robotics

MDPI

Article

Calibration of UR10 Robot Controller through Simple Auto-Tuning Approach

Cosmin Copot [1,*], Cristina Muresan [2], Clara-Mihaela Ionescu [3] and Steve Vanlanduit [1] and Robin De Keyser [3]

[1] Department of Electromechanics, University of Antwerp, Op3Mech, Groenenborgerlaan 171, 2020 Antwerp, Belgium; Steve.Vanlanduit@uantwerpen.be

[2] Department of Automation, Technical University of Cluj-Napoca, Memorandumului Street, No. 28, 400114 Cluj Napoca, Romania; Cristina.Muresan@aut.utcluj.ro

[3] Flanders Make, EEDT Group, Research Group on Dynamical Systems and Control (DySC), Department of Electrical Energy, Metals, Mechanical Constructions and Systems (EEMMeCS), Ghent University, Technologiepark 914, 9052 Ghent, Belgium; ClaraMihaela.Ionescu@ugent.be (C.-M.I.); Robain.DeKeyser@ugent.be (R.D.K.)

* Correspondence: cosmin.copot@uantwerpen.be

Received: 15 May 2018; Accepted: 3 July 2018; Published: 5 July 2018

Abstract: This paper presents a calibration approach of a manipulator robot controller using an auto-tuning technique. Since the industry requires machines to run with increasing speed and precision, an optimal controller is too demanding. Even though the robots make use of an internal controller, usually, this controller does not fulfill the user specification with respect to their applications. Therefore, in order to overcome the user requirements, an auto-tuning method based on a single sine test is employed to obtain the optimal parameters of the proportional–integral–derivative PID controller. This approach has been tested, validated and implemented on a UR10 robot. The experimental results revealed that the performances of the robot increased when the designed controller, using the auto-tuning technique, was employed.

Keywords: PID control; auto-tuning method; robot; calibration

1. Introduction

Manipulator robots can be described as automated electromechanical systems with flexible functionalities (i.e., programming according to the environmental conditions in which they operate). The main role of the manipulator robot is to help the human operators in fulfilling repetitive tasks with increased risk in an industrial environment. From the perspective of robot control, focus is put on the structure and functionality of the robot command unit. Based on the dynamic and geometrical model and the tasks that need to be performed, appropriate commands are established. In order to provide these commands to the actuators, the hardware and the software, it is required to use the feedback signals obtained from the sensory system unit [1]. Due to the complexity of the manipulator robot, the control architecture has a hierarchical structure. The upper level makes the decision with respect to the actions that need to be taken (e.g., simple test based on "if-then" actions), while the lower level carries out the control of the joints. The typical structure of the control system consists of a computer on the upper level and a system with one or more microcontrollers to drive the actuators of the joints on the lower level.

Mechatronic systems, in special robots, are very popular for control applications due to their interdisciplinary nature [2–4]. For linear mechatronic systems, the proportional-integral-derivative (PID) controller has been widely used given its simple structure and robustness [5]. Usually, the design of conventional integer-order PID controllers is based on the model of the system.

The manipulator robots are also used in medicine for precision surgery [6–8]. Surgery has become an extremely precise practice, and professionals often have to make movements in the order of hundreds of microns. The inherent limits of human dexterity are a constraint for new advances in this field. In this context, high-precision robotics are being developed and implemented for medical use [9,10]. From the observation of these devices, it is possible to acquire an idea of their versatility. Not only in medicine, but in the high-tech industry, as well, robotic automation is highly competitive and requires very high precision and better performance. Therefore, the performance specifications of control have also become extremely demanding.

In these real applications of the UR10 robot, the characteristics of the system may be subject to changes (e.g., the mass attached to the end effector, the speed of the joints (TCP respectively), the required precision, etc.). The controller, however, has to keep its performance constantly optimal. Usually, the robots already have an internal control loop, making the robot stable and performing quite well. Nevertheless, this controller cannot fulfill all the user specifications and does not perform optimally in all situations. Therefore, a new controller is employed in order to overcome the user requirements [11–14]. Hitherto, PID control is widely used in control engineering and industry. The most challenging step in employing PID controllers, for non-experts in control engineering, is the process of parameter tuning. Nowadays, the self-tuning PID digital controller provides much convenience in engineering [15,16]. Optimal control of a plant (in this particular application, the robot arm) is highly dependent on the plant behavior.

Throughout the years, several methods have been introduced [17–19] to improve the tuning principles proposed by Ziegler and Nichols. Astrom and Hagglund proposed an AMIGO (approximate M-constrained integral gain optimization) tuning rule for PI controllers [20], based on the ideas presented in [21], where the tuning of the controllers is done so as to maximize the integral gain subject to constraints on the maximum sensitivity. The same approach was later extended to PID controllers [22] and finally perfected by adding additional constraints [23] to make the tuning method suitable for a wide class of processes. Skogestad proposed a novel tuning rule based on the ideas behind IMC (internal model control) that have achieved widespread industrial acceptance [24]. These novel tuning rules, referred to as SIMC (Skogestad IMC), work well for both integrating and pure time delay processes, for both setpoint changes and load disturbances and are suitable for first-order or second-order time delay models.

In this paper, we propose simple and efficient calibration of the robot controller based on an auto-tuning procedure. The auto-tuning method attempts to design PID controllers that ensure a minimum phase margin PM = 45° and a minimum gain margin GM = 2. The method is based on defining a forbidden region in the Nyquist plane, using the performance specifications, and then searches for the optimal PID controller that ensures that the difference between the slope of the region border and the loop frequency response slope is minimum. This requirement leads to PID controllers that are robust to gain, as well as to phase and delay variations. Although the method has already been compared to various other popular PID tuning methods [25], its main features have not been fully revealed. To design the PID controller, a single sine test is applied to the robot to determine its frequency response and its frequency response slope [26], needed in the auto-tuning procedure. The original elements of this paper include, apart from a detailed description of the auto-tuning method, also the particularities of using it in a closed loop setting. Moreover, for the first time, the auto-tuning method is validated experimentally.

This paper is structured as follows: The next section gives a description of the robotic system used in this study. Section 3 discusses the automatic calibration of the robot using an auto-tuning approach. Section 4 presents the implementation and the corresponding results of the tuning procedure. The conclusion is formed in Section 5.

2. System Description

A robot manipulator, depicted in Figure 1, can be defined as a kinematic chain consisting of multiple rigid bodies, which are interconnected by joints. Every joint of the kinematic chain consists of one degree of freedom, which can be translational or rotational. In the work of Spong et al. [1], it is mentioned that a simple representation of a robot manipulator joint can be modeled as a spring with linear constant stiffness. Based on the Lagrangian formulation, the dynamics of a global manipulator robot with n revolute joints are given by [27]:

$$Q = M(q)\ddot{q} + C(q,\dot{q})\dot{q} + F(\dot{q}) + G(q) + J(q)^T\eta \tag{1}$$

where $q, \dot{q}, \ddot{q} \in \Re^n$ represent the joint variables (the generalized joint coordinates, the velocity and the acceleration), $M(q) \in \Re^{n\times n}$ is the inertia matrix, $C(q,\dot{q})$ is the $n \times 1$ vector of the Coriolis and centrifugal forces, $F(\dot{q})$ is the friction torque, $G(q)$ is the gravity force and Q is a vector representing the actuator forces associated with the joint coordinates q. The matrix $J(q)^T$ is the transpose of the Jacobian matrix of the robot, while η is the joint force vector applied at the robot end-effector.

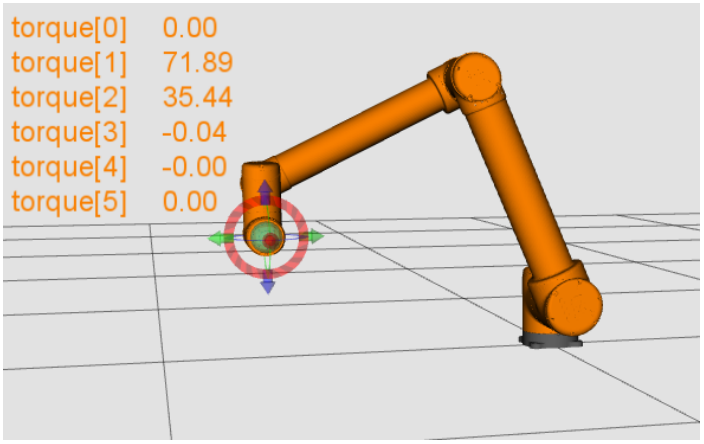


Figure 1. Schematic overview of the UR10 robot: CAD model.

The matrices M, C, F and G are composed of complex functions, which describe the kinematic link dynamics through a set of parameters $(\theta_j, d_j, a_j, \alpha_j)$, $j = 1, ..., n$, also known as Denavit–Hartenberg parameters.

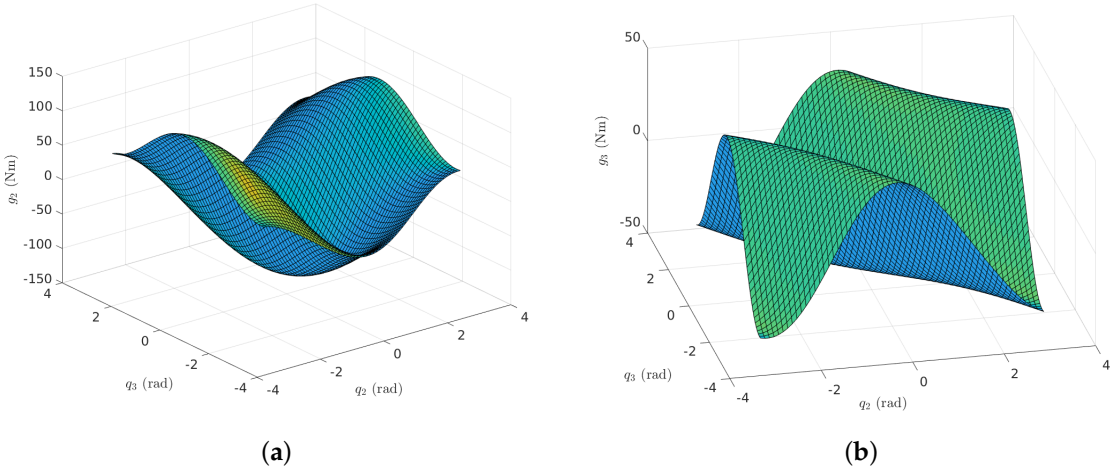
The considered UR10 robot studied in this paper has the following Denavit–Hartenberg (D-H) parameters; see Table 1.

Table 1. Denavit–Hartenberg (D-H) parameters.

Joint	θ	d	a	α
1	θ_1	0.12	0	$pi/2$
2	θ_2	0	-0.61	0
3	θ_3	0	-0.57	0
4	θ_4	0.16	0	$pi/2$
5	θ_5	0.11	0	$-pi/2$
6	θ_6	0.09	0	0

The highest variation in the robot dynamics is introduced by the gravity matrix and the inertia matrix. The gravity term is present even when the robot is stationary or moving slowly. The torque exerted on a joint due to gravity acting on the robot is strongly dependent on the position and orientation of the robot.

Due to the construction of the robot (three large joints and three small joints), we expect to have a larger variation introduced by the shoulder joint and elbow joint. For example, the gravity torque acting on the elbow joint is higher when the robot is in a position such that the elbow joint has to support the shoulder-lift joint and the wrist joint. To illustrate this aspect, we have investigated the variation of the gravity load for Joint 2 (shoulder-lift joint) and Joint 3 (elbow joint) with respect to the robot configuration. The obtained results are illustrated in Figure 2. As observed, Joint 2 has a larger variation ($\pm$100 Nm) compared to Joint 3 ($\pm$40 Nm). This analysis is important for robot design, as well as for control design in order to ensure a feasible control input for the motors.



(**a**) (**b**)

Figure 2. Gravity load variation of a UR10 robot; (**a**) joint 2 gravity load, $g_2(q_2, q_3)$ and (**b**) joint 3 gravity load, $g_3(q_2, q_3)$.

The gravity matrix is not the only term that varies with respect to the position of the robot. The other matrix that is highly dependent on the robot pose is the inertia matrix. By inspecting the elements of this matrix, we observe that it is symmetric with diagonal terms M_{jj} describing the inertia related to joint j. The other elements M_{ji} of the inertia matrix represent the coupling between joint j and joint i. In Figure 3, we can see the variation of the Joint 1 inertia with respect to the pose of the robot (here, Joint 2 and Joint 3 are varying between -180 and $+180$).

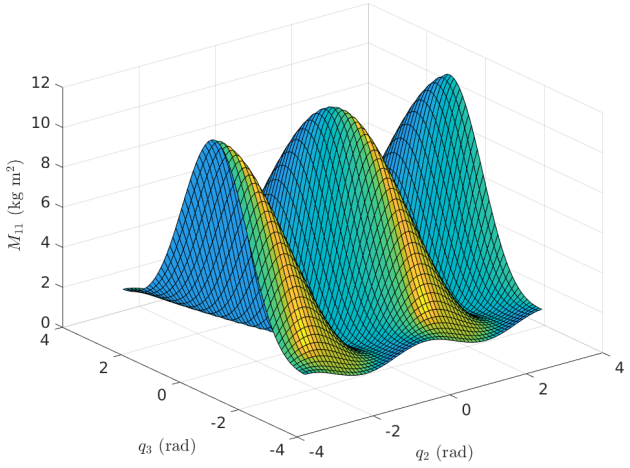


Figure 3. Inertia matrix variation (M_{11}) of a UR10 robot.

In most of the robotics applications, one needs to control the pose of the end-effector, which can be done through the direct kinematics formulation. In this research study, a six-degree of freedom manipulator robot is considered; thus, the direct kinematic equation is given as:

$$^{0}T_{6}(q) = ^{0}A_{1}(q_{1})^{1}A_{2}(q_{2})...^{5}A_{6}(q_{6}), \tag{2}$$

where $^{i-1}A_{i}(q_{i})$, $i = \overline{1,6}$ represent the homogeneous transformation matrices. The direct kinematics can be calculated via the Denavit–Hartenberg (D-H) formulation [28].

Starting from the assumption that the pose, the velocity and the acceleration are known and based on Equation (1), it is possible to calculate the required joint torques.

Given the pose configuration of the end-effector, the joint variables corresponding to this configuration can be calculated using the inverse kinematics formulation. The inverse kinematics is a very challenging problem since it is almost impossible to obtain a unique solution. In the literature, there are two approaches used to calculate the inverse kinematics: the analytical method and the numerical method [28]. Here, a numerical method was considered [27]. The existence of solutions is guaranteed only if the given end-effector position and orientation belong to the manipulator's dexterous workspace.

Nowadays, a large number of the industrial robots are driven by brushless servo motors. A schematic overview of a classical robot joint is illustrated in Figure 4.

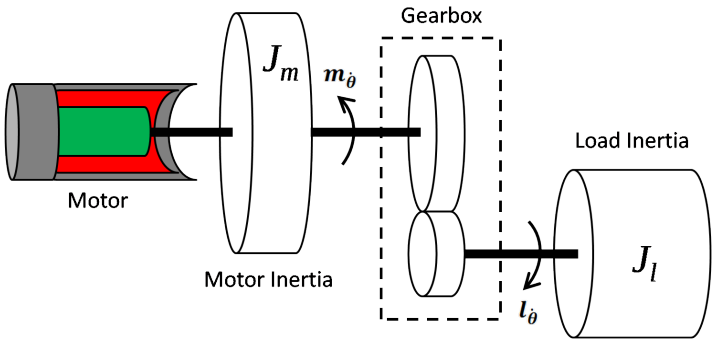


Figure 4. Schematic overview of a classical robot joint drivetrain.

Several studies in the area of the control of manipulator robots consider as control variables the Cartesian coordinates. However, in order to obtain the desired Cartesian trajectory for the end-effector attached to the robot, each joint axis must follow a specific trajectory. The common approach to control the robot joints is based on the decentralized techniques, which consider an independent controller for each joint. The industrial robot considered for our experiments is electrically actuated. If the robot joint drivetrain is driven by the current:

$$i = K_{a}u \tag{3}$$

then the torque generated by the motor is proportional to the current:

$$\tau = K_{m}i \tag{4}$$

where K_a is the transconductance of the amplifier, K_m is the motor torque constant and u is the applied control voltage. Based on this assumption, the dynamics of a motor attached to a joint j can be defined as:

$$J_{m}\dot{\omega} + B\omega + \tau_{c}(\omega) = K_{m}K_{a}u \tag{5}$$

with τ_c Coulomb's law of friction, B the viscous friction and J_m the total inertia seen by the motor for joint j, computed as:

$$J_m = J_{m_j} + \frac{1}{G_j^2} M_{jj}. \tag{6}$$

Since in real life, disturbance torques such as gravity and friction can act on the joints, the common approach where independent control systems are considered to control the robot joints is no longer efficient. The control performance decreases, leading to high overshoot and large steady-state error. A classical approach to overcome these shortcomings is to use a nested control structure composed of two loops: one outer loop and one inner loop, as depicted in Figure 5. The outer loop is used to control the position and to provide the velocity of the joints in order to minimize the position error, while the inner loop is used to minimize the error between the actual velocity of the joint and the velocity demanded by the outer loop.

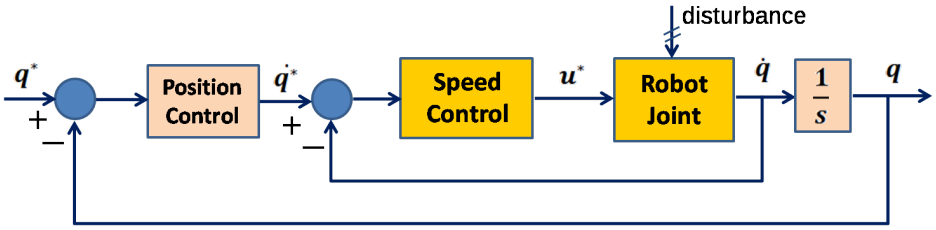


Figure 5. The structure of the nested control architecture.

For simplification purposes, if the Coulomb friction from Equation (5) is ignored, the equivalent Laplace transform of (5) is given by:

$$sJ\Omega(s) + B\Omega(s) = K_m K_a U(s) \tag{7}$$

where $\Omega(s)$ and $U(s)$ are the Laplace transform of their corresponding signal from time domain ω and u. Thus, the transfer function of a motor drivetrain attached to a robot joint can be written as:

$$\frac{\Omega(s)}{U(s)} = \frac{K_m K_a}{Js + B}. \tag{8}$$

Typically, a PI controller is used to drive the robot joint from the actual velocity to its demanded velocity. However, in this experiment, it is no need for good closed loop dynamics; therefore, a P controller has been selected (note that the P controller was used only for tests in order determine the process frequency response and process frequency response slope) for simplicity in the computations in the next section, and thus, we have:

$$u^* = K_p(\dot{q}^* - \dot{q}). \tag{9}$$

3. Robust Control Design

3.1. A Robust Method to Estimate the Process Frequency Response

The frequency response of a process at a specific frequency $\overline{\omega}$ can be written as:

$$P(j\overline{\omega}) = \frac{A_y}{A_u} e^{j\varphi_y} \tag{10}$$

where A_y stands for the amplitude of the output signal $y(t)$, A_u is the amplitude of the input signal $u(t) = A_\omega sin(\overline{\omega}t)$ and φ_y is the phase of the output signal $y(t)$. To determine experimentally the frequency response in (10), a robust procedure is needed to evaluate correctly, even in the presence of severe disturbances, the amplitude A_y and phase φ_y of the output signal $y(t)$. Such a procedure is based on the transfer function analyzer-discrete Fourier transform (TFA-DFT) [29,30].

For the process with the frequency response given in (10), the steady-state response can be written as:

$$y(t) = A_y sin(\overline{\omega}t + \varphi_y) + b + n(t) \tag{11}$$

where $n(t)$ is the stochastic disturbance with zero mean value and b is a non-zero bias term ($b \neq 0$ in the case of a nonzero average disturbance).

Consider the multiplication of the measured output $y(t)$ by sine and cosine functions of the frequency of the system input $\overline{\omega}$ and then integrated over the measurement period $T_m = k\frac{2\pi}{\overline{\omega}}$, with k an integer value [29,30]. Then, the following results are obtained:

$$y_s = \int_0^{T_m} y(t) sin(\overline{\omega}t) dt = A_y \int_0^{T_m} \left(sin(\overline{\omega}t)cos\varphi_y + cos(\overline{\omega}t)sin\varphi_y\right) sin(\overline{\omega}t) dt + \int_0^{T_m} \left(b + n(t)\right) sin(\overline{\omega}t) dt \tag{12}$$

$$y_c = \int_0^{T_m} y(t) cos(\overline{\omega}t) dt = A_y \int_0^{T_m} \left(sin(\overline{\omega}t)cos\varphi_y + cos(\overline{\omega}t)sin\varphi_y\right) cos(\overline{\omega}t) dt + \int_0^{T_m} \left(b + n(t)\right) cos(\overline{\omega}t) dt \tag{13}$$

An analysis of the right-hand side terms in (12) and (13) leads to a simplification, provided that there is an increase in the averaging time. In this case, the contribution of the last terms can be neglected compared to the first term, which is growing with T_m. Furthermore, it is assumed that for a long integration time, the noise $n(t)$ will be filtered out (i.e., zero average) [30]. In this case, (12) reduces to:

$$y_s = A_y cos\varphi_y \int_0^{T_m} sin^2(\overline{\omega}t) dt + 0.5 A_y sin\varphi_y \int_0^{T_m} 2cos(\overline{\omega}t) sin(\overline{\omega}t) dt \tag{14}$$

The result in (14) is valid even in the case of low signal-to-noise ratios provided that $n(t)$ is a stochastic disturbance with zero mean value and a sufficient number of test signal periods is selected [29].

After some mathematical rearrangement of the terms in (14), the following result is obtained:

$$y_s = 0.5 T_m A_y cos\varphi_y - 0.5 A_y cos\varphi_y \frac{sin(2\overline{\omega}T_m)}{2\overline{\omega}} - 0.5 A_y sin\varphi_y \frac{cos(2\overline{\omega}T_m) - 1}{2\overline{\omega}} \tag{15}$$

Assuming that $T_m = k\frac{2\pi}{\overline{\omega}}$, as mentioned above, namely the measurement (also integration) time is an integer number of periods, the last two terms in (15) will be zero. Then, a simplified result for (12) is finally obtained as:

$$y_s = 0.5 T_m A_y cos\varphi_y \tag{16}$$

A similar analysis can be performed for the signal in (13), leading to:

$$y_c = 0.5 T_m A_y sin\varphi_y \tag{17}$$

Using (12) and (13) leads to:

$$y_c - j y_s = \int_0^{T_m} y(t) \left(cos(\overline{\omega}t) - j sin(\overline{\omega}t)\right) dt = \int_0^{T_m} y(t) e^{-j\overline{\omega}t} dt \tag{18}$$

The integral $\int_0^{T_m} y(t) e^{-j\overline{\omega}t} dt$ in (18) can be calculated via the DFT:

$$\int_0^{T_m} y(t) e^{-j\overline{\omega}t} dt = T_s \sum_0^{N-1} y(kT_s) e^{-j\overline{\omega}kT_s} \tag{19}$$

where T_s is the sampling period, chosen such that $T_m = NT_s$. Using now (16) and (17), the next result holds:

$$y_c - j y_s = -j 0.5 T_m A_y (cos\varphi_y + j sin\varphi_y) = -j 0.5 T_m A_y e^{j\varphi_y} \tag{20}$$

which leads to:

$$A_y e^{j\varphi_y} = \frac{y_c - j y_s}{-j 0.5 T_m} \tag{21}$$

Using (18) and (19) in (21) leads to:

$$A_y e^{j\varphi_y} = \frac{T_s \sum_0^{N-1} y(kT_s) e^{-j\overline{\omega} kT_s}}{-j0.5T_m} \tag{22}$$

Taking into account that $T_m = NT_s$ and rearranging (22) leads to the following solution for determining robustly the amplitude A_y and phase φ_y of the output signal $y(t)$ [26]:

$$A_y e^{j\varphi_y} = \frac{2j}{N} \sum_0^{N-1} y(kT_s) e^{-j\overline{\omega} kT_s} \tag{23}$$

3.2. A Robust Method to Estimate the Process Frequency Response Slope

For the purpose of designing the PID controller using the KC auto-tuning method [26], the frequency response slope needs to be determined, as well. For the process $P(s)$, the frequency response slope is given as:

$$\left.\frac{dP(j\omega)}{d\omega}\right|_{\omega=\overline{\omega}} = j\frac{A_{\overline{y}}}{A_u} e^{j\varphi_{\overline{y}}} \tag{24}$$

A simple strategy to estimate (24) has been developed previously, and it is based on filtering the output signal $y(t)$ as indicated in Figure 6 [31]. The strategy is simple, but prone to errors in the case of stochastic disturbances and noise [26]. An elegant solution to eliminate this problem has been developed and consists of a modification of the basic scheme in Figure 6 as indicated in Figure 7a, where $y_{TR}(t)$ and $y_{SS}(t)$ are the transient and steady-state parts of the output signal $y(t)$, with the transient component going to zero (or to a constant value for an integrating system) [26].

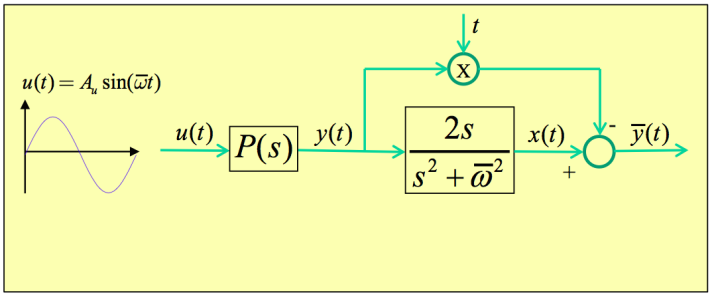


Figure 6. The basic scheme for determining the frequency response slope.

The steady-state component of the output signal can be written simply as:

$$y_{SS}(t) = A_y sin(\overline{\omega}t + \varphi_y) = Ssin(\overline{\omega}t) + Ccos(\overline{\omega}t) \tag{25}$$

where $S = A_y cos(\varphi_y)$ and $C = A_y sin(\varphi_y)$. Applying the Laplace transform to (25) leads to:

$$Y_{SS}(s) = \frac{S\overline{\omega}}{s^2 + \overline{\omega}^2} + \frac{Cs}{s^2 + \overline{\omega}^2} = \frac{S\overline{\omega} + Cs}{s^2 + \overline{\omega}^2} \tag{26}$$

Applying the derivative to (26) leads to the following:

$$\frac{dY_{SS}(s)}{ds} = \frac{C(s^2+\overline{\omega}^2) - 2s(S\overline{\omega}+Cs)}{(s^2+\overline{\omega}^2)^2} = \frac{C}{(s^2+\overline{\omega}^2)} - \frac{2s}{(s^2+\overline{\omega}^2)}\frac{S\overline{\omega}+Cs}{(s^2+\overline{\omega}^2)} = \frac{C}{(s^2+\overline{\omega}^2)} - \frac{2s}{(s^2+\overline{\omega}^2)} Y_{SS}(s) \tag{27}$$

The following result is obtained using the inverse Laplace transform on (27):

$$-tY_{SS}(t) = \frac{A_y sin(\varphi_y)}{\overline{\omega}} sin(\overline{\omega}t) - \mathfrak{L}^1\left(\frac{2s}{s^2 + \overline{\omega}^2} Y_{SS}(s)\right) \tag{28}$$

Figure 7a can now be replaced by Figure 7b, using the result in (28). The Laplace transform of the signal $x(t)$, computed based on Figure 7b, is obtained as:

$$\overline{X}(s) = \frac{2s}{s^2 + \overline{\omega}^2} Y_{TR}(s) = \frac{2s}{\overline{\omega}} Y_{TR}(s) \frac{\overline{\omega}}{s^2 + \overline{\omega}^2} \tag{29}$$

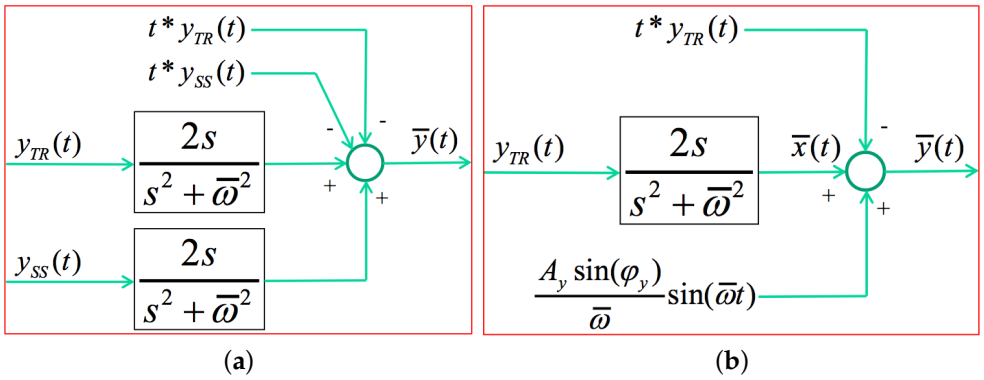


Figure 7. The robust scheme for determining the frequency response slope: (**a**) intermediary solution; (**b**) final solution.

Applying the inverse Laplace transform to (29) and considering that $\frac{\overline{\omega}}{s^2+\overline{\omega}^2} = \mathfrak{L}^{-1} sin(\overline{\omega}t)$ leads to:

$$\overline{x}(t) = \mathfrak{L}^{-1}\{\frac{2s}{\overline{\omega}} Y_{TR}(s)\} sin(\overline{\omega}t) \tag{30}$$

However, the component $t * y_{TR}(t)$ does not influence the steady-state oscillation in $\overline{y}(t)$ and the result in (30). In this case, Figure 7b reduces to the simplified version in Figure 8.

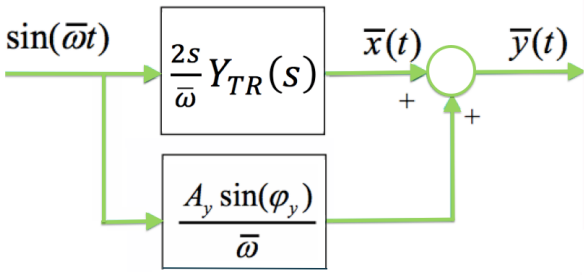


Figure 8. Simplified robust scheme for determining the frequency response slope.

Using the result in Figure 8 leads to a means of computing the amplitude $A_{\overline{y}}$ and phase $\varphi_{\overline{y}}$ of the signal $\overline{y}(t)$ at the specific frequency $\overline{\omega}$ as:

$$A_{\overline{y}} e^{j\varphi_{\overline{y}}} = A_{\overline{x}} e^{j\varphi_{\overline{x}}} + \frac{A_y sin(\varphi_y)}{\overline{\omega}} \tag{31}$$

Since the test frequency $\overline{\omega}$ is user specified and the amplitude A_y and phase φ_y of the output signal $y(t)$ are determined as mentioned previously, in order to determine the amplitude $A_{\overline{y}}$ and phase $\varphi_{\overline{y}}$, the corresponding amplitude $A_{\overline{x}}$ and phase $\varphi_{\overline{x}}$ have to be computed, as well. These can be obtained from the frequency response of the system $\frac{2s}{\overline{\omega}} Y_{TR}(s)$, as indicated in Figure 8. Using the definition of the Laplace transform, the following relation is valid:

$$\frac{2s}{\overline{\omega}} Y_{TR}(s) = \frac{2s}{\overline{\omega}} \int_0^\infty y_{TR}(t) e^{-st} dt \tag{32}$$

Then, the amplitude $A_{\overline{x}}$ and phase $\varphi_{\overline{x}}$ are computed according to:

$$A_{\overline{x}}e^{j\varphi_{\overline{x}}} = 2j\int_0^\infty y_{TR}(t)e^{-j\overline{\omega}t}dt \tag{33}$$

where the discrete Fourier transform is used to evaluate the integral on the right-hand side of (33) as:

$$\int_0^\infty y_{TR}(t)e^{-j\overline{\omega}t}dt = T_s\sum_{k=0}^{N-1} y_{TR}(kT_s)e^{-j\overline{\omega}kT_s} \tag{34}$$

3.3. The Closed Loop Implementation

The robust procedures in the previous sections assume that an input signal can be applied to the process. In case the loop cannot be opened to enter a test-sine in the process $P(s)$, a closed loop approach must be considered. For this purpose, a test-sine $w(t) = A_w sin(\overline{\omega}t)$ must be used, where $w(t)$ is the reference signal of the closed loop system. Figure 9 illustrates this approach. The output $y(t)$ is measured, and the error signal $e(t)$ is further computed as $e(t) = w(t) - y(t)$. Applying the procedure described in Sections 3.1 and 3.2 to the signals $w(t)$ and $e(t)$ leads to the frequency response $S(j\overline{\omega})$ and the frequency response slope $\frac{dS(j\omega)}{d\omega}\Big|_{\omega=\overline{\omega}}$ where $S(s)$, as indicated in Figure 10, is the following transfer function:

$$S(s) = \frac{1}{1 + P(s)C(s)} \tag{35}$$

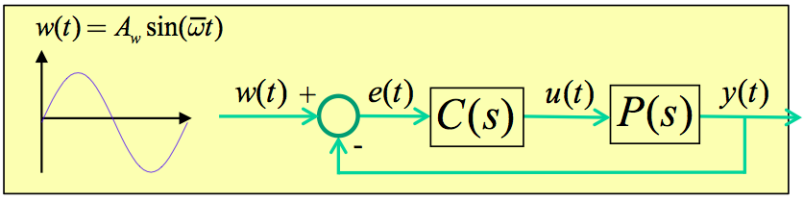


Figure 9. Closed loop scheme for determining the frequency response of a process and the frequency response slope.

Based on (35), it follows that the process frequency response $P(j\overline{\omega})$ can be easily determined as:

$$P(j\overline{\omega}) = \frac{1 - S(j\overline{\omega})}{1 - C(j\overline{\omega})S(j\overline{\omega})} \tag{36}$$

where $C(j\overline{\omega}) = 15$ is a P-type controller in the initial robot closed loop system. Computing the derivative of (35) leads to the following result:

$$\frac{dS}{ds} = -\frac{1}{(1+PC)^2}\left[P\frac{dC}{ds} - C\frac{dP}{ds}\right] \tag{37}$$

The result in (37) leads to:

$$\frac{dS(j\omega)}{d\omega}\Big|_{\omega=\overline{\omega}} = -S(j\overline{\omega})^2\left[P(j\overline{\omega})\,\frac{dC(j\omega)}{d\omega}\Big|_{\omega=\overline{\omega}} - C(j\overline{\omega})\,\frac{dP(j\omega)}{d\omega}\Big|_{\omega=\overline{\omega}}\right] \tag{38}$$

where $\frac{dC(j\omega)}{d\omega}\Big|_{\omega=\overline{\omega}} = 0$, for the P-type controller. Then, using (38), the frequency response slope $\frac{dP(j\omega)}{d\omega}\Big|_{\omega=\overline{\omega}}$ can be computed as:

$$\frac{dP(j\omega)}{d\omega}\Big|_{\omega=\overline{\omega}} = -\frac{\frac{dS(j\omega)}{d\omega}\Big|_{\omega=\overline{\omega}}}{15\;S(j\overline{\omega})^2} \tag{39}$$

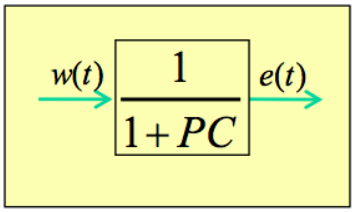


Figure 10. Simplified closed loop scheme for determining the frequency response of $S(s)$and the corresponding frequency response slope.

3.4. An Auto-Tuning PID Controller Based on the Process Frequency Response and Frequency Response Slope

A simple auto-tuning method is used in this paper to design a PID controller using solely the frequency response and its corresponding frequency response slope of the robot at a specific frequency $\overline{\omega}$. This auto-tuning method is called the KC auto-tuner and produces robust PID controllers [25]. The basic idea of the KC auto-tuning method is based on defining a circular region, including the -1 point in the Nyquist plane. This circular region is defined as a "forbidden region", which is a circle computed according to two design constraints: the gain margin (GM = 2) and the phase margin (PM = 45°); see Figure 11. The main design requirement is that the loop frequency response $L(s) = P(s)C(s)$ needs to avoid entering the forbidden region, and most importantly, the robust PID controller parameters are determined in such a way as to ensure that the slope-difference between the circle border and the loop frequency response is minimum:

$$\min_{\alpha} \left\| \left. \frac{dIm}{dRe} \right|_{\alpha} - \left. \frac{d\Im_L}{d\Re_L} \right|_{\overline{\omega}} \right\|, \quad 0 \leq \alpha \leq \alpha_{max} \tag{40}$$

where $\left. \frac{d\Im_L}{d\Re_L} \right|_{\omega=\overline{\omega}}$ is the slope of the loop frequency response in the Nyquist plane and $\left. \frac{dIm}{dRe} \right|_{\alpha}$ is the slope of the "forbidden region" border, as a function of the angle α, as indicated in Figure 11.

To compute $\left. \frac{d\Im_L}{d\Re_L} \right|_{\omega=\overline{\omega}}$, the slope of the loop frequency response has to be evaluated firstly as:

$$\left. \frac{dL(j\omega)}{d\omega} \right|_{\omega=\overline{\omega}} = P(j\overline{\omega}) \left. \frac{dC(j\omega)}{d\omega} \right|_{\omega=\overline{\omega}} + C(j\overline{\omega}) \left. \frac{dP(j\omega)}{d\omega} \right|_{\omega=\overline{\omega}} \tag{41}$$

The process frequency response $P(j\overline{\omega})$ and the process frequency response slope $\left. \frac{dP(j\omega)}{d\omega} \right|_{\omega=\overline{\omega}}$ are determined as mentioned in Sections 3.1 and 3.2. Furthermore, for a user-specified frequency $\overline{\omega}$, the frequency response of the PID controller can be determined according to:

$$C(j\overline{\omega}) = \frac{L(j\overline{\omega})}{P(j\overline{\omega})} \tag{42}$$

whereas $\left. \frac{dC(j\omega)}{d\omega} \right|_{\omega=\overline{\omega}}$ is easily calculated in a numerical way for a known controller $C(s)$ [25].

In this way, the slope of the loop frequency response in (41) can be easily broken into a real $\Re_L$ and an imaginary $\Im_L$ part as:

$$\left. \frac{dL(j\omega)}{d\omega} \right|_{\omega=\overline{\omega}} = \left. \frac{d\Re_L}{d\omega} \right|_{\omega=\overline{\omega}} + j \left. \frac{d\Im_L}{d\omega} \right|_{\omega=\overline{\omega}} \tag{43}$$

Then, the slope of the loop frequency response in the Nyquist plane can be computed as the ratio $\left. \frac{d\Im_L}{d\Re_L} \right|_{\omega=\overline{\omega}}$. Regarding Figure 11, it is obvious that any point of the forbidden region circle is defined by the radius R and the angle α. Computing the real and imaginary parts of this point in the Nyquist plane, the following equations are obtained:

$$Re(\alpha) = -C_c + Rcos(\alpha) \quad \text{and} \quad Im(\alpha) = -Rsin(\alpha) \tag{44}$$

The equation of the circle in that same point gives:

$$(Re(\alpha) + C_c)^2 + Im(\alpha)^2 = R^2 \tag{45}$$

with the derivative equal to:

$$2\,(Re(\alpha) + C_c)\,(dRe) + 2Im(\alpha)(dIm) = 0 \tag{46}$$

Then slope of the "forbidden region" is determined using the result in (46):

$$\left.\frac{dIm}{dRe}\right|_{\alpha} = -\frac{Re(\alpha) + C_c}{Im(\alpha)} = \frac{cos(\alpha)}{sin(\alpha)} \tag{47}$$

where the result in (44) has also been used. With the results in (43) and (47), the minimization problem in (40) is solved with a single for-loop where α varies from 0–α_{max} in 1° steps. Referring to Figure 11, it is obvious that $\alpha_{max} = 90°$.

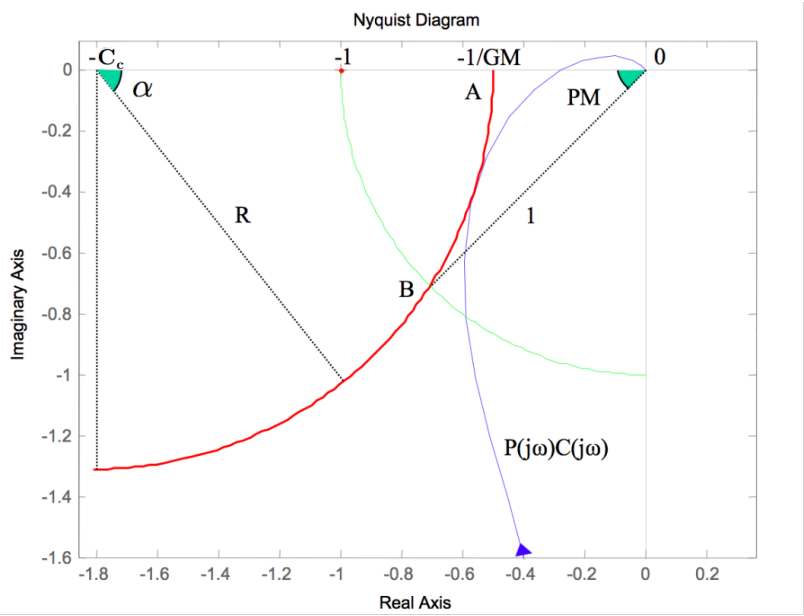


Figure 11. The loop frequency response (blue line) and the 'forbidden region' (red circle). GM, gain margin; PM, phase margin.

3.5. Tuning of the PID Controller

To tune the controller, the test frequency is selected as $\overline{\omega} = 22$ rad/s, while the gain and phase margins are GM = 2 and PM = 45°. Firstly, in the KC auto-tuning method, the process frequency response and frequency response slope at the test frequency are required. For this purpose, a reference signal as indicated next is used and applied to Joints 2 and 3 of the robot, similarly to the procedure described in Section 3.3:

$$w(t) = A_w sim(\overline{\omega} t) \tag{48}$$

with $A_w = 5$. The output signal is measured periodically, and the error signal is computed as:

$$e(t) = w(t) - y(t) \tag{49}$$

The reference and error signals are indicated in Figure 12.

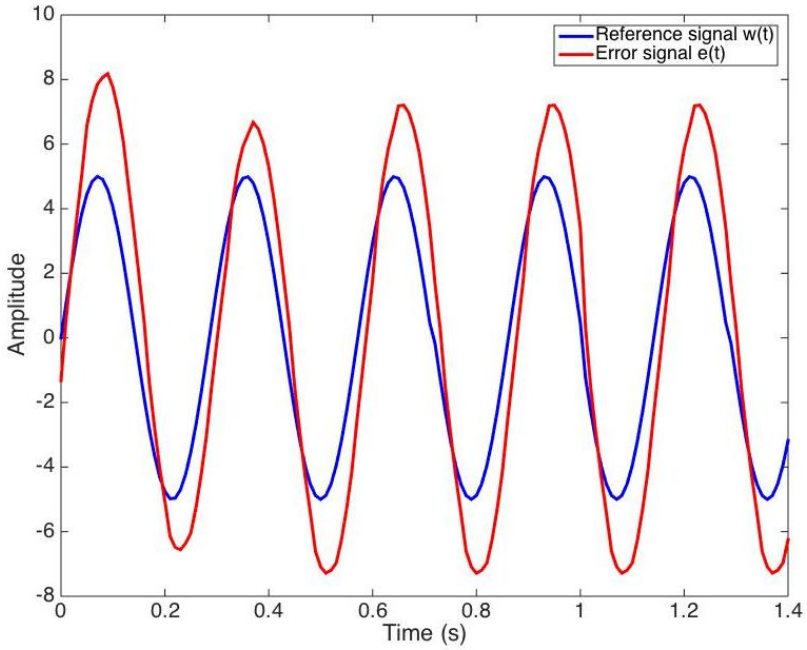


Figure 12. Reference and error signals measured experimentally.

The steady-state component of the error signal can be written simply as:

$$e_{SS}(t) = A_e sin(\overline{\omega}t + \varphi_e) = 7.4sin(\overline{\omega}t - 0.33) \tag{50}$$

and since there is no stochastic disturbance, it can be determined "graphically". Figure 13 shows the reference and error signals measured experimentally, along with the steady state component in (50). The transient component of the error signal is simply determined as the difference:

$$e_{tr}(t) = e(t) - e_{SS}(t) \tag{51}$$

and it is given in Figure 14.

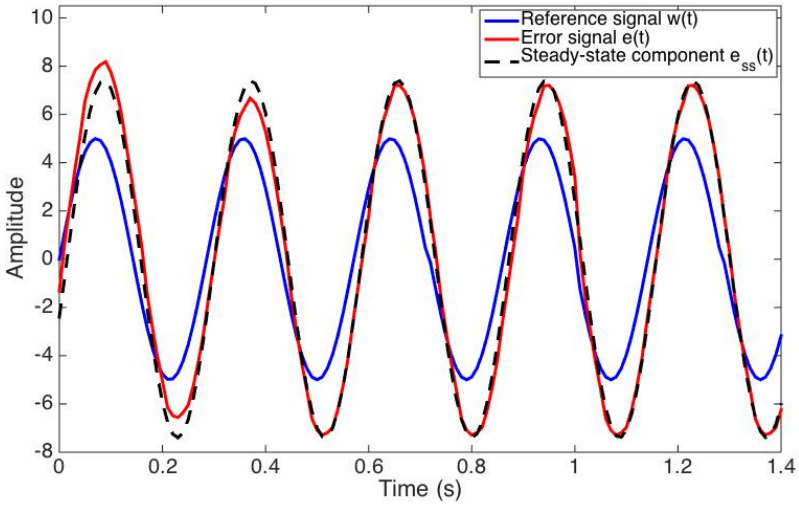


Figure 13. Reference and steady state error signals used to determine the robot frequency response and its corresponding slope.

Next, the discrete Fourier transform is used to evaluate the following integral, as in (34):

$$\int_0^\infty e_{TR}(t)e^{-j\overline{\omega}t}dt = T_s \sum_{k=0}^{N-1} e_{TR}(kT_s)e^{-j\overline{\omega}kT_s} \tag{52}$$

a result that is finally used in an equation similar to (33) to compute the amplitude $A_{\overline{x}}$ and phase $\varphi_{\overline{x}}$:

$$A_{\overline{x}}e^{j\varphi_{\overline{x}}} = 2j\int_0^\infty e_{TR}(t)e^{-j\overline{\omega}t}dt \tag{53}$$

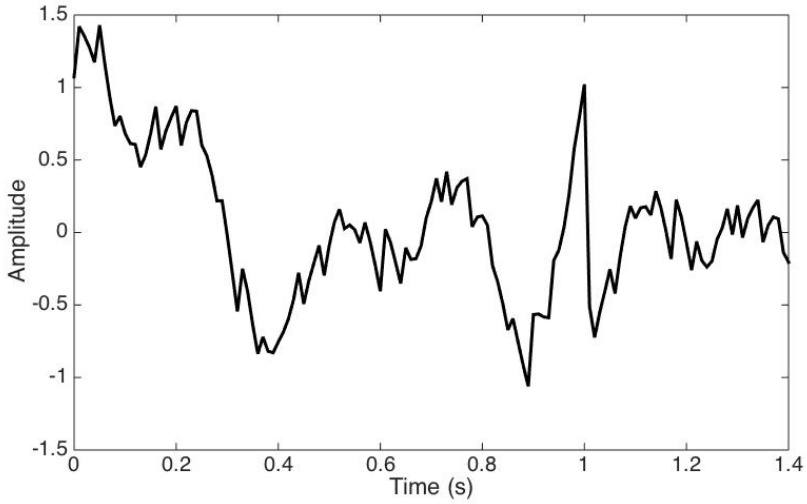


Figure 14. Transient component of the error signals used to determine the robot frequency response and its corresponding slope.

Then, using the result in Figure 8 for the derivative of the error signal $\overline{e}(t)$ leads to a means of computing the amplitude $A_{\overline{e}}$ and phase $\varphi_{\overline{e}}$ of the signal $\overline{e}(t)$ at the specific frequency $\overline{\omega}$ as:

$$A_{\overline{e}}e^{j\varphi_{\overline{e}}} = A_{\overline{x}}e^{j\varphi_{\overline{x}}} + \frac{A_e sin(\varphi_e)}{\overline{\omega}} = -0.109 + 0.0042j \tag{54}$$

with $A_e = 7.4$ and $\varphi_e = -0.33$ rad/s, as indicated in (50).

With the result in (54), the following is obtained by modifying (24):

$$\left.\frac{dS(j\omega)}{d(j\omega)}\right|_{\omega=\overline{\omega}} = j\frac{A_{\overline{e}}}{A_w}e^{j\varphi_{\overline{e}}} = -0.00084 - 0.0218j \tag{55}$$

The frequency response of $S(s)$ at the test frequency $\overline{\omega}$ is computed similarly to the equation in (10) as:

$$S(j\overline{\omega}) = \frac{A_e}{A_w}e^{j\varphi_e} = 1.4 - 0.48j \tag{56}$$

The process frequency response $P(j\overline{\omega})$ can be easily determined using (36) and the result in (56), considering that the controller used is P-type with $C(j\overline{\omega}) = 15$:

$$P(j\overline{\omega}) = \frac{1 - S(j\overline{\omega})}{1 - C(j\overline{\omega})S(j\overline{\omega})} = -0.024 + 0.0145j \tag{57}$$

The process frequency response slope is computed according to (39):

$$\left.\frac{dP(j\omega)}{d\omega}\right|_{\omega=\overline{\omega}} = -3.86 * 10^{-4} + 5.4 * 10^{-4} j \tag{58}$$

The resulting PID controller, computed according to the KC auto-tuning procedure described in Section 3.4, is the following:

$$C(s) = 9.42\left(1 + \frac{1}{0.45s} + 0.21s\right) \tag{59}$$

4. Experimental Results

This section reports the experimental results that were achieved in order to validate the auto-tuning method of a PID controller, presented in the previous section. The performance of the obtained controllers has been evaluated in the real-time application using the setup depicted in Figure 15. In order to communicate with the UR10 robot, the Robotics System Toolbox of MATLAB® and the ROS platform were used.

Figure 15. The UR10 robot available in our laboratory.

The UR10 controller uses Ethernet and the TCP/IP protocol to send and receive commands. The control and calculations of the robot are executed in MATLAB®. Although it is possible to send commands directly from MATLAB® to the controller, it was chosen to use the ROS platform for a more versatile solution. Therefore, the ROS-supported *ur_moder_driver* runs on a computer with a Linux operating system, which is directly connected to the robot controller through TCP/IP. The *ur_modern_driver* allows an easy communication with many different operating systems and publishes robot data with the use of ROS messages. As MATLAB® has a built-in ROS support, this ensures a very clear and effortless communication; for more details, please see [32]. An overview of the complete communication model is given in Figure 16. The running nodes and topics within the ROS platform related to this case study are illustrated in Figure 17.

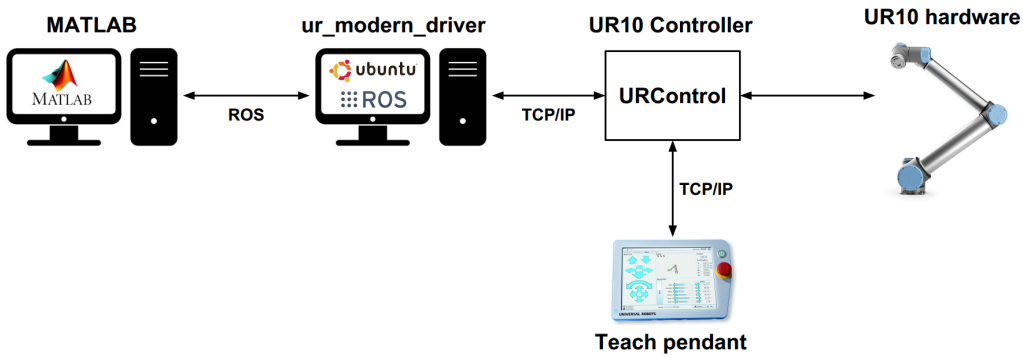


Figure 16. Communication model of the data transmission between the UR10 robot and the MATLAB® software.

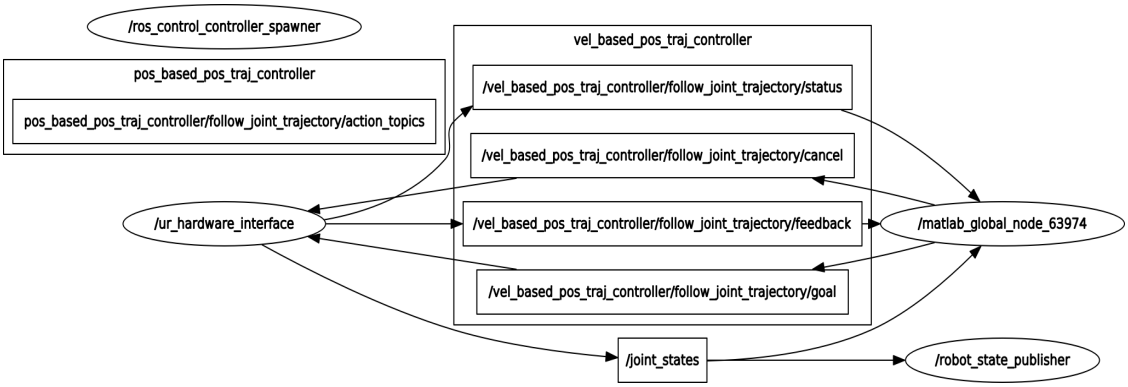


Figure 17. Visualization of the ROS computation graph.

The procedure used to control the movements of the robot is presented in the flowchart in Figure 18. Notice that the auto-tuning method implies the use of *ROS* − *controller* and *velocity_based_controller*, as well. Besides controlling the movements of the robot, it is very important to have access to the status of the robot's joint states (joint position, joint velocity, joint effort, etc.). Thus, it can be mentioned that implementing a new controller on an industrial robot is strongly dependent on the opening level of the robot controller and the type of the driver used. In order to implement the auto-tuning method presented in this paper it is required to have read/write access to the joint states of the robot.

The speed control was developed and applied to Joint 2 and Joint 3 of the manipulator robot (these two joints have a larger variation compared with other joints). The controller was tested for multiple scenarios and different configurations of the manipulated robot. The first test was conducted using the following configuration $[-80, -45, -50, -80, 260, -90]$, and the obtained results are presented in Figures 19a and 20a. The output of the manipulator robot for the second configuration ($[35, -102, -75, 90, -120, 110]$) considered in this paper is presented in Figure 19b and 20b. The third test case presented in this paper is illustrated in Figures 21 and 22. As observed, the controller performs equally for different joint configurations, while keeping the interaction between joints at a minimum. It can be seen that the PID controller designed using the proposed auto-tuning procedure converges to the imposed reference for all considered cases and presents no overshoot in comparison with the available controller. Given the problem addressed in this paper, the results provided indicate that the auto-tuning method can be successfully applied in the field of robotic control.

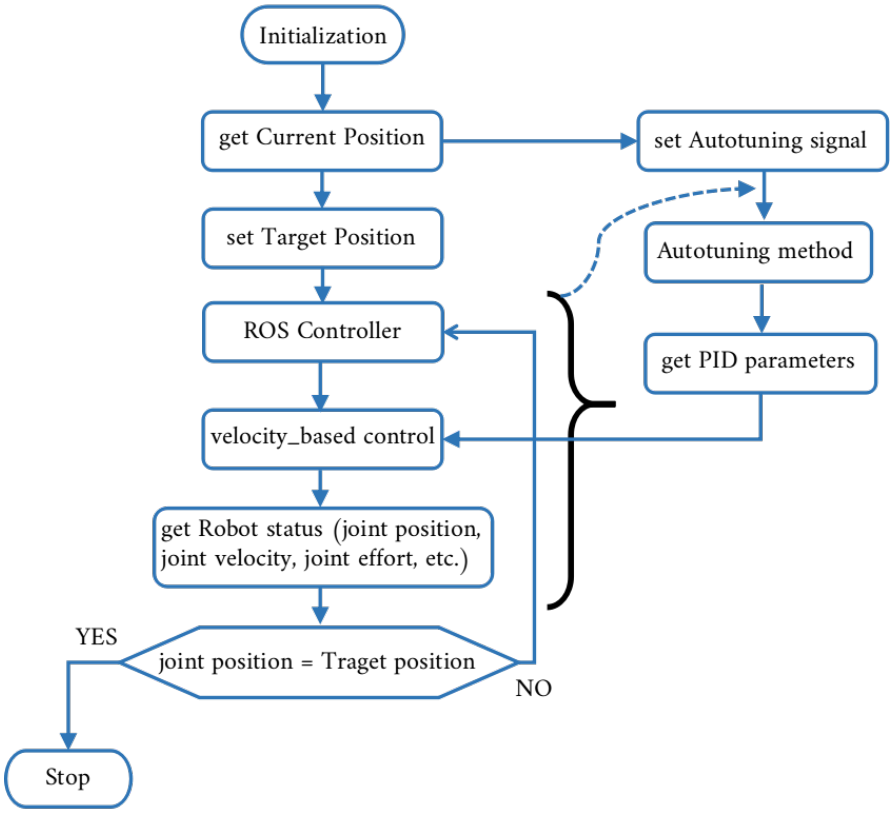


Figure 18. The flowchart for controlling the movements of the robot.

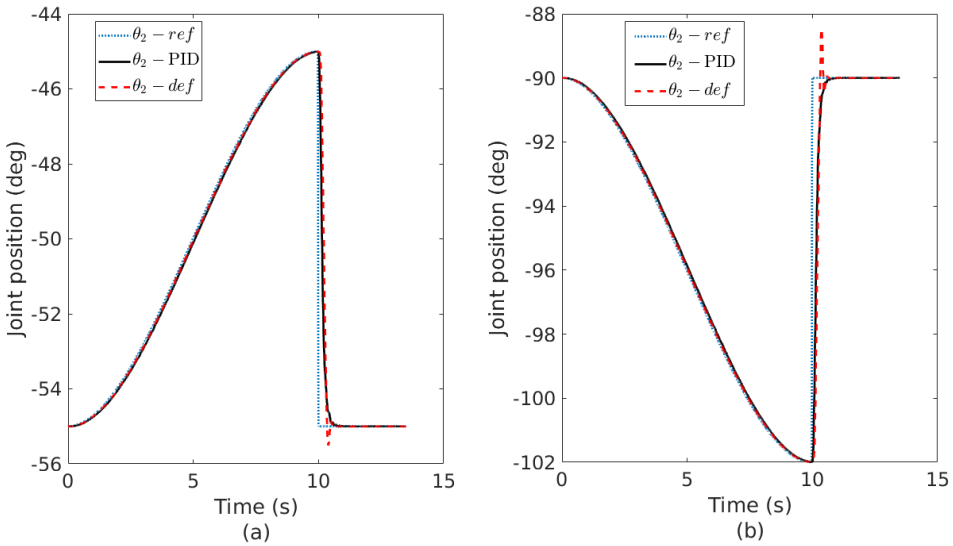


Figure 19. Step response of the UR10 robot for Joint 2(PID, the designed controller; def, the default controller available on the robot); (**a**) first robot configuration and (**b**) second robot configuration.

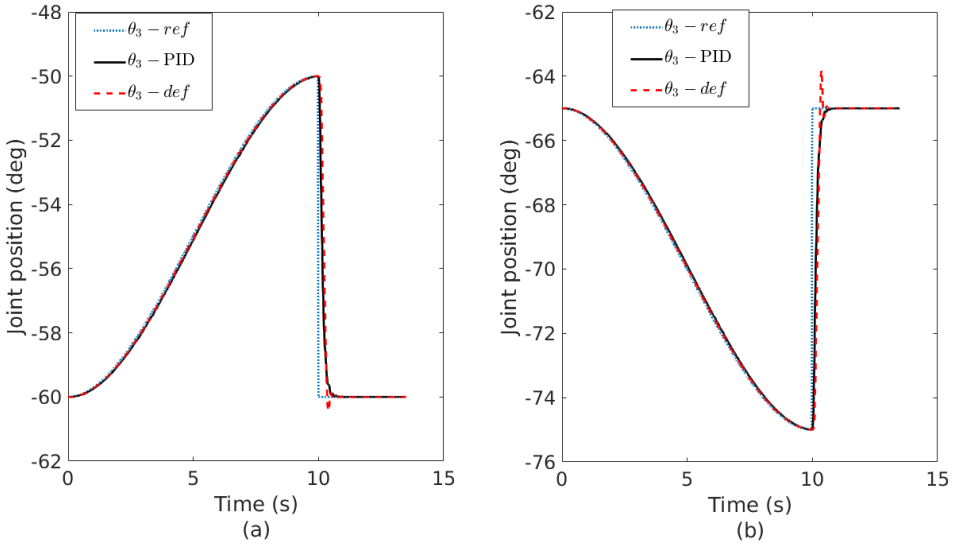


Figure 20. Step response of the UR10 robot for Joint 3; (**a**) first robot configuration and (**b**) second robot configuration.

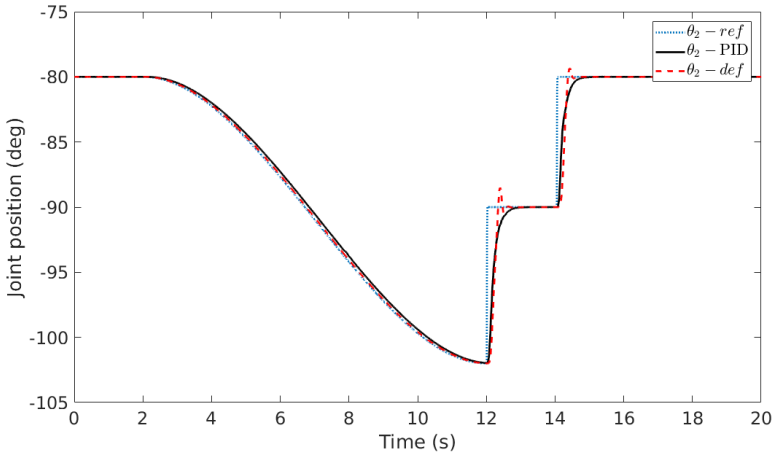


Figure 21. Response of the UR10 robot for Joint 2.

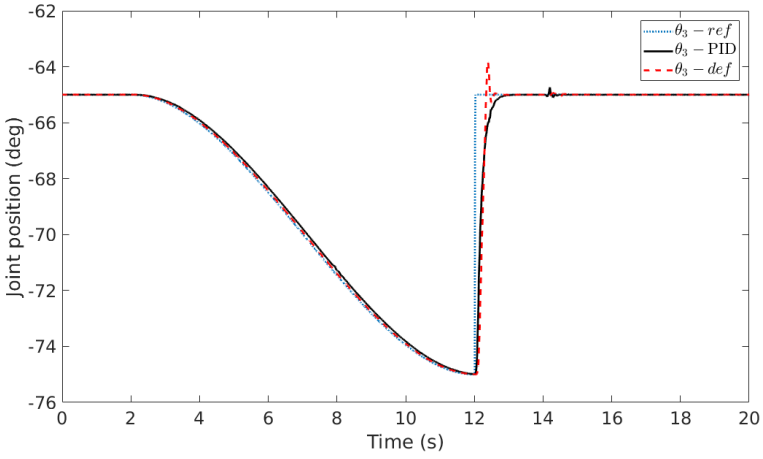


Figure 22. Response of the UR10 robot for Joint 3.

5. Conclusions

In this paper, a PID based on a single sine test auto-tuning method was designed and implemented for a robot system. Due to the advantage of the auto-tuning approach, which ensures that the difference between the slope of the region border in the Nyquist plane and the loop frequency response slope are minimum, the obtained PID controller is robust to gain, phase and delay variations. This robustness is highly demanded for a manipulator robot taking into account its dynamics (variation with respect to the position and orientation of the robot). The outcome of this paper shows that the proposed technique is a suitable method for tuning PID controllers. The experimental results indicate an increased performance of the manipulator robot when the proposed auto-tuning technique is used. Even more, this method could bring a benefit for both real-life and industrial applications given that it is time saving and is easy to use by non-experts in control engineering.

Author Contributions: R.D.K. developed the auto-tuning method. C.M. and C.-M.I. designed the controller based on experimental data and determined the process frequency response and its corresponding slope information required for the tuning of the controller. C.C. and S.V. implemented the controller and performed the experimental tests. C.C. and C.M. analyzed the results and wrote the paper.

Funding: This research received no external funding.

Acknowledgments: This work has been partially supported by the BOF-STIMPRO funding of the University of Antwerp.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Spong, M.; Hutchinson, S.; Vidyasagar, M. *Robot Modeling and Control*; Wiley: Hoboken, NJ, USA, 2006.
2. Bishop, R. *The Mechatronics Handbook*; CRC Press: Boca Raton, FL, USA, 2002.
3. Steinbuch, M.; Merry, R.; Boerlage, M.; Ronde, M.; van de Molengraft, M. Advanced Motion Control Design. In *The Control Handbook: Control System Application*; Levin, W.S., Ed.; CRC Press: Boca Raton, FL, USA, 2010.
4. Steinbuch, M.; Schootstra, G.; Bosgra, O. Robust control of a compact disc mechanism. In *Control System Applications*; Levine, W.S., Ed.; CRC Press: Boca Raton, FL, USA, 2000.
5. Astrom, K.; Hagglund, T. *Advanced PID Control*; ISA-The Instrumentation, Systems, and Automation Society: Durham, NC, USA, 2006.
6. Ionescu, C.; Copot, C.; Verellen, D. Motion compensation for robotic lung tumour radiotherapy in remote locations: A personalised medicine approach. *Acta Astronaut.* **2017**, *132*, 59–66. [CrossRef]
7. Sayeh, S.; Wang, J.; Main, W.; Kilby, W.; Maurer, C.R., Jr. Respiratory motion tracking for robotic radiosurgery. In *Treating Tumours that Move with Respiration*; Urschel, H.C., Jr., Ed.; Springer: Berlin/Heidelberg, Germany, 2007; pp. 15–29.
8. Seppenwoolde, Y.; Berbeco, R.; Nishioka, S.; Shirato, H.; Heijmen, B. Accuracy of tumour motion compensation algorithm from a robotic respiratory tracking system a simulation study. *Med. Phys.* **2007**, *34*, 2774–2784. [CrossRef] [PubMed]
9. Tang, X.; Liu, Y.; Lv, C.; Sun, D. Hand Motion Classification Using a Multi-Channel Surface Electromyography Sensor. *Sensors* **2012**, *12*, 1130–1147. [CrossRef] [PubMed]
10. Raya, R.; Rocon, E.; Gallego, J.; Ceres, R.; Pons, J. A Robust Kalman Algorithm to Facilitate Human-Computer Interaction for People with Cerebral Palsy, Using a New Interface Based on Inertial Sensors. *Sensors* **2012**, *12*, 3049–3067. [CrossRef] [PubMed]
11. Yan, R.; Tee, K.; Li, H. Nonlinear control of a robot manipulator with time-varying uncertainties. In Proceedings of the Second International Conference on Social Robotics (ICSR), Singapore, 23–24 November 2010; pp. 202–211.
12. Xu, J.; Viswanathan, B.; Qu, Z. Robust learning control for robotic manipulators with an extension to a class of non-linear systems. *Int. J. Control* **2000**, *76*, 858–870. [CrossRef]
13. Xian, B.; Dawson, D.; Queiroz, M.; Chen, J. A continuous asymptotic tracking control strategy for uncertain nonlinear systems. *IEEE Trans. Autom. Control* **2004**, *49*, 1206–1211. [CrossRef]

14. Busoniu, L.; Schutter, B.; Babuska, R. Decentralized reinforcement learning control of a robotic manipulator. In Proceedings of the 9th International Conference on Control, Automation, Robotics and Vision, Singapore, 5–8 December 2006; pp. 1347–1352.
15. Bouakrif, F.; Boukhetala, D.; Boudjema, F. Velocity observer-based iterative learning control for robot manipulators. *Int. J. Syst. Sci.* **2013**, *44*, 214–222. [CrossRef]
16. Delchev, K.; Boiadjiev, G.; Kawasaki, H.; Mouri, T. Iterative learning control with sampled-data feedback for robot manipulators. *Int. J. Arch. Control Sci.* **2014**, *24*, 299–319. [CrossRef]
17. Astrom, K.; Hagglund, T. Automatic tuning of simple regulators with specifications on phase and amplitude margins. *Automatica* **1984**, *20*, 645–651. [CrossRef]
18. Tan, K.; Lee, T.; Wang, Q. Enhanced automatic tuning procedure for process control of PI/PID controllers. *AlChE J.* **1996**, *42*, 2555–2562. [CrossRef]
19. Chen, Y.; Moore, K. Relay feedback tuning of robust PID controllers with iso-damping property. *Trans. Syst. Man. Cybern.* **2005**, *35*, 23–31. [CrossRef]
20. Hagglund, T.; Astrom, K. Revisiting the Ziegler—Nichols tuning rules for PI control. *Asian J. Control* **2002**, *4*, 364–380. [CrossRef]
21. Akkermans, S.; Stan, S. Digital servo IC for optical disc drives. *Control Eng. Pract.* **2002**, *9*, 1245–1253. [CrossRef]
22. Panagopoulos, H.; Astrom, K.; Hagglund, T. Design of PID controllers based on constrained optimisation. *IEEE Proc. Control Theory Appl.* **2002**, *149*, 32–40. [CrossRef]
23. Astrom, K.; Hagglund, T. Revisiting the Ziegler—Nichols step response method for PID control. *J. Process. Control* **2004**, *14*, 635–650. [CrossRef]
24. Skogestad, S. Simple analytic rules for model reduction and PID controller tuning. *J. Process. Control* **2003**, *13*, 291–309. [CrossRef]
25. De Keyser, R.; Ionescu, C.; Muresan, C. Comparative Evaluation of a Novel Principle for PID Autotuning. In Proceedings of the Asian Control Conference, Gold Coast, Australia, 17–20 December 2017; pp. 1164–1169.
26. De Keyser, R.; Muresan, C.; Ionescu, C. A single sine test can robustly estimate a system's frequency response slope. *Int. J. Syst. Sci.* **2018**, submitted.
27. Corke, P. *Robotics, Vision & Control*; Springer: Berlin/Heidelberg, Germany, 2011.
28. Siciliano, B.; Sciavicco, L.; Villani, L.; Oriolo, G. *Robotics, Modelling, Planning and Control*; Springer: Berlin/Heidelberg, Germany, 2009.
29. Naumovic, M.; De Keyser, R.; Robayo, F.; Ionescu, C. Developing Frequency Response Analyzer in MATLAB® Simulink Environment. In Proceedings of the 17th Telecommunications Forum TELFOR, Serbia, Belgrade, 24–26 November 2009; pp. 701–704.
30. De Keyser, R.; Ionescu, C.; Festila, C. A One-Step Procedure for Frequency Response Estimation based on a Switch-Mode Transfer Function Analyzer. In Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC), Orlando, FL, USA, 12–15 December 2011; pp. 1189–1194.
31. De Keyser, R.; Muresan, C.; Ionescu, C. A Novel Auto-tuning Method for Fractional Order PI/PD Controllers. *ISA Trans.* **2016**, *62*, 268–275. [CrossRef] [PubMed]
32. Andersen, T.T. *Optimizing the Universal Robots ROS Driver*; Technical Report; Department of Electrical Engineering, Technical University of Denmark: Lyngby, Denmark, 2015.