Image space trajectory tracking of 6-DOF robot manipulator in assisting visual servoing

Megha G. Krishnan & Ashok Sankar

AUTOMATIKA
2022, VOL. 63, NO. 2, 199–215
https://doi.org/10.1080/00051144.2021.2022889

Image space trajectory tracking of 6-DOF robot manipulator in assisting visual servoing

Megha G. Krishnan and Ashok Sankar

Department of Electrical Engineering, National Institute of Technology, Calicut, Kerala, India

ABSTRACT
As vision is a versatile sensor, vision-based control of robot is becoming more important in industrial applications. The control signal generated using the traditional control algorithms leads to undesirable movement of the end-effector during the positioning task. This movement may sometimes cause task failure due to visibility loss. In this paper, a sliding mode controller (SMC) is designed to track 2D image features in an image-based visual servoing task. The feature trajectory tracking helps to keep the image features always in the camera field of view and thereby ensures the shortest trajectory of the end-effector. SMC is the right choice to handle the depth uncertainties associated with translational motion. Stability of the closed-loop system with the proposed controller is proved by the Lyapunov method. Three feature trajectories are generated to test the efficacy of the proposed method. Simulation tests are conducted and the superiority of the proposed method over a Proportional Derivative – Sliding Mode Controller (PD-SMC) in terms of settling time and distance travelled by the end-effector is established in the presence and absence of depth uncertainties. The proposed controller is also tested in real-time by integrating the visual servoing system with a 6-DOF industrial robot manipulator, ABB IRB 1200.

ARTICLE HISTORY
Received 5 January 2021
Accepted 19 December 2021

KEYWORDS
Feature point trajectory; image space tracking; sliding mode control; depth uncertainties

1. Introduction

The requirement for automated industrial operations is driving up the demand for robots in manufacturing industries. Industrial robots, mainly robot manipulators play a key role in industrial automation. Robotic manipulators have been used in various industrial applications like spot welding, material handling, pick & place and many more. It requires high endurance, speed and meticulousness. However, the application of manipulators in industries is limited by their lack of intelligence to take decisions. To overcome this problem, a vision sensor is integrated into the robot control systems. This provides a better operation and aids the robot to navigate the landscape and avoid collisions. In visual servoing, the data acquired from the vision sensors are used to control the motion of a robot. Mathematically, the error between the desired and actual visual features is minimized. On the other hand, the loss of data while projecting the 3D information onto a 2D image plane in the camera is a challenge in vision-based control. Moreover, the non-linearities and complex structure of a manipulator robot make the problem more complex [1].

The research on the vision-based control of robots was started in the early 1990s. It includes research on computer vision, robotics and control systems. Based on the method of calculating error to generate the velocity profile, the aforementioned method is mainly categorized into two: image-based visual servoing (IBVS) and position-based visual servoing (PBVS) [2]. In PBVS, the relative pose (position as well as orientation) between the target object and end effector/camera is recovered from the images obtained by the eye-in-hand camera. Thus, a Cartesian pose error signal based on the current 3D pose and the desired 3D pose is generated to drive the robot to the target. The main difference of IBVS from PBVS is that the error is defined in task space, not in world space. Hence, 3D reconstruction of the current image is not required. The image features at the final pose implicitly define the desired camera pose relative to the target [3]. IBVS uses the desired and current feature positions in the image plane to drive the motion of the robot. From the fundamental characteristics and stability analysis, IBVS is robust towards calibration error and image noise. Furthermore, the formulation and implementation of this algorithm are simple compared to PBVS [4,5]. As the image features are highly non-linear functions of camera pose, controlling the robot motion with the IBVS scheme is a challenging control problem [6].

Numerous advanced schemes like optimal, adaptive and predictive controllers have been discussed in the literature to develop a fast, robust, stable and accurate IBVS system. The traditional position-based and image-based visual servoing uses the proportional controller, as it exponentially reduces the error. An optimal

CONTACT Megha G. Krishnan megha_p160007ee@nitc.ac.in Department of Electrical Engineering, National Institute of Technology, Calicut 673601, Kerala, India

visual PD controller to align the micropeg and hole is presented in which a genetic algorithm approach is used to tune the controller parameters [7]. For binocular visual servoing, a combined PI motion controller with PID neural network visual controller is discussed [8]. Even though P, PD, PI and PID control strategies are simple and effective, they cannot cope with the disturbances that occur in the image during the motion. Some researchers tried guidance and navigation techniques [9] and augmented IBVS [10] techniques to generate acceleration commands for the robot control. Nevertheless, these methods cannot be applied to robots that accept only the velocity command.

Optimal control scheme such as the linear quadratic method is detailed in the literature to deal with the redundant features. But, the robustness and the accuracy of the redundant feature system are not investigated [11]. IBVS control strategy is treated as a non-linear optimization problem based on predictive control strategy. This method deals with the robot workspace limitations, visibility constraints, and actuators constraints [12]. However, the computational burden in solving the optimization problem makes the method least significant for dynamically fast robotic applications. Robust model predictive control methods are introduced later to compensate for the mismatches between the actual and nominal systems caused by uncertainties [13,14]. The visibility constraint discussed in predictive control is handled by planning trajectory in image space or Cartesian space. The trajectory planning techniques which guarantee the best or near best results to accomplish a given task are introduced in PBVS. But they suffer from the PBVS drawbacks like camera calibration errors. An optimized trajectory planning technique is suggested in the literature to achieve the IBVS task where the velocity screw of the camera is parameterized using time-based profiles [15].

Benavent et al. developed SMC for visual servoing with discontinuous control signals like joint accelerations and jerks [16]. SMC is also developed to guarantee tracking performance and robustness in position-based visual servoing [17–19]. Nevertheless, the chattering problem associated with the SMC is not completely eliminated in most situations [20–22]. Ghasemi et al. elaborated a hybrid PD-SMC approach for IBVS of a 6-DOF robot manipulator. This approach guarantees stability, reduces chattering phenomenon associated with SMC and has a fast convergence rate. But the unknown path followed by the robot may violate the robot joint limits, especially during complex rotational and translational end-effector motion commands [23].

In this paper, the problem of feature loss due to the undesirable motion of the camera during image-based visual servoing is encountered. A novel method of tracking feature trajectory in an image plane using SMC is proposed for robust image-based visual servoing. This method makes the IBVS system deal with FOV constraints by simply following the generated trajectory in image space. The proposed controller deals with the depth uncertainties during the translational motion. Three trajectories are generated in image space with a pre-defined time interval. As a result, partial predetermination of the convergence time of the system is achieved. The major contributions of this paper are

(1) Introduced an online trajectory planning technique to guarantee a camera position in which the features are always in the FOV of the camera.
(2) Sliding mode controller is designed to make the IBVS task robust against depth uncertainties during translational motion.

The rest of this paper is organized as follows. The visual servoing system description and the sliding mode controller design are described in Section 2. The trajectory generation is discussed in Sections 3 and 4 elaborates the stability analysis of the closed-loop system. Experiments conducted and the performance evaluation of the proposed method are detailed in Section 5. Conclusions and future works are portrayed in Section 6.

2. System description

In image-based visual servoing, the positioning task is performed by minimizing the error between the current and desired image features in the image coordinates. The feature error vector is the input to the IBVS controller. The controller generates the end-effector velocity which, is then converted to robot motion and accordingly, the robot movement is controlled based on the visual feedback.

In this section, a brief description of the proportional IBVS controller is given. The four corner points of the target object in image plane are considered as the image features. The desired/reference image features are given by $f_d = [x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4]$. The current image features are given by $f_c = [x_1', y_1', x_2', y_2', x_3', y_3', x_4', y_4']$ which are obtained from the image taken by the camera attached to the end-effector. Thus the image feature error, e is given by

$$e = f_d - f_c(t) \tag{1}$$

The aim of the visual servo controller is to regulate this error to zero. The current values of image features $f_c(t)$ are continuously updated while the end-effector camera moves whereas the desired image features f_d is constant. Using a standard perspective projection model with focal length f, the relationship between an image point velocity and camera velocity is given by

$$\dot{e} = J_{img} V_e \tag{2}$$

where $\dot{e}$ is the time variation of the image feature error, J_{img} is the image Jacobian given by

$$J_{img} = \begin{bmatrix} \frac{-f}{Z} & 0 & \frac{x}{Z} & \frac{xy}{f} & -\frac{f^2+x^2}{f} & y \\ 0 & \frac{-f}{Z} & \frac{y}{Z} & \frac{f^2+y^2}{f} & -\frac{xy}{f} & -x \end{bmatrix} \quad (3)$$

and the spatial velocity V_e of the camera or end-effector is given by

$$V_e = \begin{bmatrix} v_e \\ \omega_e \end{bmatrix} = \begin{bmatrix} v_x \\ v_y \\ v_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \quad (4)$$

where v_e is the instantaneous linear velocity of the camera frame and ω_e is the instantaneous angular velocity of the camera frame. The image Jacobian represents the differential relationship between the pixel frame and the camera frame attached to the robot end-effector. Hence the end-effecter velocity is obtained using simple proportional control law given by,

$$V_e = -\lambda {J_{img}}^{\dagger} e \quad (5)$$

where λ is the proportional gain. Since there is no direct control over the camera motion, there is a chance of undesirable camera motion, particularly during pure rotation motion about the z-axis. This unachievable camera motion may result in task failure as the features leave the camera field of view. The camera motion is streamlined by designing a trajectory in Cartesian space or image space. Since the calculation of the desired feature point in the image plane is computationally cheap and needs to perform only once, the feature trajectory is designed in image coordinates. The computation of image Jacobian requires camera intrinsic parameters and depth of the feature points. Even though IBVS is significantly tolerant towards errors, the imprecise knowledge of the depth of point features affects the IBVS stability and convergence.

2.1. Sliding mode controller design

The traditional IBVS controller can be easily designed and tuned since a proportional gain is used to regulate the feature error. However, the above controller fails in most of the IBVS cases which require critical control law to achieve the target. Hence, a feature trajectory-based sliding mode controller for IBVS is introduced to improve the system robustness and stability in all IBVS cases where the object is initially visible to the eye-in-hand camera. A feature-based trajectory is designed in the image plane and the proposed controller is designed in such a way that the current features $f_c(t)$ reach the desired one through the feature values in pre-defined trajectory $f(t)$ in image space. Instead of constant desired features in the proportional controller, the desired feature vector is time-varying as the controller is designed for tracking the feature trajectory.

To ensure tracking of the image features, a sliding mode based control law is selected which can handle the uncertainties in the image Jacobian. In order to have the system track $f_c(t) \equiv f(t)$, a sliding surface s is designed in which $s = 0$ as $t \to \infty$.

$$s = e = f_c(t) - f(t) \quad (6)$$

where $f_c(t)$ is the 8×1 column vector of current image feature points and $f(t)$ is the 8×1 column vector of designed image trajectory and e is the tracking error in image coordinates. Differentiating Equation (6),

$$\dot{s} = \dot{f_c} - \dot{f} \quad (7)$$

From Equation (2),

$$\dot{s} = J_{img} V_e - \dot{f} \quad (8)$$

According to the constant plus proportional reaching law,

$$\dot{s} = -K_s \mathrm{sgn}(s) - K_p s \qquad K_s > 0, K_p > 0 \quad (9)$$

where $\dot{s} = -K_p s$ is the exponential term whose solution is $s = s(0)e^{-K_p t}$. When s is large, the state is forced to approach the switching manifolds faster compared to simple proportional reaching law. Equating Equations (8) and (9),

$$J_{img} V_e - \dot{f} = -K_s \mathrm{sgn}(s) - K_p s \quad (10)$$

Hence the control law is defined as

$$V_e = \hat{J}^{\dagger}_{img}(\dot{f} - K_p e - K_s \mathrm{sgn}(s)) \quad (11)$$

where $\hat{J}^{\dagger}_{img}$ is the Moore-Penrose pseudo inverse of the estimated image Jacobian, K_p and K_s are strictly positive constants and $\mathrm{sgn}(s)$ is a signum function. The joint velocity of the robot manipulator is obtained from the end-effector velocity as

$$\dot{\theta} = J^{-1}_{robot} V_e \quad (12)$$

where J_{robot} is the manipulator Jacobian of the robot. The proportional term in the control law improves the tracking performance and system speed, whereas the robustness and stability are achieved using the sliding mode controller. The workflow of the proposed algorithm is summarized in Figure 1.

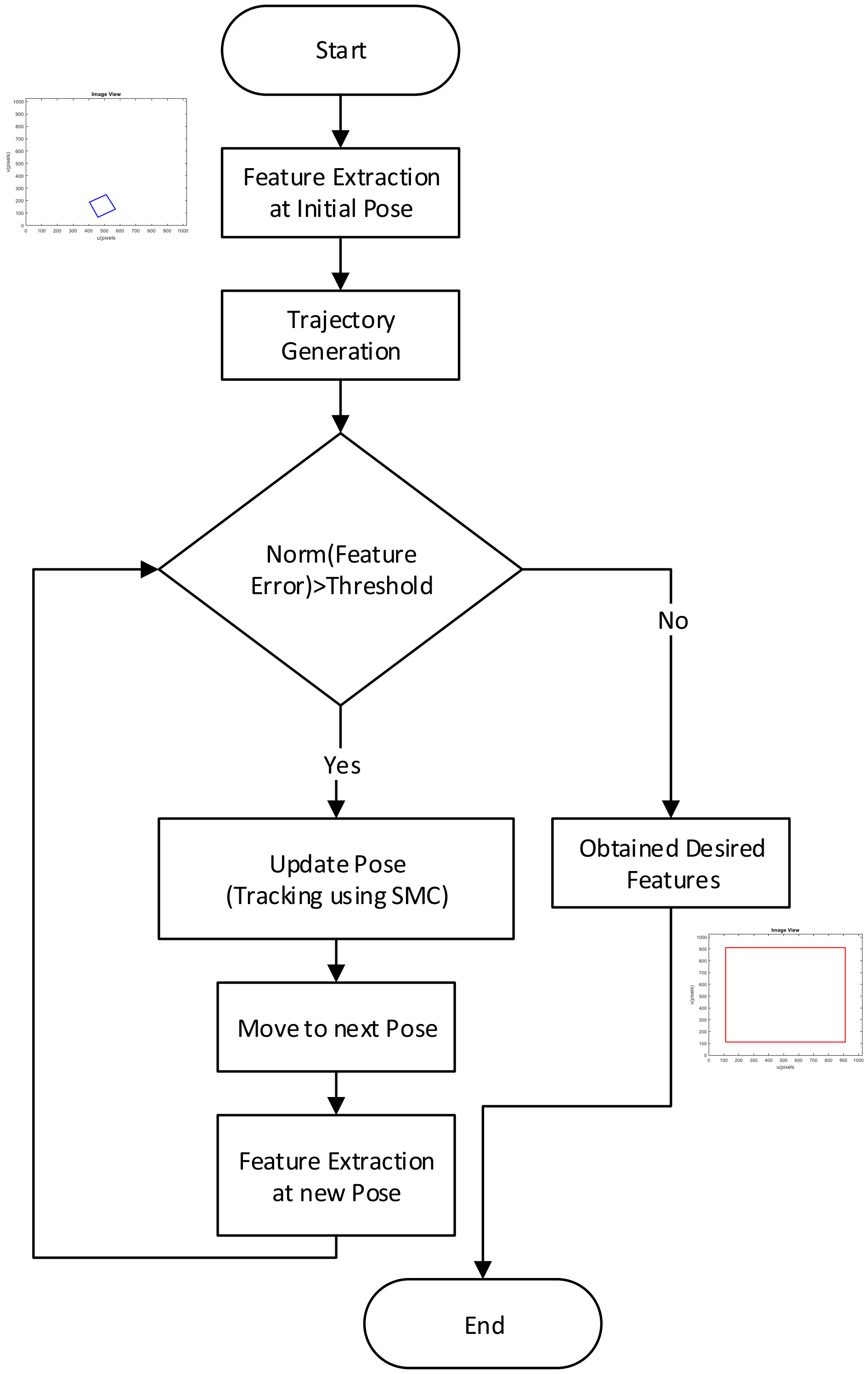
Start
Feature Extraction at Initial Pose
Trajectory Generation
Norm(Feature Error)>Threshold
No
Yes
Update Pose (Tracking using SMC)
Move to next Pose
Feature Extraction at new Pose
Obtained Desired Features
End
Image View
u(pixels
v(pixels)

Figure 1. Flowchart of the proposed algorithm for visual servoing.

3. Trajectory generation

In this section, the method of trajectory planning in image space is discussed. Instead of planning the camera path directly, the trajectory of features is generated in the image plane, thus preserving the advantages of IBVS. The IBVS scheme is designed to move the end-effector from the initial image feature location to the desired image feature location through a predefined feature trajectory. Based on the current and the desired feature values, the following three trajectories are planned.

(a) Straight line
(b) Cubic spline
(c) Bezier curve

A straight line trajectory with time interval T is given by

$$x(t) = x_0 + \frac{(x_d - x_0)t}{T}$$
$$y(t) = y_0 + \frac{(y_d - y_0)t}{T}; \quad 0 \le t \le T \qquad (13)$$

where $f_0 = (x_0, y_0)$ is the initial feature value in pixels and $f_d = (x_d, y_d)$ is the desired feature values in pixels. The controller has to be designed in such a way that, the end effector should follow a path to track the designed trajectory, $f = x(t), y(t)$ in image plane coordinates.

A cubic spline trajectory with time interval T is given as

$$x(t) = x_0 + \frac{(x_d - x_0)t}{T}$$
$$y(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3; \quad 0 \le t \le T \qquad (14)$$

where a_0, a_1, a_2, a_3 are obtained by substituting the initial and final image features. Here $x(t)$ will be a straight line and $y(t)$ will be cubic spline trajectory and y Vs x will give cubic spline trajectory in the image plane. Hence four spline trajectories are generated for four corner points in the image plane, which is the reference trajectory.

A Bezier curve is a mathematically defined curve that uses Bernstein polynomials as the basis. The curve is defined by various points like the initial points, the terminating points called anchors and two or more separate middle points which are called handles. The shape of a Bezier curve is altered by moving the handles. Given a set of $n+1$ control points like $P_0, P_1, P_2, \dots . P_n$, the corresponding Bezier curve of order n is given by

$$C(t) = \sum_{i=0}^{n} P_i B_{i,n}(t), \qquad t \in [0,1] \qquad (15)$$

where $B_{i,n}(t)$ is a Bernstein polynomial. The cubic bezier curve is defined by four points P_0, P_1, P_2 and P_3. The curve begins at P_0 and moves toward P_1, eventually arriving at P_3 from the direction of P_2. The bezier curve always passes through the first and last control points and lies within the convex hull of the control points. The control points may not lie on the curve, but it defines the bezier curve and hence the curve is always inside the convex hull of the control points [24]. The explicit form of cubic bezier curve is

$$C(t) = (1-t)^3 P_0 + 3(1-t^2)tP_1$$
$$+ 3(1-t)t^2 P_2 + t^3 P_3 \quad t \in [0,1] \qquad (16)$$

The major problem associated with the bezier curve is that change in any of the control points affects the entire curve and hence the design of the curve for specific applications seems difficult.

The above three trajectories are valid for trajectory tracking application since the four corner points obtained during trajectory generation at each instant map to one and only one camera pose which in turn corresponds to a single robot pose. In addition, smooth trajectory in image space indicates the smooth trajectory of the camera attached to the end effector. The feature trajectory planning technique mentioned above has the main benefit that merely the initial and the desired feature points are essential to plan the entire feature trajectory. Moreover, the convergence time of the entire visual servoing task is partially regulated using this method.

4. Stability analysis

To study the tracking ability of the proposed controller in the presence of parameter uncertainties, assume two conditions. First, assume that the parameter uncertainties are within the range space of the matrix J_{img}. Also, the estimated Jacobian matrix $\hat{J}_{img}$ is invertible ($\hat{J}_{img}$ must be full rank) and $\hat{J}_{img} = J_{img}$ in the absence of uncertainties. Hence, the relationship between the image Jacobian and its estimated value is written as

$$I + \Delta_{\min} \le J_{img}\hat{J}_{img} \le I + \Delta_{\max} \qquad (17)$$

where I is the $n \times n$ identity matrix.The derivation for the uncertainty matrices is given in Appendix.

According to the Lyapunov direct method of stability analysis, the system's stability is guaranteed if the following equation is satisfied [25].

$$\frac{1}{2}\frac{d}{dt}s^2 \le -\eta|s| \qquad (18)$$

where η is a strictly positive constant. The Lyapunov function is taken as

$$L = \frac{1}{2}s^T s \qquad (19)$$

The first derivative of Lyapunov function,

$$\dot{L} = s^T \dot{s} \qquad (20)$$

Substituting Equation (11) in Equation (8),

$$\dot{s} = J_{img}\hat{J}^{\dagger}_{img}(\dot{f}_d - K_p e - K_s \mathrm{sgn}(s)) - \dot{f}_d \qquad (21)$$

$$\dot{s} = (J_{img}\hat{J}^{\dagger}_{img} - I)\dot{f}_d - K_p J_{img}\hat{J}^{\dagger}_{img} e - K_s J_{img}\hat{J}^{\dagger}_{img}\mathrm{sgn}(s))$$

From eq. (17), eq. (21) is re-written as

$$\dot{s} = \Delta\dot{f}_d - K_p(I+\Delta)e - K_s(I+\Delta)\mathrm{sgn}(s)) \qquad (22)$$

The sliding condition is verified if,

$$s^T\dot{s} = \Delta\dot{f}_d s^T - (I+\Delta)s^T K_p s - K_s(I+\Delta)|s|) \le -\eta|s| \qquad (23)$$

The value of K_p is always chosen as a positive constant, which improves the decay rate of the feature error to zero. Hence $K_p > 0$. Then K_s must satisfy the following condition.

$$K_s > diag\left(\frac{\eta + \Delta_{\max}|\dot{f}_d|}{I + \Delta_{\min}}\right) \tag{24}$$

where $\Delta_{\min}$ and $\Delta_{\max}$ are the matrix of uncertainties associated with lower and upper bounds of estimated depth $Z_{\min}$ and $Z_{\max}$ respectively. Now the convergence of visual servo task of the system with parametric uncertainties is guaranteed.

5. Results and discussion

5.1. Simulation studies

The simulation studies have been carried out to analyze and compare the performance of the proposed method with the hybrid PD-SMC method discussed in [23] for different robot motion in Cartesian space. The robot model considered for simulation is ABB make IRB 1200 manipulator [26]. It is a compact and flexible 6-DOF industrial robot equipped with a pneumatic gripper for pick and place applications. The model of the robot is explained by its DH parameters given in Table 1.

In all the tests conducted, the eye-in-hand camera configuration is used in which the camera is attached to the end-effector of the robot. The visual target is assumed to be four points defined by the corners of a square object lying on a planar work object with respect to the robot base coordinate system. A class of central perspective camera is generated using the toolbox and it is assumed that the tool centre point (TCP) to camera transformation matrix, ${}^tT_c = I_4$ where I_4 is the identity matrix of dimension 4 i.e. there is no transformation between the pose of the camera and the robot TCP. The parameters of the imaging model are displayed in Table 2.

Four different tests with different camera movements are performed to validate the proposed algorithm. Simulation studies are conducted in MATLAB 2018a with the help of Robotics and Machine Vision Toolbox [27]. The initial and desired feature values selected for each test are given in Table 3. In all the tests, the sampling time is set as 0.1sec and the threshold value for feature error norm is limited to 0.0001. The image trajectory is generated for 4 s and hence the task will be completed after 4 s in all tests.

Table 2. Camera parameters.

Type	Central perspective
Focal Length	0.008 m
Principal point	(512,512) pixel
Pixel size	(1e-05, 1e-05) m/pixel
Image plane limit	(0-1024) x (0-1024) pixel

Table 3. Initial and desired feature points in pixels.

Tests		Corner 1	Corner 2	Corner 3	Corner 4
Test 1	I	(322,492)	(322,592)	(222,592)	(222,492)
	D	(712,312)	(712,712)	(312,712)	(312,312)
Test 2	I	(453,293)	(731,453)	(571,731)	(293,571)
	D	(672,352)	(672,672)	(352,672)	(352,352)
Test 3	I	(281,775)	(348,859)	(262,937)	(194,850)
	D	(712,312)	(712,712)	(312,712)	(312,312)
Test 4	I	(779,49)	(846,118)	(777,173)	(709,102)
	D	(712,312)	(712,712)	(312,712)	(312,312)

Test 1: The aim of this test is to show the performance of the proposed algorithm when a significant translational motion of the camera is needed for visual servoing. Initially, the camera is at a distance of 0.7 m from the target object where pure translational motion is needed to reach the target. The control parameters are set as $K_p = 0.9$ and $K_s = 0.01$. The straight line trajectory is designed to follow the image features in the proposed method. The image trajectory, camera velocity and feature errors of the proposed method are compared with the PD-SMC method in [23] and plotted. As the end-effector requires pure translational motion to achieve the target, the image, as well as camera trajectory, is a straight line for PD-SMC (Refer Figure 2(a)). A straight line image feature trajectory is planned and the tracking ability of the proposed method is seen in Figure 2(b). The end-effector velocity and the feature errors given in Figure 3 indicate that a smooth velocity profile is obtained for the proposed controller. The image feature errors converged to zero within 5.4 s for the proposed controller and 12.3 s for PD-SMC. Compared with the PD-SMC method, the proposed controller brings the robot manipulator to the desired pose in less time.

Test 2: The test is conducted to study the performance of the proposed method during the significant roll motion of the camera. The position of the target object is selected in such a way that a rotation of 60° about z-axis is needed to achieve the task. The value of control parameters K_p and K_s are set as 0.9 and 0.01 respectively. The image trajectory for both methods is shown in Figure 4. The straight line trajectory is

Table 1. DH parameters of IRB1200 robot.

Joint/link	θ_{n+1} (degree)	d_{n+1} (mm)	a_{n+1} (mm)	α_{n+1} (degree)	Max velocity (^{0}/s)
1	θ_1	$d_1 = 399$	0	−90	288
2	θ_2-90^0	0	$a_2 = 350$	0	240
3	θ_3	0	$a_3 = 42$	−90	300
4	θ_4	$d_4 = 351$	0	+90	400
5	θ_5	0	0	−90	405
6	θ_6-180^0	$d_6 = 82$	0	0	600

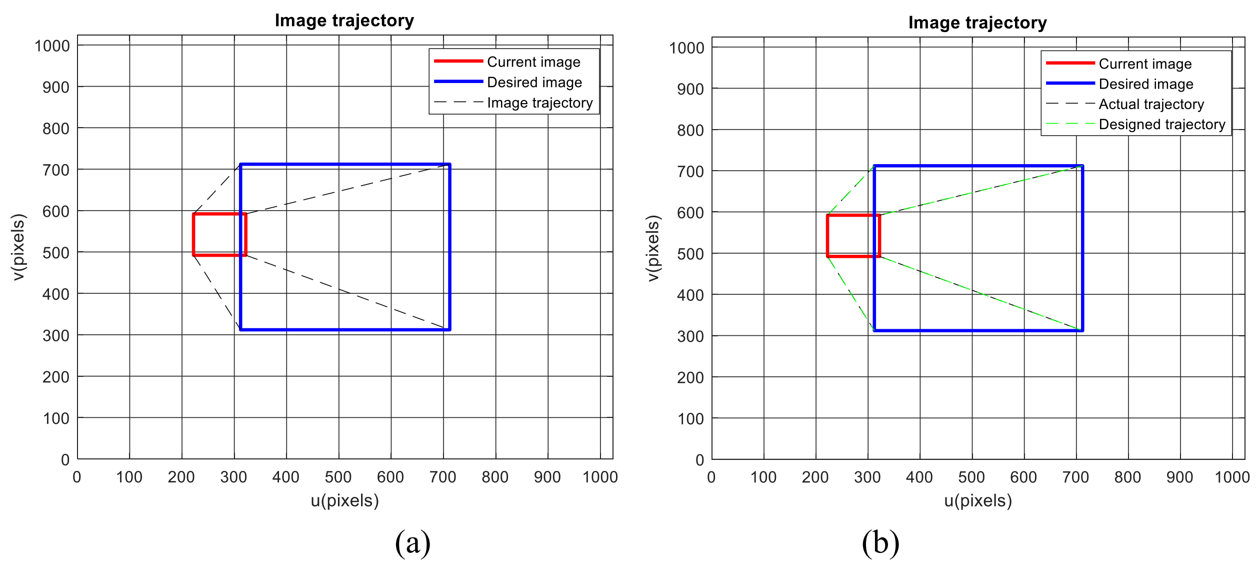

Figure 2. Image trajectory (a) PD-SMC (b) proposed method (test 1).

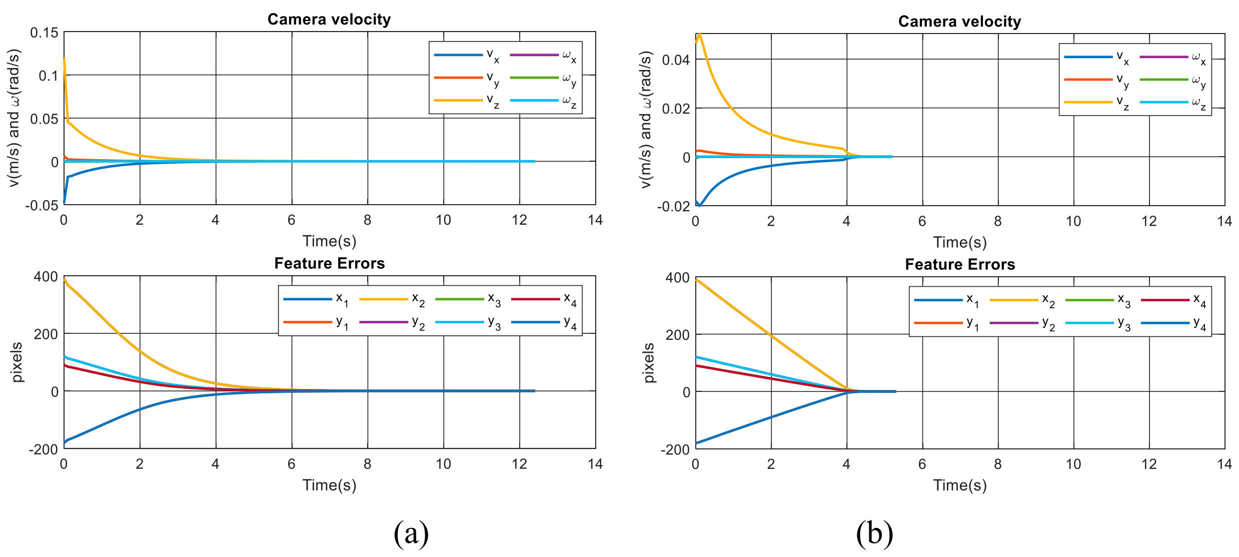

Figure 3. Camera velocity and feature errors (a) PD-SMC (b) proposed method (test 1).

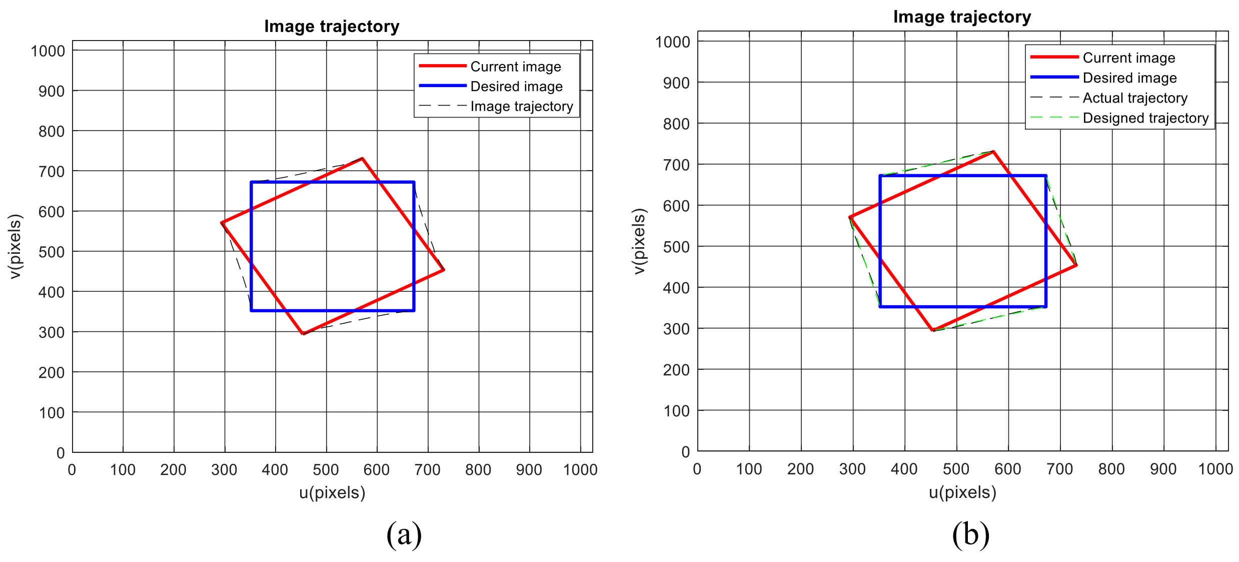

Figure 4. Feature trajectory in image plane (a) PD-SMC (b) proposed method (test 2).

designed to follow the image features in the proposed method. The image trajectory for the PD-SMC method is also straight line as the camera requires pure rotation motion to achieve the target. The camera velocity and the feature error profile given in Figure 5 indicate that the proposed controller moves the robot to the target about 3 times faster compared to the PD-SMC method. There is a spike in the Cartesian velocity curve (both translational and rotational velocity component about z-axis) of the PD-SMC method. This may cause an undesired movement in the robot end-effector, which is not acceptable in the real-time positioning application. In the case of the proposed approach, the linear and angular velocity components about z-axis generate a smooth velocity profile which guarantees a smooth robot motion in the workspace.

Test 3: In test 3, the initial camera pose is selected in such a way that a significant rotational and translational motion of the camera/end-effector is necessary for achieving the goal pose. The target object is kept at a depth of 0.8 m from the camera. The value of K_p and K_s is set as 0.9 and 0.01 respectively. While achieving the positioning task, a curved image trajectory is generated for the PD-SMC method (Figure 6(a)). In the case of the proposed method, the image features exactly follow straight line trajectory designed in the image plane (Figure 6(b)). The end-effector velocity and the corresponding feature error curves are smooth for the proposed method compared to that of the PD-SMC method (Figure 7). Figure 8 shows the camera trajectory which is almost straight for the proposed method whereas for the PD-SMC method, it is a curved trajectory.

Test 3 is also conducted by designing cubic spline and cubic bezier feature trajectory to show the tracking performance of the proposed algorithm. Cubic spline trajectory takes $t = 7.9$sec to converge with $K_p = 0.3$ and $K_s = 0.01$. For Bezier trajectory, the values of K_p

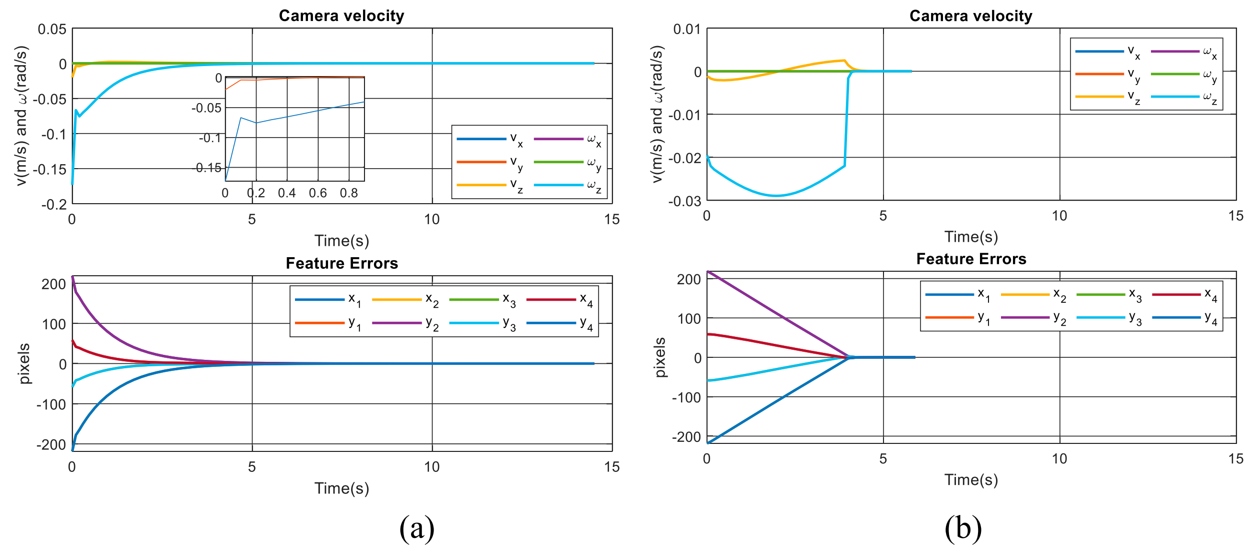

Figure 5. Camera velocity and feature error (a) PD-SMC (b) proposed method (test 2).

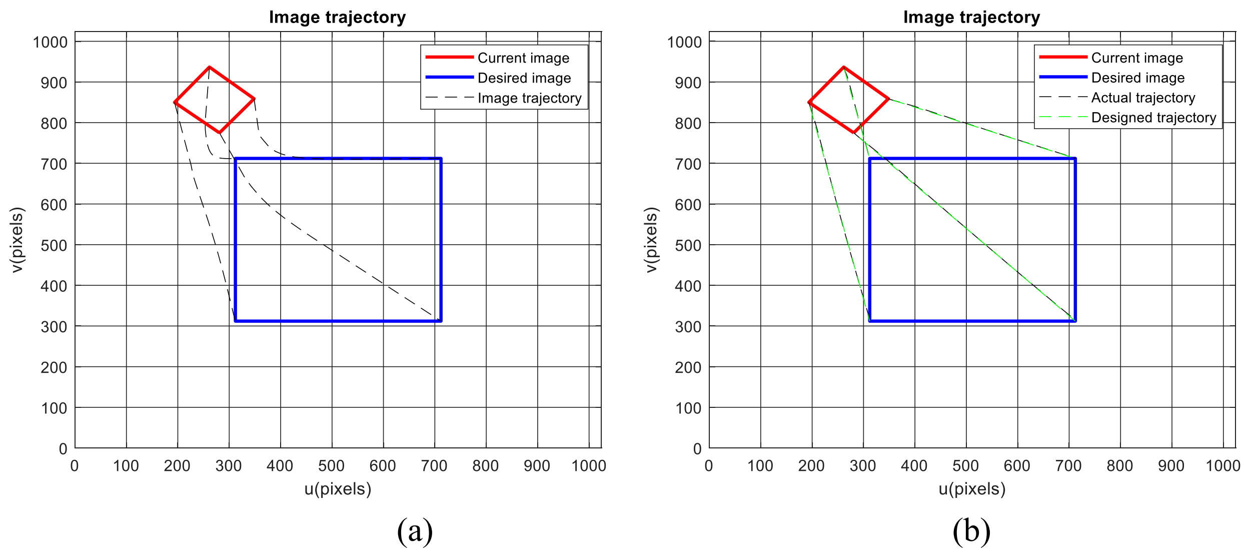

Figure 6. Image trajectory (a) PD-SMC (b) proposed method (test 3).

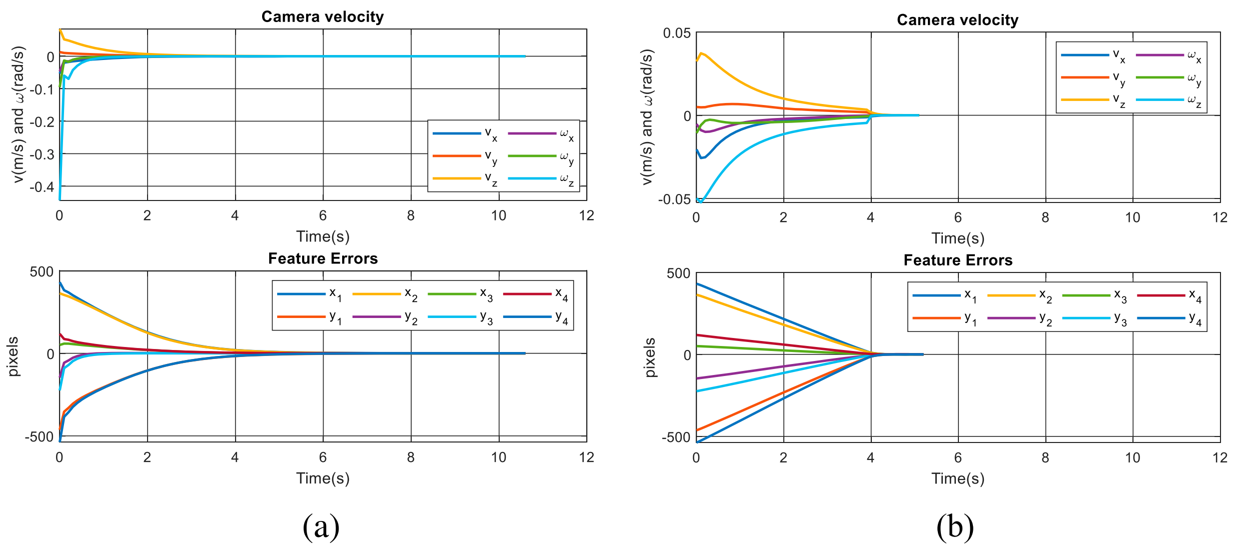

Figure 7. Camera velocity and feature errors (a) PD-SMC (b) proposed method (test 3).

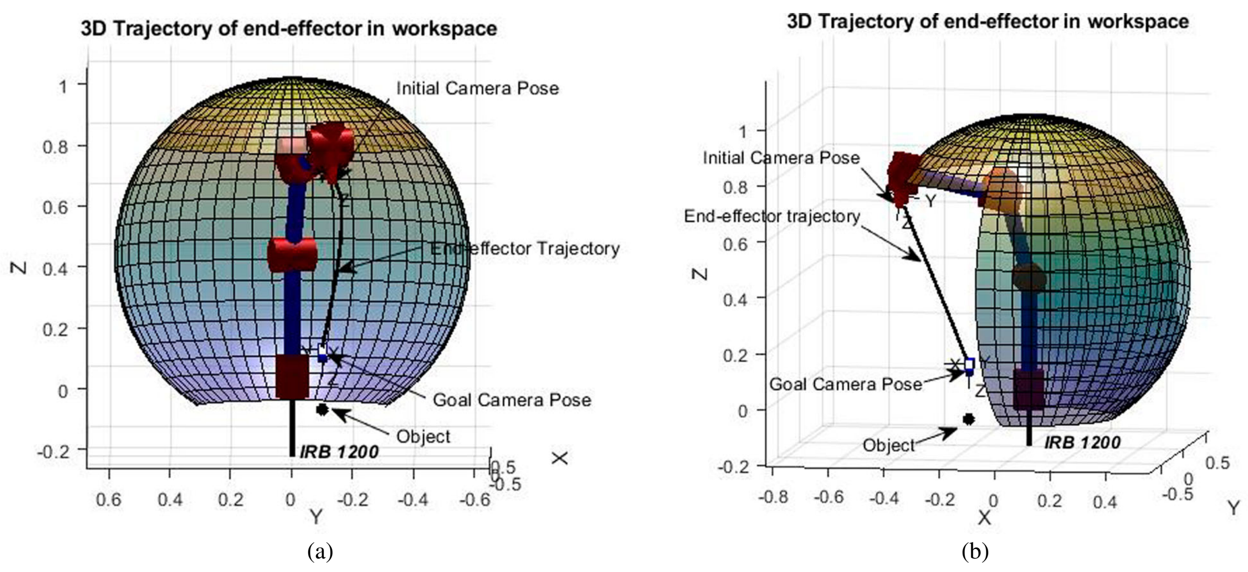

Figure 8. End-effector trajectory in Cartesian space (a) PD-SMC (b) proposed method (test 3).

and K_s are set as 0.9 and 0.01 respectively and the task converged in 4.9 sec. From the image trajectory for cubic spline given in Figure 9(a), it is observed that the image features are not able to exactly track the designed cubic spline trajectory, even though the task converged. The camera velocity profile for cubic spline trajectory is not continuous (Figure 9(b)) and these unwanted oscillations affect the smooth operation of the robot. Also, the convergence time is 7.9 s which is much more compared to that for the straight line trajectory. Figure 10(a) indicates that the proposed controller shows better tracking performance for cubic bezier trajectories. But the oscillations in both the translational and angular velocity components are undesirable for the smooth movement of the robot towards the target. The camera/end-effector trajectory in Cartesian space corresponding to the cubic spline feature trajectory in image space is shown in Figure 11(a). It is clear that the camera follows the lengthiest path to track the cubic spline feature trajectory and thereby, the distance covered by the end-effector to reach the target is also more. Figure 11(b) shows the end-effector trajectory in Cartesian space corresponding to cubic bezier curve in image space. From this, it is obvious that the proposed controller moves the robot to the target through the shortest path while following the bezier image trajectory. From the simulation results, it is evident that the proposed controller shows the best performance while tracking straight line trajectory in the image plane.

In order to show the robustness of the proposed method against external disturbances, test 3 is conducted with a disturbance added in-depth signal and the manipulator is controlled to move from the initial position to the desired position under the added disturbance. A step depth disturbance signal with magnitude 0.7 is added at time 1–2 s for both PD-SMC and the proposed method. Figure 12 depicts the image

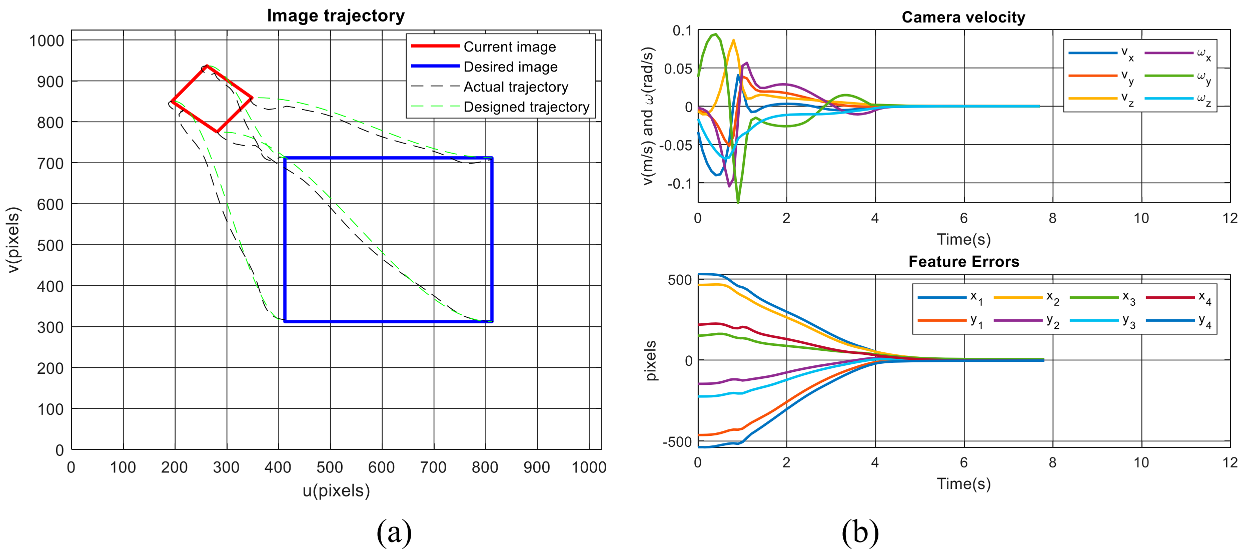

Figure 9. Test 3 using spline curve (a) image trajectory (b) camera velocity and feature errors.

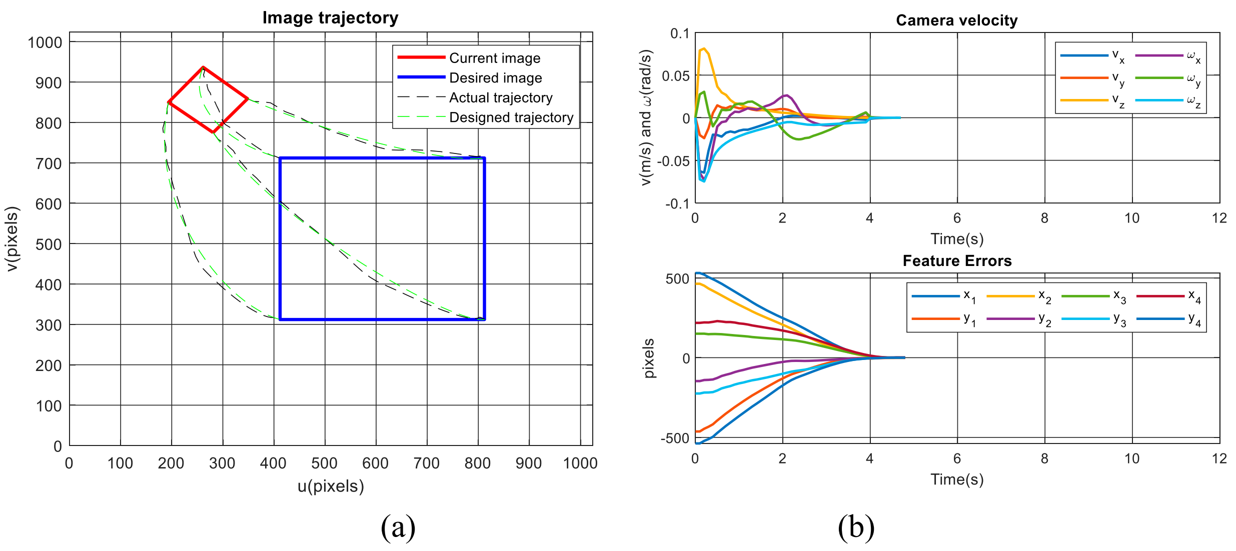

Figure 10. Test 3 using Bezier curve (a) image trajectory (b) camera velocity and feature errors.

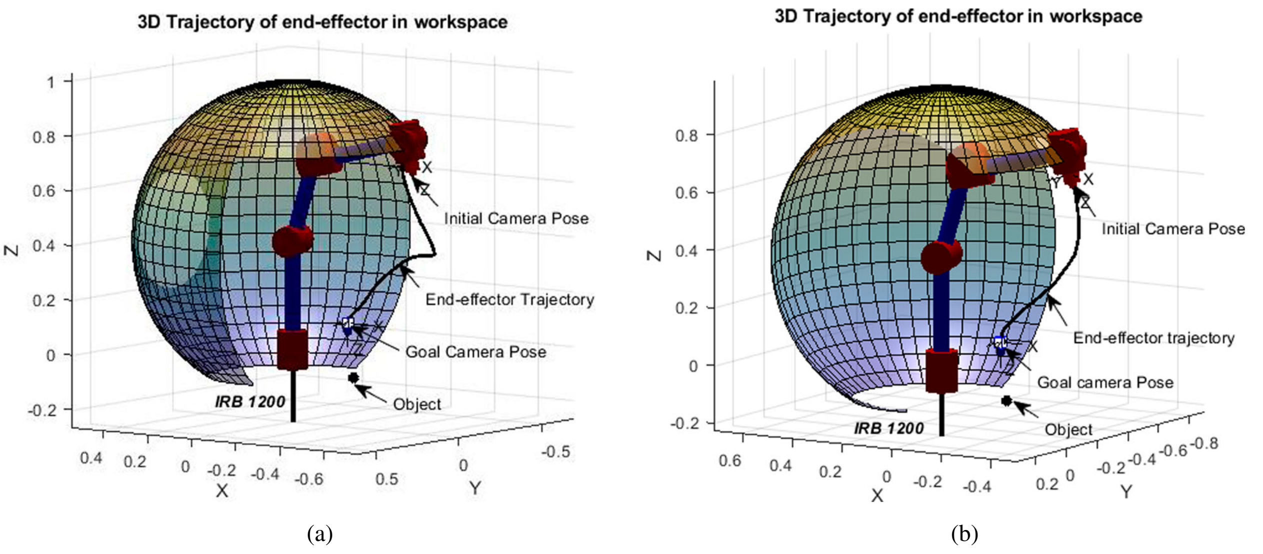

Figure 11. End-effector trajectory in Cartesian space corresponding to (a) cubic spline and (b) cubic bezier trajectory in image space.

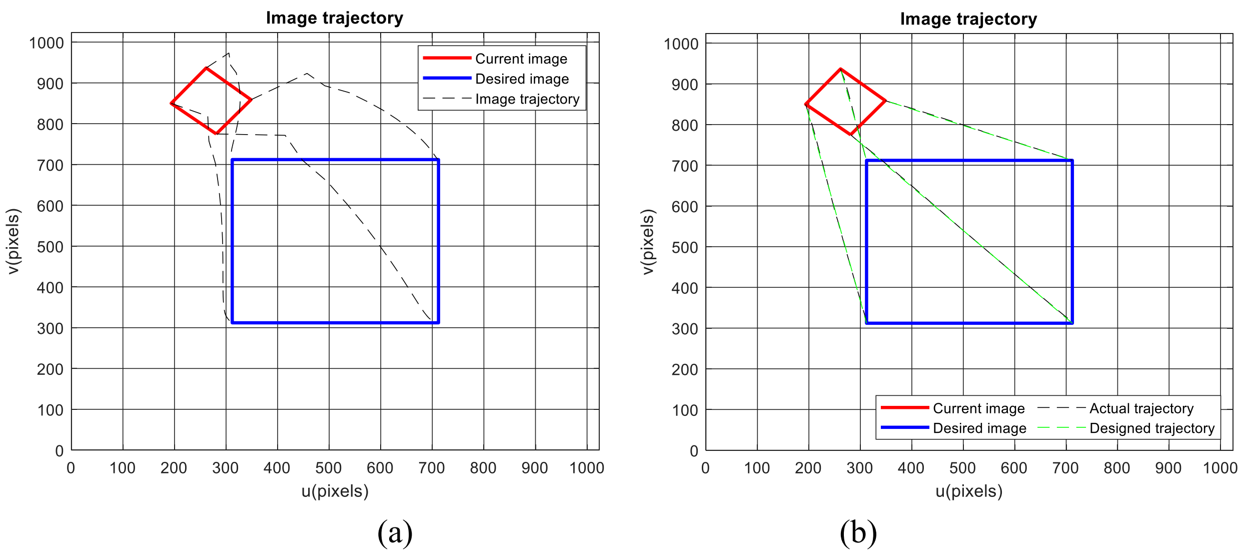

Figure 12. Image trajectory (a) PD-SMC (b) proposed method (test 3 subjected to depth disturbance).

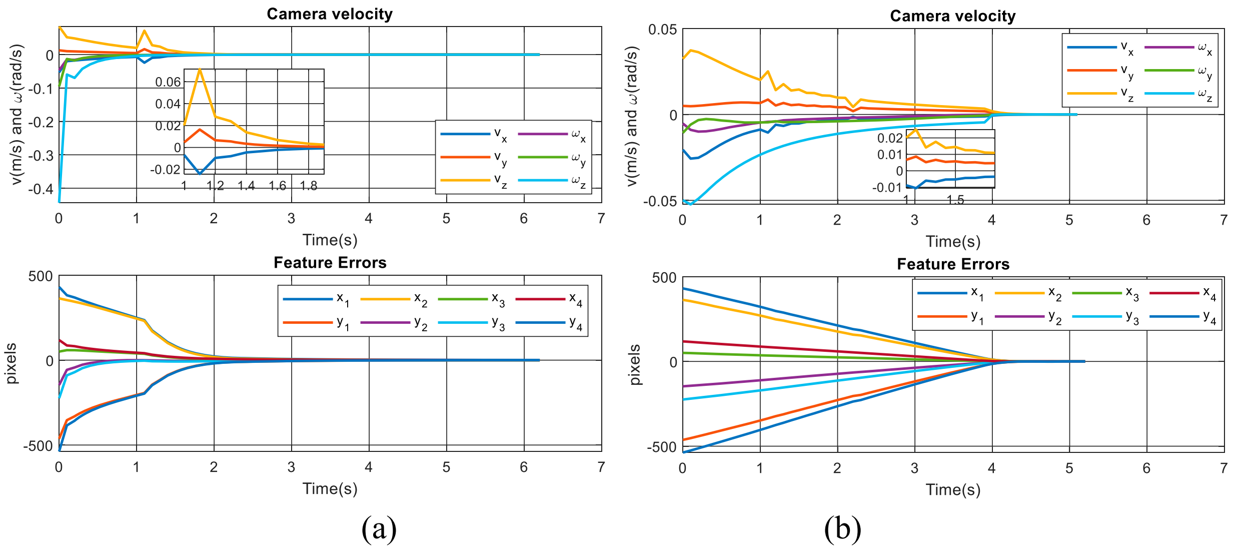

Figure 13. Camera velocity and feature errors (a) PD-SMC (b) proposed method (test 3 subjected to depth disturbance).

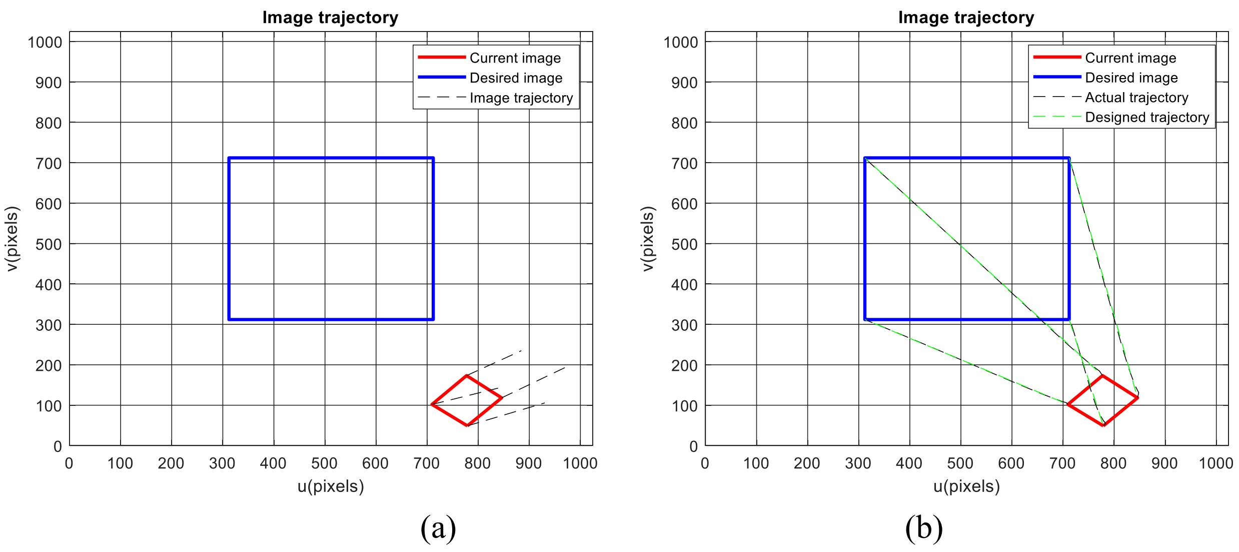

Figure 14. Image trajectory (a) PD-SMC (b) proposed method (test 4).

feature trajectory of the PD-SMC and the proposed method under depth disturbance. The image trajectory of the PD-SMC method is not as smooth as that of the proposed method. Even if the PD-SMC and proposed control law significantly reduce the influence of disturbance, the deviations in the camera velocities for the proposed method are minor (Refer Figure 13). From the simulation results, it is clear that the applied amounts of perturbations are well handled by the proposed controller. All the image feature errors are brought to zero without much change in settling time.

Test 4: The ability of the proposed controller to keep the image features in the FOV of the camera is demonstrated in test 4. The initial camera location is selected in such a way that, there is a complex rotational and translational motion of the end-effector is necessary to reach the goal position. The image trajectory graph in Figure 14 shows that the PD-SMC algorithm fails as the features left the FOV at $t = 0.3$ s. The proposed controller effectively handles the feature visibility constraints by successfully tracking the image features.

The settling time of the visual servoing task is very significant as it indicates how fast the task is completed. The undesirable motion of the robot is identified by calculating the distance travelled by the end-effector while approaching the target object. Therefore settling time and distance travelled by end-effector are considered as the performance indices in this paper and the values are calculated and summarized in Table 4 for all the tests conducted. From the table, it is obvious that the proposed controller performs better for visual servoing applications with minimum settling time. Also, the proposed controller follows the shortest path to reach the target.

Table 4. Performance comparison.

Tests	Controller	Settling time (s)	Distance travelled (mm)
Test 1	PD-SMC	12.3	647
	Proposed controller	5.4	646
Test 2	PD-SMC	14.7	70
	Proposed controller	6.0	61
Test 3	PD-SMC	10.8	672
	Proposed controller for	5.3	634
	(a) Straight line	7.9	1321
	(b) Cubic spline	4.9	798
	(c) Cubic bezier		
	PD-SMC subjected to Disturbance	6.4	672
	Proposed controller subjected to disturbance	5.3	634
Test 4	PD-SMC	Task Failure	
	Proposed controller	5.3	788

Table 5. Camera parameters.

Type	Central perspective
Focal length	0.004 m
Principal point	(512,288) pixel
Pixel size	(2.8e-06, 2.8e-06) m/pixel
Image plane limit	(0-1024) x (0-576) pixel

5.2. *Experimental studies*

In order to validate the performance of the proposed algorithm, experimental tests are also conducted. The experiment setup includes ABB make IRB 1200 robot manipulator equipped with a pneumatic gripper and Logitech camera attached to the gripper. The transformation between the TCP and the camera is given by ${}^{c}T_e = \begin{bmatrix} 1.07 & -0.006 & 0.03 & 0 & 0 & 0 \end{bmatrix}$ in $\begin{bmatrix} x & y & z & \alpha & \beta & \gamma \end{bmatrix}$ format where x, y and z are the Cartesian coordinates in metres and α, β and γ are the roll, pitch and yaw angles in radians. The camera parameters are given in Table 5. The transformation matrix of end-effector to TCP of the gripper attached is calculated as ${}^{e}T_t = \begin{bmatrix} 4.4 & 3.3 & 79.3 & 0 & 0 & 0 \end{bmatrix}$.

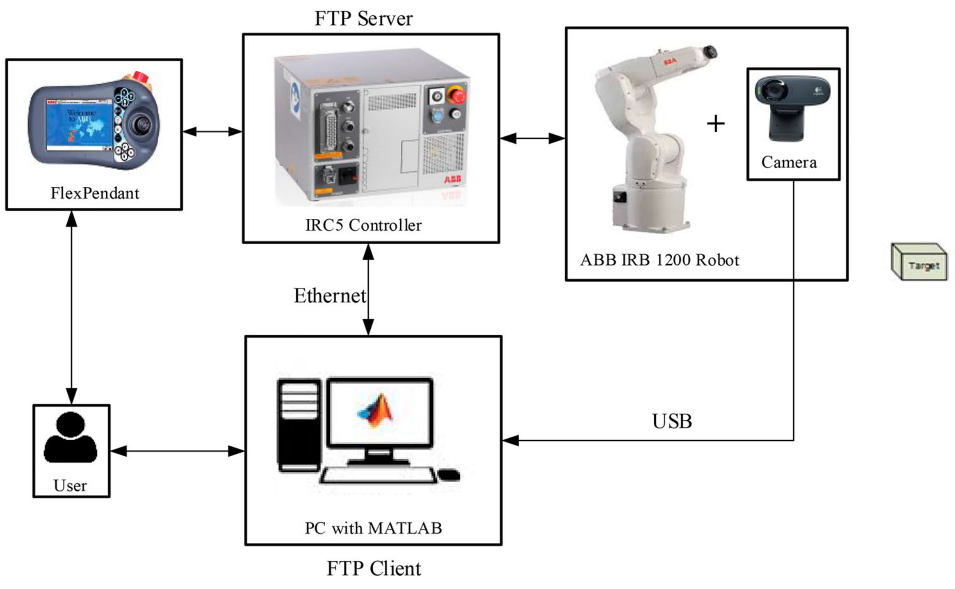

Figure 15. Experimental setup and hardware connections.

A remote PC installed with MATLAB 2018a is communicated with ABB robot controller IRC5 through a File Transfer Protocol (FTP). FTP server on the IRC5 responds to a request from an FTP client on a remote computer. MATLAB provides the basic platform for the programming and control algorithm implementation and acts as the client for FTP communication. MATLAB takes input from the camera and current pose from the IRC5 controller, then performs image processing and implements the proposed control algorithm to predict the new camera pose. The control output is transferred to the controller through Ethernet port as files and that is how the visual servoing task is achieved in real-time [28,29]. The experimental setup is shown in Figure 15.

Table 6. Initial and desired point location in pixels.

Tests		Corner 1	Corner 2	Corner 3	Corner 4
Test 5	I	(652,302)	(695,353)	(507,511)	(464,460)
	D	(974,164)	(974,420)	(27,419)	(28,161)
Test 6	I	(479,124)	(545,124)	(545,366)	(479,366)
	D	(974,164)	(974,420)	(27,419)	(28,161)

Two experimental tests have been conducted to validate the performance of the proposed algorithm. In both tests, the shape feature matching method based on the blob analysis method is used to detect the object to be manipulated. The image undergoes several preprocessing and segmentation steps such as colour space conversion, binarization using the otsu method, hole filling, Gaussian filtering and connected component labelling. Then, a workable classification of the image pixels into object and non-object is achieved. The resulting group of white pixels is called binary or blob images. From the reference image, a single white blob in the shape of the object to be manipulated is obtained. Then the current image is taken and blobs are identified. It may contain multiple blobs; it is then separated into individual blobs, each of which is inspected separately. The refined region is subject to measurements and searches for a match with the blob obtained

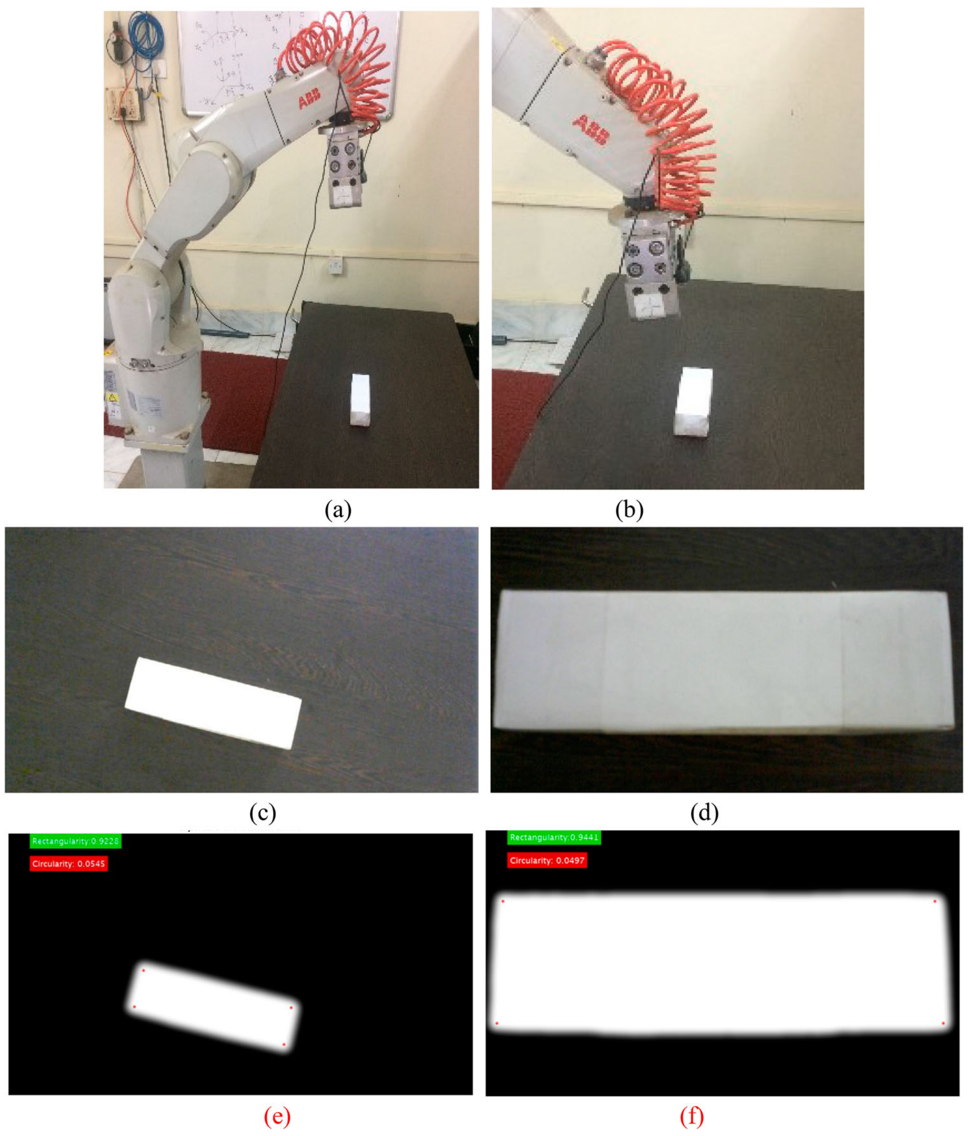

Figure 16. (a) Initial pose and (b) final pose of the robot (c) initial view and (d) final view of the object (e) filtered initial image and (f) filtered final image with detected corners (test 5).

from the desired image. The properties of the blobs in the images are measured using shape descriptors. The essential properties of efficient shape features are its identifiability, scale, rotation and translation invariance, occlusion invariance and noise rejection. The rectangularity and circularity of the blobs are calculated and are compared with that obtained from the reference image. Accordingly, the object identification is done and the corners are detected using the Harris corner detection method. The initial and desired features for test 5 and test 6 are listed in Table 6. Instead of using depth estimation techniques, the depth is directly calculated from the current and desired position of the manipulator. The value of K_s and K_p is set as 0.01 and 0.5 respectively for both experiments. Since straight line trajectory performs better in simulation studies, the experiments are conducted only with straight line.

Test 5: This test is conducted to examine the convergence of image features when the initial joint angle configuration of the robot manipulator is [0 40.7 −64.5 0 108.5 −25.2] degrees. Straight line trajectory with 40 intermediate feature points is designed to track the image features. The target object is kept at a distance of 0.8 m from the initial robot pose. The photographs of the initial and the final robot pose for test 5 and the corresponding images of the target object captured using an end-effector camera are shown in Figure 16. The rectangularity and circularity of the blob detected in the initial image are 0.9228 and 0.0545. Similarly, the rectangularity and the circularity of the reference image are calculated and obtained as 0.9441 and 0.0497. The filtered binary images of initial and final view are shown in Figure 16(e) and Figure 16(f) and the red markings indicate the corner features detected. The proposed controller guides the robot manipulator towards the final pose through the predefined straight line trajectory and the task converged within 57 iterations. The camera mounted on the end-effector is able to track the pixel trajectory as shown in Figure 17(a). The position and orientation of the camera during the task are given in Figure 17(b). The camera/end-effector velocity is smooth without any transients and image feature errors are converged to zero as shown in Figure 17(c). The camera trajectory from the initial pose to the desired one in Cartesian coordinates is nearly straight as shown in Figure 17(d).

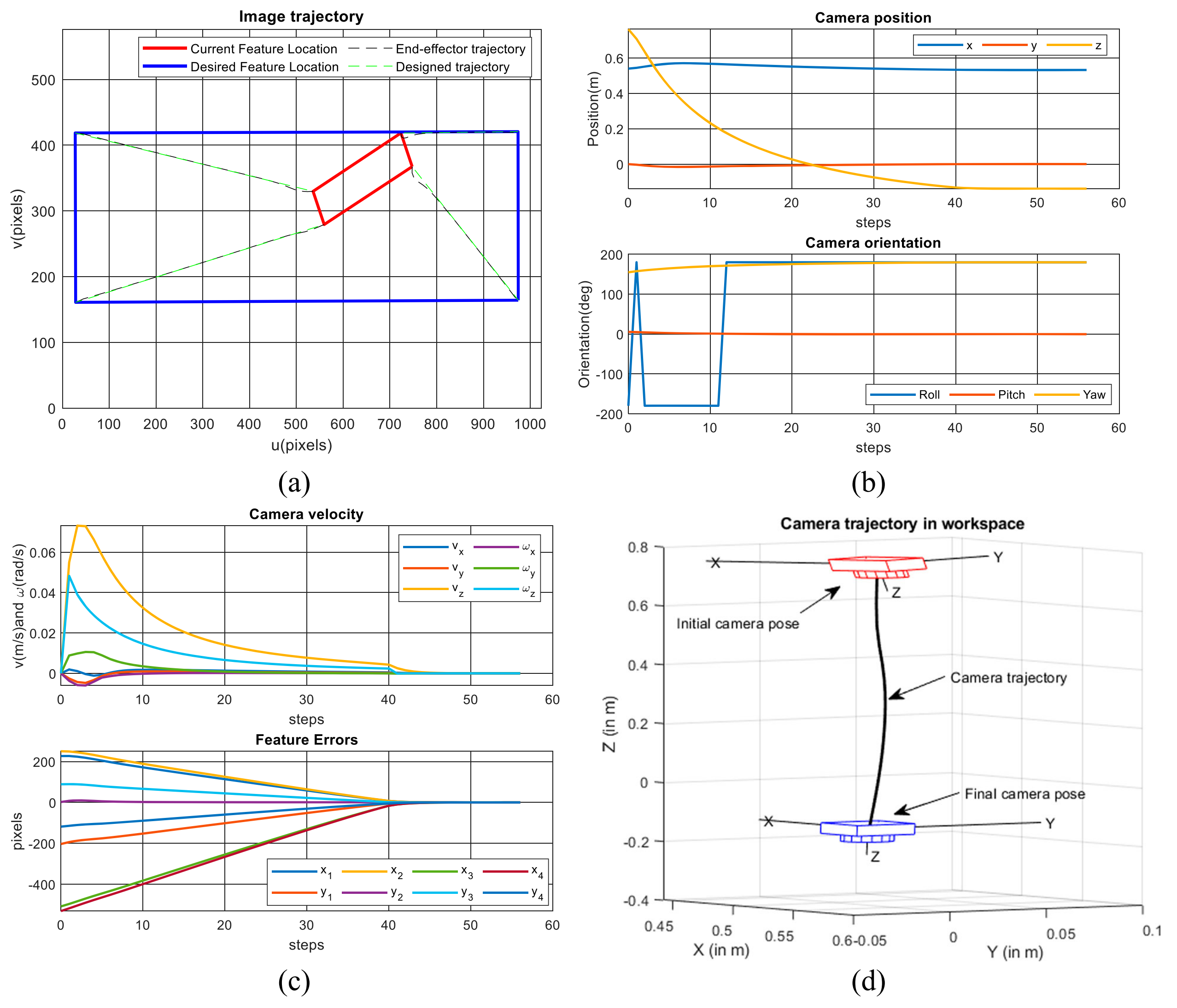

Figure 17. Test 5 results (a) image trajectory (b) camera pose (c) camera velocity and feature errors (d) camera trajectory in Cartesian space.

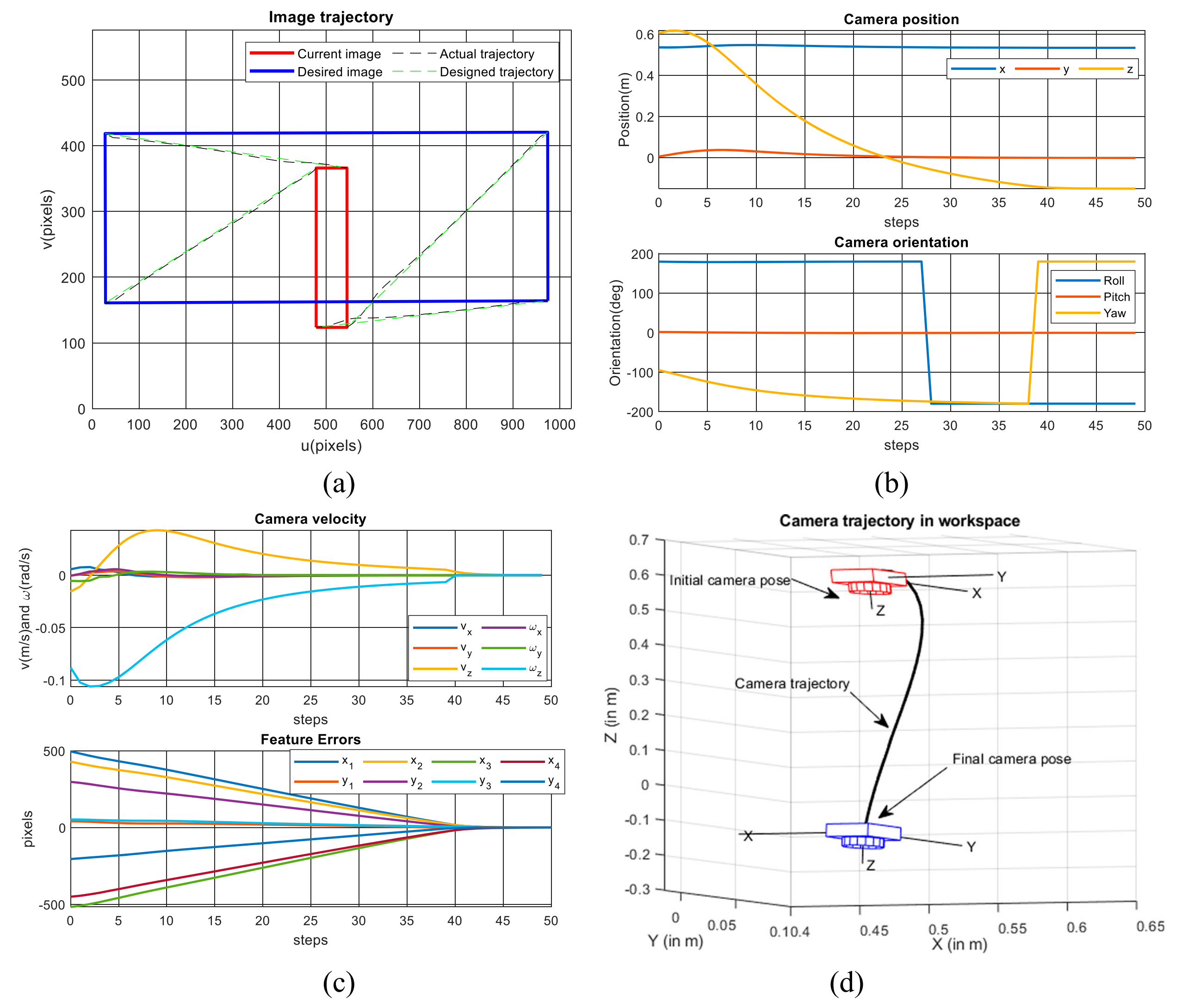

Figure 18. Test 6 results (a) image trajectory (b) camera pose (c) camera velocity and feature errors (d) camera trajectory in Cartesian space.

Test 6: In test 6, the initial pose of the robot in terms of joint angles is given by [0 31.2 −19.9 0 77.2 90] in degrees. The object is at a depth of 0.8 m from the initial camera pose and a significant rotation of the robot manipulator is required to achieve the target. A straight line pixel trajectory is designed for 40 steps and the end-effector is able to track the straight line pixel trajectory as shown in Figure 18(a). The task was completed in 51 iterations. The position and orientation of the camera during the task are given in Figure 18(b). The camera/end-effector velocity profile is smooth and the image feature errors are reduced to zero as shown in Figure 18(c). A smooth and finest camera trajectory is obtained in Cartesian coordinates during the task as shown in Figure 18(d).

6. Conclusion

In this paper, a novel visual servoing controller that incorporates feature-based trajectory planning and tracking in the image plane is proposed. The task is performed by identifying the path to be followed by the robot prior to the task and keeps the features always in the camera field of view. The idea of feature tracking assists large displacement positioning tasks where the proportional IBVS controllers fail. A sliding mode controller is designed to make the system robust against the depth uncertainties and the closed-loop system is proved to be asymptotically stable. The proposed method is validated and compared with the PD-SMC algorithm. The effectiveness of the proposed approach is highlighted in terms of settling time and distance travelled by the robot manipulator. The experimental studies are also conducted with the help of ABB IRB 1200 robot manipulator. This work can be further extended to track moving objects in the workspace. The manipulator dynamics can be incorporated to design dynamic visual feedback control to achieve high-performance control.

Disclosure statement

No potential conflict of interest was reported by the author(s).

ORCID

Megha G. Krishnan http://orcid.org/0000-0001-5549-4261
Ashok Sankar http://orcid.org/0000-0002-1898-0903

References

[1] Hutchinson S, Hager GD, Corke PI. A tutorial on visual servo control. IEEE Trans Robot Autom. 1996;12(5): 651–670.

[2] Hashimoto K. A review on vision-based control of robot manipulators. Adv Robot. 2003;17(10):969–991.

[3] Chaumette F, Hutchinson S. Visual servo control. II. Advanced approaches [tutorial]. IEEE Robot Autom Mag. 2007;14(1):109–118.

[4] Janabi-sharifi F, Deng L, Wilson WJ. Comparison of basic visual servoing methods. IEEE/ASME Trans Mechatronics. 2011;16(5):967–983.

[5] Deng L, Wilson WJ. Stability and robustness of visual servoing methods. In: Proceedings 2002 IEEE International Conference on Robotics and Automation. Washington, DC; May 2002. p. 1604–1609.

[6] Corke P. Robotics, Vision & Control. 2nd ed. Berlin: Springer; 2011.

[7] Wang J, Cho H. Micropeg and hole alignment using image moments based visual servoing method. IEEE Trans Ind Electron. 2008;55(3):1286–1294.

[8] Li G, Wang X. Binocular visual servoing based on PID neural network. 2014 International Joint Conference on Neural Networks (IJCNN), July 6–11, 2014, Beijing, China; 2014. p. 3428–3434.

[9] Keshmiri M, Xie WF. Catching moving objects using a navigation guidance technique in a robotic visual servoing system. 2013 American Control Conference, Washington, DC, USA, June 17–19, 2013; p. 6302–6307.

[10] Keshmiri M, Xie W, Mohebbi A. Augmented image-based visual servoing of a manipulator using acceleration command. IEEE Trans Ind Electron. 2014;61(10): 5444–5452.

[11] Hashimoto K, Ebine T, Kimura H. Visual servoing with hand-eye manipulator-optimal control approach. IEEE Trans Robot Autom. 1996;12(5):766–774.

[12] Allibert G, Courtial E, Chaumette F. Predictive control for constrained image-based visual servoing. IEEE Trans Robot. 2010;26(5):933–939.

[13] Ke F, Li Z, Yang C. Robust tube-based predictive control for visual servoing of Constrained differential-drive mobile robots. IEEE Trans Ind Electron. 2018;65(4): 3437–3446.

[14] Heshmati-Alamdari S, Karras GC, Marantos P, et al. A robust model predictive control approach for autonomous underwater vehicles operating in a constrained workspace. 2018 IEEE International Conference on Robotics and Automation, 2018; p. 1–5.

[15] Keshmiri M, Xie W, Ghasemi A. Visual servoing of a robotic manipulator using an optimized trajectory planning technique. Int J Control Autom Syst. 2014;15(3):1362–1373.

[16] Muñoz-Benavent P, Gracia L, Solanes JE, et al. Sliding mode control for robust and smooth reference tracking in robot visual servoing. Int J Robust Nonlinear Control. 2017;28(5):1728–1756.

[17] Parsapour M, Rayatdoost S, Taghirad HD. Position based sliding mode control for visual servoing system. Proceedings 2013 First RSI/ISM International Conference on Robotics and Mechatronics; 2013 Feb 13–15; Tehran, Iran. p. 337–342.

[18] Corke PI. The machine vision toolbox: a MATLAB toolbox for vision and vision-based control. IEEE Robot Autom Mag 2005;12:16–25.

[19] Zhao YM, Lin Y, Xi F, et al. Switch-based sliding mode control for position-based visual servoing of Robotic riveting system. J Manuf Sci Eng. 2017;139:041010-(1-11).

[20] Kim J, Kim D, Choi S, et al. Image-based visual servoing using sliding mode control. SICE-ICASE International Joint Conference; 2006 Oct 18–21. Bexco, Busan, Korea. p. 4996–5001.

[21] Parsapour M, Taghirad HD. Kernel-based sliding mode control for visual servoing system. IET Comput Vis. 2015;9(3):309–320.

[22] Becerra HM, Sagues C. A sliding mode control law for epipolar visual servoing of differential-drive robots. 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems Convention Centre, Nice, France; 2008 Sep 22–26. p. 3058–3063.

[23] Li S, Ghasemi A, Xie W, et al. An enhanced IBVS controller of a 6-DOF manipulator using hybrid PD-SMC method. Int J Control Autom Syst. 2018;16(2):844–855.

[24] Sederberg TW. Computer aided geometric design. CAGD Course Notes, Brigham Young University, Provo, UT, 84602, April 2007.

[25] Slotine JJE, Li W. Applied nonlinear control. 2nd ed. Englewood Cliffs: Prentice Hall; 1991.

[26] [cited 2019 Aug] Available from: http://new.abb.com/products/robotics/industrial-robots/irb-1200/irb-1200-data

[27] Corke P. Machine vision toolbox. IEEE Robotics and Automation Magazine. Nov. 2005;12(4):16–25.

[28] Vijayan AT, Sankar A, Sudheer AP. A comparative study on the performance of neural networks in visual guidance and feedback applications. Automatika. 2017;58(3):336–346. doi:10.1080/00051144.2018.1437-683.

[29] Krishnan MG, Vijayan AT, Ashok S. Interfacing an industrial robot and MATLAB for predictive visual servoing. Industrial Robot: The International Journal of Robotics Research and Applications. 2021;48(1):110–120. doi:10.1108/IR-05-2020-0100.

Appendix

The derivation for the uncertainty matrices$\Delta_{\min}$ and $\Delta_{\max}$ in terms of the depth bound $\hat{Z}_{\min}$ and $\hat{Z}_{\max}$ is illustrated below. During the translational motion control, the system is affected by the uncertainties in depth parameter and the image Jacobian corresponds to linear velocity is given by

$$J_v = \frac{1}{Z}\begin{pmatrix} -1 & 0 & x \\ 0 & -1 & y \end{pmatrix} \tag{A1}$$

The estimated Jacobian matrix is given by

$$\hat{J}_v = \frac{1}{\hat{Z}}\begin{pmatrix} -1 & 0 & x \\ 0 & -1 & y \end{pmatrix} \tag{A2}$$

The Moore-Pseudo inverse of the estimated Jacobian matrix is obtained as,

$$\hat{J}_v^{\dagger} = \hat{J}_v^T(\hat{J}_v\hat{J}_v^T)^{-1} \tag{A3}$$

$$\hat{J}_v^{\dagger} = \frac{\hat{Z}}{Z(1+x^2+y^2)}\begin{pmatrix} -(1+y^2) & xy \\ xy & -(1+x^2) \\ x & y \end{pmatrix}$$

Multiplying (A1) and (A3),

$$J_v \hat{J}_v^+ = \frac{\hat{Z}}{Z} I_{2x2} \tag{A4}$$

From (22), the uncertainty bounds is obtained as

$$I + \Delta_{\min} \leq \frac{\hat{Z}}{Z} I \leq I + \Delta_{\max} \tag{A5}$$

$$\Delta_{\min} \leq \frac{(\hat{Z} - Z)}{Z} I \leq \Delta_{\max} \tag{A6}$$

where the uncertainty matrices are given by

$$\Delta_{\min} = \frac{(Z - \hat{Z}_{\min})}{Z} I$$

$$\Delta_{\max} = \frac{(\hat{Z}_{\max} - Z)}{Z} I \tag{A7}$$