Networking Programmable Logic Controllers: Pneumatic Cylinder Modelling and Control

Diogo Miguel Correia dos Santos
Instituto Superior Técnico / UTL, Lisbon, Portugal

***Abstract*—Building real world reliable and robust systems nowadays is usually associated to the use of Programmable Logic Controllers (PLCs). These devices are modular digital computers that allow using a large variety, and number, of electrical input and/or output interfaces. The mechanical, hardware and software are designed to handle continuous operation in environments overwhelmed with electromagnetic and mechanical noise. These characteristics make PLCs ideal for usage within industrial environments: travel industry, aerospace, food industry and textile industry, or any other area with systems operated by controllable machines. However, as technology advances, the systems to be controlled become more complex. As a result, networking PLCs is becoming not just simply an interest, but an actual requirement. PLC network systems profit greatly from having a Supervisory Control and Data Acquisition (SCADA) system to supervise them. This system is generally composed of a remote monitoring and operation software application that allows observation and control of the industrial process. With the level of detailed surveillance and control capabilities that SCADA provides, the security and productivity of the system both increase. The objective of this thesis was to design a remote monitoring and operation interface for a physical setup with equipment that could be controlled by Programmable Logic Controllers.**

The available equipment for the construction of the physical setup was a set of pneumatic components. Pneumatics is a section of technology with a whole theory behind it, and so, it was necessary to get briefly acquainted with that theory before proceeding with any experiments. The assembled system was formed by a double acting cylinder, a double solenoid valve, single direction speed controllers, and a compressor. The purpose of this setup was to be able to control the position of the cylinder through the remote monitoring and operation interface.

The PLC model used accepted external communications (to be more precise, an Ethernet/IP based protocol named *Modbus*) while in operating mode, but only under specific conditions. A dedicated study was indispensable to ascertain the exact circumstances under which the PLC allowed external commands, as well as to find out their structure.

I. Introduction

A. Motivation

With the recent advances in the world of automation, especially in the industrial field, the use of modular digital computers has become vital. These devices can withstand very harsh environments, and offer a simple, wide variety of input/output capabilities that would otherwise be achieved with large, complex relay circuits. Throughout this thesis they will be addressed as Programmable Logic Controllers (PLCs) [1], since it is the name they are commonly known for in industrial automation.

Naturally, to control numerous machines effectively, it is usually desirable to have some means of interconnection between the PLCs that are in charge of the control of such machines. In most cases, for efficiency purposes, several networks are required, where PLCs are connected to each other through physical or wireless communication channels. These setups are common when the PLCs are responsible for the state of the system (or the state of a subsystem) as a whole, rather than the state of just one machine. Nowadays, most PLC networks are composed of Ethernet/IP infrastructures, mainly due to the speed of data transmission that Ethernet provides, as well as the simplicity of the message framing protocols that were developed for those structures.

The domain of interest in this thesis lies within the Discrete Event Systems (DES). To implement a given DES, there are three typical modelling methodologies [2]: use a GRAFCET (otherwise known as Sequentional Function Chart, SFC), use a Petri Net, or use an Automaton. The first choice, GRAFCET, is actually a graphical programming language, one of the five languages defined by IEC 61131-3 that can be used to program PLCs [3]. Therefore, its implementation in PLCs is straightforward, as the only requirement is the knowledge of the language. The second choice, Petri Net, has more modelling capacity than GRAFCET, but it does not have a direct implementation in PLCs, consequently demanding some indirect solution, or the help of software that can interpret Petri Nets on its own. However, for Petri Nets, there are several analysis methods available that provide a lot of insight about the system. This is a great advantage compared to GRAFCET, since the latter is very limited in analysis methods. Lastly, the automaton is a self-operating machine, built explicitly for the desired system.

Whichever choice is made for the modelling of the DES, if the intended purpose of the system falls within industrial automation, then a model of control for the system, along with an efficient User Interface, are both also necessary. Presently, industries rely on Supervisory Control And Data Acquisition (SCADA) [4] systems for control access and remote monitoring of their machinery. Simply put, they are the modern example of Supervised systems, resorting to software developed for monitoring and control through the acquisition of coded signals from the equipment. The frequent association between SCADA systems and PLCs is that usually the PLCs provide the signals to be interpreted by the SCADA software as a specific state of the machines they are responsible for, and in turn, the SCADA system sends back signals that are interpreted by the PLCs as commands for those machines.

Additionally, SCADA systems are also designed to grant features like failure detection, isolation capacity, and providing historical information about previous activity of the controlled system [2].

The goal of this thesis is to approach and solve efficiently an academic, industrial automation problem: the design and implementation of a SCADA system to oversee and control a Discrete Event System. In order to accomplish this, it will be necessary to model and create an industrial resembling setup, resorting to modelling dynamics tools. This setup is going to be controlled by a PLC network, using a specific message framing protocol designed for Ethernet/IP networks that is currently being used in modern industries: Modbus [5]. The available theory on modelling DES may also be useful. The work can thus be summarized into two main steps:

- Assembling a small physical setup that allows remote monitoring and operation, in which the equipment can be controlled by Programmable Logic Controllers. Specifically, the PLC will command pneumatic actuators.

- Designing the remote monitoring and operation interface for the physical setup. The PLC model available to actuate the system is Schneider Premium TSX P57 [6]. The communications protocol of the network connecting the Supervised system to the PLC is Modbus.

B. Related Work

The work *Modeling and Identification of Pneumatic Actuators* [7] presents an in-depth study of a pneumatic actuator. It offers both a defined physical model and a possible design of a non-linear parametric model for it. This knowledge is then applied in the control of a humanoid robot. Categorically, [7] provides useful information about the principles of the cylinder model, and inclusively mentions a particularly interesting phenomenon: stiffness. It is commonly known as the resistance of a body to deformation by influence of external forces, but in pneumatic actuators such as the cylinders, it may be used as a form of control. During the empirical work of this thesis, a two-chamber cylinder like the one mentioned in [7] will be used, and therefore, both the information and the described phenomenon are substantial contributions worth exploring.

Method for communicating among a plurality of programmable logic controllers each having a DMA controller [8] discusses a method for communicating among several Programmable Logic Controllers, all coupled to a common communications bus (serial). Many of the concerns that need to be approached in the method described in [8], are equivalently present in this thesis. However, the creation of X-Way communication, and the fact that it is compliant with the Programmable Logic Controllers used throughout this thesis, greatly simplifies these issues.

C. Problem Formulation

The objectives of this thesis, described in I-A, require solving two problems. The first one concerns the physical setup, whereas the second is related to communications. It is known beforehand that the available equipment for the setup consists of a kit [1] of pneumatic cylinders and valves [2]. The proposed strategy for the usage of these components is to create a simple system in which the position of a double acting cylinder[3] is measured and controlled. The need to take measurements leads to the necessity of a proper sensor. As for the control, the intent is to actuate one or more valves, to move the cylinder to a desired position. The actuation will be performed by a PLC.

Ideally, the assembled setup would have a Network composed of two PLCs. One of them would receive measurements, and the other would command the actuation based on those measurements. The specifications of the available PLC model ([6]) indicate that the inter-communications are compliant with a specific Ethernet/IP protocol, Modbus ([5]). This will be the communications protocol used in the Network. However, the PLC model does not have input analog cards. For that reason, it is suggested that an Arduino is used instead, to receive the measurements. The Arduino will be able to send the measurements via Ethernet with the use of an Ethershield. Finally, it is also expected that the Network can be monitored, and the system controlled, through a remote monitoring and operation station.

D. Thesis Outline

Section II provides detailed information about the hardware of modern Programmable Logic Controllers (PLCs).

Section III presents theory regarding Pneumatics. Specifically, it emphasizes on the pneumatic cylinder model, since it illustrates the actual cylinder used in the experiments of the dissertation. It also offers an overview of the full setup, designed to be controlled by PLCs, and explains in detail three different methods suggested to perform the position control of the cylinder.

Section IV summarizes the most relevant experiments conducted throughout the dissertation, and their respective results. This Section is divided in two parts: the first regarding the theoretical model of the pneumatic cylinder, and the second concerning the empirical model. The first experiment shows some results regarding computer simulations of the cylinder model, intended for use in the real system. Based on those results, an approximation for the system is proposed. The second experiment represents the chosen application for the available hardware and software: position control of the double acting cylinder. The results of the empirical system include Step response, Calibration methods, and Closed-loop.

Section V is a conclusion of the developed work, as well as its contributions.

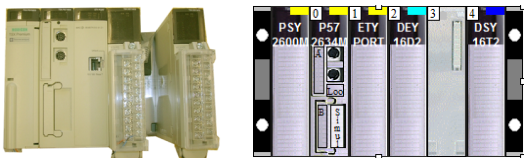
II. Programmable Logic Controllers

Section II addresses the subject of Programmable Logic Controllers (PLCs), which form the central part of the envisaged setup of the dissertation. The specific PLC model used in the dissertation is briefly introduced in this section.

[1] https://www.sjf.tuke.sk/kvtar/2/files/PNEU200.pdf

[2] http://www.datasheetarchive.com/dl/Datasheets-UD9/DSARS0040963.pdf

[3] https://www.smc.eu/portal_ssl/WebContent/digital_catalog_2/jsp/view_product_configurator.jsp?dc_product_id=36559&part_number=cd85n20&filter_type=dc_search_filter&sub_filter_type=partnumber&filter_value=cd85n20



(a) (b)

Fig. 1. PLC TSX P57, picture of real model (a) and virtual representation in Unity Pro (b)

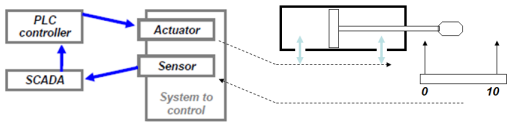


Fig. 2. Real Setup: communications sequence

A. Programmable Logic Controllers

The model at our disposal for this dissertation is Schneider's TSX P57 Premium [6]. Figure 1 displays both the real model, and its virtual representation in the software where the PLC is programmed, Unity Pro.

This model has a rack with five slots. The CPU model (P57 2634M) and the Ethernet module (ETY Port) are plugged into slots '0' and '1', respectively, although physically they are a single hardware piece. The Ethernet module accepts a connection to a TCP/IP network with X-WAY UNI-TE and Modbus messaging facility on a TCP/IP profile, but it must be configured in Unity Pro to do so. By default, the project built in Unity Pro must be loaded to the PLC via RS232/RS485 connection. The first serial port on the CPU module is used for this effect.

The Inputs module (DEY 16D2) is plugged into slot '2'. Usually, the signals received by this module dictate what actions the PLC will perform, based on how it was programmed.

The Outputs module (DSY 16T2) is plugged into slot '4'. The signals transmitted by this module are responsible for the actuation of the machines they are connected to.

III. System Process and Control

The next section describes all the components of the physical setup assembled and the methodology used to control it. The goal of the experiment is to control the position of a double acting cylinder, using a pneumatic actuating valve, which will be switching states according to the commands given by a PLC. A sensor will also be necessary, to measure the current position of the head of the cylinder, and send these values to an Arduino.

A. Closed loop System Overview

The empirical system will follow the communications sequence shown in figure 2. This figure also displays the functions that the Actuator and the Sensor will be performing. Explicitly, the PLC will be controlling the actuation of two valves, affecting the air flow in and out of the chambers of a double acting cylinder. This flow will make the cylinder move

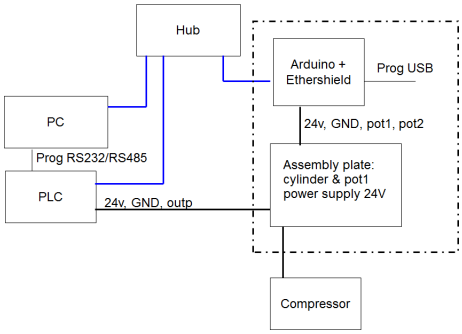


Fig. 3. Full Setup Scheme

or stop on a certain position. The sensor will be measuring the position of the head of the cylinder, which may vary between 0 and 10 centimeters. The sensor will send those measurements to a SCADA system, which, in turn, will evaluate the next command to give to the PLC, based on the value received.

Figure 3 shows all the connections between all the components of the system. Regarding the PLC Network, the Arduino also plays the role of a PLC, as it works as a programmable device with input/output capabilities.

All the communications depend on the HUB: the sensor sends the values it reads to a server; the PC connects to that server through TCP/IP to be able to access those values; the PC also connects to the PLC through TCP/IP, in order to send commands to it (using Modbus). There are also two secondary connections: the RS232/RS485 connection is necessary to send the program that the PLC must be running throughout the experiment (with Unity Pro); the arduino also needs to receive a program through the USB connection.

There are four main components in this setup: PLC, PC, Pneumatic Setup and Sensor.

The PLC is powered by a 24V DC source. There are two PLC outputs connecting to the valve: %Q0.4.3 is connected to the left contact of the valve (the one that pushes the cylinder) and %Q0.4.1 connected to the right contact of the valve (the one that pulls the cylinder). In this experiment, the PLC functions as an actuator: when given the command, it sets an output. That output is an ON/OFF signal (24V or 0V) that activates/deactivates the valve's contacts, switching the valve's state to one of the available three.

The PC will be running the SCADA software. This software will be responsible for the Control loop to manage the position of the cylinder, as well as showing the cylinder's current position to the user, as expected of a Supervised system.

B. Double Acting Cylinder model

The term pneumatic is used to describe systems that use air or other gases. In several cases, including the one studied in this thesis, pneumatic systems convert energy of compressed air to mechanical energy, which is transferred by force. This force generates movement, and thus, these systems are also known as *pneumatic actuators* [9].

Pneumatic actuating valves provide a large power output, assuming that there is a large quantity of pressurized air to offer as input. In practice, the valve characteristics are usually

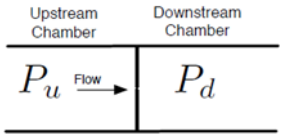


Fig. 4. Port model of a pneumatic actuating valve. Extracted from [10].

not linear; the air flow is not proportional to the valve stem position, and there are usually other nonlinear effects in play, like hysteresis.

Section III-B is dedicated to the study of a simple cylinder model. This model consists of two chambers, separated by a sliding bore. It is assumed each chamber has uniform pressure across its entire volume, and two constant pressure sources: the compressor and the atmosphere.

1) Port Model: The *port* is a small orifice between two chambers which have different pressures. The port model describes the flow of fluid (in this case, air) through the port as a function of the area of the orifice, and the upstream and downstream pressures. There are a few key assumptions in this analysis: the area of the port is small, the plate separating the chambers is thin, the fluid is a perfect gas, the temperatures in the two chambers are equal, and the flow is isentropic, meaning that there is no temperature transfer (heat production) in the process.

Given these assumptions, it can be shown that the flow $\dot{m}$ of fluid mass through the orifice is described by the following equation [7]:

$$\dot{m} = \begin{cases} a.\phi(p_u, p_d) & if P_u \geq P_d \\ -a.\phi(p_d, p_u) & if P_u < P_d \end{cases} \tag{1}$$

where a is the area of the orifice and $\phi(p_u, p_d)$ is the thin-plate flow function (non-linear until p_u is bigger than θ times p_d) [7]:

$$\phi(p_u, p_d) = \begin{cases} \alpha.p_u\sqrt{(\frac{p_d}{p_u})^{\frac{2}{k}} - (\frac{p_d}{p_u})^{\frac{k+1}{k}}} & \text{for} \quad p_u/p_d \leq \theta \\ \beta.p_u & \text{for} \quad p_u/p_d > \theta \end{cases} \tag{2}$$

The parameters α, β and θ are given by:

$$\alpha = c.\sqrt{\frac{2M}{ZRT}.\frac{k}{k-1}} \tag{3}$$

$$\beta = c.\sqrt{\frac{kM}{ZRT}.(\frac{2}{k+1})^{\frac{k+1}{k-1}}} \tag{4}$$

$$\theta = (\frac{k+1}{2})^{\frac{k}{k-1}} \tag{5}$$

Constants:

- M is the Gas Molecular Mass (0.029 Kg/mol for air)
- Z is the Gas compressibility Factor (0.99 for air)
- R is the Ideal Gas Constant (8.3144621 $m^3PaK^{-1}mol^{-1}$)
- T is the Temperature (Kelvin)
- k is the Specific heat Ratio (1.4 for air)
- c is the Discharge coefficient (0.72 dimensionless)

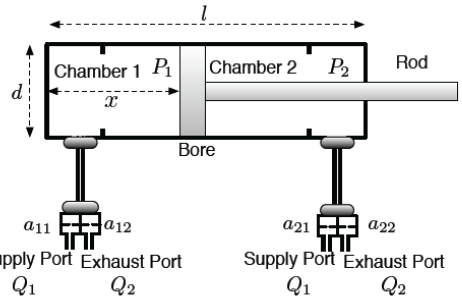


Fig. 5. Cylinder model with two ports per chamber: $a_{11} = a_{21} = a_c$ and $a_{12} = a_{22} = a_r$. Q_1 is the supply port, and Q_2 is the exhaust port. Figure extracted from [10]

2) Two Ports Chamber model: This model represents a chamber with two ports, one of which connected to the compressor, and the other connected to the room. The total flow of fluid mass is the difference between the flows.

$$\dot{m}(p, a_c, a_r) = a_c.\phi(P_c, p) - a_r.\phi(p, P_r) \tag{6}$$

p is the current pressure in the chamber. a_c and a_r are the areas of the orifices connecting to the compressor and to the room, respectively. P_c is the pressure of the compressor, and P_r is the pressure of the room.

3) Valve model: The areas of the valve ports depend only on the voltage applied to the valve, u (Volts):

$$a_c(u) = \begin{cases} 0 & \text{if} \quad u = 0 \\ a_c & \text{if} \quad u = 24 \end{cases} \tag{7}$$

4) Cylinder pressure dynamics: A double acting cylinder has two chambers and two ports for each chamber (see figure 5). The pressure dynamics of the cylinder can be written using the law of conservation of energy, assuming constant temperature T, and no heat exchange during the process:

$$\begin{cases} \dot{P}_1(p, u, V) = k\frac{RT}{V_1}\dot{m}_1 - k\frac{A\dot{x}}{V_1}P_1 \\ \dot{P}_2(p, u, V) = k\frac{RT}{V_2}\dot{m}_2 + k\frac{A\dot{x}}{V_2}P_2 \end{cases} \tag{8}$$

Equation 8 represents the velocity of the chamber pressure for each chamber [7]. The variable $\dot{x}$ is the velocity of the bore. V_1 and V_2 are the volumes of chambers one and two, respectively. $A = \pi(\frac{d}{2})^2$ is the area of the cylinder base. P_1 is the pressure in chamber one, and P_2 is the pressure in chamber two. The first term in the pressure dynamics equations in 8 is due to the flow from the valve, while the second one is due to compression from the piston.

$$V_1 = x.A \tag{9}$$

$$V_2 = (l - x).A \tag{10}$$

$$A = \pi(\frac{d}{2})^2 \tag{11}$$

It is assumed that the area of the rod is negligibly small compared to the area of the base of the bore, and so, that area is not subtracted from A in equation 10.

5) Force dynamics: There are four different forces considered in the two-chamber model that affect the position variable, x [10]. The first is the force obtained from the difference of pressures in both chambers:

$$F_1 = (P_1 - P_2).A \tag{12}$$

The second force corresponds to the sum of the dynamic friction and viscous forces:

$$F_2 = sgn(\dot{x}).k_d + k_v.\dot{x} \tag{13}$$

The third force, F_3, is the external force applied to the piston. The fourth force is the static friction, which counterbalances all other forces up to a threshold k_s:

$$F_4 = \begin{cases} -(F_1 + F_2 + F_3), & \text{if} \quad |F_1 + F_2 + F_3| < k_s \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

By applying the Newton's law, the model dynamics equation is obtained [10]:

$$\ddot{x} = \frac{1}{m}.(F_1 + F_2 + F_3 + F_4) \tag{15}$$

6) Computer Simulations: It is assumed that the model is frictionless, moving a load of constant mass, m. This hypothesis nullifies all forces except F_1, the one derived from the difference of chamber pressures. The set of simulation equations, already discretized, is the following [10]:

$$\ddot{x}_t = \frac{A}{m}.(P_1(t) - P_2(t)) \tag{16}$$

$$\dot{x}_{t+\delta} = \dot{x}_t + \ddot{x}_t.\delta \tag{17}$$

$$x_{t+\delta} = x_t + \dot{x}_t.\delta + \frac{1}{2}\ddot{x}_t.\delta^2 \tag{18}$$

$$\dot{m}_1(t) = a_{11}(t).\phi(P_c, P_1(t)) + a_{12}(t).\phi(P_r, P_1(t)) \tag{19}$$

$$\dot{m}_2(t) = a_{21}(t).\phi(P_c, P_2(t)) + a_{22}(t).\phi(P_r, P_2(t)) \tag{20}$$

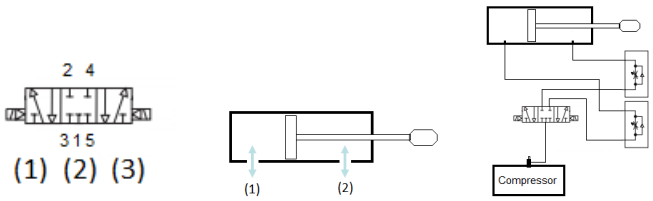
$$\dot{P}_1(t) = k\frac{RT\dot{m}_1(t) - A\dot{x}_t P_1(t)}{x_t A} \tag{21}$$

$$\dot{P}_2(t) = k\frac{RT\dot{m}_2(t) + A\dot{x}_t P_2(t)}{(l - x_t)A} \tag{22}$$

$$P_1(t + \delta) = P_1(t) + \dot{P}_1(t)\delta \tag{23}$$

$$P_2(t + \delta) = P_2(t) + \dot{P}_2(t)\delta \tag{24}$$

The process is a discretization in time, with a small time-step δ ([10]). Equations 16, 17, and 18 correspond to the acceleration, velocity and position of the bore, respectively. Pressures P_1 and P_2, as well as the position x_t and the velocity $\dot{x}_t$ are all state variables. Once they are updated, the chamber pressure variables are computed (equations 19 through 24). The variables a_{11} and a_{21} are the areas of the orifices of the valve that receive air from the compressor, whereas a_{12} and a_{22} are the areas of the exhaust ports of the valve. P_c and P_r are the pressures of the compressor and the room, respectively.



(a) (b) (c)

Fig. 6. States of the 5/2 solenoid valve, Double acting cylinder schematic, and assembled pneumatic setup

7) Pneumatic Setup: The Pneumatic Setup is composed of four components (i) Single direction speed controller, (ii) 5/2 double solenoid valve, (iii) double acting cylinder with rubber cushion, and (iv) compressor.

The 5/2 double solenoid valve is an electrical valve with three states (*5/2* means the valve has 5 ports and 2 states of air flow; the state with no air flow doesn't count as a state, in this naming convention):

- State (1) occurs when the left contact of the valve is electrically powered (with 24V DC). The air goes into the valve through port *1* and is expelled through port *5*. This air flow pushes the cylinder forward.
- State (2) occurs when both contacts of the valve are electrically powered, or when both contacts are OFF. The air flow is represented in the center of figure 6 (a): all ports are closed, and so there is no air flow. This makes the cylinder remain on its current position or decrease fast, to zero, the moving speed.
- State (3) occurs when the right contact of the valve is electrically powered. The air goes into the valve through port *1* and is expelled through port *3*. This flow pulls the cylinder back.

The direction speed controllers regulate the flow of air that reaches the cylinder. There is one for each port of the cylinder. It is possible to tune them to adjust the cylinder speed in both extension and retraction. However, if the flow is regulated to a very small speed, the friction from the rubber affects the system significantly. This situation is not favorable, and as such the tuning was made so that the cylinder didn't move too fast for measurements to be taken, but also not slow enough to make this friction considerable.

Figure 6 (b) shows the schematic of the double acting cylinder. When air flows into port *1* the cylinder extends itself. When air flows into port *2* the cylinder retracts itself. The full pneumatic setup is depicted in figure 6 (c).

C. Position Sensor Model and Identification

The Sensor measures the position of the cylinder, converting it to a voltage value that will be read by an Arduino Duemilanove. The sensor chosen was a Potentiometer of 5KΩ.

In theory, a potentiometer is a voltage divider, with a variable resistor, that depends on the position of the rotating contact [11]. In our setup the potentiometer is powered by a 15V DC source. The potentiometer's middle terminal returns the current voltage between the two resistors. There is a

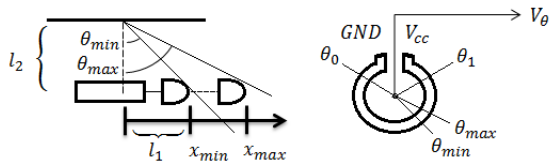


Fig. 7. Cylinder schematic (left) and Potentiometer schematic (right)

bar attached to the rotating contact of the Potentiometer that connects to the end of the cylinder.

The Potentiometer is not aligned with the end of the cylinder, and so the resting position of the bar is not vertical. The total variation of position of the cylinder is roughly 10cm. Whenever the cylinder moves, it will push or pull the bar, making the contact rotate, and thus, originating a new voltage value. This voltage value is read by the arduino (which is also powered by an external source of 5V DC; the arduino uses this voltage as reference, instead of the typical 5V it would get from the USB port of the computer, in order to reduce noise as much as possible). The arduino then returns a discretized voltage value based on its reference voltage value.

It is possible to create a map between voltage values, and their corresponding position values in centimeters. Two different approaches were chosen to do this. One of them is by using two measurements. The other is an automatic process, that uses several measurements.

Figure 7 illustrates the setup of the cylinder, the potentiometer and the bar. It also shows all of the system's variables. The voltage read by the sensor is an affine transformation of the potentiometer's angle, θ: $V_\theta = \frac{\theta-\theta_0}{\theta_1-\theta_0}.V_{cc} = a_0\theta + b_0$. As shown in figure 7, the angle θ has boundaries: $\theta \in [\theta_{min}, \theta_{max}]$. The position x is also limited: $x \in [x_{min}, x_{max}]$. Using the Pythagorean theorem, $x(\theta) = l_2.tan(\theta) - l_1$. If valid, the approximation $\tan(\theta) \simeq \theta$ would simplify this equation. It is possible to know the error being introduced by this approximation, using the actual values of the system: $l_1 = 3$[cm], $l_2 = 30$[cm], $x_{min} = 0$[cm] and $x_{max} = 10$[cm]. For $\theta_{min} = 0.099668$[rad], this approximation introduces an error $e_{\theta_{min}} = 3.3\%$. For $\theta_{max} = 0.4089$[rad], the error is $e_{\theta_{max}} = 5.97\%$. Therefore, the approximation is considered valid, and so, the movement of the cylinder is also an affine transformation of θ: $x(\theta) \simeq a_1\theta + b_1$. Finally, the composition of two affine transformations is still an affine transformation:

$$x \simeq a.V_\theta + b \tag{25}$$

It can be concluded from 25 that the position measurements (in centimeters) may be extracted directly from the voltage values (in Volts), without the need to know the value of θ.

Two pairs of values (x_i, V_{θ_i}) are sufficient to estimate the two unknown parameters, a and b.

If more pairs of values are available, it is possible to fit the data with linear regression by using, for example, Least Squares. Specifically, the residual is given by:

$$S = \sum_{i=1}^{n} [x_i - (aV_{\theta_i} + b)]^2 \tag{26}$$

where x_i is the position in centimeters, V_θ is the voltage value, a and b the unknown parameters. In matrix notation, for n pairs of values (x_i, V_{θ_i}):

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} V_{\theta_1} & 1 \\ V_{\theta_2} & 1 \\ \vdots & \vdots \\ V_{\theta_n} & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} \tag{27}$$

Equation 27 can be rewritten as:

$$\mathbf{x} = \mathbf{V}\alpha \tag{28}$$

The solution for the vector α (which contains the unknown parameters a and b) is given by the pseudo-inverse: $\alpha = (V^TV)^{-1}V^Tx$.

1) Calibration using two points: In this approach, two measurements are taken: when the cylinder is fully retracted, the voltage read corresponds to the minimum value, $V_{\theta_{min}}$; when the cylinder is fully extended, the voltage value corresponds to the maximum value, $V_{\theta_{max}}$.

It is assumed that the total excursion of the cylinder is 10cm, starting at 0cm. So, $x_{min} = 0$ and $x_{max} = 10$. Therefore, by applying the two pairs of values on equation 27 and computing the pseudo-inverse, the values a and b are obtained.

2) Calibration using several points: The procedure is similar to the calibration method described in III-C1. The calibration starts with the cylinder fully retracted. Then, a command is given to fully extend the cylinder. During this movement, all voltage values along time, $V_\theta(t)$, are collected. In this case, it is assumed that the cylinder moves at constant velocity. Also, the same assumption made in III-C1 for the values of x_{min} and x_{max} is valid in this calibration.

The data gathered can be represented by a graph starting with a constant value until time t_1, then a slope until time t_2, and again a constant value thereafter. Identifying t_1 and t_2 allows using the least squares line fit (the computations are the same as in III-C1, but instead of two pairs of values, there are several more).

D. Position Control Methodology

Subsection III-D is dedicated to explaining how the Position Control of the cylinder is actually done.

1) Control: method one: Whichever calibration is used, the line equation obtained will convert any reference given in centimeters by the user into a voltage value. This voltage value will then be compared to the voltage value given by the Potentiometer, since the latter corresponds to the cylinder's current position.

The position x is the value given by the sensor, and the reference x_{ref} is the value given by the user in the interface, converted to voltage value with the line equation. There is also a small threshold value ϵ, to allow the acceptance of a small window of values of x close to x_{ref}. When the condition $|x - x_{ref}| \leq \epsilon$ is true, x is within the accepted threshold, and

so a command to turn off the valve is issued. Otherwise, if the position x is smaller than $x_{ref} - \epsilon$, the left contact of the valve is turned on, and the right contact is turned off, so that the cylinder will be pushed. If x is bigger than $x_{ref} + \epsilon$, the right contact is turned on, and the left contact turned off, so the cylinder will pull back.

2) Control: method two: The following control method is a predictive algorithm. The goal of this method is to estimate the value of the next sample, $\hat{x}(n+1)$, at a given instant n. If the value of that sample falls within a certain range of the reference, the command to stop the valves is issued and the control loop ends. In other words, the purpose of this method is to stop the cylinder when it is estimated that it will reach the desired position, as opposed to giving that command only after the control loop as received the measurement and verified that the desired position has been reached. Since the information is available, this predictive control method uses the knowledge of both the current sample and the previous one.

Let e be the error between the current value measured $x(n)$ and the prediction of the current value $\hat{x}(n)$. Thus, the equation is given by 29. Let α be a regulation factor that is updated according to the ratio between the error e and the actual value $x(n)$, as shown in equation 30. If the predicted value is below the actual value, α increases. Otherwise, if the prediction is above the actual value, α decreases. The starting value for α is 1. Let δ be the difference between the last measurement $x(n)$ and the previous value $x(n-1)$, as given by equation 32. The predicted value for the next sample, $\hat{x}(n)$, is given by equation 32. This prediction is based on the assumption that the cylinder behaves as an integrator. If that assumption is valid, while the cylinder is moving, its slope will be the same for consecutive values, and if the position has increased by δ in the last two measurements, it is expected to increase by δ on the next sample, times the regulation factor α. This factor adjusts the precision of the prediction based on previous estimates, since it is updated on each iteration by comparing $\hat{x}$ with x, to make sure that the estimates remain as close as possible to the real values.

Finally, let ϵ be the threshold value. Since it is assumed that the error $e(n)$ follows a Gaussian distribution, using twice the absolute value of e corresponds to a confidence interval of about 95%. Since the predictions are usually very accurate, using only twice the absolute value of e would typically result in a very small threshold. Ideally, this value would actually be zero, should the prediction be equal to the measured value. So it was necessary to adjust it to a more flexible value. By adding $|\frac{\delta(n)}{2}|$ to the threshold, if the estimate falls within that margin, then it means we have found the closest point to the reference value we can possibly get based on the cylinder's current velocity. Thus, the final threshold ϵ is given by 33.

$$e(n) = x(n) - \hat{x}(n) \tag{29}$$

$$\alpha(n) = \alpha(n-1) * \left(1 + \frac{e(n)}{x(n)}\right) \tag{30}$$

$$\delta(n) = x(n) - x(n-1) \tag{31}$$

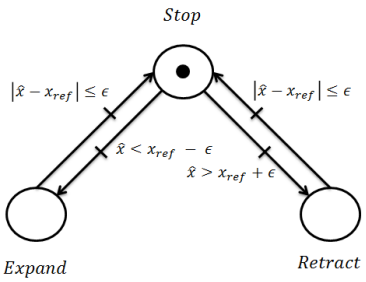


Fig. 8. Petri Net for the predictive control method.

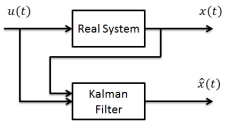


Fig. 9. Estimation of cylinder position, $\hat{x}$, using the Kalman Filter.

$$\hat{x}(n+1) = x(n) + \alpha(n) * \delta(n) \tag{32}$$

$$\epsilon(n) = 2 * |e(n)| + |\frac{\delta(n)}{2}| \tag{33}$$

The possible states of the valve while using the predictive control method may also be represented by a Petri Net. Figure 8 illustrates those states, using the criteria explained in the algorithm. Notice that, unlike the previous method, in this method the possibility of going directly from an 'Expand' state to a 'Retract' state and vice versa has been eliminated.

3) Control method three: the Kalman Filter approach: It is possible, even in cases where the theoretical model is carefully studied, that the behaviour of the real system does not entirely meet our expectancy. This may occur due not only to the interference of noise in the measurements, but also due to noise in our system. In such cases, it is plausible to resort to the Kalman Filter to provide estimates of the internal state of the system, taking into account the amount of noise from both the measurements and the process itself. Figure 9 shows the diagram of the system with the Kalman Filter. We provide both the reference signal $u_k(t)$ for the real setup and the measurement taken $x_{k-1}(t)$ to the Kalman Filter. With this information, it will be able to predict the next position.

$$\begin{cases} \frac{dx(t)}{dt} = A.x(t) + B.u(t) \\ y(t) = C.x(t) + D.u(t) \end{cases} \tag{34}$$

The set of equations 34 represents the state model of the system. In our scenario, the Kalman Filter produces a scalar of a pure integrator. This simplifies our state space model, since $A = 0$,$B = 1$, $C = 1$ and $D = 0$. Thus, we get:

$$\begin{cases} \frac{dx(t)}{dt} = b.u(t) \\ y(t) = c.x(t) \end{cases} \tag{35}$$

The Kalman Filter is usually conceptualized as two phases for each iteration: "predict" and "update" [12]. The "predict"

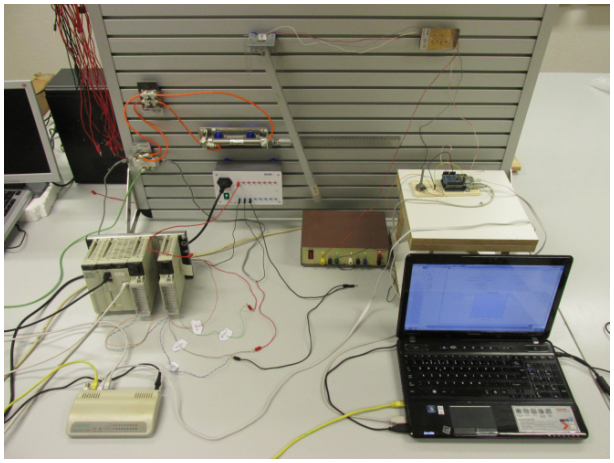


Fig. 10. Image of the full setup

phase corresponds to an *a priori* estimate of both the state $\hat{x}_k(t)$ and the covariance matrix $P_k(t)$:

$$\hat{x}_{k|k-1}(t) = \hat{x}_{k-1}(t) + u_k(t) \tag{36}$$

$$P_{k|k-1}(t) = P_{k-1}(t) + Q_k(t) \tag{37}$$

The update phase corresponds to an *a posteriori* update of both the state $\hat{x}_k(t)$ and the covariance matrix $P_k(t)$, based on the measurement residual $\tilde{y}_k(t)$ and the covariance residual $S_k(t)$. The measurement residual can be computed in the following way [12]: $\tilde{y}_k(t) = z_k(t) - \hat{x}_{k|k-1}(t)$. The covariance residual is obtained from [12]: $S_k(t) = P_{k|k-1}(t) + R_k(t)$. The variables Q_k and R_k represent the covariances of the system noise and observations noise, respectively. The variable $z_k(t)$ is our observation $x_k(t)$. The last step before we can update the state and the covariance matrix, is the introduction of the gain of the filter, the *Optimal Kalman gain*, which is given by [12]: $K_k(t) = P_{k|k-1}(t).S_k^{-1}(t)$. This gain, along with the measurement residual, have an influence on the state of the system, which can be mathematically expressed in the following manner [12]: $\hat{x}_{k|k}(t) = \hat{x}_{k|k-1}(t) + K_k(t).\tilde{y}_k(t)$. The gain also affects the covariance matrix [12]: $P_{k|k}(t) = (I - K_k(t)).P_{k|k-1}(t)$. The final equations for the *a posteriori* updates, are thus given by:

$$\hat{x}_{k|k}(t) = \hat{x}_{k|k-1}(t) + \frac{P_{k|k-1}(t)}{P_{k|k-1}(t) + R_k(t)}.(z_k(t) - \hat{x}_{k|k-1}(t)) \tag{38}$$

$$P_{k|k}(t) = \left(1 - \frac{P_{k|k-1}(t)}{P_{k|k-1}(t) + R_k(t)}\right).P_{k|k-1}(t) \tag{39}$$

The Petri Net representation for this Control method is the same as in figure 8. The difference lies on $\hat{x}$, which is given by the Kalman Filter, instead of the predictive method described in III-D2. The threshold ϵ used is a fixed value, like in III-D1.

A photo of the actual setup is presented in figure 10.

IV. Experiments and Results

Section IV discusses the simulations performed for a theoretical cylinder model system. Also, Section IV examines the results of the experiment of Control of the position of a pneumatic cylinder, when given a certain reference position by the user. In this experiment, all the topics approached throughout the thesis are present: PLC network, pneumatics, and Supervisor system.

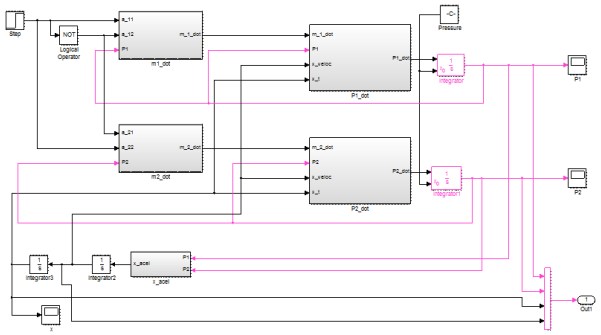


Fig. 11. Simulink model of the Pneumatic cylinder

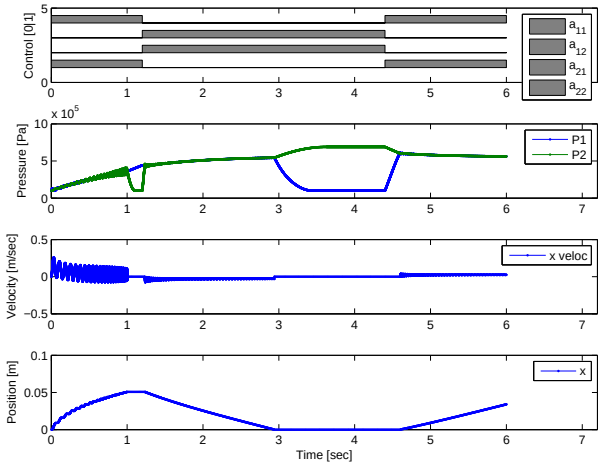
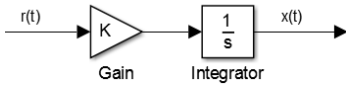


Fig. 12. Expansion from x_{min} to x_{max}, long-time retraction, expansion. Initial Chamber Pressures $P_1(t_0) = P_2(t_0) = P_r$.

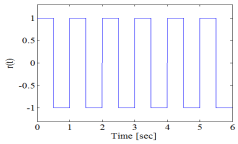
A. Pneumatic Cylinder Simulation

In order to simulate the pneumatic system, a simulink model was built, equivalent to the discrete model discussed in Section III-B6. This model is depicted in figure 11.

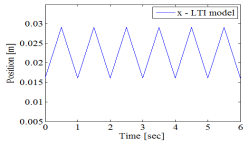
The simulation depicted in Figure 12 shows the result of the system for three consecutive commands: 'Expand', 'Retract', and 'Expand'. The chamber pressures P_1 and P_2 are initialized with the room pressure value, P_r. The initial position of the bore is x_{min}. On the first expansion, it is visible, at a certain point, that pressure P_2 starts dropping drastically, right after the position reaches x_{max}. This happens because P_2 is no longer increasing as a result of chamber two being compressed. Mathematically, the second term on equation 22 has become zero, and the first term is negative, because the air is being expelled from chamber two. Shortly after reaching x_{max}, the control command is set to 'Retract'. This command is purposely maintained for a while after x_t reaches x_{min}, to see what happens to the chamber pressures. As expected, pressure P_2 stabilizes at the compressor pressure P_c, and pressure P_1 stabilizes at room pressure P_r. This test indicates that, as the



(a) Simulink representation, considering constant gain K.



(b) Reference signal $r(t)$.



(c) Output Signal $x(t)$.

Fig. 13. LTI system approximation, block diagram, example input signal and respective output.

pressures on both chambers become higher, the step response x_t diverges from the non-linear curvy line, and draws nearer to a straight line.

B. LTI system approximation

Simulations of the pneumatic cylinder indicate that the double acting cylinder's behaviour may be approximated by a pure integrator, if certain terms are met. For example, keeping the pressure in both chambers high has been found to be a key condition. Simulations presented in IV-A show that one way to do this is by having the cylinder constantly expanding and retracting.

However, if the cylinder reached one of the limits of the chamber, the pressure in that chamber would decrease drastically rather quickly, affecting the slope of the step response. For this reason there is, implicitly, a second condition, to ensure that the first one is met, and to prevent a change in the slope: the cylinder must not be allowed to reach its extremities. Together, these conditions should allow the observation of a step response that can be approximated by a straight line.

To determine whether this hypothetical approximation was legitimate, one more simulation was prepared. The areas a_{11} through a_{22} are the control variables of the system. There are three possible control commands:

$$Expand <=> \begin{cases} a_{11} = max, & a_{12} = 0 \\ a_{21} = 0, & a_{22} = max \end{cases} \quad (40)$$

$$Retract <=> \begin{cases} a_{11} = 0, & a_{12} = max \\ a_{21} = max, & a_{22} = 0 \end{cases} \quad (41)$$

$$Stop <=> \begin{cases} a_{11} = 0, & a_{12} = 0 \\ a_{21} = 0, & a_{22} = 0 \end{cases} \quad (42)$$

In the simulation, the given commands oscillate between 'Expand' (equation 40) and 'Retract' (equation 41), each of them with a duration of 0.5 seconds. The 'Stop' command (equation 42) is not necessary for this specific simulation because, while it is true that it does not allow any air flow through the chambers' exhaust ports, it does not allow any air flow through the input ports either, so the pressures do not change at all.

The 'Expand' command is interpreted by the LTI system as a positive step signal, whereas the 'Retract' command is understood as a negative step signal. The full reference signal, $r(t)$, can be viewed in figure 13. The LTI system is essentially an integrator with a gain, just as depicted in figure 13. This figure was created in *Simulink*, but the program itself was run on a Matlab script, to allow some freedom in the calculation of the gain of the system.

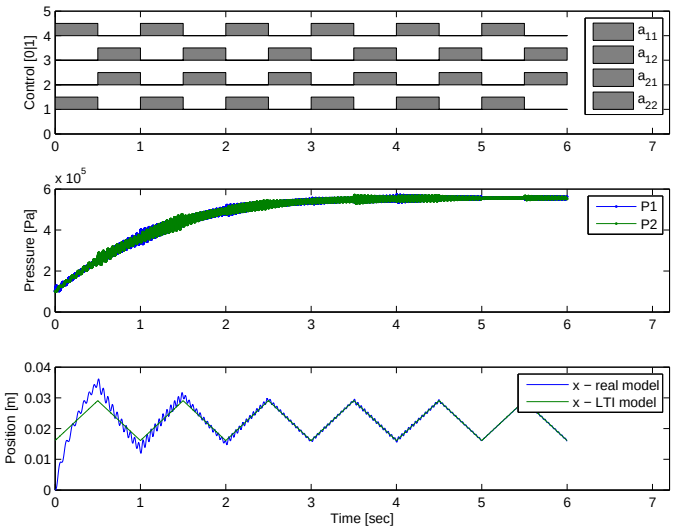


Fig. 14. LTI approximation simulation, real system input signal, real system pressures, real system and LTI system's positions.

The gain for the LTI system was calculated with equation $K = \frac{x(t_2)-x(t_1)}{t_2-t_1}$, which is based on the assumption that, after 5 seconds, the real system had already stabilized. In practical terms, this assumption means that, at a certain point in time, the chamber pressures are high enough on both chambers, and so, the step response for expansions and retractions will always be the same from that point forward, as long as the constraints remain respected. The position values for the real system, $x_t(t = 5)$ and $x_t(t = 5.5)$, were used to determine the gain of the LTI system.

Figure 14 shows the result of the aforementioned simulation. For the first 1.5 seconds, the approximation is not very accurate, which is consistent with the hypothesis, because the initial pressure values used on both chambers were low ($P_1(0) = P_2(0) = P_r = 99974.02$ Pascals), and the system was not given enough time to let those pressures increase significantly. Nonetheless, it can be seen that, after some expansions and retractions, as the pressures grow, the approximation becomes more accurate.

As one may observe in figure 14, the pressures in both chambers ascended to a much higher value than P_r, and seem to have stabilized at a value close to but yet smaller than P_c ($P_c = 689476$ Pascals).

The result observed here, i.e, that high pressures change the step response from non-linear to linear, is associated to a phenomenon called *Stiffness* [10] [7]. It is usually used as a form of control for pneumatic actuators due to the fact that it makes them resistant to exterior forces, such as gravity.

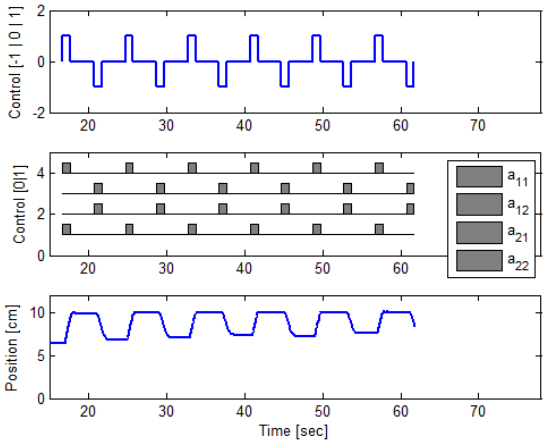


Fig. 15. Step response test

C. Step response

In order to understand how close to the expected behavior a real double acting cylinder operates, a test to the real system step response was performed.

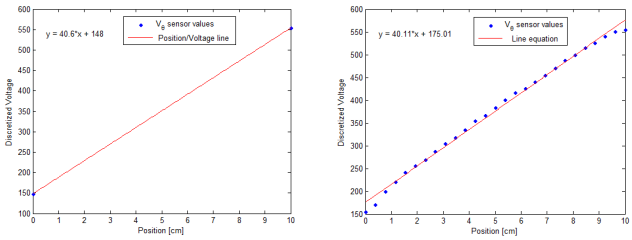
Figure 15 shows the result of the test. The first signal (labeled "Control $[-1|0|1]$") was generated to grant some insight about the commands being sent to the valve, namely in terms of starting time and duration. Value '1' means the command is 'Expand'. Value '-1' represents the command 'Retract'. Lastly, value '0' indicates the 'Stop' command. These commands are consistent with the ones explained in IV-B.

The second signal shows which ports of the valve are open. The representation is the same as in Sections III-B6 and IV-B: during an 'Expand' command, ports a_{11} and a_{22} are open. Ports a_{12} and a_{21} are open during a 'Retract' command. Finally, if the command is 'Stop', all ports are closed.

The third signal is a display of the position of the cylinder, in centimeters. The acquisition of the data starts after a series of 'Expand' and 'Retract' commands, carefully timed so that the cylinder would not reach either of its extremities. The reason behind this choice is to witness the effect of *stiffness* (discussed in IV-B) on the system's output. This is also why the cylinder does not start from an initial position of 0cm.

The results shown in figure 15 allow the withdrawal of some conclusions worth discussing. Despite the fact that the cylinder purposely reached the extremity of 10cm, there is no evidence that this occurrence had any effect on the output. The simulations examined in III-B6 suggested that, should the cylinder touch either of its extremities, the pressure in the chamber matching the extremity would decrease drastically, and thus, the *stiffness* effect would be suppressed. However, on the real system, the exhaust ports are different from the supply ports, since their areas can be manually adjusted. Also, it should be reminded that the exhaust ports are only open while one of the valve's contacts is active (if both contacts are active or inactive, all ports are closed).

In any case, the cautiousness enforced in the commands given prior to the acquisition of the data (i.e, the sequence of 'Expand/Retract' commands to try to increase the pressures on both chambers without touching the extremities), was probably not necessary, because there is no visible impact on the output suggesting that the *stiffness* is changing significantly when an extremity is touched.



(a) (b)

Fig. 16. Line equations, obtained from two data points, and from several data points using Least Squares

On a different note, it is noticeable that the velocity during an expansion is higher than the velocity during a retraction. Hence, after each periodic wave in the input signal, the position increases. This phenomenon is due to the rod connected to the piston. Throughout all of the simulations presented in III-B6 and IV-B, it was always assumed that the area of the rod was negligibly small, so its effect on the output of the system was never witnessed in those cases. However, on the real system, the rod's area is approximately half the area of the piston, so its effect is quite considerable. In fact, if the single direction speed controllers had not been placed on the setup, and had not been manually tuned to try to balance the velocities of expansion and retraction, the difference between them would be even more visible here.

It will be considered, given the results shown above, that the approximation of the step responses to straight lines is admissible. As such, when sending an input $x(t) = u(t)$, the system's output is $y(t) = t$. By applying the Laplacian Transform, the transfer function of the system will be obtained: $H(s) = \mathcal{L}\{y(t)\}/\mathcal{L}\{x(t)\} = (1/s^2)/(1/s) = 1/s$. This system has a single pole at the origin, i.e. it is a pure integrator.

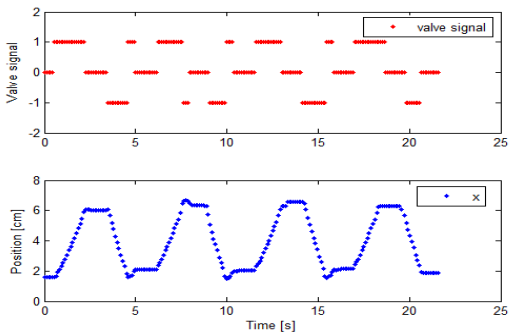
Considering what was shown in IV-B, if one assumes a frictionless model, coupled with the stiffness effect, the cylinder behaves as an integrator, with no possibility of external disturbance, unless the force is substantial enough to counteract the stiffness.

D. Calibration using two points versus Calibration using several points

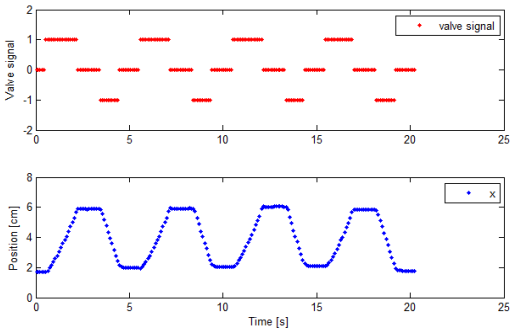
As previously stated in Sub-Section III-C1, two measurements are enough to produce a map of correspondences between voltage values and position values.

Figure 16 (a) displays the map between voltage values and position values, in centimeters, that was obtained, using the method explained in III-C1. For the values $V_{\theta_{min}} = 148$ and $V_{\theta_{max}} = 554$ (discretized voltage), obtained after a full expansion, the resulting values for the line equation $y = a * x + b$ are: $a = 40.6$ and $b = 148$.

As explained in III-C2, the other calibration method consists of collecting several points, assigning each of them to a specific position, in centimeters. The assumption that the



(a)



(b)

Fig. 17. (a): System response to a square wave between 2 and 6 cm, using Control method one. (b): System response to a square wave between 2 and 6 cm, using Control method two. In both results, the top plot shows the control signal and the bottom plot shows the position of the cylinder.

cylinder moves at a constant velocity allows the matching between voltage values V_θ and positions [cm] to be made based on the sampling time, which is also constant. Once the correspondence is made, the pseudo inverse may be computed, since both vectors, $\mathbf{x}$ and $\mathbf{V}$, are available (equation 28).

Figure 16 (b) shows the line equation obtained, using the aforementioned method. The sampling time chosen for the data collection was 100 milliseconds. The time for a complete excursion of the cylinder to occur was approximately 2.6 seconds. The result of the pseudo inverse is also visible in figure 16 (b): $a = 40.11$ and $b = 175.01$.

E. Closed-loop

Section IV-E is dedicated to the presentation and analysis of the results obtained with the empirical model, using the control methods described in Sections III-D1, III-D2 and III-D3. To demonstrate the accuracy of the control methods, it was decided to use a variable reference signal, particularly a square wave between 2 [cm] and 6 [cm]. The same reference signal is used on all of the control methods.

Figure 17 shows the result produced by the system using method one (III-D1) to control the position of the cylinder, x. The $valvesignal$ follows the same convention used in IV-C: value ”1” corresponds to an ’Expand’ command, value ”-1” to a ’Retract’ command, and value ”0” to a ’Stop’ command.

It is visible in figure 17 (a) that, while the cylinder is able to stop within the margin of the given threshold (0.5cm above or below the reference value), it often exceeds this margin on the

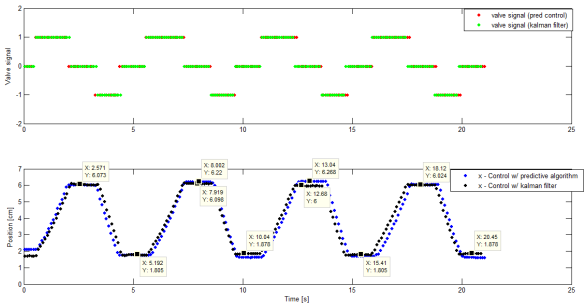


Fig. 18. Comparison between Control method two, and Control method three (Kalman Filter).

first attempt, and needs to correct it, involving the necessity of sending more commands to the PLC. This *overshoot* happens because of the delays inherent to the assembled setup. Specifically, the time it takes for the SCADA system to receive the measurement, the processing time of the current position of the cylinder in the control loop, and the command based on that evaluation actually arriving to the PLC and be executed, are all examples of small delays of the system. So, when a given command reaches the PLC, the position of the cylinder is no longer the same, because some time has passed and the cylinder has kept moving. This situation is not ideal, and so, another method of control was implemented to try to eliminate the overshoot by counterbalancing the delay.

Figure 17 (b) displays the output of the system for the same square wave, using the control method two. Since this method evaluates the predicted value of the next sample, the impact of the delays of the system is effectively reduced. In fact, the purpose of this predictive control is to nullify the delay as much as possible by sending the ’Stop’ command to the cylinder before it actually reaches the desired reference. It is assumed here that the delay between sending the ’Stop’ command and the cylinder actually stopping is approximately one sample. The command is given when the predicted value of the next sample is inside the accepted threshold. This way, as long as the estimated value is accurate, and the delay is approximately one sample, the cylinder will actually stop where it is intended. The result shown in figure 17 (b) corroborates these assumptions.

Lastly, figure 18 shows the comparison between the results of the Control method two, and the Control method using the Kalman Filter. The former method presents a maximum error of 2.68% on the upper reference value (6 [cm]), and a maximum error of 3.3% on the lower reference value (2 [cm]). The latter presents a maximum error of 1.72% on the upper reference, and 2.69% on the lower reference. Overall, relying on both the measurements and the knowledge of the system model is more effective than relying solely on the measurements.

V. Conclusions

The results of the experiments were shown and discussed. As intended, an industrial setup controllable by Programmable Logic Controllers and managed by a SCADA system, was

implemented. The goal of controlling the position of the cylinder with this developed system, was also accomplished.

The applicability of this kind of system is wide. It can be adapted to any automated process in which forward and backward movement is necessary. An example of an industrial application could be to use the cylinder with a hose attached at the extremity, and filling a set of aligned bottles, regardless of the space between each bottle in that line.

This system may also be used in Robotics, for example, as a joint of a robot. In this case, the acceleration during movement when a force is applied, and the precision of the position when that movement ends, may be important factors to analyze, depending on the goal of the implementation.

For future work, some ideas are proposed. The first proposal is to assemble the empirical system with a different network. Specifically, the system should have the sensor sending messages directly to the PLC. This will avoid or at least lessen the delays of the old system.

It is also suggested that another method of control for the closed loop is implemented. An Optimal Control approach is a recommendation.

REFERENCES

[1] K. T. Erickson, "Programmable logic controllers." Institute of Electrical and Electronics Engineers, 1996.

[2] R. David and H. Alla, "Petri nets and grafcet: tools for modelling discrete event systems," 1992.

[3] Wikipedia, "Sequential function chart — wikipedia, the free encyclopedia," 2014, [Online; accessed 18-August-2014]. [Online]. Available: http://en.wikipedia.org/w/index.php?title=Sequential_function_chart&oldid=596313068

[4] ——, "Scada — wikipedia, the free encyclopedia," 2014, [Online; accessed 18-August-2014]. [Online]. Available: http://en.wikipedia.org/w/index.php?title=SCADA&oldid=622082724

[5] "Modbus-ida.org," MODBUS Frequently Asked Questions, Dec. 6 2006. [Online]. Available: http://www.modbus.org/faq.php

[6] "Tsx p57 processors implementation manual vol1.pdf," Schneider Electric Telemanique, Jul. 7 2008. [Online]. Available: http://www.schneider-electric.com/download/hk/en/details/2247279-TSX-57PCX-57-Processors-Implementation-Manual-Volume-1/?reference=35011052K01000

[7] Y. Tassa, T. Wu, J. Movellan, and E. Todorov, "Modeling and identification of pneumatic actuators," in *Mechatronics and Automation (ICMA), 2013 IEEE International Conference on*. IEEE, 2013.

[8] D. Sexton and A. Lacy, "Method for communicating among a plurality of programmable logic controllers each having a dma controller," Dec. 10 1991, uS Patent 5,072,374. [Online]. Available: https://www.google.com/patents/US5072374

[9] K. Ogata and Y. Yang, "Modern control engineering," 1970.

[10] J. Movellan, "A pneumatic cylinder model," *Machine Perception Laboratory Tutorials, http://mplab. ucsd. edu*, 2009.

[11] Wikipedia, "Potentiometer — wikipedia, the free encyclopedia," 2014, [Online; accessed 10-October-2014]. [Online]. Available: http://en.wikipedia.org/w/index.php?title=Potentiometer&oldid=629005553

[12] ——, "Kalman filter — wikipedia, the free encyclopedia," 2015. [Online]. Available: http://en.wikipedia.org/w/index.php?title=Kalman_filter&oldid=665279693