Robotics and Autonomous Systems 58 (2010) 978–990

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

Visual servoing of redundant manipulator with Jacobian matrix estimation using self-organizing map

P. Prem Kumar, Laxmidhar Behera *

Department of Electrical Engineering, Indian Institute of Technology, Kanpur 208 016, India

ARTICLE INFO

Article history:
Received 21 August 2008
Received in revised form
8 March 2010
Accepted 1 April 2010
Available online 13 April 2010

Keywords:
Redundant manipulator
Visual servoing
Inverse Jacobian
Self-organizing map
Kinematic control
Redundancy resolution

ABSTRACT

Vision based redundant manipulator control with a neural network based learning strategy is discussed in this paper. The manipulator is visually controlled with stereo vision in an eye-to-hand configuration. A novel Kohonen's self-organizing map (KSOM) based visual servoing scheme has been proposed for a redundant manipulator with 7 degrees of freedom (DOF). The inverse kinematic relationship of the manipulator is learned using a Kohonen's self-organizing map. This learned map is shown to be an approximate estimate of the inverse Jacobian, which can then be used in conjunction with the proportional controller to achieve closed loop servoing in real-time. It is shown through Lyapunov stability analysis that the proposed learning based servoing scheme ensures global stability. A generalized weight update law is proposed for KSOM based inverse kinematic control, to resolve the redundancy during the learning phase. Unlike the existing visual servoing schemes, the proposed KSOM based scheme eliminates the computation of the pseudo-inverse of the Jacobian matrix in real-time. This makes the proposed algorithm computationally more efficient. The proposed scheme has been implemented on a 7 DOF PowerCube™ robot manipulator with visual feedback from two cameras.

1. Introduction

Vision has been employed in robotic research owing to its flexibility and accuracy [1]. Kinematic redundancy is introduced in manipulators for handling complex tasks in changing environments. Vision based redundant manipulator control is an attractive research area due to its wide range of operation and control challenges in dynamic environments.

Vision based Manipulator Control generally uses one or two cameras, where the camera is either mounted at the end-effector [2] or fixed in the work space [3]. The former is, often called the eye-in-hand [4] configuration, and the latter the eye-to-hand configuration [5].

Typically, vision based manipulator control is executed in open loop fashion, "looking" and then "moving" [6]. This results in poor positioning accuracy due to the model inaccuracies. An alternative approach is to use a visual control loop which is generally referred to as visual servoing. A detailed survey on visual servoing can be found in [7–9]. Visual servoing schemes employ a local controller [10] with an image Jacobian computed at a specified operating point since it requires the knowledge of 3-D Cartesian parameters. In addition, the computation of the pseudo-inverse at each instant is also costly. The learning based servoing scheme discussed for a non-redundant manipulator [11] also focusses mainly on local solution.

In general, visual servoing computes the end-effector screw required to achieve desired manipulation from the image features. It basically assumes that there exists a non-redundant manipulator which can generate the desired end-effector screw with its own inverse kinematic algorithm. In a dynamic environment, the end-effector cannot be positioned at the desired location due to kinematic constraints and obstacles. This necessitates the use of redundant manipulators for vision based control in dynamic environment. The excess degrees of freedom (DOF) can be effectively utilized in performing additional tasks such as joint-limit avoidance, obstacle avoidance and optimization of a specific task-based criteria. Surveys of the existing redundancy resolution techniques are discussed in [12,13].

Vision based control of redundant manipulators has not been discussed much in literature. Most of the existing literature addresses visual servoing and redundancy resolution as two separate problems. Chaumette [14] proposed an iterative approach for avoiding joint limits using redundancy and then applied the method for visual servoing. In such a case, the model inaccuracies would result in inefficient end-effector screw computation, which may result in undesired joint velocities. On the other hand, KSOM based learning schemes compute joint angles directly from the vision space but mostly tested on non-manipulators. It has been shown through experimentation in [15] that KSOM learns a smooth map for redundant manipulators, and yet a detailed

* Corresponding author. Tel.: +91 512 2597198; fax: +91 512 2590063.
E-mail addresses: premkani@iitk.ac.in (P. Prem Kumar), lbehera@iitk.ac.in (L. Behera).

0921-8890/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.robot.2010.04.001

analysis about the type of solution has not been studied. The experimental results of [16] show that the redundant manipulator is tested for end-effector collision avoidance only and would suffer from positioning inaccuracy due to the open loop mode of operation. Recently, Kumar et al. [17], proposed a KSOM network with joint space sub-clustering which allows the learning of multiple solutions for each end-effector position. The network acts as a look-up table for redundant solutions and works based on the principle of "look and move". A detailed survey of KSOM based inverse kinematic control is discussed in [18]. In a nutshell, vision based redundant manipulator control has not been discussed for a global workspace.

This motivated us to develop a holistic learning approach to resolve the redundancy directly from the image space. The proposed scheme focuses on an eye-to-hand camera configuration where two cameras are fixed overlooking the 7DOF redundant manipulator workspace. The inverse kinematic relationship learned using KSOM is used as an approximation for the inverse Jacobian and then closed loop stabilization is achieved. KSOM has been widely used for inverse kinematics control [18]. But all the existing methods work in an open loop mode and focus mostly on non-redundant manipulators. On the contrary, this paper discusses KSOM based visual servoing for redundant manipulators.

A KSOM based neural network is used to learn the inverse kinematics of the redundant manipulator offline. The input space to the network is a 4-dimensional image co-ordinate space viewed from two cameras, while the output is a 7 dimensional joint angle vector. Each neuron in KSOM approximates the inverse kinematics relation from the image space to the joint angle space over a local zone. The output of the KSOM neuron lattice consists of a joint angle configuration required to reach close to the corresponding desired image space position, and a local first order approximation of the inverse Jacobian. In this work, we are mainly interested in the learned local linear model of the inverse Jacobian. It is shown experimentally that KSOM approximates the pseudo-inverse of Jacobian matrix. This observation motivated us to use the learned KSOM for closed loop visual servoing. Classical proportional feedback [19] is chosen for closed loop control. Further experiments revealed that a globally stabilizing controller can be obtained by using conventional proportional feedback in conjunction with the inverse kinematic map learned using KSOM. Since the approximate inverse kinematic relationship is learned offline over the entire workspace, a simple proportional controller results in global stability. Lyapunov analysis shows that global stability can be achieved if the learned map accurately approximates the local inverse Jacobian. The obtained inverse Jacobian approximation also eliminates the necessity of online pseudo-inverse computation in visual servoing and makes the proposed scheme computationally efficient. With the empirical observations of convergence, the KSOM based kinematic control is further extended for redundancy resolution. The proposed scheme has been tested on 7 DOF PowerCube™ manipulator with 2 Fire-i™ cameras overlooking its workspace.

The remaining portion of this paper is organized as follows. The following section discusses the system configuration for vision based manipulator control and the experimental setup. The Sections 3 and 4 discusses KSOM based inverse kinematic control and visual servoing respectively. The problem is defined in Section 5 and the proposed control strategy is presented in Section 6. The simulations and experimental results in vision space are shown in Section 7 followed by conclusions in Section 8.

2. System and task

The schematic of visual servo control is shown in Fig. 1. It consists of a stereo-camera system, a robot manipulator and a programmable computer. Image processing as well as neural network algorithms are executed on the computer. The joint velocity vector $\dot{\boldsymbol{\theta}}$, obtained from the algorithm, is provided to the servo unit which drives the robot manipulator. The image processing unit is used to extract 4-dimensional image co-ordinate vectors for the current robot position ($\mathbf{u}_c$) and the target point ($\mathbf{u}_t$) to be reached in the workspace. (u_1, u_2) and (u_3, u_4) are 2-dimensional pixel co-ordinates obtained from each camera.

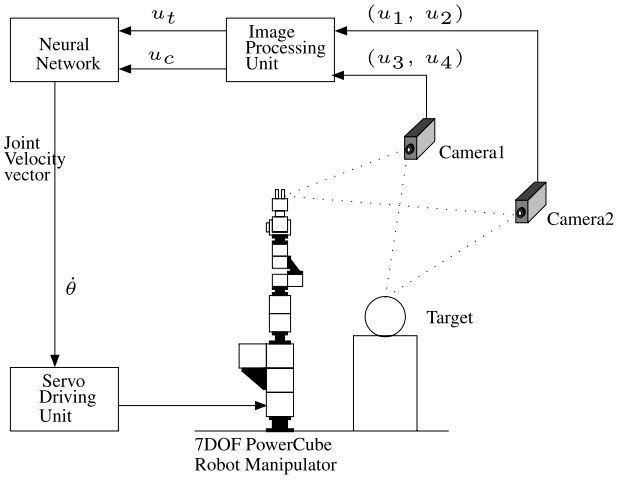


Fig. 1. Schematic of visual servo control (i) u_1, u_2, u_3, u_4: Camera co-ordinates of the objects (end-effector and target) seen through the vision system (ii) $\mathbf{u}_t$: Target position (iii) $\mathbf{u}_c$: Current position of end-effector.

2.1. Experimental setup

We use a 7DOF PowerCube™ robot manipulator [20] for experimentation. The experimental setup is show in Fig. 2. The D-H parameters are used to derive forward kinematic equations for the manipulator. The end-effector position is obtained using D-H parameters as follows:

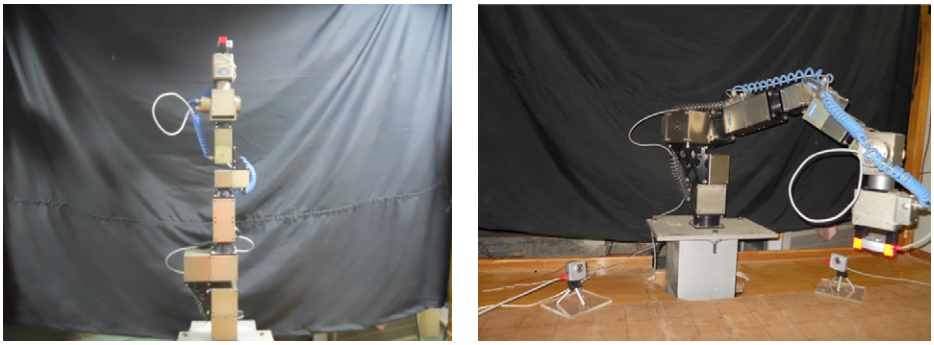
$$\begin{aligned}
x &= (-(((c_1c_2c_3 - s_1s_3)c_4 - c_1s_2s_4)c_5 + (-c_1c_2s_3 - s_1c_3)s_5)s_6 \\
&\quad + (-(c_1c_2c_3 - s_1s_3)s_4 - c_1s_2c_4)c_6)d_7 \\
&\quad + (-(c_1c_2c_3 - s_1s_3)s_4 - c_1s_2c_4)d_5 - c_1s_2d_3 \\
y &= (-(((s_1c_2c_3 + c_1s_3)c_4 - s_1s_2s_4)c_5 + (-s_1c_2s_3 + c_1c_3)s_5)s_6 \\
&\quad + (-(s_1c_2c_3 + c_1s_3)s_4 - s_1s_2c_4)c_6)d_7 \\
&\quad + (-(s_1c_2c_3 + c_1s_3)s_4 - s_1s_2c_4)d_5 - s_1s_2d_3 \\
z &= (-((s_2c_3c_4 + c_2s_4)c_5 - s_2s_3s_5)s_6 + (-s_2c_3s_4 + c_2c_4)c_6)d_7 \\
&\quad + (-s_2c_3s_4 + c_2c_4)d_5 + c_2d_3 + d_1 \qquad (1)
\end{aligned}$$

where various parameters are: $d_1 = 0.390$ m, $d_3 = 0.370$ m, $d_5 = 0.310$ m, $d_7 = 0.2656$ m, $c_i = \cos\theta_i$, $s_i = \sin\theta_i$, $i = 1, 2, \ldots, 6$. It should be noted that the end-effector position $[x\ y\ z]^T$ does not depend on the seventh joint angle since it gives rolling motion in the actual setup. In this work, the seventh joint angle is considered as a fixed link for visual servoing. Since we are interested in end-effector positioning only, the setup will have three excess degrees of freedom with a fixed seventh link.

The kinematic limits are tabulated in Table 1.

The tip of the end-effector is viewed through a stereo-camera system. We use two Fire-i™ digital cameras [21] for visualization. The workspace is captured through the cameras with an image frame of dimension 320 × 240 pixels. In real-time, the target and the robot tips are identified with LEDs. The regions of interest are extracted using thresholding and filtering operations. The centroid of the identified target and the end-effector are used by the visual servo controller to compute the necessary joint velocities.

The Cartesian co-ordinates of end-effector position are converted into the corresponding pixel co-ordinates in the image



(a) PowerCube™. (b)Setup.

Fig. 2. (a) PowerCube™ manipulator in upright pose in IIT-Kanpur. (b) Visual servo control setup with manipulator and two cameras.

Table 1
Kinematic limits of manipulator.

Joint angle	Joint velocity (rad/s)
$-160^\circ \le \theta_1 \le 160^\circ$	$2.618 \le \dot{\theta}_1 \le 1.7e{-}5$
$-95^\circ \le \theta_2 \le 95^\circ$	$2.618 \le \dot{\theta}_2 \le 1.7e{-}5$
$-160^\circ \le \theta_3 \le 160^\circ$	$2.618 \le \dot{\theta}_3 \le 1.7e{-}5$
$-50^\circ \le \theta_4 \le 120^\circ$	$2.618 \le \dot{\theta}_4 \le 1.7e{-}5$
$-90^\circ \le \theta_5 \le 90^\circ$	$2.618 \le \dot{\theta}_5 \le 1.7e{-}5$
$-120^\circ \le \theta_6 \le 120^\circ$	$4.189 \le \dot{\theta}_6 \le 1.7e{-}5$
$-360^\circ \le \theta_7 \le 360^\circ$	$6.283 \le \dot{\theta}_7 \le 1.7e{-}5$

plane during the learning phase. This necessitates a camera model and Tsai algorithm [22] is used to calibrate the camera. The inverse kinematic relationship is learned offline using manipulator forward kinematics (1) and camera model obtained from the Tsai algorithm.

3. Kinematic control using KSOM

The forward map from 7-dimensional joint angle space to 4-dimensional image co-ordinate space can be derived using the manipulator forward kinematic model (1) and camera model obtained through the Tsai algorithm. This forward mapping may be represented as

$$\mathbf{u}_t = \mathbf{f}(\boldsymbol{\theta}) \tag{2}$$

where $\mathbf{u}_t$ is the 4-dimensional image co-ordinate vector of the target position obtained from the two cameras and $\boldsymbol{\theta}$ is the 7-dimensional joint angle vector to reach $\mathbf{u}_t$. We are interested in knowing the inverse relationship since it is required to know the joint angle required to reach the target point. The inverse kinematic relationship is given by

$$\boldsymbol{\theta} = \mathbf{f}^{-1}(\mathbf{u}_t) = \mathbf{r}(\mathbf{u}_t). \tag{3}$$

The inverse kinematic problem is ill-posed and may have an infinite number of solutions. In KSOM based visual-motor co-ordination, the inverse kinematic relationship is learned and this learned map is used to reach any target position in the workspace. A brief discussion of the KSOM architecture is presented in the following subsection to aid understanding.

3.1. KSOM architecture

The inverse kinematic relationship of a redundant manipulator (3) is a one-to-many mapping and hence difficult to learn. One easier approach to this problem involves discretizing the input as well as the output space into several small cells so that a linear map from input to output space holds good within each cell. KSOM discretizes the input image space into a number of cells and associates a vector and a linear map in the output joint space for each region.

In the current work, a 3-dimensional KSOM lattice is used to discretize the input and output spaces. Lattice node indices are represented by γ, and each such node is associated with an image space vector $\mathbf{w}_\gamma$, a joint angle vector $\boldsymbol{\theta}_\gamma$ and a linear map $\mathbf{A}_\gamma$. The vector $\mathbf{w}_\gamma$ and $\boldsymbol{\theta}_\gamma$ discretizes the input and output space respectively. $\mathbf{A}_\gamma$ approximates the inverse Jacobian in each region with a linear map. The joint angle vector to reach any target position is computed using KSOM as follows:

Given a target position $\mathbf{u}_t$, a winner neuron μ is selected based on its Euclidean distance metric in the input space. The neuron whose weight vector is closest to the target is declared winner as shown below.

$$\mu = \min_\gamma \|\mathbf{u}_t - \mathbf{w}_\gamma\|_2. \tag{4}$$

The arm is given a coarse movement $\boldsymbol{\theta}_0^{\text{out}}$ given by

$$\boldsymbol{\theta}_0^{\text{out}} = s^{-1} \sum_\gamma h_\gamma (\boldsymbol{\theta}_\gamma + \mathbf{A}_\gamma (\mathbf{u}_t - \mathbf{w}_\gamma)) \tag{5}$$

where $s = \sum_\gamma h_\gamma$ and $h_\gamma = \mathrm{e}^{\left(\frac{\|\mu-\gamma\|}{2\sigma^2}\right)}$. Because of this coarse movement, the end-effector reaches a position $\mathbf{v}_0$ in image space. A correcting fine movement $\boldsymbol{\theta}_1^{\text{out}}$ is evaluated as follows:

$$\boldsymbol{\theta}_1^{\text{out}} = \boldsymbol{\theta}_0^{\text{out}} + s^{-1} \sum_\gamma h_\gamma \mathbf{A}_\gamma (\mathbf{u}_t - \mathbf{v}_0). \tag{6}$$

This corrective movement results in a final movement of the end-effector to $\mathbf{v}_1$. Although one can use several such corrective movements to increase the accuracy of tracking, usually one corrective movement is used.

3.2. Comments

The above approach has been used for visual motor co-ordination of non-redundant [6] as well as redundant manipulators [15,23]. While the application of KSOM to inverse kinematics of non-redundant manipulators has been analyzed extensively, it has not been applied to redundant manipulators, as redundancy will be lost in the learning phase. [15] demonstrates that, in case of manipulators with higher degrees of freedom, the above algorithm is capable of resolving redundancy by minimizing the variations of joint angles. Recently, Han et al. [16] used KSOM to avoid obstacles with a multiple camera setup for a 4-DOF manipulator, which involves computation of the pseudo-inverse during learning phase.

The learned map is generally used in open loop mode which suffers from positioning inaccuracy. The KSOM algorithm considers only the target position, and the current end-effector position is ignored during coarse movement. Hence the path

traversed during coarse movement from the current position to the desired position is not controlled. Since the manipulator is controlled with a joint angle reference, it is difficult to resolve redundancy for different subtasks.

4. Visual servoing

As discussed, joint angle control with KSOM suffers from positioning accuracy. A popular alternative approach is to control the manipulator in a closed loop with joint velocities.

4.1. Visual servoing and redundancy resolution

The forward kinematic relationship between the joint angle motion $\dot{\boldsymbol{\theta}}$ and end-effector motion $\dot{\mathbf{x}}$, in Cartesian space, is represented as

$$\dot{\mathbf{x}} = \mathbf{J}\dot{\boldsymbol{\theta}}. \tag{7}$$

The end-effector motion in Cartesian space results in a corresponding image feature motion $\dot{\mathbf{u}}$. The interaction matrix, $\mathbf{L}$, represents the relationship between end-effector motion and image feature motion as

$$\dot{\mathbf{u}} = \mathbf{L}\dot{\mathbf{x}}. \tag{8}$$

The control task is to compute the necessary velocity motion such that the end-effector will reach the desired position in image space asymptotically. In case of a non-redundant manipulator, it is sufficient to compute the required end-effector kinematic screw from the image features, since there exists a unique relationship from Cartesian space to the joint space.

The simple proportional control law, which results in asymptotic stabilization, is

$$\dot{\mathbf{x}}_d = K_p.\mathbf{L}^{+}.(\mathbf{u}_t - \mathbf{u}_c), \tag{9}$$

where K_p is a proportional gain and $\mathbf{L}^{+}$ is Moore–Penrose pseudo-inverse of $\mathbf{L}$. Hereafter, the notation $(.)^{+}$ will be used to indicate the Moore–Penrose pseudo-inverse of $(.)$. The above controller requires a tedious pseudo-inverse computation although it ensures global stability. To reduce the computational complexity, the pseudo-inverse of the interaction matrix is evaluated at a desired image feature and then the controller is implemented. This eliminates the continuous computation of the pseudo-inverse. But, this results in a locally stabilizing controller since the sufficient positivity condition of stability is valid in a local region only [10].

In the case of redundant manipulators, theoretically an infinite joint space velocity $\dot{\boldsymbol{\theta}}$ exists for a given end-effector screw. The different joint velocities which could generate the given end-effector screw $\dot{\mathbf{x}}$ are given by the relationship

$$\dot{\boldsymbol{\theta}} = \mathbf{J}^{+}\dot{\mathbf{x}} + (\mathbf{I} - \mathbf{J}^{+}\mathbf{J})\mathbf{K}, \tag{10}$$

where $\mathbf{J}$ is the forward Jacobian. Redundancy resolution computes the joint velocity required to achieve the desired end-effector screw using (10). The first portion of the above equation, i.e. $\mathbf{J}^{+}\dot{\mathbf{x}}$, results in a lazy arm like movement of the manipulator. While the latter corresponds to the null space of a Jacobian which results in the 'self-motion' of the redundant manipulator. This indicates the portion of joint velocity which would not result in any end-effector motion. This self-motion can be exploited to perform additional tasks, such as satisfying kinematic constraints, optimal trajectory and obstacle avoidance [24,14]. A detailed survey of various redundancy resolution techniques for Cartesian space are discussed in [12].

4.2. Comments

The vision based kinematic control of the redundant manipulator is performed in two stages, as discussed in the previous section. Almost all the existing methods are model dependant and require a complex pseudo-inverse computation. Model-dependant approaches become computationally tedious as the number of links increases. Most of the existing redundancy resolution schemes are offline methods and resolve the redundancy for trajectories defined in Cartesian space. Since the servoing loop is outside the loop of redundancy control, the model inaccuracies may result in unacceptable joint trajectories. The control task can be simplified by directly computing the joint angles from image space.

The relationship between image feature velocity and joint angle velocity can be obtained by combining (7) and (8) as

$$\dot{\mathbf{u}} = \mathbf{L}\mathbf{J}\dot{\boldsymbol{\theta}}. \tag{11}$$

The proportional controller resulting in local asymptotic stabilization is given as

$$\dot{\boldsymbol{\theta}} = K_p(\mathbf{L}\mathbf{J})^{+}_{u_t}.(\mathbf{u}_t - \mathbf{u}_c). \tag{12}$$

Since the pseudo-inverse of the Jacobian is used in the control law, the controller would result in "lazy-arm movement". The above controller would result in local asymptotic stabilization. Global stabilization can be achieved by computing the pseudo-inverse at each operating point.

5. Problem definition

As discussed in previous sections, existing visual servoing techniques are model-dependant and computationally intensive. Though model-free strategies are analyzed for position level control, they are inaccurate and not suitable for redundancy resolution. Considering these challenges in visual servoing of redundant manipulators, the problem is formulated as follows:

"Given a redundant manipulator with stereo vision overlooking the workspace in an eye-to-hand configuration, develop a model-free visual servoing technique. With any initial robot configuration $\boldsymbol{\theta}_0$ resulting in an end-effector position $\mathbf{u}_c$ and the desired end-effector position $\mathbf{u}_t$ in image space, identify the control law $\dot{\boldsymbol{\theta}} = \mathbf{f}(\boldsymbol{\theta}, \mathbf{e})$, where $\mathbf{e} = \mathbf{u}_t - \mathbf{u}_c$, such that the robot end-effector asymptotically reaches the desired position from any initial position."

In this paper, we are interested in global positioning of the end-effector through visual servoing. The following are the prime issues addressed in this paper:

- A computationally less intensive model-free architecture for visual servoing.
- Global positioning of the redundant manipulator without using orientation information.

6. Proposed strategy

As discussed in Section 3, KSOM learns the linear approximation of the inverse Jacobian in each region. In case of redundant manipulators, it is shown through simulation in [15] that KSOM resolves redundancy by learning a smooth movement in the workspace. In this paper, we analyze the solution learned through KSOM using eigenvalues and verify experimentally that the pseudo-inverse of the Jacobian matrix is learnt locally. In the following sections, we demonstrate our experiments and propose the KSOM model as an approximation of the inverse Jacobian. The learned model is then used for closed loop visual servoing.

6.1. KSOM as approximate inverse Jacobian

The correcting fine movement (6) can be rewritten as

$$\boldsymbol{\theta}_1^{\text{out}} - \boldsymbol{\theta}_0^{\text{out}} = s^{-1} \sum_{\gamma} h_\gamma \mathbf{A}_\gamma (\mathbf{u}_t - \mathbf{v}_0)$$

$$\Delta\boldsymbol{\theta}^{\text{out}} = s^{-1} \sum_{\gamma} h_\gamma \mathbf{A}_\gamma (\Delta\mathbf{u}) \tag{13}$$

where $\Delta\boldsymbol{\theta}^{\text{out}}$ represents the change in joint angles required to generate the end-effector position change of $\Delta\mathbf{u}$ in image space. The above equation can be represented in velocity form by actuating a joint velocity, $\dot{\boldsymbol{\theta}}^{\text{out}}$ for a duration Δt as follows

$$\frac{\Delta\boldsymbol{\theta}^{\text{out}}}{\Delta t} = s^{-1} \sum_{\gamma} h_\gamma \mathbf{A}_\gamma \left(\frac{\Delta\mathbf{u}}{\Delta t}\right)$$

$$\dot{\boldsymbol{\theta}}^{\text{out}} = s^{-1} \sum_{\gamma} h_\gamma \mathbf{A}_\gamma \dot{\mathbf{u}}. \tag{14}$$

By comparing (12) and (14), we can infer that KSOM approximates the inverse of Jacobian as

$$(\mathbf{LJ})^+ \simeq s^{-1} \sum_{\gamma} h_\gamma \mathbf{A}_\gamma. \tag{15}$$

We use this learned map as an approximate inverse Jacobian for visual servoing. With the learned KSOM based approximation of the inverse Jacobian, we can perform image based visual servoing from image space to joint angle space directly. The KSOM based visual servoing simplifies the following issues:

- With the learned KSOM map, the approximate pseudo-inverse from the image space to the joint space is known over the entire space. This eliminates the computation of the pseudo-inverse during servoing. It is known that only the winning neuron contributes to the learned map after the learning phase. Hence, the computation cost of real-time servoing reduces to a simple matrix multiplication.
- Since KSOM learns a unique relationship between image space and joint angle space, it resolves the redundancy in the learning phase itself. This facilitates analyzing the visual servoing and redundancy resolution in a simple integrated framework with direct computation of joint trajectories from image space.

To verify our proposition, empirical experiments are performed. For simplicity, the simulations are performed for an inverse kinematic relation from Cartesian space to the joint space. The same experiments can also be extended to the image space which also requires the computation of the image Jacobian at every point in the visible workspace.

6.2. Verification using simulation

If KSOM approximates the pseudo-inverse, the following relationships are valid.

- Around non-singular points

$$\mathbf{J} \sum_{\gamma} h_\gamma \mathbf{A}_\gamma \approx \mathbf{I}. \tag{16}$$

- Around singular points

$$\mathbf{J} \sum_{\gamma} h_\gamma \mathbf{A}_\gamma \approx \mathbf{I}' \tag{17}$$

where $\mathbf{J}$ is the Jacobian and $\sum_\gamma h_\gamma \mathbf{A}_\gamma$ is the linear approximation of the inverse Jacobian learned by KSOM. $\mathbf{I}$ is the identity matrix of order n and $\mathbf{I}'$ is a positive definite matrix of order n. The $\mathbf{I}'$ matrix of rank r will have $n - r$ eigenvalues as 0. Considering these relationships, the following simulations are performed to check whether the above properties are satisfied with KSOM.

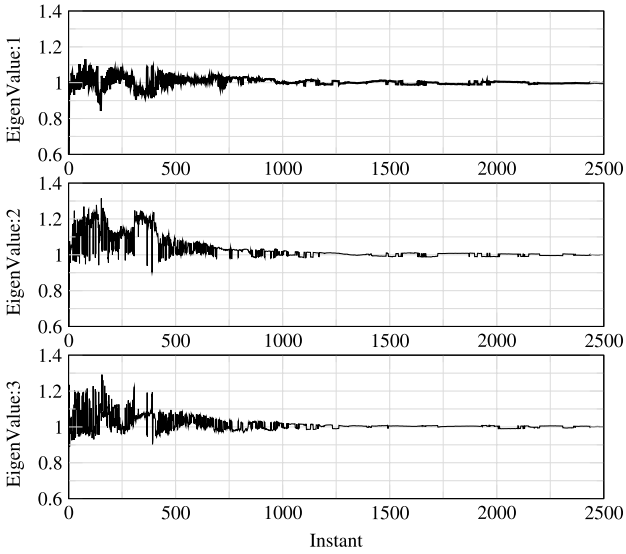


Fig. 3. Evolution of the eigenvalues.

All simulations and experiments are performed with the PowerCube™ 7 DOF manipulator discussed in Section 2.1. The parameters are taken the same as the actual setup in simulation so that it matches with the experimental result.

A 3-dimensional neural lattice with $7 \times 7 \times 7$ neurons is selected to learn the inverse kinematics. The inverse kinematic relation from Cartesian space to joint space is learned with 500,000 training patterns. The input to the KSOM network is the 3 dimensional Cartesian position of the end-effector and the output is the 6 dimensional joint angle co-ordinates.

6.2.1. Inverse Jacobian evolution in learning phase

It is observed from the inverse kinematic solutions that KSOM learns a smooth motion and the mapping is better with a greater number of patterns. It is easier to infer then, that, as the learning progresses, KSOM approaches the pseudo-inverse. To validate this assumption, a typical neuron is selected and $\mathbf{I}' = \mathbf{J}\sum_\gamma h_\gamma \mathbf{A}_\gamma$ is computed in regular intervals of learning. If KSOM learns the Moore–Penrose pseudo-inverse, then the eigenvalues of $\mathbf{I}'$ would converge to 1.

The neuron located at (4, 4, 4) of the neuron lattice is considered to check the eigenvalue evolution in the learning phase. The eigenvalues are computed at regular interval of 200 data points.

The simulation results are shown in Fig. 3. It is clear from the figure that the eigenvalues approach 1 with learning, which confirms our assumption.

6.2.2. Inverse Jacobian at centers in testing phase

The inverse Jacobian relationship at every node of the KSOM network is checked after learning. The results are shown in Fig. 4. It is clear from the figure that KSOM approximates the pseudo-inverse in most of the centers of the networks and in some of the nodes the eigenvalue has not yet converged to 1 which belongs to the neurons located at the corner of the lattice. To conclude further, the positioning accuracy at each center is checked and the result is shown in Fig. 5, which clearly shows that learning is not accurate at the corresponding centers where eigenvalues have not yet converged to 1. Hence the eigenvalues may converge to 1 if we extend the learning further.

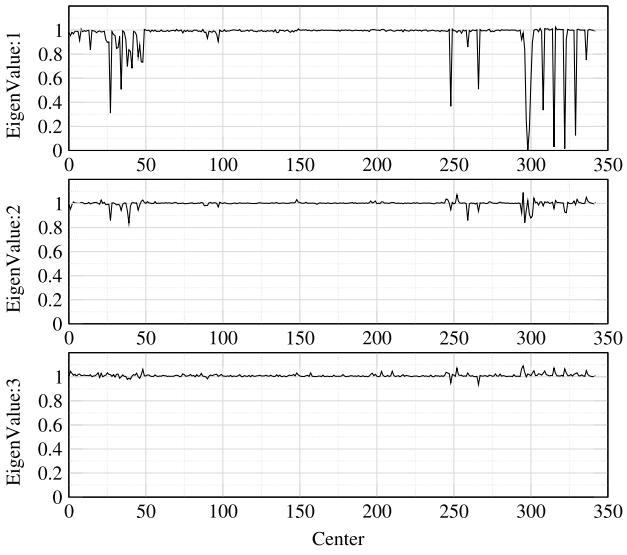


Fig. 4. Eigenvalues at centers of KSOM.

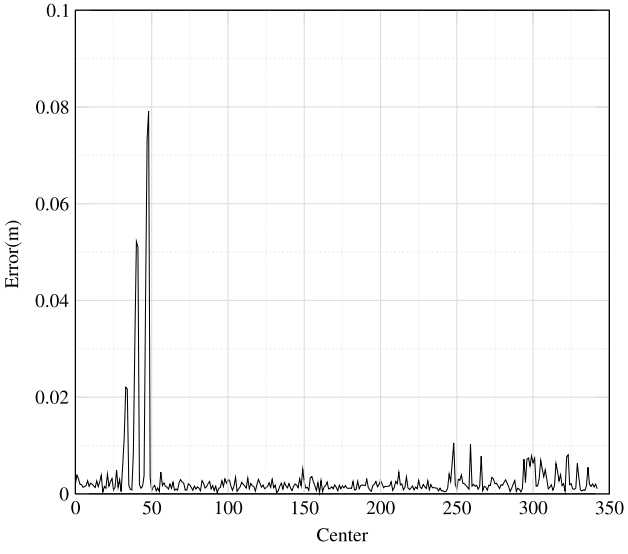


Fig. 5. Positioning error at centers of KSOM.

6.2.3. Inference

It is clear from the above two experiments that KSOM approximates the pseudo-inverse of the Jacobian and the approximation improves with learning. Though the above experiments are performed from the Cartesian space to the joint space for simplicity, it can be extended to image space too.

Our claim is that the KSOM learns the pseudo-inverse of the kinematic relationship, and this is corroborated with the empirical results. KSOM reaches the pseudo-inverse, since the linear approximation of the inverse Jacobian is learned by minimizing

$$\min \|\Delta\boldsymbol{\theta}_{01} - \mathbf{A}_i\mathbf{v}_{01}\| \tag{18}$$

where $\Delta\boldsymbol{\theta}_{01} = \boldsymbol{\theta}_1 - \boldsymbol{\theta}_0$ and $\Delta\mathbf{v}_{01} = \mathbf{v}_1 - \mathbf{v}_0$. As learning progresses, $\mathbf{A}_i$ is updated such that $\Delta\boldsymbol{\theta}_{01} = \mathbf{A}_i\mathbf{v}_{01}$, which is equivalent to

$$\dot{\boldsymbol{\theta}} = (\mathbf{LJ})^{+}\dot{\mathbf{x}}. \tag{19}$$

Hence KSOM approximates the inverse relationship which minimizes the cost function

$$\|\dot{\boldsymbol{\theta}}^T\dot{\boldsymbol{\theta}}\|. \tag{20}$$

With these observations, the learned KSOM is shown to be an approximation of the inverse Jacobian from the image space to the joint angle space. This inverse Jacobian is required in visual servoing control algorithm (12).

6.3. KSOM in closed loop visual servoing

The conventional proportional controller with a pseudo-inverse computation at the target location ensures local asymptotic stability only. Global asymptotic stability can be achieved by estimating the forward Jacobian at each instant and then computing the pseudo-inverse of the forward Jacobian. As discussed in the previous section, KSOM approximates the inverse kinematic Jacobian from the image space to the joint space at discrete operating points. Hence, the KSOM based learning approach is a holistic methodology to learn the inverse kinematic relationship over the entire workspace.

In our approach, we exploit this approximation to achieve global stabilization with a conventional proportional controller. KSOM eliminates the computation of the pseudo-inverse along the trajectory, since the inverse kinematic relationship is learned offline. With this control scheme, the input to KSOM network is given as

$$\Delta\mathbf{u} = K_p.(\mathbf{u}_t - \mathbf{u}_c). \tag{21}$$

The global stabilizing controller can be obtained only if the inverse adaptively changes along the path. In the conventional KSOM algorithm, the winning neuron is selected based on the target and hence the inverse will be fixed for a given target location, which would result in a local controller. In our approach, the winning neuron is selected based on the current end-effector position such that the inverse Jacobian changes as the end-effector traverses along the path. The desired joint velocity is then computed with the above input using (14).

After training, the winning neuron is the major contributor to the joint velocity. Hence, the computation reduces to a simple matrix multiplication in real-time, which makes the algorithm computationally cheaper. This is a major improvement in the case of visual servoing, where currently the computation poses a constraint in real-time implementation due to the cost associated with image processing.

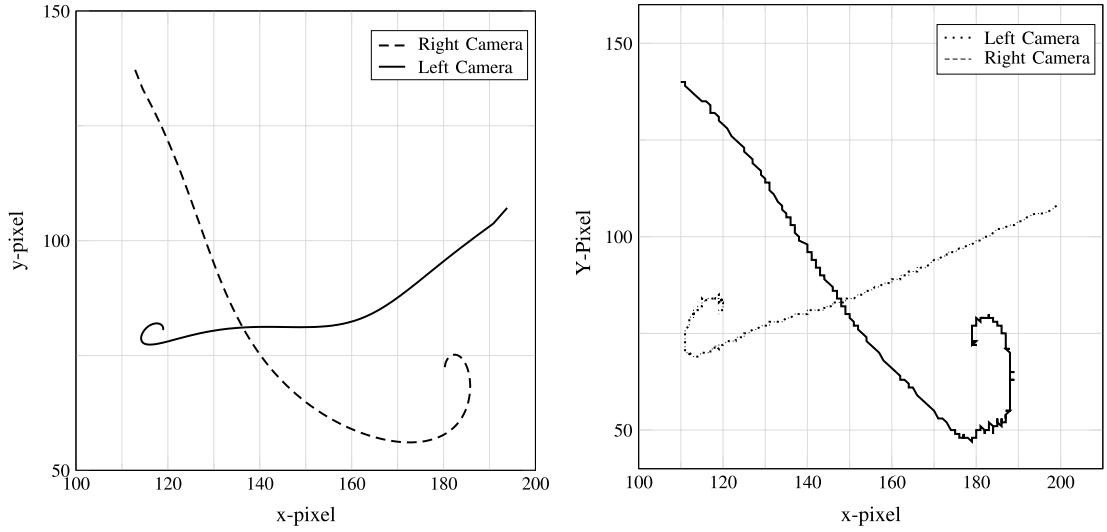
6.3.1. Stability analysis

It is clear from the empirical observation that KSOM approximates the pseudo-inverse of the Jacobian. We will check the Lyapunov stability of the proposed control scheme. Let us consider the quadratic Lyapunov function for positioning as

$$V = \mathbf{e}^T\mathbf{e} \tag{22}$$

where the error, $\mathbf{e} = \mathbf{u}_t - \mathbf{u}_c$ and $\mathbf{u}_t$ is constant for the positioning task. The time derivative of the Lyapunov function is given by,

$$\begin{aligned}
\dot{V} &= \mathbf{e}^T\dot{\mathbf{e}} \\
&= -\mathbf{e}^T\dot{\mathbf{u}}_c \\
&= -\mathbf{e}^T(\mathbf{LJ})\dot{\boldsymbol{\theta}} \\
&= -\mathbf{e}^T(\mathbf{LJ}).ks^{-1}\sum_{\gamma} h_\gamma\mathbf{A}_\gamma\mathbf{e} \\
&= -ks^{-1}\mathbf{e}^T(\mathbf{LJ}).\sum_{\gamma}(h_\gamma\mathbf{A}_\gamma)\mathbf{e} \\
&= -ks^{-1}\mathbf{e}^T(\mathbf{LJ})\left\{(\mathbf{LJ})^{+} - (\mathbf{LJ})^{+} + \left(\sum_{\gamma} h_\gamma\mathbf{A}_\gamma\right)\right\}\mathbf{e} \\
&= -ks^{-1}\mathbf{e}^T(\mathbf{LJ})(\mathbf{LJ})^{+}\mathbf{e} - ks^{-1}\mathbf{e}^T(\mathbf{LJ})\left\{(\mathbf{LJ})^{+} - \left(\sum_{\gamma} h_\gamma\mathbf{A}_\gamma\right)\right\}\mathbf{e} \\
&= -ks^{-1}\mathbf{e}^T\mathbf{I}'\mathbf{e} - ks^{-1}\mathbf{e}^T(\mathbf{LJ})(\tilde{\mathbf{A}})\mathbf{e} \\
&= -ks^{-1}\mathbf{e}^T\mathbf{I}'\mathbf{e} - ks^{-1}\mathbf{e}^T\tilde{\mathbf{I}}\mathbf{e}
\end{aligned} \tag{23}$$



(a) Simulation. (b) Real-time.

Fig. 6. End-effector motion in the vision space: Both the right and left camera views are shown (pixel). Smooth end-effector motion is generated from the initial position to the final position.

Table 2
Cartesian workspace limit.

Cartesian workspace
$-0.4\text{ m} \le x \le 0.4\text{ m}$ $0.3\text{ m} \le y \le 0.8\text{ m}$ $-0.15\text{ m} \le z \le 0.38\text{ m}$

Table 3
Initial and final end-effector positions.

Position	x (m)	y (m)	z (m)
Initial	−0.1	0.55	0.15
Final	0.1	0.75	0.35

where $\mathbf{I}' = (\mathbf{LJ})(\mathbf{LJ})^+$, $\tilde{\mathbf{A}} = \left\{(\mathbf{LJ})^+ - \left(\sum_\gamma h_\gamma \mathbf{A}_\gamma\right)\right\}$ is the approximation error of the KSOM network and $\tilde{\mathbf{I}} = (\mathbf{LJ})\tilde{\mathbf{A}}$. It is well known that, $\mathbf{I}' > 0$ and $\tilde{\mathbf{I}}$ is sign indefinite.

The above equation can be further simplified as,

$$\dot{V} < -ks^{-1}\mathbf{e}^T\mathbf{I}'\mathbf{e} + ks^{-1}\|\tilde{\mathbf{I}}\|.\|\mathbf{e}\|. \tag{24}$$

It is clear from the above equation that $\dot{V}$ is negative definite, if

$$\mathbf{e}^T\mathbf{I}'\mathbf{e} > \|\tilde{\mathbf{I}}\|.\|\mathbf{e}\|. \tag{25}$$

It is clear from the empirical observation that the KSOM linear Jacobian matrix approaches the local pseudo-inverse of the kinematic Jacobian with training and, hence, $\tilde{\mathbf{I}} \approx 0$. Thus Eq. (25) is true, which implies that the stability condition given by Eq. (24) is also satisfied. To make the algorithm robust, one can increase the number of neurons, which would increase the discretization of the workspace and, hence, $\|\tilde{\mathbf{I}}\|$ will be bounded. The global Lyapunov stability of the proposed scheme is thus guaranteed, with accurate offline learning of KSOM network.

6.4. Redundancy resolution

The proposed closed loop strategy is further extended to resolve the redundancy. As discussed in 6.2.3, KSOM generates a minimum joint motion since the Jacobian update law minimizes (18). Similarly, the KSOM based inverse kinematic algorithm is generalized to resolve the redundancy by minimizing the weighted least-norm discussed in [25]. The weighted norm solution penalizes the joint motion for achieving the desired additional task. The joint velocity $\dot{\boldsymbol{\theta}}$ which minimizes the weighted norm, $\|\dot{\boldsymbol{\theta}}^T\mathbf{W}\dot{\boldsymbol{\theta}}\|$ is given by

$$\dot{\boldsymbol{\theta}} = \mathbf{W}^{-1/2}(\mathbf{LJ})_w^+\dot{\mathbf{x}} \tag{26}$$

where $\mathbf{W}$ is the weight matrix which penalizes the joint motion to achieve the additional task, $(\mathbf{LJ})_w = (\mathbf{LJ})\mathbf{W}^{-1/2}$ and $(\mathbf{LJ})_w^+ = \mathbf{W}^{-T/2}(\mathbf{LJ})^T((\mathbf{LJ})\mathbf{W}^{-1}(\mathbf{LJ})^T)^{-1}$. A detailed discussion of the weighted least norm solution is available in [25]. Comparing (19) and (26), the Jacobian matrix of KSOM is updated to minimize

$$\min \|\Delta\boldsymbol{\theta}_{01} - \mathbf{W}^{-1/2}\mathbf{A}_i\mathbf{v}_{01}\|. \tag{27}$$

The above equation is analogous to (18), which converges to the minimum norm solution (19). The above update law is the same as (18) if $\mathbf{W} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix. Hence, the existing KSOM based learning method [6] is a particular case of the proposed generalized update law.

Similarly the coarse and the fine movement generated by the KSOM network is generalized as,

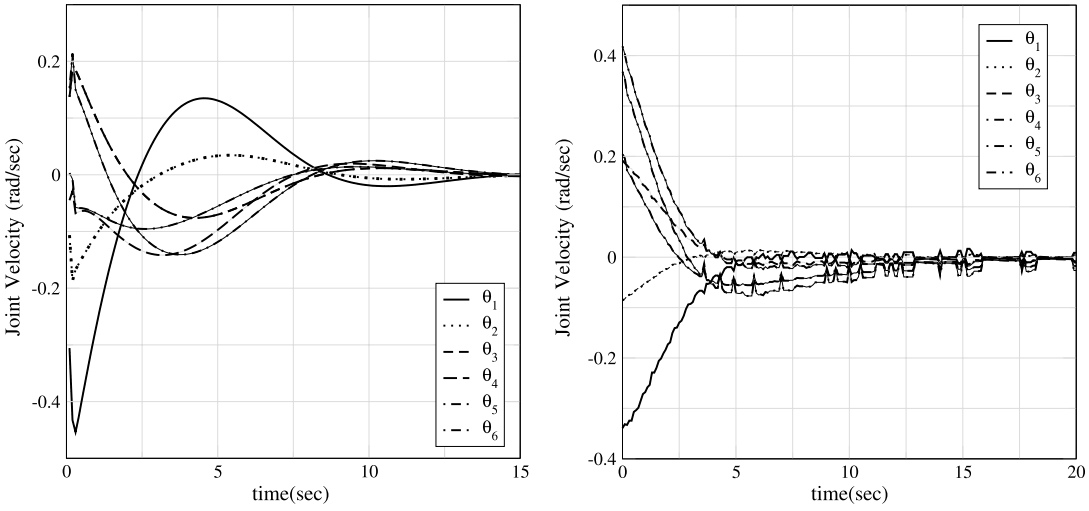
$$\boldsymbol{\theta}_0^{\text{out}} = s^{-1}\sum_\gamma h_\gamma(\boldsymbol{\theta}_\gamma + \mathbf{W}^{-1/2}\mathbf{A}_\gamma(\mathbf{u}_t - \mathbf{w}_\gamma)) \tag{28}$$

$$\boldsymbol{\theta}_1^{\text{out}} = \boldsymbol{\theta}_0^{\text{out}} + s^{-1}\sum_\gamma \mathbf{W}^{-1/2}h_\gamma\mathbf{A}_\gamma(\mathbf{u}_t - \mathbf{v}_0). \tag{29}$$

The above algorithm penalizes the joint motion based on the additional task and hence is expected to resolve the redundancy during the learning phase. It will be shown with empirical results that the proposed generalized scheme indeed resolves the redundancy by penalizing the joint motion with the weight matrix W.

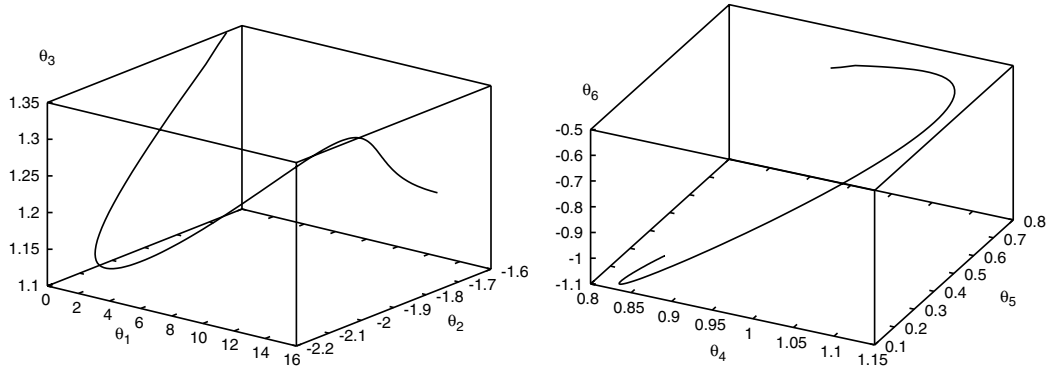
7. Results and discussion

The performance of the proposed controller scheme is tested from vision space in both simulation and real time. The controller is tested for the positioning task first within the learned workspace. The performance is then analyzed for tracking a straight line and an elliptical trajectory.



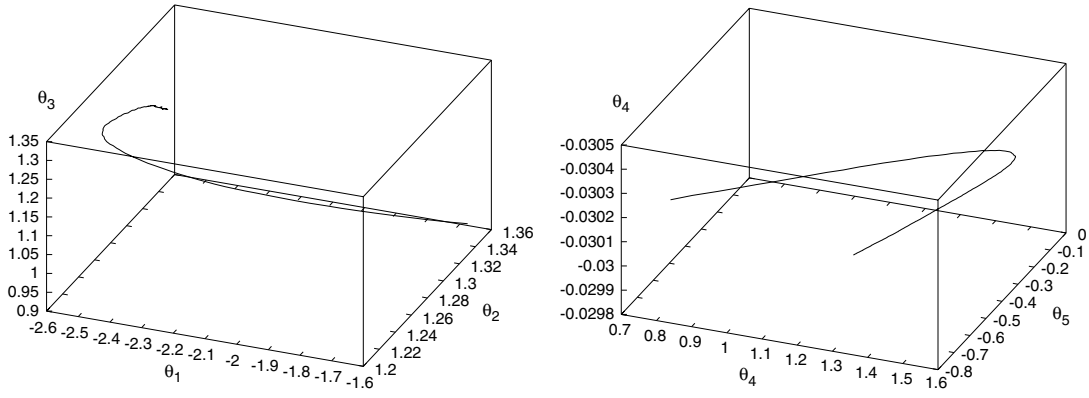
(a) Simulation. (b) Real-time.

Fig. 7. Joint velocity of all links (rad/s). The joint velocities are within the limits and smooth, resulting a smooth motion. All joint velocities converge to zero as the end-effector reaches the desired position.



(a) Joint angles 1–3. (b) Joint angles 4–6.

Fig. 8. Joint motion for the positioning task in simulation: (a) Joint angle: Links 1, 2 and 3 (rad) (b) Joint angle: Links 4, 5 and 6 (rad).



(a) Joint angles 1–3. (b) Joint angles 4–6.

Fig. 9. Joint motion for the positioning task in real-time: (a) Joint angle: Links 1, 2 and 3 (rad) (b) Joint angle: Links 4, 5 and 6 (rad).

7.1. KSOM learning

A 3-dimensional neural lattice with $7 \times 7 \times 7$ nodes is selected for learning the inverse kinematic map from the vision space to the joint space. Each node in the KSOM lattice is associated with an input weight vector, $\mathbf{w}_\gamma$ of dimension 4×1, which represents the pixel co-ordinates of the target in the stereo vision system.

Training data is generated using the forward kinematic model (1) and camera model obtained using the Tsai Algorithm [26]. The dimensions of the workspace, visible through the stereo vision, are

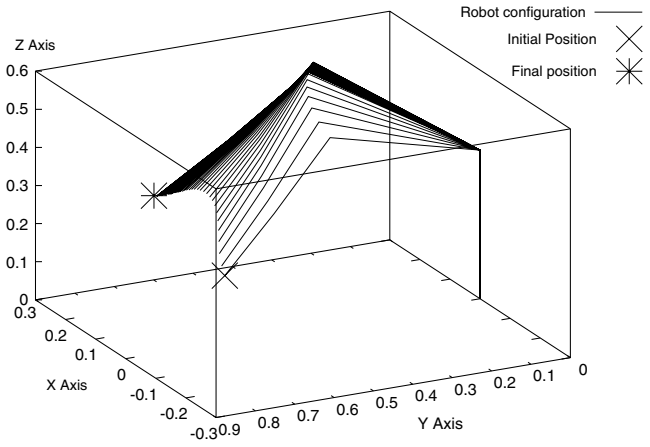


Fig. 10. End-effector motion in Cartesian space from the initial position to the final position (m).

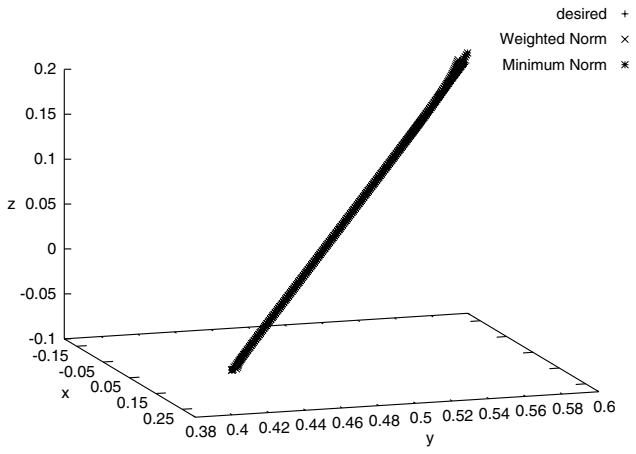


Fig. 11. End-effector motion in Cartesian space for tracking a line in (m).

tabulated in Table 2. Joint angle values are generated randomly within the specified bounds and only those end-effector positions are retained which lie within the visible workspace volume.

The inverse kinematic relationship is learned with 50,000 random points, which resulted in an average positioning accuracy of 0.12 m over the entire workspace.

7.2. Closed loop positioning results

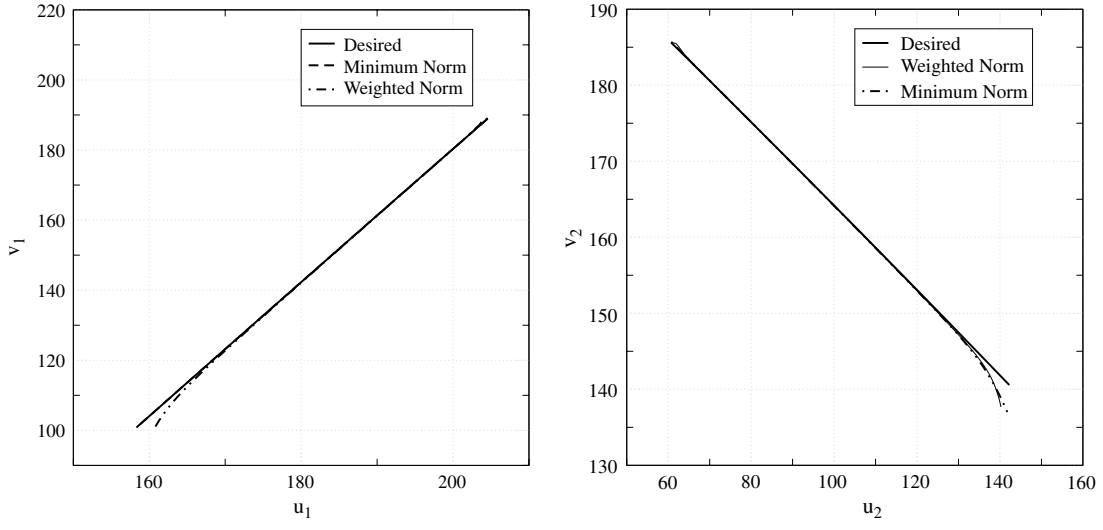
The learned map is used for closed loop control of the redundant manipulator. The initial and the final positions are considered the same in both simulation and real-time experiment. The chosen initial and final positions of the target in Cartesian space are tabulated in Table 3. The learned map is used as an approximation of the inverse Jacobian in closed loop visual servoing. The joint velocities computed using (21) are applied to the robot. The proportional gain is chosen as $K_p = 0.05$. The sampling rate is chosen as 0.1 s, similar to the experimental setup.

The end-effector motion in image space is shown in Fig. 6. The trajectory is smooth in simulation, however it is noisy in real-time. There is noise in the trajectory measurement during real-time implementation due to the inaccuracies in the image processing method. It is clear from the figure that the real-time and simulation results are very similar, indicating that the controller is performing well even though the learning is approximate.

The joint velocities are shown in Fig. 7. In real-time, the joint velocities are noisy due to noisy measurement of the end-effector position and numerical differentiation. It is clear from Fig. 7 and Table 1, that in both simulation and real-time the joint velocities are within their limits, and finally go to zero as the desired position is reached, indicating the stability of the proposed algorithm.

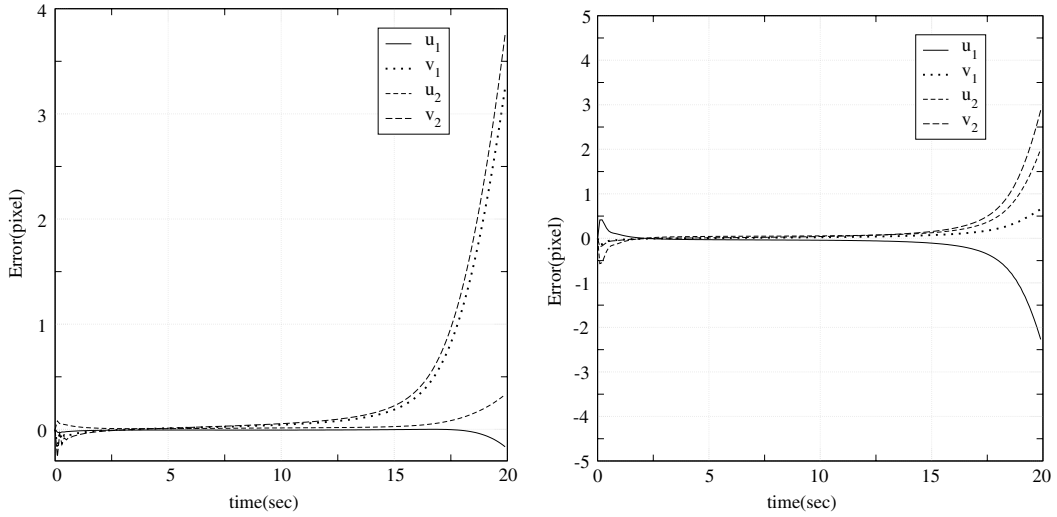
The joint angle trajectory along the motion is shown in Figs. 8 and 9 respectively for simulation and real-time. The joint angle variation is smooth and the angles are within the limits.

The end-effector motion in Cartesian space in the real-time experiment is presented in Fig. 10. It is observed that the end-effector reaches the final position with a 2 mm accuracy. In simulation, it is observed that the desired position can be reached with an accuracy of 0.24 pixel error. This accuracy can be further improved by executing the simulation for longer intervals. In real-time we could achieve a 1 pixel error due to the image processing limitation, which resulted in 2 mm error. The real-time performance is influenced by the measurement noise, which affects the positioning accuracy. The response is slightly sluggish in real-time compared to that in simulation. This is due to the image processing noise which would be comparable as the end-effector approaches the desired position.



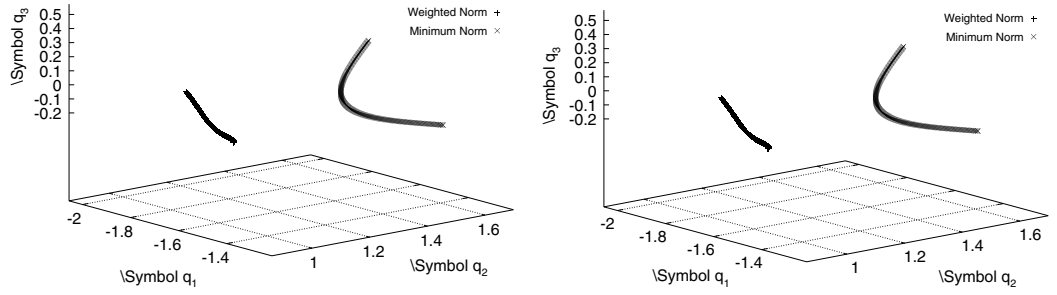
(a) Camera 1. (b) Camera 2.

Fig. 12. Robot end-effector position in vision space while tracking the straight line: (a) Camera: 1 (b) Camera: 2.



(a) Minimum norm. (b) Weighted norm.

Fig. 13. Vision space error while tracking the straight line: (a) Minimum norm (b) Weighted norm.



(a) Joint angles 1–3. (b) Joint angles 4–6.

Fig. 14. Joint motion while tracking a line: (a) Joint angle configuration: Links 1, 2 and 3 (rad) (b) Joint angle configuration: Links 4, 5 and 6 (rad).

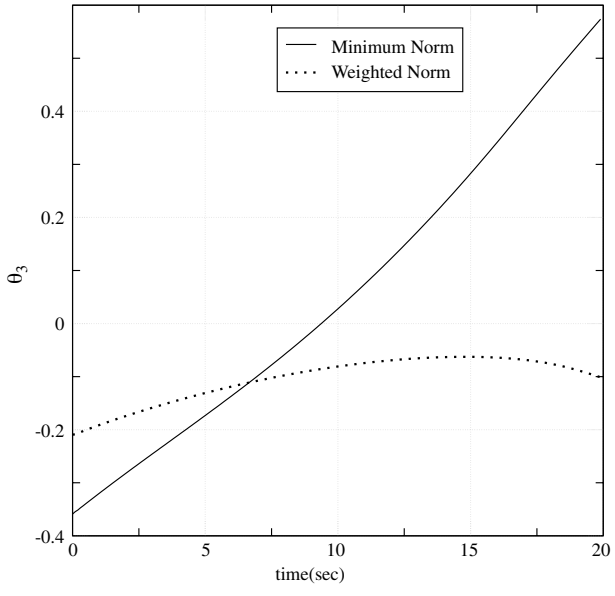


Fig. 15. Motion of the 3rd joint while tracking a straight line. A minimum change is observed in the case of the weighted norm solution.

7.3. Tracking trajectories defined in Cartesian space

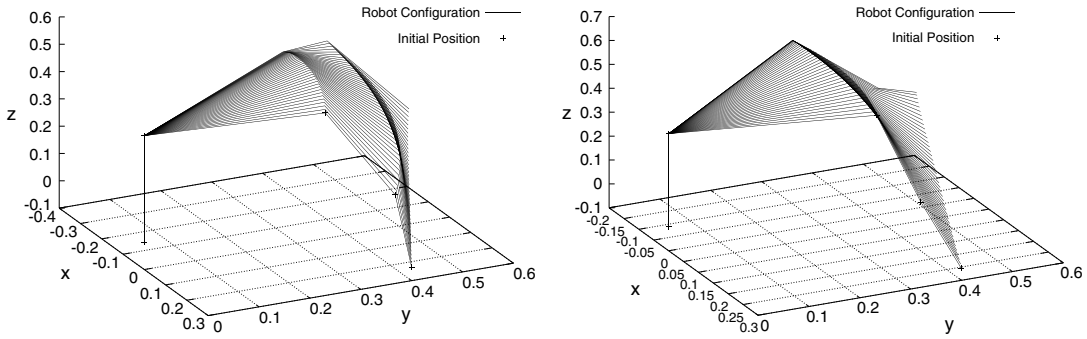
In this section, we will discuss the simulations performed to analyze the tracking performance of the proposed controller. The simulation is performed with the end-effector trajectories defined in the Cartesian space. The main purpose of this simulation is to show that the KSOM network learns the inverse kinematic map over the entire workspace and hence it guarantees global stability. To demonstrate our proposition we have simulated two trajectories in Cartesian space: (i) a straight line and (ii) an ellipse.

The trajectories are tracked with KSOMs learned by minimizing the least norm and weighted norm of joint velocity. The weighted norm solution is learned with the weight matrix, $\mathbf{W} = \text{diag}(1, 1, 100, 1, 1, 1, 1)$, which constrains the motion of the third joint of the manipulator. In case of the weighted norm, the manipulator is expected to track the trajectory with constrained motion of the third joint.

7.3.1. Tracking a straight line

The proposed scheme is tested first for tracking a straight line in the Cartesian space. The straight line is particularly chosen since it is well known that tracking a straight line with a revolute joint manipulator is much more difficult than a smooth curved trajectory. The desired end-effector position in vision space is obtained for the straight line, using the Camera model, and is given as input to the controller. A straight line passing across the entire workspace can be tracked only if the inverse Jacobian $(\mathbf{LJ})^+$ is learned accurately around each operating point.

The line connecting the points $[0.3, 0.7, 0.05]^T$ and $[-0.2, 0.6, 0.28]^T$ is considered for tracking. The end-effector trajectory while moving along the line is shown in Fig. 11. The corresponding vision space trajectory is shown in Fig. 12. The r.m.s. tracking errors with the minimum norm and the weighted minimum norm solution



(a) Minimum norm. (b) Weighted norm.

Fig. 16. Robot configuration while tracking a straight line: (a) Minimum norm solution (b) Weighted norm solution.

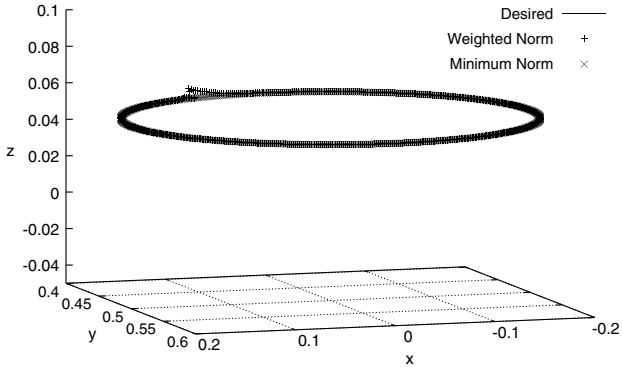


Fig. 17. End-effector motion in Cartesian space from the initial position to the final position (m).

are 0.7 mm and 1.3 mm respectively. The corresponding errors in vision space are observed as 0.067 and 0.27 pixels respectively.

The instantaneous tracking error in vision space is shown in Fig. 13, which shows that the controller tracks the trajectory with an accuracy of ± 3 pixel. The tracking error is less than ± 1 pixel along the major portion of the trajectory. A large deviation is observed in a narrow operating zone indicating that learning is not complete at those locations.

The joint angle trajectories are shown in Fig. 14 and it is clear that the joint trajectories are smooth along the trajectory due to the topology conservation nature of KSOM network. The angular configuration of the third joint is less in the case of the weighted norm solution. To analyze the effect of redundancy resolution on the third joint, its trajectory is shown separately in Fig. 15. It is clear from the figure that the motion of the third joint is constrained in the case of the weighted norm solution.

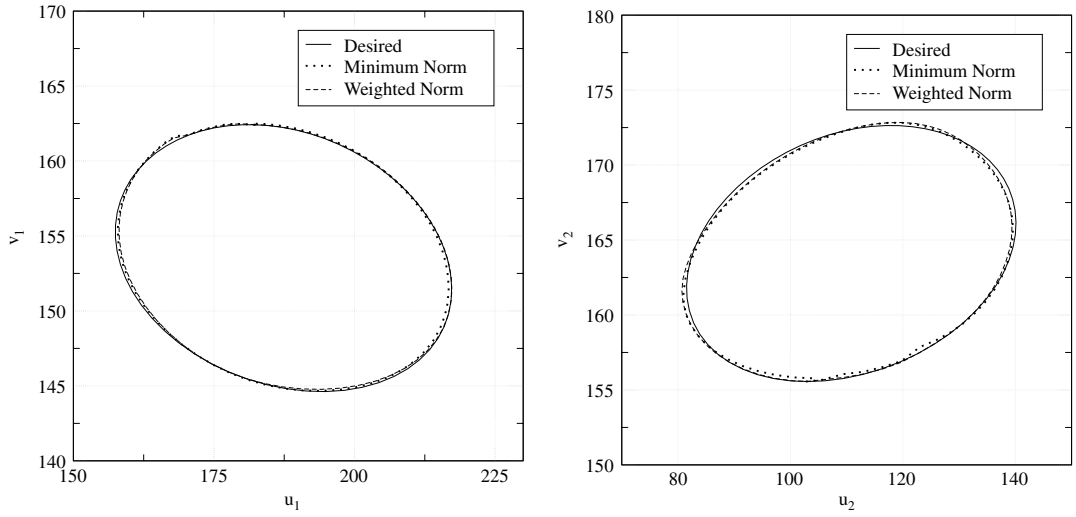
The robot joint configuration while moving along the straight line is shown in Fig. 16, which shows the effect of the weighted norm on each joint.

7.4. Tracking an elliptical trajectory

An elliptical trajectory is further tested to check the performance along the closed path. The tracking result is shown in Fig. 17. The demonstrated result shows the controller performance while tracking the elliptical trajectory given by

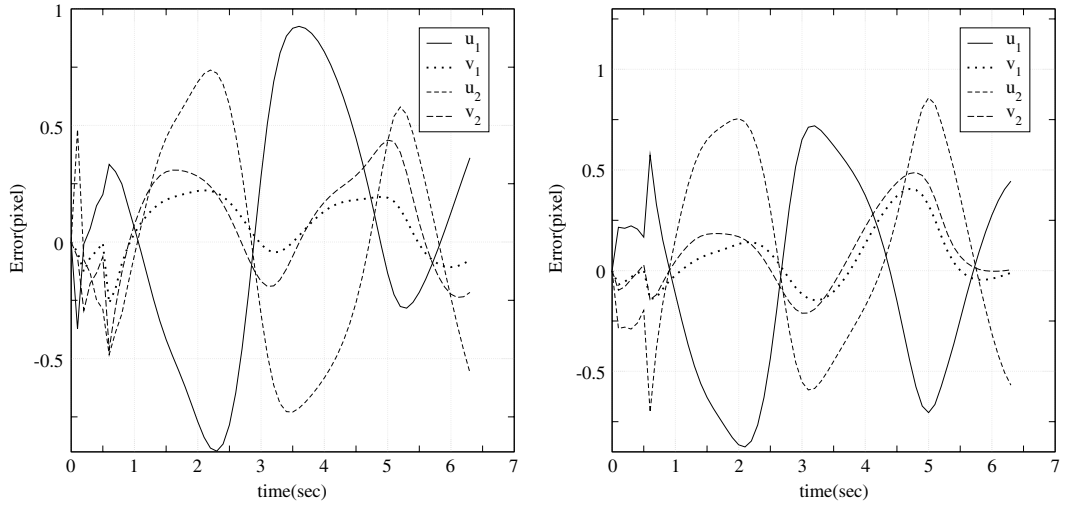
$$\begin{aligned} x &= 0.2\sin(t) \\ y &= 0.5 + 0.1\cos(t) \\ z &= 0.05. \end{aligned} \tag{30}$$

The r.m.s. tracking errors in the Cartesian space are observed as 0.68 mm and 0.62 mm for the minimum norm and weighted minimum norm solutions respectively. The end-effector trajectory in the vision space is shown in Fig. 18. The r.m.s. tracking error in vision space is observed as 0.165 pixels for the minimum norm solution and 0.164 pixels for the weighed minimum norm solution.



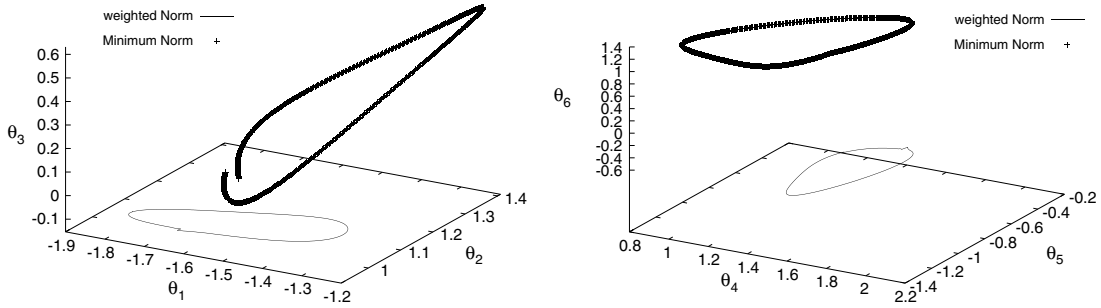
(a) Camera 1. (b) Camera 2.

Fig. 18. Robot end-effector position in vision space, while tracking the ellipse: (a) Camera: 1 (b) Camera: 2.



(a) Minimum norm. (b) Weighted norm.

Fig. 19. Vision space error while tracking the elliptical path: (a) Minimum norm (b) Weighted norm.



(a) Joint angles 1–3. (b) Joint angles 4–6.

Fig. 20. Joint motion in simulation: (a) Joint angle configuration: Links 1, 2 and 3 (rad) (b) Joint angle configuration: Links 4, 5 and 6 (rad).

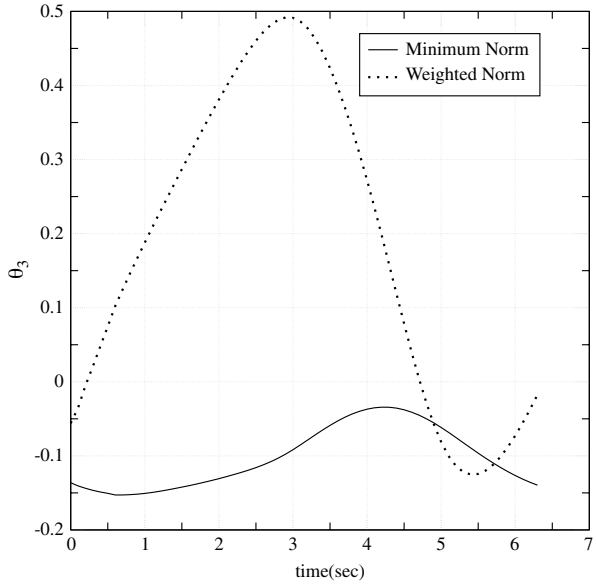


Fig. 21. Motion of the 3rd joint while tracking the ellipse. A minimum change is observed in the case of the weighted norm solution.

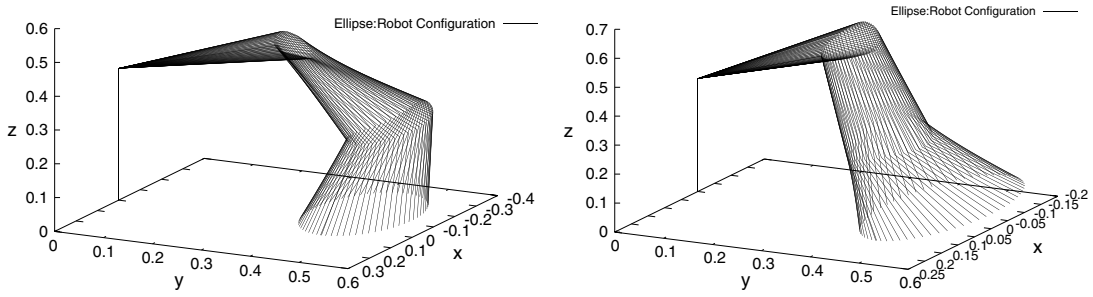
The instantaneous tracking error in vision space is shown in Fig. 19, which shows that controller tracks the trajectory with an accuracy of ± 1 pixel.

The joint angle trajectories are shown in Fig. 20 and the motion of the third joint is presented in Fig. 21. It is clear from the figures that the weighed norm constrains the motion of the third joint and the joint angle trajectory also follows a closed path.

The robot joint configuration is shown in Fig. 22, which clearly shows that the weighted norm solution constrains the motion of the third joint, which in turn effects a larger movement of the other joints.

8. Conclusion

A globally stable visual servoing scheme for redundant manipulators operating in an eye-to-hand configuration is discussed. A novel KSOM based stabilizing controller is proposed which ensures global stabilization in conjunction with the conventional proportional controller. Initially, KSOM is trained to learn the inverse kinematic relationship of the redundant manipulator from the image space to the joint space. It has been observed experimentally that KSOM approximates the pseudo-inverse of the Jacobian over the entire workspace. This eliminates the computation of the pseudo-inverse required for visual servoing. With this observation, a generalized learning algorithm is proposed for KSOM based inverse kinematic control to resolve the redundancy in the learning phase. The learned KSOM is used in conjunction with a proportional controller in real-time for closed loop visual servoing. It is shown through Lyapunov stability analysis that the proposed controller guarantees global stability if the learned map is sufficiently accurate. The proposed scheme is tested for a positioning



(a) Minimum norm. (b) Weighted norm.

Fig. 22. Robot configuration while tracking the ellipse: (a) Minimum norm solution (b) Weighted norm solution.

task in both simulation and real-time. The performance of the controller is analyzed in simulation with a straight line and an elliptical trajectory defined in Cartesian space. The redundancy is resolved during tracking by constraining the movement of the third joint of the manipulator. All the experiments are performed with a 7 DOF PowerCube™ manipulator in real-time.

In the future, the proposed control scheme will be further tested in performing additional tasks such as avoiding kinematic constraints and obstacles. The proposed scheme requires offline learning of the inverse Jacobian $(\mathbf{LJ})^{+}$. This work will be further extended to real-time estimation of this inverse Jacobian.

Acknowledgement

This research was supported by Defence Research and Development Organization (DRDO), Govt. of India, under the project titled "Neural Network based Visual Motor Coordination of a 7 DOF Redundant Manipulator and FPGA implementation of Neural Control Algorithms". The project number is "DRDO/EE/20080119".

References

[1] S. Hutchinson, G.D. Hager, P.I. Corke, A tutorial on visual servo control, IEEE Transactions on Robotics and Automation 12 (5) (1996) 651–670.
[2] E. Malis, S. Benhimane, A unified approach to visual tracking and servoing, Robotics and Autonomous Systems 52 (1) (2005) 39–52.
[3] D. Kragic, H.I. Christensen, Cue integration for visual servoing, IEEE Transactions on Robotics and Automation 17 (1) (2001) 18–27.
[4] E. Marchand, F. Chaumette, Statistically robust 2-D visual servoing, IEEE Transactions on Robotics and Automation 22 (2) (2006) 415–420.
[5] W.-C. Chang, Precise positioning of binocular eye-to-hand robotic manipulators, Journal of Intelligent and Robotics Systems 49 (2007) 219–236.
[6] L. Behera, N. Kirubanandan, A hybrid neural control scheme for visual-motor coordination, IEEE Control System Magazine 19 (4) (1999) 34–41.
[7] F. Chaumette, S. Hutchinson, Visual servo control part I: basic approaches, IEEE Robotics and Automation Magazine 13 (4) (2006) 82–90.
[8] F. Chaumette, S. Hutchinson, Visual servo control part II: advanced approaches (tutorial), IEEE Robotics and Automation Magazine 14 (1) (2007) 109–118.
[9] D. Kragic, H. Christensen, Survey on visual servoing for manipulation, http://citeseer.ist.psu.edu/484743.html.
[10] F. Chaumette, Potential problems of stability and convergence in image based and position based visual servoing, in: D. Kreigman, G. Hager, S. Morse (Eds.), T. Confluence of Vision, Control, in: Lecture notes in Control and Information Sciences, vol. 237, Springer-Verlag, New York, 1998, pp. 66–78.
[11] J.-T. Lapreste, F. Jurie, M. Dhome, F. Chaumette, An efficient method to compute the inverse Jacobian matrix in visual servoing, in: Proceedings of IEEE International Conference on Robotics and Automation, vol. 1, IEEE, New Orleans, LA, 2004, pp. 727–732.
[12] B. Siciliano, Kinematic control of redundant robot manipulators: a tutorial, Journal of Intelligent and Robotic systems 4 (4) (1990) 201–212.
[13] D.E. DeMers, K.K. Kreutz-Delgado, Solving the inverse kinematics problem for robots with excess degrees-of-freedom, http://citeseer.ist.psu.edu/375197.html.
[14] F. Chaumette, E. Marchand, A redundancy-based iterative approach for avoiding joint limits: application to visual servoing, IEEE Transactions on Robotics and Automation 17 (5) (2001) 719–730.
[15] T. Martinetz, H. Ritter, K. Schulten, Learning of visuomotor-coordination of a robot arm with redundant degrees of freedom, in: Proceedings of the Int. Conf. on Parallel Processing in Neural Systems and Computers, 1990, pp. 431–434.
[16] M. Han, N. Okada, E. Kondo, Coordination of an uncalibrated 3-D visuo-motor system based on multiple self-organizing maps, JSME International Journal, Series C 49 (1) (2006) 230–239.
[17] S. Kumar, P. Prem Kumar, A. Dutta, L. Behera, Visual motor control of a 7DOF redundant manipulator using redundancy preserving learning network, Robotica (2009) 1–16 (first view).
[18] G.D.A. Barreto, A.F.R. Araujo, Self-organizing feature maps for modeling and control of robtoic manipulators, Journal of Intelligent and Robotic Systems 36 (2003) 407–450.
[19] F. Chaumette, Image moments: a general and useful set of features for visual servoing, IEEE Transactions on Robotics and Automation 20 (4) (2004) 713–723.
[20] Amtec robotics, http://www.amtec-robotics.com/.
[21] Unibrain, http://www.unibrain.com/.
[22] R.Y. Tsai, A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses, IEEE Journal of Robotics and Automation RA-3 (4) (1987) 323–344.
[23] T. Hesselroth, K. Sarkar, P.P. Smagt, K. Schulten, Neural network control of a pneumatic robot arm, IEEE Transactions on Systems, Man and Cybernetics 24 (1) (1994) 28–38.
[24] Y. Zhang, J. Wang, Y. Xu, A dual neural network bi-criteria kinematic control of redundant manipulators, IEEE Transactions on Robotics and Automation 18 (6) (2002) 923–931.
[25] T.F. Chan, R.V. Dubey, A weighted least-norm based solution scheme for avoiding joint limits for redundant joint manipulators, IEEE Transactions on Robotics and Automation 11 (2) (1995) 286–292.
[26] R. Wilson, Tsai camera calibration software, http://www.cs.cmu.edu/~rgw/TsaiCode.html.

P. Prem Kumar received his B.E degree in Electrical and Electronics Engineering from Thiyagarajar College Of Engineering, Madurai, India, in 2003 and M. Tech degree in Control System Engineering from the Indian Institute of Technology Kanpur, India in 2005. From 2005 to 2006, he was a design engineer at Larsen & Toubro Ltd., India. He is currently pursuing the Ph.D. degree in the Department of Electrical Engineering, Indian Institute of Technology Kanpur, India. His primary research interests include Vision based Redundant Manipulator Control in dynamic environments and Neuro-Fuzzy control schemes.

Laxmidhar Behera is currently Associate Professor, Department of Electrical Engineering, IIT Kanpur. He joined the Intelligent Systems Research Center (ISRC), University of Ulster, UK, as a reader on sabbatical from IIT Kanpur during 2007–09. He obtained his B.Sc. (engineering) and M.Sc. (engineering) from NIT Rourkela in 1988 and 1990 respectively. A Ph.D. from IIT Delhi, he has worked as Assistant Professor in BITS Pilani during 1995–99 and pursued his post-doctoral studies in the German National Research Center for Information Technology, GMD, Sank Augustin, Germany during 2000–01. He has also worked as visiting researcher/professor in FHG, Germany, and ETH, Zurich, Switzerland. He has more than 120 papers to his credit published in refereed journals and presented in conference proceedings. His research interests include intelligent control, robotics, neural networks, and cognitive modeling.