CS 223b: Introduction to Computer Vision
Assignment #2 Solutions

1 Calibration with MATLAB

Here's an example solution:

```
function [R, T, fx, ox, oy, alpha] = calibrate(pts3,pts2)

A = [pts3 pts2];
[rows,cols] = size(A);
%
% set up as a least squares problem
%
% the row pair are one row at this point
%
        aa = [ A(:,1) A(:,2) A(:,3) ones(rows,1) zeros(rows,4)          ...
               -A(:,4).*A(:,1) -A(:,4).*A(:,2) -A(:,4).*A(:,3)          ...
                zeros(rows,4) A(:,1) A(:,2) A(:,3) ones(rows,1)         ...
               -A(:,5).*A(:,1) -A(:,5).*A(:,2) -A(:,5).*A(:,3)];
%
% reshape the matrix, so that the rows interleave
%
        aa = reshape(aa',11, rows*2)';

        bb = reshape( [A(:,4) A(:,5)]', 1, rows*2)';

        C = aa\bb;        % least squares solution
        resid = max(max(abs(aa * C - bb)));
        fprintf('maxm residual %f pixels.\n', resid);
        C = reshape([C;1]',4,3)';

q3 = C(3,1:3);
gamma = sqrt(sum(q3.*q3));
Cn = C/gamma;
```

```
q1 = Cn(1,1:3);
q2 = Cn(2,1:3);
q3 = Cn(3,1:3);
q4 = [Cn(1,4), Cn(2,4), Cn(3,4)];

% origin of world ref frame is in front of Camera
%  so Tz > 0 which means that
sigma = 1;
if Cn(3,4) < 0
        sigma = -1;
end
fprintf(1, 'sigma = %d\n', sigma);
Tz = sigma*Cn(3,4);

R=zeros(3,3);
for i=1:3
  R(3,i) = sigma*Cn(3,i);
end

ox = dot(q1,q3);
oy = dot(q2,q3);

fx = sqrt(dot(q1,q1) - ox*ox);
fy = sqrt(dot(q2,q2) - oy*oy);

for i=1:3
  R(1,i) = sigma*(ox*Cn(3,i) - Cn(1,i))/fx;
  R(2,i) = sigma*(oy*Cn(3,i) - Cn(2,i))/fy;
end

Tx = sigma*(ox*Tz - Cn(1,4))/fx;
Ty = sigma*(oy*Tz - Cn(2,4))/fy;

alpha = fx/fy;
T = [Tx Ty Tz];
```

2 Short Answer:

1. Since the median operator is insensitive to outliers it does a much better job of handling impulse noise. For example, consider a 1D sequence $[12, 13, 12, 11, 255]$. The mean of this

sequence will be 60.6 while the median will be 12, a much more reasonable value.

2. Hysteresis thresholding is important in connecting up strong edges with weaker ones. This allows for the recovery of complete edges as opposed to ones with many small discontinuities. Non-maximal suppression is useful for distilling edges down to their strongest, thinnest (ideally 1 pixel wide) representations. This procedure makes for sharper edges in the final image.

3. A ball is definitely better. The idea here is that any projection of a sphere onto a plane will be a circle. So, any image of a ball should ideally look like a circle. To the extent that such an image doesn't, the cause is likely a consequence of the aspect ratio of the camera. A circle on a plane could also work, but pain must be taken to insure that this pattern is orthogonal to the optical axis. The ball should be placed as close to the image center as possible, where lens distortion is at a minimum.