parallel to the sensor plane. To find the image of this old image on the sensor plane under projective camera P is the objective of this homework.

2 Camera Models

A general projective camera $P_{3\times 4} = [\vec{p}_1, \vec{p}_2, \vec{p}_3, \vec{p}_4]$ maps world points $\vec{X}$ to image points $\vec{x}$ by $\vec{x} = P\vec{X}$. The points $\vec{X} = (X, Y, Z, 1)^T$ and $\vec{x} = (x, y, 1)^T$ are homogenous represented. As shown in class, we know that the homogenous representation of the image point corresponding to world point at infinity along the world X, Y and Z directions are the column vectors $\vec{p}_1$,$\vec{p}_2$ and $\vec{p}_3$ of the camera matrix P. In addition, the 4th column vector $\vec{p}_4$ is the homogenous representation of the image of world origin.

Therefore, given the data from homework problem, we have

$$P = \begin{pmatrix} 5 & 400 & 500 & 20 \\ 100 & 300 & 490 & 20 \\ 1 & 1 & 1 & 5 \end{pmatrix}. \tag{1}$$

If we check the rank of the left 3×3 submatrix of P, we can find that the submatrix $[\vec{p}_1, \vec{p}_2, \vec{p}_3]$ is non-singular. Therefore, it is a FINITE PROJECTIVE camera.

If we denote the center of projection of the camera located in the world coordinate (camera center) as $\vec{C}$ (homogenous representation), we have $P\vec{C} = 0$, i.e., the camera center is the null space basis of the matrix P. So, the camera is located at

$$\vec{C} = [-4.1453, -4.2807, 3.4260, 1]^T. \tag{2}$$

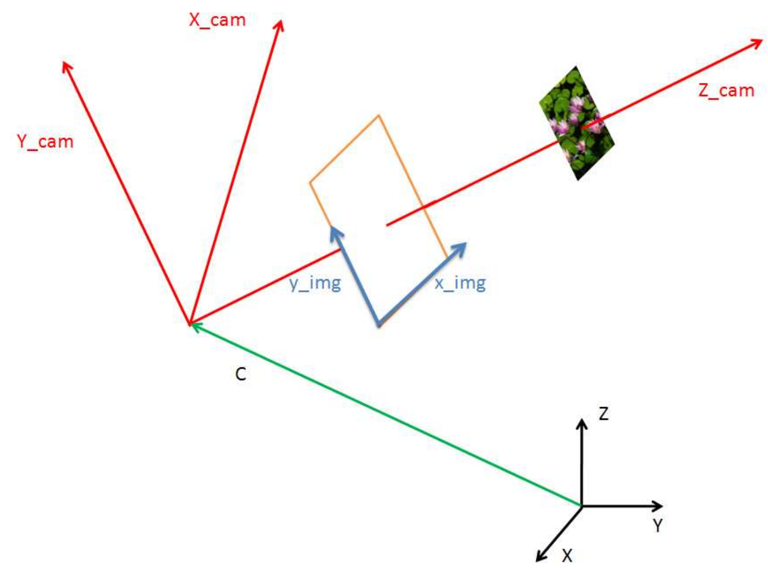

Figure 1: Camera model

3 Camera Projection

Now, we are considering the projection of the object in world coordinate into the camera sensor plane. Figure 1 illustrates the coordinate relationship among world coordinates, camera coordinates and sensor plane coordinates.

From class, we know that $P = KR[I, -\vec{\tilde{C}}]$ where $\vec{C} = \begin{bmatrix} \vec{\tilde{C}} \\ 1 \end{bmatrix}$, $K = \begin{bmatrix} m_x f & s & m_x p_x \\ 0 & m_y f & m_y p_x \\ 0 & 0 & 1 \end{bmatrix}$ is a upper triangular matrix, R is the rotation matrix. We are now trying to determine the camera parameters by RQ decomposition. The method used is the procedure described in textbook A4.1.1. The MATLAB code is listed below.

Note that in order to guarantee the elements $m_x f$ and $m_y f$ be positive, the matrix R can be adjusted accordingly.

The matrix K obtained is

$$K = \begin{pmatrix} 115.2390 & 351.7622 & 522.5020 \\ 0 & 275.8019 & 513.8417 \\ 0 & 0 & 1.7321 \end{pmatrix}. \tag{3}$$

It is normalized to

$$K = \begin{pmatrix} 66.5333 & 203.0900 & 301.6667 \\ 0 & 159.2343 & 296.6667 \\ 0 & 0 & 1 \end{pmatrix}. \tag{4}$$

After obtaining camera parameters, we are going to reproject an old image to the camera sensor plane. Assume the old image is placed at a distance of $2f$ from the camera center (parallel to the sensor plane), with the principal axis passing through the old image center as shown in fig. 1. If we now consider the object (old image) point in the camera coordinate, we can simplify the problem by neglecting the rotation matrix R and camera translation vector $\vec{C}$. We have

$$\vec{x} = K[I, \vec{0}^T]\vec{X}_{cam} \tag{5}$$

and

$$\vec{X}_{cam} = [X_{cam}, Y_{cam}, 2f, 1]^T \tag{6}$$

where X_{cam} and Y_{cam} are simply the pixel locations of the old image.

Note that, the old image matrix pixel coordinates are different from the camera coordinates, so is the reprojected image and sensor plane as shown in fig. 2.

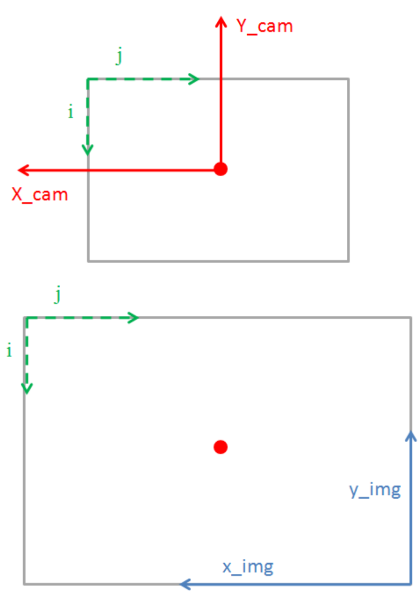

Figure 2: Coordinate relationship (top: old image plane ($Z_{cam} = 2f$); bottom: sensor plane. red dot represents intersection with principal axis)

4 Results

In this section, the reprojected images are shown. We can see from the results that the camera corresponded with matrix P is an affine camera. The original old image is affine distorted after the reprojection. $f = 120$ was used for the projection below.

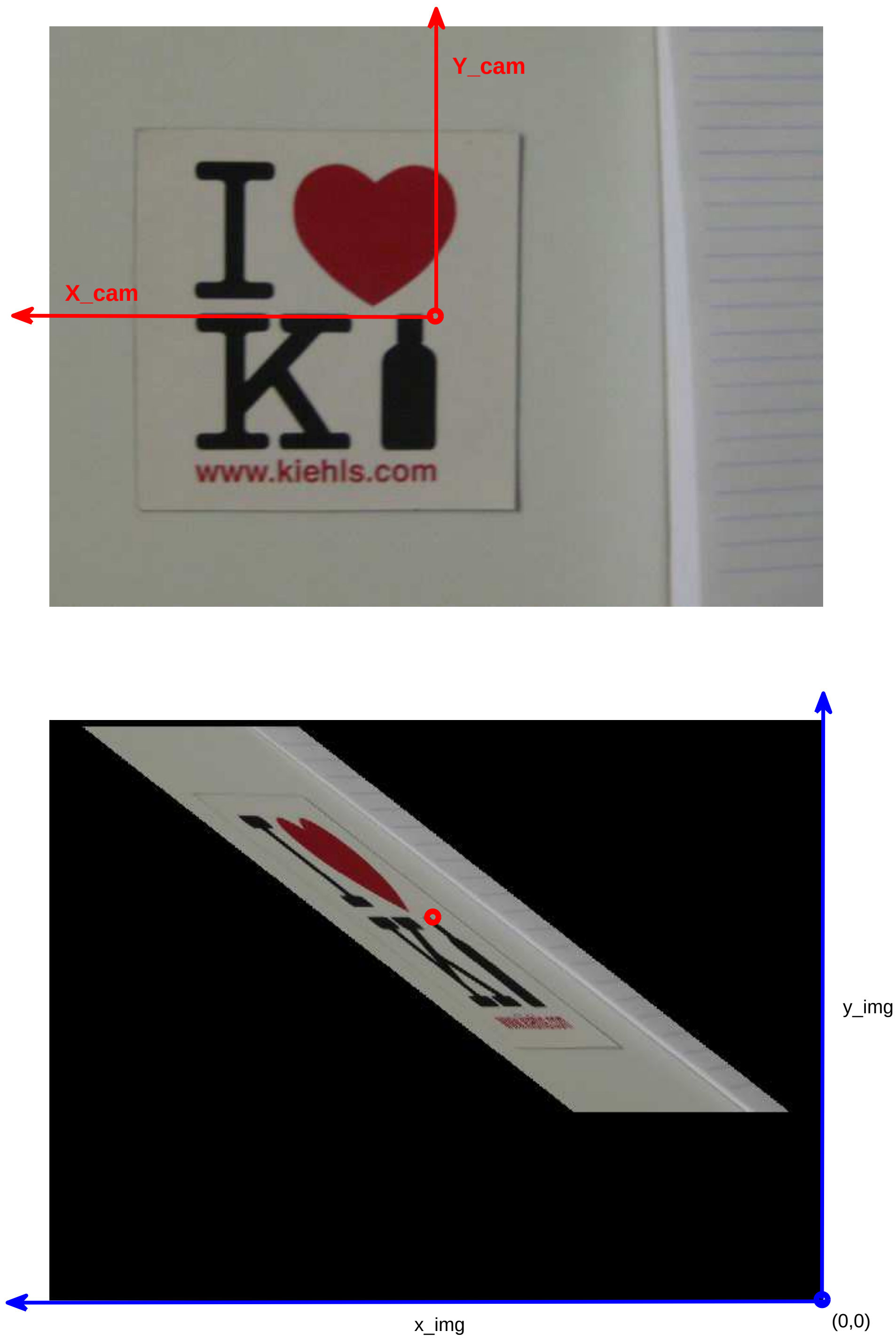

Figure 3: Top: old image; bottom: reprojected image (red dot represents intersection with principal axis)

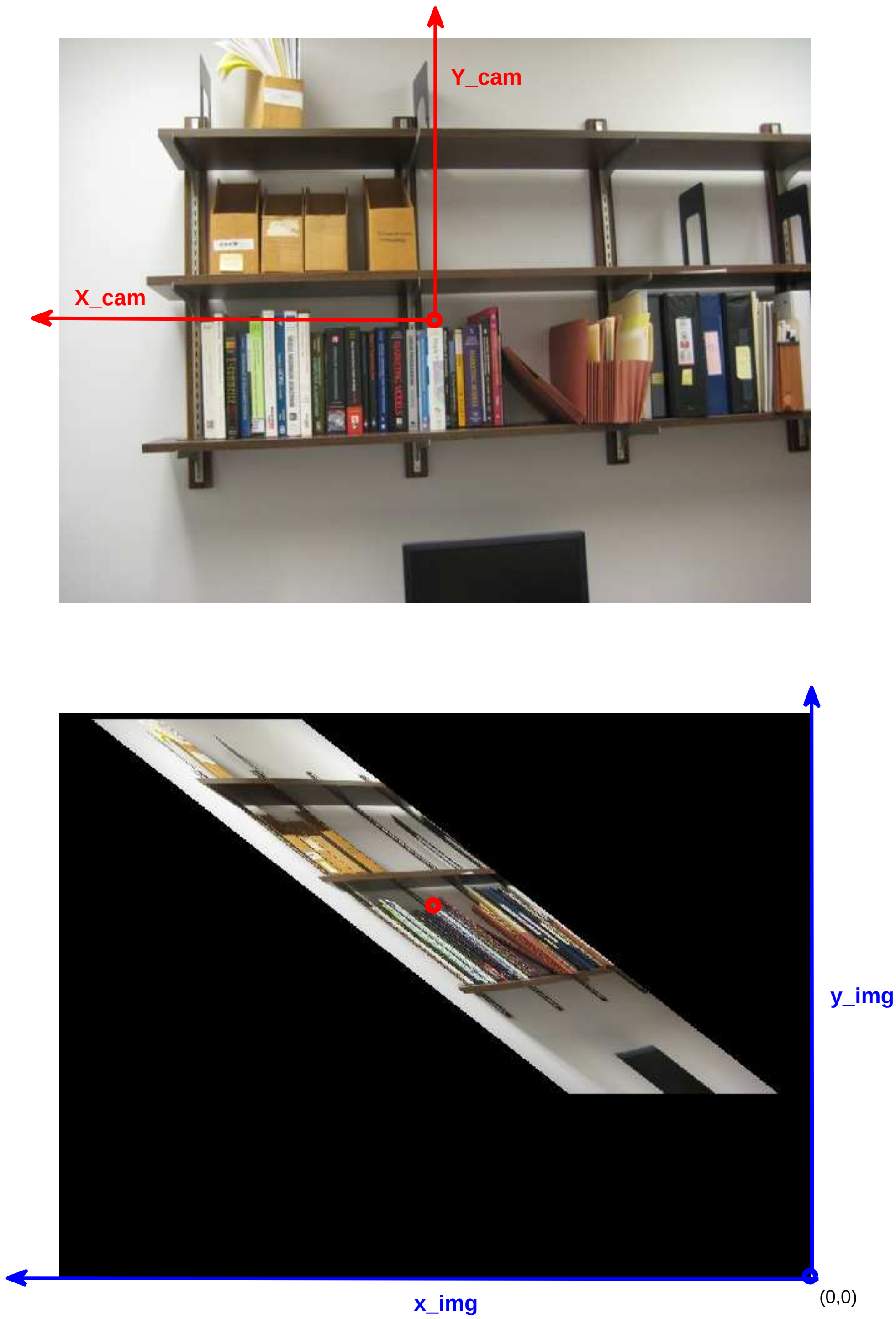

Figure 4: Top: old image; bottom: reprojected image (red dot represents intersection with principal axis)

```
P = [5 400 500 20; 100 300 490 20; 1 1 1 5];
A = P(:,1:3);

syms a
Qx = [1 0 0;0 cos(a) -sin(a); 0 sin(a) cos(a)];
Qy = [cos(a) 0 sin(a);0 1 0; -sin(a) 0 cos(a)];
Qz = [cos(a) -sin(a) 0;sin(a) cos(a) 0; 0 0 1];

tmpR = A*Qx;
y = tmpR(3,2);
ax_v = solve(y);
Qx_v = subs(Qx, 'a', ax_v);

A = A*Qx_v;
tmpR = A*Qy;
y = tmpR(3,1);
ay_v = solve(y);
Qy_v = subs(Qy, 'a', ay_v);

A = A*Qy_v;
tmpR = A*Qz;
y = tmpR(2,1);
az_v = solve(y);
Qz_v = subs(Qz, 'a', az_v);

R = A*Qz_v;
Q = Qz_v'*Qy_v'*Qx_v';

C = null(P);
C = C/C(4);
Ct = C(1:3);

% R =
% -115.2390  -351.7622  522.5020
%      0     -275.8019  513.8417
%      0         0      1.7321

% make alpha_x and alpha_y positive
R(:,1) = -R(:,1);
Q(1,:) = -Q(1,:);
R(:,2) = -R(:,2);
Q(2,:) = -Q(2,:);

% R =
% 115.2390  351.7622  522.5020
%      0    275.8019  513.8417
%      0       0      1.7321

% normalize
R = R/R(3,3);
% R =
%  66.5333  203.0900  301.6667
%    0      159.2343  296.6667
%    0         0          1

R*Q*[eye(3) -Ct]
```

```matlab
img = imread('m2a.jpg');
[height, width, channel] = size(img);
imgout = zeros(height, width, channel);

K = R;

f = 120;
for i = 1:height
    for j = 1:width
        X_cam = [width/2-j height/2-i  2*f 1]';
        x_img = K*[eye(3,3) zeros(3,1)]*X_cam;
        curj = max(1,min(width,round(x_img(1)/x_img(3))));
        curi = max(1,min(height,round(x_img(2)/x_img(3))));
        %[curi curj]
        imgout(curi,curj,:) = img(i,j,:);
    end
end
imgout = flipdim(imgout,1);
imgout = flipdim(imgout,2);
imshow(uint8(imgout));
hold on;
% plot the origin of sensor plane
plot(width,height,'bo','linewidth',3);
% plot the intersection of principle axis and sensor plane
plot(width-round(K(1,3)),height-round(K(2,3)),'ro','linewidth',3);
```