FUNDAMENTALS OF ROBOTICS

Lab exercise

Stäubli

AULINAS Josep (u1043469)
GARCIA Frederic (u1038431)
Introduction

The aim of this tutorial is to give a brief overview on the Stäubli Robot System describing its main components, how to manipulate it using the teach pendant and finally how to program it offline and execute the programmed application.

The goals are:

- learn the basics about Staubli system
- get some basic training on the use of the teach pendant
- program a simple application
- run the application on the robot

Part 1: Theoretical concepts

P1.1. System overview

The system is composed by four main elements (Figure 1.1): the manipulator, the controller, the teach pendant (SP1) and the Stäubli studio in a PC. The controller is the central element connected to the rest of the elements. The programmer can send orders to the manipulator directly using the teach pendant, which sends the user requested operations to the PLC and this transmit the order to the robot. However, a better procedure is to program with VAL3 language a sequence using the Stäubli Studio, which is a programmer interface. Then this code is transferred to the control unit and finally sent to the robot.

Figure 1.1: Stäubli system in robotics laboratory.
P1.2. Controller

The control SC8C (Figure 1.2) is composed by a calculator device, which is the intelligent part of the system. This device guides the robot through dedicated power amplifiers for each axis of the manipulator. The control unit cycle behaves as follows:

- Read inputs and outputs.
- Execute one program cycle
- Write outputs
- Waits to the end of the specified cycle length.

Figure 1.2: control unit of the robot

Figure 1.3 shows the main technical characteristics of the controller. Note the programming language, which is VAL3, and the communication system, which can be serial and Ethernet.

<table>
<thead>
<tr>
<th>TECHNICAL CHARACTERISTICS</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Dimensions: H x L x D</td>
<td>331 x 445 x 455 mm</td>
</tr>
<tr>
<td></td>
<td>481 x 445 x 455 mm</td>
</tr>
<tr>
<td>Memory capacity</td>
<td>64 MB de RAM (min.)</td>
</tr>
<tr>
<td>Backup</td>
<td>64 MB (min.) Compact Flash Disk and USB Port</td>
</tr>
<tr>
<td>System/Programming language</td>
<td>VAL 3 (multitask interpreted language)</td>
</tr>
<tr>
<td></td>
<td>PLC IEC 61131-3 language</td>
</tr>
<tr>
<td>Communication</td>
<td>RS232/422 serial link Modbus server - Ethernet</td>
</tr>
<tr>
<td>Inputs/Outputs (IO)</td>
<td>1 or 2 boards 16/16 digital inputs/outputs, optional</td>
</tr>
</tbody>
</table>

Figure 1.3: technical characteristics of the controller
P1.3. Teach pendant

The Teach Pendant is a hand-held robot control terminal that provides a convenient means to move the robot, teach locations, and run robot programs. The teach pendant is a useful tool, allowing the user to move away from the host computer terminal and control the robot locally. Typically a robot application is programmed with teachable variables used later in the programming language. Once the application is setup, it is convenient to use the teach pendant to teach the locations for the application. Figure 1.4 shows the teach pendant, and also in the left image the security button is pointed (#).

![Figure 1.4: teach pendant back, in the left; teach pendant front in the right.](image)

P1.4. Manipulator

The Stäubli TX60 series (Figure 1.5) is a six rotational degrees of freedom manipulator, whose main features and benefits are:

- Arm integral casing
- Rigid Mechanical structure
- Spherical work envelope
- Compact and robust wrist
- User’s connections near to the tool
- High speeds and accelerations
- High accuracy & repeatability
- Compact footprint
- High dexterity
- High reliability & low maintenance

![Figure 1.5: Stäubli manipulator](image)
The manipulator consists of segments or members interconnected by joints. The movements on the arm joints are generated by servomotors coupled with position sensors. Each of these motors is equipped with a parking break. This is a reliable and robust system that allows the absolute position of the robot to be known at all times. The arm assembly is sufficiently flexible and is able to perform a great variety of applications, such as, handling of loads, assembly, process, application of adhesive beads and control/check and clean room applications.

The various elements of the robot’s arm are (Figure 1.6): the base (A), the shoulder (B), the arm (C), the elbow (D), the forearm (E) and the wrist (F). Also the degrees of freedom are shown in figure 1.6. Figure 1.7 presents the main robot specifications. Figure 1.8 and 1.9 shows the working range of the robot.

<table>
<thead>
<tr>
<th>STAUBLI TX60 SPECIFICATIONS</th>
<th>TX60 (Standard Reach)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Degrees of Freedom</td>
<td>6</td>
</tr>
<tr>
<td>Nominal Load Capacity</td>
<td>3.5 kg</td>
</tr>
<tr>
<td>Maximum Load Capacity</td>
<td>9 kg</td>
</tr>
<tr>
<td>Reach at Wrist</td>
<td>670 mm</td>
</tr>
<tr>
<td>Repeatability</td>
<td>±0.02 mm</td>
</tr>
<tr>
<td>Protection Class (*Wrist)</td>
<td>IP 65 (*IP67)</td>
</tr>
</tbody>
</table>

Figure 1.6: Stäubli schematic.

Figure 1.7: Stäubli main specifications

Figure 1.8: Front working range view.

Figure 1.9: Top working range view.
P1.5. Getting started: Teach pendant training

Figure 1.10 shows the teach pendant image, with all the buttons and its functions. Some of them are described, at least the ones that will be use in this tutorial.

![Teach Pendant Image](image)

**Figure 1.10:** teach pendant.

(1) **Mode selection**

- **Local mode:** the robot executes an application stored in its controller.
- **Displaced Mode:** the robot works
- **Manual Mode:** the user can control the robot manually selecting the displacement mode described in (5).
- **Testing Mode:** the user can run a certain task step by step.

(2) **Power on / off**

This button activates the power supply of the robot. It indicates power on when the surrounding light is on.
(3) **Emergency stop**

The emergency stop must be pushed in order to block the robot, no matter the task it is performing or the actual configuration.

(4) **Movement buttons**

These buttons allows the user to move the robot according to the displacement mode (5).

(5) **Selecting displacement mode**

![Displacement modes](Image)

- **Joint**: move the robot with respect to its joint axis
- **Frame**: move the robot with respect to its base reference system
- **Tool**: move the robot with respect to the tool coordinate system.
- **Point**: move the robot to a certain point.

(6) **Speed control**

To control the speed manually.

(7) **Contextual menu**

Used to validate the information given just up the buttons in the screen

(8) **Alfa numeric keyboard**

Useful to enter application data.

(9) **Interface and navigation buttons**

Typical interface with up, down, left and right buttons, as well as Esc, Help, Menu selector buttons.

(10) **Applications Command**

This are the buttons that allows to run a programmed application.

- **Move** Hold: is the play/ pause button.
- **Run**: is the load application to be run button
- **Stop**: terminates the current application procedure.
Part 2: Practical exercise

P2: 1. Stäubli Robotics Studio

Figure 2.1 shows the Stäubli Robotics Studio which allow the user to perform any kind of operation over the Stäubli Robot, such as execute VAL3 Studio to modify or create a new code or execute the Transfer Manager to send the code to the Robot’s control unit or from the Robot’s control unit to the computer.

Concretely, during this practical exercise only two of the modules presented in the Stäubli Robotics Studio have been used, VAL3 Studio and Transfer Manager.

P2: 2. VAL3 Studio

VAL3 Studio is a visual environment (view Figure 2.2) to develop applications for Stäubli Robots.

VAL3 is a high level language created to manipulate Stäubli robots in industrial applications and palletizing environments. VAL3 integrates tools to control the robot, the geometric modelization and the i/o.
The interface can be divided into four main sections:
1. Toolbars: Allow the user to execute the standard operations of every interface, such as generate a new file, open an existing file, copy or paste. Besides, the tool bar allows the user to compile the code or to transfer it to the robot’s control unit.

2. From this section it is possible to change from one workspace to another. Workspaces can be:
   a. Programas: Consist on the VAL3 editor, where the user can write or modify their own code (view Figure 2.2).
   b. Librerías: This tab allow the user to add or modify libraries to the code.
   c. Variables: This tab shows the variables to be used in the code. VAL3 doesn’t allow to declare variable in the code, then, it is necessary to declare all the variables in this section, otherwise a compiler error will be show.

P2: 2.1. VAL3 Studio: Variables

As has been mentioned in the previous section, all the variables which appear in the VAL3 code have to be declared in the Variables section (view Figure 2.4).

![Figure 2.4. Variables section and declaration.](image)

VAL3 defines the following variable types:

- **dio**: this kind of variable allow to link a VAL3 variable to a system i/o. In our example, **dGripper** is a dio variable to make a link between this variable and the valve1 of the Robot setup.

```val3
dioLink(dGripper, io:valve1)
```
- **flange**: any reference to the robot’s tool need to be a flange type variable.

![Flange type variable.](image)

- **joint**: any reference to the robot’s joints need to be a joint type variable.

- **mdesc**: this variable type is related with the robot’s acceleration and velocity specifications. This variable type allow to define the acceleration, deceleration and velocity for a defined movement.

![Mdesc type variable.](image)

- **num**: this variable type allow the user to define variables which have to store integer values, such as counters.

- **world**: this variable type defines a world position, a specific robot configuration. In our case, these variables have been declared using the touch pendant and saving the desired configuration of the robot.

![World type variable.](image)

**P2: 3. VAL3 Code: Cleaning the blackboard**

The practical exercise consists on moving the robot from the initial position to a specific position on the table. Take the cleaner and move to the blackboard, then clean it and finally move to the end position. The environment of the lab is shown in Figure 2.8 and 2.9.

In this practical exercise there are different kind of movements, such as lineal movements or angular movements, rotations of the joints, tool control, acceleration, deceleration and velocity control depending on the movement, etc.
Figure 2.8. Stäubli control and computer in robotics laboratory.

Figure 2.9. Stäubli robot in lab environment.
The VAL3 code is:

```
Begin
  // Go to initial position
  jDest = {0,0,0,0,0,0};
  movej(jDest,flange,mNomSpeed)
  waitEndMove()
  // Go to position 1: to take the bar
  // Move to the security position before taking the cleaner
  movej(p2,flange,mNomSpeed)
  waitEndMove()
  // Move to the cleaner (linear movement)
  movel(p1,flange,mNomSpeed)
  waitEndMove()
  // Take the cleaner...
  //dioLink(dGripper, io:valve1) // Connect dgripper to the io of the valve1 system
  //oTime=1 // Time to open the tool (seconds)
  //cTime=1 // Time to close the tool (seconds)
  // Take the cleaner
  close(tGripper)
  delay(cTime)
  open(tGripper)
  delay(oTime)
  // Move to the security position after taking the cleaner (linear movement)
  movel(p2,flange,mNomSpeed)
  waitEndMove()
  // Move to the initial position rotating the tool
  jDest.j6 = -270
  movej(jDest,flange,mNomSpeed)
  waitEndMove()
  jDest.j6 = 0
  movej(jDest,flange,mGirTool)
  waitEndMove()
  // Move to the security position before cleaning the blackboard
  movel(p3,flange,mNomSpeed)
  waitEndMove()
  // Move to the blackboard
  movel(p4,flange,mNomSpeed)
  waitEndMove()
  // Clean the blackboard (Cleaning...)
  movel(p5,flange,mCleanBB)
  waitEndMove()
  movel(p4,flange,mCleanBB)
  waitEndMove()
  movel(p5,flange,mCleanBB)
  waitEndMove()
  movel(p4,flange,mCleanBB)
  waitEndMove()
  movel(p5,flange,mCleanBB)
  waitEndMove()
  // Move to the security position after cleaning the blackboard
  movel(p3,flange,mNomSpeed)
  waitEndMove()
  // Move to the end position
  movej(p6,flange,mNomSpeed)
  waitEndMove()
  // Give me the cleaner
  open(tGripper)
  delay(oTime)
  close(tGripper)
  delay(cTime)
end
```

VAL3
Few handicaps appeared when it was necessary to close the gripper of the robot. After declare a tool type variable, to open or close the gripper it is necessary to specify always the actual state of the gripper and then the new state (p.e. if the gripper is opened and we want to close it, it is necessary to specify close the gripper and after that open the gripper). We think that open is equal to open the valve, then the gripper will close and close is equivalent to close the valve then the gripper will be open. Besides, it is necessary to specify a delay after close() or open() execution.

In order to provide the robot with pneumatic power, the valve shown in Figure 2.10a must be switched on as shown in Figure 2.10b.

![Figure 2.10. a) pneumatic system closed; b) pneumatic system open.](image)

**P2: 4. Transfer Manager**

When the code is finished and compiles without errors, it can be transferred to the robot by the tool bar button of the VAL3 Studio which will open the Transfer Manager, mentioned in the former section or directly selecting the Transfer Manager from the Stäubli Robotics Studio (view Figure 2.1).

To transfer a code from the PC to the control unit it is only necessary to select the file from the PC and select transfer (Transferir) (view Figure 2.11).
On the other hand, if the transfer must be from the control unit to the PC (e.g. to save the robot configuration, world type variables) then the selected file will be in the control unit section (view Figure 2.12).