University of Burgundy

Masters in Computer Vision and Robotics

Visual Servoing Module

Report for Image Based Visual Servoing

By:

Vamshi Kodipaka

vamshikodipaka@gmail.com

Supervisor:

Dr. Omar Tahri

Dated: 22-Dec-2019

UNIVERSITÉ DE BOURGOGNE

IMAGE BASED VISUAL SERVOING

1 Introduction

Visual servo control refers to the use of computer vision data to control the motion of a robot by position and motion refinement. Visual servo control relies on techniques from image processing, computer vision, and control theory.

Image Based Visual Servoing was proposed by Weiss and Sanderson. The control law is based on the error between current and desired features on the image plane, and does not involve any estimate of the pose of the target. The features may be the coordinates of visual features, lines or moments of regions. IBVS has difficulties with motions very large rotations, which has come to be called camera retreat.

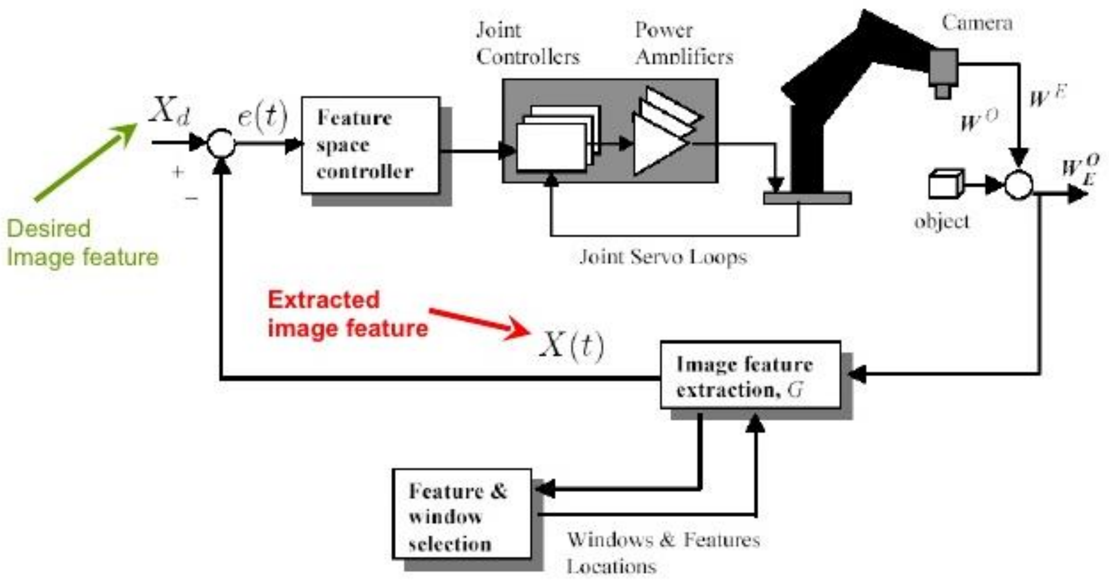


Fig-1: Block Representation of Image based Visual Servoing

Problem Statement: We want to move a camera from its current location to a desired location relative to a pattern (see Fig. 1) by using only image point features. We assume that we have only visual feedback. We can take images and detect image points of the given pattern at 50 Hz.

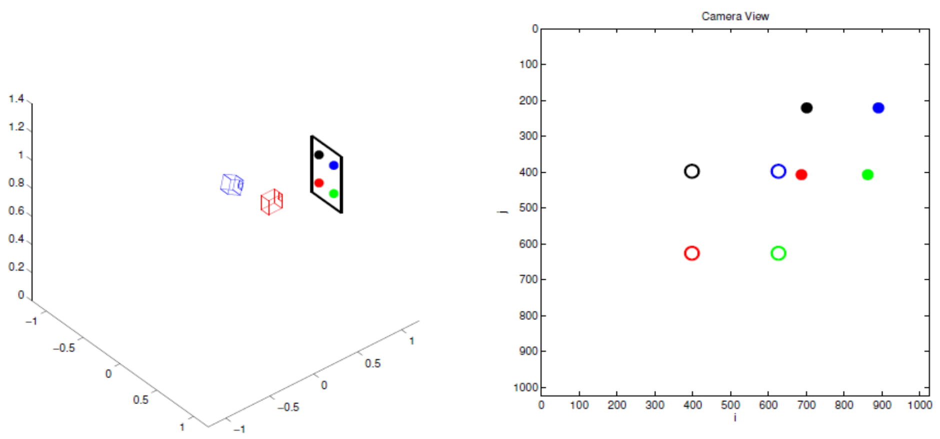


Fig.2: Explaining the problem statement of IBVS

Solution-1: Realizing the Image Based Control

Image based control of the camera pose using point features. Blue camera is current location, and the red camera is desired location of the camera w.r.t a pattern with four points (left fig). Empty dots are the desired image points observed from desired red camera location, and the full dots are the current image points observed from the current blue camera location (right fig).

Assuming current camera location at A w.r.t the fixed reference frame at R, then can be realize by considering:

1. 4 point features
2. 3 point features
3. 2 point features
4. Coplanar problem
5. Divergence problem
6. Local minimum problem

Solution-2: Mathematical Formulation

2.1 The visual features parameters extracted from the image are a function of the camera poses as given by

$$\mathbf{s} = \mathbf{s}(\mathbf{p}_t) \quad \dots (1)$$

s – order of **2nX1**

2.2 By taking the derivative of the above relation we obtain

$$\dot{\mathbf{S}} = \mathbf{L}_s \mathbf{V} \quad \dots (2)$$

where **L***s* the so called *Interaction matrix* or *feature Jacobian* of order **2nX6** and **V** the camera (Kinematic screw) denoted by **V** = ($\vec{v}$; $\vec{w}$). Thus the **V** contains 3 translations and 3 rotations.

2.3 The goal of the control law is to minimize the error given by:

$$\mathbf{e}(t) = \mathbf{S}(\mathbf{p}_t) - \mathbf{S}^* \quad \dots (3)$$

where **S*** the desired value of the feature. Using equations (1) and (3)

2.4 Then the time variation of the error is:

$$\dot{\mathbf{e}} = \mathbf{L}_s \mathbf{V} \quad \dots (4)$$

2.5 A good control law is to have an exponential decoupled decrease of the error $\dot{\mathbf{e}} = -\lambda\, \mathbf{e}$. We obtain:

$$\mathbf{V} = -\lambda \mathbf{L}_s^+ \mathbf{e} \quad . \quad (\lambda > 0) \quad \dots (5)$$

where $\mathbf{L}_s^+ \in \Re^{6\times k}$: is chosen as the Moore-Penrose pseudoinverse of **L**= $(\mathbf{L}^T\mathbf{L})^{-1}\mathbf{L}^T$ when **L** is of full rank 6. In real visual servo systems, it is impossible to know perfectly in practice either **L** or $\boldsymbol{L}^+$, so an approximation or an estimation $\boldsymbol{L}_s^+$ must be realized.

This corresponds to: $\mathbf{V} = -\lambda \widehat{\mathbf{L}_s^+}(\mathbf{S} - \mathbf{S}^*)$. … (6)

2.6 A image point s_i= [xi; yi]^T can be derived from the following two equations:

$$\mathbf{X} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}, \quad s(\mathbf{X}) = \begin{bmatrix} \mathbf{t} \\ \mathbf{u}\,\theta \end{bmatrix} \quad \dots (7a)$$

2.7 Finally, control law can be modified as:

$$\xi_{A/R} = \begin{bmatrix} \mathbf{R} & [\mathbf{t}]_\times \mathbf{R} \\ \mathbf{0}_{3\times 3} & \mathbf{R} \end{bmatrix}_{A/R} \xi_{A/A} \quad \ldots (7b)$$

.

2.6 Interaction matrix is given by the following equation:

$$\begin{bmatrix} -\frac{1}{Z} & 0 & \frac{x}{Z} & xy & -(1+x^2) & y \\ 0 & -\frac{1}{Z} & \frac{y}{Z} & 1+x^2 & -xy & -x \end{bmatrix} \quad \ldots (8)$$

which is a 2X6 matrix with the first 3 columns corresponds to a translation and the last 3 to a rotation.

‘

2.6 Stability analysis:

$$\begin{cases} \widehat{\mathbf{L}_s\mathbf{L}_s^+} = \mathbf{I} & \text{Ideal behavior} \\ \widehat{\mathbf{L}_s\mathbf{L}_s^+} > 0 & \text{The error e decreases} \\ \widehat{\mathbf{L}_s\mathbf{L}_s^+} < 0 & \text{The error e grows} \end{cases}$$

Solution-3: Block Diagram of IBVS (Refer Fig.1 above)

Solution-4: Algorithm

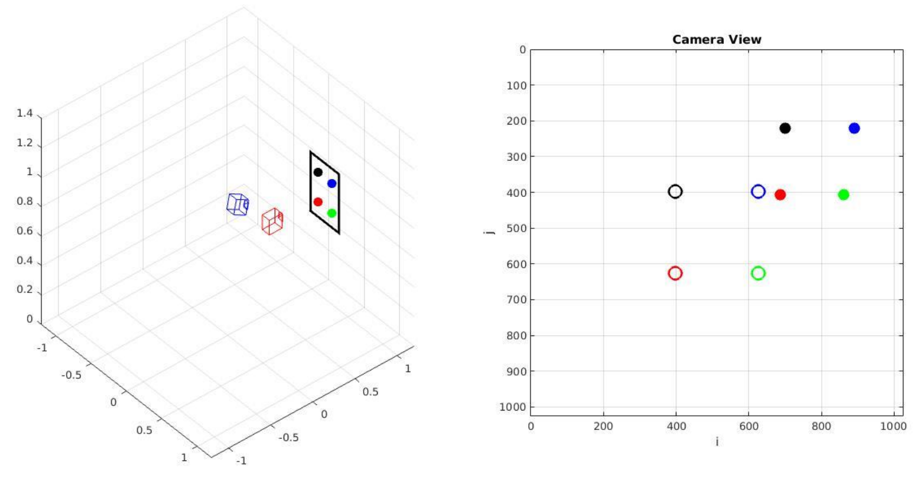
Algorithm: Image Based Visual Servoing

```
while not at end of time do
    Build the feature vector $s_A$
    Calculate the interaction matrix using (8)
    Find the error $\mathbf{e}$
    Calculate $\widehat{\mathbf{L}_s^+}$
    Calculate the control law
        $\xi_{A/R} = \begin{bmatrix} \mathbf{R} & [\mathbf{t}]_\times \\ \mathbf{O}_{3\times 3} & \mathbf{R} \end{bmatrix}_{A/R} \xi_{A/A}, \; \xi_{A/A} = \begin{bmatrix} v \\ \omega \end{bmatrix}$
    Apply the control law and get new camera location
end
```

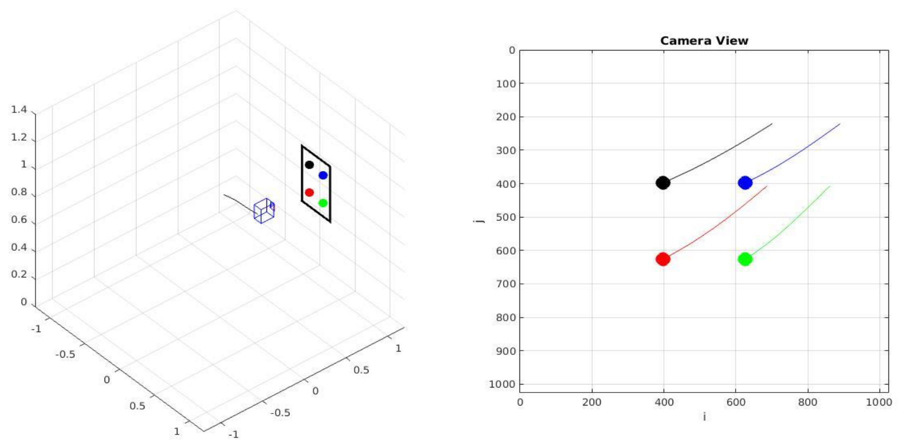
Solution-5: Matlab Simulation

5.1 Example of IBVS with 4 points (normal case):

In the first example we have 4 points, and the pattern is clearly visible from them. From the simulation results on Fig.3, we observe that the camera follows a very good and straight path. The velocity is getting smaller and smaller as we approaching the target. We notice that the interaction matrix has some very high eigenvalues and some very small ones. So the DoF or the eigenvalues with big values is the only ones that drives the control of the system.



a. Initial Camera view



b. After Motion

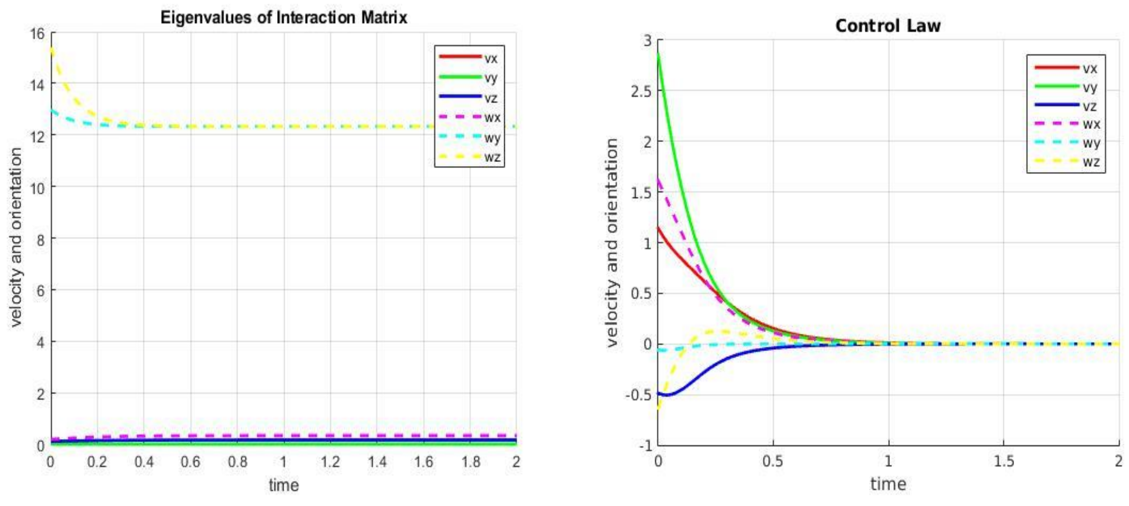
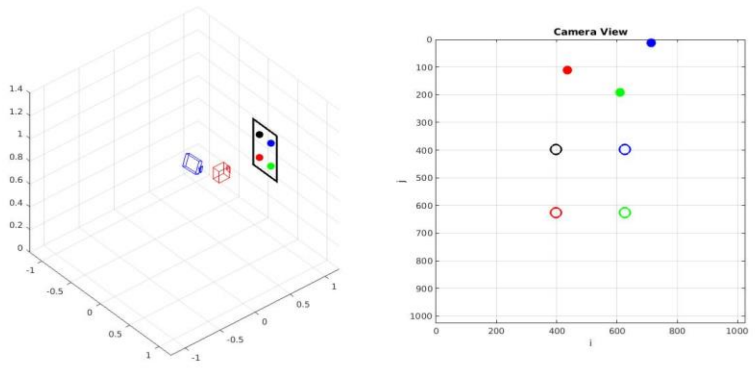


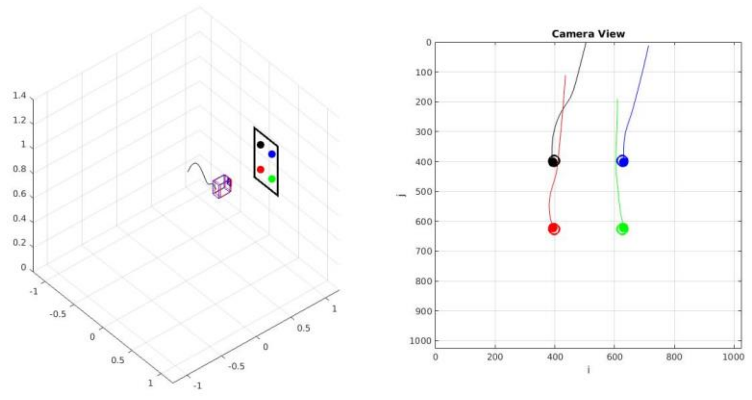
Fig.3: Example of IBVS with 4 points (normal case)

5.2 Example of IBVS with 3 points (losing one feature):

Here we only have 3 points, and the pattern is clearly visible from them. From the simulation results on Fig.4, we observe that the camera follows a deviation after a straight path till some extent and reaches target. We track only 3 points, even if all points are presented on the camera. The camera motion is far away from ideal, it even goes behind the scene, increasing the error, but it finally manage to get a good solution. The velocity is getting reduced at first and raised a bit and then reduced again as we approaching the target. We notice that the interaction matrix has repeated or symmetric values. So the DoF or the eigenvalues with non zero will contribute for the control system.



a. Initial Camera view



b. After Motion

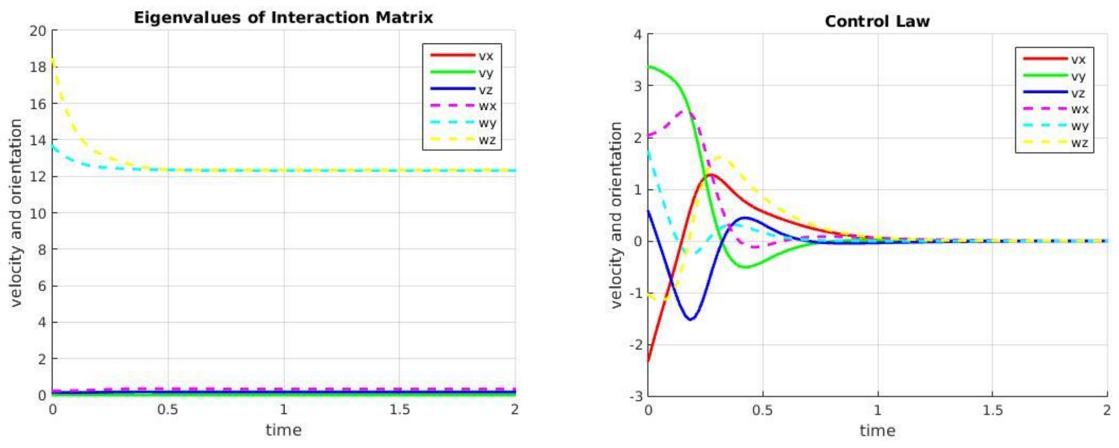
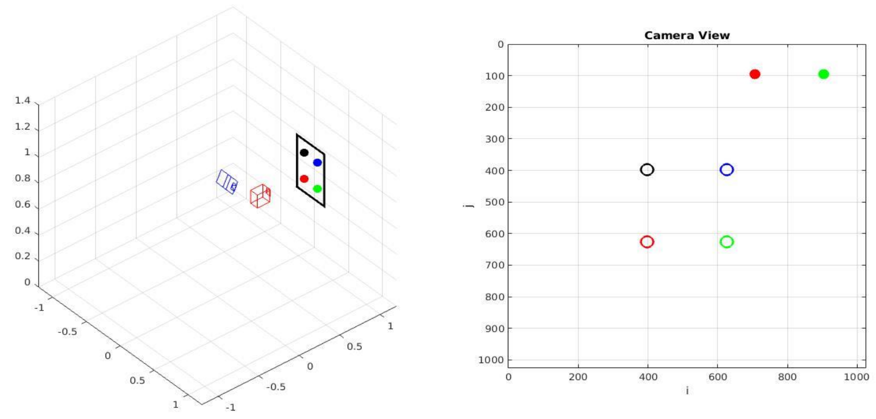


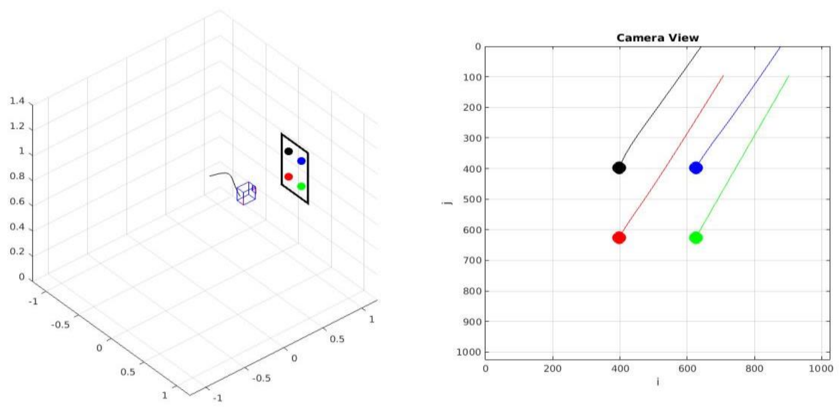
Fig.4: Example of IBVS with 3 points

5.3 Example of IBVS with 2 points (losing two features):

Here we only have 2 points, and the pattern is clearly visible from them. From the simulation results on Fig.5, we observe that the camera follows a path till reaches target except the other two feature points. Camera motion is far away from ideal, it even goes behind the scene, error tends to zero in the end. The velocity is getting reduced as we approaching the target.



a. Initial Camera view



b. After Motion

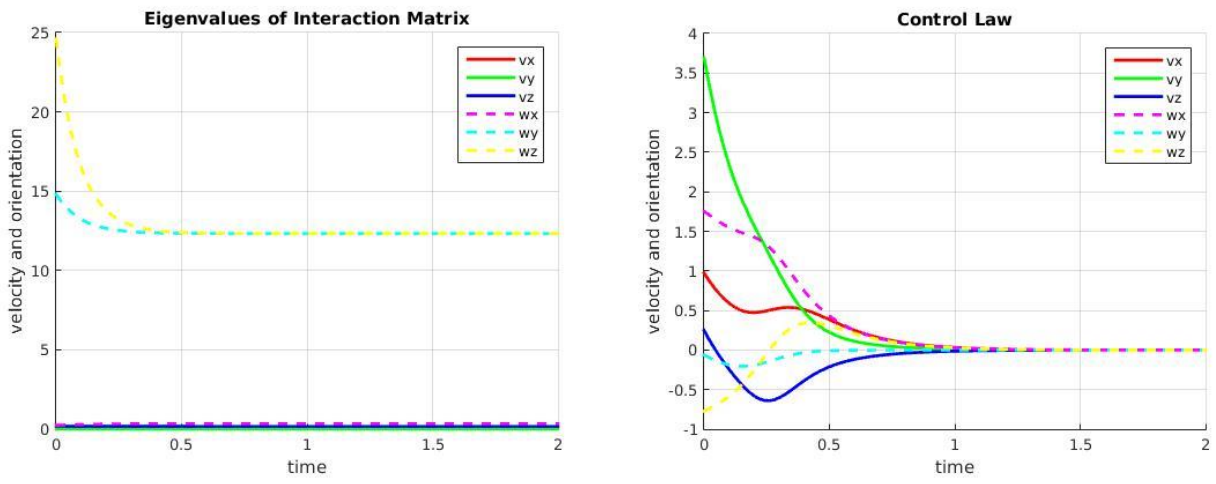
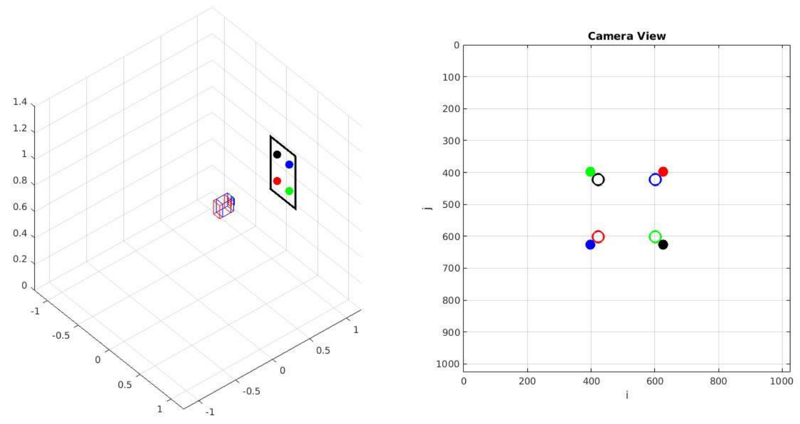


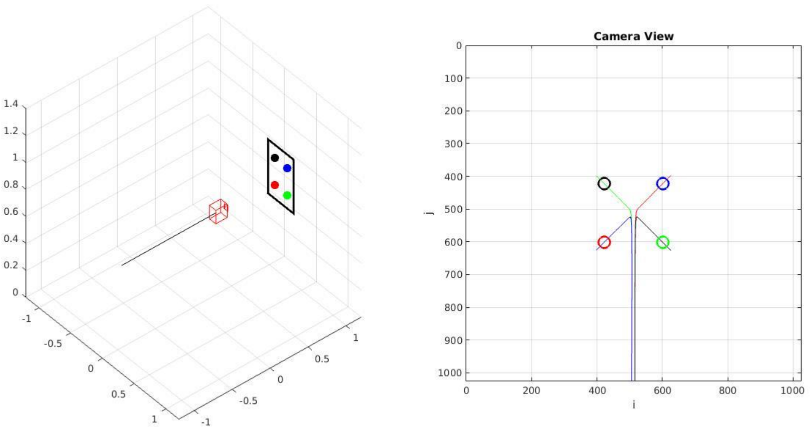
Fig.5: Example of IBVS with 2 points

5.4 IBVS with Coplanar problem:

Here we can suggest solution that is just a rotation. But the solution to minimize the distances is to move the camera backwards as the error is decreasing. All the eigen values are small in this case. The simulation results is given in Fig.6.



a. Initial Camera view



b. After Motion

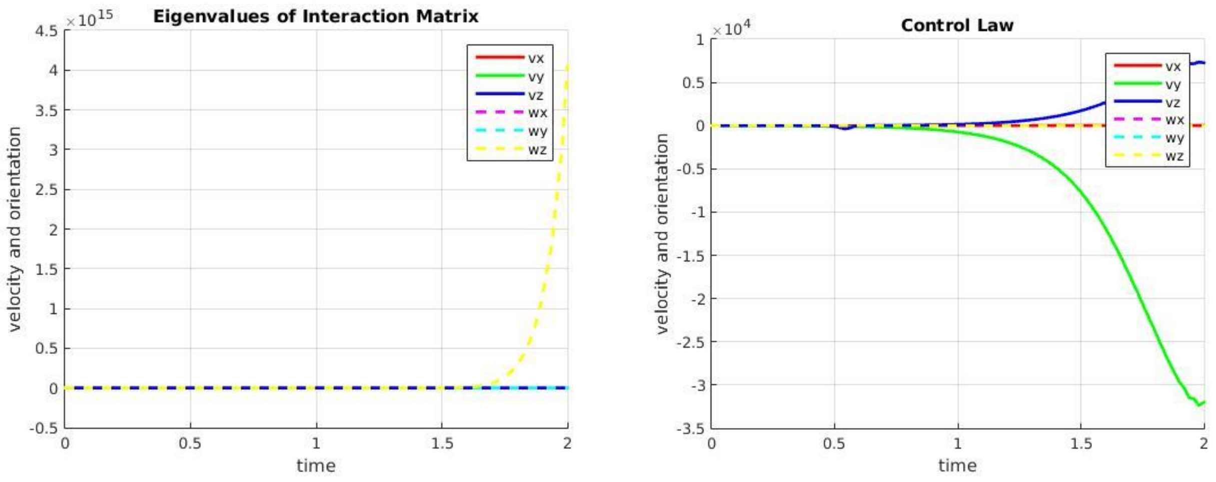
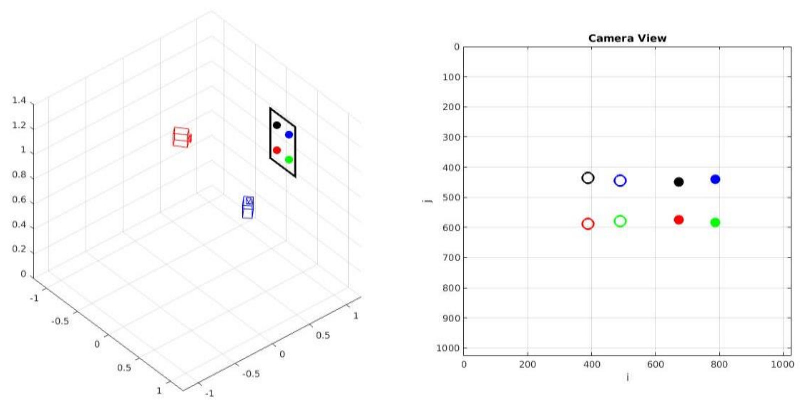


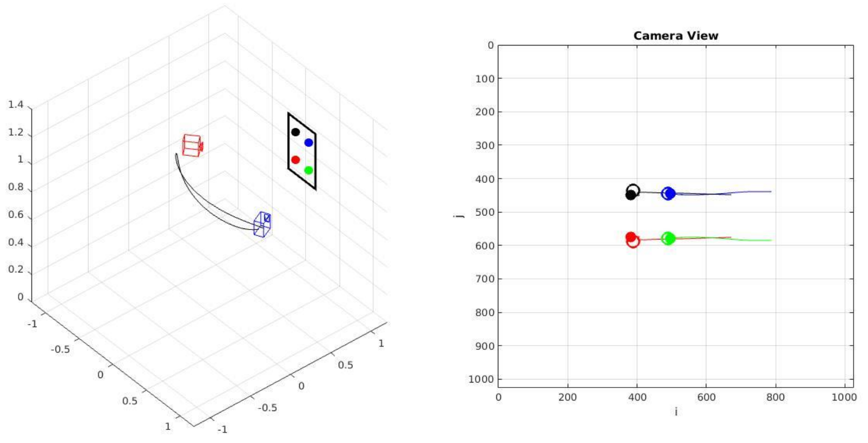
Fig.6: IBVS with Coplanar problem

5.5 IBVS with Local Minimum problem:

The matrix $\mathbf{L}\boldsymbol{L}_s^+$ *belongs to* $\Re$ *of order kxk* is at most of rank 6. With 4 points we have $k = 8$ so it have a non-trivial null space. Thus configurations that corresponds to local minima exists. In the setup (See Fig.7), the simulation runs into a local minima and the control loop was unable to find a smooth path or a valid solution.



a. Initial Camera view



b. After Motion

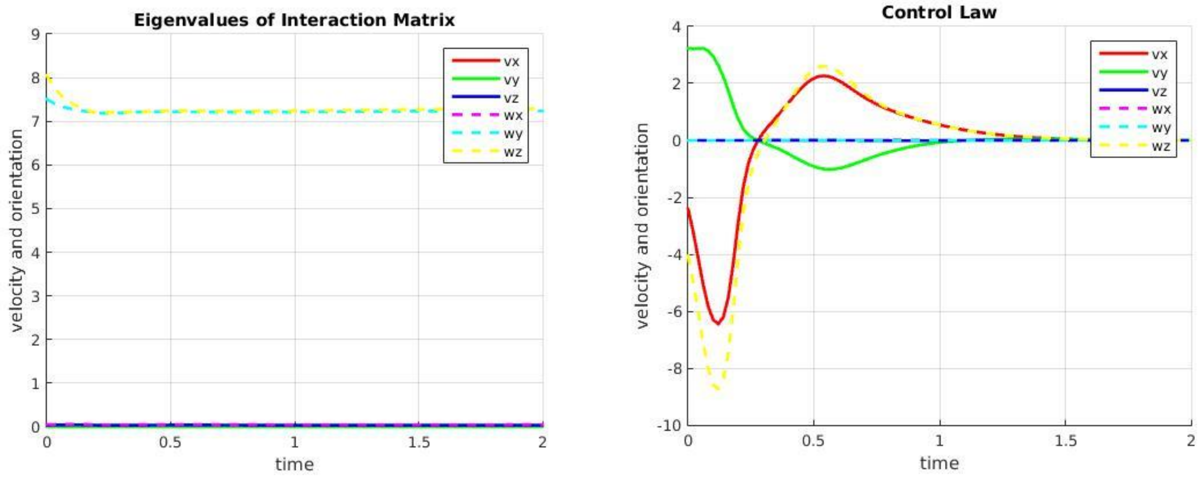
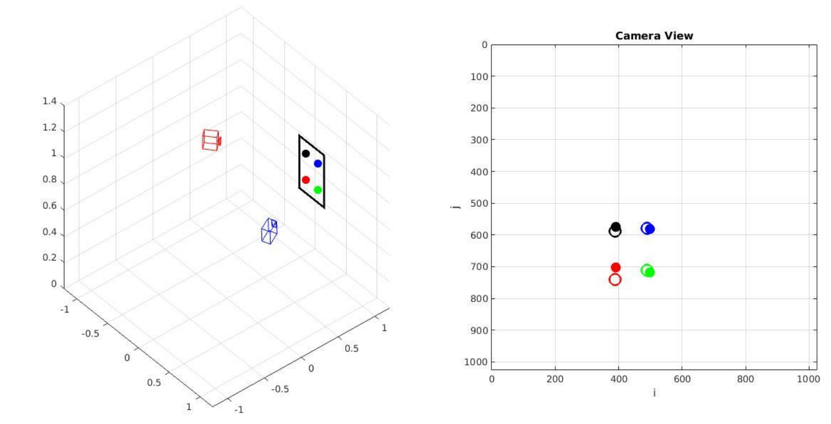


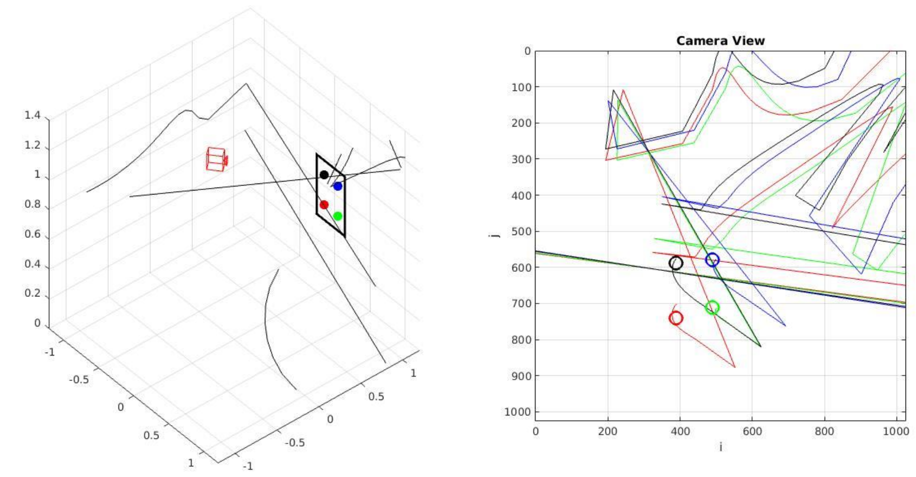
Fig.7: IBVS with Local Minimum problem

5.6 IBVS with Divergence problem:

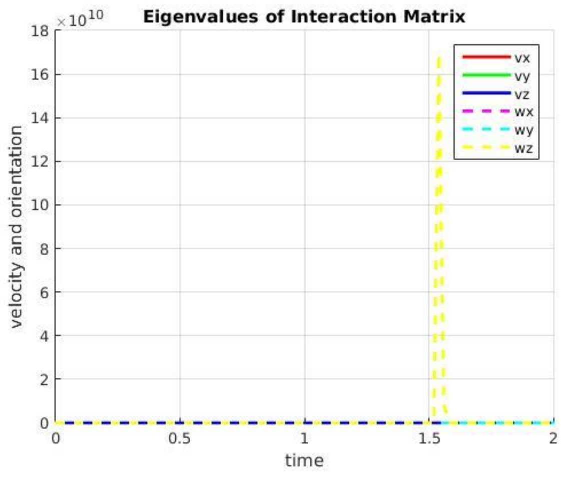
Here a case where the control algorithm is diverging is demonstrated. It has discrete motion with out of field camera view. The simulation results is given at Fig.8



a. Initial Camera view



b. After Motion



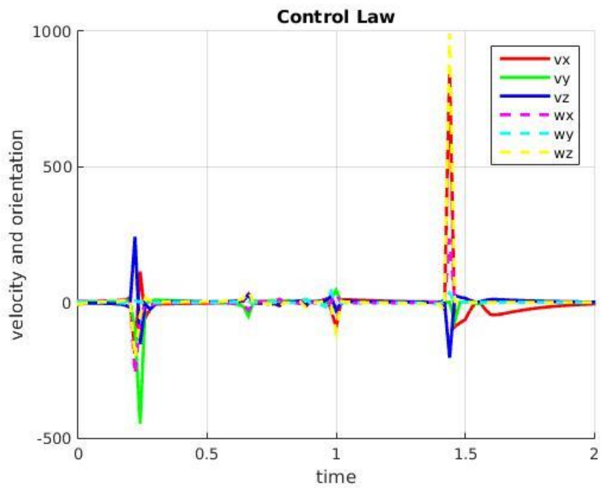
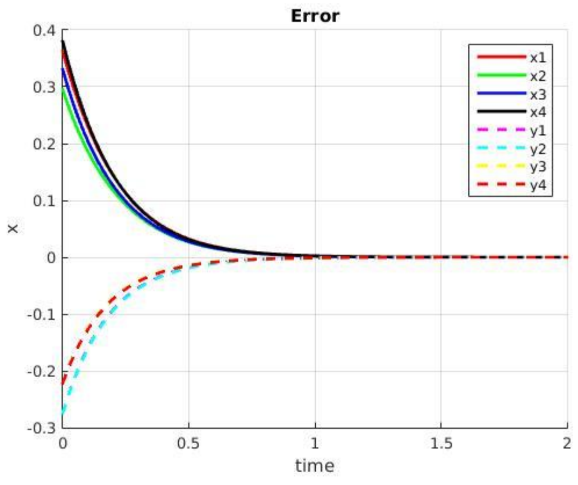
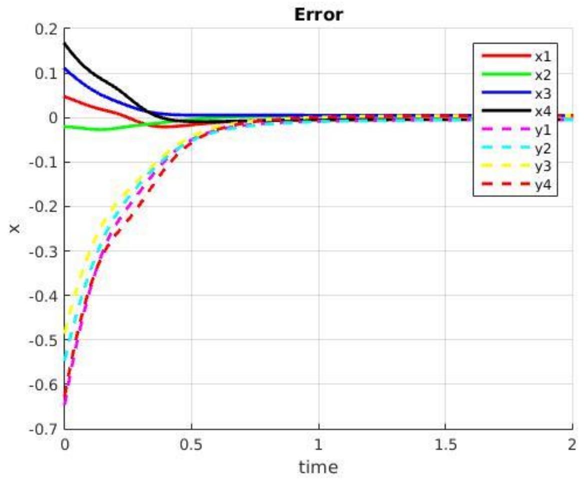


Fig.8: IBVS with Divergence problem

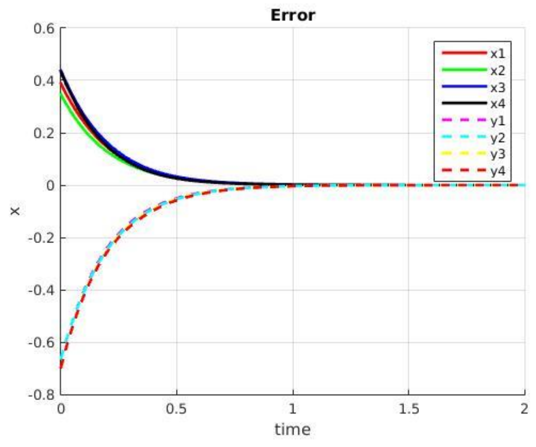
Solution-6: Plotting Errors



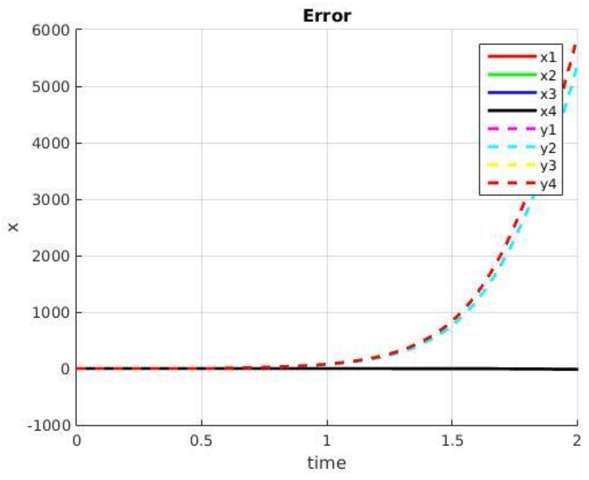
IBVS Error: 4 points



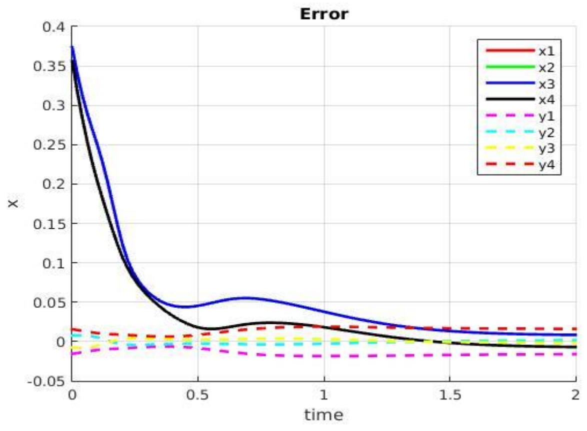
IBVS Error: 3 points



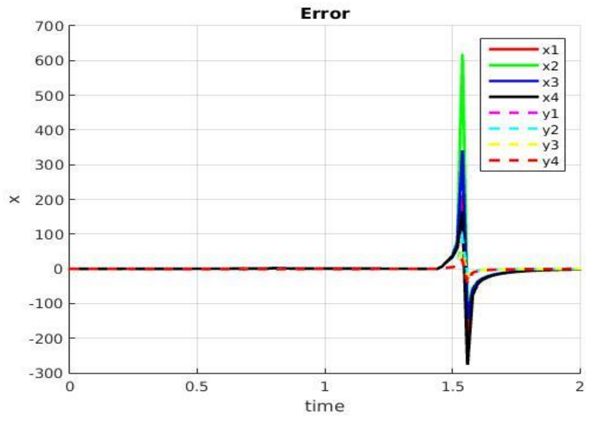
IBVS Error: 2 points



IBVS Error: Coplanar



IBVS Error: Local Minimum



IBVS Error: Divergence

Appendix - Matlab Functions

x = uthetat2dq(u, theta, R, t) gives the dual quaternion x from given rotation unit axis u, rotation angle theta, and translation vector t.

(u, theta, R, t) = dualq2uthetaRt(x) computes the rotation unit axis u, rotation angle theta, rotation matrix R, and translation vector t from given dual quaternion x.

x = muldualpq(p, q) multiplies two given dual quaternions, e.g., p and q, and gives the result dual quaternion, e.g., x.

x^{-1} **= conjdualqsimple(x)** computes the inverse of a given dual quaternion.

plot pose(X, color) plots the 3D frame of a given Cartesian pose X in a defined color, e.g., for red 'r', for blue 'b'.

plot camera(X, color) plots the camera at a given Cartesian pose X with a defined color, e.g., for red 'r', for blue 'b'.

s = skew(t) computes the skew symmetric matrix, e.g., s, of a given vector, e.g., t.

points3D = put pattern(X) puts the pattern to a given Cartesian pose X and returns 4 3D pattern points, e.g., here noted as points3D = [P1;P2;P3;P4] where P = $[X;\ Y; Z]^T$.

plot pattern(points3D) plots the pattern points in the 3D scene.

Conclusion:
IBVS is a useful for controlling a robot using visual information. But as it it does not take into account the position of the camera in its 3D world and have many stability issues. When only one image is used, even small errors in the image measurements can lead to errors in the pose that can impact significantly the accuracy of the system.

Results:

To see the results: RUN:: ibvsTP.m file in the zipped folder

References

[1] *Lecture slide, matlab source code and lab notes of Prof.Omar Tahri*.
[2] Farrokh Janabi-Sharifi, Visual Servoing : Theory and applications, Chapter 15 (book unknown)
[3] F. Chaumette, S. Hutchinso, *Visual Servo Control, Part I: Basic Approaches*. IEEE Robotics and Automation Magazine, 13(4):82-90, December 2006.
[4] F. Chaumette, S. Hutchinson, *Visual Servo Control, Part II: Advanced Approaches*. Addison Wesley, Massachusetts, IEEE Robotics and Automation Magazine, 14(1):109-118, March 2007
[5] S. A. Hutchinson, G. D. Hager, and P. I. Corke, *A tutorial on visual servo control*. IEEE Trans. Robot. Automat., 12(5):651—670, Oct. 1996.
[6] P. I. Corke, *Robotics, Vision and Control: Fundamental Algorithms in MATLAB*. Springer, 2013.
[7] Git-hub: https://github.com/jtsagata/VisualServoingLab

THE END