



A review and performance comparison of visual servoing controls

Vo Duy Cong^{1,3} · Le Duc Hanh^{2,3}

Received: 14 March 2022 / Accepted: 5 January 2023 / Published online: 24 January 2023
© The Author(s), under exclusive licence to Springer Nature Singapore Pte Ltd. 2023

Abstract

Visual servoing is a technique for robot control, in which visual feedback is used in a closed-loop control to improve the accuracy and performance of robot systems. The control tasks in visual servoing are defined to control the robot using visual features extracted from the image. There are various problems when applying Visual servoing such as local minima, singularity, and visibility of feature points. These problems can be solved by using different features or using different control schemes. This paper provides a review of visual servoing for robot manipulators and conducts comparisons of five visual servoing approaches. First, the general theory of visual servoing and five different schemes for the comparison and evaluation are presented. Next, the behaviors of the Visual servoing system depending on the selection of visual features are also presented in detail. In addition, the enhancement and combination schemes are also presented, which are a combination of visual servoing with various control techniques that help increase the robustness of visual servoing. To overcome some issues of the image-based visual servoing scheme, different methods to approximate the interaction matrix are presented. After that, the five visual servoing schemes are simulated on Matlab for performance comparison and evaluation. To conduct the assessment, the simulations are implemented with typical control tasks that are translational movements and rotation around the X, Y and Z axes. The evaluations are conducted with the varying motion parameters and the varying effects of noise in the image. The results of the criteria are visually displayed as 3D charts, from which the reviews and comparisons of schemes are drawn. In addition, the paper also evaluates the schemes when performing general movements. General tasks are simulated by using the PUMA 560 robot. Each task has its own purpose to show issues in some schemes as well as how others overcome them.

Keywords Visual servoing · Vision-based control · Computer vision · Robustness · Global stability

1 Introduction

Nowadays, in conjunction with the non-stop improvement of science and technology, robots increasingly play a crucial role in all regions of life and society, gradually change humans for difficult and dangerous tasks to create products of high quality and precision. Therefore, the requirements

for flexibility, accuracy, and robustness of robots are increasing and robots must have the ability to operate in distinctive environments. To meet such requirements, robots are frequently incorporated with sensors to measure and perceive the surroundings (particularly when the robots operate in unstructured environments). Commonly used sensors are pressure sensors, ultrasonic sensors, lasers, vision sensors, etc. Among them, vision sensors (e.g., cameras) are the most common because vision sensors perceive the environment as the same as humans and allow for non-contact measurements.

Cameras began to be used to control the motion of robots in the 60 s (Shirai and Inoue 1973). Feedback control using visual information from vision sensors (closed-loop control) was only developed in the 90 s to increase the accuracy of the robot system (Hashimoto 1993; Cong and Hanh 2022). This closed-loop control system is called Visual Servoing.

✉ Vo Duy Cong
congvd@hcmut.edu.vn

¹ Industrial Maintenance Training Center, Ho Chi Minh City University of Technology (HCMUT), 268 Ly Thuong Kiet Street, District 10, Ho Chi Minh City, Vietnam

² Department of Mechatronics, Faculty of Mechanical Engineering, Ho Chi Minh City University of Technology (HCMUT), 268 Ly Thuong Kiet Street, District 10, Ho Chi Minh City, Vietnam

³ Vietnam National University Ho Chi Minh City, Linh Trung Ward, Thu Duc City, Ho Chi Minh City, Vietnam

Visual servoing (VS) is a feedback control technique that uses visual information to enhance the accuracy and versatility of robot systems (Cong and Hanh 2022; Chaumette et al. 2016; Zhao et al. 2021). The control tasks in visual servoing are defined to manipulate the movements of robots using visual features extracted from images of interesting objects in real-time (Li et al. 2018; Cong and Hanh 2021). The error function in VS is defined as the error of visual features at the current and the desired position of the robot/camera. The purpose of a visual control schemes is to regulate the error and make it zeros. One, two or more cameras can be used to gain visual information of the objects for the robot control task (Cong and Hanh 2022, 2021; Zhao et al. 2021; Wu et al. 2022).

There are two main configurations for combining camera and robot in visual servoing applications. The first configuration is eye-in-hand, the camera is mounted on the end-effector of the robot and moves with the robot. A transformation between the camera's frame and the end-effector's frame is defined to convert motions from the camera's frame to the end-effector's frame. In the second configuration, called eye-to-hand, one or more cameras are fixed in the workspace to observe both the robot and the objects. This configuration needs to calculate the transformation matrix between the camera and robot coordinate frame at each iteration. The hybrid configuration can use both eye-in-hand and eye-to-hand configurations.

Two main control schemes that are commonly applied in visual servoing controls are position-based visual servoing (PBVS) scheme and image-based visual servoing (IBVS) scheme (Cong and Hanh 2022; Zhao et al. 2021; Wu et al. 2022; Sanderson and Weiss 1980). The combination of two control schemes creates a hybrid scheme called 2-1/2D visual servoing (Malis et al. 1999). In the PBVS system, the control inputs are calculated in the 3D space. The 3D coordinates of the objects are estimated from the image features. There are different methods for determining the position of an object [see (Dementhon and Davis June 1995)]. All methods require knowing the complete geometry of the objects and the camera's calibration parameters. In the IBVS system, the control inputs are directly calculated in the 2D image space. This method is robust to the camera and robot calibration errors (Espiau 1993; Malis et al. 2010; Xiaolin and Hongwen 2020). However, convergence is only guaranteed in a small region (completely impossible to analyze to determine) around the desired location. An expressive graph to illustrate the VS is shown in Fig. 1.

Two essential aspects that significantly affect the behaviour of a visual servoing system are the choice of visual features to be used as input by the controller and the control schemes. With the same set of features, the system has different behaviours when used in different control schemes, and the same control rule gives different

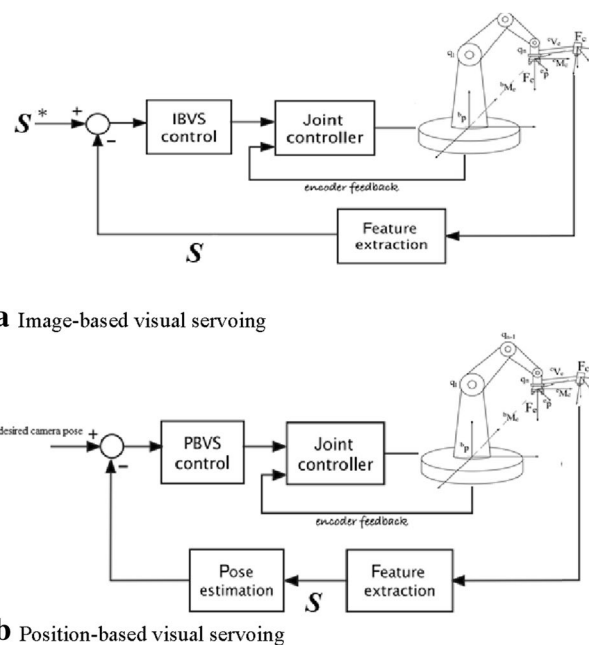


Fig. 1 Two visual servoing control schemes

behaviours when considering different features. The behaviour acquired with the combination of two choices is often not as expected, choosing a particular set of features or a particular control scheme can result in some stability and convergence problems.

This paper reviews common techniques in visual servoing control and makes assessments based on the efficiency and problem-solving ability of control schemes. The popular schemes are simulated on Matlab to obtain performance indicators and behavior graphs. To conduct the assessment, the paper gives typical control tasks that are translation and rotation around the X, Y and Z axes. The evaluations are conducted with the varying motion parameters and the varying effects of noise. After all the tests have been performed, the main conclusions drawn from them are presented and discussed.

2 Review of visual servoing

2.1 General theory of visual servoing

In visual servoing control, visual features are extracted from the image to control the motion of the robot (Ren et al. 2020; Zhong et al. 2019; Chwa 2020). The control rule is designed so that the current visual features $s(t)$ are equal to the desired features s_d . Therefore, all visual servoing tasks are aimed at eliminating errors $e(t)$ is defined as (Hanh and Cong 2022):

$$\mathbf{e} = \mathbf{s}(t) - \mathbf{s}_d \tag{1}$$

In visual servoing, the relationship between the derivative of the vector \mathbf{s} and the relative velocity between the camera and the object is determined by the interaction matrix:

$$\dot{\mathbf{s}} = \mathbf{L}_s^c \mathbf{v}_o + \frac{\partial \mathbf{s}}{\partial t} \tag{2}$$

where $\partial \mathbf{s} / \partial t$ is the derivative of \mathbf{s} caused by the object's own motion, $\mathbf{L}_s \in \mathbf{R}^{k \times n}$ is the interaction matrix and ${}^c \mathbf{v}_o$ the relative velocity between the camera and the object.

In the case of a non-moving object, $\partial \mathbf{s} / \partial t = 0$, we have:

$$\dot{\mathbf{e}} = \dot{\mathbf{s}} = \mathbf{L}_s \mathbf{v}_c \tag{3}$$

To ensure that the error of the feature vector is suppressed, the following first-order control law is used:

$$\dot{\mathbf{e}} = \mathbf{K} \mathbf{e} \tag{4}$$

where \mathbf{K} is a constant gain matrix. Combined (4) with (1) and (3) we get:

$$\mathbf{v}_c = -\mathbf{K} \mathbf{L}_s^+ \mathbf{e} = -\mathbf{K} \mathbf{L}_s^+ (\mathbf{s} - \mathbf{s}_d) \tag{5}$$

where $\mathbf{L}_s^+ = (\mathbf{L}_s^T \mathbf{L}_s)^{-1} \mathbf{L}_s^T$ is the pseudo-inverse of the interaction matrix.

2.2 Visual servoing control schemes

Control schemes are mainly different in the way that visual information is used. Different control rules (Wang et al. 2017; Ren et al. 2020; Liu et al. 2020; Malis Feb. 2004) will have different effects on the response of the system. This section presents visual servoing control schemes, starting from traditional control schemes to hybrid schemes and enhanced control schemes. To form control rules for visual servoing schemes, 2D pixel coordinate or 3D coordinates are used. Other visual features will be detailed in the next section.

2.2.1 Position-based visual servoing (PBVS)

In PBVS the error signal is computed in the 3D Cartesian space. The state vector is the pose of the robot $\mathbf{P} = [\mathbf{t}_c^d, \theta \mathbf{u}]^T$, where \mathbf{t}_c^d is the translation vector from the current position to the desired position, determined in the current camera frame \mathbf{F} , $\theta \mathbf{u}$ is the vector/angle representation of the orientation. We have the relationship:

$$\dot{\mathbf{P}} = \mathbf{L}_S \mathbf{v}_c = \begin{bmatrix} \mathbf{R} & 0 \\ 0 & \mathbf{L}_W \end{bmatrix} \mathbf{v}_c \tag{6}$$

where \mathbf{R} is the rotation matrix, and \mathbf{L}_W is the interaction matrix of rotation motion

$$\mathbf{L}_W = \mathbf{I}_3 - \frac{\theta}{2} [\mathbf{u}]_{\times} + \left(1 - \frac{\text{sinc} \theta}{\text{sinc}^2 \frac{\theta}{2}} \right) [\mathbf{u}]_{\times}^2,$$

\mathbf{L}_W^{-1} can be approximated by the identity matrix.

Assume $\mathbf{v}_c = (\mathbf{v}_t, \boldsymbol{\omega}_c)^T$ where \mathbf{v}_t is the translation.

velocity and $\boldsymbol{\omega}_c$ is the angular velocity of the camera.

From (5) we have:

$$\begin{cases} \mathbf{v}_t = -\mathbf{K}_t \mathbf{R}^T \mathbf{t}_c^d = -\mathbf{K}_t \mathbf{t} \\ \boldsymbol{\omega}_c = -\mathbf{K}_\omega \theta \mathbf{u} \end{cases} \tag{7}$$

where $\mathbf{t} = (t_x, t_y, t_z)^T$ is the translation vector from the current camera \mathbf{F} to the desired camera frame \mathbf{F}^* , determined in the desired camera frame.

2.2.2 Image-based visual servoing (IBVS)

Let $\mathbf{P}_i = (X_i, Y_i, Z_i)^T$ is the 3D coordinate of a feature point and the image coordinate of \mathbf{P}_i is $\mathbf{p}_i = (x_i, y_i)^T$. Using perspective projection in the pinhole camera model, we have:

$$\begin{aligned} x_i &= f \frac{X_i}{Z_i} \\ y_i &= f \frac{Y_i}{Z_i} \end{aligned} \tag{8}$$

where f is the focal length. The velocity of point \mathbf{P}_i relates to the camera velocity by the equation:

$$\dot{\mathbf{P}}_i = -\mathbf{v}_c - \boldsymbol{\omega}_c \times \mathbf{P}_i \tag{9}$$

Taking the time derivative of Eq. (8) and combining it with Eq. (9), we obtain:

$$\dot{\mathbf{p}}_i = \mathbf{L}_{si} \mathbf{v}_c \tag{10}$$

where:

$$\mathbf{L}_{si} = \begin{bmatrix} \frac{-1}{Z_i} & 0 & \frac{x_i}{Z_i} & x_i y_i & -(1+x_i^2) & y_i \\ 0 & \frac{-1}{Z_i} & \frac{y_i}{Z_i} & 1+y_i^2 & -x_i y_i & -x_i \end{bmatrix} \tag{11}$$

Define feature vector $\mathbf{s} = (\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_n)^T$, n is the number of feature points. The interaction matrix is obtained by stacking the interaction matrices of each feature point determined by (11):

$$\mathbf{L}_S = (\mathbf{L}_{s1}, \mathbf{L}_{s2}, \dots, \mathbf{L}_{sn})^T \in \mathbf{R}^{2n \times 6} \tag{12}$$

There is a commonly mentioned issue when applying IBVS called ‘‘camera retreat’’ in which the camera moves away from the target in a normal direction and then returns (back-and-forth movements) as shown in Fig. 2.

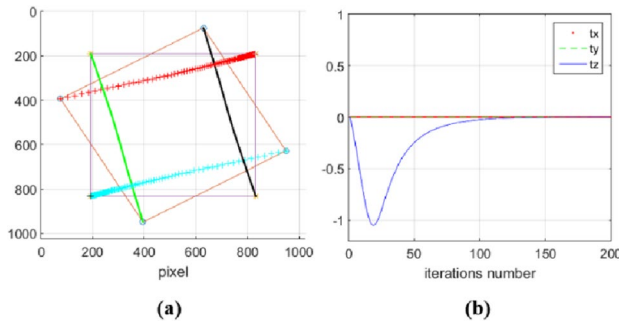


Fig. 2 Camera retreat problem. **a** image feature trajectories. **b** translation along the X, Y and Z axes

2.2.3 2 1/2 D Visual Servoing (2.5DVS)

The 2 1/2 D visual servoing control scheme was proposed by Malis (Malis et al. 1999) in 1999 with the aim of combining the advantages of both IBVS and PBVS schemes. Hence this method is also known as hybrid visual servoing. Different from the two control diagrams above, the 2.5DVS scheme separates the camera translation and rotation control laws from each other. The axis-angle parameters θu obtained from the rotation matrix are used to calculate angular velocity commands (Malis et al. 1999):

$$\frac{d(\theta u)}{dt} = \begin{bmatrix} 0 & L_w \end{bmatrix} v_c \quad (13)$$

For calculating the translation velocities, define the extended image features $m_e = (x, y, \log(Z))^T$. According to Malis et al. (1999), we have:

$$\dot{m}_e = \begin{bmatrix} \frac{1}{d^*} L_v & L_{v\omega} \end{bmatrix} v_c \quad (14)$$

where:

$$L_v = \frac{1}{\rho_1} \begin{bmatrix} -1 & 0 & x \\ 0 & -1 & y \\ 0 & 0 & -1 \end{bmatrix} \quad (15)$$

$$L_{v\omega} = \begin{bmatrix} xy & -(1+x^2) & y \\ 1+y^2 & -xy & -x \\ -y & x & 0 \end{bmatrix}$$

d^* is the distance from the camera at the desired position to a fixed plane π containing the feature points, ρ_1 is a coefficient defined as (Malis et al. 1999):

$$\rho_1 = \frac{Z}{d^*} \quad (16)$$

Define feature vectors $s = (m_e, \theta u)^T$ and $s_d = (m_{ed}, 0)^T$. The control task is to minimize the error

$e = ((m_e - m_{ed})^T, \theta u^T)^T$. The derivative of the error and the camera's velocity is related by (3) with the interaction matrix (Malis et al. 1999):

$$L_S = \begin{bmatrix} \frac{1}{d^*} L_v & L_{v\omega} \\ 0 & L_w \end{bmatrix} v_c \quad (17)$$

and the velocity of camera obtained from (5) is (Malis et al. 1999):

$$v_c = -K \begin{bmatrix} d^* L_v^{-1} & -d^* L_v^{-1} L_{v\omega} \\ 0 & I_3 \end{bmatrix} \begin{bmatrix} x - x_d \\ y - y_d \\ \log(Z/Z_d) \\ \theta u \end{bmatrix} \quad (18)$$

The 2.5D VS separates the camera translation and rotation control laws from each other, so it can avoid “camera retreat” problem as in the coupled control law of IBVS.

In Eq. (18), the ratio ρ_1 , the ratio Z/Z_d and the vector/angle parameter θu is estimated by using the homography matrix. The homography matrix can be decomposed (Malis et al. 1999):

$$H = R + t_d n^{*T} \quad (19)$$

where R is the rotation matrix between the camera's current frame F and the camera's desired frame F^* , n^* is the unit vector normal to plane π expressed in frame F^* , and t_d is defined as t/d^* , t being the translation vector between F and F^* . The ratio ρ_1 is computed as (Malis et al. 1999):

$$\rho_1 = \frac{\det(H)}{(Rn^*)^T m} \quad (20)$$

and the ratio Z/Z_d is determined as (Malis et al. 1999):

$$\frac{Z}{Z_d} = \rho_1 n^{*T} m \quad (21)$$

with m is the normalized coordinates of an image point.

2.2.4 Partitioned visual servoing (PVS)

The 2 1/2 D visual servoing control scheme described above is designed to separate rotation from translational motion, by selecting features in both 2D and 3D spaces. Another scheme proposed by Corke and Hutchinson (Corke and Hutchinson 2001) also decouples displacements in the Z direction using only features extracted directly in the image with the aim of solving some problems in IBVS, mainly the “camera retreat” problem that occurs when a rotation around the optical axis is required. Perform decoupling Z-axis motion:

$$\dot{s} = L_s v_c = L_{xy} v_{xy} + L_z v_z \quad (22)$$

where the matrix L_{xy} includes the 1st, 2nd, 4th, and 5th columns of the L_s matrix and the L_z matrix contains the 3th and 6th columns of the L_s matrix.

Since v_z is calculated separately, from Eq. (22) the velocity in the x and y directions can be determined:

$$v_{xy} = L_{xy}^+ (-Ke - L_z v_z) \tag{23}$$

The velocities in the Z direction are determined using angle feature and area feature extracted from the image:

$$\begin{aligned} \omega_z &= k_{\omega_z} (\theta_m^* - \theta_m) \\ v_{tz} &= k_{v_z} (\epsilon^* - \epsilon) \end{aligned} \tag{24}$$

2.2.5 Shortest path visual servoing (SPVS)

Shortest Path Visual Servoing was presented by Kyrki and Kragic (Kyrki et al. 2004) with the aim of solving two fundamental problems in visual servoing: the possibility of objects coming out of the camera’s field of view and the joints getting too close to the allowed limit. This method is designed to ensure that the camera trajectory is a straight line from the starting point to the desired position by using the position-based control for the translation:

$$v_t = -K_t t \tag{25}$$

The rotation angle around the z-axis also remains the same control law:

$$\omega_z = k_{\omega_z} u_z \theta \tag{26}$$

And the rotation around the x and y axes will be controlled by using a single feature point in the image to ensure the visibility of target points:

$$[\omega_x, \omega_y]^T = L_{\omega_{xy}}^{-1} (-Ke - L_t v_t - L_{\omega_z} \omega_z) \tag{27}$$

where $L_v, L_{\omega_z}, L_{\omega_{xy}}$ are the interaction matrix for translation, rotation around Z and X, Y axes, respectively.

2.3 Selection of visual features

The behavior of the VS system depends on the selection of visual features. The visual features are observed by the visual sensor and generate input for the control scheme. A visual sensor provides potential visual features, but if the features are not selected properly, it will lead to stability problems and the robot’s displacement will be large (Ren et al. 2020). Therefore, the selection of visual features in VS is important because it determines the speed, accuracy and reliability of the visual measurements. And so, determines the accuracy and robustness of the VS system (Ren

et al. 2020; Feddema et al. 1991; Janabi-Sharifi and Wilson 1997). For this reason, it is necessary to design the best visual features for VS systems that meet the following criteria: avoid local stability, robustness with calibration error and modeling error, non-singularity, the reasonable trajectory of the camera and of the features in the image and finally, maximum decoupling of degrees of freedom and for a linear relationship between the visual features and the controlled degrees of freedom.

2.3.1 Geometric features

The most common visual features are geometric features (e.g., points, lines, circles, etc.). Geometric features are defined to describe the geometric content in an image (2D visual feature) or the relationship between a coordinate system attached to the robot system and the coordinate system attached to the object (3D visual feature). Both 2D and 3D features can be used at the same time in hybrid diagrams.

2D visual features are often extracted from 2D images such as points, lines, ellipses, areas of interest or contours (Feddema et al. 1991; Janabi-Sharifi and Wilson 1997; Shi and Tomasi 1994; Andreff et al. 2000). These features are extracted by image processing algorithms. In the case of a feature point, Cartesian coordinates are often used, but polar or cylindrical coordinates can also sometimes be used (Chaumette Aug. 2004).

In addition, image moments and moment invariants can also be used in VS (Chaumette 2002, 2004; Tahri and Chaumette 2003, 2004, 2005, 2005; Zhao et al. 2013). Using image moments has many outstanding advantages over traditional VS, as it allows a general representation that not only allows solving for basic geometric objects but also for complex objects with unspecified shapes. In (Chaumette 2004) the use of image moments is discussed to form the expression of the Jacobian visual matrix. This expression allows separate degrees of freedom based on the selected moment type. And in Tahri and Chaumette (2005) shows the use of moment invariants to design a decoupled 2D visual servoing control scheme.

Visual features can be selected in 3D space such as the position or coordinates of the 3D point (Martinet et al. 1996; Deng et al. 2003; Wilson et al. 1996). The model of the object and the measurement in the image are often used to estimate the relative position of the object relative to the camera. In (Cervera et al. 2003), the 3D coordinates of the object are used as feature vectors and it is required to know the camera calibration parameters in advance. In PBVS, an object’s orientation can be represented by roll-pitch-yaw or axis-rotation (Wang and Wilson 1992) or quaternions (Hu et al. 2010).

Several combinations of feature types can be considered, for example, the combination of both 2D and 3D

features are presented in Wilson et al. (1996); Deng et al. 2002; Marchand and Chaumette June 2005) and the combination of polar coordinates and Cartesian coordinates of pixels are presented in Corke et al. (2009).

2.3.2 Photometry features

Contrary to using geometric features in VS, recently photometric features calculated from the luminance of pixels have been used in VS. Using photometry features does not require complex image processing such as feature extraction, matching and tracking. Furthermore, the effect of some parts being masked and the depth estimation is not great. This approach is implemented by considering the entire image as a set of features to calculate the control input signal (Nayar et al. 1996; Deguchi 2000). The control input can belong to the eigenspace or the kernel of the pixel, or it can also be defined as the set of all pixels.

In (Nayar et al. 1996), pixel intensity is not used directly, but eigen spatial decomposition is performed first to reduce the dimension of the image data. Then, control is performed in eigenspace, not directly from the pixels' intensity. Furthermore, this approach requires computing off-line the eigenspace and performing image projection on this subspace for each new frame. An interesting approach, also considering pixels' intensity, was recently proposed in Kallem et al. (2007). This approach is based on the use of kernel methods resulting in highly decoupled control laws. It is also possible to use the brightness of all pixels in the image as a visual feature set (Collewet et al. 2008; Collewet and Marchand 2009a, b).

2.3.3 The velocity field features

The velocity field in the image is used as the visual feature in Dong and Zhang (2020), and the relationship between the change of the velocity field of features and the velocity of the camera is modelled. This method is used to drive the camera to the position parallel to a plane and follow a trajectory. The camera's movement is controlled so that the velocity field in the image is equal to the velocity field at the desired location (Xu et al. 2018). In (Kelly et al. 2004), the velocity field is used in the VS of the robotic arm under the fixed sphere of the camera (Fig. 3). The desired velocity field v_d is defined in the image space, which is a tangent vector representing the desired image velocity feature \dot{x} at each point of the image space. The velocity field error is defined as the difference between the desired velocity field v_{dx} and the image velocity feature \dot{x} . The velocity field is also used in Kelly et al. (2006) to control a mobile robot with a fixed camera.

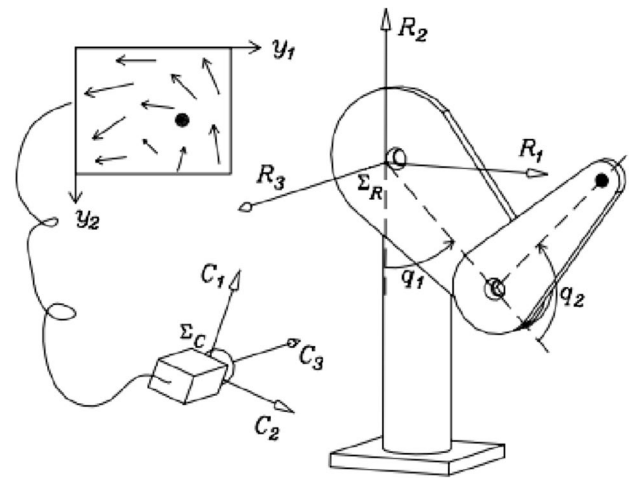


Fig. 3 Visual servoing using velocity field (Kelly et al. 2004)

2.4 Combined and enhanced control schemes

To increase system performance and overcome the disadvantages of visual servoing control schemes, in addition to using advanced control schemes (2.5D VS, Partitioned VS, Shorted Path VS), VS control schemes can be combined with each other or combined VS with other controllers to take advantage of the controllers and overcome the disadvantages of VS scheme for improving system performance.

Some studies have been done in this direction, such as combining VS with sliding mode control (Zanne et al. 2000), partitioning approaches (Corke and Hutchinson 2521; Pages et al. 2006; Gans et al. 2003), planning approaches (Shu et al. 2018; Keshmiri et al. 2017), switching schemes (Gans and Hutchinson 2002, 2007; Norouzi-Ghazbi and Janabi-Sharifi 2021; Zhao et al. 2017; Ghassemi et al. 2019, 2020)

2.4.1 Switching approaches

One of the ways to use multiple controllers in combination is to use a switching scheme, in which one controller is selected at a time depending on which criteria need to be optimized (Norouzi-Ghazbi and Janabi-Sharifi 2021; Abhilash and Ashok 2016). To implement this scheme, two levels of control strategy are required: the lower level is used to implement the VS control rules, and the higher level of control strategy determines which control scheme should be applied.

A switching system is represented by the differential equation:

$$\dot{x}(t) = f_{\sigma(t)}(x, t, u_{\sigma(t)}) : \sigma \in \{1 \dots n\} \quad (28)$$

where $f_{\sigma(t)}$ is the set of n discriminant functions. The switching is directly affected by the choice of the control input u . Each visual servoing control scheme provides a controlled velocity $u = [T_x, T_y, T_z, \omega_x, \omega_y, \omega_z]^T$ and a switching rule determines the actual control input used at each control cycle. The stability of the switching system is not guaranteed by the stability of the individual controllers. A set of stable controllers may become unstable if an inappropriate switching rule is applied, and unstable systems may become asymptotically stable if switching approaches are applied. Therefore, the overall stability of the system is difficult to guarantee, however, under certain conditions, the stability of the system can be demonstrated (Malis et al. 1999; Dementhon and Davis 1995).

Nicholas R. Gans (Gans and Hutchinson 2007) implemented a switching system between two control laws PBVS and IBVS based on the value of the Lyapunov function. The system starts with the Gan's IBVS controller, considering the Lyapunov function for the PBVS controller defined by $L = \frac{1}{2} \|e(t)\|^2$. At any point in time, if the value of the Lyapunov function exceeds the threshold γ_p , the system switches to the PBVS scheme. And in the process of using PBVS, if at any time, the value of the Lyapunov function for IBVS exceeds the threshold γ_i , the system switches to the IBVS scheme. If the thresholds are chosen appropriately, the system can obtain the relative advantages of IBVS and PBVS and limit the shortcomings.

Gan (Gans and Hutchinson 2002) implemented the switching scheme between the two controllers based on homography matrix and affine approximation. The homography-based scheme can be used in the case of general motion, including rotations around the x and y axes. If the motion does not include rotations around the x and y axes, the performance of the two methods is the same. However, when noise is present, the affine method is more efficient. Three different switching rules are implemented and compared: "Deterministic Switching", "Random Switching" and "Biased Random Switching". The result shows that the Biased Random rule gives better results than other switching rules.

2.4.2 Task sequencing

The original approach, at the beginning of studies of visual servoing control, constrained all degrees of freedom of the robot in one task. However, in the early stages of the control process, this is not necessary. The typical situation is that some features reach the desired position before others. In addition, the classical methods of choosing a trajectory may not be suitable, and not optimal.

A more efficient way to control the system is that uses some of the robot's degrees of freedom to perform secondary tasks when the robot is very far from the target. These

secondary tasks can improve control system robustness, including avoiding joint limits, ensuring that the targets remain within the camera's field of view, or avoiding collision with obstacles (Alatartsev et al. 2015; Ahmadi et al. 2022). When the robot approaches the targets, all the degrees of freedom of the robot are controlled to reach the desired position. This approach is called task sequencing (Kurtser and Edan 2020; Diyaley et al. 2020).

To use this approach, it is necessary to ensure that all sub-tasks have different priorities and the sub-tasks do not affect the main task. To do that, redundancy formalism is used (Mansard and Chaumette 2004, 2005).

The task sequencing approach proposes a solution to increase the robustness of the system by dividing the global task into multiple subtasks. This is done by adding and removing those subtasks from a stack, according to the conditions of the environment.

To build an environment-adaptive system, which may include trajectory constraints, add more subtasks to the stack. This can be done using the cost function, to determine the safe position of the robot. This function can be defined in the joint space and has a high value for dangerous situations and low for safe situations. Using predictive control, it is possible to estimate when the system is in danger with actual task stacks and change them to remediate. If the cost function exceeds a certain threshold value, then performing a task stack change. This method has been successfully implemented and tested by Mansard on a real robot and described in Mansard and Chaumette (2004) and Mansard and Chaumette (2005).

2.4.3 Feature trajectory planning

In order to fulfill the requirements of obstacle avoidance, joint angle limitation or collision avoidance, it is possible to create a trajectory for the robot and use visual servoing to control the robot to follow the established trajectory (Hosoda et al. 1995; Mezouar and Chaumette 2002; Dejun and Kam 2020). Constraints can integrate concurrently. Trajectories of features s_d allow the camera to reach the desired position while ensuring that constraints are satisfied by using path planning techniques, such as the famous "potential field" method (Zhao et al. 2020).

Separating trajectory planning from tracking allows a significant improvement in the robustness of the visual servoing to modeling errors. Indeed, modeling errors have a large effect when the error $s - s_d$ is large, but a small effect when $s - s_d$ is small. When the desired feature point trajectory $s_d(t)$ satisfying $s_d(0) = s(0)$ has been designed in the planning stage, the scheme can be adapted to the actual requirement of the changes of s_d , and makes the error $s - s_d$ remain small. More precisely, we have:

$$\dot{e} = \dot{s} - \dot{s}_d = L_e v_c - \dot{s}_d \quad (29)$$

From that, using the control law $\dot{e} = -\lambda e$, the velocity can be obtained:

$$v_c = -\lambda \widehat{L}_e^+ e + \widehat{L}_e^+ \dot{s}_d \quad (30)$$

The second component of this control law predicts the variation of s_d , eliminating the tracking error.

2.5 Problems in visual servoing

Choosing a suitable set of visual features and designing good control schemes should be considered to avoid errors when performing visual servoing. Very common problems in visual servoing control tasks are directly affected by this choice, which includes local minima, singularity, and visibility of visual features.

2.5.1 Local minima

In general, the local minima problem only occurs with specific configurations. When stuck in a local minimum, the camera velocity is zero while the errors of the features remain uncorrected to zero. This results in converging to a final position that is different from the desired position. When the feature vector s is generated from three points in the image and L_S has full rank, then we have $\text{Ker}(L_S^+) = 0$, implying that there are no local minima. However, when using three points, the same images of three points can be seen from four different camera positions, which means that exist four camera positions such that $s = s_d$, corresponding to four global minimums. When using four points, there is theoretically only one position of the camera. However, $\dim(L_S) = 8 \times 6$, implies that $\dim \text{Ker}(L_S) = 2$. Using four points, the control law tries to control 8 constraints on the image trajectory while the system has only six degrees of freedom. In that case, due to the existence of impracticable motions in the image computed by the control law, the system may reach a local minimum.

Several control strategies have been developed to avoid local minima in visual servoing such as using hybrid schemes or trajectory planning.

2.5.2 Singularity

When the interaction matrix becomes singular, the camera velocity tends to infinity thereby causing instability of the system. The singularity can appear when selected pixels are image features. For example, when four points are used and the required motion of the camera is to rotate around the optical axis at an angle of 180° , the feature point trajectories

will be straight lines passing through the center of the image and the Jacobian matrix becomes singular. For this movement, it is not appropriate to use pixel coordinates. If these points are replaced by cylindrical coordinates (ρ, θ) , the singularity may not occur when performing a rotation of 180° around the optical axis.

In PBVS, problems of local minima and singularity can be avoided depending on the choice of the error e , a straight line from the starting position to the desired position is obtained when the error is defined in the desired camera system. The singularity can be avoided when using partitioned visual servoing, 2 1/2 D visual servoing, switching, and PBVS scheme.

2.5.3 Visibility of the features

Using 2D and 3D visual servoing schemes with poor calibration and an initial position far from the desired position, the target can be out of the camera's field of view. So, visual servoing control rules must be designed to keep features in the camera's field of view resulting in reliable feedback during visual servoing. To minimize the probability of features leaving the FOV, a "repulsive potential field" can be applied, creating a strategy for path planning, using a schematic transformation map as well as using structure light.

2.6 Interaction matrix approximation for IBVS

When using IBVS, the following stability and convergence problems are encountered:

- The system reaches a local minimum far from the desired position. This happens when the interaction matrix has an incomplete rank.
- The interaction matrix becomes singular and leads to an unstable system.
- Unnecessary back and forth movements of the camera when performing rotation around the optical axis to ensure that the trajectory of the feature points is a straight line.
- When asked to perform a rotation around the optical axis at an angle of 180° , the camera performs an infinity retreat.

In this section, methods for approximating the interaction matrix are presented to solve the above problems and analyze the advantages and disadvantages of each method.

Constant Jacobian matrix: \widehat{L}^+ has a fixed value and is equal to the pseudo-inverse of the interaction matrix at the desired position. Therefore, it is only necessary to define the features and depth at the desired position to calculate the interaction matrix. The notation L^* is the interaction matrix

at the desired position, if L^* has a full rank, the camera's velocity can be calculated by:

$$v = -K(L^*)^+ e(t) = -K\left((L^*)^T L^*\right)^{-1} (L^*)^T e(t) \tag{31}$$

With this method, the stability of the system is only guaranteed in a small area around the desired location. This area is difficult and complex to define. Trajectories of feature points in the image are not straight lines and are difficult to determine in advance. Therefore, some features may be out of the camera's field of view, especially when the start position is far from the desired position.

Varying Jacobian matrix: the depth $z(t)$ of each feature is estimated if the 3D model of the object is known in advance or from the camera motion measurement. Therefore, the interaction matrix can be calculated using expression (11). Notation $L' = L(s(t), \widehat{z}(t))$, if the matrix L' has full rank, the camera speed can be calculated by:

$$v = -K(L')^+ e(t) = -K\left(L'^T L'\right)^{-1} L'^T e(t) \tag{32}$$

The trajectories of the feature points of this method are straight lines from the initial position to the desired position. However, the camera can reach local minima and the interaction matrix becomes singular.

Pseudo-inverse of the mean of the Jacobians: The approximation of the interaction matrix is obtained by averaging the two matrices in the above method:

$$\widehat{L} = \frac{L' + L^*}{2} \tag{33}$$

In general, the rank of the average matrix is not related to the rank of the two matrices L' and L^* . If this matrix has a full rank, the camera speed is calculated by:

$$v = -K\left(\frac{L' + L^*}{2}\right)^+ e(t) \tag{34}$$

This scheme shows good performance for translation and rotation movements around the camera's optical axis without any retreating motion. However, the camera goes to infinity when the requirement movement is a rotation of 180° around the optical axis. Because there is no backward motion, the trajectories of the features are curves with great curvature, and some of the features may be out of the camera's field of view.

Mean of the Jacobian pseudo-inverses: This method directly approximates the pseudo-inverse matrix:

$$\widehat{L}^+ = \frac{(L')^+ + (L^*)^+}{2} \tag{35}$$

and the camera's speed is calculated by:

$$v = -K \frac{(L')^+ + (L^*)^+}{2} e(t) \tag{36}$$

If the required motion is only rotations around the camera's Z-axis, there is no retreat motion. However, a small retreat occurs if the required motion of the camera is a combination of translation and rotation around the optical axis. The retreat motion makes the trajectories of the feature points almost linear. Thus, it helps to ensure that the features do not leave the camera's field of view.

The MJPM scheme has an advantage over the PMJM scheme when the required camera motion is very close to 180° , the MJPM converges with a smooth trajectory, but the PMJM shows an unreasonable trajectory. The camera performs a large rotation around the optical axis at the beginning of the motion because the interaction matrix is close to the singularity (Cong and Hanh 2019).

2.7 Parametric approximations for the Jacobian matrix

E. Nematollahi and F. Janabi-Sharifi have proposed a new class of parametric approximations for interaction matrices (Nematollahi and Janabi-Sharifi 2009) to solve some of the difficulties of IBVS systems related to interaction matrix approximations. This paper proposes three methods in which the third method is an extension of the other two methods. The first and second methods are special cases of the third method. So, we only discuss the third method.

Consider the following system of equations:

$$\begin{aligned} (\delta_1 L' + \delta_2 L^*) v &= -K(\delta_1 + \delta_2) e(t) \\ (\eta_1 L' + \eta_2 L^*) v &= -K(\eta_1 + \eta_2) e(t) \end{aligned} \tag{37}$$

where $\delta_1, \delta_2, \eta_1, \eta_2$ are the coefficients that need to be determined.

The camera velocity is calculated by:

$$v = -K \begin{bmatrix} L_\delta \\ L_\eta \end{bmatrix}^+ \begin{bmatrix} (\delta_1 + \delta_2) e(t) \\ (\eta_1 + \eta_2) e(t) \end{bmatrix} = -KA^{-1} B^T e(t) \tag{38}$$

where $L_\delta = \delta_1 L' + \delta_2 L^*, L_\eta = \eta_1 L' + \eta_2 L^*$

$$A = L_\delta^T L_\delta + L_\eta^T L_\eta$$

$$B = (\delta_1 + \delta_2) L_\delta + (\eta_1 + \eta_2) L_\eta$$

We can see that this method includes all the above methods except the MJPM method. For example, when choosing $\delta_1 = \delta_2 = 1/2$ and $\eta_1 = \eta_2 = 0$, the PMJC method will be obtained. So, if the parameters are chosen appropriately, this method shows superior behavior over other methods. This

scheme can control the camera to perform a pure rotation of 180° around the optical axis without any retreat motion.

3 Assessment of visual servoing techniques

3.1 Description of evaluation criteria and methods

In order to make the most objective evaluation of visual servoing control techniques, it is important to adhere to the evaluation criteria rigorously. Towards this goal, the paper builds a series of control tasks that are commonly found in visual servoing. In addition, it also proposes quantitative criteria for evaluation. Techniques are evaluated when applied in tasks where visual servoing systems often encounter difficulties and reveal defects.

3.1.1 Control tasks

Various visual servoing control methods have been developed to meet the specific requirements of different tasks. Therefore, to evaluate visual servoing techniques, the paper has selected four control tasks that most often cause problems in visual servoing.

Task 1: Rotation around the optical axis. The first task considered is pure rotation around the optical axis. With this task, the IBVS algorithms expose problems such as unnecessary movements along the optical axis (“camera retreat” and “camera advance” problem), and the camera movement to infinity when required a rotation of 180° . Several diagrams have been developed to solve this problem such as PC&SH method, 2 ½D VS, SPVS or solved by using different approximations for Jacobian matrices.

Task 2: Translation along the optical axis. The second task is pure motion along the optical axis with an initial position from a distance of 1 m in front of the target to a distance of 1 m behind the target. This particular motion is chosen because, in essence, visual servoing control schemes depend on depth estimation.

Task 3: Rotation movement around the Y-axis of the camera. The third task corresponds to pure rotation around the Y-axis of the coordinate system attached to the camera. This task represents rotations around axes parallel to the image plane. The target rotation angles will be from 10° to 70° . When the rotation angle is greater than 70° , the feature points will almost align, so visual servoing cannot be performed.

Task 4: Motion along the Y-axis of the camera. This task represents the result of translation along any axis in a plane parallel to the image plane.

Task 5: General motion. The final task is generalized movements that require the visual servoing system to perform translational and rotational movements simultaneously.

In this task, in order to have a better assessment of the performance of VS algorithms, the simulation is performed with the PUMA560 robot. The velocity obtained during visual servoing is used to control the rotations through the robot’s Jacobian matrix. Figure 4 shows the PUMA560 robot from the Robotics Toolbox used for simulation.

3.1.2 Performance metrics

For the evaluations to be quantitative, it is necessary to define a set of performance metrics that are used to evaluate the schemes. The following metrics are selected for evaluation:

Number of iterations to convergence: Visual servoing is considered successful and stops when the average error of the feature points is less than 1 pixel. If the average error of five consecutive iterations differs by no more than 0.1 pixel, the error is considered to converge to a constant value and the process is stopped. The number of iterations can be increased or decreased if the coefficient λ is changed. However, it is a parameter that deeply evaluates the performance of many systems or of a system with many different tasks.

Error at termination: At the halt of visual servoing, the pixel-error of each feature point from the desired position is calculated. Visual servoing stops if the error decreases to zero or converges to a constant value as mentioned above. Additionally, if over 300 iterations had been performed without convergence visual servoing was halted. Finally, the VS process is halted if the camera has moved back 10 m from the target, advanced to a depth of 0 m, or the feature points have moved more than 3000 pixels from the image center.

Maximum feature excursion: At each iteration, the distance from the feature point to the center of the image is calculated. The maximum value, in pixels, achieved during the entire process was used for the evaluation.

Maximum camera translation: At each iteration, the distance of the camera from the desired position is calculated. The maximum value is used to evaluate.

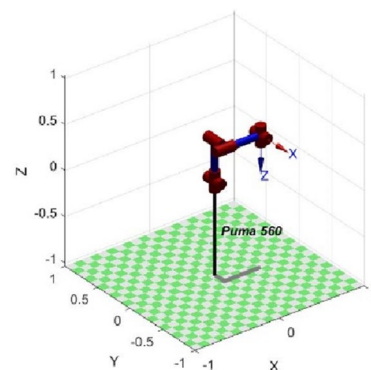


Fig. 4 PUMA560 robot

3.1.3 Simulation methodology

The whole simulation uses Matlab with the support of Machine Vision Toolbox and Robotics Toolbox (Zhong et al. 2019). For each simulation, the feature points are the vertices of the square in the 3D coordinate system. The square has a side length of 0.1 m and the desired position of the camera is at a position of 1 m relative to the square plane. To estimate the homography matrix, it is required that the feature points be coplanar. Using the vertices of the square gives a uniform result that a configuration of any shape may not be able to achieve.

Simulation using a camera with pixel sizes of $10^{-5} \times 10^{-5}$ m, focal length $f=0.008$ m. The image plane can be considered to be infinite, but if a feature goes out of the distance of 6000 pixels from the center of the image, the visual servoing system will stop. Furthermore, if the system fails to zero error, or does not converge to a fixed error after 300 iterations, visual servoing also halts.

The gain for each system was a 6×6 diagonal matrix, allowing to adjust the convergence rate independently. The gains are chosen so as to adjust to zero an error of a certain degree of freedom in 30 iterations while keeping the error of other degrees of freedom at zero.

The performance of the visual servoing system is affected by various conditions such as signal noise, error in calibration, and error in the kinematic parameters of the robot. The paper performs simulations for visual servoing with noise in images. The noise causes a large shift in the coordinates of the feature points. This causes errors when calculating the camera's displacement. To measure the influence of noise on different systems, the paper simulates noise during feature detection by adding an "offset" value to the pixel coordinates of features. This offset value follows a Gaussian distribution with a mean of zero and variance ranging from 0 to 1. Since the noise is random, the simulation is performed 100 times for the entire range of motion and averaged to smooth the results and remove outliers.

3.2 Simulation results and evaluation

In this section, the simulation results are presented for the five visual servoing control schemes described: Image-based Visual Servoing (IBVS), Position-based Visual Servoing (PBVS), 2 1/2 D Visual Servoing (2.5D VS), Partitioned Visual Servoing (PVS) and Shorted Path Visual Servoing (SPVS). Each system is simulated with four tasks. Each subsection of this section details a control task. At the beginning of the subsections, the notable results are summarized, followed by detailed descriptions for each performance metric.

Graphs are shown in groups of 5 figures, each for a different VS scheme. The figures arranged in clockwise order are IBVS, PBVS, 2.5D VS, PVS and SPVS. The simulation

results under the influence of noise are presented in 3D graphs with the variance of noise increasing along the right axis, and the motion variables (rotation, translation) increasing along the left axis. Performance index variables increase along the vertical axis.

3.2.1 Rotation around optical axis

The trajectories of the feature points are shown in Fig. 5. The IBVS method has a trajectory that is a straight line from the starting position to the ending position. The rest of the methods give orbits that are circular arcs. When the rotation is close to 180° , the features in the IBVS go to the center of the image, the Jacobian matrix becomes singular and the system cannot converge.

(a) Remaining pixel error

The simulation results are shown in Fig. 6. IBVS regulates the error to zero when rotations are less than 160° , even as the effect of noise increases. However, when the angle of rotation is greater than 160° , the camera moves back to infinity, bringing the feature points to the center of the image and cannot be converged.

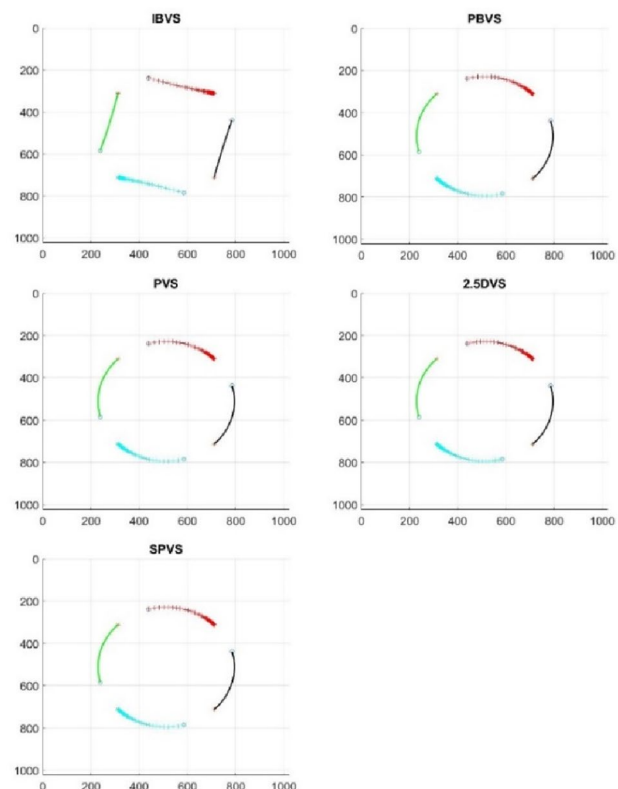


Fig. 5 Trajectories of feature points with a rotation of 60° around the Z axis

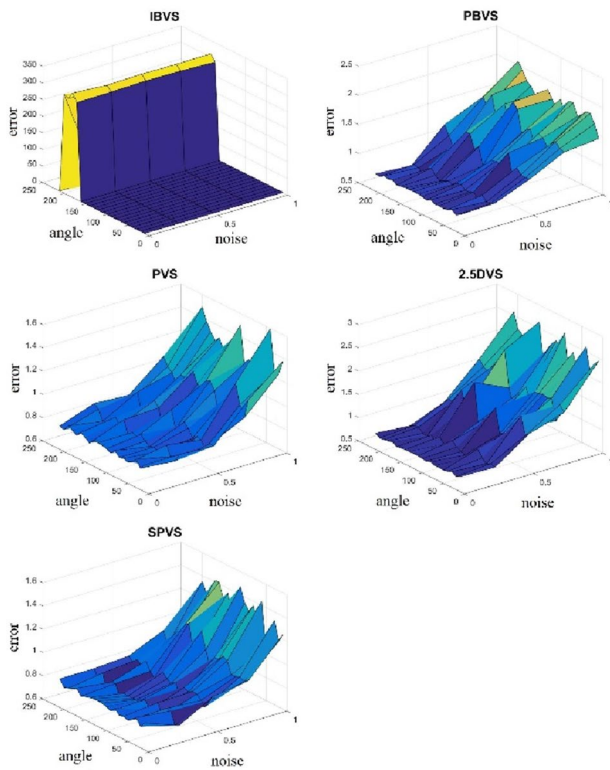


Fig. 6 Average pixel error when rotating around the Z axis

The remaining methods show the dependence of the error on the noise level. In which, the PVS method shows less influence of noise, while the influence of the other three methods is almost the same.

When there is no effect of noise or the effect is small, the algorithms are capable of adjusting the error to zero. However, when increasing the value of noise, after 300 iterations, the error still remains at 1.5 to 3 pixels (depending on the magnitude of the noise). The error of the IBVS algorithm also increases with the magnitude of the noise, although it is not shown in Fig. 6.

The rotation angle has almost no effect on the error.

(b) Number of iterations

Figure 7 shows the change of the number of iterations to convergence or failure with different values of rotation angle and noise.

The graphs show that all methods have a dependence on the number of iterations on the noise level and the rotation angle value. The IBVS method has a sudden change in the number of iterations when the rotation angle is close to 180° . This is because the camera performs a retreat to the limit value causing the system to stop, the retreating speed is very fast, so the number of iterations is also very small. Compared with other methods, IBVS is less affected by noise. When the

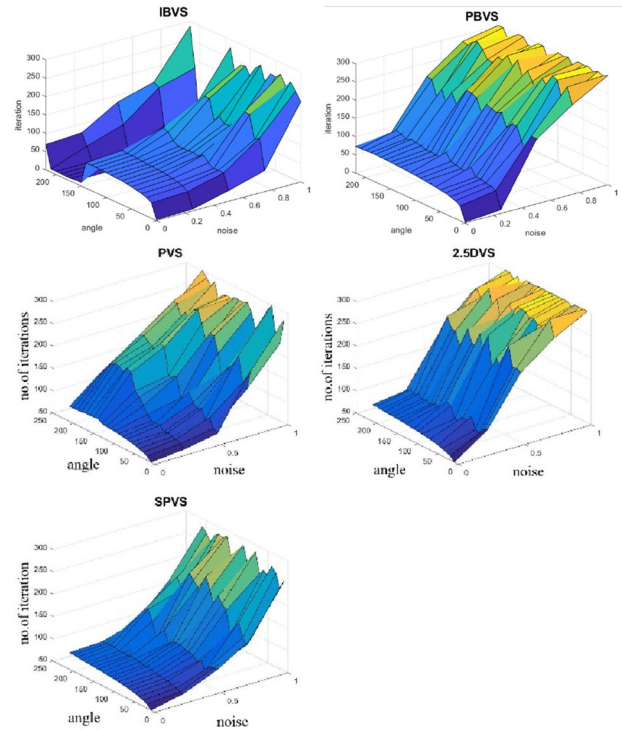


Fig. 7 Average iterations until convergence with rotation around the Z-axis

variance of noise is less than 0.5, the number of iterations is almost unchanged. When the variance is greater than 0.5, the number of iterations increases but is still less than 300 iterations. For rotation angles less than 30° , the number of iterations increases slightly as the rotation increases and is almost unchanged for rotation angles from 30° to 160° .

The two methods PBVS and 2.5DVS give almost the same results. The number of iterations starts to increase rapidly when the noise is larger than 0.25. When the noise is equal to one, these methods all take more than 300 iterations to stop. As the rotation angle value changes, there is also a slight increase in the number of iterations.

For the PVS method, the number of iterations increases when noise and rotation angle increase. Especially, if the rotation angle is large, the number of iterations increases sharply as the value of the noise increases.

(c) Maximum feature point excursion

The result of the dependence of the maximum distance of the feature points with the change of rotation angle and noise is shown in Fig. 8.

The results are similar to the graph for the number of iterations. Except for the PVS method, all other methods show that the distance increases with the noise value and is independent of the rotation angle. In the IBVS method when the

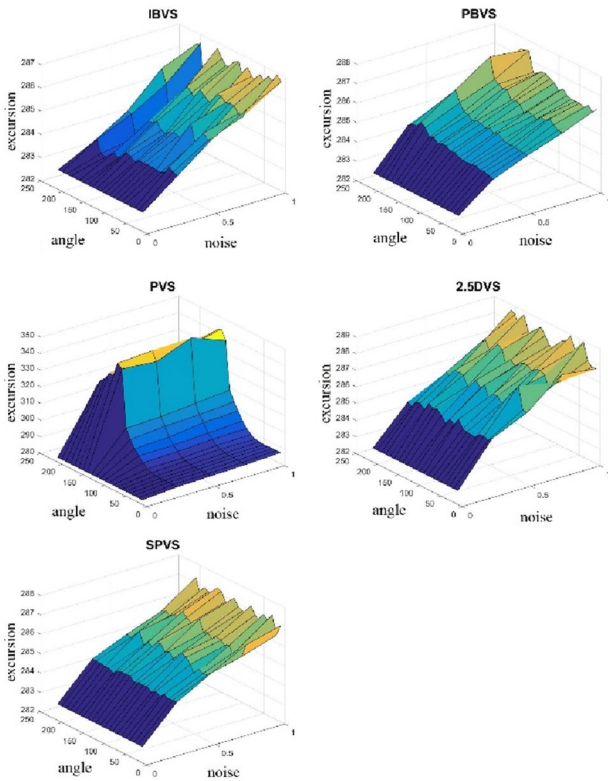


Fig. 8 Average maximum feature point excursion with rotation around the Z-axis

rotation angle is greater than 160° , the feature points move towards the center of the image, so the maximum excursion remains constant and equal to the value at the initial position. As for the PVS method, the trajectory of the feature points changes as the noise value increases, causing the displacement distance to increase. In the presence of noise, this distance depends only on the rotation angle and not on the noise. The excursion increases sharply as the angles increase and reach a maximum when the rotation is 180° .

(d) *Maximum Camera Translation*

In rotation around the optical axis, camera displacement is the most interesting feature. The graph for this parameter is shown in Fig. 9.

As mentioned before, IBVS has a large retreat movement. The translation of the camera increases exponentially with the value of the rotation angle. When the angle of rotation reaches 165° , this retreat exceeds 10 m and the process stops.

PBVS has the smallest camera translation. There is a small increase in this translation value as the magnitude of the noise increases.

The camera translates a large distance in the PVS method. This value increases with the rotation angle and reaches a maximum value of 1.23 m when the rotation angle is 140°

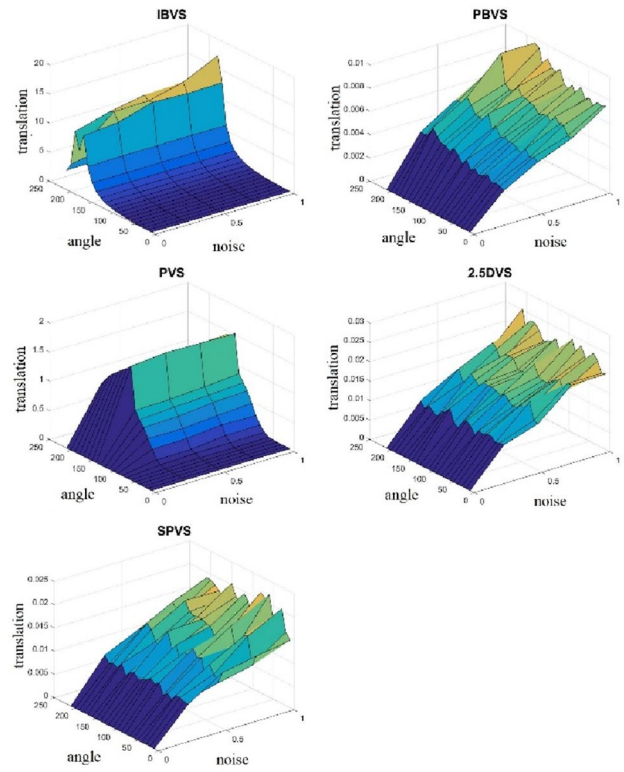


Fig. 9 Average maximum camera translation with rotation around the Z-axis

and then gradually decreases. The translation value does not depend on the magnitude of the noise.

The two methods SPVS and 2.5DVS have almost the same graph, with the camera displacement increasing with noise and almost unchanged with the rotation angle. The maximum retreat value is only about 25 mm.

3.2.2 Translation along the optical axis of the camera

The trajectories of the feature points in the image for motion along the Z-axis are almost the same (almost straight line). Figure 10 shows the trajectory for the IBVS method.

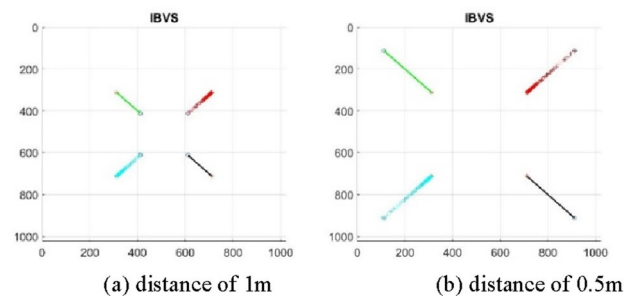


Fig. 10 Trajectories of feature points in translation along the Z-axis

There is no problem of stability with the translation along the optical axis task, and the systems have approximately the same graph of the criterion.

(a) Remaining pixel error

The pixel error graphs are shown in Fig. 11, giving the same results for all schemes. The error increases with the increasing noise and is almost independent to the initial distance. IBVS gives the smallest error value, with error values varying from 0.8 to 1.2 pixels. PBVS and 2.5DVS have the largest error with a range of 0.5–2.5 pixels. The error in the SPVS method is quite uniform. In some cases, when the noise variance is equal to 1, the error increases to about 1.8 pixels. The error range of the PVS method is from 0.5 to 1.5 pixels.

(b) Number of iterations

The graphs for the number of iterations are shown in Fig. 12. The graph shows that three methods IBVS, PVS, SPVS have nearly the same results, all have the average number of iterations less than 300, the number of iterations increases rapidly when the value of the noise is greater than 0.25. IBVS gives the smallest number of iterations and is less affected by noise than other methods. When the noise

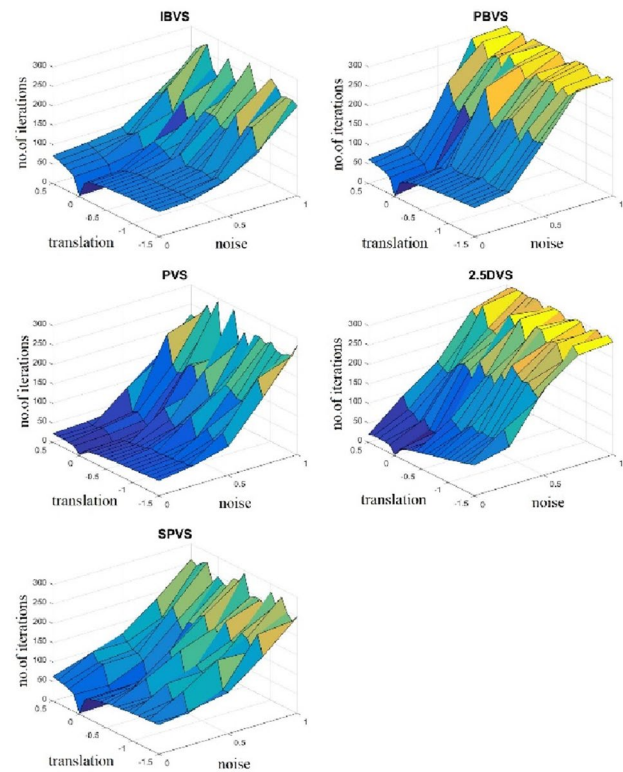


Fig. 12 Average iterations with translation along the Z-axis

is equal to 0.25 and 0.5, the number of iterations is almost unchanged. The two remaining methods have the same results with the number of iterations increasing sharply as the noise increases, and when the noise is greater than 0.75, the number of iterations reaches 300.

When the motion value increases, the number of iterations increases if there is no noise, when there is noise, the motion value does not affect the number of iterations (except for the 2.5DVS method which still changes).

3.2.3 Rotation around the Y-axis of the camera

Based on the trajectories of the feature points when rotation around the Y axis in Fig. 13, it can be seen that the trajectories of the features of all methods are almost straight lines, in which PVS and 2.5DVS give trajectories of relatively small curvature.

(a) Remaining pixel error

The pixel error graph is shown in Fig. 14, the results show that there is a difference between the methods.

All methods have an error value that increases as the noise increases. If the noise is greater than 1, all methods give an error of more than 1 pixel. Two methods IBVS and

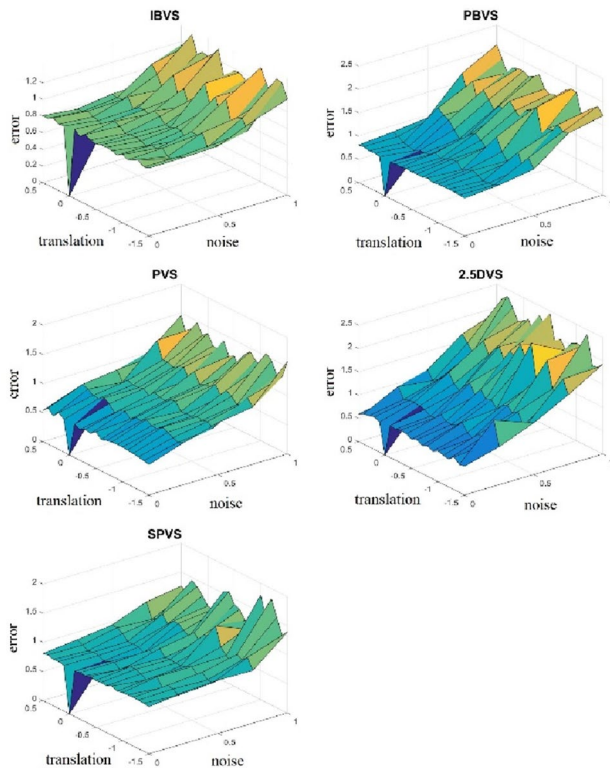


Fig. 11 Average pixel error when translation along the Z axis

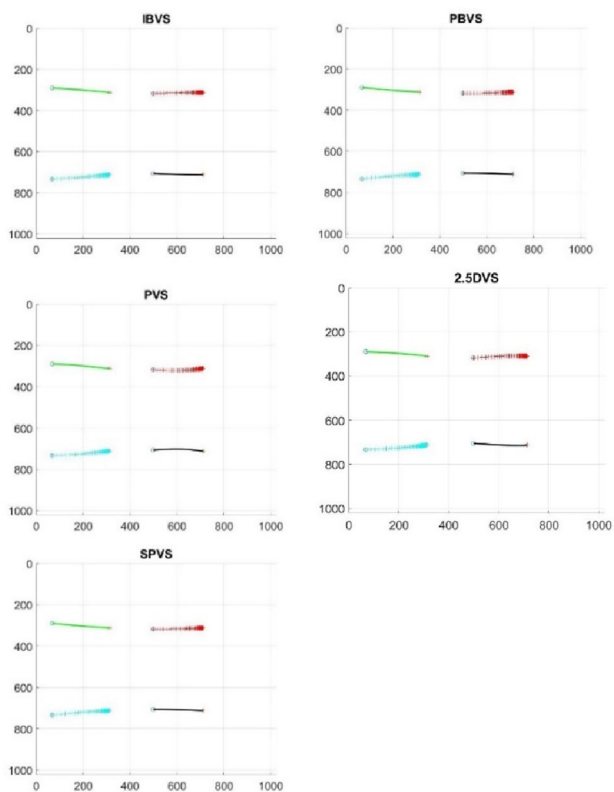


Fig. 13 Trajectories of feature points when rotated around the Y-axis

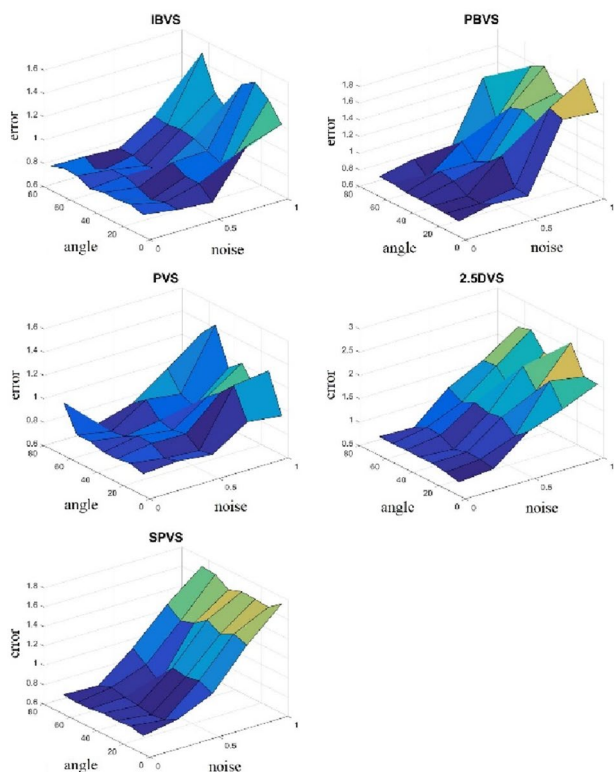


Fig. 14 Average pixel error when rotating around the Y axis

PVS give lower error results than other methods. When the noise magnitude is less than 1, these two methods give an error of less than 1 pixel. The maximum error when noise is equal to 1 is about 1.4 pixels.

2.5DVS and SPVS give the same error graph results, the error increases as the noise size increases. However, SPVS has a smaller error with a maximum error of about 1.7 pixels, and 2.5DVS is 2.5 pixels. PBVS has a complex noise change graph, but in general, the error increases with noise and reaches a maximum of 1.8 pixels.

The angle of rotation has almost no effect on the error.

(b) Number of iterations

Figure 15 shows the number of iterations to stop when rotating around the Y axis. Based on that, it can be seen that all methods have dependent on the number of iterations on the noise level and the rotation angle. The number of iterations increases as the rotation angle value increases. The 2.5DVS method has a rapid increase in the number of iterations to more than 100 when increasing the angle value close to 80°.

The graphs of the three methods 2.5DVS, PBVS and SPVS show that the number of iterations increases rapidly when the magnitude of the noise increases, in which 2.5DVS has the most influence, all three methods have 300 iterations

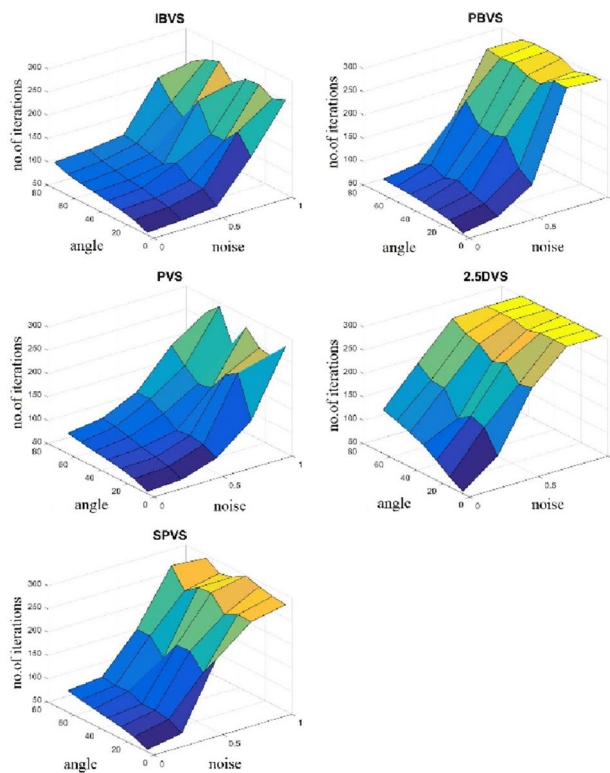


Fig. 15 Average iterations when rotating around the Y axis

with noise variance greater than 0.75. The two methods IBVS and PVS are less affected by noise, the number of iterations is still guaranteed to be less than 300 iterations when the noise variance is equal to 1.

(c) Maximum camera translation

The graphs of the camera translation with different rotation angles and noise are shown in Fig. 16. It can be seen that the two methods PBVS and SPVS have almost no camera displacement because these two methods directly use the control law according to the translation vector for the translational velocities in three axes. When there is the influence of noise, the displacement of the camera of these two methods increases with the noise, the maximum displacement value is 6 mm when the noise is equal to 1.

The other three methods show that the camera has unnecessary movements when the movement requires only rotation around the Y-axis of the camera. This can be explained by the fact that these methods use image features to control the translational velocity of the camera. The displacement value of the camera increases with the rotation angle value and is almost unaffected by noise except for IBVS whose displacement value increases slightly as noise increases. The

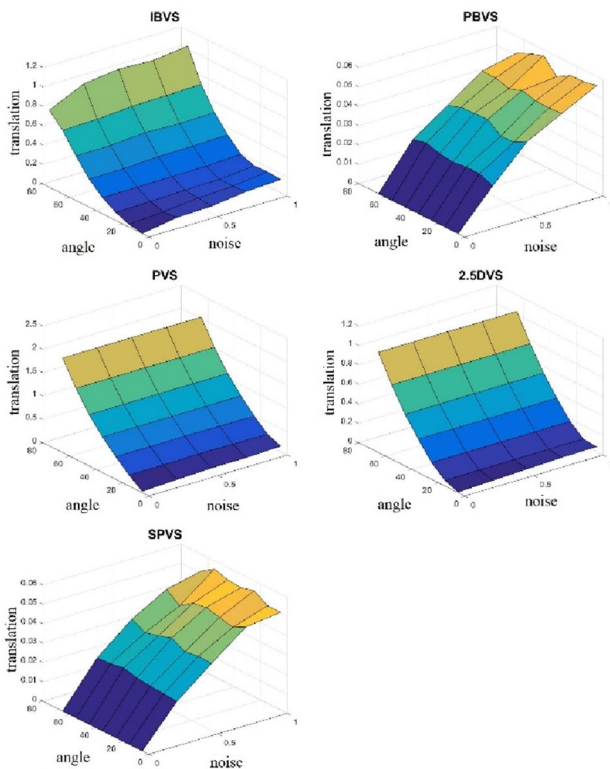


Fig. 16 Average maximum camera translation with rotation around the Y axis

PVS method has the largest displacement, with a maximum displacement of 2.2 m when the rotation angle is 80° . The two methods IBVS and 2.5DVS have the same displacement with a maximum value of about 1 m when the rotation angle is equal to 80° .

3.2.4 Translation along the Y-axis of the camera

The trajectories in the image of the feature points for motion along the Y axis are approximately the same. Figure 17 shows the trajectory for the PBVS method.

For this task, all methods yielded roughly the same performance metrics. Distance of motion does not have much effect on performance. IBVS and PVS are less affected by noise than the other methods.

(a) Remaining pixel error

Figure 18 shows the average pixel error when translating along the Y axis. The results are almost the same for all schemes. The error increases as the noise increases and is independent of the initial distance. PBVS and 2.5DVS have the largest error with a range of 0.75–2.5 pixels. The error in the PVS method is quite uniform, when the noise increases to 1, the error only increases to about 1.5 pixels. The other two methods have an error range of 0.75–2 pixels.

(b) Number of iterations

The graphs of the number of iterations are shown in Fig. 19. The graphs show that the number of iterations of all methods depends on the noise level and the motion value.

The three methods PBVS, 2.5DVS and SPVS have up to 300 iterations when the noise is greater than 0.75. The other two methods, IBVS and PVS, give a number of iterations

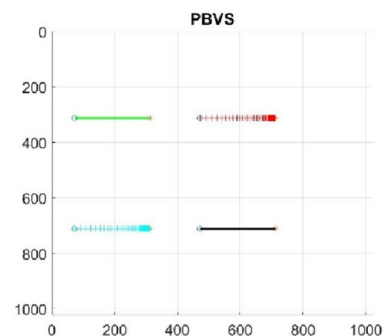


Fig. 17 Trajectories of feature points when translating along the Y-axis

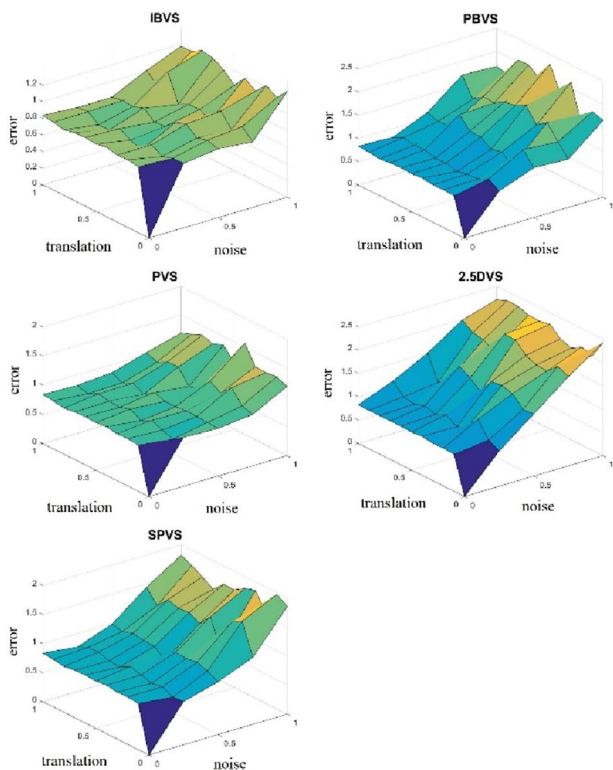


Fig. 18 Average pixel error when translating along the Y axis

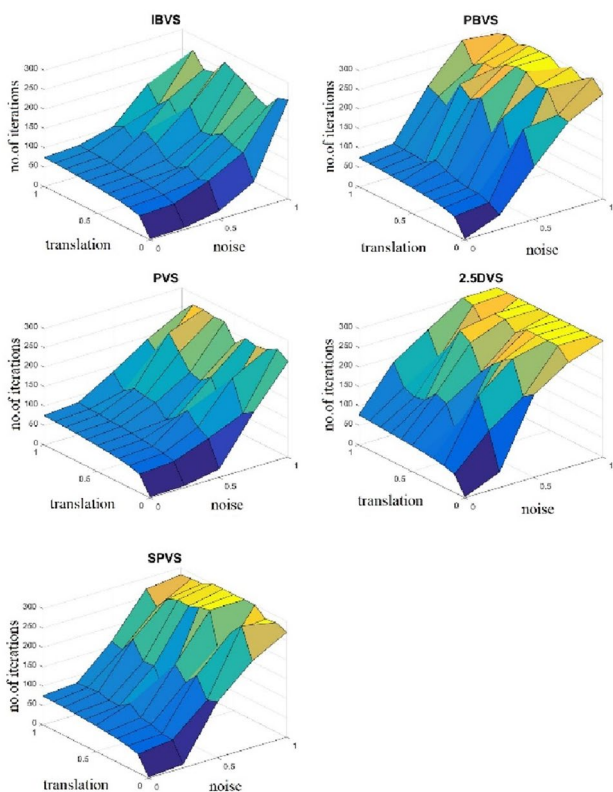


Fig. 19 Average iterations when translating along the Y axis

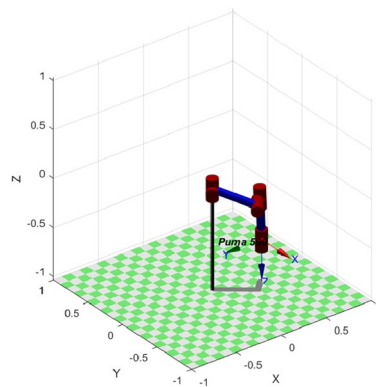


Fig. 20 The desired configuration of the robot

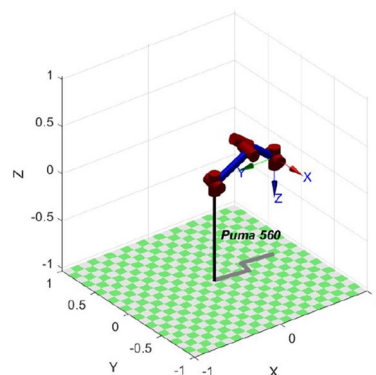


Fig. 21 The initial configuration of the robot in Test 1

less than 300 when the noise is equal to 1 and the number of iterations is almost unchanged if the noise is less than 0.5.

3.2.5 General motion

Tasks with the initial position of the camera in the general position are performed by using the PUMA 560 robot. The desired position of the robot is at a distance of 1 m from the feature points with the end-effector at the pose with coordinates $P_0 = [0.4318, -0.15, -0.6]$ and the three angles roll, pitch, yaw are $\pi, \pi, \pi/2$ as shown in Fig. 20.

The tasks performed in general motion were all performed under the condition that the noise in the image was white noise with the noise variance equal to 1. The tasks were performed 50 times and averaged for the criteria.

(a) Test 1

Test 1 is a combination of three translational motions along the X, Y and Z axes with the translation vector $P = [0.3, 0.2, 0.6]$. The robot configuration at this location is shown in Fig. 21.

Table 1 Performance metric when executing test 1

Scheme	Error (pixel)	Number of iteration	Excursion (pixel)
IBVS	0.87	82.26	274.44
PBVS	0.81	74.12	270.04
PVS	0.85	82.88	275.16
2.5DVS	0.87	93.58	275.26
SPVS	0.85	71.60	275.26

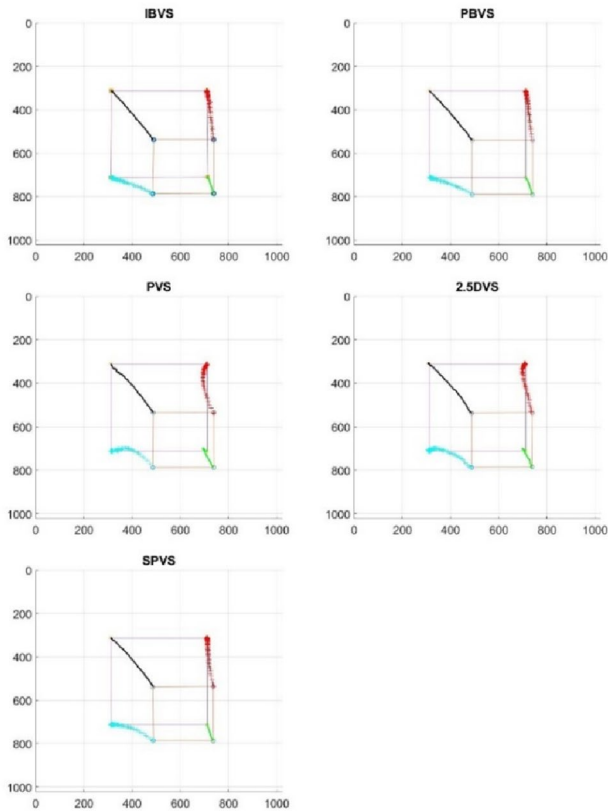
**Fig. 22** The trajectories of the feature points in Test 1

Table 1 shows the criteria for each method. Figures 22 and 23 show the trajectory of the feature points in the image and the trajectory of the camera for each method, respectively. It can be seen that, for test 1, which only requires translational motions along the three axes X, Y, and Z, the result is almost the same. All methods can regulate the error to zero. The number of iterations for convergence of each method is almost the same. The 2.5DVS method has a slower convergence rate than other methods. This can be explained when looking at the velocity graph in Fig. 24, the Z-axis velocity of the 2.5DVS method is only half of the other methods, so the convergence is slower. The PVS method also has a small Z velocity, but a larger X and Y velocity, so the convergence speed is also equal to other methods.

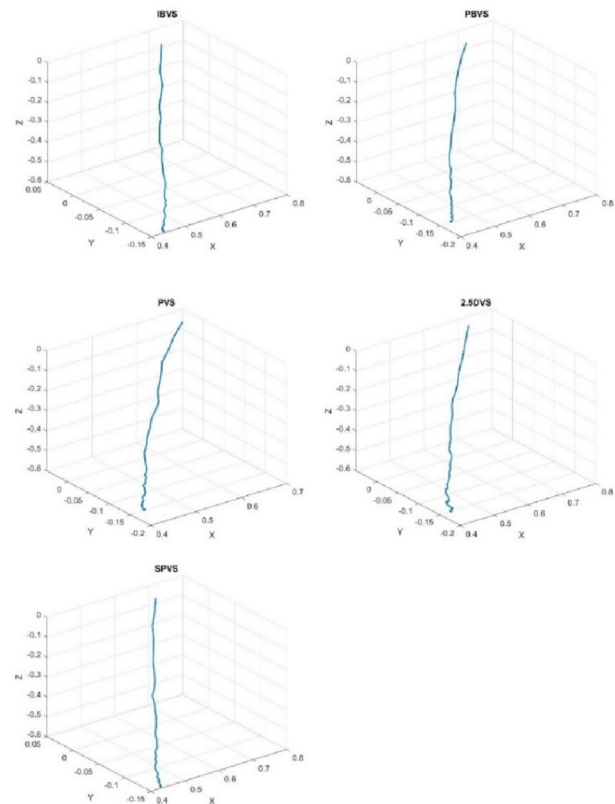
All systems have a feature point trajectory that is almost a straight line from the initial coordinates to the desired coordinates. In which the trajectories of PVS and 2.5DVS methods have small curvature. The camera's trajectories in 3D space are also straight lines.

(b) Test 2

Test 2 is a combination of three rotations around the X, Y, and Z axes with rotation angles of 10° , 15° and -90° , respectively. There is no translational motion required in test 2. The objective of this test is to investigate the independence of rotational and translational motions. Table 2 presents the performance metrics for test 2.

Based on the data in Table 2, it can be seen that all methods can be able to drive the camera to the desired position. In particular, the 2.5DVS method gives faster convergence results than other methods. The PVS method has the displacement of the feature points larger than half of the image size, so the feature points will be out of the image boundary. All other methods keep the feature points in the image as shown in Figs. 25 and 26

Figures 27 and 28 show the angular and translational velocities of the camera, respectively. The translational

**Fig. 23** The 3D trajectory of robot in Test 1

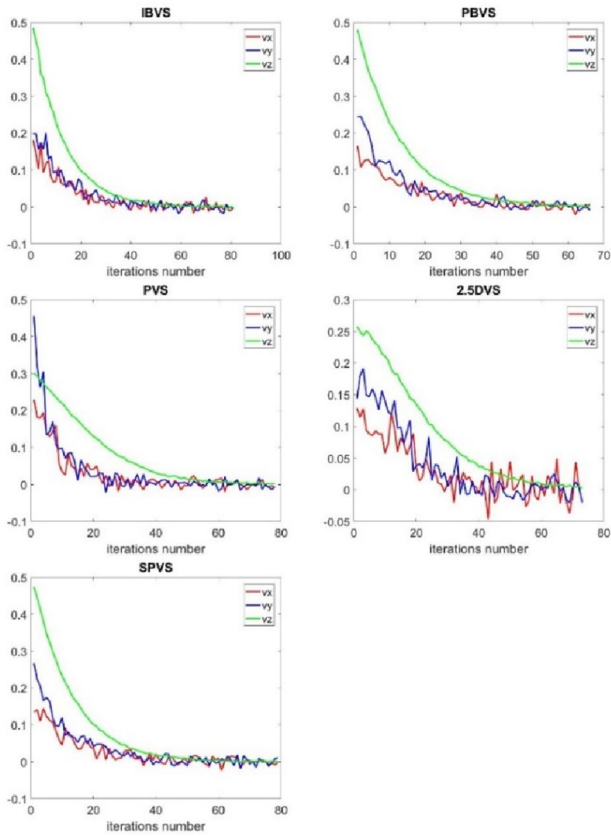


Fig. 24 The translational velocity of robot in Test 1

Table 2 Performance metric when executing test 2

Scheme	Error (pixel)	Number of iteration	Excursion (pixel)
IBVS	0.85	80.84	497.24
PBVS	0.87	81.56	512.08
PVS	0.86	85.92	525.31
2.5DVS	0.84	57.48	497.49
SPVS	0.81	80.56	497.59

velocities of the two methods PBVS and SPVS are almost zero, so there is no translational motion. The IBVS method has the highest velocity in the Z direction, the PVS method has the largest velocity in the Y direction. The angular velocity graphs of the three methods PBVS, 2.5DVS and SPVS are almost the same. Although the required motions are rotation in three axes, these three methods only require two rotations around the axis. The two methods IBVS and PVS require all three rotations around three axes to achieve the desired position.

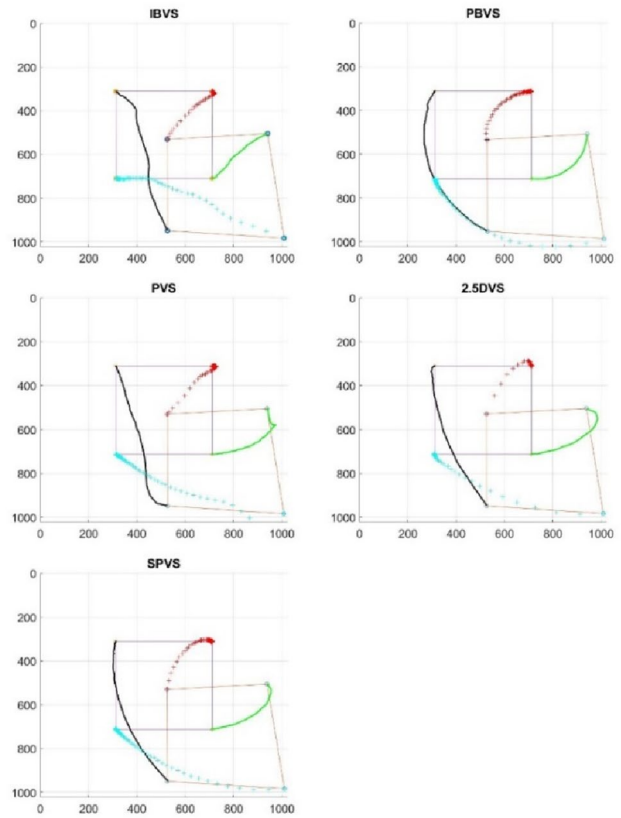


Fig. 25 The trajectories of the feature points in Test 2

Figure 26 shows the change in the 3D coordinates of the robot, it can be seen that the two methods PBVS and SPBVS have no redundant motion along the X, Y and Z axes. The other methods have unnecessary motions along the X, Y, and Z axes. In which the IBVS method has the largest motion of 0.3 m in the Z direction. The 2.5DVS method has the smallest motion in all directions.

(c) Test 3

Test 3 requires translation in the three X, Y and Z axes and rotation around the Z axis. The initial position of the camera relative to the desired position has the coordinate $P = [0.2, 0, 0.5]$ and the required rotation angle is 180° around the Z axis. Figure 29 shows the configuration of the robot at the initial position for test 3.

Based on the data obtained in Table 3, it can be seen that except for the IBVS method, all other methods have converged results after about 90 iterations. Because test 3 requires 180° rotation, IBVS performs a retreat to infinity,

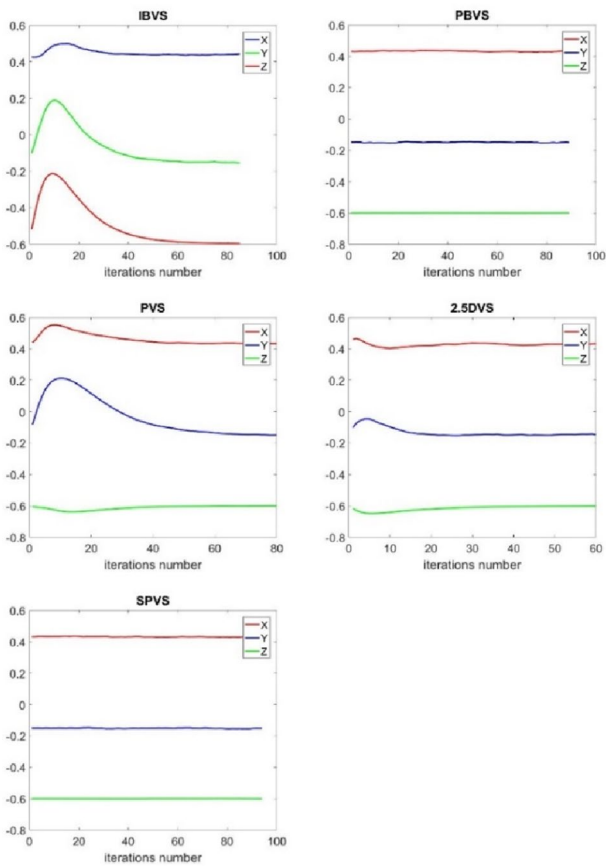


Fig. 26 The change in the 3D coordinates of the robot in Test 2

however, due to the robot's limitation, after only 5 iterations, the system stops due to exceeding the robot's workspace.

To better see the behavior of each system, consider the graphs of the feature point trajectories in Fig. 30 and the 3D trajectories of the camera in Fig. 31. Each method gives different trajectories. The two methods PBVS and SPVS have similar trajectories of feature points in the image because they share the same control law for translational and rotational motion around the Z axis. The 2.5DVS method gives slightly different trajectories because a different control law is used for translational motion. The PVS method gives a different trajectory than the other methods in which the feature points follow curves, so the distance of feature points from the center of the image is also larger than other methods (Table 3). The 3D trajectory of the PVS method is also a curve with great curvature, while the PBVS methods are almost linear. Test 3 does not have motions in the X and Y axes from

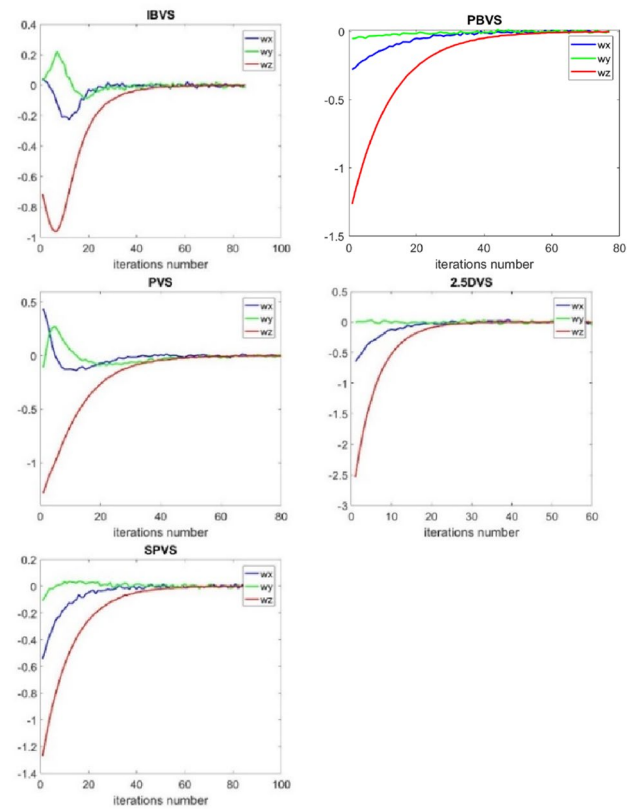


Fig. 27 The angular velocity of the robot in Test 2

which it can be seen that only the SPVS method doesn't have unnecessary movements, the trajectory is a straight line from the initial to the final position. The two methods PBVS and 2.5DVS have redundant movements on the Y axis. IBVS method has a retraction movement in the Z direction, the camera has retreated up to a height of 0.8 m, exceeding the limit of the robot.

Figures 32 and 33 are velocity plots. It can be seen that the two methods PBVS and SPVS have similar velocity graphs (the same control law). The PVS method gives a velocity graph with very large variations in the velocity values in the X and Y directions. The IBVS has a large velocity in the Z-axis to perform the retreat motion.

(d) Test 4

Test 4 is a general motion that requires translation and rotation in the X, Y, and Z axes. The initial position of the

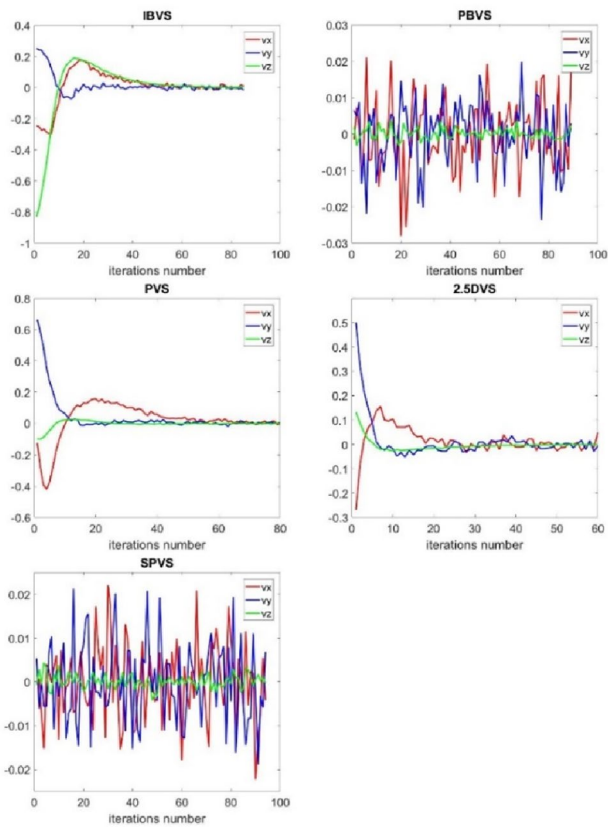


Fig. 28 The translational velocity of the robot in Test 2

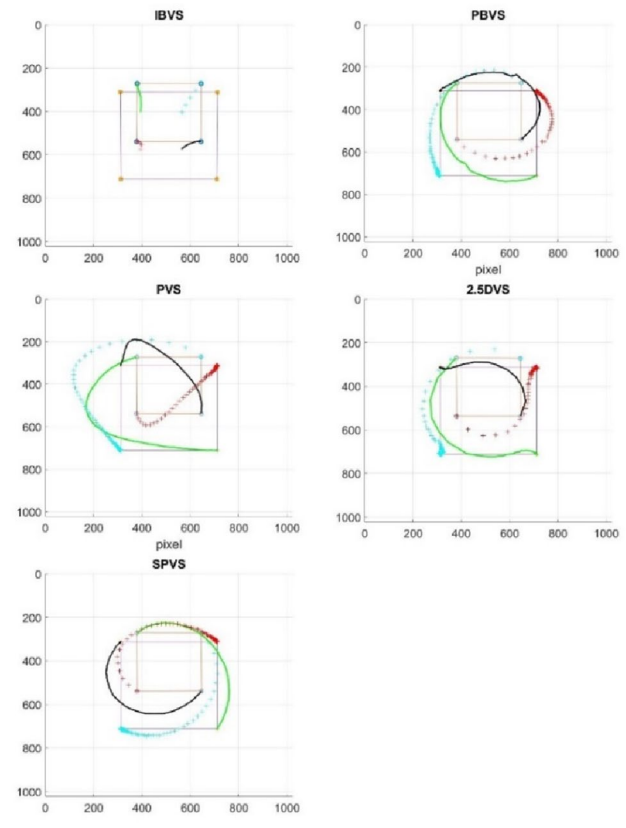


Fig. 30 The trajectories of the feature points in Test 3

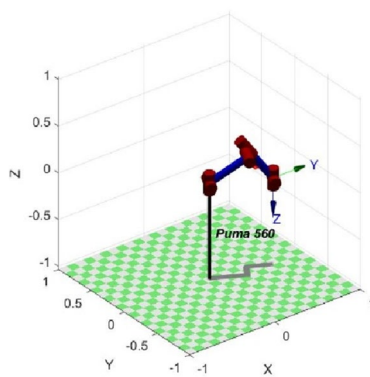


Fig. 29 The initial configuration of the robot in Test 3

Table 3 Performance metric when executing test 3

Scheme	Error (pixel)	Number of iteration	Excursion (pixel)
IBVS	404.79	5	240.28
PBVS	0.86	91.10	301.48
PVS	0.86	93.20	398.69
2.5DVS	0.85	90.92	281.10
SPVS	0.82	88.62	287.08

camera relative to the desired position is $P = [-0.3, -0.2, 0.4]$, the rotation angles are $\pi/6, -\pi/6, -\pi/2$. The configuration of the robot in the initial position is shown in Fig. 34.

In this test, the features are located near the edge of the image, so that the phenomenon of the features coming out of the camera’s field of view can be seen. Table 4 shows the performance metrics for test 4. All methods can drive the camera to the desired position with an average error of about 0.8 pixels and a number of iterations between 80 and 90. The SPVS method has the largest number of iterations with 94.1 iterations. The two methods PBVS and PVS have the distance from the feature points to the center of the image larger than half the size of the image, so these points can be out of the camera’s field of view. The remaining methods still ensure that the features are in the field of view of the camera. The trajectories of the feature points are shown in Fig. 35.

In Fig. 35, it can be seen that the feature points trajectories of the two methods PBVS and PVS are curves with a large radius, so the two lower points are out of the camera’s field of view. The remaining three methods use pixel coordinates in control schemes, so they always tend to bring the feature points towards the desired position, thus keeping the feature points in the image. The SPVS method has the trajectories of the feature points closest to straight lines.

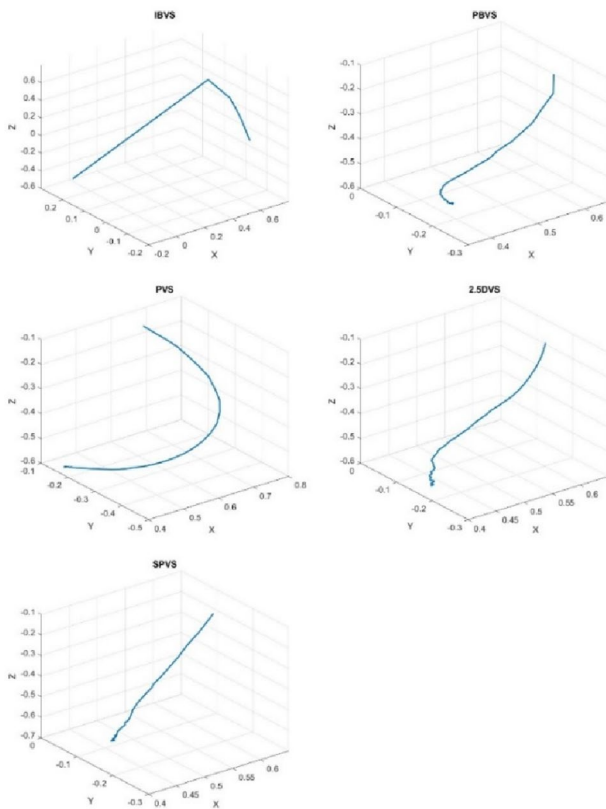


Fig. 31 The 3D trajectory of the robot in Test 3

Figure 36 shows the 3D trajectories of the camera. It can be seen that the trajectories in the two methods IBVS and PVS are arbitrary curves. In the PBVS method, the trajectory is a curve with small curvature. The SPVS method has a straight-line trajectory. Thus, the SPVS method has the shortest trajectory both in 3D space and in the image. The 2.5D VS method also has an almost straight-line trajectory.

The camera velocities are shown in Fig. 37 and Fig. 38. The two methods IBVS and PVS give a large change in velocity values, while in the other methods, the velocities decrease exponentially to zero.

4 Conclusion

The paper has presented the general theory of visual servoing, the two main components in visual servoing are control schemes and visual features. Several enhancement and combination schemes are used to increase the robustness

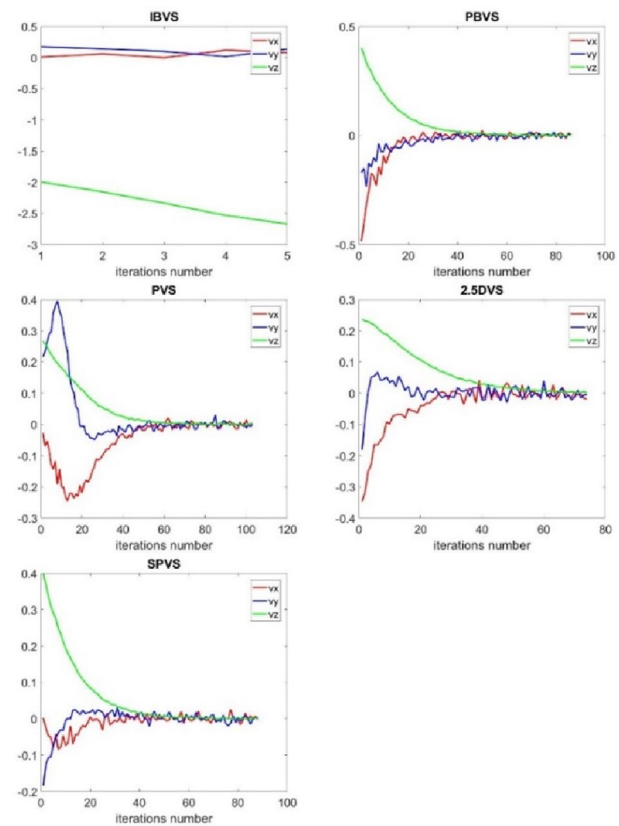


Fig. 32 The translational velocity of the robot in Test 3

of the system. Problems in visual servoing are presented and interactive matrix approximations are used to solve them in IBVS.

Several standardized tests have been performed in the face of image errors to evaluate system performance against difficult tasks. This data can be used to select appropriate visual servo systems for specific tasks.

The two methods IBVS and PVS are less affected by noise than other methods due to directly using the control laws in the image. However, they create unnecessary movements of the robot. The two methods PBVS and 2.5D are influenced by noise, which may not achieve the desired error if the value of the noise is large.

Through the general tasks, it can be concluded that the two methods PVS and PBVS have complex trajectories in the image and feature points easily out of the camera's field of view, especially when these points are located near the boundary. The two methods IBVS and PVS have redundant movements as mentioned above. The SPVS method

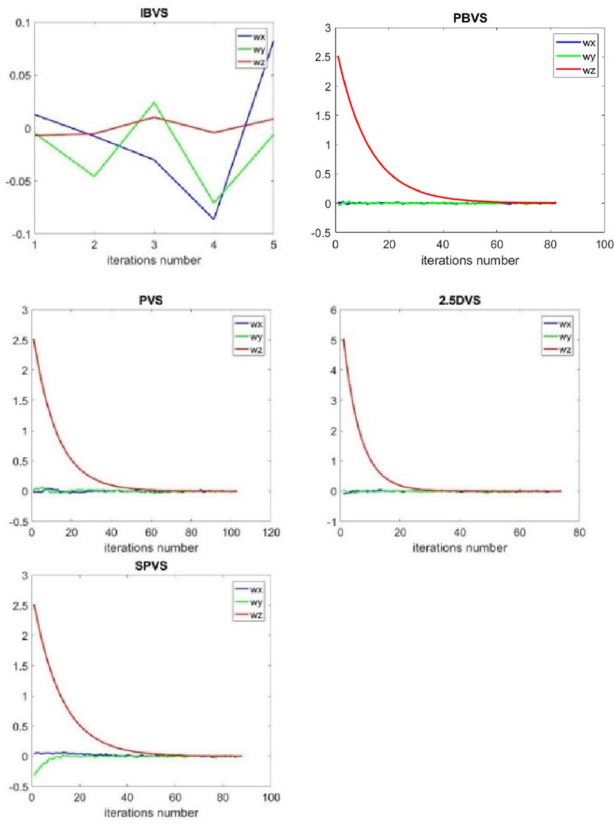


Fig. 33 The angular velocity of the robot in Test 3

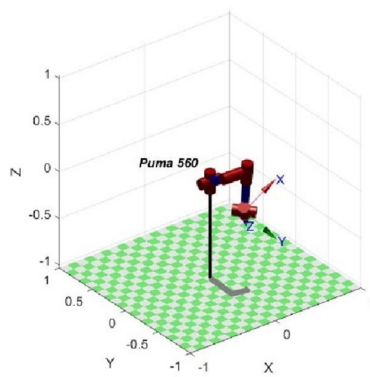


Fig. 34 The initial configuration of the robot in Test 4

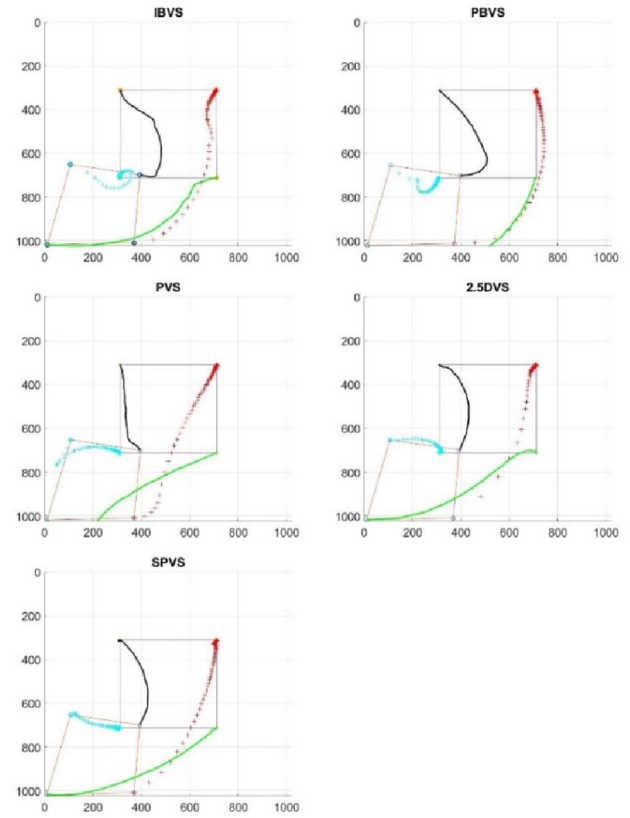


Fig. 35 The trajectories of the feature points in Test 4

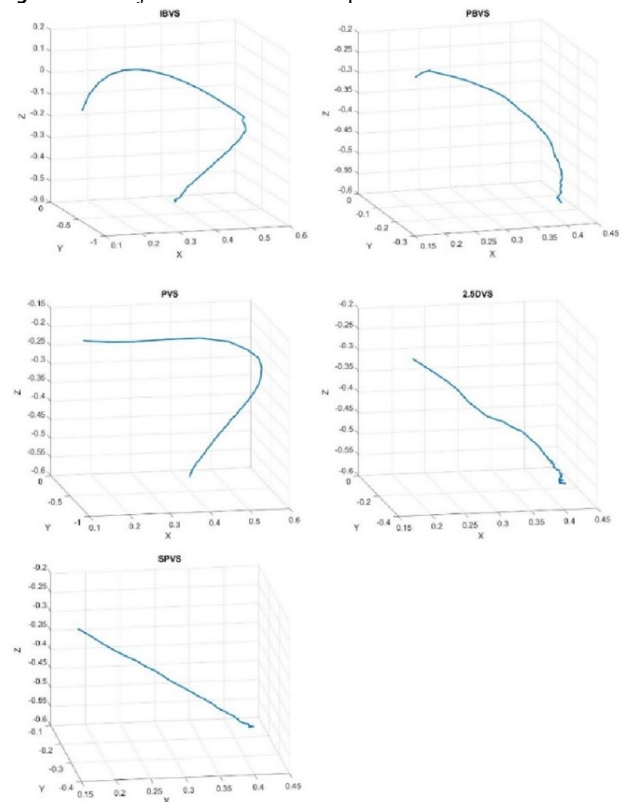


Fig. 36 The 3D trajectory of robot in Test 4

Table 4 Performance metric when executing test 4

Scheme	Error (pixel)	Number of iteration	Excursion (pixel)
IBVS	0.84	91.90	519.69
PBVS	0.85	86.68	576.02
PVS	0.87	89.34	617.05
2.5DVS	0.89	83.54	505.73
SPVS	0.84	94.10	508.15

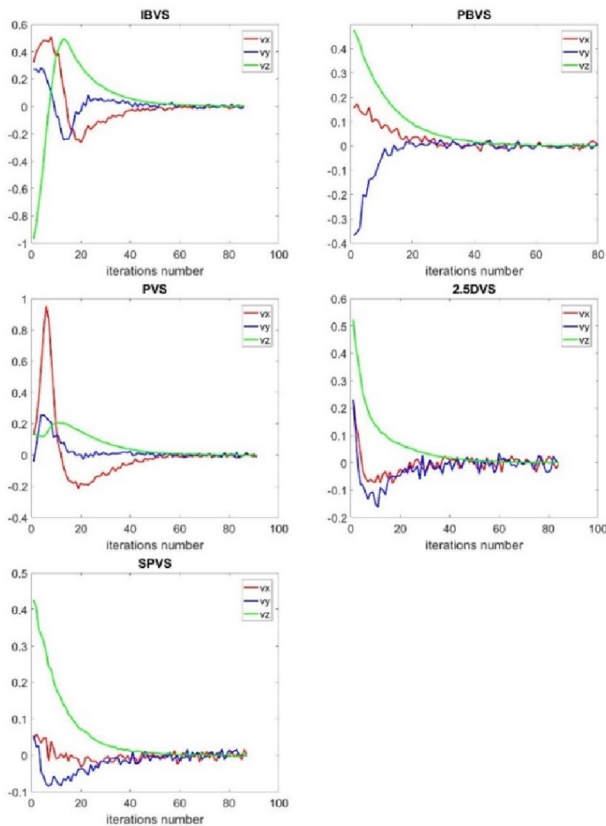


Fig. 37 The translational velocity of robot in Test 4

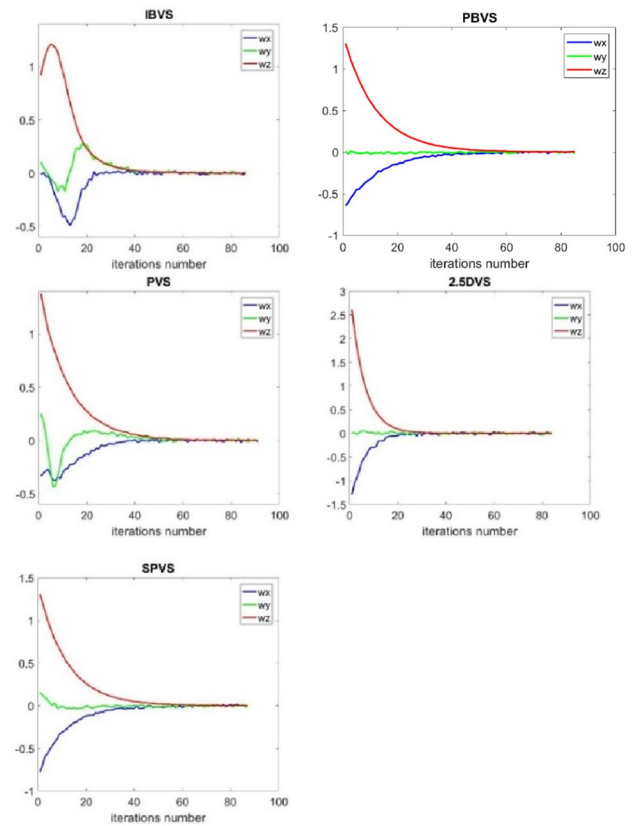


Fig. 38 The angular velocity of robot in Test 4

has the best results with a 3D trajectory that is a straight line from the starting position to the desired position without unwanted motion and keeping the features in the camera's field of view. The 2.5DVS method has performance somewhere in the middle.

Acknowledgements We acknowledge Ho Chi Minh City University of Technology (HCMUT), VNU-HCM for supporting this study.

Data availability The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

References

- Abhilash, T.V., Ashok, S.: Visual Servoing of a Switched System with Supervisory Control. In: *International Journal of Scientific & Engineering Research*, vol. 7, No. 1. (2016)
- Ahmadi, B., Zakeri, E., Xie, W.F.: Optimal image-based task-sequence/path planning and robust hybrid vision/force control of industrial robots. *IEEE Access* **10**, 26347–26368 (2022)
- Alatartsev, S., Stellmacher, S., Ortmeier, F.: Robotic task sequencing problem: a survey. *J Intell Robot Syst* **80**, 279–298 (2015)
- Andreff, N., Espiau, B., Horaud, R.: Visual servoing from lines. In: *IEEE International Conference on Robotics and Automation*, pp. 2070–2075. (2000)
- Cervera, E., Pobil, A., Berry, F., Martinet, P.: Improving image-based visual servoing with three-dimensional features. *Int. J. Robot. Res.* **22**(10–11), 821–840 (2003)
- Chaumette, F.: A first step toward visual servoing using image moments. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1, pp. 378–383. (2002)
- Chaumette, F.: Image moments: a general and useful set of features for visual servoing. *IEEE Trans. Rob.* **20**(4), 713–723 (2004)
- Chaumette, F., Hutchinson, S., Corke, P.: Visual servoing. In: Siciliano, B., Khatib, O. (eds.) *Handbook of robotics*, 2nd edn., pp. 841–866. Springer (2016)
- Chwa, D.: Integral-sliding-mode-observer-based structure and motion estimation of a single object in general motion using a monocular dynamic camera. *IEEE Access* **8**, 14207–14222 (2020)
- Collewet, C., Marchand, E.: Colorimetry-based visual servoing. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5438–5443. (2009a)
- Collewet, C., Marchand, E.: Photometry-based visual servoing using light reflexion models. In: *IEEE International Conference on Robotics and Automation*, pp. 701–706. (2009b)
- Collewet, C., Marchand, E., Chaumette, F.: Visual servoing set free from image processing. In: *IEEE International Conference on Robotics and Automation*, pp. 81–86. (2008)
- Cong, V.D., Hanh, L.D.: Evaluate Control Laws Related to Interaction Matrix for Image-Based Visual Servoing. In: *2019 6th NAFOSTED Conference on Information and Computer Science (NICS)*, pp. 454–459. Hanoi, Vietnam (2019)
- Cong, V.D., Hanh, L.D.: Combination of two visual servoing techniques in contour following task. In: *2021 International*

- Conference on System Science and Engineering (ICSSE), pp. 382–386. (2021)
- Cong, V.D., Hanh, L.D.: A new decoupled control law for image-based visual servoing control of robot manipulators. *Int. J. Intell. Robot. Appl.* (2022). <https://doi.org/10.1007/s41315-022-00223-5>
- Corke, P.I., Hutchinson, S.A.: A new partitioned approach to image-based visual servo control. *IEEE Trans. Robot. Autom.* **17**(4), 507–515 (2001)
- Corke, P.I., Hutchinson, S.A.: A new hybrid image-based visual servo control scheme. In: *Proceedings of the 39th IEEE Conference on Decision and Control*, vol. 3, pp. 2521–2526. (2000)
- Corke, P.I., Spindler, F., Chaumette, F.: Combining Cartesian and polar coordinates in IBVS. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5962–5967. (2009)
- Deguchi, K.: A direct interpretation of dynamic images with camera and object motions for vision guided robot control. *Int. J. Comput. Vision* **37**(1), 7–20 (2000)
- Dejun, G., Kam, K.L.: Image-based estimation, planning, and control for high-speed flying through multiple openings. *Int. J. Robot. Res.* **39**(9), 1122–1137 (2020)
- Dementhon, D., Davis, L.S.: Model-based object pose in 25 lines of code. *Int. J. Comput. Vision* **15**(2), 123–141 (1995)
- Deng, L., Janabi-Sharifi, F., Wilson, W.J.: Hybrid strategies for image constraints avoidance in visual servoing. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1, pp. 348–353. (2002)
- Deng, L., Wilson, W.J., Janabi-Sharifi, F.: Dynamic performance of the position-based visual servoing method in the Cartesian and image spaces. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1, pp. 510–515. (2003)
- Diyaley, S., Aditya, A., Chakraborty, S.: Optimization of the multi-hole drilling path sequence for concentric circular patterns. *Opsearch* **57**(3), 746–764 (2020)
- Dong, J., Zhang, J.: A new image-based visual servoing method with velocity direction control. *J. Frankl. Inst.* **357**, 3993–4007 (2020)
- Espiau, B.: Effect of camera calibration errors on visual servoing in robotics. In: *Experimental Robotics III: The 3rd Int. Symp. on Experimental Robotics*, pp. 187–193. (1993)
- Feddema, J.T., Lee, C.S.G., Mitchell, O.R.: Weighted selection of image features for resolved rate visual feedback control. *IEEE Trans. Robot. Autom.* **7**(1), 31–47 (1991)
- Gans, N.R., Hutchinson, S.A.: Stable visual servoing through hybrid switched-system control. *IEEE Trans. Rob.* **23**(3), 530–540 (2007)
- Gans, N., Hutchinson, S., Corke, P.: Performance tests for visual servo control systems, with application to partitioned approaches to visual servo control. *Int. J. Robot. Res.* **22**(10/11), 955–981 (2003)
- Gans, N., Hutchinson, S.: A switching approach to visual servo control. In: *2002 IEEE International Symposium on Intelligent Control*, pp. 770–776. (2002)
- Ghasemi, A., Li, P., Xie, W.F., Tian, W.: Enhanced switch image-based visual servoing dealing with features loss. *Electronics* **8**(8), 1–20 (2019)
- Ghasemi, A., Li, P., Xie, W.: Adaptive switch image-based visual servoing for industrial robots. *Int. J. Control Autom. Syst.* **18**, 1324–1334 (2020)
- Hanh, L.D., Cong, V.D.: Implement contour following task of objects with unknown geometric models by using combination of two visual servoing techniques. *Int. J. Comput. vis. Robot.* **12**(5), 464 (2022)
- Hashimoto, K.: *Visual Servoing - Real-Time Control of Robot Manipulators Based on Visual Sensory Feedback*. World Scientific, Singapore (1993)
- Hosoda, K., Sakamoto, K., Asada, M.: Trajectory generation for obstacle avoidance of uncalibrated stereo visual servoing without 3-D reconstruction. *IEEE/RSJ Int. Conf. Intell. Robots Syst.* **3**, 29–34 (1995)
- Hu, G., Gans, N., Fitz-Coy, N., Dixon, W.: Adaptive homography-based visual servo tracking control via a quaternion formulation. *IEEE Trans. Control Syst. Technol.* **18**(1), 128–135 (2010)
- Janabi-Sharifi, F., Wilson, W.J.: Automatic selection of image features for visual servoing. *IEEE Trans. Robot. Autom.* **13**(6), 890–903 (1997)
- Kallem, V., Dewan, M., Swensen, J.P., Hager, G.D., Cowan, N.J.: Kernel based visual servoing. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1975–1980. (2007)
- Kelly, R., Moreno, J., Campa, R.: Visual servoing of planar robots via velocity fields. In: *IEEE Conf. Decis. Control (CDC)* **4**, 4028–4033 (2004)
- Kelly, R., Bugarin, E., Sanchez, V.: Image-based visual control of nonholonomic mobile robots via velocity fields: Case of partially calibrated inclined camera. In: *IEEE Conference on Decision and Control*, pp. 3071–3076. (2006)
- Keshmiri, M., Xie, W.F., Ghasemi, A.: Visual servoing using an optimized trajectory planning technique for a 4 dofs robotic manipulator. *Int. J. Control Autom. Syst.* **15**(3), 1362–1373 (2017)
- Kurtser, P., Edan, Y.: Planning the sequence of tasks for harvesting robots. *Robot. Autom. Syst.* **131**, 103591 (2020)
- Kyrki, V., Kragic, D., Christensen, H.I.: New shortest-path approaches to visual servoing. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 349–354, Sendai (2004)
- Li, S., Ghasemi, A., Xie, W., Gao, Y.: An enhanced ibvs controller of a 6 dof manipulator using hybrid pd-smc method”. *Int. J. Control Autom. Syst.* **16**, 844–855 (2018)
- Liu, H., Zhu, W., Dong, H., Ke, Y.: Hybrid visual servoing for rivet-in-hole insertion based on super-twisting sliding mode control. *Int. J. Control Autom. Syst.* **18**, 2145–2156 (2020)
- Malis, E.: Visual servoing invariant to changes in camera-intrinsic parameters”. *IEEE Trans. Robot. Autom.* **20**(1), 72–81 (2004)
- Malis, E., Mezouar, Y., Rives, P.: Robustness of image-based visual servoing with a calibrated camera in the presence of uncertainties in the three-dimensional structure. *IEEE Trans. Robot.* **26**(1), 112–120 (2010)
- Malis, E., Chaumette, F., Boudet, S.: 2–1/2 D visual servoing. *IEEE Trans. Robot. Autom.* **15**(2), 238–250 (1999)
- Mansard, N., Chaumette, F.: Tasks sequencing for visual servoing. In: *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'04)*, Sendai, Japan (2004)
- Mansard, N., Chaumette, F.: A new redundancy formalism for avoidance in visual servoing. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, (2005)
- Mansard, N., Chaumette, F.: Task sequencing for high level sensor-based control. *IEEE Trans. Robot.* **23**, 60–72 (2007)
- Marchand, E., Chaumette, F.: Feature tracking for visual servoing purposes. *Robot. Auton. Syst.* **52**(1), 53–70 (2005)
- Martinet, P., Gallice, J., Khadraoui, D.: Vision Based Control Law using 3D Visual Features. In: *Committees, Econometrica*, pp. 497–502. (1996)
- Mezouar, Y., Chaumette, F.: Path planning for robust image-based control. *IEEE Trans. Robot. Autom.* **18**, 534–549 (2002)
- Nayar, S.K., Nene, S.A., Murase, H.: Subspace methods for robot vision. *IEEE Trans. Robot. Autom.* **12**(5), 750–758 (1996)
- Nematollahi, E., Janabi-Sharifi, F.: Generalizations to control laws of image-based visual servoing. *Int. J. Optomechatronics* **3**, 167–186 (2009)
- Norouzi-Ghazbi, S., Janabi-Sharifi, F.: A switching image-based visual servoing method for cooperative continuum robots. *J. Intell Robot Syst* **103**, 42 (2021)

- Pages, J., Collewet, C., Chaumette, F., Salvi, J.: Optimizing plane-to-plane positioning tasks by image-based visual servoing and structured light. *IEEE Trans. Rob.* **22**(5), 1000–1010 (2006)
- Ren, X., Li, H., Li, Y.: Image-based visual servoing control of robot manipulators using hybrid algorithm with feature constraints. *IEEE Access* **8**, 223495–223508 (2020)
- Sanderson, A.C., Weiss, L.E.: Image-based visual servo control using relational graph error signals. In: *IEEE International Conference on Cybernetics and Society*, pp. 1074–1077. (1980)
- Shi, J., Tomasi, C.: Good features to track. In: *IEEE Conference on Computer Society*, pp 593–600. (1994)
- Shirai, Y., Inoue, H.: Guiding a robot by visual feedback in assembly tasks. *Pattern Recogn* **5**, 99–108 (1973)
- Shu, T., Gharaaty, S., Xie, W.F., Joubair, A., Bonev, I.: Dynamic path tracking of industrial robots with high accuracy using photogrammetry sensor. *IEEE/ASME Trans. Mechatron.* **23**(3), 1159–1170 (2018)
- Tahri, O., Chaumette, F.: Complex objects pose estimation based on image moment invariants. In: *IEEE International Conference on Robotics and Automation*, pp. 436–441. (2005)
- Tahri, O., Chaumette, F.: Application of moment invariants to visual servoing. In: *IEEE International Conference on Robotics and Automation*, vol. 3, pp. 4276–4281. (2003)
- Tahri O., Chaumette, F.: Image moments: generic descriptors for decoupled image-based visual servo. In: *IEEE International Conference on Robotics and Automation*, vol. 2, pp. 1185–1190. (2004)
- Tahri, O., Chaumette, F.: Point-based and region-based image moments for visual servoing of planar objects. *IEEE Trans. Rob.* **21**(6), 1116–1127 (2005)
- Wang, H., Yang, B., Liu, Y., Chen, W., Liang, X., Pfeifer, R.: Visual servoing of soft robot manipulator in constrained environments with an adaptive controller. *IEEE/ASME Trans. Mechatron.* **22**(1), 41–50 (2017)
- Wang, J., Wilson, W.J.: 3D relative position and orientation estimation using Kalman filter for robot control. In: *IEEE International Conference on Robotics and Automation*, pp. 2638–2645. (1992)
- Wilson, W.J., Williams Hulls, C.C., Bell, G.S.: Relative end-effector control using cartesian position based visual servoing. *IEEE Trans. Robot. Autom.* **12**(5), 684–696 (1996)
- Wu, J., Jin, Z., Liu, A., Yu, J., Yang, F.: A survey Of learning-based control of robotic visual servoing systems. *J. Franklin Inst.* **359**(1), 556–577 (2022)
- Xiaolin, R., Hongwen, L.: Uncalibrated image-based visual servoing control with maximum correntropy kalman filter. *IFAC-PapersOnLine* **53**(5), 560 (2020)
- Xu, X., Hu, Y., Zhai, J.M., Li, L.Z., Guo, P.S.: A novel non-collision trajectory planning algorithm based on velocity potential field for robotic manipulator. *Int. J. Adv. Robot. Syst.* **15**(4), 1–13 (2018)
- Zanne, P., Morel, G., Piestan, F.: Robust vision-based 3D trajectory tracking using sliding mode control. In *IEEE Int. Conf. Robot. Autom.* **3**, 2088–2093 (2000)
- Zhao, Y., Xie, W.F., Liu, S.: Image-based visual servoing using improved image moments in 6-DOF robot systems. *Int. J. Control Autom. Syst.* **11**(3), 586–596 (2013)
- Zhao, Y.M., Lin, Y., Xi, F., Guo, S., Ouyang, P.: Switch-based sliding mode control for position-based visual servoing of robotic riveting system. *J. Manuf. Sci. Eng. Trans. ASME.* **139**(4), 1–11 (2017)

Zhao, T., Li, H., Dian, S.: Multi-robot path planning based on improved artificial potential field and fuzzy inference system1. *J. Intell. Fuzzy Syst.* **39**(5), 7621–7637 (2020)

Zhao, X., Emami, M.R., Zhang, S.: Image-based control for rendezvous and synchronization with a tumbling space debris. *Acta Astronaut.* **179**, 56–68 (2021)

Zhong, X., Zhong, X., Hu, H., Peng, X.: Adaptive neuro-filtering based visual servo control of a robotic manipulator. *IEEE Access* **7**, 76891–76901 (2019)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.



Vo Duy Cong received the M.Sc. degree in Mechatronics Engineering from Ho Chi Minh City University of Technology in 2021. He joined the Industrial Maintenance Training Center, Ho Chi Minh City University of Technology, Vietnam National University-Ho Chi Minh City, since 2018. His research interests are mechanical design, vision and robotics.



Le Duc Hanh received the M.Sc. degree in Mechanical and Automotive Engineering from University of Ulsan, South Korea in 2008 and Ph.D degree in Mechanical Engineering from National Taiwan University of Science and Technology, Taiwan in 2012. He joined the Department of Mechanical Engineering, Ho Chi Minh City University of Technology, Vietnam National University-Ho Chi Minh City, since 2008. His research interests are intelligent control, vision and robotics.