Mini- Project 5: Vision Guided Motion

Robotics 1

Professor John Wen

Varun Dhir

MANE 4560

10/30/21

https://github.com/Varun-ABC/Robotics1/tree/main/mini_project_5

I, Varun Dhir, certify that the following work is my own and completed in accordance with the academic integrity policy as described in the Robotics I course syllabus.

Table Of Contents

Abstract

This project deals with using pinhole cameras as a sensor. This has a lot of applications within any field of robotics such as precision manufacturing and autonomous cars.
First we looked at how and what a pinhole camera sees and the effect that moving it has on the image.
Then how to calibrate the camera was inspected, this included, finding the intrinsic and the extrinsic parameters of the camera. Intrinsic parameters are internal to the camera and do not change, extrinsic parameters are external to the camera and can be changed based on how the camera is mounted.
Finally, we looked at how to derive error from images in the task space and reduce it to guide inverse kinematics. The 2 methods investigated were Position Based Visual Servoing and Image Based Visual Servoing.

Approach- (What and How)

1. Camera Model and Projective Geometry (approach)

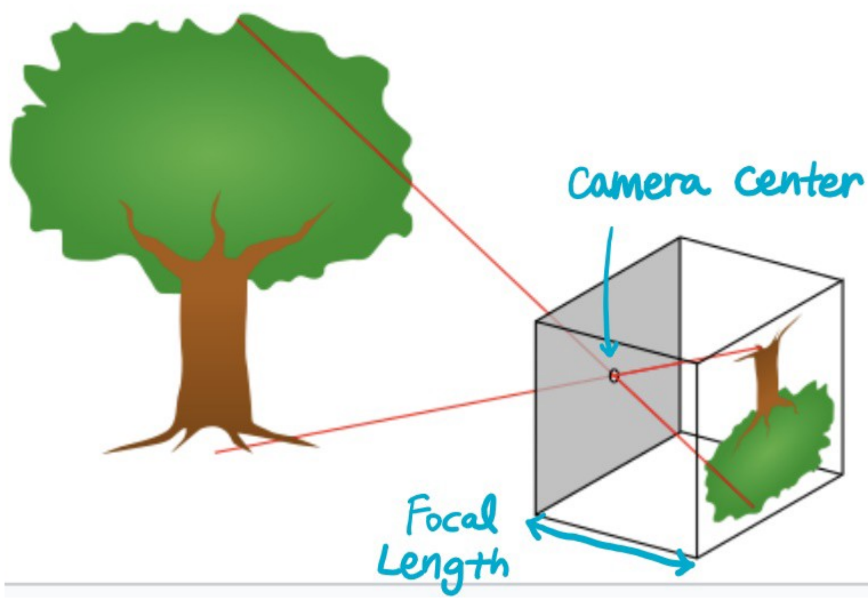
Problem Description: With given parameters and from a stationary position take a picture of a 3D target into a 2D plane with a pinhole camera.

Approach:

The given camera parameters are:

```
W = 1280; % width in pixels (1280)
H = 1024; % height in pixels (1024)
rhow = 1e-5; % width per pixel (10um)
rhoh = 1e-5; % height per pixel (10um)
f = .015; % focal length (0.015m)
u0=W/2; %center of image plane in pixel coordinate
v0=H/2; %center of image plane in pixel coordinate
```

On a pinhole camera, these parameters are shown below:



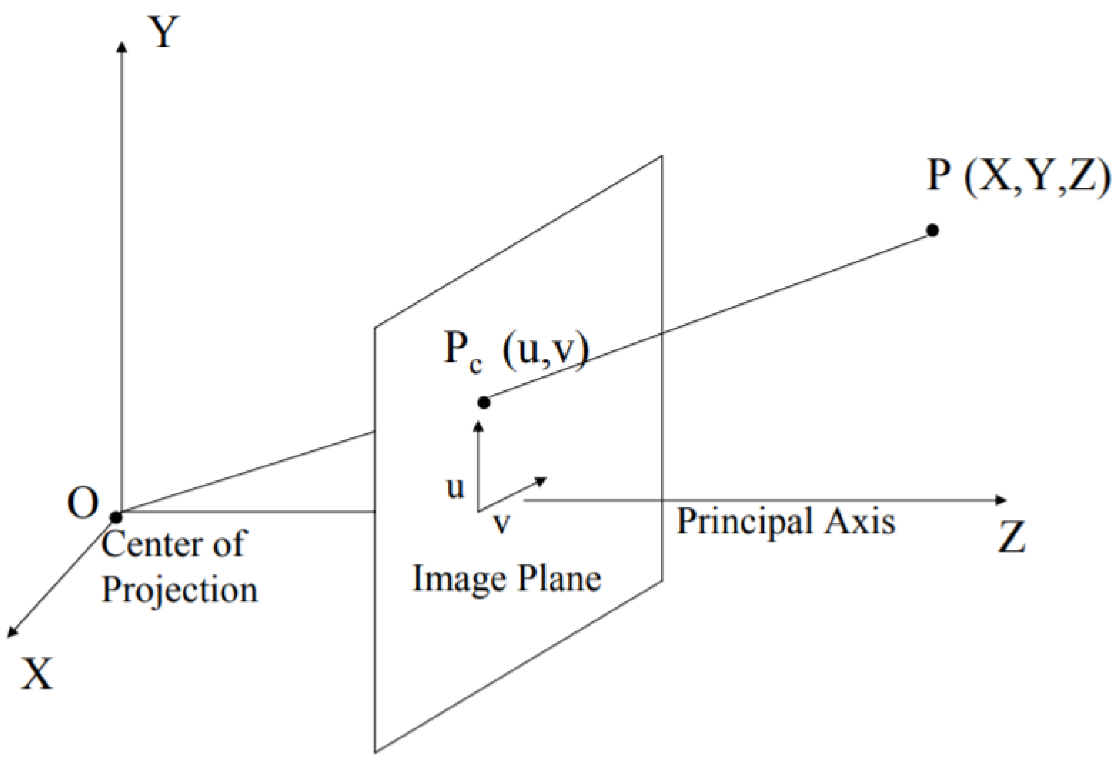
(source: Wikipedia)

By changing the distance from the target, we can grow or shrink the camera image or move it in its frame.

2. Intrinsic Camera Parameter Calibration (approach)

a) Problem Description: Explain the derivation of the camera calibration process

Camera calibration refers to both the intrinsic and extrinsic calibrations. The intrinsic calibration determines the optical properties of the camera lens as shown above. The extrinsic properties describe the camera's relationship to the outside (translation and rotation). By looking at a known object in many different views we can determine these properties.



From the image above, we can use geometry to get the following expression:

$$\frac{f}{Z} = \frac{u}{X} = \frac{v}{Y}$$

Thus:

$$v = \frac{fY}{Z} \text{and } u = \frac{fX}{Z}$$

The homogeneous transform of this looks like:

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

Then to account the translations in X and Y where if the origin of the image coordinate is not on the Z - axis t_u and t_v are applied.

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} f & 0 & t_u \\ 0 & f & t_v \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

Finally, to apply the factor that converts from pixels to inches we need the conversion factor, here we will call it α.

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} \alpha_u f & 0 & \alpha_u t_u \\ 0 & \alpha_v f & \alpha_v t_v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

Which turns into:

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} \lambda_u & 0 & u_0 \\ 0 & \lambda_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

Here the middle matrix is K, the matrix of intrinsic parameters.

The exteresic parameter matrix is given by concatenating the rotation matrix and the rotation matrix multiplied by the transform matrix.

$$E = (R \mid RT)$$

Thus the entire camera transform can be represented l

$$K(R \mid RT) = (KR \mid KRT) = KR(I \mid T)$$

ike this:

The projection of point p onto the image plane is:

$$P_c = KR(I \mid T)P = CP$$

Using a planer target this becomes:

$$\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \underbrace{\frac{1}{w_i} K \begin{bmatrix} R_{C0}^{(1)} & R_{C0}^{(2)} & p_{C0} \end{bmatrix}}_{\frac{H}{w_i}} \begin{bmatrix} X_i \\ Y_i \\ 1 \end{bmatrix}.$$

Here H is also known as the homography matrix. For a single known target point, we have u_i, v_i, $X_{i\text{-}i}$, and Y_i. From this equation a system of equations be derived:

$R_{C0}^{(1)} = K^{-1}$

To solve for the parameters in H, many images are taken at known positions, and then are input into a least squares solver. The more points we have to solve camera calibration the more robust the least squares solver will be.

b) Problem Description: Comment on the effect of the number of target points and number of views on the estimated camera intrinsic parameters.

Increasing the number of target points and views increases the accuracy of camera calibration, this is because the more points there are per view the more equations each view generates, and the more views there are the more over all equations there are. However, with sensor noise too many target points may have some that are too close to each other and the noise may confuse the calibration process ending up in improper calibration, so having spaced out points is often more beneficial than having points that are too close to each other.

3. Camera/Robot Calibration (approach)

Problem Description: Find the extrinsic parameters

Once the intrinsic parameters of the robot are calibrated, the extrinsic parameters. Eye- in-hand calibration is where the camera is rigidly mounted to the robot end effector as is the case here. From the problem statement, P_{0B} and P_{TC} are unknown, however, R_{TC} and R_{B0} are known. For each of M calibration pose, there are 3 equations and 6 unknowns:

$R_{BO}R_{OC} = R_{BT}R_{TC}$: All of these parameters are known, but if the camera orientation was not known (as is the case in the main semester project) this equation would need to be solved.

$R_{BO}P_{OC} + P_{BO} = R_{BT}P_{TC} + P_{BT}$ Since all the position vectors are 3 elements, this expression can be broken into X, Y, and Z components. For ease, the z- axis of the camera is aligned with the x- axis of the tool frame.

The algorithm find these parameters broken down below:

1) Perform intrinsic camera calibration as described above to get the intrinsic parameters
2) Have or generate a known image, in our case the 3D S curve
3) Generate a pose where the robot is looking at the target
4) Generate the camera image and get the image coordinate of P0 (uv) for each point on the curve
5) Use PNP (Perspective N Point) methods to match the 3D points of the S-curve to the 2D points in the image plane
6) The PNP method uses linear regression in the form:

$$\underbrace{\begin{bmatrix} I \otimes P_{01}^T & I & y_1 \\ \vdots & \vdots & \vdots \\ I \otimes P_{0N}^T & I & y_N \end{bmatrix}}_{C \in \mathbb{R}^{3N \times (12+N)}} \begin{bmatrix} R^{(1)} \\ R^{(2)} \\ R^{(3)} \\ p_{C0} \\ w_1 \\ \vdots \\ w_N \end{bmatrix} = 0$$

Where Y is

$$y = K^{-1} * [uv; 1]$$

7) PNP calibration works by using SVD to estimate the null space of C

a) This is done by checking the smallest singular values to make sure there is a one-dimensional approximate null space

8) Then C is normalized to ensure estimated R_{C0} exists in 3 dimensions.

9) Then use the same scaling to find P_{C0} and w_i's.

4. Visual Servoing (approach)

Problem Description:
Visual Servoing is used to reduce error using feedback from imagery, this can be done in 2 ways as discussed below, Position Based Visual Servoing (PBVS) and Image Based Visual Servoing (IBVS).

Position Based Visual Servoing (PBVS)

To do PBVS, an image is taken to find the desired position and orientation, then the error is formed between the current and the desired pose. This error is then decreased iteratively. The process follows these steps:

1. The current pose is calculated using input q values
2. In this case, it is possible that the whole target is not in the image frame, if that is the case move the robot so that the whole image comes into frame
3. When the whole target is in view, the camera takes an image and finds the desired pose
4. Once the desired pose is found, inverse kinematics is run, in this case an iterative method is run using damped least squares until the max iterations are reached or the error is sufficiently small

Mathematically, the process is summed like this:

$$T^{*}_{BT} = T_{BT} T_{TC} T^{*}_{0T} T_{TC}^{-1}$$

$$min(||T_{BT} - T^{*}_{BT}||)$$

Image Based Visual Servoing (IBVS)

To do IBVS, error is reduced directly in the image plane.

1. The current pose is calculated using input q values
2. In this case, it is possible that the whole target is not in the image frame, if that is the case move the robot so that the whole image comes into frame
3. When the whole target is in view, the camera takes an image and finds the desired pose

4. Then the transformation from the target to the camera is calculated as well as the jacobian to the target.
5. Once the desired pose is found, iterative inverse kinematics is run, where error remains in the 2D image space.

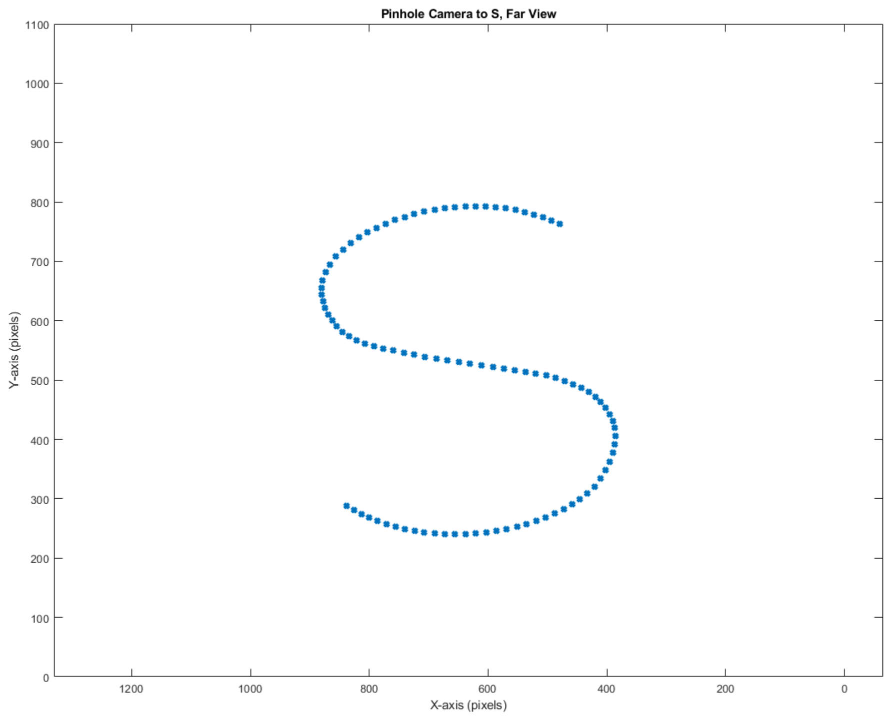
Mathematically this can be shown as:

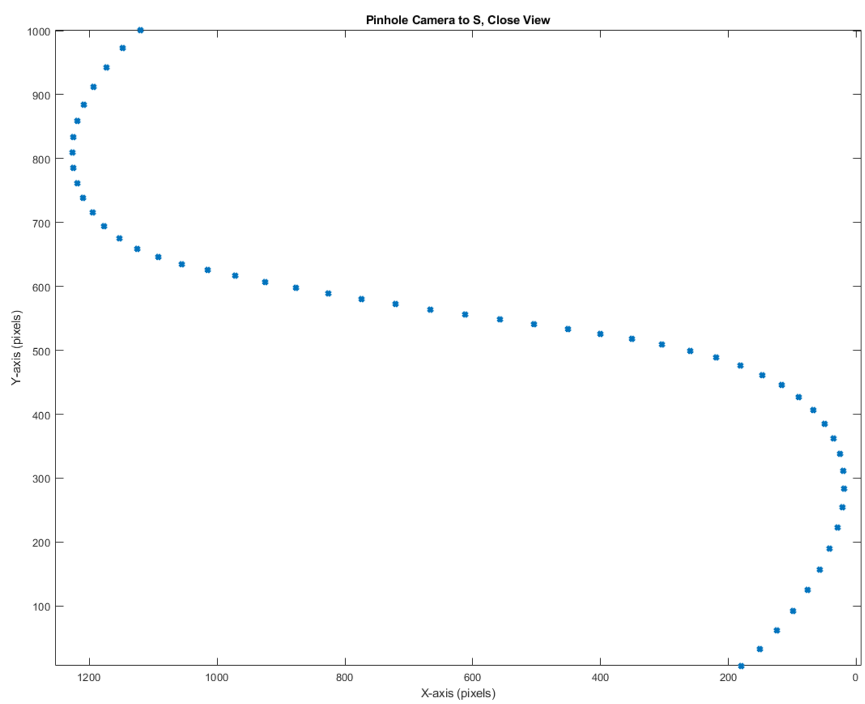
$$\begin{bmatrix} \dot{u}_i \\ \dot{v}_i \end{bmatrix} = J_{I_i}\dot{q}$$

$$\min_{\dot{q}} \|J_I\dot{q} - (-K_p(y - y^*))\| + \epsilon \|\dot{q}\|^2$$

Results

1. Camera Model and Projective Geometry (Results)





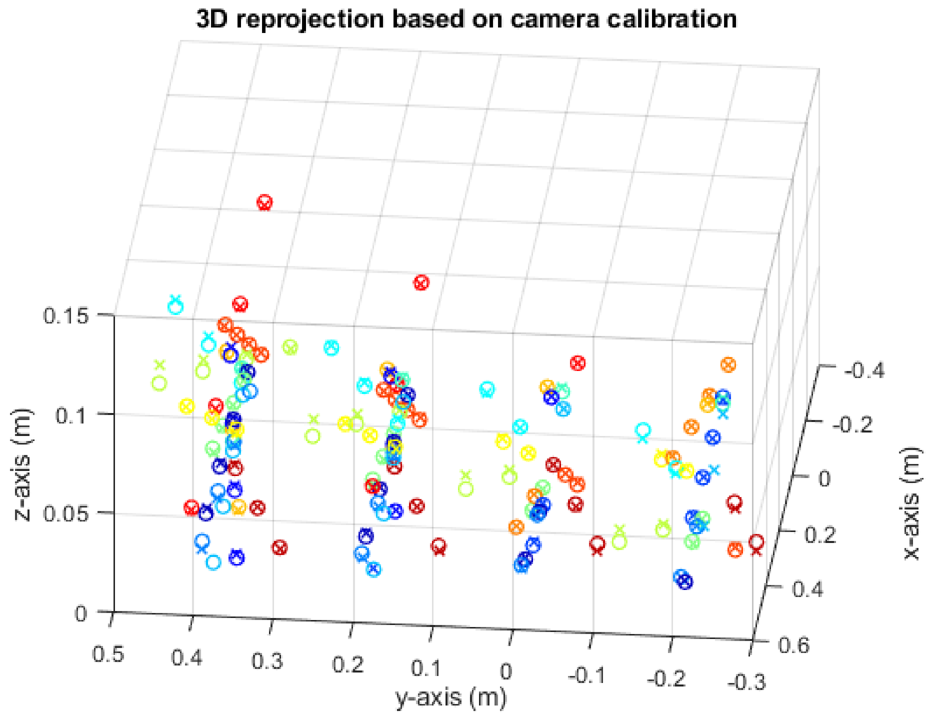
2. Intrinsic Camera Parameter Calibration (Results)

To get the maximum percent error in K under 1 percent and the Maximum Reprojection error under 1 pixel, the values used for M is 15 and N is 16. The target was defined using these parameters:

nx1=-.2; kx=.2; nx2=.4;

ny1=-.2; ky=.2; ny2=.4;

```
**** Singular Values ****: [  5.37,  4.108,  0.2749,  0.0004482,  0.0001064,  1.789e-09  ]
K
   1.0e+03 *

    1.5000         0    0.6400
         0    1.5000    0.5120
         0         0    0.0010

Estimated K
   1.0e+03 *

    1.4867    0.0003    0.6368
         0    1.4870    0.5118
         0         0    0.0010

percentage error in K
 0.89, -, -
 -, 0.86, -
 0.5, 0.039, 0

**** Maximum Reprojection Error:    0.193
```


3D reprojection based on camera calibration
0.15
0.1
0.05
0
z-axis (m)
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
y-axis (m)
-0.4
-0.2
0
0.2
0.4
0.6
x-axis (m)

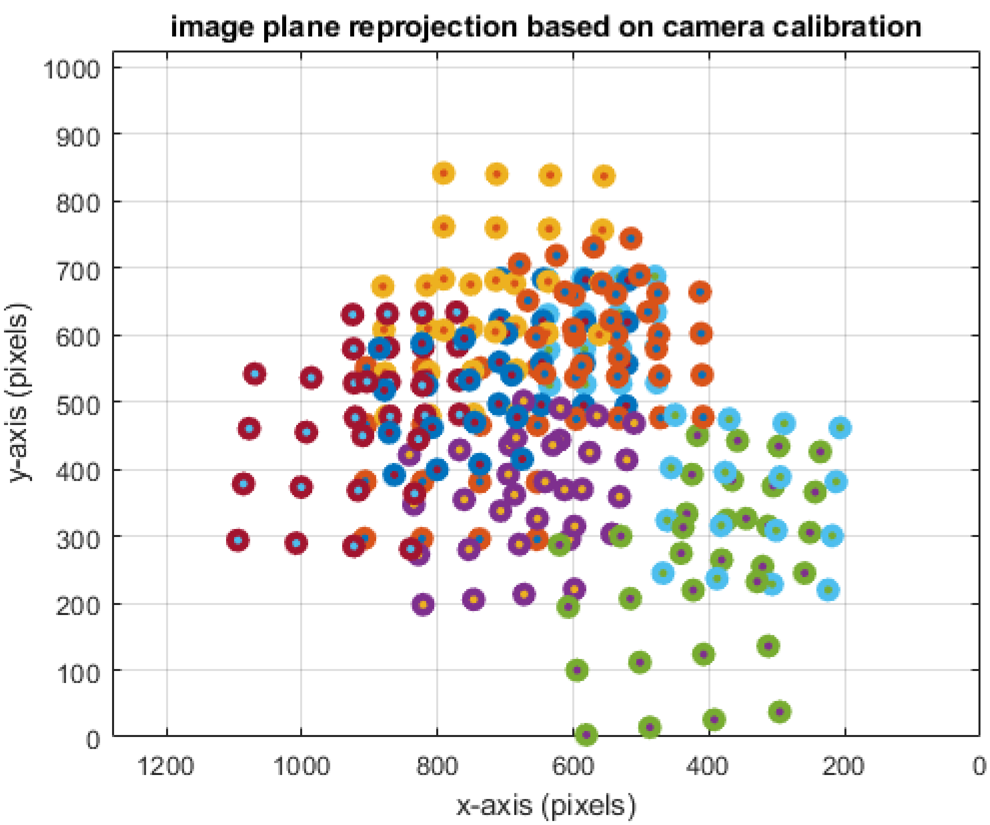
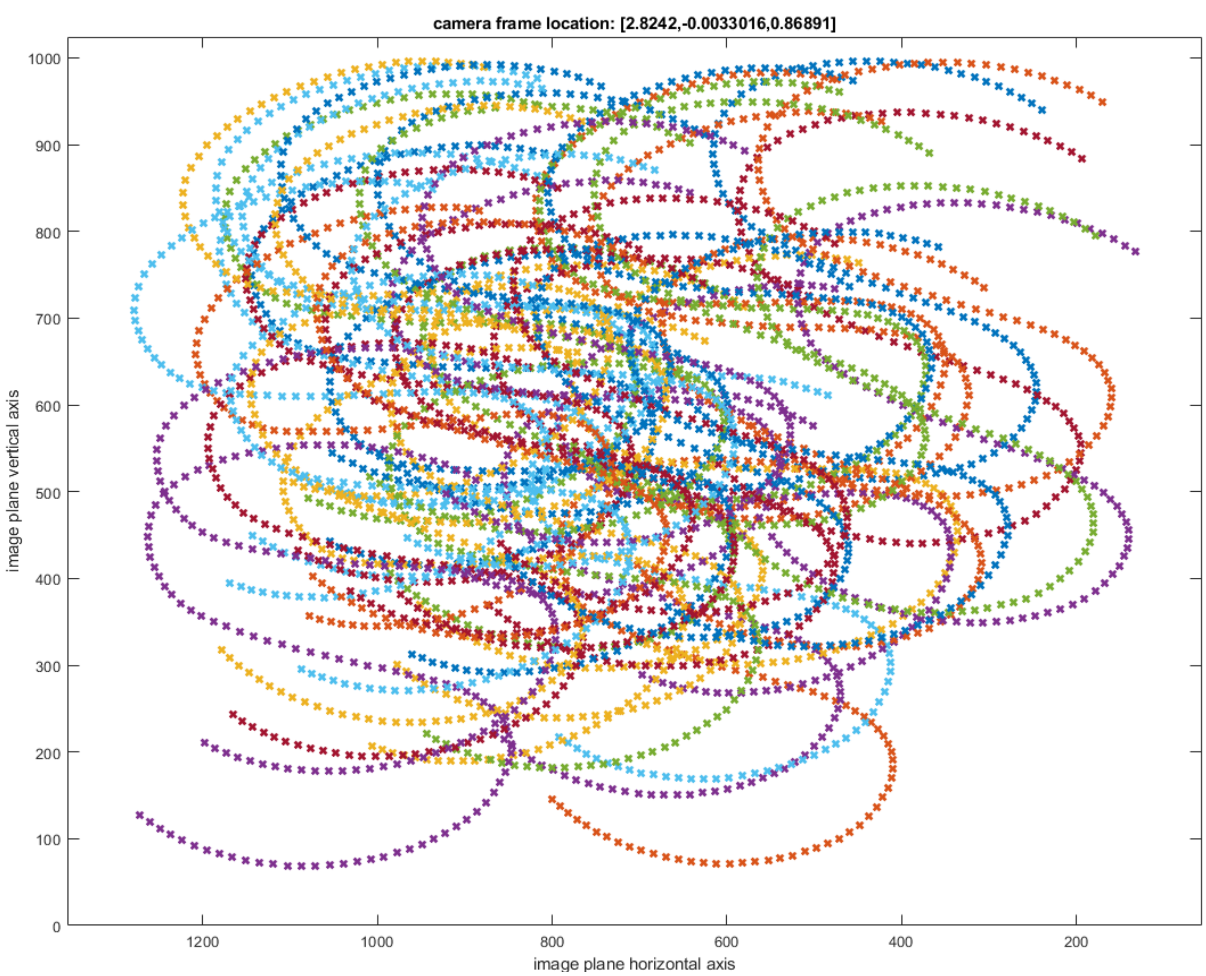


image plane reprojection based on camera calibration
1000
900
800
700
600
500
400
300
200
100
0
y-axis (pixels)
1200
1000
800
600
400
200
0
x-axis (pixels)

3. Camera/Robot Calibration (Results)

To find the error, the 2 norm of the two position arrays was taken. To achieve an error below 1mm, M is found to be 50.

```
p_TC vs. estimate
         0   -0.0010
         0    0.0001
    0.0500    0.0513

Error in T_TC: 0.0016785
p_OB vs. estimate
    2.5000    2.4994
         0    0.0002
         0   -0.0014

Error in T_OB: 0.0015176
```



4. Visual Servoing (Results)

Position Based Visual Servoing (PBVS)

Results from PBVS using 100 interactions, a step size of .1, and weighting of [1;1;1;10;10;10] to compensate for units.

```
T_0C estimated from PBVS
    0.0002   -0.0017   -1.0000    1.8000
   -1.0000    0.0007   -0.0002    0.0001
    0.0007    1.0000   -0.0017    0.5501
         0         0         0    1.0000

T_0C Desired
         0         0   -1.0000    1.8000
   -1.0000         0         0         0
         0    1.0000         0    0.5500
         0         0         0    1.0000

2 Norm Error
    0.0018
```

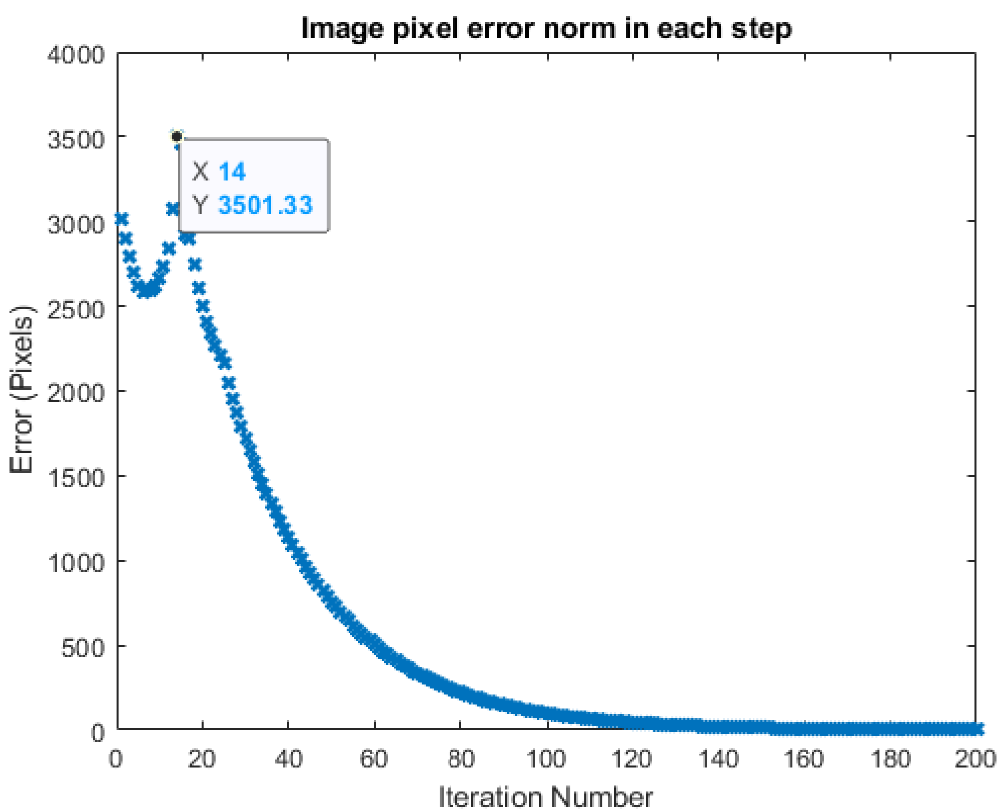
PBVS.avi shows a video of the convergence

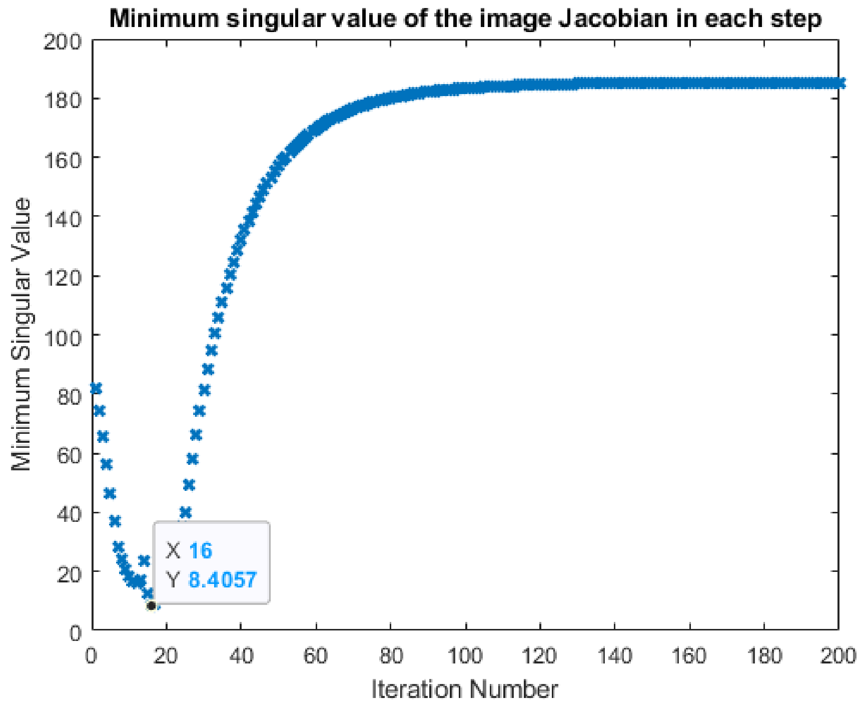
(https://github.com/Varun-ABC/Robotics1/blob/main/mini_project_5/figures/PBVS.avi).

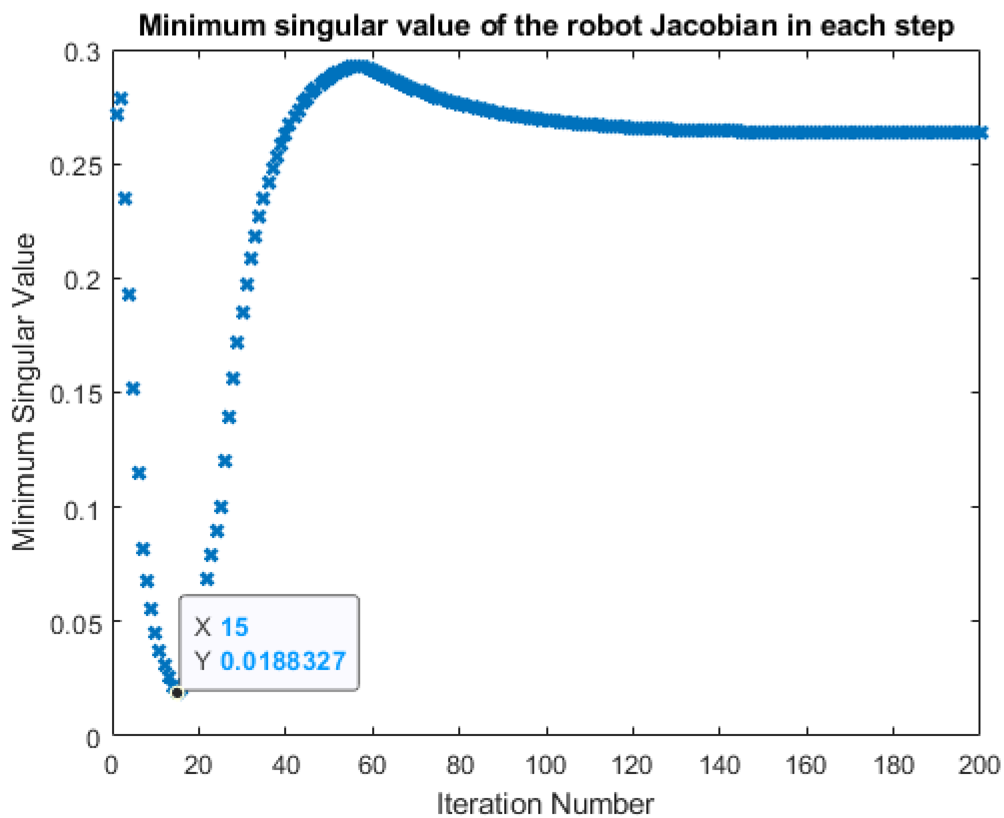
<u>Image Based Visual Servoing (IBVS)</u>

For IBVS, 200 steps were used and the step size was .03 with the same weights as before.

```
T_0C estimated from IBVS
   -0.0001   -0.0000   -1.0000    1.8002
   -1.0000    0.0000    0.0001   -0.0000
    0.0000    1.0000   -0.0000    0.5501
         0         0         0    1.0000

T_0C Desired
         0         0   -1.0000    1.8000
   -1.0000         0         0         0
         0    1.0000         0    0.5500
         0         0         0    1.0000

2 Norm Error
   2.7687e-04
```







IBVS.avi shows a video of the convergence (https://github.com/Varun-ABC/Robotics1/blob/main/mini_project_5/figures/IBVS.avi).

Conclusion

For camera calibration, increasing the number of targets seems to lead to decreases in error for both intrinsic and extrinsic parameters, this makes sense as it is giving the least squares algorithm more data points to base its estimate on.

For calibrating the intrinsic parameters, in the presence of noise, too many points per target seemed to decrease performance as the points were conflicting and at the spacing was at a scale where the noise dominated.

For Visual Servoing, both PBVS and IBVS converged to a good solution, however, IBVS is much more sensitive to jacobian singularity than PBVS, this is shown by the figures that show the minimum singular value of the robot and image jacobian being the same point where the error spikes. IBVS was also much more sensitive to changing the step size than PBVS.

References

(1) Wen, J. T. (2021). ”MANE 4560 - Mini-Project 4: Jacobian, Singularity” Rensselaer Polytechnic Institute

(2) Peng, Yuan-Chih. *Comparing Position- and Image-Based ... - Welcome to ECSE*. https://ecse.rpi.edu/~rjradke/papers/peng-case20.pdf.

(3)“Pinhole Camera.” *Wikipedia*, Wikimedia Foundation, 4 Nov. 2021, https://en.wikipedia.org/wiki/Pinhole_camera.