ECE 661 Homework 7

Minwoong Kim

November 18, 2012

1 Calibration Procedure

Full camera calibration requires to find all five intrinsic parameters that encompass focal length and principle point, and six extrinsic parameters that describes its pose with respect to a world reference. The first step of the calibration procedure is to record the images of the calibration pattern from different views. Zhang's calibration algorithm uses a pattern that consists of a number of aligned black squares against a white background.

1.1 Images of the calibration pattern

The printout of the calibration pattern is attached on the wall to be capture by a camera at different views. The reference of the world origin point should be marked first to measure the world coordinates of the corners of the black squares on the calibration pattern. Then, more than 20 images should be recorded at different positions and orientations of the camera. We assume that the camera is thought of as a pinhole camera.

1.2 Corner detection

The corners in the recorded images should be isolated because the corners of the black squares will be used as the correspondences to compute the homography between the world plane and each image plane. The corners in each recorded image will be detected and labelled by the following steps:

1. Apply the Canny edge detector to the image.
2. Use the Hough transform to fit straight lines to the Canny edges.
3. Group the lines into horizontal and vertical line sets.
4. Define the corners as the intersections of the horizontal and vertical lines.
5. Use a Harris corner detector to find the strongest nearby corner to each intersection.

The detected corners finally should be labelled with integers in the same order for all the images; the corresponding corner points in all the images should have the same integer label.

1.3 Camera calibration

In this section, we will apply Zhang's camera calibration algorithm to find the intrinsic parameters and extrinsic parameters of the camera. The corner points detected in each image are used to estimate the homography between the world plane and the image plane. The world origin is set to the first corner located at the most upper left position labelled with 1 in Fig. 1. The measured distance between each

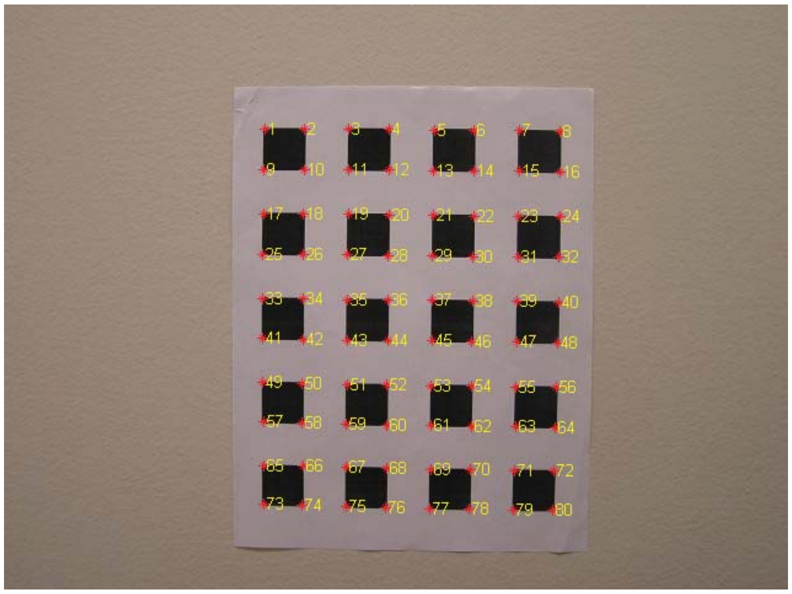


Figure 1: Detected corners labelled with integers in P1010001s.jpg

corner is 24 *mm* in the real printout of the calibration pattern with a ruler. Therefore, the world coordinates of the corners can be represented as (0, 0), (24, 0), (48, 0), ..., (0, 24), ..., (168, 216) in ascending order of the label. We only care about the X and Y coordinates of the corners since the printout is attached on the wall and they have the same Z value. Likewise, we assume that the model plane is on $Z = 0$ of the world coordinate system without loss of generality. The homogeneous coordinates of a 2D point in the image plane is denoted by $\mathbf{x} = [x, y, 1]^\top$ and the homogeneous coordinates of a 3D point in the world coordinate system is denoted by $\mathbf{X} = [X, Y, Z, 1]^\top$. The relationship between a 3D point $\mathbf{X}$ and its image projection $\mathbf{x}$ is given by

$$\mathbf{x} = \mathbf{K} \begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix} \mathbf{X}, \tag{1}$$

where ($\mathbf{R}$, $\mathbf{t}$), called the extrinsic parameters, is the rotation and transition, and $\mathbf{K}$, called the camera intrinsic matrix, is given by

$$\begin{bmatrix} \alpha_x & s & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

with (x_0, y_0) the coordinates of the principal point, α_x and α_y the scale factors in image x and y axes, and s the parameter describing the skewness of the two image axes. Since we assume that the model plane is on $Z = 0$, we have

$$\begin{aligned}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} &= \mathbf{K}\begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 & \mathbf{t}\end{bmatrix}\begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix} \\ &= \mathbf{K}\begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t}\end{bmatrix}\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix},\end{aligned} \tag{3}$$

where the i^{th} column of the rotation matrix $\mathbf{R}$ by $\mathbf{r}_i$. Therefore, a model point $\mathbf{X}$ and its image $\mathbf{x}$ is related by a homography $\mathbf{H}$:

$$\mathbf{x} = \mathbf{H}\mathbf{X} \qquad \text{with} \qquad \mathbf{H} = \mathbf{K}\begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t}\end{bmatrix}. \tag{4}$$

As we learned in the class, the image of the absolute conic is $\mathbf{K}^{-\top}\mathbf{K}^{-1}$ for the given homography from (4). Therefore, two circular points, $\mathbf{HI} = \mathbf{h}_1 + i\mathbf{h}_2$ and $\mathbf{HJ} = \mathbf{h}_1 - i\mathbf{h}_2$ are on the absolute conic where $\mathbf{H} = \begin{bmatrix}\mathbf{h}_1 & \mathbf{h}_2 & \mathbf{h}_3\end{bmatrix}$. This fact gives two constraints as follows:

$$\mathbf{h}_1^\top\mathbf{K}^{-\top}\mathbf{K}^{-1}\mathbf{h}_2 = 0 \tag{5}$$

$$\mathbf{h}_1^\top\mathbf{K}^{-\top}\mathbf{K}^{-1}\mathbf{h}_1 = \mathbf{h}_2^\top\mathbf{K}^{-\top}\mathbf{K}^{-1}\mathbf{h}_2\,. \tag{6}$$

To solve the two constraints, let $\mathbf{B} = \mathbf{K}^{-\top}\mathbf{K}^{-1} \equiv \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{12} & B_{22} & B_{23} \\ B_{13} & B_{23} & B_{33}\end{bmatrix}$ and the i^{th} column vector of $\mathbf{H}$ be $\mathbf{h}_i = [h_{i1}, h_{i2}, h_{i3}]^\top$. Since $\mathbf{B}$ is symmetric, it can be expressed as a 6D vector as $\mathbf{b} = [B_{11}, B_{12}, B_{22}, B_{13}, B_{23}, B_{33}]^\top$. Then, we have

$$\mathbf{h}_i^\top\mathbf{B}\mathbf{h}_j = \mathbf{v}_{ij}^\top\mathbf{b} \tag{7}$$

with

$$\mathbf{v}_{ij}^\top = \begin{bmatrix} h_{i1}h_{j1} & h_{i1}h_{j2} + h_{i2}h_{j1} & h_{i2}h_{j2} & h_{i3}h_{j1} + h_{i1}h_{j3} & h_{i3}h_{j2} + h_{i2}h_{j3} & h_{i3}h_{j3}\end{bmatrix}.$$

Now, the two fundamental constraints (5) and (6) can be solved as the following two homogeneous equations:

$$\begin{bmatrix} \mathbf{v}_{12}^\top \\ (\mathbf{v}_{11} - \mathbf{v}_{22})^\top \end{bmatrix}\mathbf{b} = \mathbf{0}\,. \tag{8}$$

If we have n images of the model planes observed, we can stack $2n$ rows in the matrix form of homogeneous equations. As we solved the LLS problems, the solution can be obtained by **SVD** to find a column vector of $\mathbf{V}$ corresponding to the smallest singular value. Once $\mathbf{b}$ is estimated, the intrinsic parameters can be calculated as follows:

$$\begin{aligned}
y_0 &= (B_{12}B_{13} - B_{11}B_{23})/(B_{11}B_{22} - B_{12}^2) \\
\lambda &= B_{33} - [B_{13}^2 + y_0(B_{12}B_{13} - B_{11}B_{23})]/B_{11} \\
\alpha_x &= \sqrt{\lambda/B_{11}} \\
\alpha_y &= \sqrt{\lambda B_{11}/(B_{11}B_{22} - B_{12}^2)} \\
s &= -B_{12}\alpha_x^2\alpha_y/\lambda \\
x_0 &= sy_0/\alpha_y - B_{13}\alpha_x^2/\lambda\,.
\end{aligned}$$

Now, we know all the elements of $\mathbf{K}$ and the extrinsic parameters can be readily computed using $\mathbf{K}$ from (4):

$$\begin{aligned}
\mathbf{r}_1 &= \lambda\mathbf{K}^{-1}\mathbf{h}_1 \\
\mathbf{r}_2 &= \lambda\mathbf{K}^{-1}\mathbf{h}_2 \\
\mathbf{r}_3 &= \mathbf{r}_1 \times \mathbf{r}_2 \\
\mathbf{t} &= \lambda\mathbf{K}^{-1}\mathbf{h}_3\,.
\end{aligned}$$

1.4 Refinement of the calibration parameters

We refine the calibration parameters by minimizing the projection error. Let the j^{th} point on the calibration pattern by $\vec{x}_{M,j}$. Let $\vec{x}_{ij}$ denote the corresponding j^{th} point in the i^{th} image.

$$d_{geom}^2 = \sum_i \sum_j \|\vec{x}_{ij} - \hat{\vec{x}}_{M,j}\|^2\,, \tag{9}$$

where $\hat{\vec{x}}_{M,j}$ is the reprojection of $\vec{x}_{M,j}$ back into the i^{th} camera image and it can be rewritten as

$$d_{geom}^2 = \sum_i \sum_j \|\vec{x}_{ij} - f(\mathbf{K}, \mathbf{R}_i, \mathbf{t}_i, \vec{x}_{M,j})\|^2\,. \tag{10}$$

Since actual degree of freedom of $\mathbf{R}_i$ is 3, we need to place the 3 elements describing $\mathbf{R}_i$ instead of direct 9 elements of the rotation matrix $\mathbf{R}_i$. The rotation matrix can be represented as

$$\mathbf{R} = 1 + \frac{sin(\varphi)}{\varphi}[\vec{w}]_\times + \frac{1 - cos(\varphi)}{\varphi}[\vec{w}]_\times^2\,, \tag{11}$$

where $\vec{w} = [w_x\ w_y\ w_z]^\top$, $\varphi = \|\vec{w}\|$, and $[\vec{w}]_\times = \begin{bmatrix} 0 & -w_z & w_y \\ -w_z & 0 & w_x \\ -w_y & w_x & 0 \end{bmatrix}$. Therefore, we can find the 11 parameters (5 for $\mathbf{K}$, 3 for $\mathbf{R}_i$, 3 for $\mathbf{t}_i$) minimizing the projection error by the Levenberg-Marquardt.

1.5 Conditioning the rotation matrix

Let $\mathbf{Q}$ be the actual calculated rotation matrix, and then we want to find $\mathbf{R}$ that $min_{\mathbf{R}} \|\mathbf{R} - \mathbf{Q}\|_F^2$ subject to $\mathbf{R}^\top \mathbf{R} = \mathbf{I}$. The solution can be obtained by $\mathbf{R} = \mathbf{U}\mathbf{V}^\top$ where $\mathbf{Q} = \mathbf{U}\mathbf{S}\mathbf{V}^\top$.

1.6 Radial distortion

Let (x, y) be the actual pixels without radial distortion. Then, the pixel coordinates distorted by the radial distortion is

$$\begin{aligned} x_{rad} &= x + (x - x_0)(k_1 r^2 + k_2 r^4) \\ y_{rad} &= y + (y - y_0)(k_1 r^2 + k_2 r^4) \\ r^2 &= (x - x_0)^2 + (y - y_0)^2 \end{aligned} \tag{12}$$

The radial distortion parameters k_1 and k_2 also can be refined by adding the parameters to the Levenberg-Marquardt method for minimizing the projection error.

$$d_{geom}^2 = \sum_i \sum_j \|\vec{x}_{ij} - f(\mathbf{K}, k_1, k_2, \mathbf{R}_i, \mathbf{t}_i, \vec{x}_{M,j})\|^2 . \tag{13}$$

Finally, all the refined intrinsic parameters and extrinsic parameters are found, and we completely know the matrix of pinhole camera model $\mathbf{P} = \mathbf{K} \begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix}$.

2 Examples of Extracting and Labelling Corner Points

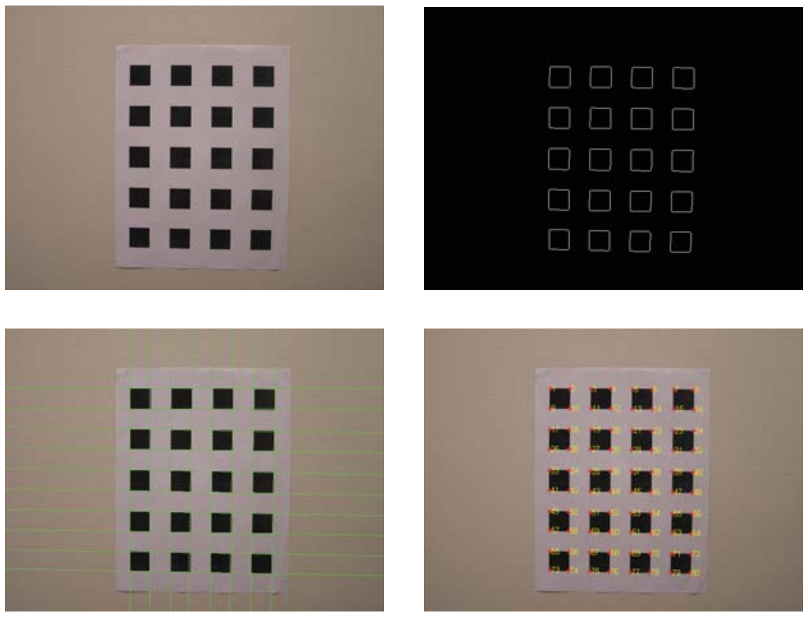


Figure 2: Original, Canny edges, Hough lines, labelled corners of P1010001s.jpg

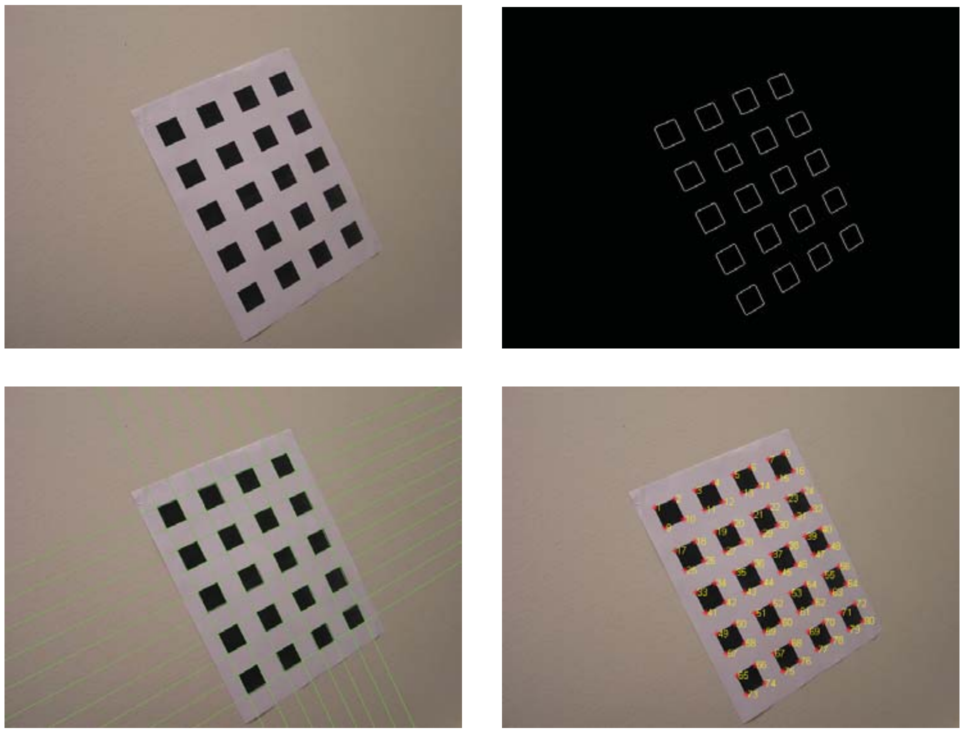


Figure 3: Original, Canny edges, Hough lines, labelled corners of P1010013s.jpg

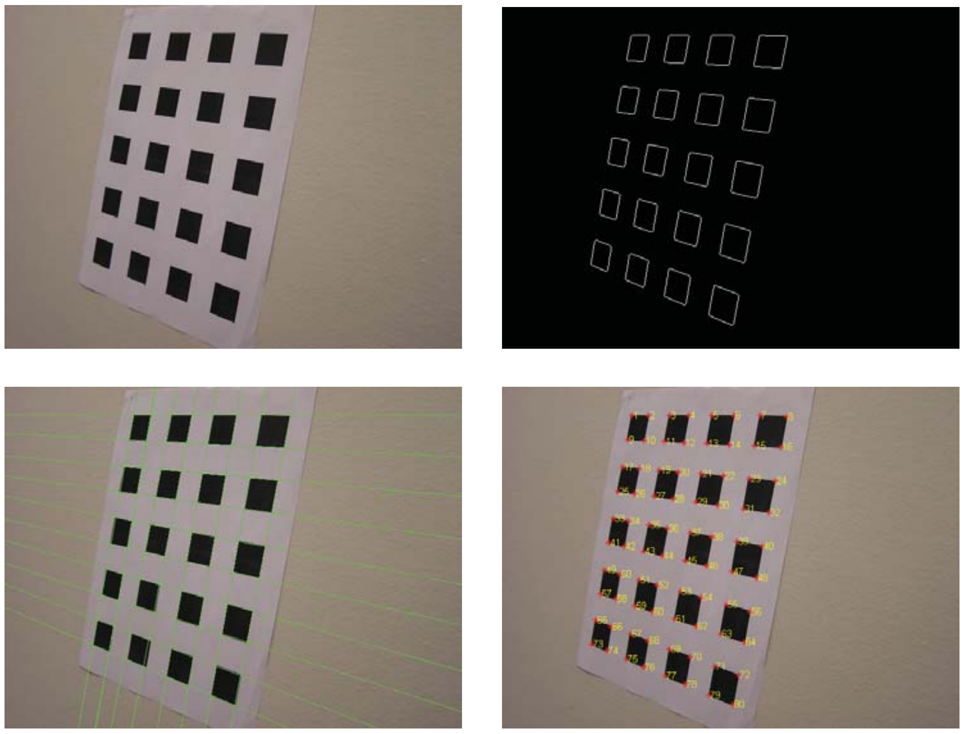


Figure 4: Original, Canny edges, Hough lines, labelled corners of P1010055s.jpg

For self-recorded calibration pattern images:

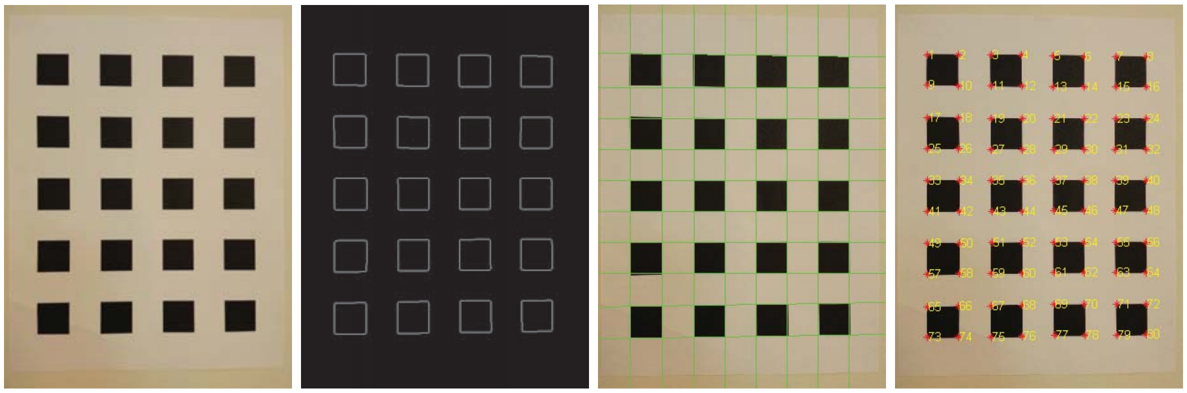


Figure 5: Original, Canny edges, Hough lines, labelled corners of self01.jpg

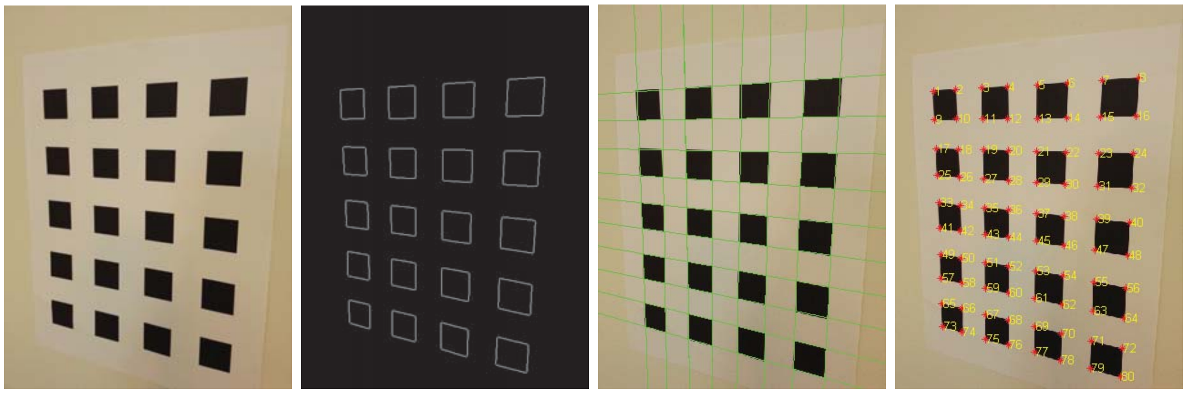


Figure 6: Original, Canny edges, Hough lines, labelled corners of self11.jpg

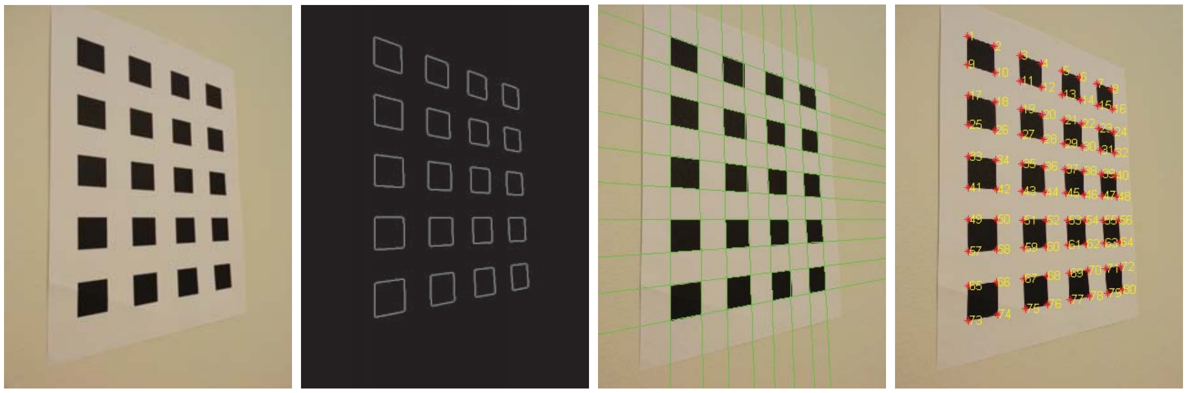


Figure 7: Original, Canny edges, Hough lines, labelled corners of self14.jpg

3 Intrinsic and Extrinsic Parameters

3.1 Intrinsic parameters

For the given calibration pattern images:

$\alpha_x = 716.8271$
$\alpha_y = 718.0265$
$s = 1.6271$ (skewness factor)
$x_0 = 318.9192$
$y_0 = 236.8156$

$$\mathbf{K} = \begin{bmatrix} 716.8271 & 1.6271 & 318.9192 \\ 0 & 718.0265 & 236.8156 \\ 0 & 0 & 1 \end{bmatrix}$$

For the self-recorded calibration pattern images:

$\alpha_x = 527.6535$
$\alpha_y = 526.8709$
$s = 0.1066$ (skewness factor)
$x_0 = 250.6981$
$y_0 = 317.2400$

$$\mathbf{K} = \begin{bmatrix} 527.6535 & 0.1066 & 250.6981 \\ 0 & 526.8709 & 317.2400 \\ 0 & 0 & 1 \end{bmatrix}$$

3.2 Extrinsic parameters

For the given calibration pattern images:

P1010001s.jpg

$$\mathbf{r}_1 = 10^{-5} \begin{bmatrix} 1.1148 \\ 0.0144 \\ -0.0370 \end{bmatrix}, \quad \mathbf{r}_2 = 10^{-5} \begin{bmatrix} -0.0136 \\ 1.1153 \\ 0.0487 \end{bmatrix}, \quad \mathbf{r}_3 = 10^{-9} \begin{bmatrix} 0.0042 \\ -0.0054 \\ 0.1244 \end{bmatrix}, \quad \mathbf{t} = 10^{-5} \begin{bmatrix} -83.0694 \\ -105.3429 \\ 556.8522 \end{bmatrix},$$

$$\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix} = \begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 & \mathbf{t}\end{bmatrix} = 10^{-5}\begin{bmatrix} 1.1148 & -0.0136 & 0.0000 & -83.0694 \\ 0.0144 & 1.1153 & -0.0000 & -105.3429 \\ -0.0370 & 0.0487 & 0.0000 & 556.8522 \end{bmatrix}$$

P1010013s.jpg

$$\mathbf{r}_1 = 10^{-5}\begin{bmatrix} 0.9938 \\ 0.1591 \\ -0.0047 \end{bmatrix}, \quad \mathbf{r}_2 = 10^{-5}\begin{bmatrix} -0.1578 \\ 0.9920 \\ 0.0664 \end{bmatrix}, \quad \mathbf{r}_3 = 10^{-9}\begin{bmatrix} 0.0015 \\ -0.0065 \\ 0.1011 \end{bmatrix}, \quad \mathbf{t} = 10^{-5}\begin{bmatrix} -49.2164 \\ -113.7906 \\ 503.4485 \end{bmatrix},$$

$$\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix} = \begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 & \mathbf{t}\end{bmatrix} = 10^{-5}\begin{bmatrix} 0.9938 & -0.1578 & 0.0000 & -49.2164 \\ 0.1591 & 0.9920 & -0.0000 & -113.7906 \\ -0.0047 & 0.0664 & 0.0000 & 503.4485 \end{bmatrix}$$

For the self-recorded calibration pattern images:

self01.jpg

$$\mathbf{r}_1 = 10^{-4}\begin{bmatrix} 0.6516 \\ -0.0013 \\ 0.0197 \end{bmatrix}, \quad \mathbf{r}_2 = 10^{-4}\begin{bmatrix} 0.0018 \\ 0.6516 \\ -0.0006 \end{bmatrix}, \quad \mathbf{r}_3 = 10^{-8}\begin{bmatrix} -0.0128 \\ 0.0004 \\ 0.4246 \end{bmatrix}, \quad \mathbf{t} = 10^{-4}\begin{bmatrix} -58.6349 \\ -70.7426 \\ 158.9277 \end{bmatrix},$$

$$\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix} = \begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 & \mathbf{t}\end{bmatrix} = 10^{-4}\begin{bmatrix} 0.6516 & 0.0018 & -0.0000 & -58.6349 \\ -0.0013 & 0.6516 & 0.0000 & -70.7426 \\ 0.0197 & -0.0006 & 0.0000 & 158.9277 \end{bmatrix}$$

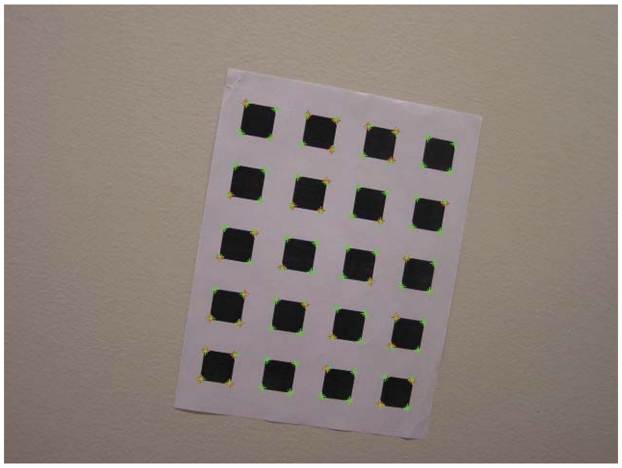
self14.jpg

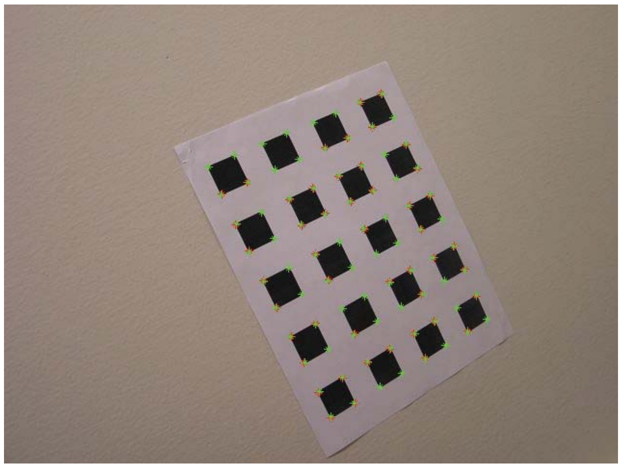
$$\mathbf{r}_1 = 10^{-4}\begin{bmatrix} 0.3754 \\ 0.0218 \\ 0.2817 \end{bmatrix}, \quad \mathbf{r}_2 = 10^{-4}\begin{bmatrix} 0.0134 \\ 0.4664 \\ -0.0499 \end{bmatrix}, \quad \mathbf{r}_3 = 10^{-8}\begin{bmatrix} -0.1325 \\ 0.0225 \\ 0.1748 \end{bmatrix}, \quad \mathbf{t} = 10^{-4}\begin{bmatrix} -28.7303 \\ -59.2300 \\ 118.4851 \end{bmatrix},$$

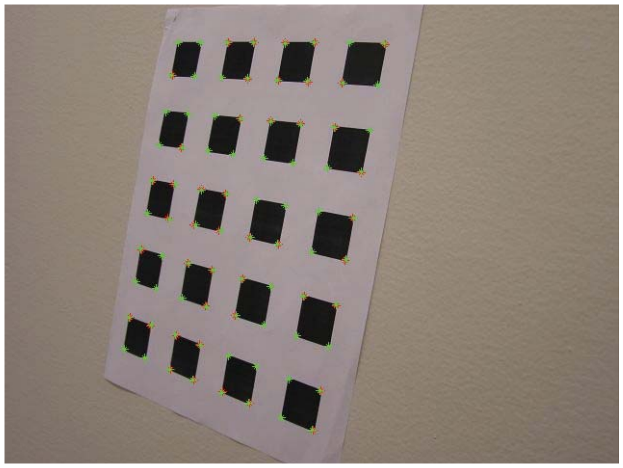
$$\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix} = \begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 & \mathbf{t}\end{bmatrix} = 10^{-4}\begin{bmatrix} 0.3754 & 0.0134 & -0.0000 & -28.7303 \\ 0.0218 & 0.4664 & 0.0000 & -59.2300 \\ 0.2817 & -0.0499 & 0.0000 & 118.4851 \end{bmatrix}$$

4 Reprojected Corner Points

4.1 Given images


(a)


(b)


(c)

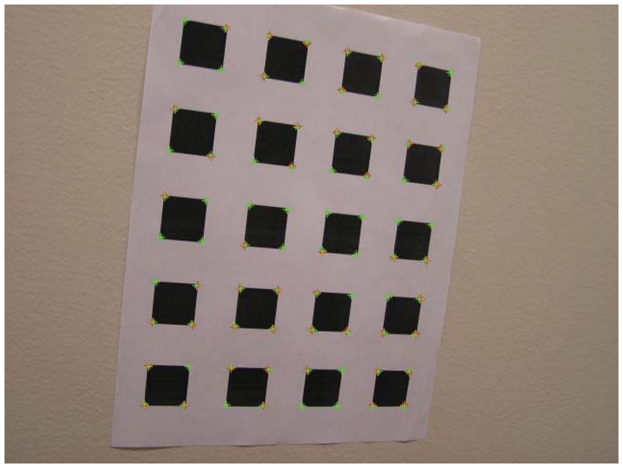
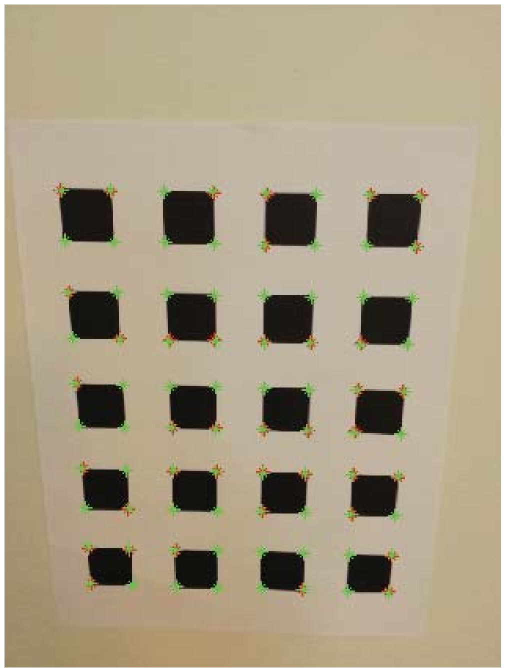
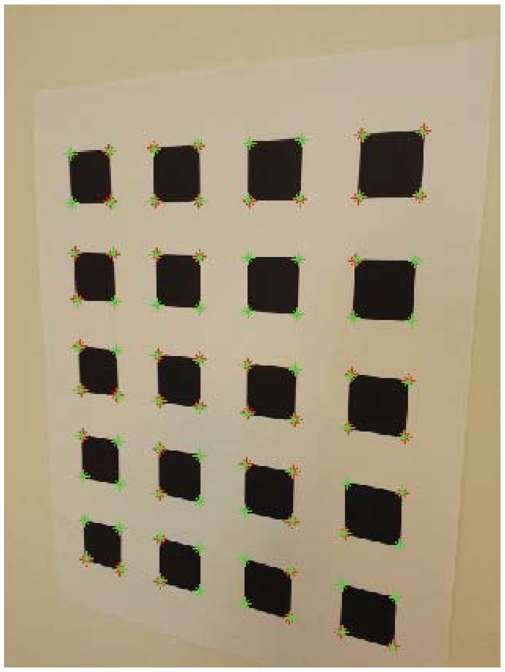
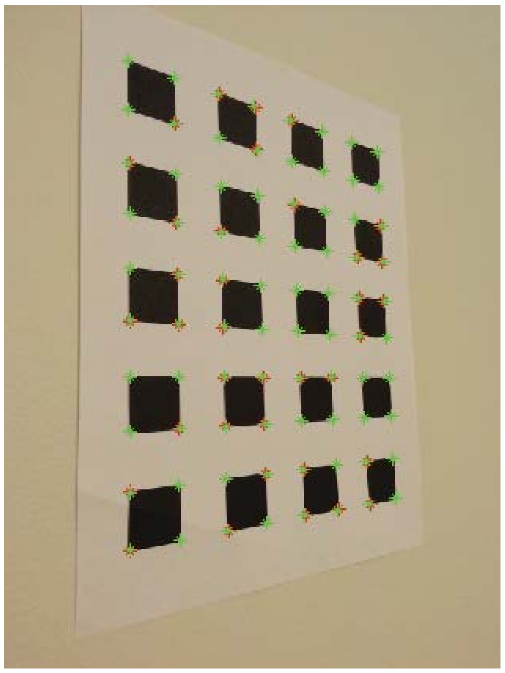

(d)

Figure 8: The corner points extracted from P1010001s.jpg are reprojected to (a) P1010003s.jpg, (b) P1010013s.jpg, (c) P1010055s.jpg, (d) P1010066s.jpg; red asteroids are originally detected corners and green asteroids are reprojected corners.

4.2 Self-recorded images


(a)


(b)


(c)

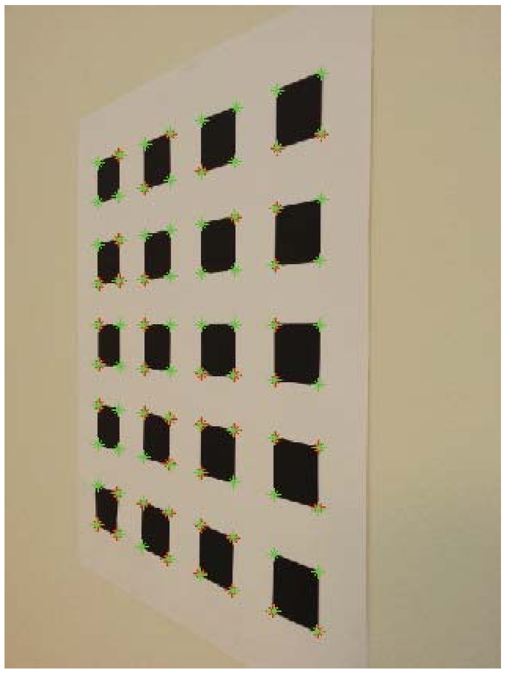

(d)

Figure 9: The corner points extracted from self01.jpg are reprojected to (a) self05.jpg, (b) self11.jpg, (c) self14.jpg, (d) self18.jpg; red asteroids are originally detected corners and green asteroids are reprojected corners.

5 Source Code

```
clear all; close all; clc;
imNum = '01020304050607081011121314151617181920474849505152535455565758596016626364656 6';
HOMO = cell(1,length(imNum)/2);  % Cell to store homographies
CON = cell(1,length(imNum)/2);   % Cell to store the coordinates of corners
FNAME = cell(1,length(imNum)/2);   % Cell to store file names

for Num = 1:length(imNum)/2;
    %% Image Load %%
    fname = ['P10100',imNum(2*Num-1:2*Num),'s.jpg'];
    FNAME{Num} = fname;
    im_RGB = imread(fname);                   % RGB color image
    im_gray = rgb2gray(im_RGB);               % Gray-scale image
    [M,N] = size(im_gray);                    % Size of image

    %% Canny Edge Detection and Hough Transform %%
    BW = edge(im_gray,'canny',0.7);     % Binary image from Canny detector
%     figure; imshow(BW);
    [H,T,R] = hough(BW);                      % Hough transform
    P = houghpeaks(H,18,'Threshold',1);
    lines = houghlines(BW,T,R,P,'FillGap',100,'MinLength',100);
    % Plot Hough lines
    x = 1:N;
    ab = zeros(length(lines),2);
%     figure; imshow(im_RGB); hold on;
    for k = 1:length(lines)
        delta = lines(k).point2 - lines(k).point1;
        a = delta(2)/delta(1);                % Slope of line
        if a == inf
%             plot(lines(k).point1(1)*ones(1,M),1:M,'g');
            b = inf;
        else
            b = lines(k).point1(2) - a*lines(k).point1(1);
%             plot(x,a*x+b,'g');
        end
        ab(k,1) = a;                          % Save line parameters a and b
        ab(k,2) = b;                          % where y = a*x + b
    end
%     hold off;

    %% Corner Detection and Label %%
    [S,I] = sort(abs(ab(:,1)));               % Find horizontal lines
    a_H = ab(I(1:10),1);                      % a of horizontal lines
    b_H = ab(I(1:10),2);                      % b of horizontal lines
    [b_H,I] = sort(b_H);
    a_H = a_H(I);
    Ico = corner(im_gray,80);                 % Corners in image plane
    label = zeros(80,1);                      % Initialize integer label
    for k = 1:10
        d2 = (a_H(k)*Ico(:,1)-Ico(:,2)+b_H(k)).^2/(a_H(k)^2+1);
        [d2,I] = sort(d2);
        I = I(1:8);
        [S,J] = sort(Ico(I,1));
        label(8*(k-1)+1:8*k) = I(J);
    end
    CON{Num} = Ico(label,:);
%     figure; imshow(im_RGB); hold on;
%     for k = 1:80
%         plot(Ico(label(k),1),Ico(label(k),2),'*r');
%         text(Ico(label(k),1)+3,Ico(label(k),2),num2str(k),'color','y');
%     end
%     hold off;

    %% Find Homograhy by LLS %%
    for k = 1:10
        Wco(8*(k-1)+1:8*k,1) = 0:24:168;    % Corners in world plane (mm)
        Wco(8*(k-1)+1:8*k,2) = 24*(k-1);
    end
    A = zeros(2*80,9);
    for k = 1:80
        A(2*k-1:2*k,:)=...
        [0,0,0,-[Wco(k,:),1],Ico(label(k),2)*[Wco(k,:),1];
        [Wco(k,:),1],0,0,0,-Ico(label(k),1)*[Wco(k,:),1]];
    end
    [U,D,V] = svd(A);
    h = V(:,9);     % Homography estimated by LLS with 80 correspondences
    HOMO{Num} = vec2mat(h,3);                 % Save homographies
    v12 = [h(1)*h(2),h(1)*h(5)+h(4)*h(2),h(4)*h(5),h(7)*h(2)+h(1)*h(8),...
           h(7)*h(5)+h(4)*h(8),h(7)*h(8)];
    v11 = [h(1)*h(1),h(1)*h(4)+h(4)*h(1),h(4)*h(4),h(7)*h(1)+h(1)*h(7),...
           h(7)*h(4)+h(4)*h(7),h(7)*h(7)];
    v22 = [h(2)*h(2),h(2)*h(5)+h(5)*h(2),h(5)*h(5),h(8)*h(2)+h(2)*h(8),...
           h(8)*h(5)+h(5)*h(8),h(8)*h(8)];
    B(2*Num-1:2*Num,:) = [v12;v11-v22];
end
[U,D,V] = svd(B);
v = V(:,6);
B = [v(1),v(2),v(4);v(2),v(3),v(5);v(4),v(5),v(6)];

%% Compute Intrinsic Parameters in K %%
y0 = (v(2)*v(4)-v(1)*v(5))/(v(1)*v(3)-v(2)^2);          % Principle point y
lambda = v(6)-(v(4)^2+y0*(v(2)*v(4)-v(1)*v(5)))/v(1);
a_x = sqrt(lambda/v(1));                                 % Scale factor in x
a_y = sqrt(lambda*v(1)/(v(1)*v(3)-v(2)^2));              % Scale factor in y
s = -v(2)*a_x^2*a_y/lambda;                              % Skewness factor
x0 = s*y0/a_y-v(4)*a_x^2/lambda;                         % Principle point x
K = [a_x s x0; 0 a_y y0; 0 0 1];            % Intrinsic parameter matrix K

%% Compute Extrinsic Parameters %%
Rt = cell(1,Num);       % Cell to store extrinsic parameters for each image
P = cell(1,Num);        % Cell to store camera matrix for each image
for k = 1:Num
    H = HOMO{k};
    r1 = lambda*K\H(:,1);
    r2 = lambda*K\H(:,2);
    r3 = cross(r1,r2);
    t = lambda*K\H(:,3);
    R = [r1,r2,r3];
%     phi = acos((trace(R)-1)/2);
%     w = phi*[R(3,2)-R(2,3);R(1,3)-R(3,1);R(2,1)-R(1,2)]/(2*sin(phi));
%     % Optimization by Levenberg-Marquardt method
%     opt = optimset('Algorithm','levenberg-marquardt');
%     p = [a_x,s,x0,a_y,y0,w(1),w(2),w(3),t(1),t(2),t(3)];
%     q = lsqcurvefit(@fun,p,Wco,CON{k},[],[],opt);     % Refined parameters
%     K = [q(1) q(2) q(3); 0 q(4) q(5); 0 0 1];
%     R = exp([0,-q(8),q(7);q(8),0,-q(6);-q(7),q(6),0]);
%     t = [q(9),q(10),q(11)]';
    Rt{k} = [R,t];                          % Extrinsic parameter matrix [R|t]
    P{k} = K*Rt{k};
end

%% Test Accuracy by Re-projecting Corner Points %%
n1 = 1;                     % Image Number having corners to be reprojected
n2 = 3;                     % Image Number which the corners reprojected to
im = imread(FNAME{n2});
figure; imshow(im); hold on;
for k = 1:80
    plot(CON{n2}(k,1),CON{n2}(k,2),'*r');
    X = P{n1}'/(P{n1}*P{n1}')*[CON{n1}(k,:),1]';
    x = P{n2}*X;
    plot(x(1)/x(3),x(2)/x(3),'*g');
end
hold off;
```

Function fun( ) describing the camera projection

```
function [ F ] = fun( p,X )
% This function describes the camera projection
% from the world plane to the image plane.
% This function is used for the Levenberg-Marquardt method.

K = [p(1) p(2) p(3); 0 p(4) p(5); 0 0 1];        % Intrinsic parameter matrix K
w = [p(6) p(7) p(8)]';
t = [p(9) p(10) p(11)]';
Wx = [0,-w(3),w(2);w(3),0,-w(1);-w(2),w(1),0];
phi = sqrt(w'*w);
R = 1 + Wx*sin(phi)/phi + (Wx.^2)*(1-cos(phi))/phi;
P = K * [R t];
[M,N] = size(X);
F = zeros(M,N);
for k = 1:M
    Y = P * [X(k,:),0,1]';
    F(k,1) = Y(1)/Y(3);
    F(k,2) = Y(2)/Y(3);
end

end
```