Expert Systems with Applications 40 (2013) 1721–1736

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

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

Neural network Reinforcement Learning for visual control of robot manipulators

Zoran Miljković [a,*], Marko Mitić [a], Mihailo Lazarević [b], Bojan Babić [a]

[a] University of Belgrade, Faculty of Mechanical Engineering, Production Engineering Department, Kraljice Marije 16, 11120 Belgrade 35, Serbia
[b] University of Belgrade, Faculty of Mechanical Engineering, Department of Mechanics, Kraljice Marije 16, 11120 Belgrade 35, Serbia

ARTICLE INFO

Keywords:
Reinforcement Learning
Neural network
Robot manipulator
Image Based Visual Servo control
Intelligent hybrid control

ABSTRACT

It is known that most of the key problems in visual servo control of robots are related to the performance analysis of the system considering measurement and modeling errors. In this paper, the development and performance evaluation of a novel intelligent visual servo controller for a robot manipulator using neural network Reinforcement Learning is presented. By implementing machine learning techniques into the vision based control scheme, the robot is enabled to improve its performance online and to adapt to the changing conditions in the environment. Two different temporal difference algorithms (Q-learning and SARSA) coupled with neural networks are developed and tested through different visual control scenarios. A database of representative learning samples is employed so as to speed up the convergence of the neural network and real-time learning of robot behavior. Moreover, the visual servoing task is divided into two steps in order to ensure the visibility of the features: in the first step centering behavior of the robot is conducted using neural network Reinforcement Learning controller, while the second step involves switching control between the traditional Image Based Visual Servoing and the neural network Reinforcement Learning for enabling approaching behavior of the manipulator. The correction in robot motion is achieved with the definition of the areas of interest for the image features independently in both control steps. Various simulations are developed in order to present the robustness of the developed system regarding calibration error, modeling error, and image noise. In addition, a comparison with the traditional Image Based Visual Servoing is presented. Real world experiments on a robot manipulator with the low cost vision system demonstrate the effectiveness of the proposed approach.

1. Introduction

In recent years, a wide variety of applications regarding autonomous robot behavior in unstructured and unknown environments have been developed. Similarly to biological systems, new generations robots are able to learn and to adapt to changing conditions in real time. This property is necessary when facing difficult tasks in practice such as search and rescue missions, reconnaissance, surveillance, and inspection in complex and dangerous surroundings. The possibility of robot vision can be crucial in these assignments since it mimics human sense and allows for noncontact measurement of the environment (Hutchinson, Hager, & Corke, 1996).

Visual servoing or visual servo control represents a known solution to control the motion of robot manipulators in structured environments. It involves various techniques from image processing, computer vision, and control theory (Chaumette & Hutchinson, 2006). By using these approaches, sophisticated autonomous systems containing low cost sensors and actuators can be developed. In visual servo control, the information from one or more cameras is used within the control loop in order to ensure the desired position of the robot as required by a task. The vision data is acquired from the camera which is placed directly onto the manipulator (eye in hand configuration) or in a fixed position over the scene (eye to hand configuration). In the next step, the features on the image plane are servo controlled to their goal positions. It is well known that points are the simplest features that can be extracted from an image, both from a geometrical and an image processing point of view (Fomena, Omar, & Chaumette, 2011). Therefore, most of the applications in visual servoing are based on an image of points, such as visual homing (Basri, Rivlin, & Shimshoni, 1999), navigation and path planning of the robot manipulators (Cowan & Koditschek, 1999; Mezouar & Chaumette, 2002), mobile robot navigation (Ma, Kosecka, & Sastry, 1999; Mariottini, Oriolo, & Prattichizzo, 2007; Miljković, Vuković, Mitić, & Babić, in press), and stabilization of aerial vehicles (Ceren & Altuğ, 2012; Hamel & Mahony, 2002).

One of the many challenges in visual servo control includes the maintenance of visual features within the field of view of the camera. Also, robustness to camera calibration parameters and image noise is very important in real world applications. Likewise, unknown disturbances during the motion as well as the robot motion itself can result in none of the visual features in the image plane. In

* Corresponding author. Tel.: +381 11 3302 468; fax: +381 11 3370 364.
E-mail address: zmiljkovic@mas.bg.ac.rs (Z. Miljković).
URL: http://cent.mas.bg.ac.rs/english/staff/zmiljkovic.htm (Z. Miljković).

0957-4174/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2012.09.010

these cases, the traditional Image Based Visual Servo (IBVS) or Position Based Visual Servo (PBVS) schemes would fail. To prevent this from happening several interesting approaches are reported in literature. Chesi, Hashimoto, Prattichizzo, and Vicino (2004) proposed an approach which consists of a switching among Position Based control strategies and backward motion for keeping the visual features in the field of view. In Corke and Hutchinson (2001), an Image Based Visual Servoing scheme with a potential function that repels feature points from the boundary of the image plane is developed. Similarly, a new path planning scheme for visibility constraint problem and optimal visual servoing has been proposed in Chesi and Hung (2007). Schramm and Morel (2006) presented a solution based on the path planning technique for an uncalibrated camera that guarantees visibility of the observed target. Specific visual features which ensure that the robot navigates within the visibility path are defined in Remazeilles and Chaumette (2007). An interesting approach for avoiding typical problems of disturbance based disappearance of image features with cooperative cameras in eye in hand and eye to hand configurations is proposed in Garcia-Aracil, Perez-Vidal, Sabater, Morales, and Badesa (2011). However, none of these approaches incorporates learning methods for online improvement of the systems performance in changing real time conditions. In this paper, a Reinforcement Learning (RL) solution for a visual servoing task regarding a visibility problem, incorrect calibration parameters, white image noise, and modeling error is developed.

Reinforcement Learning (Sutton & Barto, 1998) is a popular area of research, widely used in various disciplines such as manufacturing technology (Shin, Ryu, & Jung, 2012), multi agent technology (Jiang & Sheng, 2009) or computer vision (Sahba, Tizhoosh, & Salama, 2008). RL in robotics is often applied for control of wheeled mobile robots (Mitić, Miljković, & Babić, 2011), robot manipulators (Song & Chu, 1998), or humanoid robots (Khan, Herrmann, Lewis, Pipe, & Melhuish, 2012). However, results on integration of Reinforcement Learning and visual servoing for robot applications have not been reported very often in literature. One of the first solutions that combine these two techniques is presented in Distante, Anglani, and Taurisano (2000). In this work, Q-learning controllers for target reaching and grasping are developed and implemented on an industrial robot manipulator. However, the study does not consider a visibility problem or camera disturbances. Also, computationally expensive methods for image segmentation and two step image feature extraction are used within the control scheme. More drawbacks of the proposed setup are offline training with traditional Q-learning until convergence is achieved, and the use of a spherical object of known size for robot grasping. Likewise, a certain number of applications of the aforementioned methods are developed for mobile robots. To overcome the difficulties due to the generalization problem for a system with continuous variables, a neural network (NN) based Q-learning for visual servoing of a mobile robot is presented in Gaskett (2002) and Gaskett, Fletcher, and Zelinsky (2000). A similar approach for learning a direct mapping from image space to actuator command using Reinforcement Learning is given in Takahashi, Takeda, and Asada (1999). Some other studies regarding this integration for mobile robots are given in Asada, Noda, Tawaratsumida, and Hosoda (1996), Busquets, de Mantaras, Sierra, and Ditterich (2002), Hafner and Riedmiller (2003) and Martínez-Marín and Duckett (2005).

Recently, a work that uses Q-learning and eye to hand IBVS for robust grasping has been presented in Wang, Lang, and de Silva (2010). Here, the Q-learning and IBVS controller are integrated into the hybrid system for servoing task of a mobile robot. Compared with the control scheme proposed in this paper, several important differences need to be stressed out. Firstly, the algorithm in Wang et al. (2010) is not intended for robot manipulators because of its limited workspace. The switching between two controllers in hybrid scheme presented in Wang et al. (2010) is not recommended in the starting pose of a robot manipulator because of the chance of joint damage, especially when the pose is close to the boundaries of the workspace. A more advanced control solution with two machine learning based phases and two independent areas of interest is employed in the proposed control scheme. Additionally, the depth parameter is not included in the state space in our approach so as to simplify and enhance the learning process. Secondly, traditional Q-learning algorithm with offline training of the Q-function is implemented in the hybrid system in Wang et al. (2010). In our method, neural network with the database of learning samples is used for speeding up the convergence of the algorithm. In addition, two Reinforcement Learning algorithms with neural networks (Q-learning and SARSA) are separately developed and tested through different simulation scenarios. Thirdly, the method in Wang et al. (2010) does not consider challenging conditions such as changing scene illumination, incorrect camera calibration parameters, noise in the acquired image, and/or modeling error. An advanced camera is used for image acquisition and independent color blob tracking software is employed to track the image coordinates of the target object in Wang et al. (2010). Moreover, laser sensor is used for depth estimation. In this study, the developed control scheme is tested through various simulations regarding calibration error, white image noise, and modeling error. The experiments in the real world with the low cost vision system demonstrate the robustness of the presented hybrid algorithm.

The paper is organized as follows. The main contributions are explicitly stated in Section 2. A traditional Image Based Visual Servoing scheme is stated in Section 3, while the description of the neural network Reinforcement Learning controller is presented in Section 4. In Section 5, an intelligent hybrid visual control scheme incorporating developed controllers is explained in details. The proposed approach is evaluated in various simulation scenarios provided in Section 6. Experimental results on a robot manipulator are given in Section 7, followed by Section 8 which summarizes findings and the contributions of this study.

2. Contributions of the paper

This study differs from all of the aforementioned approaches. The fundamental characteristics of the developed intelligent vision based control approach are:

- Neural network Reinforcement Learning algorithm is implemented into the hybrid control scheme in order to enable the online capability of learning and adapting to the changing conditions in the environment. Two different temporal difference algorithms (Q-learning and SARSA) coupled with neural networks are developed separately and tested through different visual control scenarios. An implemented database with the representative learning samples enables the acceleration of the NN convergence to the near optimal Q-function. Moreover, to the authors' best knowledge, this is the first paper that implements neural network SARSA learning in a robot based task.
- The visual control task of a robot manipulator is separated into two independent steps. The neural network Reinforcement Learning controller with a database of learning samples is used for centering the robot with the target. The approaching behavior in the second step includes switching control between the Image Based Visual Servo controller and the neural network Reinforcement Learning controller so as to enable the robot to remain on the preferred path. The independent areas of interest for visual features are defined in the image plane separately for both control steps. In this way, the visibility of an image feature is ensured in every control loop.

– Various simulations are developed in Matlab to show robustness of the system concerning calibration error, modeling error, and image noise. Moreover, a comparison with the traditional Image Based Visual Servo control is presented.
– Real world experiments are performed on a robot manipulator *NeuroArm Manipulator System* from *Neurorobotics* with a low cost camera in order to prove the effectiveness of the proposed intelligent system.

3. Image Based Visual Servo control

This study presents a novel hybrid visual control of a robot manipulator with a camera on the robot's end-effector observing the object of interest (eye in hand configuration). A short review of the traditional Image Based Visual Servoing scheme (Chaumette & Hutchinson, 2006; Hutchinson et al., 1996) as used in this work is presented in this section.

A visual servo control (or visual servoing) task refers to a minimization of an error vector defined in the image plane. This vector is typically defined as

$$\mathbf{e}(t) = \mathbf{s}(\mathbf{m}(t), \mathbf{a}) - \mathbf{s}^*. \tag{1}$$

In this equation, $\mathbf{m}(t)$ represents a set of visual or image measurements, e.g. coordinates of the desired image points or the coordinates of the centroid of an object. With these measurements the $\mathbf{s}(\mathbf{m}(t),\mathbf{a})$ vector containing k visual features is calculated. Intrinsic camera parameters or the 3D model of the object of interest is described by parameter $\mathbf{a}$. The desired feature information is defined with $\mathbf{s}^*$.

The definition of parameter $\mathbf{s}$ determines the visual servo control scheme. In this study, the Image Based Visual Servo control scheme is used so that vector $\mathbf{s}$ consists of a set of visual features acquired from the image plane. To design the visual servo controller, a relationship between the time derivative of $\mathbf{s}$ and camera velocity must be determined first. This relationship is given as

$$\dot{\mathbf{s}} = \mathbf{L}_\mathbf{s}\mathbf{v}_c, \tag{2}$$

where $\mathbf{v}_c = (v_c, w_c)$ represents spatial camera velocity, with v_c and w_c as instantaneous linear and angular velocity, respectively. $\mathbf{L}_\mathbf{s} \in \mathrm{R}^{k\times 6}$ denotes the interaction matrix (i.e. image Jacobian or feature sensitivity matrix). A relationship between camera velocity and error vector is obtained from Eqs. (1) and (2), considering that $\mathbf{s}^*$ is constant parameter because of the fixed goal pose, as

$$\dot{\mathbf{e}} = \mathbf{L}_\mathbf{e}\mathbf{v}_c, \tag{3}$$

where $\mathbf{L}_\mathbf{e} = \mathbf{L}_\mathbf{s}$ (see Chaumette & Hutchinson, 2006). In order to decrease the error exponentially, i.e. $\dot{\mathbf{e}} = -\lambda\mathbf{e}$, camera velocity is presented as

$$\mathbf{v}_c = -\lambda\mathbf{L}_\mathbf{e}^+\mathbf{e}, \tag{4}$$

where $\mathbf{L}_\mathbf{e}^+ \in \mathbf{R}^{k\times 6}$ is the Moore–Penrose pseudo inverse matrix of $\mathbf{L}_\mathbf{e}$. When $\mathbf{L}_\mathbf{e}$ holds rank 6, $\mathbf{L}_\mathbf{e}^+$ is calculated as $\mathbf{L}_\mathbf{e}^+ = \left(\mathbf{L}_\mathbf{e}^T\mathbf{L}_\mathbf{e}\right)^{-1}\mathbf{L}_\mathbf{e}^T$ which allows $\|\mathbf{v}_c\|$ and $\|\dot{\mathbf{e}} - \lambda\mathbf{L}_\mathbf{e}\mathbf{L}_\mathbf{e}^+\mathbf{e}\|$ to be minimal. If $\det\mathbf{L}_\mathbf{e} \neq 0$ when $k = 6$, the invert of $\mathbf{L}_\mathbf{e}$ can be introduced, resulting in the control equation $\mathbf{v}_c = -\lambda\mathbf{L}_\mathbf{e}^{-1}\mathbf{e}$. Since $\mathbf{L}_\mathbf{e}$ or $\mathbf{L}_\mathbf{e}^+$ cannot be calculated in real conditions, the approximation of $\mathbf{L}_\mathbf{e}$, $\widehat{\mathbf{L}_\mathbf{e}^+}$, must be introduced. Thus, the control low of most visual servo controllers is given by

$$\mathbf{v}_c = -\lambda\widehat{\mathbf{L}_\mathbf{e}^+}\mathbf{e}. \tag{5}$$

In this paper, we used the coordinates of the interest points in the image plane to define vector $\mathbf{s}(\mathbf{m}(t),\mathbf{a})$. As stated, $\mathbf{m}$ represents the image measurements while $\mathbf{a}$ refers to intrinsic camera parameters. The relationship between the camera frame and the image frame is presented in Fig. 1. A 3D point $\mathbf{P}$ can project into the image plane as a 2D point using the prospective projection as

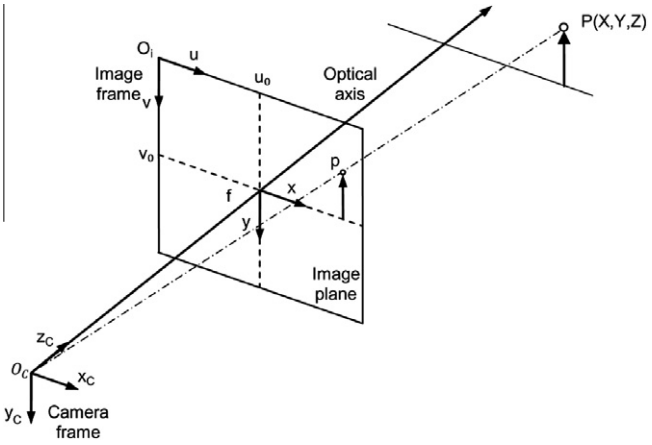

Fig. 1. The central-perspective model.

$$\begin{aligned} x &= \frac{X}{Z} = \frac{(u-u_0)}{f_u}, \\ y &= \frac{Y}{Z} = \frac{(v-v_0)}{f_v}, \end{aligned} \tag{6}$$

where $\mathbf{m} = (u, v)$ represents the image plane coordinates of the point of interest in pixels, and $\mathbf{a} = (u_0, v_0, f_u, f_v)$ is the camera intrinsic parameters vector. u_0 and v_0 are coordinates of the principal point, while f_u and f_v denote focal lengths. The velocity of the 3D point regarding the camera frame is

$$\dot{\mathbf{P}} = -v_c - w_c \times \mathbf{P}, \tag{7}$$

or in scalar form, considering that $\mathbf{P} = (X, Y, Z)$, $v_c = (v_x, v_y, v_z)$ and $w_c = (w_x, w_y, w_z)$, as

$$\begin{aligned} \dot{X} &= -v_x - w_y \times Z + w_z Y, \\ \dot{Y} &= -v_y - w_z \times X + w_x Z, \\ \dot{Z} &= -v_z - w_x \times Y + w_y X. \end{aligned} \tag{8}$$

The time derivative of Eq. (6) is

$$\begin{aligned} \dot{x} &= \frac{\dot{X}}{Z} - \frac{X\dot{Z}}{Z^2} = \frac{\dot{X} - x\dot{Z}}{Z}, \\ \dot{y} &= \frac{\dot{Y}}{Z} - \frac{Y\dot{Z}}{Z^2} = \frac{\dot{Y} - y\dot{Z}}{Z}. \end{aligned} \tag{9}$$

From Eqs. (8) and (9) we obtain

$$\begin{aligned} \dot{x} &= -\frac{v_x}{Z} + \frac{x v_z}{Z} + xyw_z - (1+x^2)w_y + yw_z, \\ \dot{y} &= -\frac{v_{xy}}{Z} + \frac{y v_z}{Z} + (1+y^2)w_x - xyw_y - xw_z, \end{aligned} \tag{10}$$

which can be rewritten as

$$\dot{\mathbf{x}} = \mathbf{L}_\mathbf{x}\mathbf{v}_c, \tag{11}$$

where $\mathbf{x} = (x, y)$, with $\mathbf{L}_\mathbf{x}$ representing the interaction matrix in the form

$$\mathbf{L}_\mathbf{x} = \begin{bmatrix} -\frac{1}{Z} & 0 & -\frac{x}{Z} & xy & -(1+x^2) & y \\ 0 & -\frac{1}{Z} & -\frac{y}{Z} & 1+y^2 & -xy & -x \end{bmatrix}. \tag{12}$$

Every control system that uses this form of interaction matrix must have an accurate value of depth Z. If that information is not available, an estimation of the $\mathbf{L}_\mathbf{x}$ matrix is used. Recommendations for the selection of this estimation $\widehat{\mathbf{L}_\mathbf{x}}$ are given in Espiau, Chaumette, and Rives (1992), Hutchinson et al. (1996) and Malis (2004).

4. Neural network Reinforcement Learning controller

In this section, the features of the neural network based Reinforcement Learning controller are described.

4.1. Q-learning algorithm

Q-learning represents a temporal difference Reinforcement Learning algorithm proposed by Watkins (1989) as an iterative method for finding the optimal policy of actions of the learning agents. The algorithm incorporates the Markov decision processes (Alpaydin, 2004) and utilizes the perceived states s, taken actions a, and the received reinforcement signals r to compute and update the Q-value or Q-function.

Q-learning algorithm enables the agent to estimate the value of a taken action by means of received reward r. The reward is designed in such a way that the desirable behavior is favored by assigning the positive numerical value to the action that guided the agent to a desired state. Also, the negative (or zero) reward signal is assigned for unwanted agent conduct. After taking an action and receiving the appropriate reward, the agent updates its current state Q-value.

An important characteristic of Q-learning is that it is an off-policy algorithm, meaning that the optimal mapping of states and actions is independent of the policy being followed. Also, it does not require a model of the environment which makes it a "model-free" algorithm. In Q-learning algorithm, a quality value is assigned to each visited state-action pair, and every time an action is taken the Q-value is updated according to

$$Q_{t+1}(s_t,a_t) \leftarrow Q_t(s_t,a_t) + \alpha_t\left[r(s_t,a_t) + \gamma \max_{a_{t+1}} Q_t(s_{t+1},a_{t+1}) - Q_t(s_t,a_t)\right], \quad (13)$$

where $r(s_t,a_t)$ is the immediate reward, and γ and α are the discount parameter and the learning rate, respectively, with values between 0 and 1.

This algorithm is proved to converge if the environment satisfies the Markov property and if all the state-action pairs are visited infinitely (Watkins & Dayan, 1992). When starting with interactions, an agent needs to choose actions more randomly so as to collect initial information of the environment. With a number of iterations conducted, the agent also must exploit its current knowledge in order to perform well. In literature, this problem is known as an exploration/exploitation problem. One way to balance exploration with exploitation is the ε-greedy policy

$$a_t = \begin{cases} a \in \arg\max_a Q(s,a) \text{ with probability } (1-\varepsilon) \\ \text{random with probability of } \varepsilon \end{cases}, \quad (14)$$

where $\varepsilon \in (0,1)$ is the exploration probability.

Traditional Q-learning algorithm is a discrete algorithm, so the Q-function is presented as table that can become very large in the case of discretization of the continuous values. In the case of online applications, algorithm requires that a certain amount of data is collected previously in order to enable the convergence of the action-value (Q) function. In order to make the Reinforcement Learning algorithm more suitable for real time robot control, the approximation of the Q-value using neural network is used in this study.

4.2. SARSA algorithm

SARSA is an on-policy temporal difference algorithm, similar to Q-learning (Sutton & Barto, 1998). A SARSA agent is able to interact with the environment and to update the policy based on the actions taken. Unlike the off-policy Q-learning algorithm which updates the estimate for the optimal policy while following explorative policy, the SARSA algorithm updates the same policy as it follows. The policy selects the action for the next step and uses this action in two ways: for updating the Q-value and for changing the pose of an agent. The SARSA update function is therefore defined as

$$Q_{t+1}(s_t,a_t) \leftarrow Q_t(s_t,a_t) + \alpha_t[r(s_t,a_t) + \gamma Q_t(s_{t+1},a_{t+1}) - Q_t(s_t,a_t)]. \quad (15)$$

As stated, the main difference between presented temporal difference algorithms is the important fact that, in SARSA, the on-policy update rule updates Q-value based on the action which is actually being taken in the next step. The main advantage of the algorithm is faster learning of the process, because of its optimization of the same policy that is being followed. The disadvantage is that the non-explorative policy used after learning is different from the policy which is optimized during the learning phase.

For the reasons of conciseness and clarity, and because of the similarity of the well known Reinforcement Learning algorithms, the neural network Q-learning algorithm with a database of representative learning samples is presented in the next section. The neural network SARSA learning is developed in the same manner, with regards to the previously mentioned differences.

4.3. Neural network Q-learning

In this study, the generalization problem is solved using a neural network with a database of learning samples, as presented in Perez (2003). For this purpose, a feedforward neural network with the backpropagation algorithm is used (Miljković & Aleksendrić, 2009). The inputs to the network are the scaled values of states and actions and the output is the one dimensional Q-value. Approximation of the Q-function in this way is known as direct Q-learning (Baird, 1995). This approach has two important phases. In the forward phase, a vector from the input layer is propagated through the network towards the output layer. Then, the error of the network is calculated in accordance with the following equation

$$Q(s_t,a_t) = r_t + \gamma \max_{a_{t+1}} Q(s_{t+1},a_{t+1}). \quad (16)$$

That error, along with the learning rate, is used to update the weights of the neural network in the backward phase with a signal propagated from the output to the input layer. At the beginning of the learning process, the neural network weights are initially set to small random values. In this study, the hyperbolic and linear activation functions are used for hidden and output layer, respectively.

Incorrect approximation of the nonlinear functions may occur if the two learning processes with mutual dependencies are active. This problem occurs because the neural network is used to update the Q-values and also for computing an error in accordance with the Q-learning algorithm. The solution to this problem is the definition of the database of representative learning samples which is used for updating the neural based Q-function. The content of the database is the current sample and the samples visited earlier. Each learning sample consists of the current state, the action, the new state, and the corresponding reward. The learning samples are added to the database in each learning iteration. After that, the most similar learning sample is replaced with a new one. The similarity is defined with a resemblance parameter, which directly influences the size of the database. The replacement rule is in accordance with the following equation

$$r_{par} = \sqrt{w_{e_1}(s_{new}-s_{old})^2 + w_{e_2}(a_{new}-a_{old})^2 + w_{e_3}(r_{new}-r_{old})^2}. \quad (17)$$

In this paper, a slightly different rule then the original one is introduced. A weighting coefficient for each component of the learning sample is defined, similar to Lin, Xie, Zhang, and Shen (2010). In this way, we can more accurately set the replacement technique for each learning sample in the database. The neural network Q-learning method adapted to the described case is presented in Algorithm 1.

Algorithm 1. Neural network Q-learning algorithm for Image Based robot control

Initialize: (1) weights of the neural network for $Q(s,a)$; (2) $r(s,a)$ for any s, a; (3) γ, α, ε, r_{par}
Repeat:
Observe the new world state s depending on the feature position on the image
Select an action according to Eq. (14)
Take action and observe the new world state s according to the new position of the feature on the image plane
Calculate the reward $r(s,a)$ and update the $Q(s,a)$ as stated in Eq. (13) (forward NN phase)
Update the database with the new learning sample in accordance with the resemblance parameter r_{par}
Train the NN with the entire database of learning samples (backward NN phase)
Until no more interaction

5. Intelligent visual control for robot manipulators

An intelligent hybrid visual servo controller for a robot manipulator is presented in this section. It consists of two independent steps: one with the neural network Reinforcement Learning controller and the other with the switching scheme between the neural network Reinforcement Learning and the traditional Image Based Visual Servo control. By introducing control steps the entire control process is decoupled into two phases for the visual servoing task. Firstly, the error between the feature state in the current image plane and the feature state in the prerecorded target image is calculated. Then, a correction in the robot pose for aligning the base with the object of interest is conducted using the neural network Reinforcement Learning controller. The end of the manipulator's centering motion represents the end of the first step. In the second step, the approaching behavior of the robot is carried out by using switching based control which depends on the feature position in the image plane (i.e. the Reinforcement Learning world state). The proposed control scheme is shown in Fig. 2.

To further elucidate the developed intelligent controller, flowchart presented in Fig. 3 is explained in details. Starting from an arbitrarily pose in the structured environment, the robot manipulator is to move to the desired pose as indicated by the prerecorded target image, taken at its desired position and orientation. Firstly, image in the current pose is acquired and the feature points are extracted. Then, the current world state is obtained regarding the position of the feature in the image. If the feature is in the first area of interest, the approaching (second) step is conducted. In the other case (feature is not in the first area of interest) the correction in the robot pose is carried out by choosing an optimal action with the neural network Reinforcement Learning controller. After the new world state is determined, the reward signal is calculated and the Q-value update is conducted (forward NN phase). Then, the training of the neural network is done with the entire updated database. The first control step is terminated when the feature visits the first desired area. After this, the second approaching control step with the second independent desired area starts immediately. Here, the distance between the current world state of the feature and second desired area is investigated. If the feature is within the area of interest, Image Based Visual Servo controller is active. This includes the estimation of the image Jacobian and error vector estimation, as indicated in Fig. 3. In the other case (feature is not in the second area of interest) the robot pose correction is carried out by using neural network Reinforcement Learning controller in the same manner as described previously. Logically, the optimal action

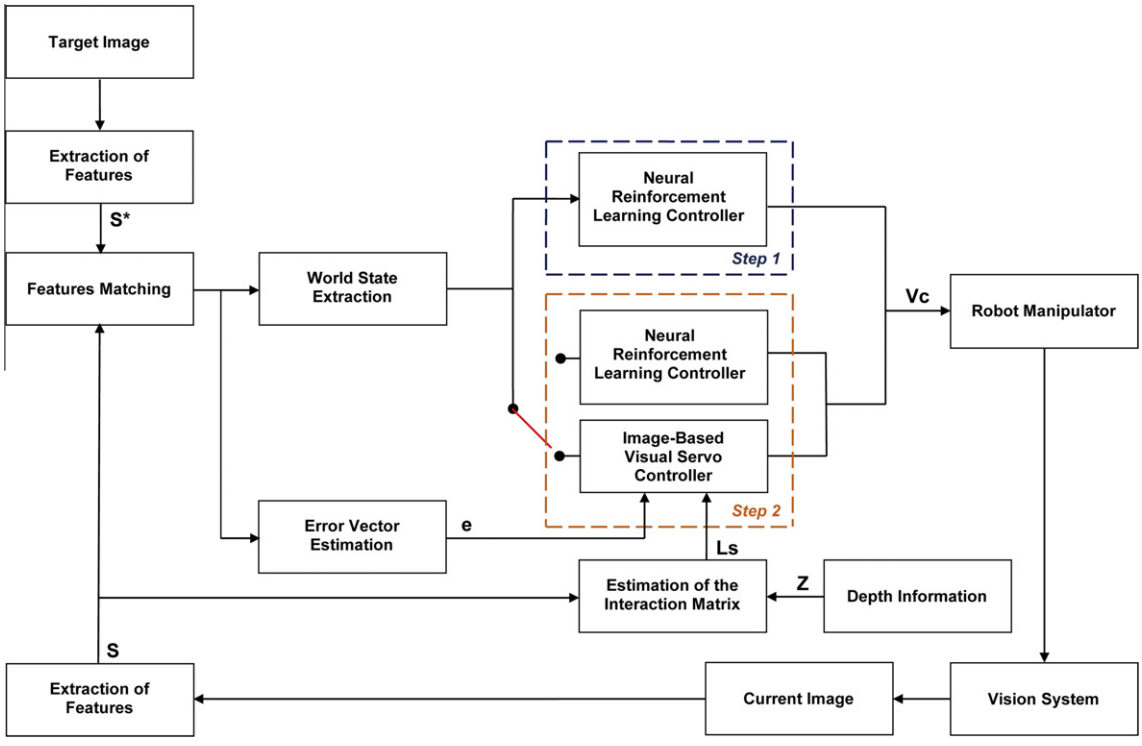

Fig. 2. Proposed intelligent hybrid control scheme.

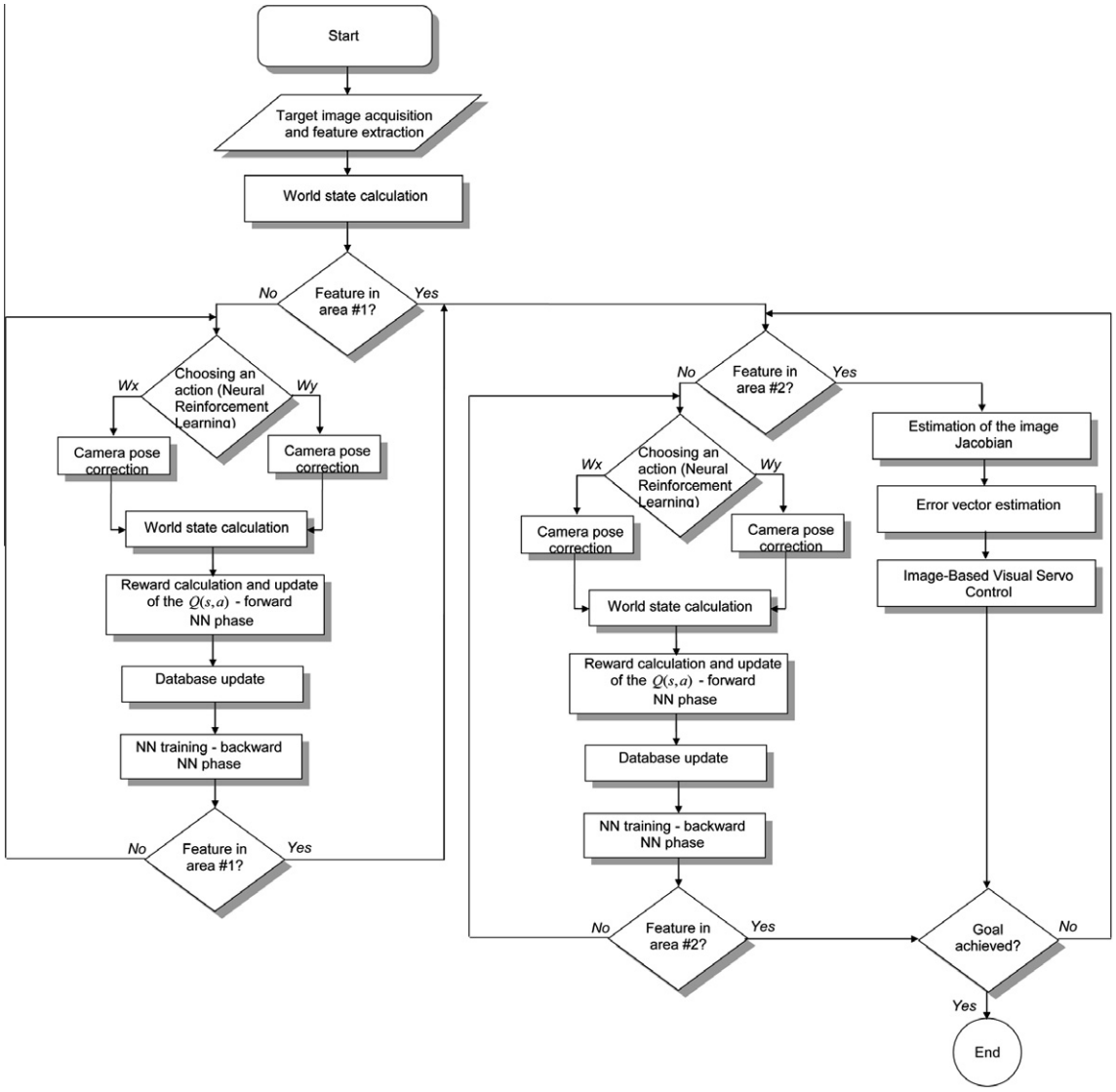

Fig. 3. Flowchart of the intelligent controller.

of the neural network Reinforcement Learning controller tends to correct the robot motion in order to move the feature closer to the area. Depending on the feature position in the second step, one of the described controllers is active. Finally, the robot motion is finished if the control goal is achieved, i.e. if the norm of the image features is smaller then the previously defined value (error tolerance set by the designer).

In Fig. 4 the discrete grid defining the world states with the assigned desired areas for both steps is presented. From Fig. 4 one can notice that the image plane consists of a discrete grid world of 32×24 cells. Each cell has a length of 20 pixels and represents one state for an image feature. In every control loop, an image is grabbed from the camera and the coordinates of the feature in the grid world are determined. Depending on the feature position in the second step one of the aforementioned controllers is activated. As mentioned, if the feature is within the area of interest, the IBVS controller is employed. Otherwise, a correction in robot pose is conducted using the Reinforcement Learning controller. In this way, the visibility of visual features during the entire robot motion is ensured. At the same time, the preferred path of the robot system is maintained for the whole visual servoing task. Also, due to the learning capabilities in both steps, the hybrid system can adapt to the changing conditions in the environment and improve the behavior in continuity and online.

Only two actions of the neural network Reinforcement Learning controller are defined. In the simulations, the camera orientation velocities w_x and w_y are predefined to be 1 deg/s. For the real experiments, the camera linear velocities v_x and v_y correspond to the values of 1 cm/s. After the robot takes an action in accordance with the ε-greedy strategy, the reward is assigned in dependence of the visual feature position in the image plane. The reward is defined in accordance with the following equation

$$r = \begin{cases} +100, & \text{if cur\_state} \in \text{des\_area} \\ +30, & \text{if dist(cur\_state, des\_area)} < \text{dist(prev\_state, des\_area)} \\ -30, & \text{if dist(cur\_state, des\_area)} > \text{dist(prev\_state, des\_area)} \\ 0, & \text{if dist(cur\_state, des\_area)} == \text{dist(prev\_state, des\_area)} \end{cases}, \tag{18}$$

where 'dist' represents a shortest distance between world states and desired areas.

As it is obvious from Eq. (18), the reinforcement signal rewards the actions that push the visual feature towards the desired area and punishes the opposite motion of the point in the image plane.

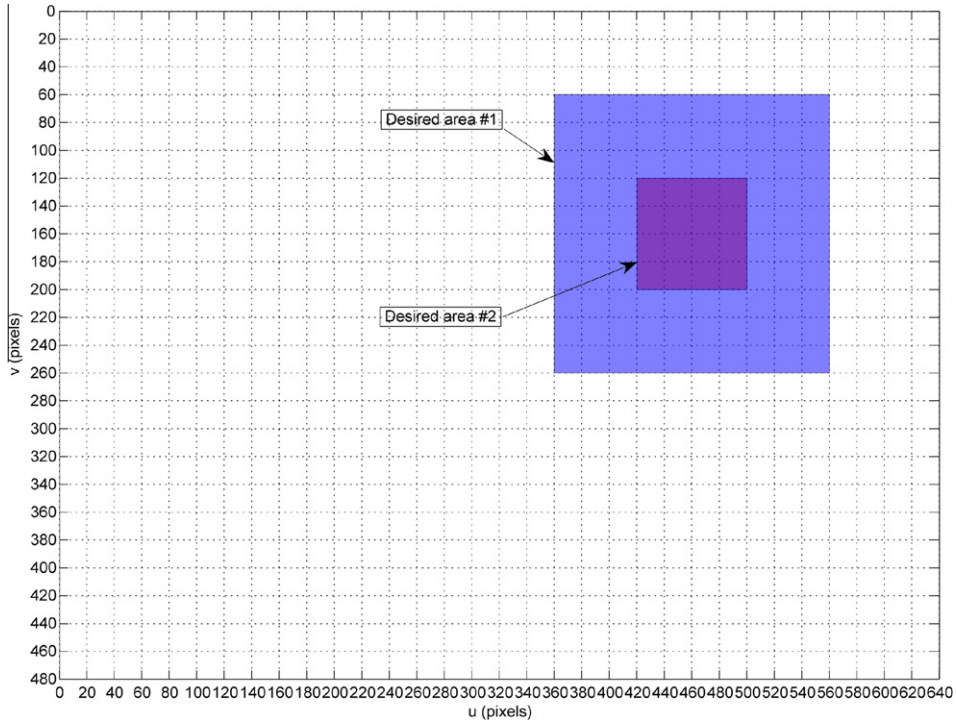

Fig. 4. Areas of interest for visual feature in the image.

After the reward is assigned, an update of the Q-value is carried using Eq. (13) or Eq. (15) depending on the algorithm used. Next, the new learning sample is added to the database and the one with the highest resemblance degree is replaced (see Eq. (17)). Finally, the loop of the neural network Reinforcement Learning controller ends after the neural network is trained with the learning samples in the entire database. In the following sections, through the various simulations and real world experiments we prove the robustness of the proposed intelligent system regarding calibration error, modeling error, and image noise. The comparison with the traditional Image Based Visual Servo control is given in addition.

6. Simulation results

In this section, simulation results in Matlab are provided to evaluate the performance of the proposed approach. The simulated data consist of four 3D points projected into the camera's image plane in each iteration of the control loop. Because the stationary pattern of the points is considered, the reference feature vector $\mathbf{s}^*$ does not change over time. Therefore, $\mathbf{s}^*$ can be calculated before the main control loop of the simulation. On the other hand, feature vector $\mathbf{s}$ is not constant due to the camera motion and is obtained in each time instant. With calculated $\mathbf{s}^*$ and $\mathbf{s}$, the input to the controller is defined as an error vector in the form $\mathbf{e}(t) = \mathbf{s} - \mathbf{s}^*$. It is important to note that the size of the error vector is not constant. It depends on which controller is active: $\mathbf{e}$ has the size of $[2 \times 1]$ (i.e. u and v values of one feature point) in the case of the neural network Reinforcement Learning controller, or $[2 \times 4]$ (i.e. u and v values of four feature points) for the IBVS controller.

The network with the same architecture is used in all simulations for the Reinforcement Learning controller. A feedforward multilayer neural network with the backpropagation algorithm (Miljković & Aleksendrić, 2009) and with hyperbolic and linear activation functions for hidden and output layers (respectively) is developed. For the IBVS controller, the interaction matrix and its inverse have been calculated as previously mentioned. The Vision System block (Fig. 2) consists of a perspective projection model and camera internal parameters, and is used for acquiring an image. The depth in simulations is obtained through the calculation of the camera's pose with reference to the stationary pattern. For both steps of the intelligent controller, the desired areas regarding the image feature are defined in the image plane. The size of images in the simulations is 640×480 pixels.

Camera initial and target poses in Cartesian space are presented in Fig. 5. One can notice that the axis of camera frames and the axis of world frame are parallel and in the same direction. Also, the points in the stationary pattern are presented with the reference number beside. This description of the scene setup is the same for all conducted simulations.

In this work, three simulation studies are given. The first study represents the case in which the control approach is tested with all features projected into the image plane in the initial robot pose. In other words, the camera's field of view constraint is eliminated in the starting pose. Also, this case investigates the behavior of the

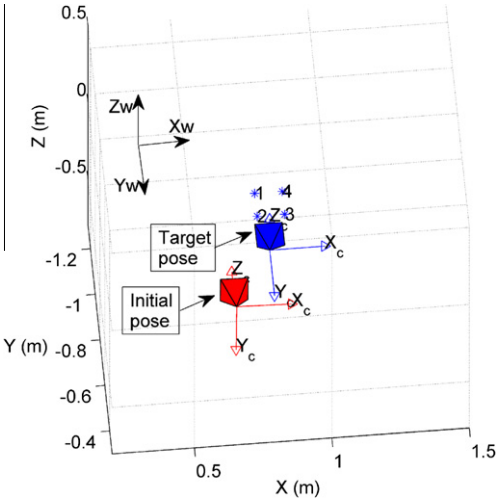

Fig. 5. Initial and target camera pose in Cartesian space.

intelligent controller with respect to the image noise. The second study treats the field of view problem in the starting robot pose along with calibration and modeling error of the camera. The third simulation study compares proposed intelligent hybrid control approach with the traditional Image Based Visual Servoing.

6.1. First simulation study

As mentioned, in this section simulation results regarding camera's field of view problem in the initial pose are presented. Two cases are described: an ideal case with no disturbances and the case of the camera motion regarding the white image noise. In both cases, initial and target camera poses are the same (Fig. 5). The desired areas defined in the same manner in both simulations, as showed in Fig. 6(a). The setup data in both cases are defined as presented in Table 1.

6.1.1. Ideal case

The systems response in an ideal case is investigated here. Correct camera calibration parameters, true depth information and zero image noise are set in this simulation. The image plane consisting of feature projections in the initial and target camera poses is presented in Fig. 6(b). The neural network used for approximation of the Q-value has 3 inputs (state vector of size $[2 \times 1]$ and the action value, all scaled from 0 to 1), 15 nods in the hidden layer, and one output (Q-value).

In Fig. 7 changes in camera pose and in camera Cartesian velocity are presented. Fig. 7(a) gives the results of the neural network Q-learning controller, while Fig. 7(b) depicts the effect of the hybrid controller with the neural network SARSA learning. Comparing Fig. 7(a) and Table 1 one can observe that the final position and orientation of the camera correspond to the desired values. In Fig. 7(b), the Cartesian velocity of the camera is presented. Since the first desired area is large, step 1 of the controller ends at approximately $t = 6$ s. As mentioned earlier, action space is designed in a way that one action corresponds to either w_x or w_y angular velocity of ±1 deg/s. The beginning of the second step is obvious from the lower part of Fig. 7(b), since angular velocity is different from zero. Note that because of the small second desired area, the neural network Reinforcement Learning controller is active at the beginning of the second step. From Fig. 7 we can conclude that the Cartesian velocity completely corresponds to the changes in camera poses, meaning that both neural network Reinforcement Learning algorithms successfully converged. In other words, neural network Q-learning and neural network SARSA recommended the same optimal action in the same robot pose regarding identical simulation conditions.

6.1.2. Camera behavior due to the image noise

In this section, the same initial and target camera poses as in the previous one are used (Table 1). The desired areas in both steps of the intelligent controller are defined in the same way as in previous study. Due to the image noise, the features in the initial camera pose have different values when compared with earlier simulation. The case with the large standard deviation of $\sigma = 5$ pixels is investigated in order to prove the robustness of the proposed approach. Feature projections into the image plane in the initial and target camera poses are presented in Fig. 8.

The camera velocity vectors for this simulation are given in Fig. 9(a). Angular velocities w_x and w_y in the lower part of Fig. 9(a) indicate the activation periods of the neural network Q-learning controller. The angular velocity suggests the correction in camera orientation in order to keep the visual feature in the desired area. Upper part of Fig. 9(b) shows pixel evolution during the simulation run by using neural network SARSA controller. The difference between the final and the target feature position is due to the satisfactory final value of the norm of the error vector. In the lower part of Fig. 9(b) the switching parameter for this setup is presented. This parameter δ is of value 1 if the neural network Reinforcement Learning controller is active; otherwise, the parameter δ is equal to 0. It is obvious that the neural network SARSA controller is much more active compared to the earlier case. Due to the pixel noise the IBVS controller can not ensure that the feature projection in the current pose will always be in the smaller second desired area. As in the previous case, the behavior of the neural network Q-learning and the neural network SARSA learning is the same, i.e. the convergence of the algorithm is achieved. Finally,

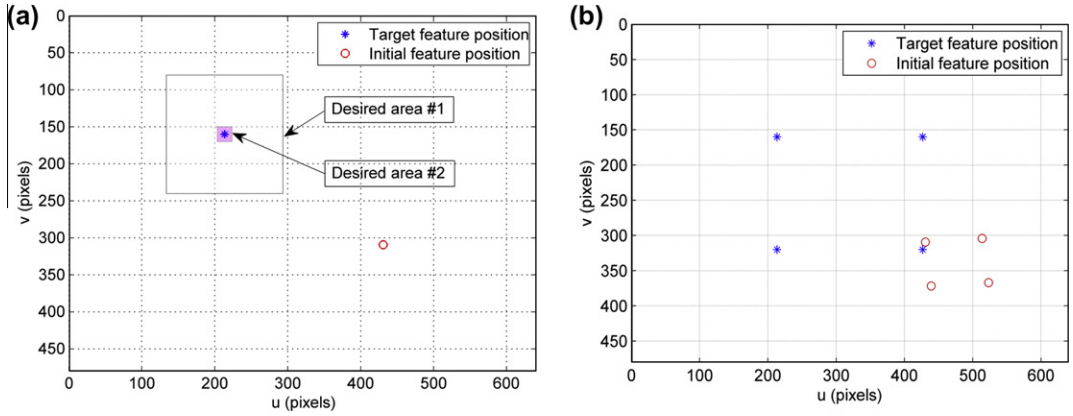

Fig. 6. Camera initial setup in ideal case: (a) desired areas for both steps and (b) projections of features in initial and target camera pose.

Table 1
Initial configuration for first simulation study.

Initial position [m]			Initial orientation [deg]			Desired position [m]			Desired orientation [deg]			λ Parameter (Eq. (5))	NN database size
X	*Y*	*Z*	*R*	*P*	*Y*	*X*	*Y*	*Z*	*R*	*P*	*Y*		
0.688	−0.7	−0.5	5	−5	5	0.8	−0.6	0	0	0	0	0.008	550

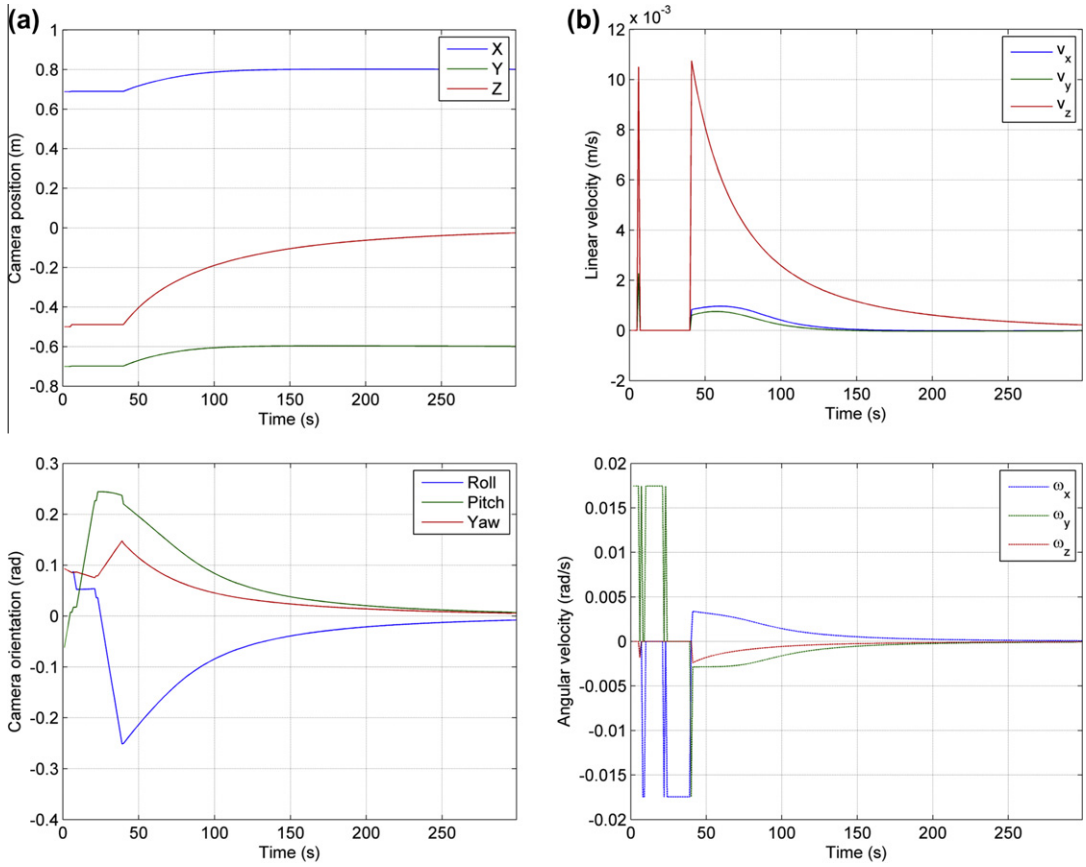

Fig. 7. Ideal case: (a) camera position and orientation by using neural network Q-learning controller and (b) camera Cartesian velocity by using neural network SARSA controller.

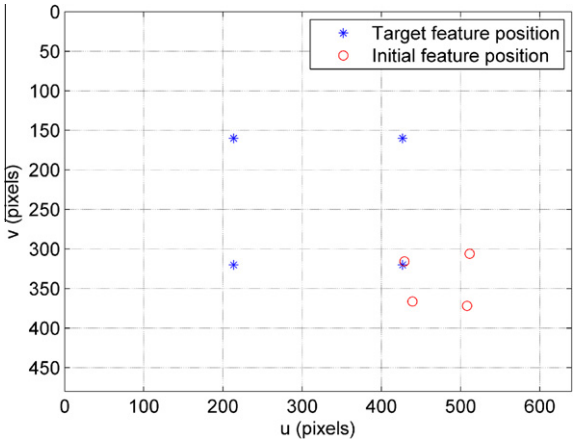

Fig. 8. Projections of features in initial and target camera pose in the case of image noise of $\sigma = 5$.

from Fig. 9 it can be concluded that camera stayed on the preferred path, with small position and orientation error at the end of the motion.

6.2. Second simulation study

In this study, the behavior of the intelligent control system is tested in case when less than 3 points are available at the camera's starting pose. For this setup, the neural network Reinforcement Learning controller is necessary to be active first since it is known that, in general, a minimum of three feature points are needed to control the position and orientation (6 DOF) of the camera in 3D space (for eye in hand system) (Chaumette & Hutchinson, 2006; Kragic & Christensen, 2002). Furthermore, this section investigates robustness of the proposed approach in terms of calibration and modeling errors. In both cases, the initial and target poses of the camera are the same, Fig. 10(a). In Fig. 10(b), projections of the features into the image plane in both poses are shown. The parameters for both simulation cases are given in Table 2.

6.2.1. Calibration error

Here, the influence of the calibration parameters on the system is investigated. With conditions $\hat{f}_u > 0$ and $\hat{f}_v > 0$, intrinsic camera parameters are corrupted with error of $\%5f_u$, $\%-5f_v$, $\%10u_0$, $\%-10v_0$ (see Ceren & Altuğ, 2012). The desired areas in this case are identical as in previous study. Like in previous cases, Fig. 11(a) shows simulation results with neural network Q-learning controller, while the camera performance with the neural network SARSA controller is shown in Fig. 11(b).

One can clearly see from Fig. 11(a) that in this case the camera follows the desired trajectory. Moreover, the final error in the position and orientation of the camera is minimal. Evolution of the features in the image plane is presented in the upper part of Fig. 11(b). Activation periods of switching parameter are given in the lower part of Fig. 11(b). As shown in the figures, the convergence of the system in this case is not very sensitive to change in calibration parameters. Much more intensive employment of the neural network Reinforcement Learning controller was obvious in the previous case with image noise. However, the preferred camera path is not achievable without the employment of the neural network Reinforcement Learning controller in the second control step.

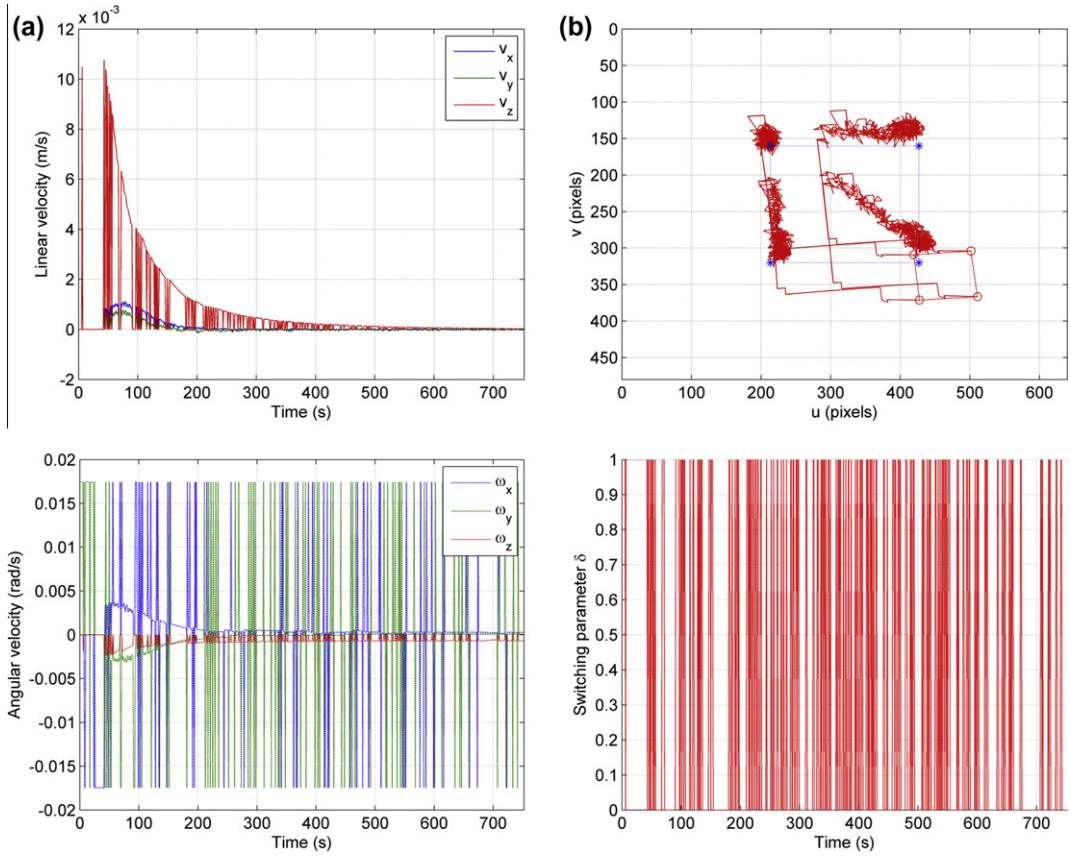

Fig. 9. Image noise of $\sigma = 5$: (a) camera Cartesian velocity by using neural network Q-learning controller and (b) feature evolution and switching parameter by using neural network SARSA controller.

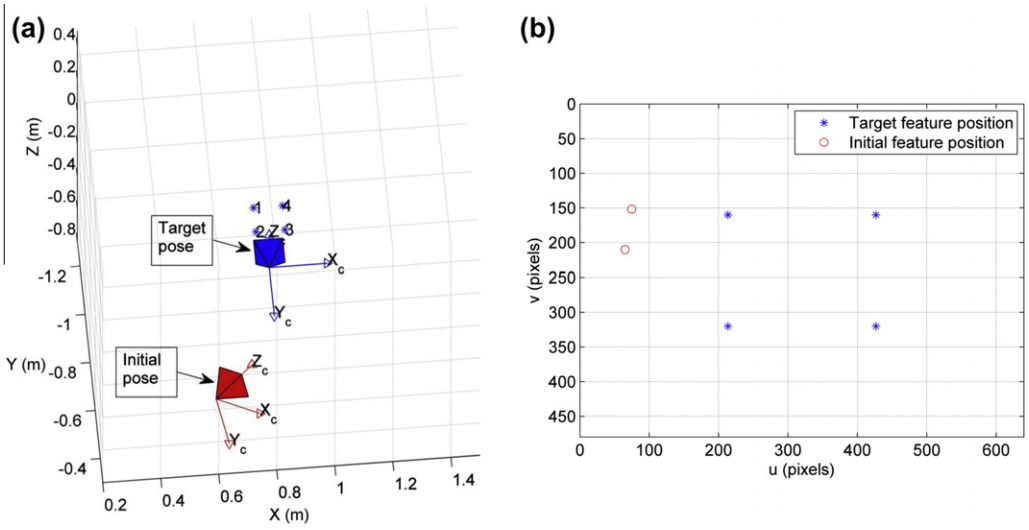

Fig. 10. Second simulation study: (a) initial and target camera pose and (b) projections of features into the image plane in both poses.

Table 2
Initial configuration for second simulation study.

Initial position [m]			Initial orientation [deg]			Desired position [m]			Desired orientation [deg]			λ Parameter (Eq. (5))	NN database size
X	Y	Z	R	P	Y	X	Y	Z	R	P	Y		
0.6	−0.41	−0.5	5	40	−5	0.8	−0.6	0	0	0	0	0.008	550

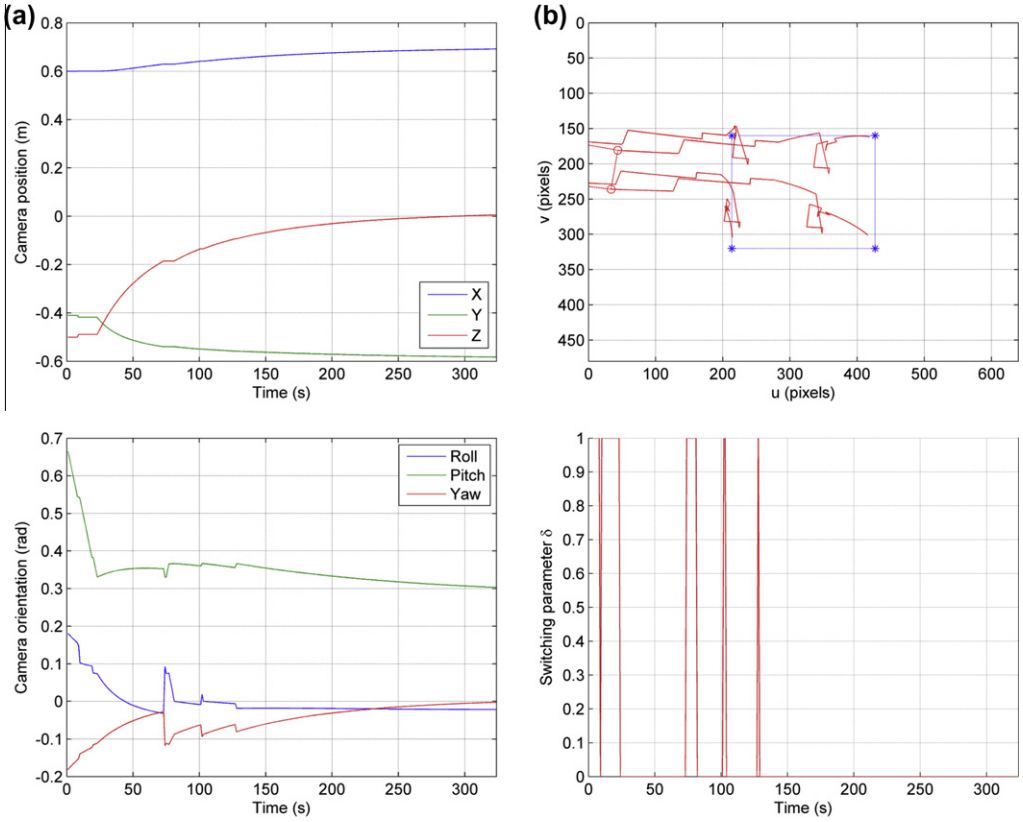

Fig. 11. Calibration error: (a) camera position and orientation by using neural network Q-learning controller and (b) feature evolution and switching parameter by using neural network SARSA controller.

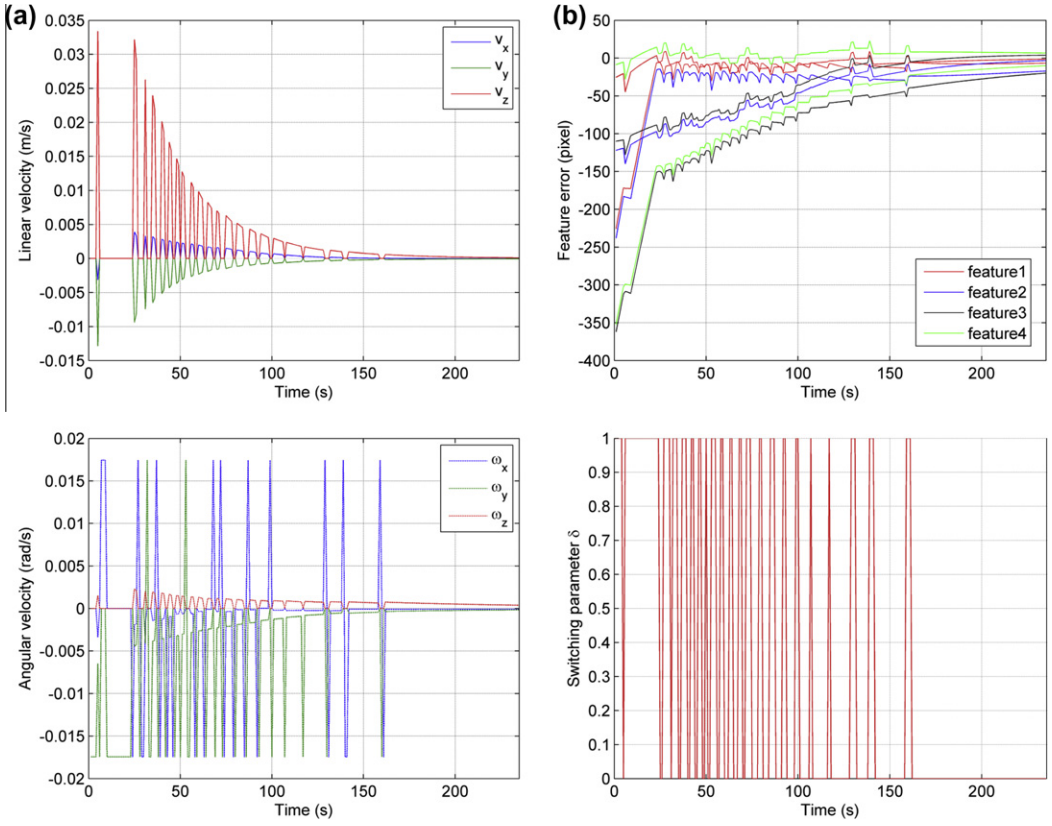

Fig. 12. Modeling error of $Z = 3Z$: (a) camera Cartesian velocity by using neural network Q-learning controller and (b) feature error and switching parameter by using neural network SARSA controller.

6.2.2. Modeling error

In this section, the simulation results regarding modeling error are presented. The influence of the corrupted distance to (i.e. the depth of) each 3D point is investigated here. As previously mentioned, the depth in simulation is estimated from the pose of the camera relative to the stationary pattern consisting of four 3D points. The analysis was conducted in the case of $Z = 3Z$, meaning that the estimated depth is 3 times larger then actual one. The desired areas are the same as in the previous section. Fig. 12(a) presents camera behavior with the employment of neural network Q-learning controller, while Fig. 12(b) gives the results when the neural network SARSA controller is used.

From Fig. 12(a) one can conclude that the correction in camera orientation is performed frequently, in order to keep the visual feature in the desired area. The neural network Q-learning controller is active even after the 150th iteration. Results given in the upper part of Fig. 12(b) confirms that the error in the image features decreases as expected. Comparing the lower part of Fig. 12(a) and (b), it can be concluded that the neural network SARSA learning and neural network Q-learning are active in the same time instants. As well as in the previous simulations, the final iteration in Fig. 12 indicates a desired steady state of the camera.

6.3. Third simulation study – comparison with the traditional Image Based Visual Servo control

Finally, the comparison between the performance of the developed intelligent controller and the traditional Image Based Visual Servoing is given. Since the IBVS control can not be applied unless a minimum of three feature points are available in the starting camera pose (for the control of 6 DOF, as in our case), the same initial conditions as in the first simulation study (Table 1) are used here. In order to validate the proposed approach, an image noise of $\sigma = 10$, a modeling error of $Z = 0.3Z$ and a $\%5f_u$, $\%-5f_v$, $\%10u_0$, $\%-10v_0$ calibration error is employed. Since both neural network Reinforcement Learning controllers showed same performance in previous simulations, neural network Q-learning is used here. The sizes of the desired feature areas are the same as in earlier described simulation studies.

In Fig. 13(a) the camera motion in 3D space and a corresponding feature evolution in the image plane for the intelligent controller is given. Fig. 13(b) depicts the results with the Image Based Visual Servo controller. Clearly, because of the employed corrupted information, the pose of the camera in the final iteration is somewhat different from the target pose. The camera poses in the same time instants in both control approaches are presented in Fig. 13. It is obvious from the upper parts of the figure that the path of the camera is much shorter in the case of the intelligent control. This is the direct consequence of the developed two phase control scheme and a small feature area of interest in the second switching control step. The neural network Reinforcement Learning enables such camera motion in which the correction in the camera orientation is conducted whenever the feature is outside of the desired area. This is particularly apparent from the lower part of Fig. 13(a), since the position of the image features in the case of intelligent control indicate the preferred shortest path of the camera. These adjustments in camera motion may be crucial from the aspect of desired camera path in real unpredicted conditions. In this way, by using

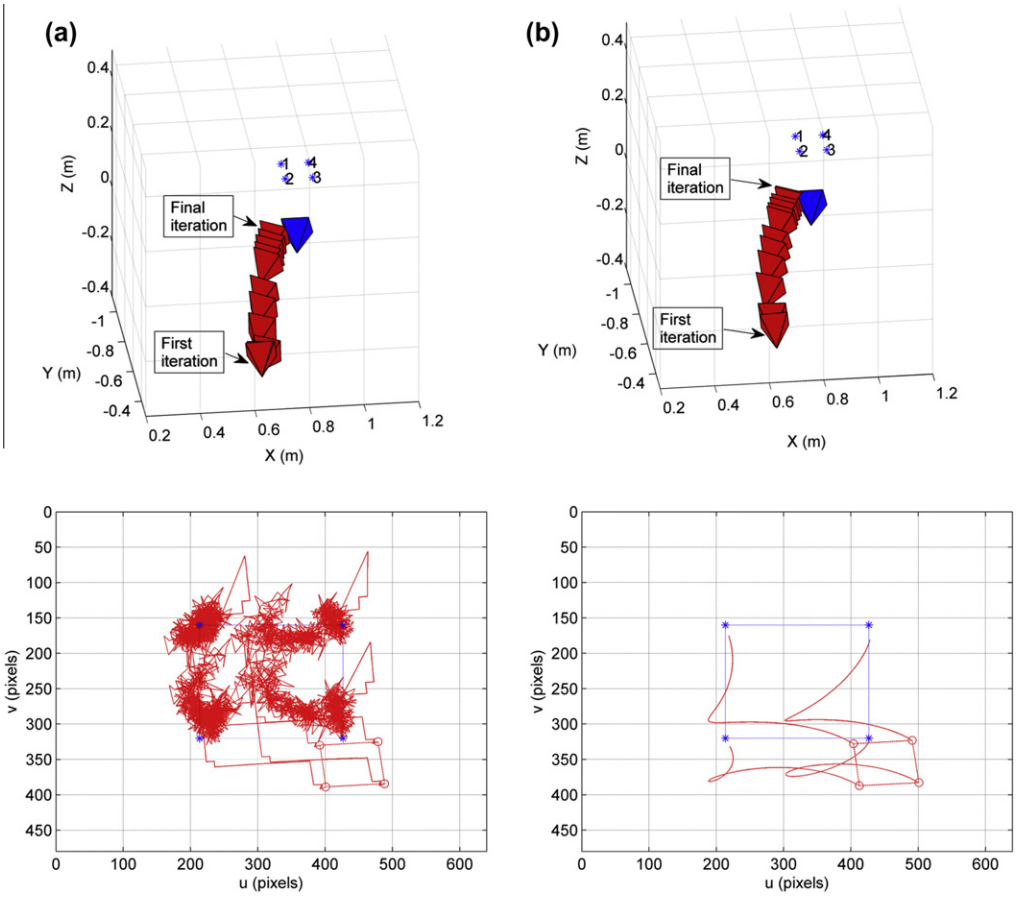

Fig. 13. Comparison between the intelligent hybrid controller and the traditional Image Based Visual Servoing: (a) camera position and orientation (3D space) and feature evolution by using neural network Q-learning and (b) camera position and orientation (3D space) and feature evolution by using IBVS.

Table 3
Norm of the error in features at various control iterations – comparison between the proposed intelligent control and traditional Image Based Visual Servoing.

Iteration	1	10	50	100	150	200	350	400	500	600	Final
Proposed hybrid control approach	403.7609	273.9805	213.1674	215.5603	207.8240	202.9533	110.3929	94.1564	69.0146	80.4814	28.2320
Traditional Image Based Control	403.7609	308.0754	244.9577	273.1588	260.7672	235.5450	164.3518	146.0785	116.3632	93.6269	29.9702

the intelligent hybrid controller, the norm of the error is decreased faster and the motion of the camera is completed earlier. The final pose of the camera in the case of intelligent control is obtained after 843 control iterations, while the 1189 iterations are needed for positioning the camera in the IBVS case. The error norm of the image features for both controllers (in iterations as presented in Fig. 13) is stated in Table 3.

7. Experimental evaluation

In order to evaluate the intelligent control scheme developed in this paper, the experiments on a real robot manipulator are carried out. Because the simulation results proved robustness in terms of calibration errors and image noise we used a low cost image acquiring sensor. Moreover, since both Reinforcement Learning controllers showed same optimal performance, the neural network Q-learning controller is employed in the experiments. The robotic system used for experimental validation consists of a 6 DOF robot *NeuroArm Manipulator System* from *NeuroRobotics* and a low cost camera (Fig. 14). The control and data processing are conducted with an AMD Athlon II X4 630 2.8 GHz processor desktop computer with 3 GBs RAM on Windows XP. The image processing is facilitated with black blob features on the scene. In the experiments, two types of black blob patterns were selected to simplify and speed up the camera performance in real time (Fig. 14). Captured low resolution greyscale images are in the size of 177 $\times$ 144 pixels. The extraction and matching of the image points is conducted by using the SURF algorithm (Bay, Ess, Tuytelaars, & Gool, 2008).

The neural network used here is the same as in the simulation. A multilayer feedforward network with backpropagation algorithm and with the same architecture and activation functions is employed. Also, the size of the database is exact. As explained previously, in each iteration of the control loop the new learning sample in the database replaces the old one with the highest resemblance. Then, the neural network is trained with all the samples in the database using backpropagation algorithm. It is obvious that the value of the resemblance parameter controls the size of the database and thus, consequently, affects the computational cost of the control loops. The desired areas are defined in the opposite manner regarding Fig. 6(a). This means that the smaller one is used in the first step (the size of the areas is the same). This is because we want to ensure the highest accuracy of the robot position before the approaching behavior.

In that way, two experiments are proposed and presented as follows. The first experiment relates to the case when more than 3 visual features are projected into the image plane in the starting robot pose. The second one investigates the behavior of the robot regarding the field of view problem in the starting robot pose (only one feature available). Because of the placement of the camera (Fig. 14), only the change in the system position is tested in the experiments. Nevertheless, the presented results clearly point out the applicability of the proposed solution. Also, it is important to mention that because of the satisfactory results of the simulation in case of modeling error, the depth in experiments is adopted to be constant. Experimental results showed that this approximation has no influence on the convergence of the proposed system.

7.1. First experiment

In this case, the action space for the neural network Q-learning controller consists of the predefined values for two linear velocities, v_x and v_y. As shown in Fig. 15(a), the linear velocities are defined as 1.5 cm/s and $v_y$and 1 cm/s for v_x and v_y, respectively. Therefore, one of these actions is used in each iteration, depending on the position of the feature in the image and the approximated Q value. Fig. 15(a) shows that only the v_y velocity is employed during the first step. At the start of the second step, both the v_x and v_y are alternately active. From Fig. 15(b) one can notice that the feature error decreases in each time instant. Unlike the previous cases where the feature errors relating to all 3D points are given, the error of one feature in the image plane is shown here. Also, the activation of the neural network Q-learning controller is noticeable towards the end of the experiment, Fig. 15(a).

The initial, target and final image are given in Fig. 16. A somewhat larger difference in the final and target image is due to the predefined parameter regarding the tolerance of the error norm in the final iteration. For this experiment, this parameter is set to be 10 pixels, see Fig. 15(b). From the obtained experimental results

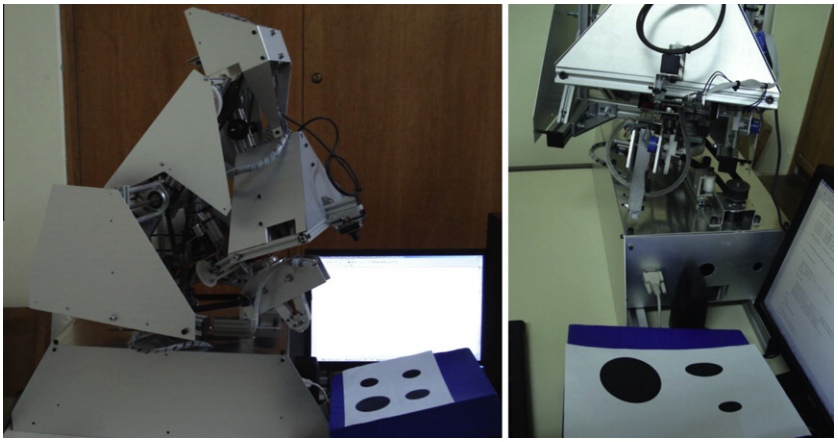

Fig. 14. Experimental setup.

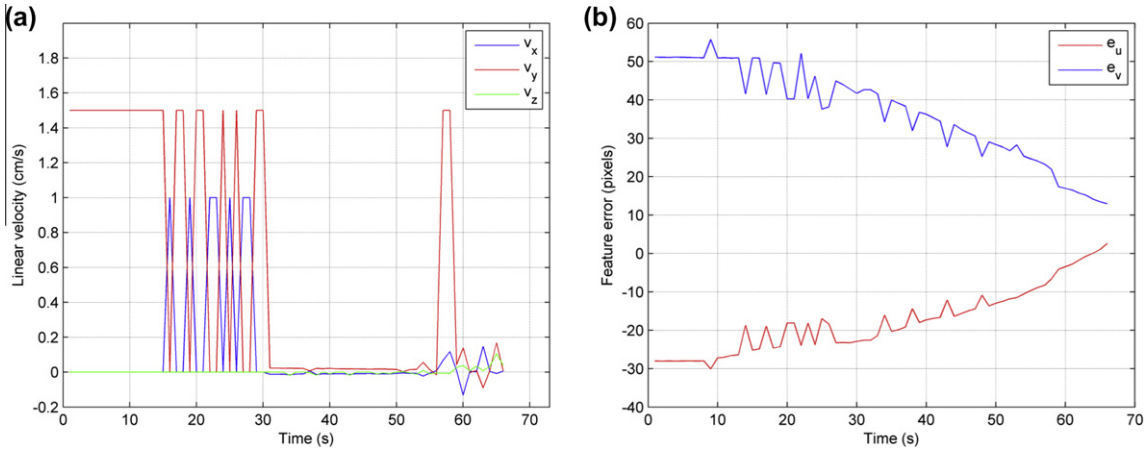

Fig. 15. First experiment: (a) camera linear velocity and (b) feature error.

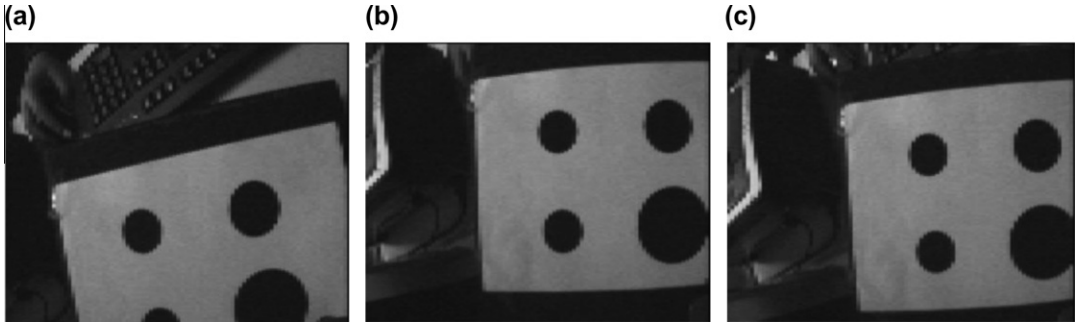

Fig. 16. Camera images from the first experiment: (a) initial, (b) target and (c) final.

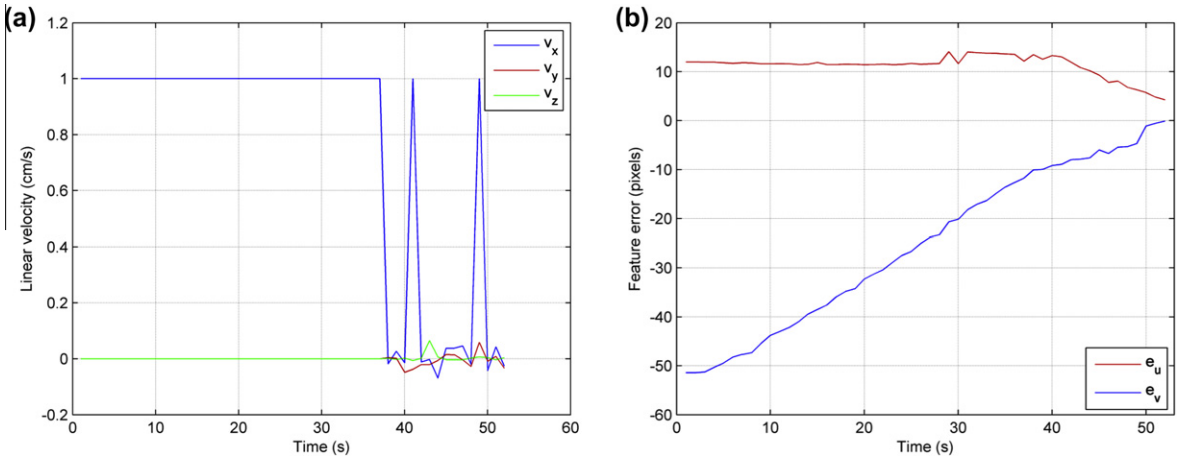

Fig. 17. Second experiment: (a) camera linear velocity and (b) feature error.

it is obvious that the hybrid controller proved to be robust to the real time conditions.

7.2. Second experiment

Behavior of the proposed control system regarding the field of view problem is investigated in this section. As previously stated, in starting robot pose only one feature in the image plane is available. Fig. 17(a) denotes the linear velocity in the experiment. The value of the v_x indicate that the neural network Q-learning controller is active longer compared to the previous case. Also, the correction of the robot position is engaged two times in the second step. Similarly to the previous experiment, the decreasing trend of the feature error is shown in Fig. 17(b).

Fig. 18 depicts the initial, target and final image, respectively. The difference between the target and final image is much smaller compared with the first experiment. This is due to the smaller final error norm tolerance, see Fig. 17(b) (being 5 pixels in this experiment).

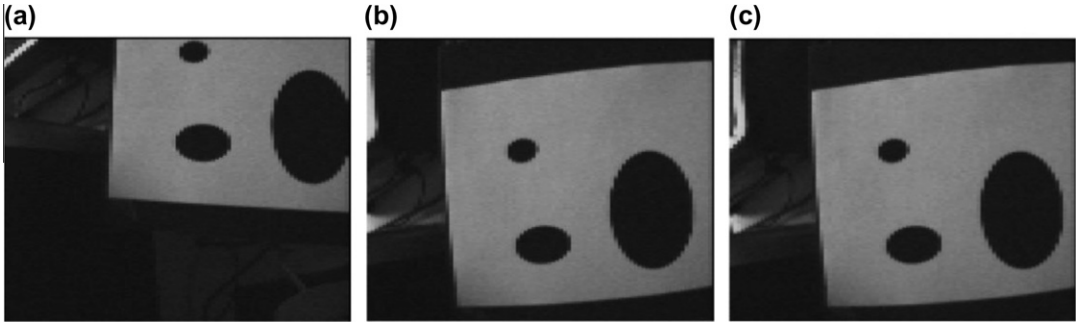

Fig. 18. Camera images from the second experiment: (a) initial, (b) target and (c) final.

Thus, the final pose of the camera is as expected. These results evidence the potential of the proposed intelligent Image Based controller in spite of the integration of low cost sensor suite.

8. Conclusions

This paper studied the development and performance evaluation of a novel intelligent controller for visual control of a robot manipulator. A direct mapping from the image space to the actuator command is developed by using two different temporal difference algorithms (Q-learning and SARSA) coupled with neural networks. In order to speed up the convergence of the algorithms, a database of representative learning samples is employed in the hybrid control scheme. Because the training of the neural network includes all the samples in the database, the Q-value is updated several times in each interaction with the environment. The experimental results show that an intelligent controller is able to select an optimal action despite challenging conditions such as the presence of the calibration error, modeling error, and image noise.

The control task is divided into two steps in order to ensure the existence of visual features within the field of view of the camera. In the first step, the neural network Reinforcement Learning controller enables the centering behavior of the robot. Secondly, a controller that switches between the traditional Image Based Visual Servo control and the neural network Reinforcement Learning is employed for approaching behavior of the manipulator. The correction in robot motion is conducted with a definition of two independent image point areas of interests for two control phases. By separating the visual servoing task into two steps with machine learning capabilities, robustness of the entire robot motion regarding unpredictable conditions in real time is achieved.

Various simulations are developed in Matlab in order to prove the robustness of the developed hybrid system regarding calibration error, modeling error and image noise. In addition, the comparison with the traditional Image Based Visual Servoing is presented. The convergence is ensured in every task, with the very good resulting performance considering the challenging simulated conditions. Finally, the experiments are conducted on a real robot manipulator with a low cost vision system in order to prove the effectiveness of the proposed approach. The results show that the proposed method can provide a high accuracy of a manipulator positioning in a situation when the low resolution image is used.

Acknowledgement

This work is supported by the Serbian Government – the Ministry of Education, Science and Technological Development – through the projects TR35004 and TR35006.

References

Alpaydin, E. (2004). *Introduction to machine learning*. MIT Press.
Asada, M., Noda, S., Tawaratsumida, S., & Hosoda, K. (1996). Purposive behavior acquisition on a real robot by vision-based reinforcement learning. *Machine Learning, 23*, 279–303.
Baird, L. (1995). Residual algorithms: Reinforcement learning with function approximation. In *Machine learning: 12th international conference, San Francisco*.
Basri, R., Rivlin, E., & Shimshoni, I. (1999). Visual homing: Surfing on the epipoles. *International Journal of Computer Vision, 33*(2), 117–137.
Bay, H., Ess, A., Tuytelaars, T., & Gool, L. V. (2008). SURF: Speeded up robust features. *Computer Vision & Image Understanding, 110*(3), 346–359.
Busquets, D., de Mantaras, R. L., Sierra, C., & Ditterich, T. G. (2002). Reinforcement learning for landmark-based robot navigation. In *Proceedings of the international joint conference on autonomous agents and multiagent systems (AAMAS 2002)* (pp. 841–843).
Ceren, Z., & Altuğ, E. (2012). Image based and hybrid visual servo control of an unmanned aerial vehicle. *Journal of Intelligent & Robotic Systems, 65*(1–4), 325–344.
Chaumette, F., & Hutchinson, S. (2006). Visual servo control. Part I: Basic approaches. *IEEE Robotics and Automation Magazine, 13*(4), 82–90.
Chesi, G., Hashimoto, K., Prattichizzo, D., & Vicino, A. (2004). Keeping features in the field of view in eye-in-hand visual servoing: A switching approach. *IEEE Transactions on Robotics, 20*(5), 908–913.
Chesi, G., & Hung, Y. S. (2007). Global path-planning for constrained and optimal visual servoing. *IEEE Transactions on Robotics, 23*(5), 1050–1060.
Corke, P. I., & Hutchinson, S. A. (2001). A new partitioned approach to Image Based Visual servo control. *IEEE Transactions on Robotics and Automation, 17*(4), 507–515.
Cowan, N. J., & Koditschek, D. E. (1999). Planar image based visual servoing as a navigation problem. In *Proceedings of the 1999 IEEE international conference on robotics and automation (ICRA 1999)* (pp. 611–617).
Distante, C., Anglani, A., & Taurisano, F. (2000). Target reaching by using visual information and Q-learning controllers. *Autonomous Robots, 9*, 41–50.
Espiau, B., Chaumette, F., & Rives, P. (1992). A new approach to visual servoing in robotics. *IEEE Transactions on Robotics and Automation, 8*(3), 313–326.
Fomena, R. T., Omar, T., & Chaumette, F. (2011). Distance-based and orientation-based visual servoing from three points. *IEEE Transactions on Robotics, 27*(2), 266–273.
Garcia-Aracil, N., Perez-Vidal, C., Sabater, J. M., Morales, R., & Badesa, F. J. (2011). Robust and cooperative Image Based Visual servoing system using a redundant architecture. *Sensors, 11*(12), 11885–11900.
Gaskett, C. (2002). *Q-Learning for robot control*. PhD thesis, The Australian National University.
Gaskett, C., Fletcher, L., & Zelinsky, A. (2000). Reinforcement learning for a vision based mobile robot. In *Proceedings of the 2000 IEEE/RSJ international conference on intelligent robots and systems (IROS 2000)* (pp. 403–409).
Hafner, R., & Riedmiller, M. (2003). Reinforcement learning on an omnidirectional mobile robot. In *Proceedings of the 2003 IEEE/RSJ international conference on intelligent robots and systems (IROS 2003)* (pp. 418–423).
Hamel, T., & Mahony, R. (2002). Visual servoing of an under-actuated dynamic rigid-body system: An Image Based approach. *IEEE Transactions on Robotics and Automation, 18*(2), 187–198.
Hutchinson, S., Hager, G., & Corke, P. (1996). A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation, 12*, 651–670.
Jiang, C., & Sheng, Z. (2009). Case-based reinforcement learning for dynamic inventory control in a multi-agent supply-chain system. *Expert Systems with Applications, 36*, 6520–6526.
Khan, S. G., Herrmann, G., Lewis, F. L., Pipe, T., & Melhuish, C. (2012). Reinforcement learning and optimal adaptive control: An overview and implementation examples. *Annual Reviews in Control, 36*, 42–59.
Kragic, D., & Christensen, H. I. (2002). *Survey on visual servoing for manipulation*. Technical report ISRN KTH/NA/P-02/01-SE, Royal Institute of Technology, Stockholm.

Lin, L., Xie, H., Zhang, D., & Shen, L. (2010). Supervised neural Q_learning based motion control for bionic underwater robots. *Journal of Bionic Engineering, 7*(Suppl), S177–S184.

Ma, Y., Kosecka, J., & Sastry, S. S. (1999). Vision guided navigation for a nonholonomic mobile robot. *IEEE Transactions on Robotics and Automation, 15*(3), 521–536.

Malis, E. (2004). Improving vision-based control using efficient second-order minimization techniques. In *Proceedings of the 2004 IEEE international conference on robotics and automation (ICRA 2004)* (pp. 1843–1848).

Mariottini, G. L., Oriolo, G., & Prattichizzo, D. (2007). Image Based Visual servoing for nonholonomic mobile robots using epipolar geometry. *IEEE Transactions on Robotics, 23*(1), 87–100.

Martínez-Marín, T., & Duckett, T. (2005). Fast reinforcement learning for vision-guided mobile robots. In *Proceedings of the 2005 IEEE international conference on robotics and automation (ICRA 2005)* (pp. 4170–4175).

Mezouar, Y., & Chaumette, F. (2002). Path planning for robust Image Based control. *IEEE Transactions on Robotics and Automation, 18*(4), 534–549.

Miljković, Z., & Aleksendrić, D. (2009). *Artificial neural networks – Solved examples with theoretical background* (in Serbian). University of Belgrade, Faculty of Mechanical Engineering, Belgrade.

Miljković, Z., Vuković, N., Mitić, M., & Babić, B. (in press). New Hybrid vision-based control approach for automated guided vehicles. *The International Journal of Advanced Manufacturing Technology*. http://dx.doi.org/10.1007/s00170-012-4321-y.

Mitić, M., Miljković, Z., & Babić, B. (2011). Empirical control system development for intelligent mobile robot based on the elements of the reinforcement machine learning and axiomatic design theory. *FME Transactions, 39*(1), 1–8.

Perez, M. C. (2003). *A proposal of a behavior based control architecture with reinforcement learning for an autonomous underwater robot*. PhD thesis, University of Girona.

Remazeilles, A., & Chaumette, F. (2007). Image Based robot navigation from an image memory. *Robotics and Autonomous Systems, 55*(4), 345–356.

Sahba, F., Tizhoosh, H. R., & Salama, M. M. M. A. (2008). A reinforcement agent for object segmentation in ultrasound images. *Expert Systems with Applications, 35*, 772–780.

Schramm, F., & Morel, G. (2006). Ensuring visibility in calibration-free path planning for Image Based Visual servoing. *IEEE Transactions on Robotics, 22*(4), 848–854.

Shin, M., Ryu, K., & Jung, M. (2012). Reinforcement learning approach to goal-regulation in a self-evolutionary manufacturing system. *Expert Systems with Applications, 39*, 8736–8743.

Song, K.-T., & Chu, T.-S. (1998). Reinforcement learning and its application to force control of an industrial robot. *Control Engineering Practice, 6*(1), 37–44.

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning: An introduction*. Cambridge: MIT Press.

Takahashi, Y., Takeda, M., & Asada, M. (1999). Continuous valued Q-learning for vision-guided behaviour acquisition. In *Proceedings of the 1999 IEEE international conference on multisensor fusion and integration for intelligent systems* (pp. 255–260).

Wang, Y., Lang, H. X., & de Silva, C. W. (2010). A hybrid visual servo controller for robust grasping by wheeled mobile robots. *IEEE/ASME Transactions on Mechatronics, 15*(5), 757–769.

Watkins, C. (1989). *Learning from delayed rewards*. PhD thesis, King's College, Cambridge.

Watkins, C., & Dayan, P. (1992). Technical note: Q-learning. *Machine Learning, 8*(3–4), 279–292.