Intelligent Predictive Control of a Six-Joint Robotic Manipulator Towards a Vision Based Application

Raşit Köker
International University of Sarajevo
Faculty of Engineering and Natural Sciences
Sarajevo/Bosnia and Herzegovina
E-mail: rkoker@ius.edu.ba
Tel: 00 387 62 693 631

Abstract
Recently, the automation systems with vision based robots have taken a great deal of attention because of the advances in computer technology and expanding demand for such techniques in the industry. In this study, an intelligent robot control system was designed based on using Generalized Predictive Control (GPC) and Elman neural network to reduce the process time towards a vision based study. A six-joint robotic manipulator was firstly controlled by using GPC. A prepared simulation software was used in the implementation of GPC towards vision based application. Meanwhile, the control parameters are recorded in a file. Then, using this data set an Elman neural network was trained to control the robotic manipulator. On the other hand, an industrial vision system was designed and the control simulations are done according to the designed vision system. Image processing algorithms and GPC algorithm take too much process time that is why to reduce the process time of the control algorithm will be useful for an industrial application. The results of the robot control simulations and implemented image processing algorithms were given.

Keywords: Elman neural network, intelligent robot control, vision system, image processing, automation system

1 Introduction
Recently, automated picking and placing systems have received a great deal of attention in many industrial robotic applications. Similarly, much effort has been focused on automating various production systems in the industry by applying machine vision technology. The most important one is moving object gripping on a conveyor. Robotic manipulators have been also used extensively to perform the task of picking and placing. A robot can be defined as a multi degree freedom open loop chain of mechanical linkages and joints. These mechanisms driven by actuators are capable of moving an object in space from initial to final locations or along prescribed trajectories. Tracking accuracy is of prime concern in assembly
task. Hence, a robot controller should be able to achieve fast and at the same time accurate tracking and positioning control of the gripper. The dynamic equations of a mechanical manipulator are highly nonlinear and complex. To overcome these difficulties, several advanced adaptive control techniques for dynamic control of mechanical manipulators have been proposed (Ozsoy and Kazan, 1991; Liu, 1986). Generalized Predictive Control is a member of digital control methods called Model-Based Predictive Control (MBPC). Clarke and his friends presented the GPC algorithm in 1987. MBPC techniques have been analyzed and performed successfully in the control industries since the end of the 1970’s and continue to be used because of the fact that they can take into account real plant constraints in the real-time, systematically. GPC is used to control non-minimum phase plants, open-loop unstable plants and plants with variable or unknown dead time, and it is also robust with respect to modeling errors, over and under parameterization, and sensor noise (Soloway and Haley, 1996). GPC algorithm, which belongs to a class of digital control methods and known as Model Based Predictive Control, requires long computational time because of highly mathematical computations. Therefore, to reduce the process time, in other words, to avoid from the highly mathematical computational structure of GPC, a neural network based intelligent control system can be designed for very dynamical systems such as for robotic manipulators (Koker, 2006; Koker and Ferikoglu, 2004).

In this study, a 6-joint Stanford robotic manipulator was controlled based on a vision system. Koker previously presented two papers about the intelligent robot control using GPC and Neural networks (Koker, 2006; Koker and Ferikoglu, 2004). Here, an intelligent controller is designed for a vision based application. During the study, the robotic manipulator was firstly controlled by using GPC algorithm according to work area of the vision system. Meanwhile, the control parameters were recorded in a file to prepare a training set for the Elman neural network. Then, an Elman network controller was designed. The image processing algorithms and GPC algorithm are very time consuming because of highly mathematical computations. That is the reason why it is better to design a neural network controller, which can work faster and online.

2 Generalized Predictive Control Algorithm

Basically, GPC is the algorithm predicting $y(t)$ output throughout finite horizon and, control signals string $u(t)$ is computed using the minimization of a quadratic performance criterion according to the some assumptions interested in the control signals that implements appropriate response. To design the controller, the discrete time CARIMA (Controlled Auto-Regressive Integrated Moving Average) model is used given in Equation (1).

$$A(q^{-1}).y(t) = B(q^{-1}).u(t - 1).e(t)/\Delta$$

(1)

Here, $y(t)$ refers to the output of the process, $u(t)$ refers to its output, and $e(t)$ denotes an uncorrelated random noise. $A(q^{-1})$ and $B(q^{-1})$ are polynomials of $q^{-1}$ as the backward shift operator and $\Delta$ is computed as $\Delta = 1 - q^{-1}$. 
In the GPC algorithm, the control increment minimizes the cost function given below:

\[
J = \sum_{j=1}^{N_2} \left\{ [y(t+j) - w(t+j)]^T [y(t+j) - w(t+j)] \right\}
+ \sum_{j=1}^{N_u} \Delta u. (t+j-1). \Lambda(j). \Delta u. (t+j-1)
\]

where, \(\Lambda(j)\) and \(w\) refers to the weight vector in control increases and the reference trajectory, respectively. Additionally, \(N_1\) is minimum costing horizon, \(N_u\) is the control horizon, \(y\) is the predicted output and, is the control input weighting factor that is selected very small. Detailed computational structure of the GPC algorithm can be found in the references (Clarke et al., 1987a,b)

![Figure 1: The block diagram of the GPC for a robotic manipulator](image)

The block diagram of the GPC algorithm has been given in the Fig. (1). It has three components, which are controller, robotic manipulator and parameter predictor (Durmus et al., 2009). The parameter prediction process has been performed by using the recursive least-squares (RLS) algorithm (Ozsoy and Kazan, 1991). In Fig.1, \(u, y\) and \(yr\) refer to process input, process output and process output reference value, respectively. The computation of the torque values shown as \(u\) is the aim of the controller in this study (Koker, 2002).

3 Robot model used in the simulations

A six-joint robot model, which is known as Stanford robot and shown in 1, has been used in the simulation. The simulation software includes dynamics and kinematics equations for the given robot model. As it is well known, the dynamics of a robot arm can be expressed by the following general set of equations (Ozsoy and Kazan, 1991; Acosta et al., 1999) as given in Equation (3):

\[
u(t) = D(\Theta)\ddot{\Theta} + Q(\Theta, \dot{\Theta}) + G(\Theta)
\]

In this equation, \(\Theta, \dot{\Theta}\) and \(\ddot{\Theta}\) are m-dimensional vectors, showing the joint position, speed and acceleration, respectively. \(D(\Theta)\) is a \((m \times m)\) symmetric matrix
including the coefficients regarding the acceleration of the joints and effects of link inertia. $Q(\Theta, \dot{\Theta})$ is a $m$-dimensional vector representing the torque values because of the gravity. The $m$-dimensional vector $u(t)$ is the input of the system.

Figure 2: Stanford robotic manipulator (Fu et al., 1987).

Stanford robotic manipulator has been used in the vision based simulation studies. The cubic trajectory planning is used for trajectory planning. The sampling period has been selected, as 0.0020 s. and totally 500 steps has been performed during the simulations. The reference position and speed trajectories have been computed using the equations given below. Its equation, which has been used in the computation of reference position and speed values, is given in Equation (4) below.

$$\Theta_i(t) = \Theta_{i0} + \frac{3}{t_f^2}(\Theta_{if} - \Theta_{i0})t^2 - \frac{2}{t_f^3}(\Theta_{if} - \Theta_{i0})t^3 \quad (i = 1...n) \quad (4)$$

where; $t_f$ is the total simulation time, $\Theta_{if}$ is the final result of angular position for $i$th joint, $\Theta_{i0}$ is the starting position of $i$th joint, $t$ is the time and $n$ is the number of joints.

By the derivation process, the speed equation is obtained and presented in Equation (5)

$$\dot{\Theta}_i(t) = 2 \left(\frac{3}{t_f^2}(\Theta_{if} - \Theta_{i0})t\right) + 3 \left(\frac{2}{t_f^3}(\Theta_{if} - \Theta_{i0})t^2\right) \quad (i = 1...n) \quad (5)$$
4 Elman network controller design

In this part, the Elman network working online to control the Stanford robotic manipulator based on GPC algorithm is presented towards vision based application. The robotic manipulator is firstly controlled by using GPC algorithm for many different uniformly selected trajectories near the work area. The data set should be prepared very carefully especially for the vision system, otherwise to train a neural network for the all work volume of the robot will be very difficult and time consuming. The designed neural network has 24 inputs and 6 outputs. To obtain the torque value at time “t” as an output, the torque values at time $(t-1)$, $(t-2)$, and $y$ and $y_{references}$ at time $(t-1)$ are used in input stage as 24 elements. Payload variations are taken between 0 gram and 5000 gram in the simulations.

In off-line training of back propagation neural network, 8000 input and output vector sets are generated using robot control simulation software. 7000 of these are used as training set, and others are used in test. The number of neurons in the hidden layer was determined experimentally during the training process. At the learning, the number of neurons in hidden layer has been decided as 50. Error at the end of the learning is 0.0015791. The training process has been completed approximately in 3.850,000 iterations. After that off-line neural network training is finished, the neural network that works online is coded with obtained synaptic weights. The neural network toolbox of the Matlab program has been used in the training of network training. Conventional back propagation error-learning algorithm, which uses a threshold with a sigmoid activation function and gradient descent-learning algorithm, has been used. A sample obtained torque curve from the Elman network with an actual curve was given in Fig. 3 below. Additionally, the block diagram of the online implementation of the vision based intelligent robot control system was given in Fig. 4.

![Figure 3: Obtained torque curve from Elman network with actual torque curve.](image-url)
Figure 4: Block diagram of the implemented vision based robot control system.
Additionally, as it is seen on the Fig.4, the inverse kinematics solution was done by using a neural network. The inverse kinematics problem for a robotic manipulator is finding the required manipulator joint angles for a given desired end point position and orientation. In this study, the obtained target position by using image processing is in cartesian coordinate frame. The robotic manipulator needs angular joint and orientation information to reach target point. In this study, an Elman neural network with a sigmoid activation function was used in the inverse kinematics solution. Back propagation learning algorithm was used in the training. The designed neural network for the inverse kinematics solution has 12 inputs and 6 outputs. The inputs are target cartesian coordinate and orientation information for the end effector, and the outputs are six joint angle for the robotic manipulator. For the inverse kinematics problem solution, 6000 data was prepared by using Denavit-Hartenberg method. 4000 of them was used as training set, remaining part was used as test set. The learning was completed approximately in 4,200,000 iterations. The error at the end of learning computed by using mean square error was for training set 1,160758 and for test set 1,9782392.

5 Image Processing

In this section, the related image processing algorithms were given for the vision based robot control system. In the prepared image processing software, the care is taken of time due to its importance in the system to preserve the practicality.

5.1 Low-Level image processing

An experimental captured image is a 256 colored gray image. In prepared software, initial image size is reduced down to $256 \times 256$ pixels to decrease the processing time. A median filter is used to eliminate the undesirable effects due to the noise and other effects. Because of the sensitivity of moment invariants to the noise, filtering is important in this system. The illumination is provided to have an object image without shadow and reflection using two light sources. To automate the thresholding operation, we have used the method of Optimal Thresholding by Minimizing Within-Group Variance. This method is a reasonably good thresholding method for more uniformity, better shape of the object in the binary image and short processing time (Fu et al., 1987; Cokal et al., 1996; Sahoo and Soltani, 1988).

5.2 Intermediate level image processing

Edge detection can be categorized in intermediate level processing. This stage of the system aims to simplify the object image. Among the large number of edge detection algorithms, Sobel is used due to its popularity on computational simplicity (Laplante and Stoyenko, 1996; Koker et al., 2001). The centroid and area are computed by using edge map of image to prepare position information for the robotic manipulator. The samples of processed object images are given in Fig.5.
5.3 Estimation of the object position

In this study, the object area was used to understand whether the object entered totally to the work area on the conveyor or not. The area can be easily computed by using threshold image since it is the number of black pixels in the threshold image as given in Fig. 5 (b). The computation of the centroid is also important for the robot. The centroid is a good parameter for specifying the location of an object. The centroid information can be used as a target position for the end effector. Additionally, the centroid can also be used to understand whether the object is totally in the work area on the conveyor or not. The equations for the computation of the centroid and area have been given below in the Equations (6) and (7). In the Equation (6), $n$ is the total number of pixels in the image. It is the point having coordinates $x'$, $y'$ such that the sum of the square of the distance from it to all other points within the object is a minimum.

$$\text{Centroid } (x', y') = \left( \frac{1}{n} \sum_x x, \frac{1}{n} \sum_y y \right)$$ \hspace{1cm} (6)

$$\text{Area} = \sum_x \sum_y f(x, y)$$ \hspace{1cm} (7)

The calculation of the speed of the object is done by using centroid changes versus time. The position estimation is done by using this computed object speed moving on the conveyor. All measurements belonging to the designed vision system including a conveyor in the x,y,z cartesian space are prepared as references. Therefore, these references can be used to estimate the position of the object for capture process. This estimated position is also known as meeting point for robot’s end effector and object.
6 Hardware and Software Implementation

The implemented vision based robot control system has been given in Fig.6. The vision system is real-time experienced on a moving conveyor with a capture card and a CCD camera. The work area has been selected on the conveyor’s moving surface. At the beginning of the conveyor the image frames have been captured sequentially with intervals of 200 ms. By using these captured images the area of the object have been computed. The area changes have been compared to understand whether the whole object image is ready to be captured or not in the work area. When the whole object is at the work area main image have been captured to be processed. Firstly, centroid is computed to have information about object position. The velocity of the conveyor has been computed by using centroid changes. Using this velocity the position of object has been computed after a certain time approximately at the end of the conveyor. This position is an appointment point with robot end-effector. The z dimensions of objects have written to a file, and taken by using the prepared software. The robot control software should be coded to get this position information of object to join with centroid information to be used in the inverse kinematics problem solution.

![Figure 6: A view from the implemented vision based intelligent robot control system.](image)

The inverse kinematics problem solution has been carried out by training an Elman neural network to find the angular position information of the robotic manipulator for any given x,y,z cartesian position information. A limitation in this study is that third dimension of the objects (height) are accepted known and taken from a file to join with the x,y cartesian coordinate information.

7 Results and Discussion

A sample obtained speed and position results in the simulation studies have been given below graphically. Simulations have been done for both GPC and Elman network controller for the defined starting and final positions. The result of the Elman neural network controller has been compared with the traditional GPC controller.
Table 1: The initial and final speed and position information for each joints used in sample simulation

<table>
<thead>
<tr>
<th>Joint</th>
<th>Starting Angles (Radian)</th>
<th>Speeds</th>
<th>Final Angles (Radian)</th>
<th>Speeds</th>
</tr>
</thead>
<tbody>
<tr>
<td>Joint #1</td>
<td>0.52359</td>
<td>0</td>
<td>1.13446</td>
<td>0</td>
</tr>
<tr>
<td>Joint #2</td>
<td>0.43633</td>
<td>0</td>
<td>0.78539</td>
<td>0</td>
</tr>
<tr>
<td>Joint #3</td>
<td>0.20 m</td>
<td>0</td>
<td>0.45 m</td>
<td>0</td>
</tr>
<tr>
<td>Joint #4</td>
<td>0.69813</td>
<td>0</td>
<td>0.26179</td>
<td>0</td>
</tr>
<tr>
<td>Joint #5</td>
<td>0.52359</td>
<td>0</td>
<td>0.95993</td>
<td>0</td>
</tr>
<tr>
<td>Joint #6</td>
<td>0.95993</td>
<td>0</td>
<td>0.43633</td>
<td>0</td>
</tr>
</tbody>
</table>

As an example the results of the fifth joint is presented in the Figures 7-10. Figure 7 and 8 shows the position and speed results for the GPC algorithm. Figure 9 and 10 shows the position and speed curves for the designed intelligent controller. As it is clearly seen on the graphics given in Figure 7-10, the satisfactory results have been obtained. The Elman controller has some errors on the speed curves due to its structure. The Elman network is trying to approximate the obtained results from GPC since used in the training set. However, the Elman network controller will be very useful in some applications where the process time is very important.

![Figure 7: Reference and obtained position curves for the 5th joint obtained using GPC algorithm.](image-url)
Figure 8: Reference and obtained speed curves for the 5th joint obtained using GPC algorithm.

Figure 9: Reference and obtained position curves for the 5th joint obtained from Elman network controller.
In this paper, a vision based intelligent robot control system was presented. The vision applications have been experienced by using a prepared vision system including a camera, a capture card and a conveyor. The control studies were done based on using prepared simulation software. Elman neural network was used to design the controller based on training by using the prepared data during the implementation of traditional GPC algorithm. The aim of this study is to reduce the process time by avoiding very complex mathematical structure of the GPC algorithm. Neural networks can work online and fast. On the other hand, the process time is more important in case of vision based application because of the fact that image processing algorithms are also includes too many matrixes. Therefore, using Elman network controller with a vision system gives a great advantage about the reduction of the process time. This will give an advantage about the reduction in the cost of hardware equipments. However, the Elman network was only trained according to the defined work area to get better results, otherwise it is very difficult to train the Elman network to obtain an intelligent controller based GPC for all work volume of the robotic manipulator because of the huge data set.

Figure 10: Reference and obtained speed curves for the 5th joint obtained from Elman network controller.
References


Rasit Koker, Cemil OZ, and Abdullah Ferikoglu (2001), Development of a Vision Based Object Classification System For an Industrial Robotic Manipulator, in *The 8th IEEE Intern. Conf. on Electronics, Circuits and Systems*, Malta.


Can Ozsoy and Recep Kazan (1991), Self Tuning Predictive Controllers For Robotic Manipulators, Beograd, Yugoslavia, in *2nd Project Workshop On CIM and Robotics Applications*.

