ROB501 Assignment 4 Report

Overview

In computer vision, Image-Based Visual Servoing (IBVS) is a technique that has gained significant attention for its ability to enhance robotic manipulation and navigation. IBVS involves the use of image data to control the motion of a robot by adjusting its position and orientation in relation to specific visual features within its environment. This is achieved through a continuous feedback loop where the differences between the current and desired image features are minimized, guiding the robot towards a predetermined target state. The versatility of IBVS makes it an invaluable tool in various applications, ranging from precision assembly to autonomous navigation.

This report aims to examine the performance of a simple IBVS controller implemented in Python under different circumstances: when the feature depth was fully known and when the feature depth was estimated. The performance of the two scenarios is evaluated against two initial camera poses to mitigate potential bias that could arise in a single trial.

Camera Setup

The camera has a intrinsic matrix K that is

$$\begin{bmatrix} 500.0 & 0 & 400.0 \\ 0 & 500.0 & 300.0 \\ 0 & 0 & 1 \end{bmatrix}$$

The feature points in target frame is

$$\begin{bmatrix} -0.75 & 0.75 & -0.75 & 0.75 \\ -0.50 & -0.50 & 0.50 & 0.50 \\ 0.00 & 0.00 & 0.00 & 0.00 \end{bmatrix}$$

Baseline

Using the sample initial and final camera poses given in the learner example and a gain value of 0.1, the controller with known depth converged within 69 iterations. In contrast, the controller with estimated depth converged in 70 iterations. The difference is very small but no conclusion can be drawn due to the fact that these models are not tuned yet.

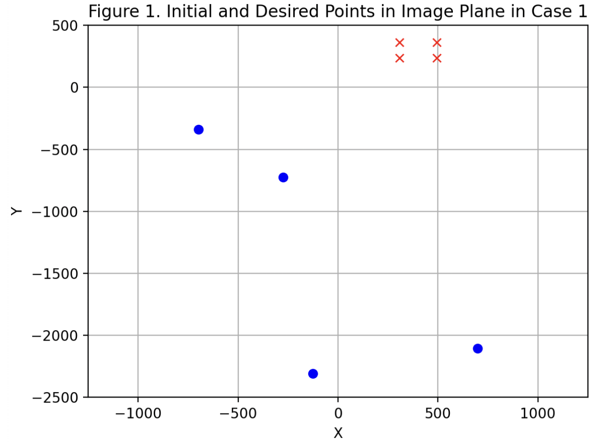
Case 1

The initial camera pose was being set as

$$\begin{bmatrix} 0.9045085 & 0.20307481 & -0.375 & 1.0 \\ -0.29389263 & 0.93401699 & -0.20307481 & 1.0 \\ 0.30901699 & 0.29389263 & 0.9045085 & 0.1 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{bmatrix}$$

while the desired pose stayed the same. This introduces a large rotation and translation in all directions. The initial and desired image points in 2D are shown in *Figure 1* below where the blue point is the initial points observed in 2D and the red crosses are the desired points.

Figure 1. Initial and Desired Points in Image Plane in Case 1



Analysis

Using a loop that iterates over the gain value from 0.01 to 2 (0 is omitted because it does not result in updates to the equation) with a step size of 0.01, the number of iterations required for convergence was recorded for both models and plotted below in *Figure 2* and *Figure 3*. Note that for a gain value greater than 1.7, the controller with estimated depth fails to converge, as the large gain value causes overshooting, leading to a failure in solving the least squares problem.

Figure 2. Number of Iterations versus Gain of Controller with Known Depth for Case 1

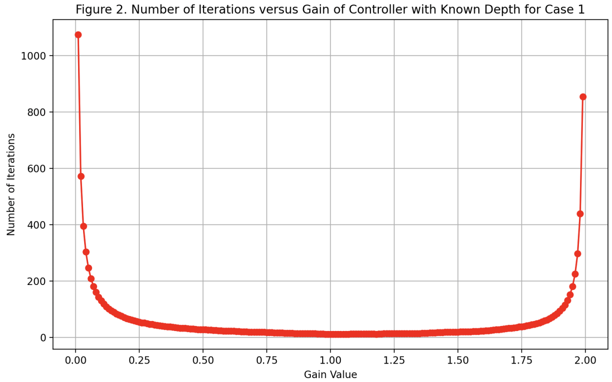
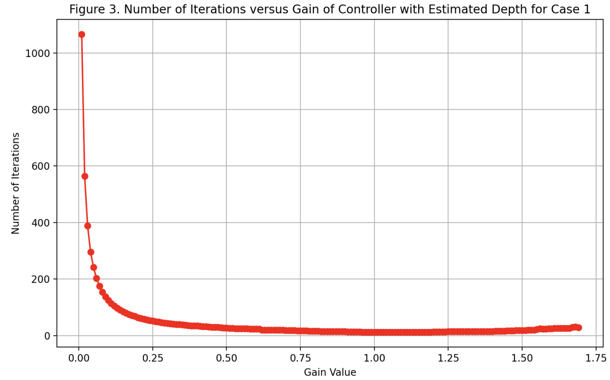


Figure 3. Number of Iterations versus Gain of Controller with Estimated Depth for Case 1



Given a gain value between **0.98 and 1.07**, the controller with known depths could achieve convergence to the desired pose in a minimum of **11 iterations**. The controller with estimated depths was able to converge to the desired pose within **12 iterations** using a gain value of **0.96 to 1.19**.

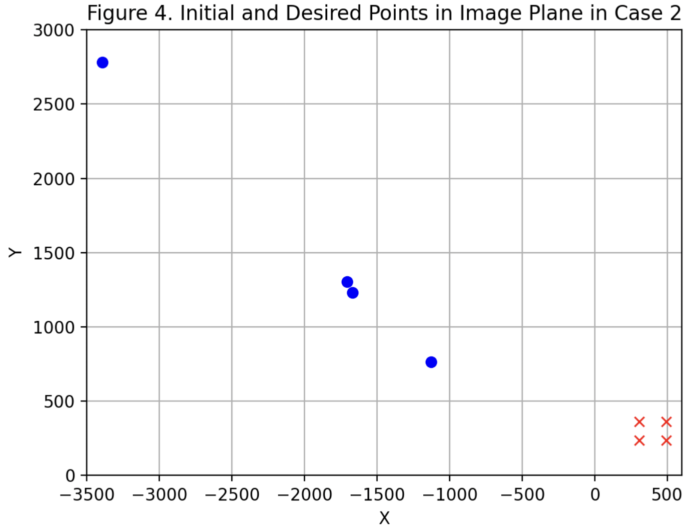
Case 2

The initial camera pose was being set as

$$\begin{bmatrix} 0.85355339 & 0.15463688 & -0.49752793 & 4.0 \\ -0.35355339 & 0.87332645 & -0.33511328 & -3.0 \\ 0.38268343 & 0.46193977 & 0.80010315 & -0.1 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{bmatrix}$$

while the desired pose stayed the same. This introduces an even larger rotation in the rho direction. Figure 4 shows the initial and desired image points in 2D, where the blue points represent the initial observations, and the red crosses indicate the desired points.

Figure 4. Initial and Desired Points in Image Plane in Case 2



Analysis

Using the same method as in Case 1, the number of iterations required for convergence were recorded for both models and plotted in *Figure 5* and *Figure 6*. Similar to Case 1, for a gain value greater than 1.4, the controller with estimated depth fails to solve the least squares problem.

Figure 5. Number of Iterations versus Gain of Controller with Known Depth for Case 2

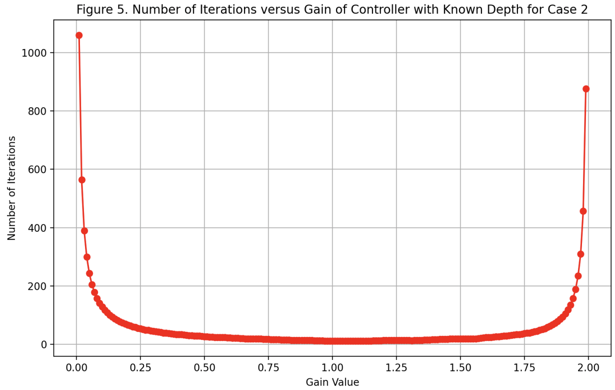
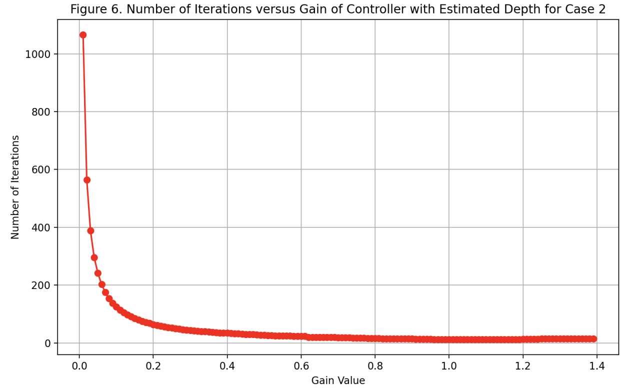


Figure 6. Number of Iterations versus Gain of Controller with Estimated Depth for Case 2



Given a gain value between **0.97 and 1.15**, the controller with known depths could converge in a minimum of **11 iterations**. Using a gain value of **0.97 to 1.04**, the controller with estimated depths could converge in as few as **10 iterations**.

Conclusion

Compared to known depths, the IBVS controller with estimated depths shows only marginal differences in the number of iterations required for convergence. In the baseline and Case 1 scenarios, the estimated depth controller performed slightly worse, but the difference was minimal (about one iteration). However, in Case 2, the estimated depth controller showed a marginally better performance.

The slight differences observed do not conclusively indicate that depth estimation significantly degrades the IBVS system's performance. This suggests that the IBVS controller's performance is robust to variations in depth estimation within the tested scenarios. However, the results are only specific to the conditions of the conducted experiments, and performance may be otherwise under different settings.