ASSIGNMENT A1

Shantnu Kakkar

CS 5320/6320, Spring 2016

January 27, 2016

Section 1: Intro:

This assignment is based on the Perspective Camera Projection Model learnt in the lecture. This model is based the perspective projection equation:

$$p = \frac{1}{Z} MP$$

Where,

- p is the image coordinates in Camera's reference frame. It's dimension is 3*1
- Z is the z-coordinate of the image in the Camera's Camera's reference frame. It's dimension is 3*1
- M is K(R t), where K is the internal calibration matrix. R and t are the rotation and translation matrices that relate Camera and World Frame
- M is composed of 5 intrinsic factors alpha, beta, theta, x0 and y0.
- alpha and beta are magnifications defined as alpha = k*f and beta = l*f expressed in pixel units, where $\frac{1}{k}$ *and* $\frac{1}{l}$ represent pixel dimensions in pixel/m
- x0 and y0 define the position of the image center in the retinal coordinate system. That means it is the coordinates of c_o in Figure1 below
- theta is the skew angle in the camera coordinate system that may arise due to manufacturing defect

Figure 1 shows the various parameters described above:

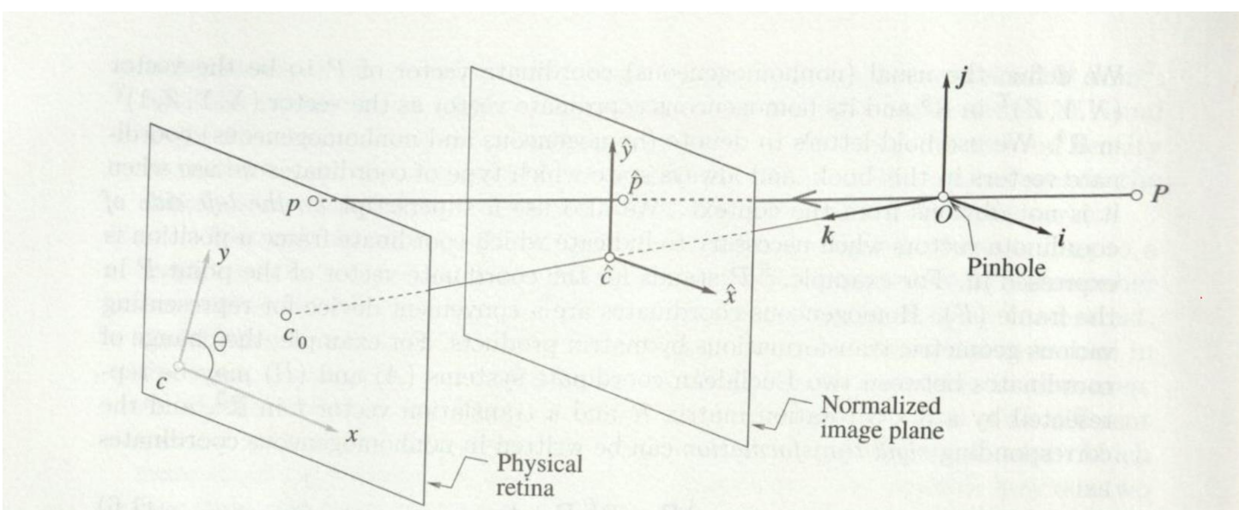


Figure 1 Physical and normalized image coordinate systems

The following question is going to be answered in this report:

1. How does the error in image point location change in terms of change in the various intrinsic parameters?

Section 2: Method:

Matlab would be used to carry out the experiments.

We would be doing two experiments by observing the following two plots:

- Trial Number (from 1 to 100) v/s Mean error for each trial
- Variance in the parameter v/s mean error in the point locations in the image

Following functions are made to implement the Perspective projection model:

- CS5320_camera(WorldPoints,alpha,beta,theta,x0,y0,R,t) - produces camera model perspective projection. It takes in input various intrinsic (alpha,beta,theta,x0,y0) and extrinsic parameters and the World points (example world points of a cube or a sphere), and outputs the image points corresponding to those world points.
- CS5320_gen_sphere(C,radius,del_x,del_p) - generate 3D points on the surface of a sphere, with center C, radius radius, del_x step size along major axis to generate points, and del_ p distance between points on the sphere. It outputs a sphere (4xk array): homogeneous coordinates for points on the sphere
- CS5320_gen_cube(C,del_x,S) - generate 3D points on the edges of a cube. C (3x1 vector): center of sphere, del_x (float): step size along edges to generate point, S (float): length of side of cube. It outputs cube (4xk array): homogeneous coordinates for points on the cube
- CS5320_gen_R(u,theta) - generate a rotation matrix about an arbitrary vector. It takes in input as unit vector about which to rotate (u) and theat as amount to rotate (in radians)
- CS5320_question_answer_plot1() - This function is used to plot Trial Number (from 1 to 100) v/s Mean error for each trial, and Variance in the parameter v/s mean error in the point locations in the image. It outputs error vector for number 100 trials, mean of error, variance of error and confidence interval of error. The parameter to be varied is changed inside the function itself. This function also computes mean error , variance or error and confidence interval of error. No inputs are required to give to this function as the value of intrinsic and extrinsic parameter are set inside the function itself.
- CS5320_question_answer_plot1() - This function is used to plot Variance in the parameter v/s mean error in the point locations in the image. The parameter to be varied is changed inside the function itself. This function computes mean error vector, variance vector and confidence interval low and high vectors of error for each variance in index ranging from 0.1 to 1 with increment of 0.1. No inputs are required to give to this function as the value of intrinsic and extrinsic parameter are set inside the function itself.

The following two algorithms are used for getting above two plots respectively:

PseudoCode to get Trial Number (from 1 to 100) v/s Mean error for each trial

1. Set a baseline value for all parameters and get the baseline image; e.g.: alpha = 1, beta = 1; theta = pi/2, x0 = 0, y0 = 0, R = eye(3,3), t = [0;0;0]

   im = CS5320_camera(WorldPoints,alpha,beta,theta,x0,y0,R,t);
2. pick a noise level based on the Normal distribution (because variance is directly related to noise). e.g., use the standard normal distribution N(0,1).
3. Pick the parameter to vary and number of trials to run
```
for t = 1:100
    alphan = alpha + randn;
    im_a = CS5320_camera(WorldPoints,alphan,beta,theta,x0,y0,R,t);
    error(t) = average error in locations of points;
 end
```
4. Calculate the mean, variance and confidence interval of the error:

5. Do 3 and 4 for all the intrinsic parameters

PseudoCode to get Variance in the parameter v/s mean error in the point locations in the image

1. Set a baseline value for all parameters and get the baseline image; e.g.: alpha = 1, beta = 1; theta = pi/2, x0 = 0, y0 = 0, R = eye(3,3), t = [0;0;0]

   im = CS5320_camera(WorldPoints,alpha,beta,theta,x0,y0,R,t);
2. pick a noise level based on the Normal distribution (because variance is directly related to noise). e.g., use the standard normal distribution N(0,1).
3. Pick the parameter to vary
```
vs = [0.1:0.1:1];
num_vs = length(vs);
for v_index = 1:num_vs  % set variance
   v = vs(v_index);
   for t = 1:100   % number of trials
      alphan = alpha + sqrt(v)*randn;
      im_a = CS5320_camera(pts,alphan,beta,theta,x0,y0,R,t);
      % calculate mean error in pt location, etc.
      errors(t) = ...
   end
   means(v_index) = mean(errors);
   vars(v_index) = var(errors);
    ci_dn(v_index) = means(v_index) - 1.66*sqrt(vars(v_index)/100);
    ci_up(v_index) = means(v_index) + 1.66*sqrt(vars(v_index)/100);
end
```
4. Do 3 for all the intrinsic parameters

Section 3: Verification:

The following figure validates that my code work. Also, the videos A1_trans and A1_rotate validate that my rode is running fine.

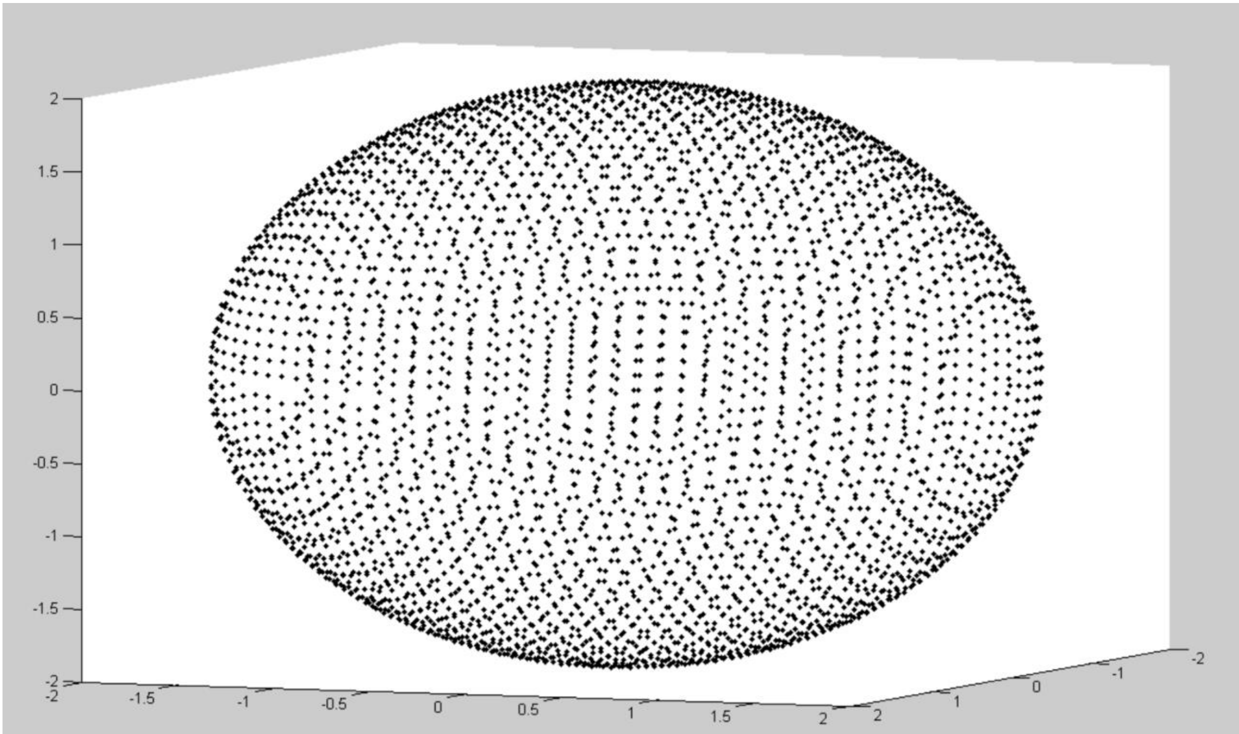


Figure 2 Getting a good sphere on call of CS5320_gen_sphere([0;0;0],2,0.1,0.1);

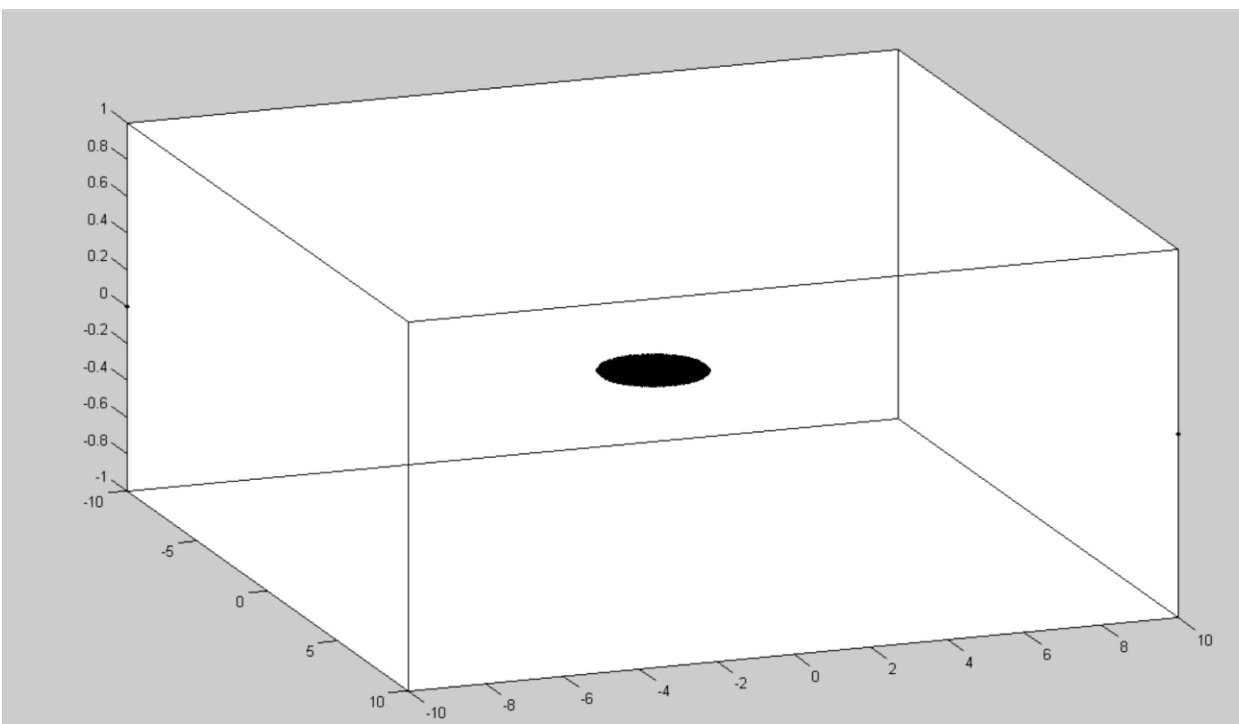


Figure 3 Getting correct 2-d figure (that is a circle) of the sphere generated in Figure 2 using = CS5320_camera function

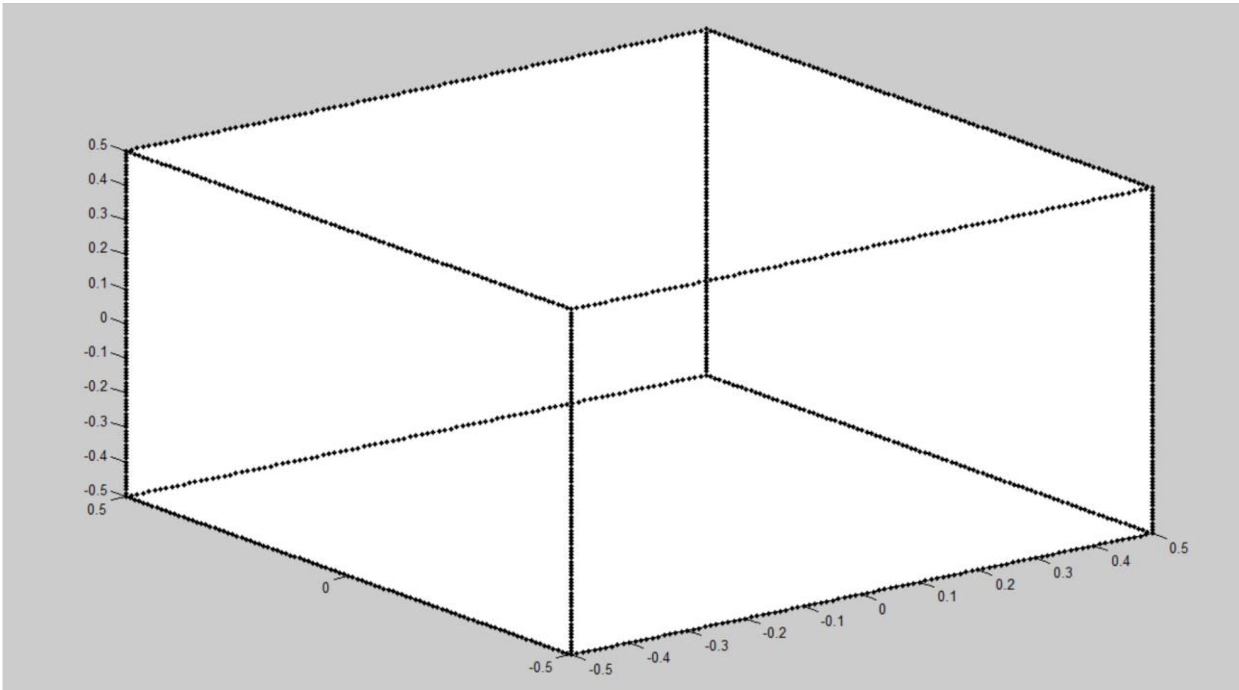


Figure 4 Getting correct cube on call of cube = CS5320_gen_cube([0;0;0],0.01,1);

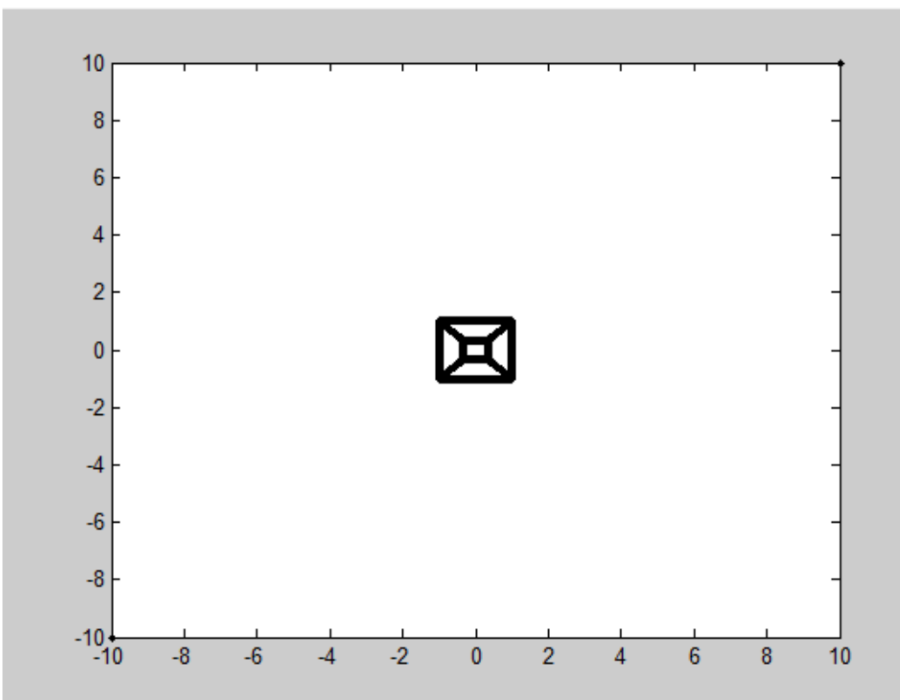


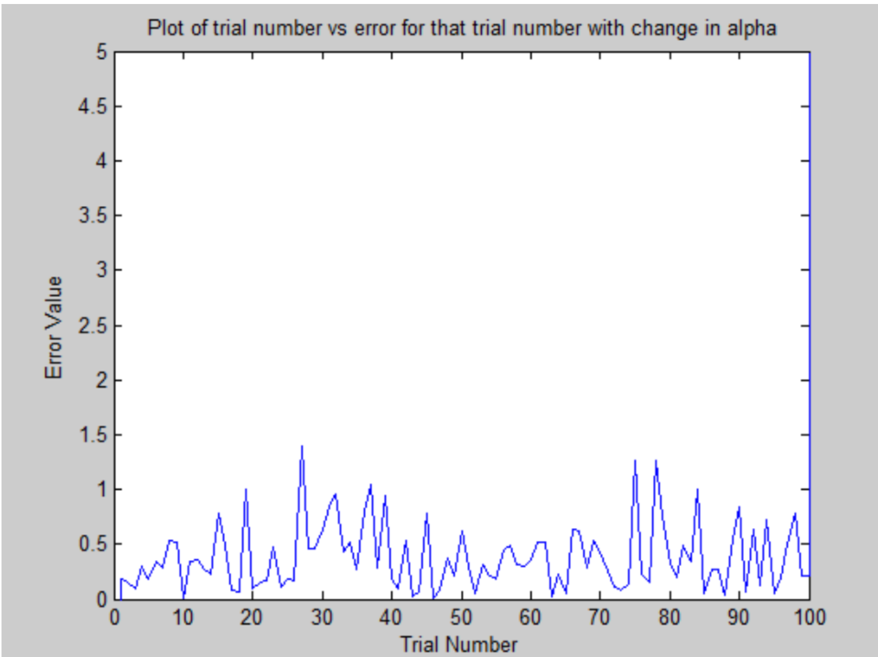
Figure 5 Getting image of cube in figure 4 correctly on call of im = CS5320_camera(cube,1,1,pi/2,0,0,eye(3,3),[0;0;1]);

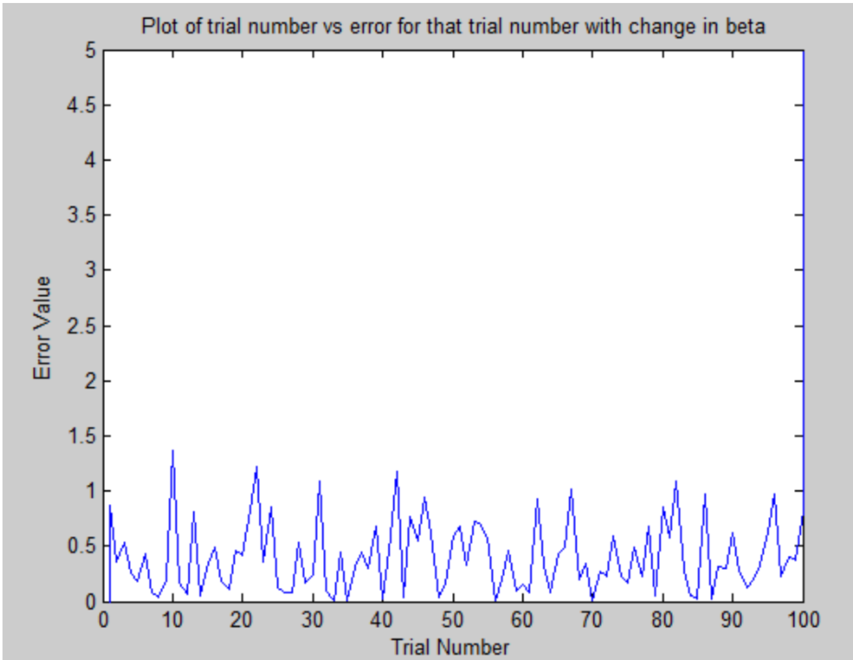
- a call to CS5320_gen_R with arguments u = [0;0;1] and theta = 0 is giving me the identity matrix.
- I am not able to see my image point If Z is positive, which proves that my CS5320_camera correctly recognies the positive and the negative sides of the camera.

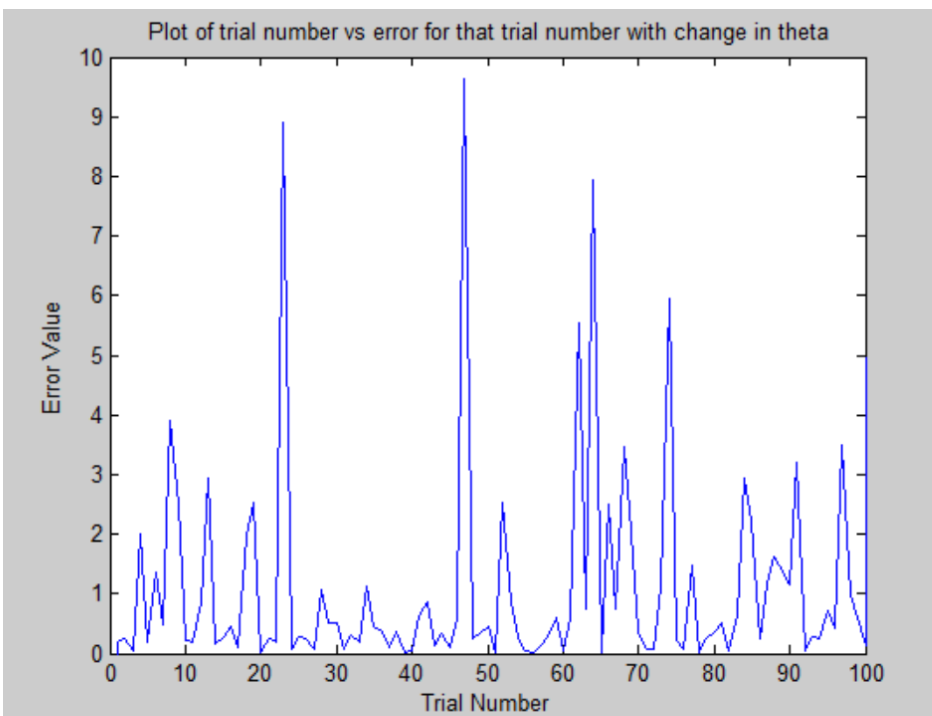
Section 4: Data:

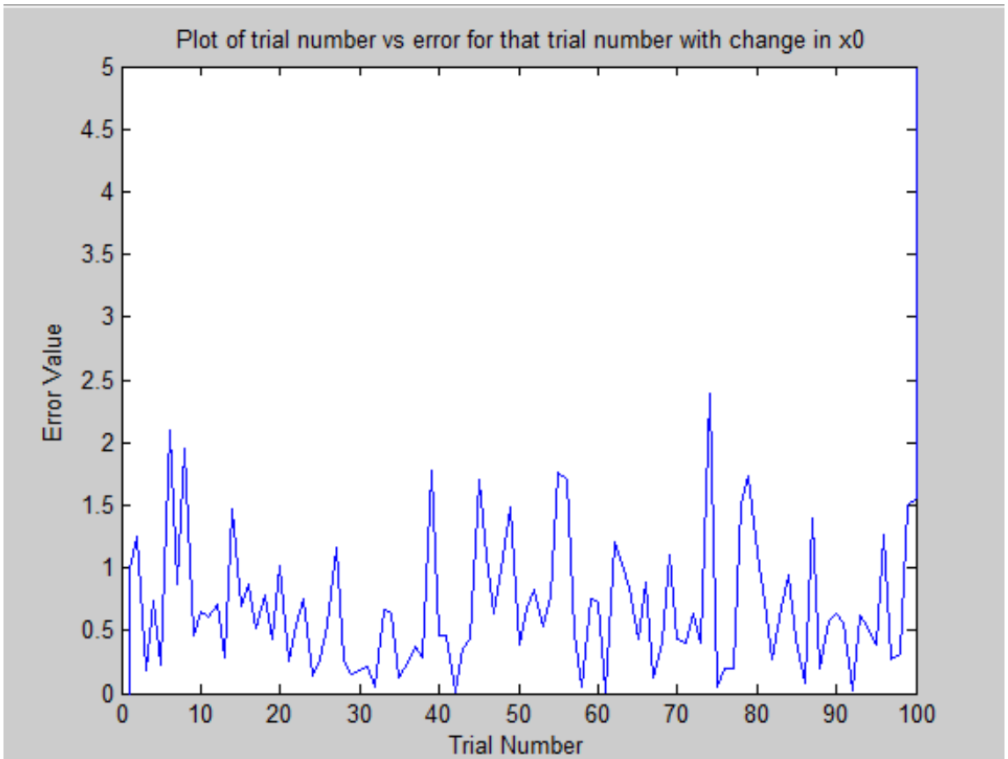
Figure 2 and 4 show the world coordinate points of a sphere and a cube respectively, that go into camera function. Figure 3 and 5 show the respective output.

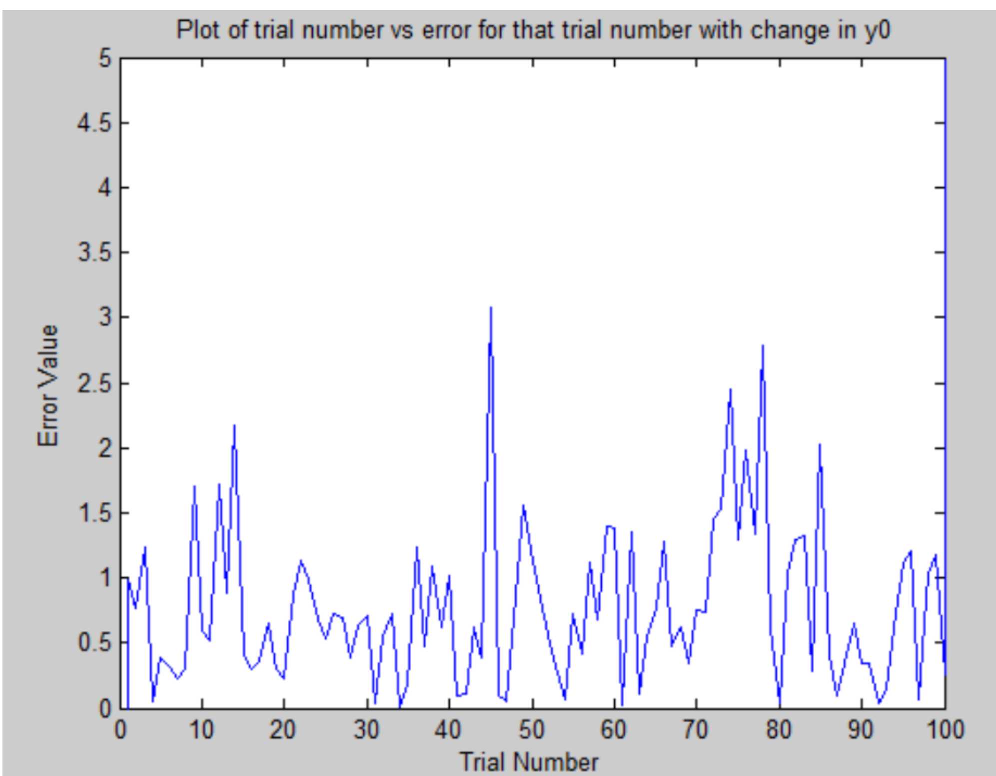
EXPERIMENT 1: The following figure show plots for 100 trials with change in various intrinsic parameters. Parameter name is there in the title



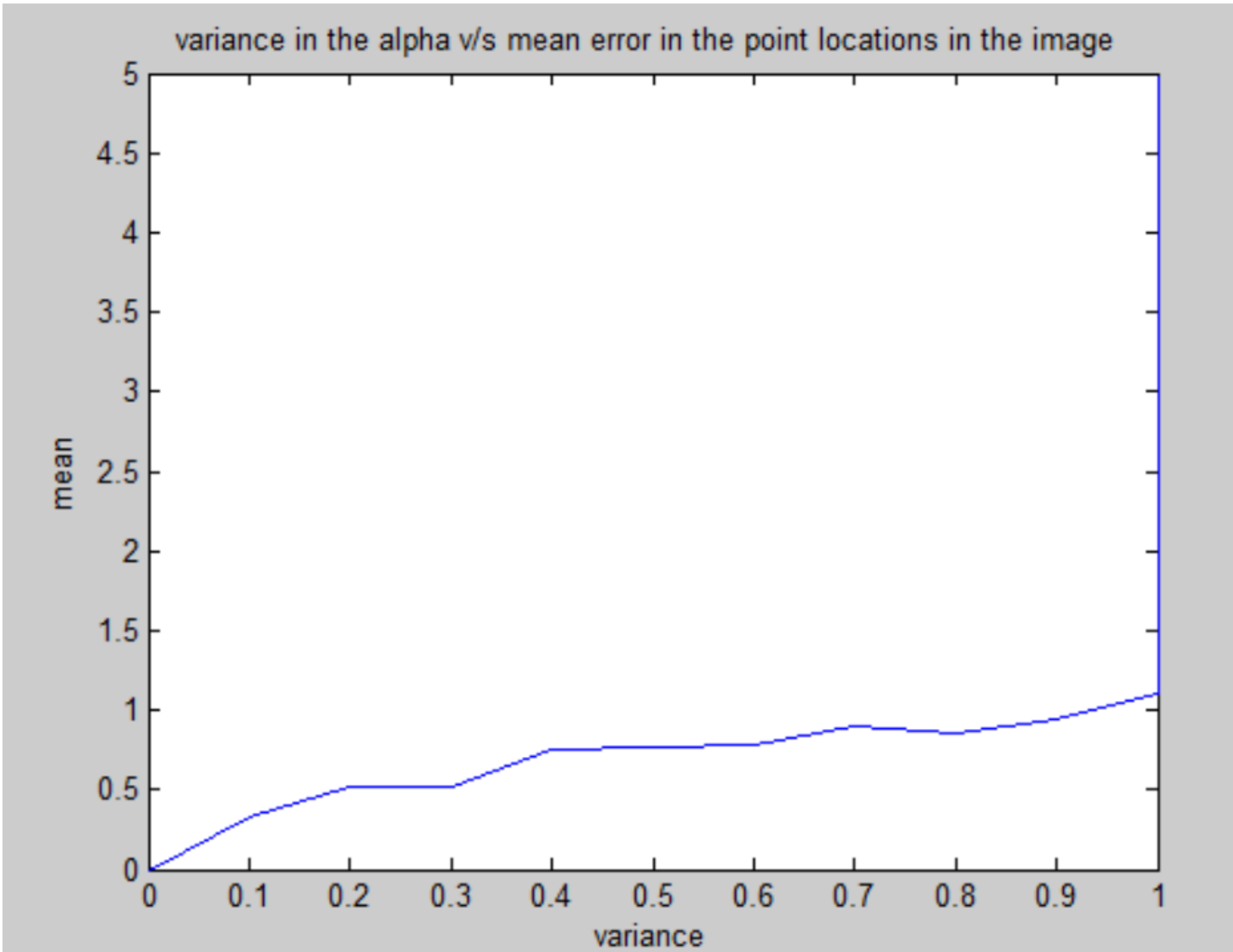

Plot of trial number vs error for that trial number with change in beta
Error Value
Trial Number
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
10
20
30
40
50
60
70
80
90
100

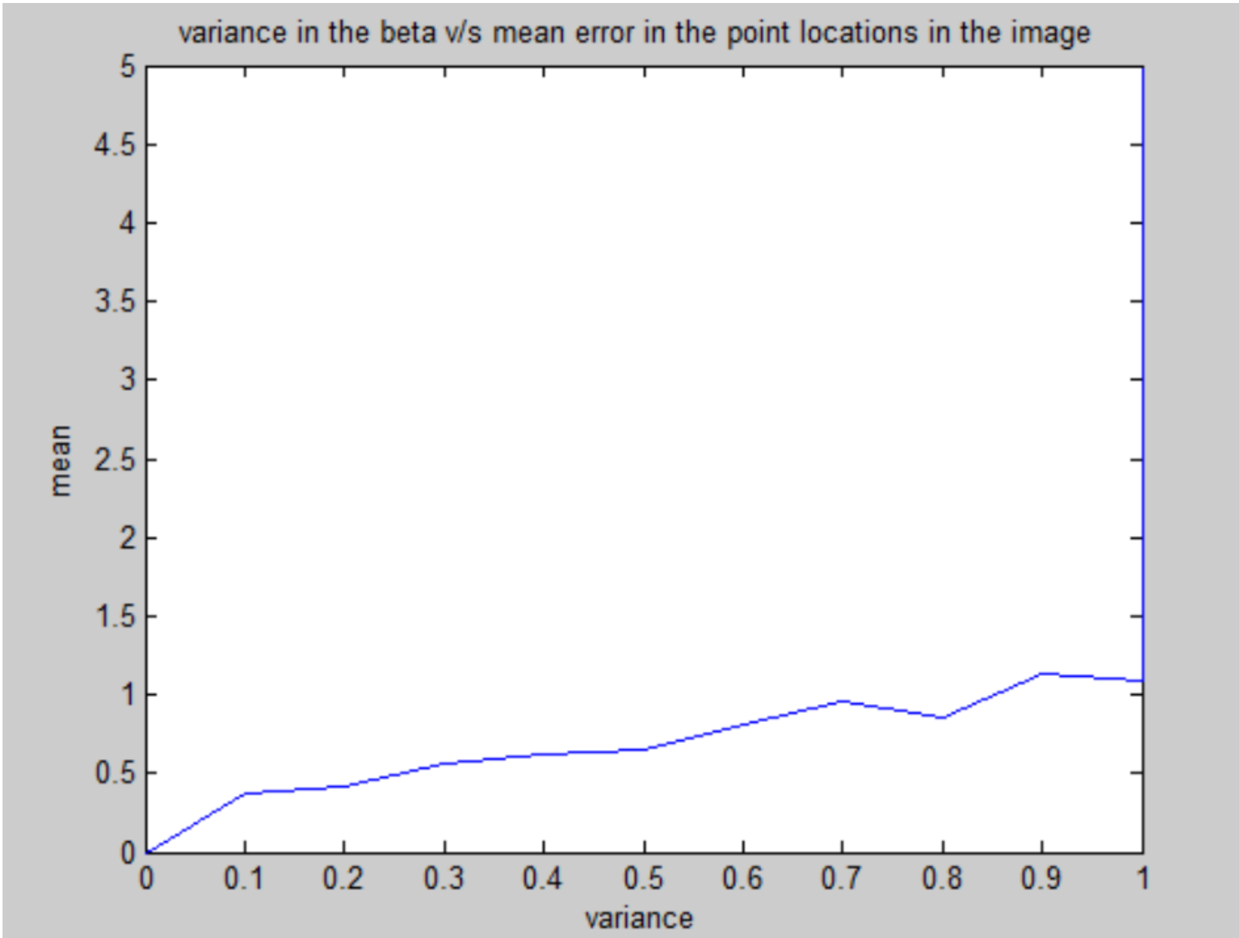

Plot of trial number vs error for that trial number with change in theta
Error Value
Trial Number
0
1
2
3
4
5
6
7
8
9
10
20
30
40
50
60
70
80
90
100

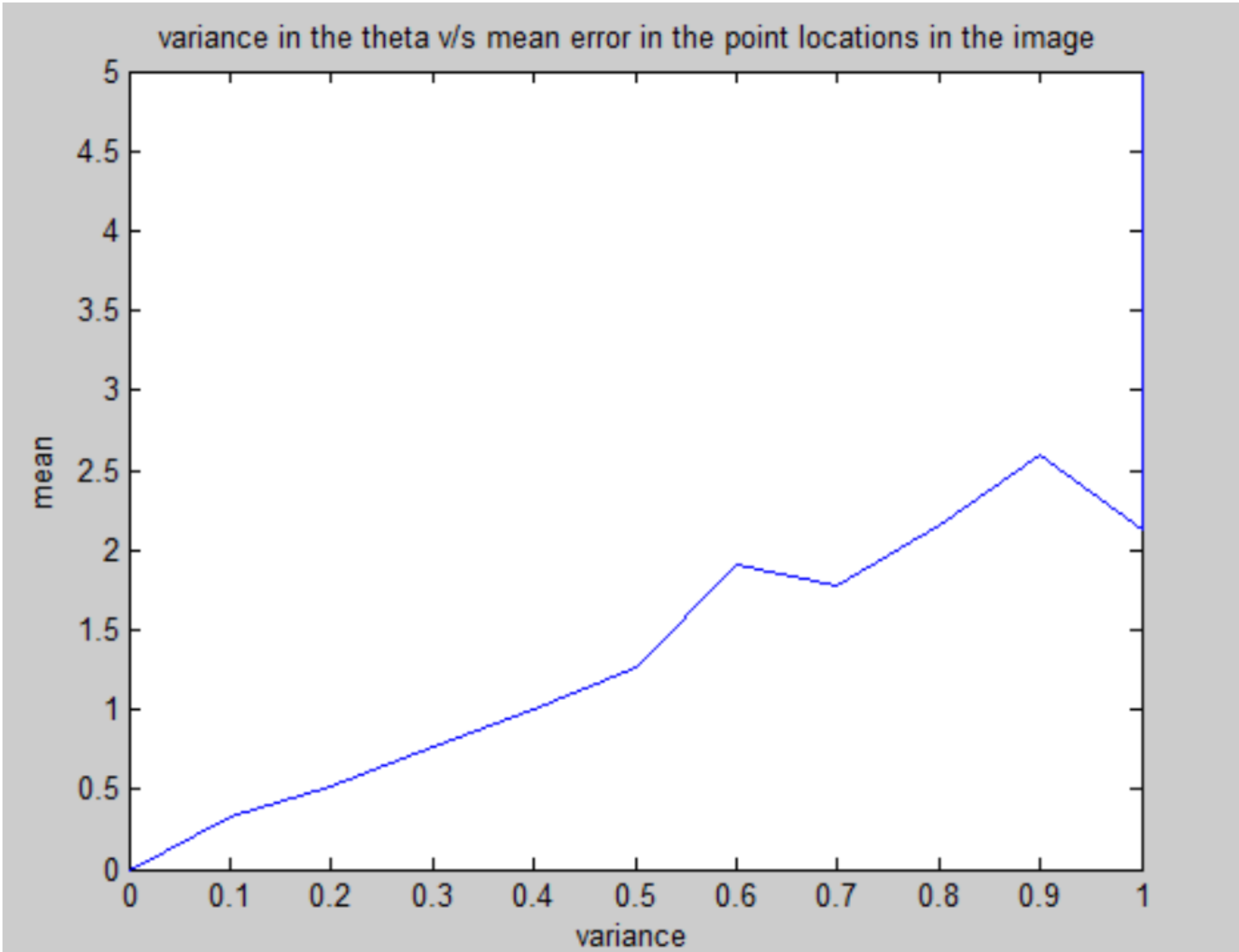

Plot of trial number vs error for that trial number with change in x0
Error Value
Trial Number
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
10
20
30
40
50
60
70
80
90
100


Plot of trial number vs error for that trial number with change in y0
Error Value
Trial Number
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
10
20
30
40
50
60
70
80
90
100

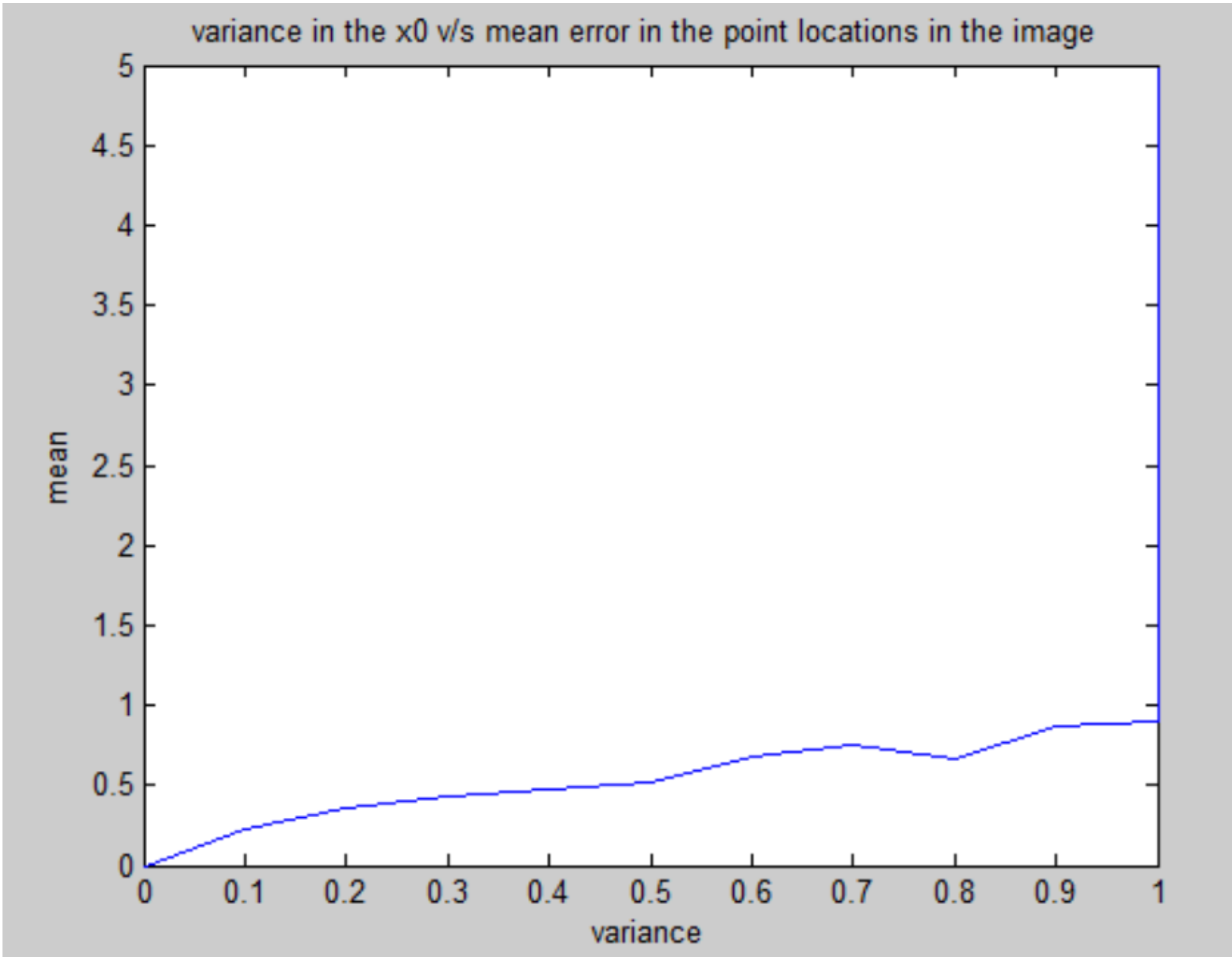
EXPERIMENT 2: The following figures show variance in the parameter v/s mean error in the point locations in the image. Parameter name is there in the title

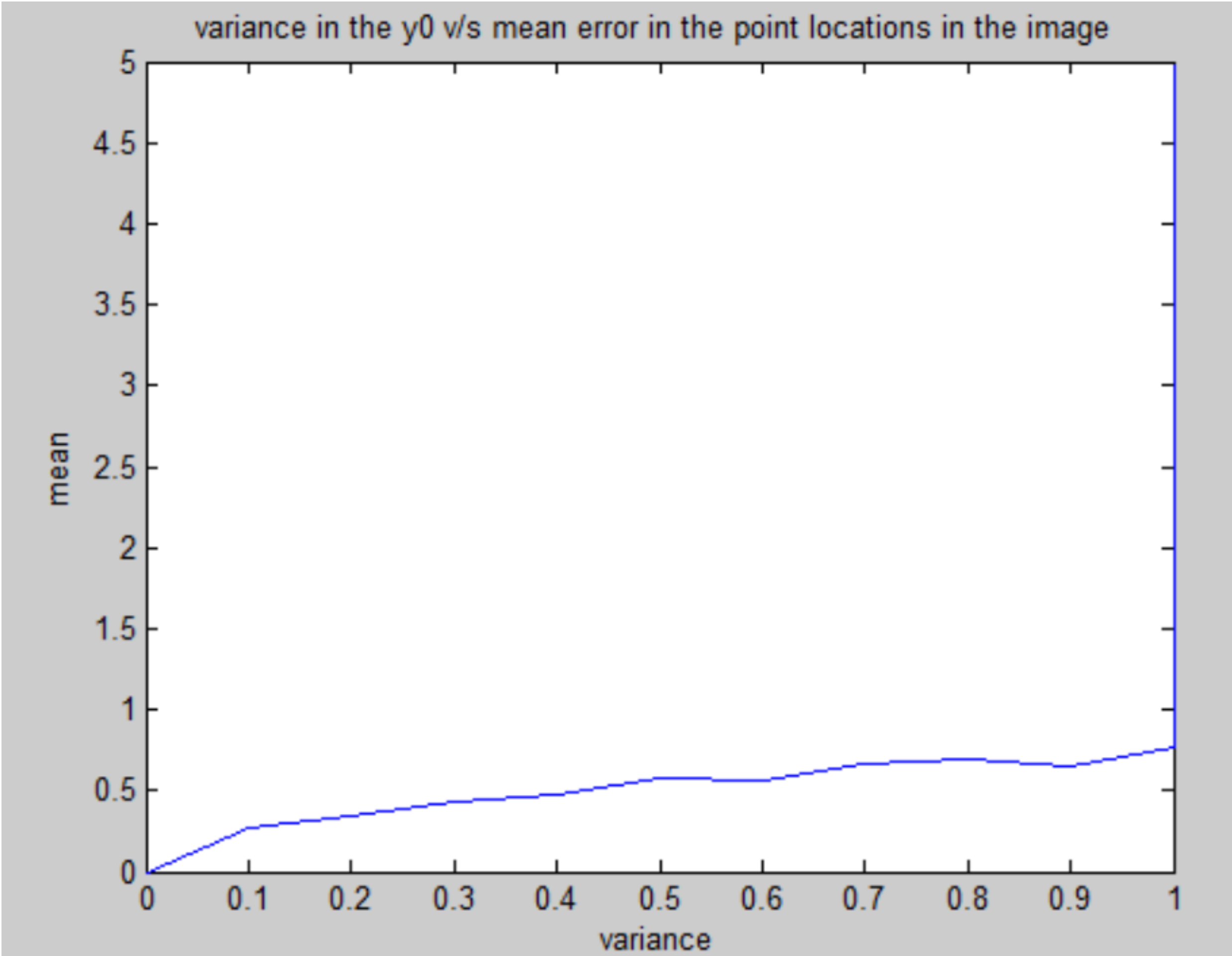



variance in the beta v/s mean error in the point locations in the image
mean
variance
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9



variance in the theta v/s mean error in the point locations in the image
mean
variance
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


variance in the x0 v/s mean error in the point locations in the image
mean
variance
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1



Section 5: Analysis:

Experiment 1: Observing plots for 100 trials with change in various intrinsic parameters.

Results for change in **alpha**:

Mean of errors for 100 trials: 0.385577222105104

Variance of errors for 100 trials: 0.0943099191691657

Confidence interval low for 100 trails: 0.334598754762259

Confidence interval high for 100 trails: 0.436555689447949

Results for change in **beta**:

Mean of errors for 100 trials: 0.409614792733834

Variance of errors for 100 trials: 0.106647033525596

Confidence interval low for 100 trails: 0.463825176018140

Confidence interval high for 100 trails: 0.355404409449529

Results for change in **theta**:

Mean of errors for 100 trials: 1.09415188278690

Variance of errors for 100 trials: 3.19702289963163

Confidence interval low for 100 trails: 0.797340220242701

Confidence interval high for 100 trails: 1.39096354533109

Results for change in **y0**:

Mean of errors for 100 trials: 0.698803642702128

Variance of errors for 100 trials: 0.271938541882653

Confidence interval low for 100 trails: 0.612238416080127

Confidence interval high for 100 trails: 0.785368869324129

Results for change in **y0**:

Mean of errors for 100 trials: 0.771248446639295

Variance of errors for 100 trials: 0.376216407474687

Confidence interval low for 100 trails: 0.669429885834437

Confidence interval high for 100 trails: 0.873067007444152

Experiment 2: **Observing plots variance in the parameter v/s mean error in the point locations in the image**

Other than observing variance in the parameter v/s mean error in the point locations in the image (as plotted in figures above), I observed the variance in the parameter v/s variance in error in the point locations, and variance in the parameter vs confidence intervals in error in the point locations. Although I did not draw figures for these for better clarity of the report.

Section 6: Interpretation:

The following are my observations:

- Since the noise is Gaussian, the trial number vs error curve is very erratic.

- Changes in x0 and y0 almost double the error in image compared to the error produced by change in alpha and beta
- Change in theta produces the maximum error
- There is almost a linear (not exactly linear but very close to linear as can be seen from Experiment 2 figures above) increase in the mean error for variance in parameters in the range 0.1 to 1 with 0.1 increment. This means that mean error in the point locations in the image is almost linearly directly proportional to variance in the parameter

Section 7: Critique:

I better learnt the concept of Perspective Camera Projection Model. The book reading makes more sense now

The experiment could be improved by following ways:

- Drawing variance in the point locations in the image is almost linearly directly proportional to variance in the parameter
- Drawing confidence interval in the point locations in the image is almost linearly directly proportional to variance in the parameter
- Try adding other kind of noises
- Try other world coordinates other than sphere and cube.