CMSC 828D: Fundamentals of Computer Vision
Homework 7

Instructors: Larry Davis, Ramani Duraiswami,
Daniel DeMenthon, and Yiannis Aloimonos

Solution based on homework submitted by Haiying Liu

1. **Write a Matlab function that outputs the homogeneous coordinates of the 12 lines that are the images of the edges of the cube (these are lines in the image plane. Refer to slide 11 of the class on Projective Geometry, and use the Matlab function cross).**

Solution: Please see appendix for Matlab script and results.

2. **Find the homogeneous coordinates of the 3 vanishing points of the image of the cube. These are the intersections of the image lines corresponding to parallel edges of the cube (refer to slide 11 of Projective Geometry again for a method for finding intersections between lines. Refer to slide 36 of class 3 on cameras for a review on vanishing points).**

Solution: Please see appendix for Matlab script and results.

3. **Each vanishing point is the image of a point at infinity of the form (d, 0), where d is a Euclidean vector with 3 coordinates expressing the direction of a cube edge. Show that the coordinates of a vanishing point v can be expressed as v = K R d, where K is the calibration matrix and R is the rotation matrix between the camera and world coordinate system (use slides of Calibration class).**

Solution: Note that a group of parallel lines (parallel edges in our problem) intersects at one point at infinite. Without loss of generality, we select or construct one of the lines, denoted by l_0, that goes through the origin of world coordinate system. Assume the angle between l_0 and x, y, z axis's are a, b, g respectfully. Then any point on l_0 can be expressed as (r, a, b, g) in polar coordinates, $(r\cos a, r\cos b, r\cos g)$ in Euclidean coordinate, or $(\cos a, \cos b, \cos g, r)$ in homogeneous coordinates. When $r \to 0$, l_0 reaches the infinite point. In class, we already derived the relationship between a point in world and its image. Use the notation in class and apply the relationship to a point $(\cos a, \cos b, \cos g, r)$ on l_0, we have:

$$
\begin{aligned}
v &= \lim_{r\to 0} \mathbf{K}\left[\mathbf{I}_3 \,|\, \mathbf{0}_3\right] \begin{bmatrix} \mathbf{R} & \mathbf{T} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \cos a \\ \cos b \\ \cos g \\ r \end{bmatrix} \\
&= \mathbf{K}\left[\mathbf{I}_3 \,|\, \mathbf{0}_3\right] \begin{bmatrix} \mathbf{R} & \mathbf{T} \\ 0 & \mathbf{1} \end{bmatrix} \begin{bmatrix} \cos a \\ \cos b \\ \cos g \\ 0 \end{bmatrix} \\
&= \mathbf{K}\left[\mathbf{I}_3 \,|\, \mathbf{0}_3\right] \begin{bmatrix} \mathbf{Rd} \\ 0 \end{bmatrix} \\
&= \mathbf{KI}_3\mathbf{Rd} \\
&= \mathbf{KRd}
\end{aligned}
$$

where $d = [\cos a, \cos b, \cos g]^{\mathrm{T}}$ is the Euclidean vector with 3 coordinates expressing the direction of l_0.

4. Express an edge direction d as a function of K, R and v.

Solution: In class, we already derive that the $\mathbf{K}$ is in the form:

$$\mathbf{K} = \begin{bmatrix} fk_u & fk_u \cot q & u_0 \\ 0 & fk_v / \sin q & v_0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Since $|\mathbf{K}| = f^2 k_u k_v / \sin q \neq 0$, $\mathbf{K}^{-1}$ exists. Note that any rotation can be decomposed as combination of three single rotations around x, y, z axis's respectfully, i.e.

$$\mathbf{R} = \mathbf{R}_x\mathbf{R}_y\mathbf{R}_z = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos q_x & -\sin q_x \\ 0 & \sin q_x & \cos q_x \end{bmatrix} \begin{bmatrix} \cos q_y & 0 & -\sin q_y \\ 0 & 1 & 0 \\ \sin q_y & 0 & \cos q_y \end{bmatrix} \begin{bmatrix} \cos q_z & -\sin q_z & 0 \\ \sin q_z & \cos q_z & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Because $|\mathbf{R}_i| = \cos q_i^{\,2} + \sin q_i^{\,2} = 1, i = x, y, z$, $|\mathbf{R}| = |\mathbf{R}_x\mathbf{R}_y\mathbf{R}_z| = |\mathbf{R}_x||\mathbf{R}_y||\mathbf{R}_z| = 1 \neq 0$, $\mathbf{R}^{-1}$ exists. Therefore, we can express $\mathbf{d}$ as $\mathbf{d} = \mathbf{R}^{-1}\mathbf{K}^{-1}\mathbf{v}$.

5. The 3 directions of cube edges that give rise to the 3 vanishing points are mutually perpendicular, therefore the dot product between two directions is zero. Show that this condition leads to an equation in which the unknown is the calibration matrix K. Such an equation can be written for each of the 3 pairs of vanishing points. Note: this equation expresses that the vanishing points belong to a conic that is the image of the so-called absolute conic.

Solution: From last problem, we have $\mathbf{d} = \mathbf{R}^{-1}\mathbf{K}^{-1}\mathbf{v} \Rightarrow \mathbf{Rd} = \mathbf{K}^{-1}\mathbf{v}$. Note that rotation matrix $\mathbf{R}$ is a unitary matrix, i.e. $\mathbf{R}^{\mathrm{T}}\mathbf{R} = \mathbf{I}$, we have,

$$\begin{aligned}
\mathbf{d}_i \cdot \mathbf{d}_j &= 0 \Rightarrow \\
\mathbf{d}_i^{\mathrm{T}} \mathbf{d}_j &= 0 \Rightarrow \\
\mathbf{d}_i^{\mathrm{T}} \left(\mathbf{R}^{\mathrm{T}} \mathbf{R}\right) \mathbf{d}_j &= 0 \Rightarrow \\
\left(\mathbf{R}\mathbf{d}_i\right)^{\mathrm{T}} \left(\mathbf{R}\mathbf{d}_j\right) &= 0 \Rightarrow \\
\left(\mathbf{R}\mathbf{d}_i\right) \cdot \left(\mathbf{R}\mathbf{d}_j\right) &= 0 \Rightarrow \\
\left(\mathbf{K}^{-1}\mathbf{v}_i\right) \cdot \left(\mathbf{K}^{-1}\mathbf{v}_j\right) &= 0
\end{aligned}$$

Equation 1

Where notation "$\mathbf{a} \cdot \mathbf{b}$" means dot product of vectors $\mathbf{a}$ and $\mathbf{b}$. This is an equation (relationship) for each of the three pairs of vanishing points in image $i \neq j$.

6. **The equation just found leads to nonlinear conditions between the elements of the matrix K, so we will not attempt to solve the system, but only verify that the matrix K found last week indeed is a solution. Verify that the equation above is verified for the vanishing points found in (2), for a calibration matrix in which the skew is zero, the 2 focal lengths are equal to 690, and the image center is at (300, 250).**

Solution: Given $fk_u = fk_v = 690$, $(u_0, v_0) = (300, 250)$, and skew is zero, i.e. $q = 90°$, we have:

$$\mathbf{K} = \begin{bmatrix} fk_u & fk_u \cot q & u_0 \\ 0 & fk_v / \sin q & v_0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 690 & 0 & 300 \\ 0 & 690 & 250 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{K}^{-1} = \begin{bmatrix} \dfrac{1}{fk_u} & -\dfrac{\cot q \sin q}{fk_v} & \dfrac{k_u v_0 \cot q \sin q - k_v u_0}{fk_u k_v} \\ 0 & \dfrac{\sin q}{fk_v} & -\dfrac{v_0 \sin q}{fk_v} \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0.0014 & 0 & -0.4348 \\ 0 & 0.0014 & -0.3623 \\ 0 & 0 & 1.0000 \end{bmatrix}$$

Please see Matlab script in appendix for detail verification. From the experiment, we verified that the calibration matrix defined by $\mathbf{K}$ satisfies [Equation 1] with error tolerance less than 10^{-8} .

Appendix:

- **hw7.m**:

```
function hw7
% Syntax: hw7
%
% Description: CMSC828D HW7
%
% Author: Haiying Liu
%   Date: Oct. 12, 2000
%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

dbstop if error

msg = nargchk(0, 0, nargin);
if (~isempty(msg))
  error(strcat('ERROR:', msg));
end

clear msg;

%======================================================================
%= Turn on the diary to save the result.

diary off;

filename = 'hw7.txt';
if (exist(filename, 'file'))
  delete(filename);
end

eval(['diary ', filename]);

disp(' ');
disp('» hw7');

%======================================================================
%= Initialization.

world_coord = [ ...
     2,  2,  2; ...
    -2,  2,  2; ...
    -2,  2, -2; ...
     2,  2, -2; ...
     2, -2,  2; ...
    -2, -2,  2; ...
    -2, -2, -2; ...
     2, -2, -2; ...
  ];

image_coord = [ ...
    422, 323; ... % m1
    178, 323; ... % m2
    118, 483; ... % m3
    482, 483; ... % m4
    438,  73; ... % m5
    162,  73; ... % m6
     78, 117; ... % m7
    522, 117; ... % m8
  ];

%=====================================================================
%= 1. Write a Matlab function that outputs the homogeneous coordinates
%=    of the 12 lines that are the images of the edges of the cube
%=    (these are lines in the image plane. Refer to slide 11 of the
%=    class on Projective Geometry, and use the Matlab function cross).

% Note that a line going through two points m1 and m2 is represented
% by the cross-product m1 x m2. Any point x on the line satisfies
% x'(m1 x m2) = 0.

disp(' ');
disp(':::::::::::::');
disp(':: Part 1. ::');
disp(':::::::::::::');
disp(' ');

% Define a connectivity of the eight points.
%  m1   m2   m3   m4   m5   m6   m7   m8
connect = [ ...
    1, 2; ...
    4, 3; ...
    5, 6; ...
    8, 7; ... % group 1
    1, 4; ...
    2, 3; ...
    5, 8; ...
    6, 7; ... % group 2
    1, 5; ...
    2, 6; ...
    3, 7; ...
    4, 8; ... % group 3
  ];

% Compute edges.
disp('Twelve edges:');
disp(' ');

nEdges = size(connect, 1);

for index = 1:nEdges
  ptIndex_i      = connect(index, 1);
  ptIndex_j      = connect(index, 2);
  mi             = [image_coord(ptIndex_i, :), 1];
  mj             = [image_coord(ptIndex_j, :), 1];
  edge(index, :) = cross(mi, mj);

  disp(['m', num2str(ptIndex_i), '~m', num2str(ptIndex_j), ...
      ': ', num2str(edge(index, :))]);
end
```

```
%======================================================================
%= 2. Find the homogeneous coordinates of the 3 vanishing points of
%= the image of the cube. These are the intersections of the image
%= lines corresponding to parallel edges of the cube (refer to slide
%= 11 of Projective Geometry again for a method for finding
%= intersections between lines. Refer to slide 36 of class 3 on
%= cameras for a review on vanishing points).

disp(' ');
disp(':::::::::::::');
disp(':: Part 2. ::');
disp(':::::::::::::');
disp(' ');

% The three group of parallel lines are:
% m1~m2, m3~m4, m5~m6, m7~m8;
% m1~m4, m2~m3, m5~m8, m6~m7;
% m1~m5, m2~m6, m3~m7, m4~m8;

% Note the intersect of two lines L1 and L2 is L1 x L2.

% Calculate the mean of vanishing point for each group of
% parallel lines.
ptIndex     = 0;
vanishPoint = zeros(3);
for index = 1:4:nEdges - 3
  % Form matrix L for L.p = 0
  L = zeros(4, 3);
  for row = index:index + 3
    L(row - index + 1, :) = edge(row, :);
  end

  % Solve L.p = 0 by DLT
  [U, S, V] = svd(L);
  nCol_V = size(V, 2);
  ptIndex   = ptIndex + 1;
  vanishPoint(ptIndex, :) = V(:, nCol_V)';
end

vanishPoint

%======================================================================
%= 6. The equation just found leads to nonlinear conditions between
%= the elements of the matrix K, so we will not attempt to solve the
%= system, but only verify that the matrix K found last week indeed is
%= a solution.  Verify that the equation above is verified for the
%= vanishing points found in (2), for a calibration matrix in which
%= the skew is zero, the 2 focal lengths are equal to 690, and the
%= image center is at (300, 250).

disp(' ');
disp(':::::::::::::');
disp(':: Part 2. ::');
disp(':::::::::::::');

K = [ ...
    690   0     300; ...
    0     690   250; ...
    0     0     1  ; ...
  ]

K_inv = inv(K);

nVanPt = size(vanishPoint, 1);
for idx1 = 1:nVanPt - 1
  for idx2 = idx1 + 1:nVanPt
    vi   = vanishPoint(idx1, :);
    vj   = vanishPoint(idx2, :);
    ri   = K_inv * vi';
    rj   = K_inv * vj';

    disp('-----------');
    disp(['r', num2str(idx1), ' = inv(K) * v', num2str(idx1), ...
        ' = [', num2str(ri'), ']''']);
    disp(['r', num2str(idx2), ' = inv(K) * v', ...
        num2str(idx2), ' = [', num2str(rj'), ']''']);
    disp(['r', num2str(idx1), ' . r', num2str(idx2), ...
      ' = ', num2str(ri' * rj)]);
  end
end

disp(' ');

%=====================================================================
%= Stop recording

diary off;
```

- **Result**:

```
» hw7

:::::::::::::
:: Part 1. ::
:::::::::::::

Twelve edges:

m1~m2: 0   -244  78812
m4~m3: 0    -364  175812
m5~m6: 0   -276  20148
m8~m7: 0   -444  51948
m1~m4: -160     60  48140
m2~m3: -160    -60  47860
m5~m8: -44     84  13140
m6~m7: -44    -84  13260
m1~m5: 250      16 -110668
m2~m6: 250    -16 -39332
m3~m7: 366    -40 -23868
m4~m8: 366      40 -195732
```

```
:::::::::::::
:: Part 2. ::
:::::::::::::


vanishPoint =

    1.0000         0         0
   -1.0000    0.0011   -0.0033
    0.1376    0.9905    0.0005


:::::::::::::
:: Part 2. ::
:::::::::::::

K =

   690     0   300
     0   690   250
     0     0     1

-----------
r1 = inv(K) * v1 = [0.0014493           0           0]'
r2 = inv(K) * v2 = [-1.2947e-008   0.0012092  -0.0033333]'
r1 . r2 = -1.8764e-011
-----------
r1 = inv(K) * v1 = [0.0014493           0           0]'
r3 = inv(K) * v3 = [3.0874e-010   0.0012693   0.0004586]'
r1 . r3 = 4.4745e-013
-----------
r2 = inv(K) * v2 = [-1.2947e-008   0.0012092  -0.0033333]'
r3 = inv(K) * v3 = [3.0874e-010   0.0012693   0.0004586]'
r2 . r3 = 6.2837e-009
```