ROBOT NEUMÁTICO CARTESIANO PARA EL APRENDIZAJE DE LOS LENGUAJES DE PROGRAMACIÓN EN ROBÓTICA

JULIO CÉSAR GALVIS CHACÓN
DANIEL ALBERTO SIERRA GALLO

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICOMECÁNICAS
ESCUELA DE INGENIERÍA MECÁNICA
BUCARAMANGA
2014

ROBOT NEUMÁTICO CARTESIANO PARA EL APRENDIZAJE DE LOS LENGUAJES DE PROGRAMACIÓN EN ROBÓTICA

JULIO CÉSAR GALVIS CHACÓN
DANIEL ALBERTO SIERRA GALLO

Trabajo de grado para optar al título de
INGENIERO MECÁNICO

Director
JORGE ENRIQUE MENESES FLOREZ
Ingeniero Mecánico, Msc.

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICOMECÁNICAS
ESCUELA DE INGENIERÍA MECÁNICA
BUCARAMANGA
2014

DEDICATORIA

A Dios por la ayuda recibida en todo momento.

A mi mamá por el apoyo recibido durante toda la carrera.

Daniel Alberto Sierra Gallo

A Dios, por ser mi fuerza espiritual.

A mis padres Julio y Beatriz que lo han dado todo para el logro de mis metas.

A Javier y Alejandra por acompañarme en esta etapa de mi vida.

Julio César Galvis Chacón

AGRADECIMIENTOS

A nuestro director de proyecto el ingeniero Jorge Enrique Meneses Florez, por su compromiso, guía y aporte de conocimientos para el desarrollo del proyecto; también a nuestros compañeros del laboratorio de automatización industrial por su constante apoyo durante la realización de todo el proyecto.

A todas aquellas personas que de manera directa o indirecta contribuyeron al desarrollo de este proyecto.

CONTENIDO

pág.

INTRODUCCIÓN 22

1. FORMULACIÓN DEL PROBLEMA 24

1.1 IDENTIFICACIÓN DEL PROBLEMA 24

2. JUSTIFICACIÓN 26

3. OBJETIVOS 28

3.1 OBJETIVO GENERAL 28

3.2 OBJETIVOS ESPECÍFICOS 28

4. ROBOT NEUMÁTICO CARTESIANO: DESCRIPCIÓN DEL PROYECTO 29

4.1 COMPARACIÓN DESDE EL HARDWARE: 33

4.2 COMPARACIÓN DESDE EL SOFTWARE 36

4.3 DESARROLLO DEL PROYECTO DE GRADO 40

5. ROBOT NEUMÁTICO: SISTEMAS DE CONTROL 44

5.1 LAZO DE CONTROL: SUCCIÓN DE ELEMENTOS 44

5.2 LAZOS DE CONTROL: MOVIMIENTO DEL ROBOT 47

6. ROBOT NEUMÁTICO: INTERFAZ – VENTANA DE DATOS – PROGRAMA EN EL AUTÓMATA 79

6.1 SISTEMA INICIAL DE CALIBRACIÓN 80

6.2 SISTEMA GUIADO ACTIVO 82

6.3 LENGUAJE TEXTUAL NIVEL ROBOT 97

6.4 LENGUAJE TEXTUAL NIVEL TAREA 118

6.5 LENGUAJES DE PROGRAMACIÓN DESARROLLADOS EN EL PROYECTO: DEMOSTRACIÓN COMO MODELOS VÁLIDOS PARA EL APRENDIZAJE 128

7. ROBOT NEUMÁTICO: MANUAL DE USUARIO 152

7.1 MANUAL DEL USUARIO: CAPÍTULO 1 153

7.2 MANUAL DEL USUARIO: CAPÍTULO 2 154

7.3 MANUAL DEL USUARIO: CAPÍTULO 3 154

7.4 MANUAL DEL USUARIO: CAPÍTULO 4 155

8. CONCLUSIONES 157

9. RECOMENDACIONES 159

BIBLIOGRAFÍA 160

ANEXOS 162

LISTA DE FIGURAS

pág.

Figura 1 Robot/Manipulador cartesiano primer proyecto de grado 31

Figura 2 Manipulador Cartesiano segundo proyecto 31

Figura 3 Robot neumático cartesiano proyecto actual 32

Figura 4 Composición robot neumático 33

Figura 5 Comparación punto de vista HARDWARE 34

Figura 6 Software interfaces Manipulador 37

Figura 7 Software Robot en el autómata 38

Figura 8 Interfaces Robot para el computador 39

Figura 9 Descripción general del proyecto 41

Figura 10 Sistema de succión 45

Figura 11 Control sistema de succión 46

Figura 12 Sistema de control movimiento 47

Figura 13 Ejes en el robot cartesiano 48

Figura 14 Modelo dinámico 50

Figura 15 Actuador lineal artículo 52

Figura 16 Apertura válvula como función de la señal 56

Figura 17 Posiciones válvula proporcional 57

Figura 18 Posición contra tiempo sistema neumático 59

Figura 19 Velocidad constante en el sistema neumático 60

Figura 20 Velocidad promedio para la válvula proporcional del eje X en diferentes condiciones 62

Figura 21 Velocidad promedio para la válvula proporcional del eje Y en diferentes condiciones 62

Figura 22 Velocidad promedio para la válvula proporcional del eje Z en diferentes condiciones 63

Figura 23 Tiempo inicial eje X para varios estados 63

Figura 24 Tiempo inicial eje Y para varios estados 64

Figura 25 Tiempo inicial eje Z para varios estados 64

Figura 26 Recorrido final eje X para varios estados 65

Figura 27 Recorrido final eje Y para varios estados 65

Figura 28 Recorrido final eje Z para varios estados 66

Figura 29 Procesos del lazo de control 67

Figura 30 Bloque de cálculos 1 68

Figura 31 Bloque de cálculos 2 69

Figura 32 Bloque de cálculos 3 70

Figura 33 Bloque de cálculos 4 71

Figura 34 Bloque de cálculos 5 72

Figura 35 Bloque de cálculos 6 73

Figura 36 Bloque de cálculos 7 74

Figura 37 Bloque de cálculos 8 75

Figura 38 Ciclo de movimiento 1 75

Figura 39 Ciclo de movimiento 2 76

Figura 40 Ciclo de movimiento 3 76

Figura 41 Bloque de cálculos segundo control 77

Figura 42 Ciclo de movimiento segundo control 1 78

Figura 43 Ciclo de movimiento segundo control 2 78

Figura 44 Robot neumático cartesiano sistema final 79

Figura 45 Estructura de trabajo Robot neumático 80

Figura 46 Interfaz inicial: proceso de calibración 81

Figura 47 Programa inicial de calibración en el autómata 82

Figura 48 Joystick sistema guiado activo 83

Figura 49 Estructura interfaz del lenguaje guiado activo 84

Figura 50 Visualización remota robot en el sistema guiado 86

Figura 51 Sistema guiado activo: programa en el autómata 1 95

Figura 52 Sistema guiado activo: programa en el autómata 2 96

Figura 53 Sistema guiado activo: programa en el autómata 3 96

Figura 54 Sistema guiado activo: programa en el autómata 4 97

Figura 55 Estructura lenguaje textual nivel robot 97

Figura 56 Monitoreo externo editor textual nivel robot 99

Figura 57 Editor textual para el lenguaje nivel robot 100

Figura 58 Interfaz para los cálculos 102

Figura 59 Estructura código escrito en el lenguaje textual nivel robot 104

Figura 60 Proceso general de la función principal 109

Figura 61 Definición de variables 111

Figura 62 Estructura de una asignación 112

Figura 63 Lenguaje textual nivel robot: programa en el autómata 117

Figura 64 Espiral cuadrada 119

Figura 65 Interfaz lenguaje textual nivel tarea 119

Figura 66 Lenguaje textual nivel tarea: Visualización 122

Figura 67 Lenguaje textual nivel tarea: programa autómata 1 126

Figura 68 Lenguaje textual nivel tarea: programa autómata 2 126

Figura 69 Lenguaje textual nivel tarea: programa autómata 3 127

Figura 70 Lenguaje textual nivel tarea: programa autómata 4 127

Figura 71 Esquema de trabajo del joystick 129

Figura 72 Interfaz del sistema guiado activo 130

Figura 73 Entorno de programación textual enfocado al robot 132

Figura 74 Estructura de un procedimiento lenguaje textual enfocado al robot 133

Figura 75 Definición de las variables en el lenguaje textual enfocado al robot 134

Figura 76 Variables numéricas lenguaje textual enfocado al robot 136

Figura 77 Constantes numéricas lenguaje textual enfocado al robot 136

Figura 78 Variables booleanas lenguaje textual enfocado al robot 137

Figura 79 Contantes booleanas lenguaje textual enfocado al robot 137

Figura 80 Matrices unidimensionales lenguaje textual enfocado al robot 138

Figura 81 Matrices bidimensionales lenguaje textual enfocado al robot 138

Figura 82 Variables tipo string lenguaje textual enfocado al robot 139

Figura 83 Constantes tipo string lenguaje textual enfocado al robot 139

Figura 84 Variables tipo posición lenguaje textual enfocado al robot 140

Figura 85 Instrucción mover lenguaje textual enfocado al robot 140

Figura 86 Interfaz adicional para el cálculo de los límites del tiempo 141

Figura 87 Instrucción SUCCION para el lenguaje textual enfocado al robot 141

Figura 88 Instrucción SUCCION para el lenguaje textual enfocado al robot 142

Figura 89 Sensores internos para el lenguaje textual enfocado al robot 143

Figura 90 Ciclo if lenguaje textual enfocado al robot 144

Figura 91 Ciclo For lenguaje textual enfocado al robot 145

Figura 92 Ciclo While lenguaje textual enfocado al robot 146

Figura 93 Interfaz general lenguaje de programación nivel tarea 147

Figura 94 Estructura Manual de usuario 152

Figura 95 Capítulo 1 manual robot neumático cartesiano 153

Figura 96 Capítulo 2 manual robot neumático cartesiano 154

Figura 97 Capítulo 3 manual robot neumático cartesiano 155

Figura 98 Capítulo 4 manual robot neumático cartesiano 156

Figura 99 Regulador todo o nada 171

Figura 100 Regulador PID 171

Figura 101 Compresor SULLAIR ES - 6 10H 173

Figura 102 Unidad de mantenimiento 174

Figura 103 Esquema general: tratamiento y generación aire comprimido 174

Figura 104 Sistema de control (Distribución y operación) 175

Figura 105 Regulador de voltaje para el autómata (PS 307) 176

Figura 106 CPU 315F - 2 PN/DP 177

Figura 107 Módulo de 16 entradas y 16 salidas digitales 178

Figura 108 Módulo de regulación PID FM 355C 179

Figura 109 Módulo salidas analógicas SM 332 180

Figura 110 Electroválvula MFH - 3 - 1/4 181

Figura 111 Interior conexiones electroválvula MFH - 3 - 1/4 182

Figura 112 Ubicación de los ejes en el robot neumático cartesiano 183

Figura 113 Estructura de una válvula distribuidora proporcional 184

Figura 114 Cilindro neumático DGPL-25-500-PPV-A-B-KF 186

Figura 115 Cilindro neumático DNCM-32-200-POT1-S20-A 186

Figura 116 Ventosa FESTO 187

Figura 117 Potenciómetro lineal 187

Figura 118 Sensor de proximidad magnético FESTO 188

Figura 119 Sensor de proximidad o presencia 188

Figura 120 Vacuostato 189

Figura 121 Disposición de los módulos en un bastidor 193

LISTA DE TABLAS

pág.

Tabla 1 Lista de variables modo guiado 1 91

Tabla 2 Lista de variables modo guiado 2 92

Tabla 3 Lista de variables modo guiado 3 93

Tabla 4 Lista de variables modo guiado 4 94

Tabla 5 Lista de variables modo guiado 5 95

Tabla 6 Lenguaje textual nivel robot: Ventana de datos 1 116

Tabla 7 Lenguaje textual nivel tarea: ventana de datos 1 124

Tabla 8 Lenguaje textual nivel tarea: ventana de datos 2 125

Tabla 9 Descripción de componentes PLC siemens S7-300 190

LISTA DE ANEXOS

pág.

ANEXO A. DEFINICIÓN DE ROBOT INDUSTRIAL **163**

ANEXO B. LOS LENGUAJES DE PROGRAMACIÓN EN ROBOTS **164**

ANEXO C. ESPECIFICACIONES DE LAS ESTRUCTURAS DE LOS LENGUAJES DE PROGRAMACIÓN **165**

ANEXO D. CLASIFICACIÓN DE LOS LENGUAJES DE PROGRAMACIÓN DE ROBOTS **168**

ANEXO E. SISTEMAS DE CONTROL **170**

ANEXO F. COMPONENTES DEL MANIPULADOR Y ROBOT NEUMÁTICOS CARTESIANOS **172**

ANEXO G. AUTOMATAS PROGRAMABLES: CONCEPTOS BÁSICOS **190**

RESUMEN

TÍTULO: ROBOT NEUMÁTICO CARTESIANO PARA EL APRENDIZAJE DE LOS LENGUAJES DE PROGRAMACIÓN EN ROBÓTICA*

AUTORES: Julio César Galvis Chacón

Daniel Alberto Sierra Gallo **

PALABRAS CLAVES: Robot industrial, lenguajes de programación, válvulas proporcionales distribuidoras, cilindros neumáticos, sistemas de posición neumáticos.

DESCRIPCIÓN:

El objetivo de este proyecto es proveer a la escuela de Ingeniería Mecánica con un medio de bajo costo para el aprendizaje de los lenguajes de programación en sistemas robóticos. Este se consigue al modificar y adicionar los componentes necesarios a un manipulador neumático cartesiano existente en el laboratorio de Automatización Industrial con el fin de llevarlo al nivel de un robot industrial. Dentro de los desarrollos necesarios para cumplir esta meta se encuentran:

Implementación de un modelo predictivo de control para la posición en sistemas neumáticos, basado en la velocidad promedio, recorrido final y el tiempo de arranque obtenidos de la curva desplazamiento - tiempo del elemento deslizante de un cilindro neumático para las diferentes aperturas de los puertos en la válvula proporcional que controla el fluido que llega al actuador; creando una forma válida para el movimiento de un sistema neumático compuesto por válvulas proporcionales como elementos de pre actuación y un conjunto de cilindros neumáticos sin vástago o con vástago pasante como actuadores. Lo anterior, brinda una nueva y más practica alternativa sobre los métodos de control para elementos neumáticos, aplicable a los sistemas fundamentados en el uso de autómatas programables industriales, micro controladores o cualquier otro sistema que tenga la capacidad de ser programado y posea un conjunto de módulos especializados en la lectura de entradas digitales/analógicas, además de la capacidad de modificar salidas digitales/analógicas.

Debido a que la unidad de proceso del robot industrial es un autómata programable, se realizan tres conjuntos interfaz aplicada en un computador – ventana de datos – programación para el autómata, que posibilitan el control del robot industrial por medio de tres sistemas de programación, tales como: guiado activo, textual nivel robot y textual nivel tarea.

---

* Trabajo de grado

** Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingeniería Mecánica. Director: MSc. Ing. Jorge Enrique Meneses Flórez.

ABSTRACT

TITLE: PNEUMATIC CARTESIAN ROBOT FOR LEARNING ABOUT PROGRAMMING LANGUAGES IN ROBOTICS*

AUTHORS: Julio César Galvis Chacón

Daniel Alberto Sierra Gallo **

KEYWORDS: Industrial robot, Programming languages, Proportional directional control valves, Pneumatic cylinders, Pneumatic position systems.

DESCRIPTION:

The objective of this project is to provide the mechanical engineering school with a tool of low-cost for learning about programming languages in robotic systems. This is achieved by modifying and adding the necessary components to a Pneumatic Cartesian Manipulator in the laboratory of Industrial Automation in order to upgrade it reaching an industrial robot level. Within the developments necessary to meet this goal are:

Implementation of a predictive model for the control position in pneumatic systems, based on average speed, final displacement and time to start obtained from the curve displacement - time of the sliding element of a pneumatic cylinder for different openings of the ports in the proportional valve that controls fluid reaching the actuator; creating a valid way for the movement of a pneumatic system composed of proportional valves as elements of pre action and a set of pneumatic rodless cylinders and pneumatic piston rod cylinders as actuators. It offers a new and more practical alternative on the control methods for pneumatic elements, applicably to the systems based on the use of programmable logic controllers, microcontrollers or any other system that has the ability to be programmed and possess a set of modules specialized in the reading of digital/analog inputs, as well as the ability to customize digital/analog outputs.

Due to the fact that the process unit of the industrial robot is a programmable logic controller, are performed three sets interface implemented in a computer - data window - programming for the PLC, which enable the control of the industrial robot by means of three programming systems, such as: active guidance, textual robot level and textual task level.

---

* Degree Work

** Faculty of Physics-Mechanics Engineering. School of Mechanical Engineering. Project director MSc. Ing. Jorge Enrique Meneses Flórez.

INTRODUCCIÓN

En los procesos industriales actuales, el uso de líneas automatizadas de producción se establece como una norma que se deben cumplir para mantener su vigencia. La naturaleza del pensamiento actual en este campo pretende crear sistemas que tengan la capacidad de adaptarse al cambio en las necesidades presentadas en las líneas de producción, como resultado en la búsqueda de elementos bajo este planteamiento surgen una serie de máquinas automáticas denominadas robots industriales.

Desde su estructura y funcionamiento, los robots industriales se encuentran diseñados en pro de adaptarse a las necesidades presentadas por la planta. Una propiedad fundamental en los mismos que demuestra este punto, es la capacidad de ser reprogramados según las tareas dadas por un usuario, el orden como estas acciones son estipuladas por el operador define la eficiencia en los procesos de la línea de manufactura. Por esta razón los profesionales llamados a desempeñarse en esta área deben poseer los conocimientos y las experiencias que garanticen un excelente aprendizaje sobre los sistemas de programación para robots industriales. En búsqueda de lograr este objetivo la escuela de Ingeniería Mecánica de la Universidad Industrial de Santander, a través del Laboratorio de Automatización Industrial ofrece la oportunidad de desarrollar las competencias necesarias para la programación de sistemas robóticos a partir de un banco de prácticas desarrollado por el presente proyecto de grado.

Por lo anterior, el presente proyecto tiene como objetivo la transformación de un manipulador cartesiano (banco de trabajo presente en el laboratorio de Automatización Industrial) al nivel de un sistema robótico (lazos de control e interfaces), de manera tal que se obtenga un medio de bajo costo para el aprendizaje de los lenguajes de programación en robótica; además se pretende aplicar el aprendizaje de los sistemas robóticos a contenidos académicos

propiciando al mejoramiento del perfil del egresado del programa de Ingeniería Mecánica.

Para el logro de dicho objetivo fue necesario restaurar todos los componentes físicos (sensores, preactuadores y controlador) que posibilitan el uso del banco como robot, mejorando algunas condiciones de trabajo en los mismos; además de la creación de interfaces hombre - máquina y lazos de control para el movimiento del sistema propios de un robot industrial, el cual puede ser programado por medio de tres lenguajes diferentes tales como: guiado activo, textual nivel robot y textual nivel tarea. En consecuencia, el elemento generado como resultado de la ejecución de este trabajo de grado es un robot neumático cartesiano, para el cual las instrucciones son dadas por el usuario por medio de tres diferentes sistemas de programación.

Finalmente, el documento está estructurado de la siguiente manera; seguido de la introducción se encuentran los cinco capítulos que comprenden el proyecto, finalizando con las conclusiones, recomendaciones y anexos. El primer capítulo plantea la motivación del presente proyecto, comprendiendo la descripción del problema y objetivos del mismo. En el segundo describe de forma general los elementos desarrollados presentando al lector la información necesaria sobre la ejecución del proyecto. En el tercero se detalla los lazos de control diseñados para el robot neumático cartesiano para la sujeción y transporte de elementos. El cuarto trata sobre el funcionamiento del conjunto interfaz – ventana de datos – programa en el autómata creado para cada lenguaje de programación para sistemas robóticos. El quinto capítulo describe el manual que describe los fundamentos de los lenguajes de programación en robótica seguido de un conjunto de prácticas que ayudan a fortalecer el aprendizaje del estudiante.

1. FORMULACIÓN DEL PROBLEMA

El presente capítulo tiene como meta presentar al lector la justificación que promovió el desarrollo de este proyecto, los objetivos que se plantearon para la ejecución del mismo y el cumplimiento de cada uno de ellos.

1.1 IDENTIFICACIÓN DEL PROBLEMA

Los procesos productivos industriales tienen como base la realización de procedimientos repetitivos, los cuales al ser efectuados por el hombre pueden generarle problemas de estrés, monotonía y agotamiento físico. Además muchas de las tareas a realizar en las plantas industriales requieren de ambientes que son perjudiciales para el capital humano, por esta razón, muchas empresas deciden implementar en los ciclos de producción sistemas automatizados basados en robots diseñados especialmente para este sector económico.

Los sistemas robóticos son el resultado del proceso de una ingeniería concurrente la cual usa los principios de diferentes disciplinas para crear un diseño de alta complejidad. De acuerdo a lo anterior, los diseñadores crean interfaces que permiten la interacción entre el hombre y estos dispositivos, que funcionan como lenguajes textuales, sistemas guiados o sistemas de aprendizaje.

Ahora bien, debido a las tendencias actuales de la implementación de sistemas robóticos en casi todas las líneas de automatización, los profesionales que se desempeñan en esta área deben contar con las competencias que se requieren para el logro de una programación eficiente de sistemas robóticos.

En ese sentido, el laboratorio de Automatización Industrial de la Escuela de Ingeniería Mecánica, en su compromiso de permanecer en constante evolución, es consciente de la importancia de involucrar los lenguajes de programación de

robots como parte de los contenidos dados en la asignatura de autómatas programables. Al interior del laboratorio se cuenta con un manipulador neumático cartesiano, en el cual se realizan prácticas de eventos de control para sistemas discretos.

Por esta razón se pretende llevar este banco al nivel de un sistema robótico; incorporando actuadores, sistemas de control e interfaces necesarias que permitan desarrollar prácticas haciendo uso de lenguajes de programación de robots.

2. JUSTIFICACIÓN

En la actualidad, los procesos presentados por la industria están recurriendo en una forma más frecuente al uso de sistemas automáticos en las líneas de producción. Por esta razón una de las competencias fundamentales que debe adquirir un ingeniero mecánico, para afrontar los retos que implican los actuales sistemas de manufactura, consiste en desarrollar su capacidad para interactuar con los componentes automáticos con el objetivo que le permitan crear ciclos de trabajo más eficientes.

Un claro exponente de este tipo de sistemas automáticos es el robot industrial, el cual es el resultado del esfuerzo conjunto de varias disciplinas de la ingeniería, esto lo hace un sistema nada sencillo de comprender. Con el objetivo que sea operado de forma sencilla por un usuario, los diseñadores desarrollaron una serie de sistemas de programación, que basan sus estructuras acorde al tipo de robot y a las características del entorno de trabajo. Estos lenguajes de programación para robots poseen una serie de funciones e instrucciones que permiten el control de los movimientos y operaciones del autómata.

Para el aprendizaje de los lenguajes de programación en sistemas robóticos, los entes educativos dan preferencia al uso de sistemas computarizados que emulan el dispositivo físico; puesto que los modelos reales tienen costos elevados. Ésta decisión presenta una deficiencia importante, la cual es la de no permitir que se experimente de forma completa el proceso de manipulación de un autómata de este tipo. Es en este punto donde la solución presentada por este proyecto toma su importancia, ya que se desarrolló un robot industrial (parte operativa y sistema de interfaces hombre máquina) a partir de un manipulador cartesiano del laboratorio de Automatización Industrial, lo cual brinda a la Escuela de Ingeniería Mecánica de la Universidad Industrial de Santander una solución de bajo costo y

eficiente para el aprendizaje de los lenguajes de programación en sistemas robóticos.

3. OBJETIVOS

3.1 OBJETIVO GENERAL

Aportar al desarrollo de los contenidos académicos de los estudiantes de la escuela de Ingeniería Mecánica, mediante la transformación de un manipulador neumático cartesiano de tres grados de libertad a un sistema robótico, ampliando la temática ofrecida por el laboratorio de Automatización Industrial, al incluir dentro de las áreas de trabajo, teoría sobre robótica; contribuyendo a fortalecer la misión de la Universidad Industrial de Santander en formar profesionales con alta calidad técnica y científica fomentando la investigación y el desarrollo.

3.2 OBJETIVOS ESPECÍFICOS

Adaptar el sistema físico parte operativa del manipulador cartesiano neumático de eventos discretos (sistema de control, sensores y actuadores) con el fin que permita la ejecución de procesos propios de un robot industrial.

Desarrollar un sistema de control cuasicontinuo con datos muestreados para la posición de los actuadores del robot neumático cartesiano.

Diseñar una interfaz hombre maquina (HMI), en la que se presente de modo interactivo los contenidos necesarios para el aprendizaje de los siguientes métodos de programación para robots: programación guiada directa mediante dispositivo de enseñanza, textual explicita nivel robot y textual nivel tarea.

4. ROBOT NEUMÁTICO CARTESIANO: DESCRIPCIÓN DEL PROYECTO

En los sistemas actuales de producción el uso de líneas automatizadas es casi una regla para el avance y el mejoramiento de los procesos de manufactura, las nuevas metodologías de trabajo buscan sistemas flexibles que se adapten a las diferentes necesidades que se desarrollan durante la ejecución de las tareas dentro de la planta. en pro de satisfacer estos requerimientos surgen una serie de máquinas automáticas diseñadas, con el objetivo principal de configurarse a los cambios que surgen en las labores en el proceso de producción, éstas son denominadas como robots industriales (Ver anexo A) y logran su adaptabilidad ya que pueden ser programados a libertad por un usuario.

La eficiencia de los robots industriales, dentro de una línea de producción, depende en primer lugar de la forma como las instrucciones son dadas por el usuario, puesto que entre más sencillo resulte este proceso para el operador del sistema automático, el tiempo y los errores presentes al realizar dicha programación del equipo disminuyen. Por esta razón, se hace importante que los profesionales llamados a desempeñarse en esta área tengan un conocimiento previo sobre el uso de los lenguajes (Ver anexo B) que se estructuran para programar este tipo de sistemas automáticos.

Uno de los grandes problemas para el aprendizaje de los lenguajes de programación en robótica surge debido a que el método de enseñanza impartido generalmente usa una simulación virtual del modelo físico que compone a un robot industrial; lo cual restringe la posibilidad al aprendiz de experimentar las limitaciones presentes en un sistema real, esto sucede porque los dispositivos robóticos poseen un alto costo económico.

La meta principal del proyecto es la transformación de un manipulador cartesiano (banco de trabajo presente en el laboratorio de Automatización Industrial) al nivel de un sistema robótico (lazos de control e interfaces), permitiendo obtener un medio de bajo costo para el aprendizaje de los lenguajes de programación en robótica.

Sobre el banco de trabajo, presente en el laboratorio, cabe resaltar que es el resultado de dos proyectos de grado anteriores al desarrollado en esta ocasión. En el primero[1] se enfatizó sobre el diseño y construcción del sistema con la posibilidad de ser usado como robot neumático y manipulador cartesiano; por tanto, al momento de realizar el proyecto actual se cuenta con los elementos necesarios desde el punto de vista de sensores, preactuadores, actuadores y controlador para ser usados como parte de la estructura de un sistema robótico. Lo anterior, es un punto a favor aunque el modelo de interfaces y lazos de control establecidos en dicho trabajo no satisfacen las necesidades (Ver anexo C) para la enseñanza de los lenguajes de programación en robots industriales (Figura 1). Sin embargo, debido a la última modificación, realizada en el segundo trabajo de grado[2] (Figura 2), los elementos que constituyen al robot neumático no se encontraban en uso, pues su desarrollo trató al sistema sólo como manipulador cartesiano. Por tanto, para lograr el objetivo del presente proyecto es necesario instaurar de nuevo todos los componentes físicos (sensores, preactuadores y controlador) que posibilitan el uso del banco como robot, mejorando algunas condiciones de trabajo en los mismos, además de la creación de interfaces hombre - máquina y lazos de control para el movimiento del sistema propios de un robot industrial, el cual puede ser programado por medio de tres lenguajes

[1] BUITRAGO, Jhon Edisson y CEDIEL Nelson Gustavo. Robot Cartesiano Neumático para el Laboratorio de Sistemas Mecatrónicos de la Escuela de Ingeniería Mecánica. Diseño y Construcción. Trabajo de grado Ingeniero Mecánico. Bucaramanga. Universidad Industrial De Santander. 2010. Pág. 68.

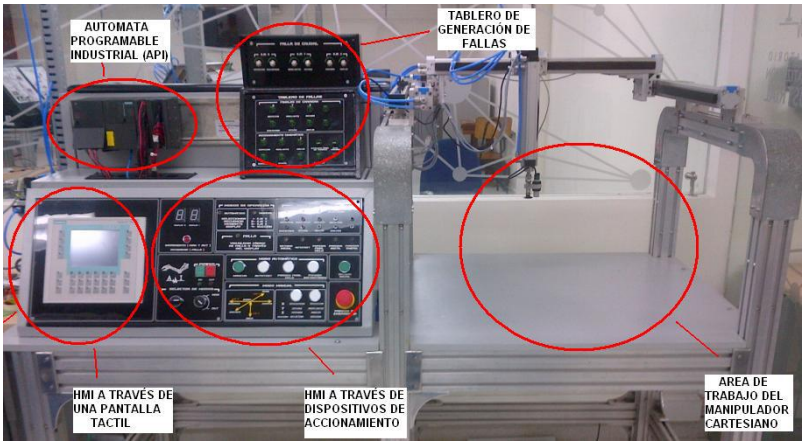
[2] DAVID, Roberto Fabio y LIZARAZO Rubén Darío. Guía de estudio de los modos de marchas y paradas (Gemma) para el Manipulador Cartesiano del Laboratorio de Automatización Industrial: Diseño y Construcción. Trabajo de grado Ingeniero Mecánico. Bucaramanga. Universidad Industrial De Santander. 2012. Pág. 34.

diferentes tales como: guiado activo, textual nivel robot y textual nivel tarea(Ver anexo C).

Figura 1. Robot/Manipulador cartesiano primer proyecto de grado

Fuente: Robot cartesiano neumático para el laboratorio de sistemas mecatrónicos de la Escuela de Ingeniería Mecánica, diseño y construcción[3].

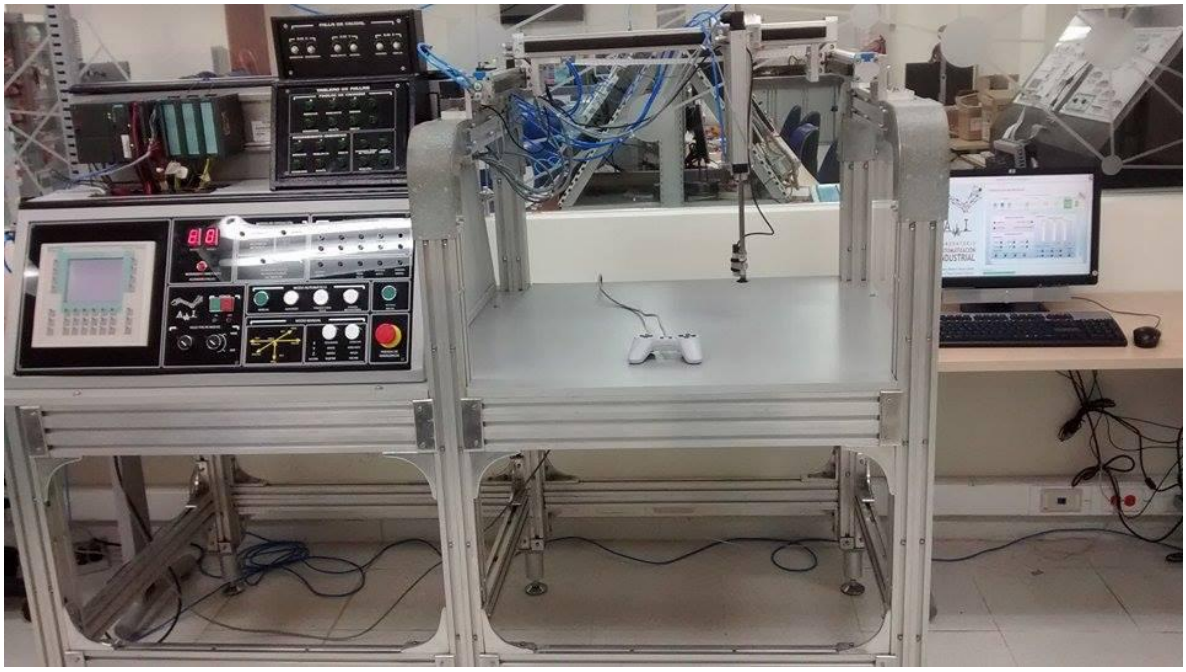
Figura 2. Manipulador Cartesiano segundo proyecto



Fuente: Guía de estudio de los modos de marchas y paradas (GEMMA) para el manipulador cartesiano del laboratorio de Automatización Industrial: diseño y construcción[4].

[3] BUITRAGO, Jhon Edisson y CEDIEL, Nelson Gustavo. Óp. cit. Pág. 68.
[4] DAVID, Roberto Fabio y LIZARAZO Rubén Darío. Óp. cit. Pág. 34.

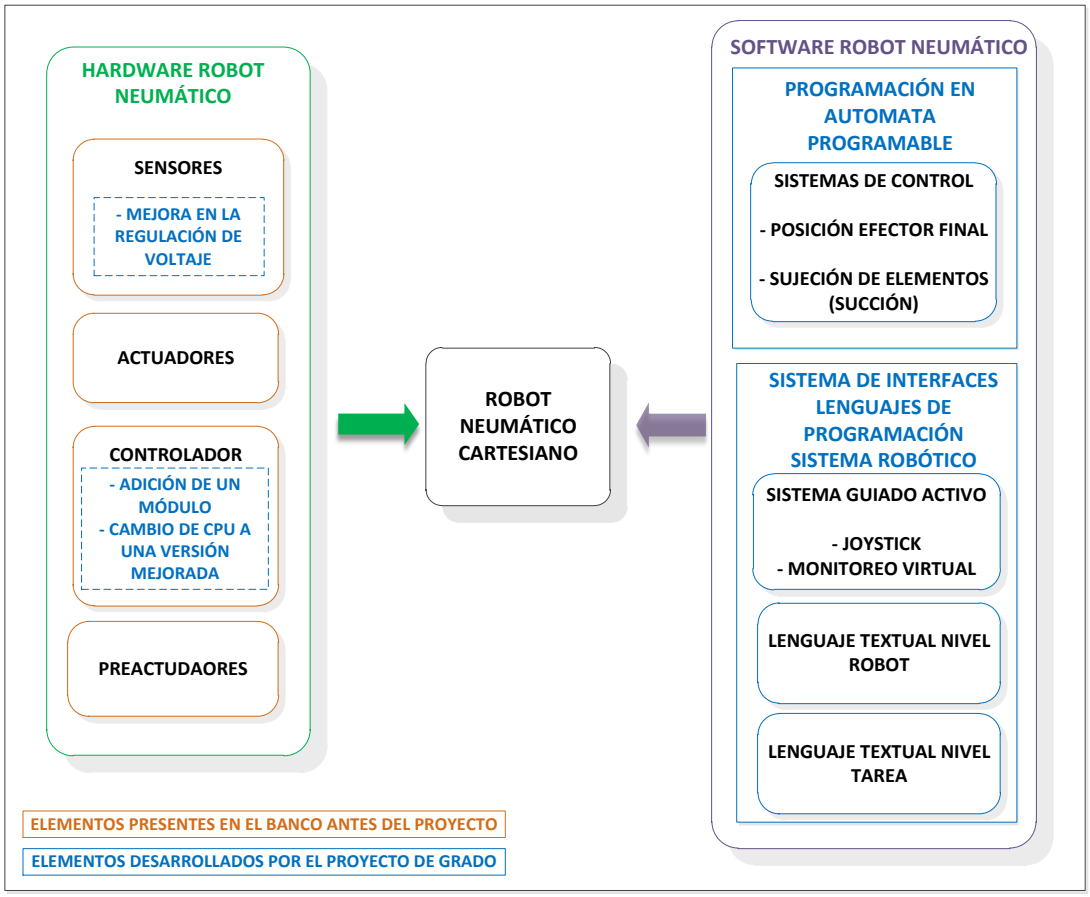
Figura 3.Robot neumático cartesiano proyecto actual



Fuente: Banco de trabajo, Laboratorio de Automatización Industrial

La diferencia entre un manipulador cartesiano y un sistema robótico radica en los lazos de control para el movimiento del efector final, además de las interfaces hombre – máquina. Pues para un robot éstas características pertenecen a un nivel mucho más complejo que las dadas para el manipulador. Una descripción del sistema robótico y su desarrollo se presentan en la Figura 4.

Figura 4. Composición robot neumático



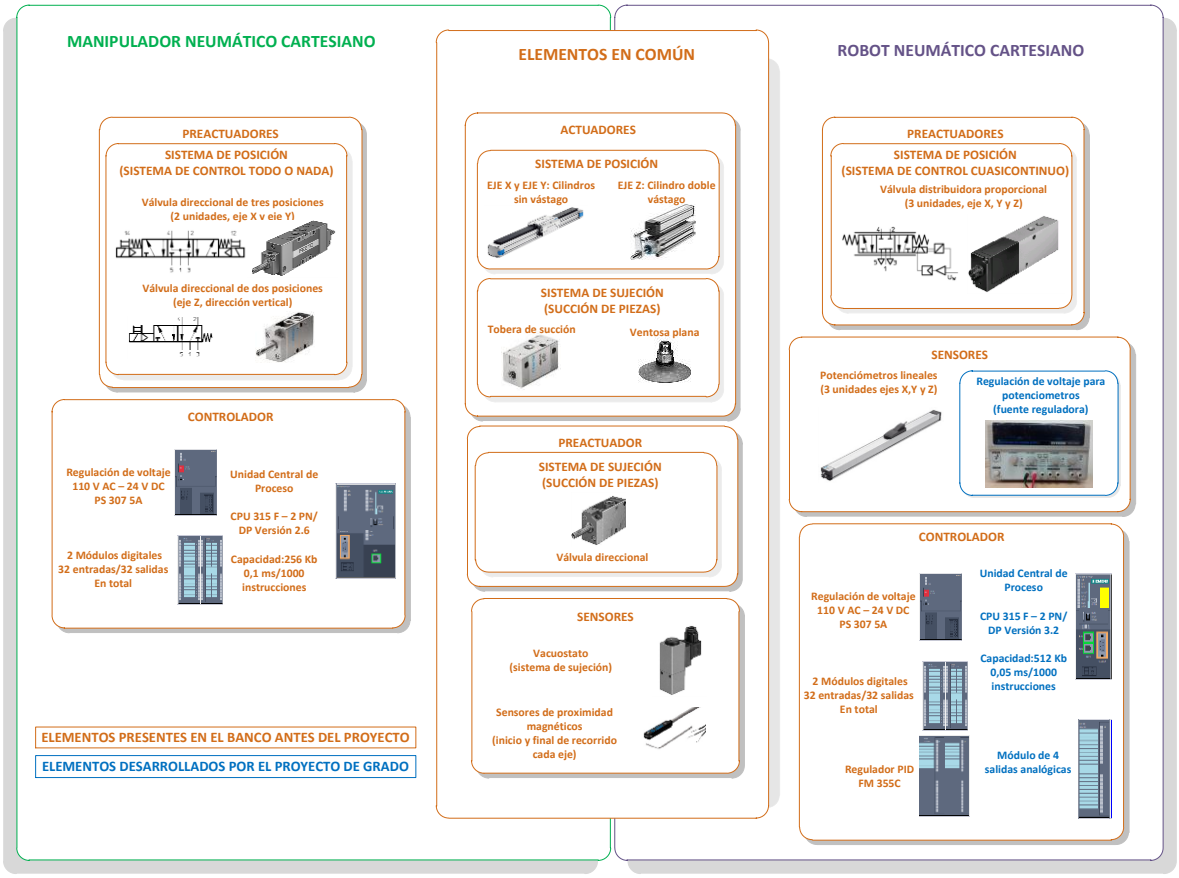
Para ilustrar al lector sobre la base de construcción y el resultado final obtenido por el proyecto se presenta la siguiente descripción.

4.1 COMPARACIÓN DESDE EL HARDWARE:

Las actividades que realizan tanto el manipulador como el robot neumático pueden considerarse similares, ya que el objetivo principal de ambos sistemas es la sujeción y transporte de elementos de un lugar a otro, donde el espacio de trabajo para ambos se establece en un sistema de coordenadas cartesianas de tres dimensiones, en el cual, el desplazamiento en cada eje se da de forma independiente por cada actuador.

Por esta razón se hace posible que elementos tales como los actuadores, un preactuador (usado en el sistema para la succión) y algunos sensores digitales se presenten en ambos sistemas. La Figura 5 representa de forma específica los elementos comunes y en los que difieren ambos sistemas con el fin de establecer los cambios realizados para el proyecto de grado.

Figura 5.Comparación punto de vista HARDWARE



Fuente: FESTO®, SIEMENS®.

Las diferencias observadas en la Figura 5, demuestran la complejidad en la estructura física de un robot sobre la presentada en un manipulador, en especial en el controlador y la parte sensorial puesto que si se observa de forma detallada, para los lazos de control de posición, el sistema robótico requiere del uso de potenciómetros lineales (Ver anexo F), los cuales requieren una regulación de voltaje especializada. Otra diferencia importante viene dada desde el punto de

vista de los preactudores, pues en el manipulador cartesiano se usan válvulas distribuidoras *todo o nada*, mientras que para el robot se presentan válvulas distribuidoras *proporcionales*; por último cabe resaltar los cambios presentados en el controlador, siendo de mayor potencia en el robot puesto que aquí se usa una CPU modelo 315F – 2 PN/DP versión 3.2 con capacidad de almacenamiento de 512 Kb y de procesamiento de 0,05 ms/1000 instrucciones. En tanto que para el manipulador se usa el mismo modelo de CPU pero versión 2.6 cuya capacidad de almacenamiento es de 256 Kb y con un procesamiento de 0,1 ms/1000 instrucciones. Aunque en el proyecto de grado,[5] que construyó el sistema con miras a un robot cartesiano, usaba la versión 2.6; para el proyecto actual se requirió la presencia de la versión 3.2, debido al tamaño de los nuevos bloques de programación creados para los lazos de control en el movimiento del efector final y las distintas funciones ofrecidas por los lenguajes de programación para robots. Por último cabe resaltar la presencia de dos módulos adicionales en el robot neumático los cuales son: el sistema de regulación PID (Ver anexo E) FM 355C (Ver anexo F), que en el proyecto inicial[1] era el único encargado del control de la posición de los actuadores. Para el presente trabajo de grado, dicho sistema de posicionamiento no es suficiente, lo cual necesita que el uso del mismo sufra muchas modificaciones desde el punto de vista de su programación y forma de trabajo, ya que para este proyecto sólo se requieren los canales para la lectura de entradas analógicas del mismo. Lo anterior, puesto que sus salidas analógicas son sustituidas por un módulo externo (dedicado solo a salidas analógicas), esto se debe a que si se usa tanto el canal de entrada como el de salida del módulo FM, el procesamiento (este es realizado en el módulo y no en la unidad central de proceso “CPU”) tomaría demasiado tiempo para las necesidades presentadas por los lazos de control establecidos en el sistema actual de trabajo.

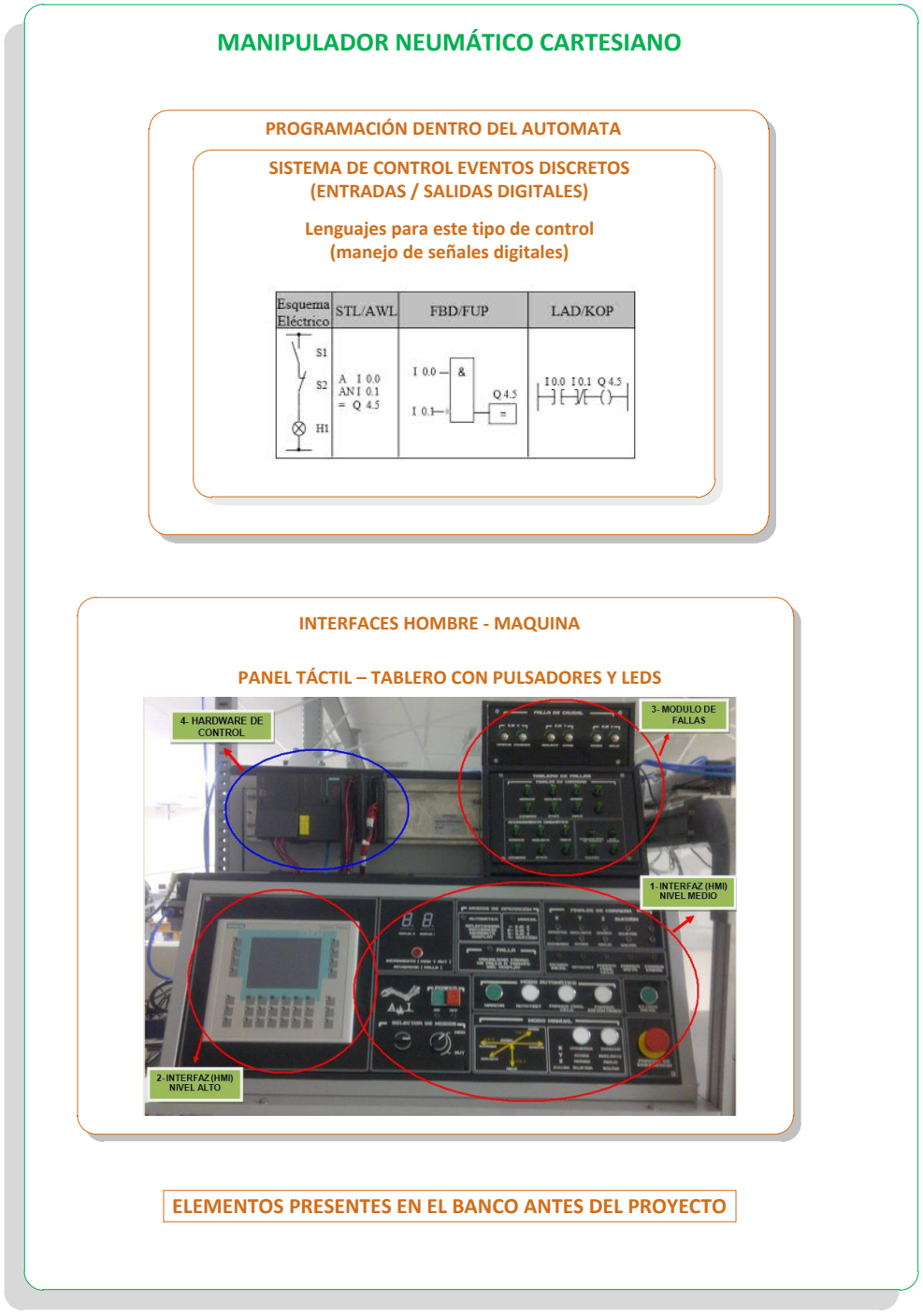
[5] BUITRAGO, Jhon Edisson y CEDIEL Nelson Gustavo. Óp. cit. Pág. 24.

4.2 COMPARACIÓN DESDE EL SOFTWARE

Este punto de vista comprende desde los sistemas de control para las tareas de sujetar y transportar elementos, los cuales son programados dentro del autómata (Ver anexo G), hasta las interfaces hombre - máquina presentes en el computador.

Las diferencias en el sistema de software y las interfaces hombre - máquina presentes en el manipulador cartesiano y el robot pueden verse en las Figuras 6, 7 y 8.

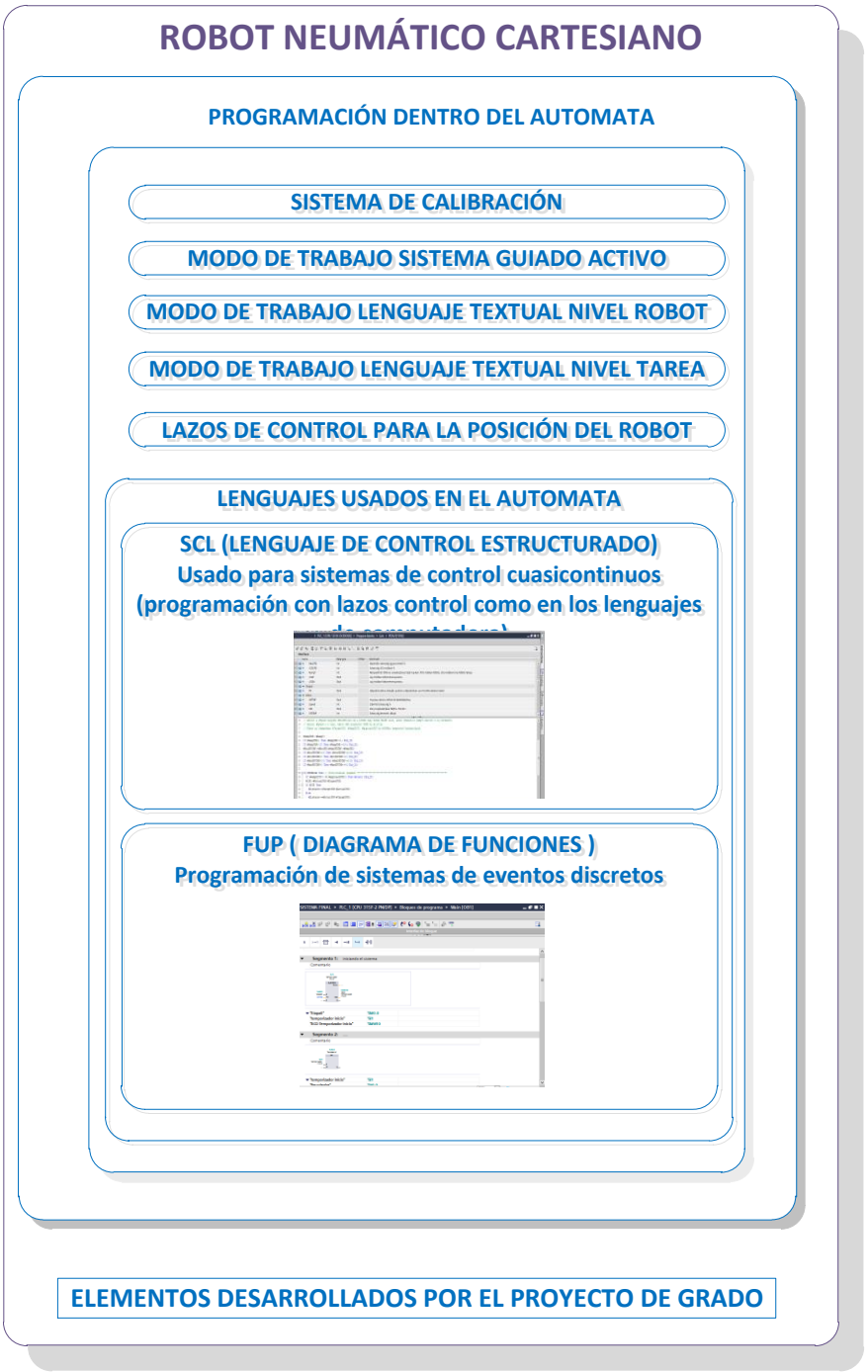
Figura 6. Software interfaces Manipulador



Fuente: Guía de estudio de los modos de marchas y paradas (GEMMA) para el manipulador cartesiano del laboratorio de Automatización Industrial: diseño y construcción[6].

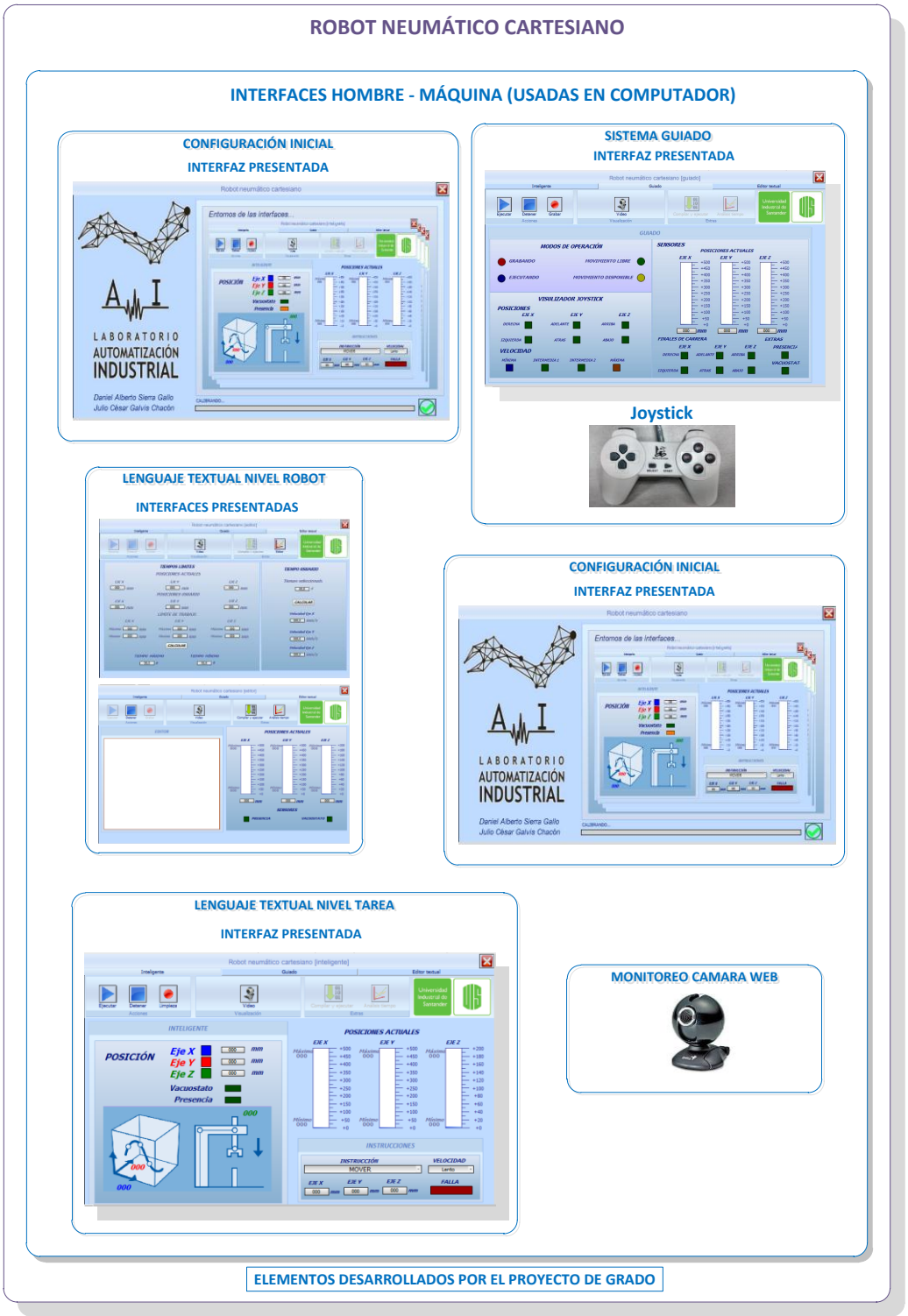
[6] DAVID, Roberto Fabio y LIZARAZO Rubén Darío. Óp. cit. Pág. 34.

Figura 7. Software Robot en el autómata



Fuente: TIA PORTAL V11, SIEMENS®

Figura 8. Interfaces Robot para el computador



Las diferencias entre la programación presentada en el manipulador y el robot cartesiano para el autómata son muy grandes ya que para el primero sus lazos de control corresponden a los sistemas de eventos discretos aquellos cuyas señales se presentan en dos estados disponibles (digitales) mientras que para el segundo sus sistemas requieren la inclusión de sistemas de control cuasicontinuos con datos muestreados y también de eventos discretos.

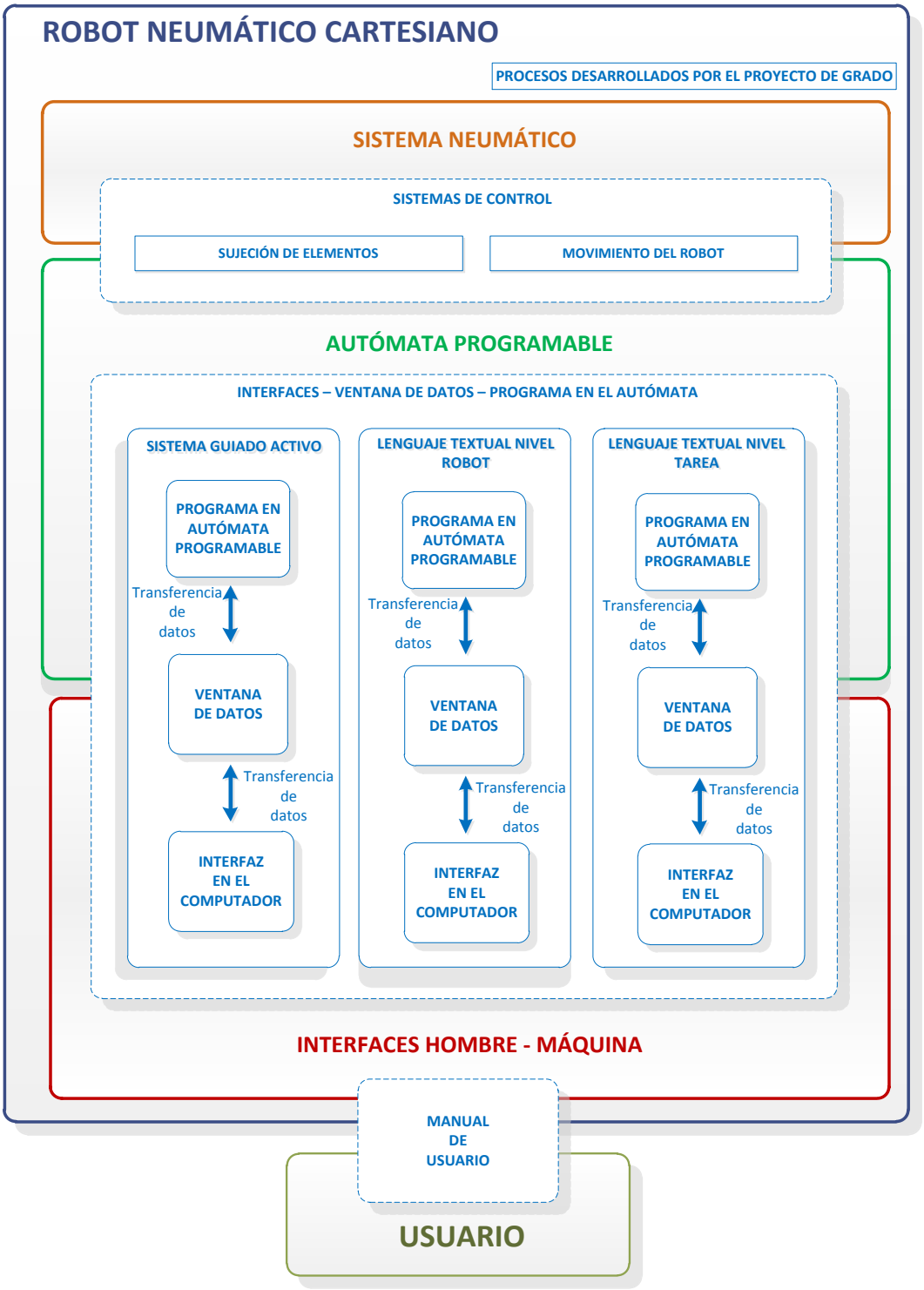
Respecto a las interfaces presentadas para cada sistema se observa fácilmente que se tornan más complejas para el robot puesto que requieren la creación de varias ventanas diseñadas especialmente para el trabajo de los diferentes lenguajes, además de la adecuación de un mando (joystick) para el robot neumático más ameno al trabajo del usuario y la creación de un sistema de monitoreo remoto del banco de trabajo.

4.3 DESARROLLO DEL PROYECTO DE GRADO

Como lo muestra la figura 4, el presente proyecto centró casi todo su desarrollo en el sistema software usado dentro del robot neumático.

Una descripción general de las actividades realizadas por el proyecto en función de los elementos que componen al robot neumático cartesiano se presenta en la Figura 9.

Figura 9. Descripción general del proyecto



A partir de la Figura 9, se presenta claramente los procesos que desarrolla el proyecto de grado como función de las tareas dadas en objetivos específicos con el fin de lograr el objetivo principal, es decir, crear un robot neumático cartesiano que permita al estudiante la interacción con los lenguajes de programación en sistemas robóticos.

Los procesos especificados son:

- Sistemas de control:

Se establecieron entre el sistema neumático y el autómata programable, fue necesario desarrollar los lazos de control encargados del movimiento del robot y el agarre de los objetos de trabajo, ingresarlos dentro de la programación del autómata, ya que éste actúa sobre los elementos neumáticos por medio de los sensores y preactuadores acoplados a sus módulos.

- Interfaces – Venta de datos – Programa en el autómata

Para cada lenguaje de programación desarrollado como forma de controlar el robot fue necesaria la creación de un conjunto completo de trabajo, que consiste en una interfaz desarrollada según los requerimientos del sistema de programación de robots presentada al usuario por medio de un computador. Dicha interfaz, a partir de una serie de variables compartidas con el autómata programable (Ventana de datos), puede modificar su estado o el estado del sistema robótico según lo programado dentro del autómata.

- Manual del usuario

En cumplimiento a lo planteado en los objetivos específicos se realiza un manual de usuario, en el cual se estipulan los conocimientos teóricos necesarios para que el estudiante o el operador aprendan de forma detallada los lenguajes de

programación en sistemas robóticos complementándolos con una serie de ejercicios prácticos que le permiten experimentar de manera completa las limitaciones de un modelo real.

5. ROBOT NEUMÁTICO: SISTEMAS DE CONTROL

La función principal del robot neumático cartesiano es el transporte de elementos de un lugar a otro, siendo este el resultado de la composición de dos tareas, la primera de ellas trata sobre la forma como los objetos son enganchados por el efector final del sistema, la segunda se enfoca al movimiento del elemento final de actuación en el robot.

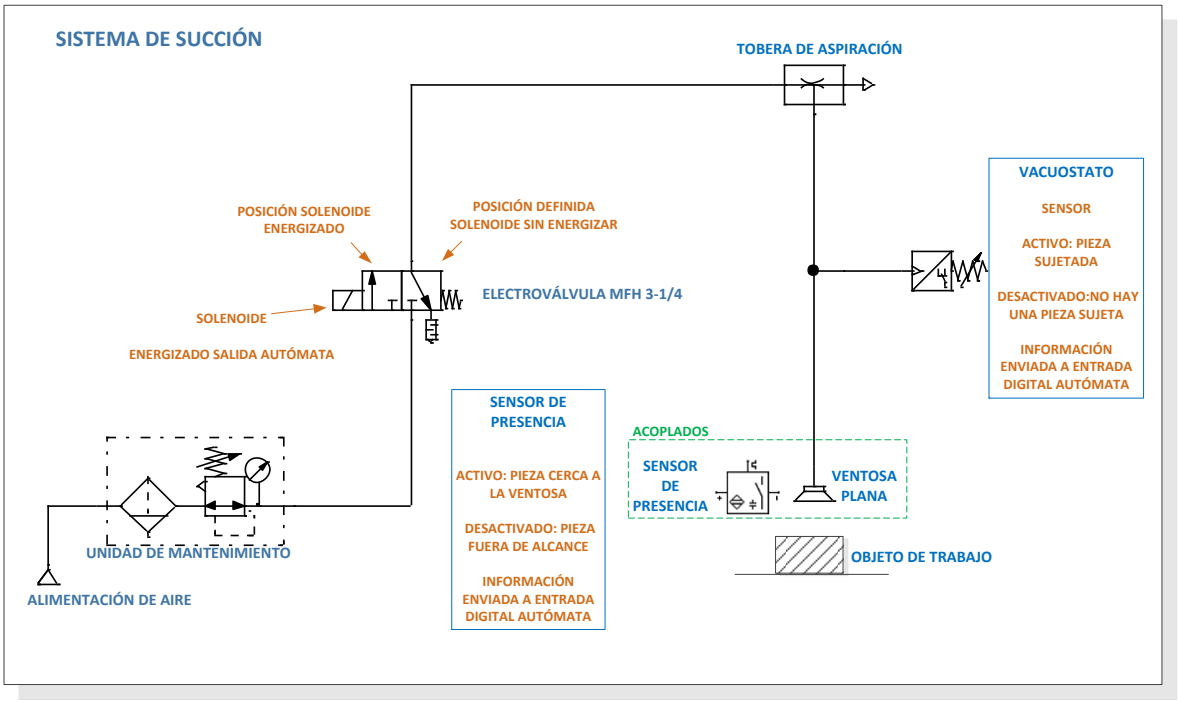
Este capítulo se destina a explicar el desarrollo y aplicación de los sistemas de control establecidos para las tareas de succión de elementos y movimiento del efector final del robot.

5.1 LAZO DE CONTROL: SUCCIÓN DE ELEMENTOS

La forma como el sistema robótico sujeta a los objetos de trabajo toma como actuador principal a una ventosa plana (Ver anexo F), la cual por medio de un conjunto de generación de presión de vacío crea una manera para mantener los elementos unidos al actuador a través de la succión de los mismos.

El sistema físico presente en el banco para cumplir con la tarea de sujetar los elementos, se ilustra en la Figura 10.

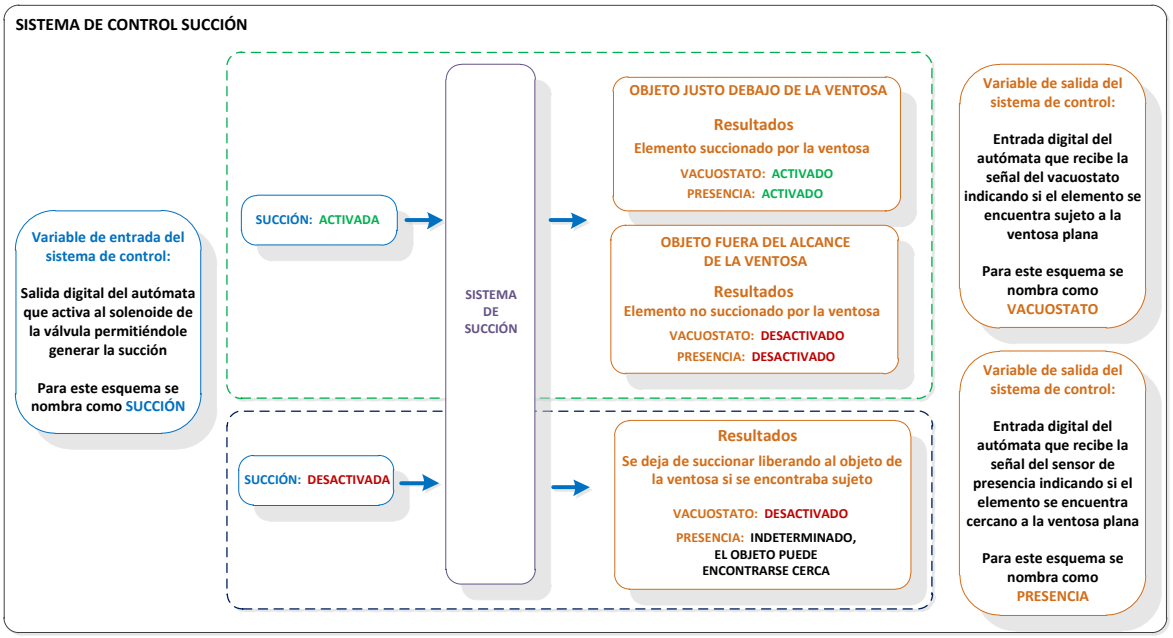
Figura 10. Sistema de succión



El sistema de succión está compuesto por una unidad de mantenimiento, una válvula direccional o distribuidora cuyo cambio de posición de trabajo se da por medio de la activación de un solenoide, una tobera de aspiración por vació la cual en conjunto con la ventosa plana generan la succión de los objetos y por último como sensores se presentan: un vacuostato, que permite retroalimentar al sistema para conocer que el elemento se encuentra sujeto por el actuador y el sensor de presencia que permite conocer si una pieza de trabajo se encuentra cerca a la ventosa.

El sistema de control se puede resumir como se muestra en la Figura 11.

Figura 11. Control sistema de succión



Como se observa en la Figura 11, el sistema de control se basa en dos parámetros importantes; el primero de ellos es el estado del solenoide la válvula direccional ilustrada en la Figura 10, el cual se encuentra acoplado a una de las salidas digitales del autómata, por tanto el usuario puede programar las condiciones que lo activan o lo desactivan. No obstante, para lograr la succión de un elemento es necesaria una segunda condición, que el elemento se encuentre justo debajo de la ventosa, porque en caso contrario será imposible esta tarea.

Para conocer si una pieza de trabajo se encuentra cerca al campo de actuación de la ventosa se usa un sensor de presencia, el cual se activa cuando este evento sucede, en caso contrario permanece desactivado, la señal dada por él es enviada a la entrada digital del autómata con el fin que pueda ser usada en la programación del usuario.

El elemento que permite conocer si un objeto se encuentra sujeto por la ventosa, es el vacuostato mostrado en la Figura 10, su operación se ve reflejada en una entrada digital del autómata al cual está conectado, de esta manera el robot puede

conocer el estado del lazo de succión y el usuario usarlo como parte de su programación.

5.2 LAZOS DE CONTROL: MOVIMIENTO DEL ROBOT

El sistema que compone el lazo de control para el movimiento del robot se muestra en la Figura 12.

Figura 12. Sistema de control movimiento

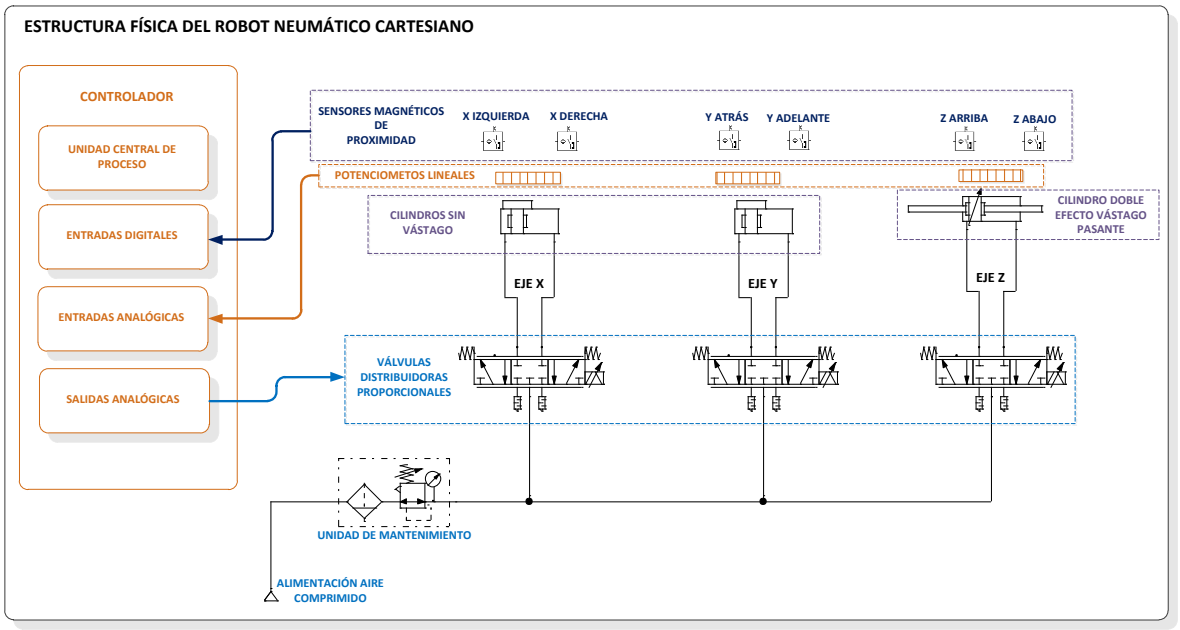
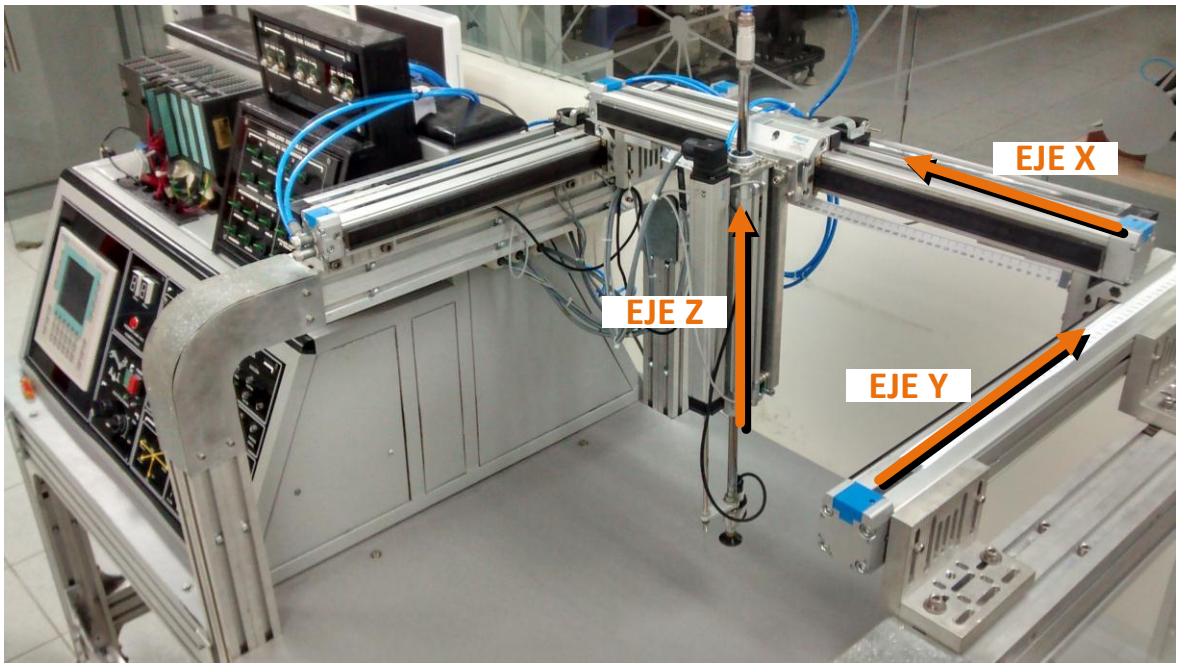


Figura 13. Ejes en el robot cartesiano



Fuente: Banco de trabajo, Laboratorio de Automatización Industrial

Antes de presentar los lazos de control para el movimiento desarrollados en el proyecto actual, es importante ilustrar al lector sobre la realidad física de los sistemas neumáticos, ya que ésta posee un alto nivel de complejidad; pues al establecer el aire comprimido como fluido de trabajo, siendo un material cuyas propiedades pueden verse afectadas por muchos factores tales como la temperatura, las diferencias de presión entre aberturas y las fugas hacia el exterior; las ecuaciones que describen el modelo dinámico de un sistema de este tipo son del tipo no lineales, basadas en una gran cantidad de constantes, éstas últimas deben ser obtenidas de forma experimental y son relativas a cada conjunto. En la presente sección se muestra de forma general un estudio[7] del

[7] NOURI, Bashir M, et al. Modelling a pneumatic servo positioning system with friction. En Proceedings American Control Conference, Junio de 2000. Vol. 2.

departamento de Ingeniería Mecánica de la Universidad Católica de Lovaina, Bélgica, en el que se detalla la naturaleza de este tipo de sistemas para la configuración: Válvula proporcional – Cilindro sin vástago. De otra parte, debido a la complejidad en el modelo, se puede pensar en obtener el controlador por medio de los métodos experimentales de sintonía pero debido a lo presentado en el capítulo 6 del trabajo "*Montaje, programación y puesta en marcha de un robot neumático de estructura paralela*"[8], se demuestra que para sistemas neumáticos MIMO (sistemas con varias entradas y varias salidas, para este caso dos lazos de control relacionados por el caudal común de alimentación) abordar esta tarea, desde este punto de vista, hace que la obtención de las características para el controlador sea un proceso demasiado complicado y los resultados obtenidos no son los más favorables. Por tal motivo, para el proyecto se realiza un control ajustado a las condiciones del modelo especifico de la planta, debido al enfoque tomado en este puede considerarse con una naturaleza predictiva pues las acciones de control se dan como resultado de un procedimiento de calibración previo al funcionamiento del proceso.

- **Modelo dinámico de una configuración Válvula proporcional – Cilindro sin vástago**

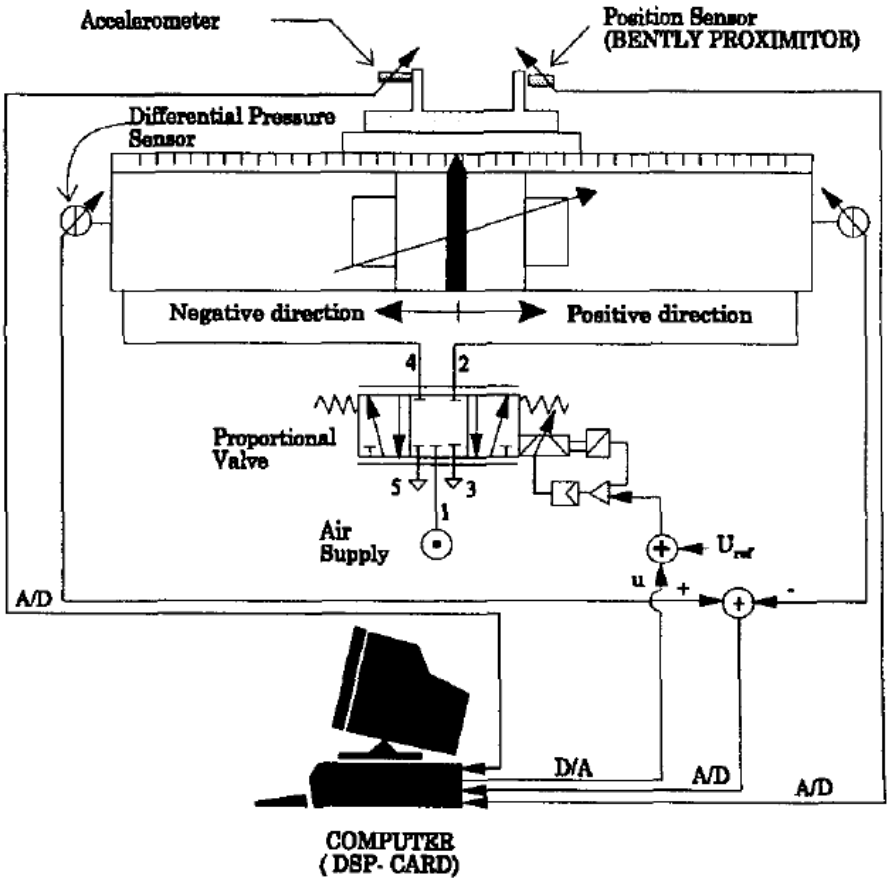
En un informe presentado por el departamento de Ingeniería Mecánica de la Universidad Católica de Lovaina, Bélgica, se establece el modelo de ecuaciones que describen a un sistema servocontrolado el cual consiste en el uso de una válvula proporcional, ésta a partir de la señal de control que recibe modifica el caudal que llega a las cámaras de un cilindro sin vástago. Los modelos usados para el estudio corresponden a una válvula proporcional distribuidora MPYE 5-1/8 y un cilindro DGPIL-25-1250-GK-KF-AH ambos ofrecidos por la empresa FESTO®

[8] MESEGUER, Alejandro R. Sintonía experimental de controladores. En: Montaje, programación y puesta en marcha de un robot neumático de escritura paralela. Trabajo de grado Ingeniero de Telecomunicaciones. Cartagena. Universidad Politécnica de Cartagena. 2008. Pág. 105 - 134.

los cuales pueden tomarse como válidos para la comparación pues el sistema de movimiento del presente proyecto usa válvulas proporcionales cuyas referencias son MPYE 5-1/4 y sus cilindros corresponden a la línea DGPL-25-500-PPV-A-B-KF ofrecidas por FESTO®.

El sistema de experimentación presentado en el artículo se muestra la Figura 14.

Figura 14. Modelo dinámico



Fuente: "Modelling a pneumatic servo positiong system with friction".

El modelo descrito obedece a las siguientes ecuaciones:

El flujo másico de aire comprimido que va desde los puertos de la válvula hacia los del actuador, obedece la siguiente relación:

$$\dot{m} = (\pm)\, A_{eff}\, P_1 \sqrt{\frac{2\,k}{R\,T\,(k+1)}} \left[\frac{2}{k+1}\right]^{1/(k+1)} \Psi \left(P_2/P_1\right)$$

Para la cual: $(\pm)$ El signo depende del sentido del flujo másico, P_1 indica la presión antes de pasar por la válvula, P_2 indica la presión después de pasar por la válvula, k corresponde a la constante isentrópica, R es la constante de los gases, T indica la temperatura del aire desde la fuente.

El área efectiva es igual a:

$$A_{eff} = A_m \; \frac{\left(1 - \left(\frac{u - U_s}{U_O - U_s}\right)^c\right)^D}{a_2 {P_2}^2 + a_1 P_2 + a_0}$$

Para la cual: A_m Máxima área alcanzable, en el documento se experimenta con una válvula de tensión, u potencial enviado al solenoide de la válvula, U_s potencial de saturación, U_0 potencial de banda muerta, C, d, a_0, a_1, a_2 son constantes que deben obtenerse por medio de la experimentación del sistema, P_s presión dada por la fuente.

El coeficiente Ψ depende del sentido de trabajo:

Si se está cargando el cilindro el valor usado es el siguiente:

$$\Psi_C = \left(1 - \left[\frac{\frac{P_2}{P_s} - b}{1 - b}\right]^{\beta}\right)^{\gamma}$$

Donde b es la relación de presión critica, β, γ son constantes.

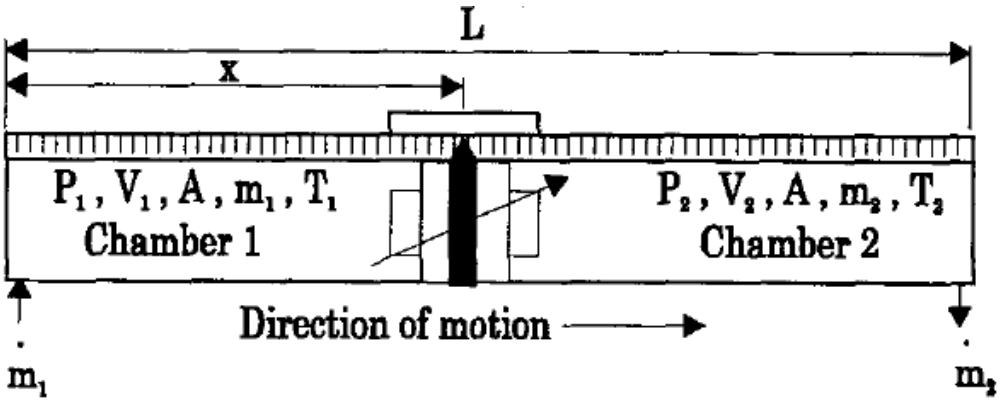
Si se está descargando el cilindro el valor usado es el siguiente:

$$\Psi_D = \sum A_n \left(\frac{P_{atm}}{P_2}\right)^n$$

Donde A_n es una constante que representa el área, n es el número de puertos.

Al realizar un análisis en el actuador lineal, se obtiene:

Figura 15. Actuador lineal artículo



Fuente: "Modelling a pneumatic servo positiong system with friction".

Se presentan las siguientes relaciones:

$$\dot{P}_1 = \frac{R\,T}{A\,x\ + V_{1D}}\,\dot{m}_1 - \frac{A\ P_1}{A\,x\ + V_{1D}}\dot{x} \qquad \dot{P}_2 = \frac{R\,T}{A\,(L-x)\ + V_{2D}}\,\dot{m}_2 - \frac{A\ P_2}{A\,x\ + V_{2D}}\,\dot{x}$$

Para la cual: A el área de la sección transversal, P_1 la presión en una de las cámaras, P_2 la presión en la cámara opuesta, T temperatura del aire se toma igual para ambas cámaras del cilindro, x posición relativa del vástago, L carrera total del actuador lineal, V_{1D} , V_{2D} Volumen de las cámaras del actuador lineal.

Al realizar un análisis de la aceleración del actuador se obtiene la siguiente ecuación:

$$\ddot{x} = \frac{A}{M}(P_1 - P_2) - \frac{1}{M}\,F_R$$

Donde M es la masa del elemento deslizante del actuador, A es el área del pistón sobre el cual se aplica la presión, F_R es la fuerza de rozamiento a la cual es sometido el elemento deslizante.

Como se puede observar, las ecuaciones que relacionan la primera y segunda derivada de la posición de las cuales se podría resolver la naturaleza matemática que define de forma explícita la ubicación del elemento deslizante dependen de muchos factores tales como la fricción del sistema, además de la temperatura del fluido de trabajo y muchas otras características que obedecen a las condiciones en las que se encuentra el sistema al momento de moverse. Lo anterior, permite ilustrar al lector sobre la complejidad de un sistema neumático y como una pequeña variación en las condiciones de trabajo ya sea en el exterior del conjunto o dentro del mismo causan grandes modificaciones en el funcionamiento del proceso de movimiento.

Para el proyecto actual el nivel de complejidad aumenta ya que el modelo estudiado en el artículo "*Modelling a pneumatic servo positiong system with friction*" solo corresponde al lazo dado por una sola válvula y cilindro, en el sistema presentado en la Figura 12. Se cuenta con tres conjuntos de este tipo.

Es importante resaltar que las propiedades del sistema pueden ser medidas a partir del uso de sensores tales como caudalímetros, sensores para la presión, que deben ser ubicados en los puertos tanto de la válvula como del cilindro. Por tal motivo, pretender modelar el sistema de forma matemática no es una opción válida ya que requería la adición de dichos sensores en todo el sistema, esto se desvía del objetivo principal del trabajo de grado, siendo el de realizar la transformación de un manipulador a un robot usando los elementos físicos ya presentes como resultado de proyectos anteriores con el fin de crear un sistema que no necesitase una mayor inversión económica en su desarrollo.

- **Sintonía del controlador de forma experimental**

Una segunda forma de generar el análisis para obtener el sistema de control se basa en determinar los parámetros del controlador por medio de los métodos de experimentación dados en la teoría de control. Desde este punto de vista, el proyecto se basa en los resultados plasmados en el capítulo 6 del trabajo *"Montaje, programación y puesta en marcha de un robot neumático de estructura paralela"*[9], el cual analiza la sintonía de controladores para un robot neumático cuyo sistema de movimiento se da por la composición de dos conjuntos válvula proporcional – cilindro neumático.

En el proyecto "*Montaje, programación y puesta en marcha de un robot neumático de estructura paralela*"' el autor plasma los resultados obtenidos para la sincronización de los parámetros del controlador de un sistema de movimiento basado en una válvula neumática proporcional y un cilindro neumático. Este autor, los analiza por medio de los métodos de prueba y error, Ziegler-Nichols y Harriot, obteniendo resultados favorables para el sistema evaluado de forma individual, pero al acoplar los dos conjuntos válvula – cilindro, los efectos presentados no son los esperados; demostrando así que el único método posible para la sintonía del controlador es el uso de la prueba y el error, lo cual es un proceso tedioso y puede causar daños al robot ya que como resultado de algunas configuraciones se puede forzar a realizar movimientos que afectan la integridad de la estructura.

Para profundizar sobre el artículo "*Montaje, programación y puesta en marcha de un robot neumático de estructura paralela*", el lector puede descargarlo desde el link http://hdl.handle.net/10317/174, el cual conduce al repositorio digital de la Universidad Politécnica de Cartagena, España.

[9] MESEGUER, Alejandro R. Óp. cit. Pág. 105 – 134.

- **Sistemas de control desarrollados para el robot neumático**

Como se observa anteriormente, tanto el modelo dinámico como la sintonía de controladores de forma experimental no son maneras viables para crear un sistema de control en este proyecto. Por esta razón y a través de la experimentación detallada de la estructura física, se logra determinar una forma práctica y efectiva para controlar la posición del sistema neumático del robot cartesiano, aplicable a los sistemas basados en el control por medio de autómatas programables, micro controladores o cualquier otro sistema que tenga la capacidad de ser programado y posea un conjunto de módulos especializados en la lectura de entradas digitales/analógicas además de la capacidad de modificar salidas digitales/analógicas.

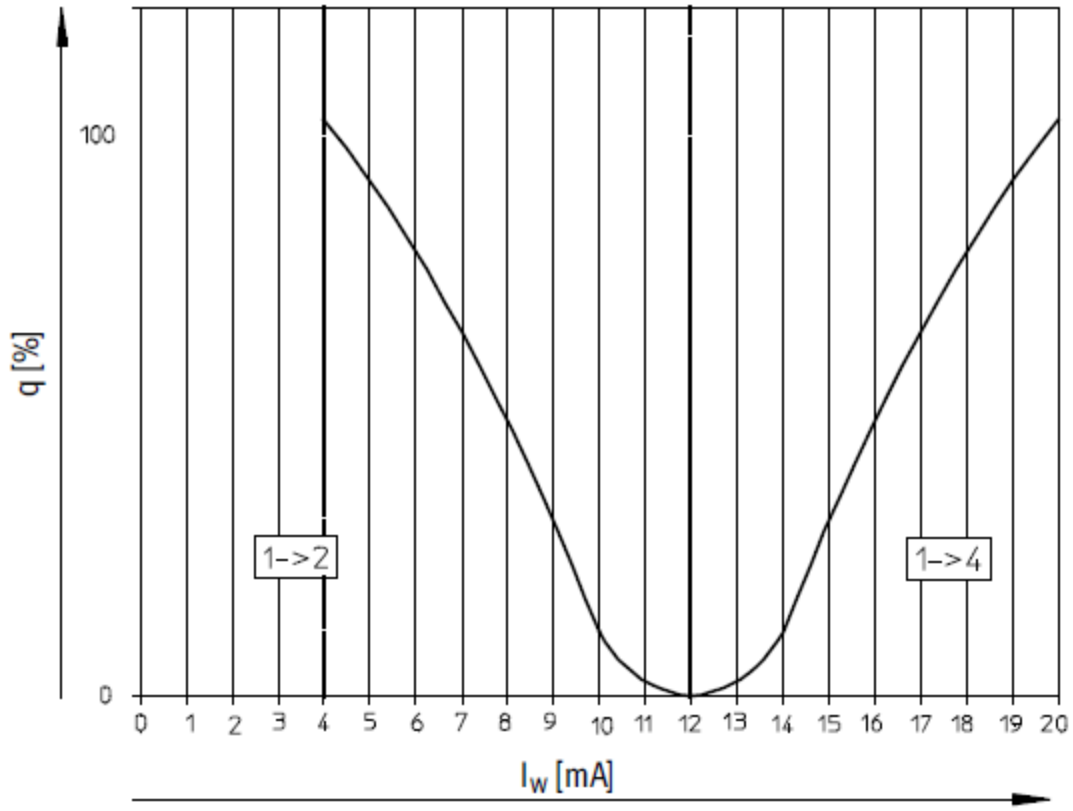
Existen dos tipos de sistemas, el primero de ellos consiste en generar un lazo de control de posición en el que el usuario puede determinar el tiempo que toma el proceso en realizarse, esto implica que se ejecuta a diferentes velocidades. Mientras que el segundo basa su algoritmo de control de posición al movimiento en una velocidad fija, la cual es la mínima presentada en cada eje para el robot neumático.

Antes de poder presentar al lector el funcionamiento de ambos sistemas es importante dar a conocer los principios en los cuales se fundamenta el control predictivo del robot neumático cartesiano, además de mostrar las conclusiones que los validan para poder ser usados.

En el sistema neumático presentado en la Figura 12, el movimiento del elemento dentro de los cilindros se da como una función del caudal que llega a las cámaras de estos actuadores.

La válvula proporcional distribuidora se encarga de modificar el caudal que llega a los actuadores y de esta forma modifica la posición a la que se encuentra el elemento deslizante del cilindro. Esto lo logra al modificar el área de apertura para el paso del fluido por sus puertos, la forma como se puede cambiar dicha propiedad es a través de la señal de control que recibe el solenoide de la válvula, para los modelos presentes en el robot neumático la relación entre la apertura y la señal de control se muestra en la Figura 16.

Figura 16. Apertura válvula como función de la señal

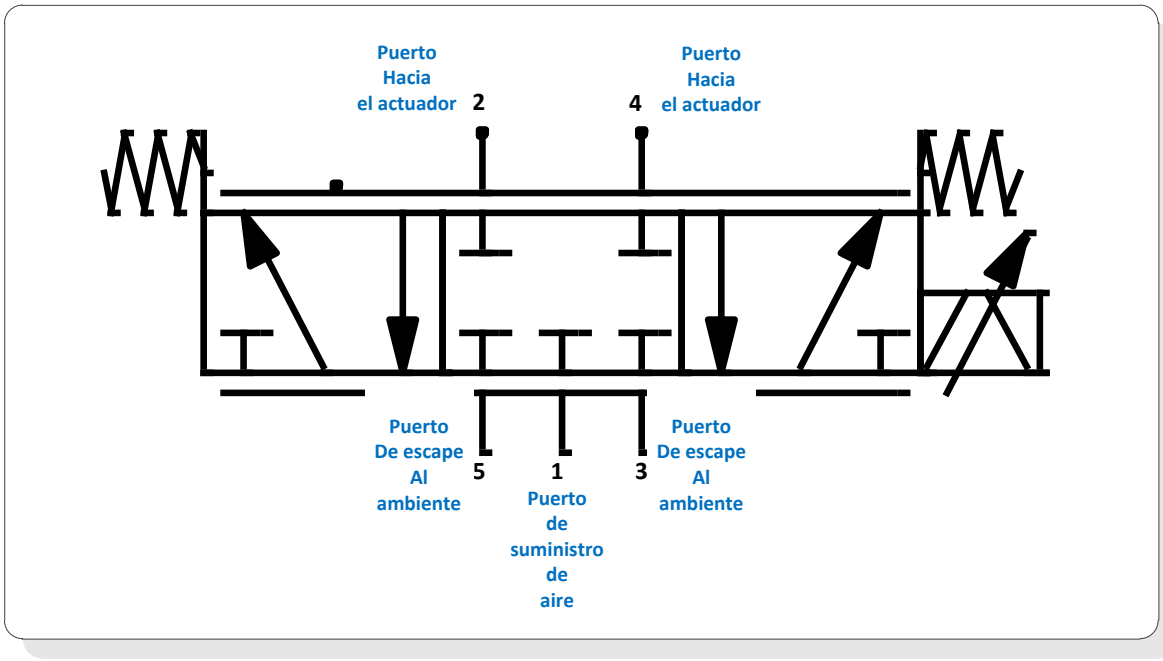


Fuente: Válvulas posicionadoras MPYE, FESTO®

Como se observa en la Figura 16, la señal de control de intensidad eléctrica se da en una escala de 4 a 20 mA, la naturaleza de la válvula indica tres sentidos de trabajo (Figura 17), en uno de ellos se conecta el puerto de suministro de aire con el que se dirige a la cámara del actuador mientras la otra salida del actuador se

conecta con uno de los puertos de salida de aire; el otro de ellos, realiza el mismo proceso pero intercambiando los puertos del actuador sobre el que realiza la acción. De esta manera el elemento del cilindro puede moverse en un sentido y en el otro, estos dos estados según la Figura 16 se presentan en los rangos de trabajo [12 - zona muerta a 4 mA] y [12 + zona muerta a 20 mA], donde la zona muerta corresponde a una pequeña área cerca a la magnitud de la señal que mantiene centrada a la válvula donde no se presenta movimiento alguno del actuador.

Figura 17. Posiciones válvula proporcional



Fuente: Válvulas posicionadoras MPYE, FESTO®

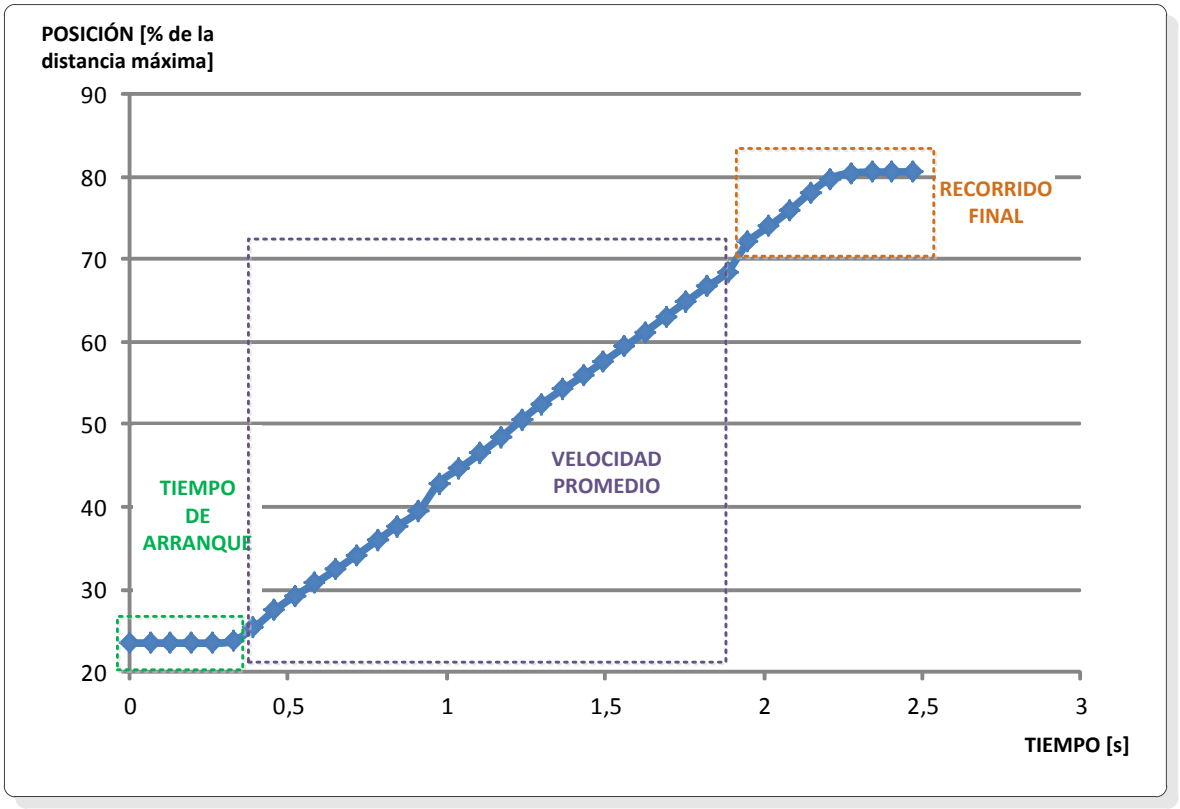
El tercer estado consiste en la posición donde la válvula se encuentra centrada lo que quiere decir que los puertos del actuador se bloquean evitando el movimiento del elemento dentro del cilindro, aunque si la válvula se centra cuando el sistema se encontraba en movimiento se experimenta un desplazamiento antes de

detenerse, esto se debe a los efectos inerciales debido a la velocidad que trae el objeto deslizante antes de ser detenido.

El modelo de control planteado para el robot neumático requiere de la adquisición de tres variables: Velocidad promedio del desplazamiento, Recorrido final (este se refiere a la distancia recorrida antes de detenerse, se produce debido al efecto inercial del sistema, se mide a partir del momento en que la válvula es centrada hasta el punto donde el robot se detiene), Tiempo de Arranque (es el tiempo que le toma al elemento iniciar su movimiento una vez la válvula se ha iniciado a un porcentaje de apertura determinado), las cuales se dan en función de los diferentes valores de apertura de la válvula. Sin embargo, debido a que no hay retroalimentación de dicha propiedad, se miden en relación al porcentaje de la señal de control enviada al preactuador, tomando como 0% el valor de 4 mA, 50% son 12 mA y el 100% corresponden a 20 mA.

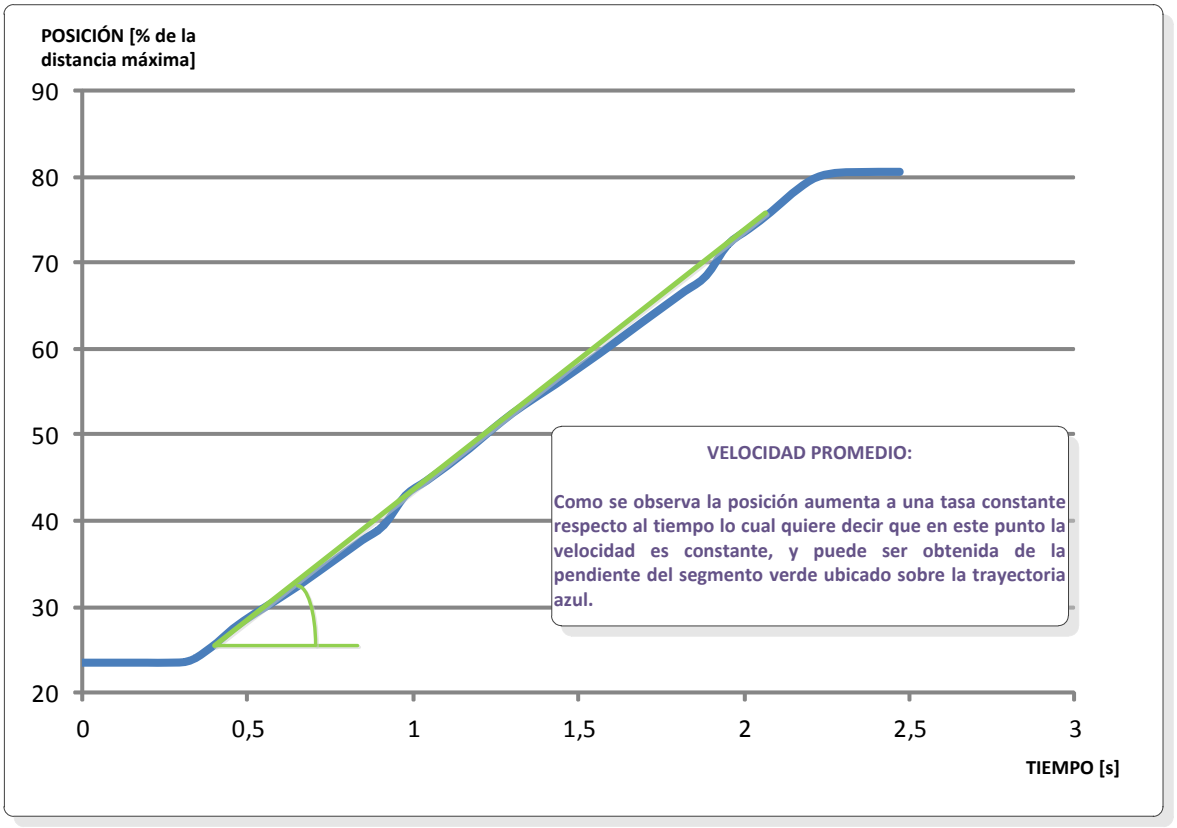
El proceso para medir las variables se realiza de la siguiente manera: en primer lugar se envía una señal de control constante al solenoide de la válvula con el fin de crear una apertura determinada en el puerto que conecta al preactuador con el actuador. Una vez se activa el solenoide con esta señal el sistema toma los datos de la posición respecto al tiempo resultando una curva como la presentada en la Figura 18 de estos datos para la posición el robot mide los valores de la velocidad promedio del sistema, recorrido final y tiempo de arranque para cada porcentaje de la señal de control enviada a la válvula.

Figura 18. Posición contra tiempo sistema neumático



Se puede hablar de velocidad promedio ya que como se observa en la Figura 19, el proceso se realiza casi a una velocidad constante.

Figura 19. Velocidad constante en el sistema neumático



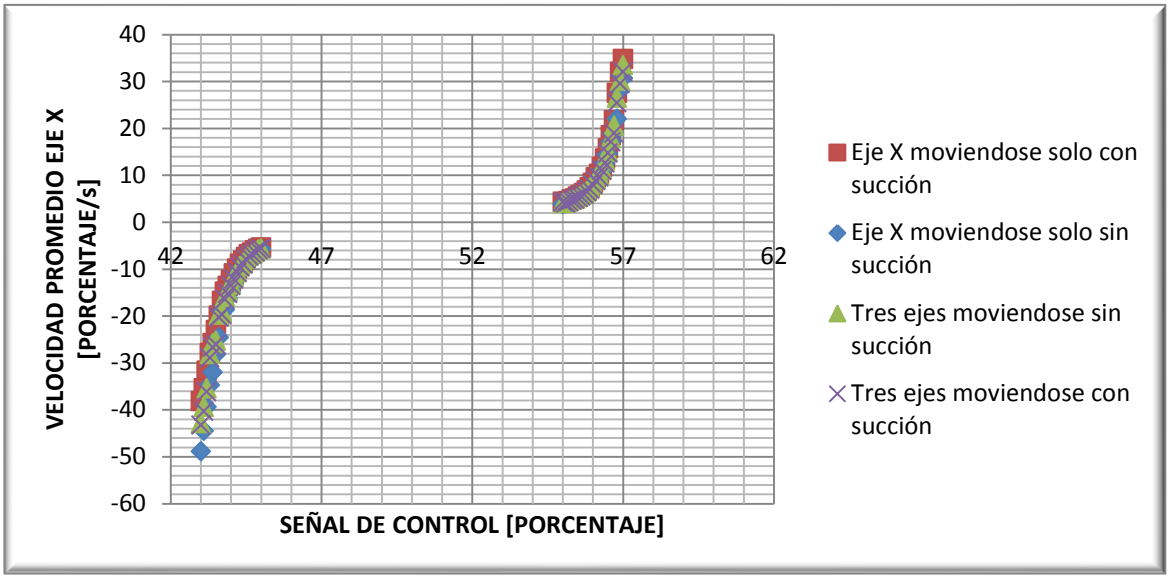
Debido al sistema actual no se hace necesario tomar estos valores para todas las aperturas, puesto que los valores dentro de la zona muerta no merecen ser estudiados ya que no existe movimiento. En esta parte se introduce un nuevo concepto el cual para este proyecto se denomina Zona de Velocidad Intermitente, el cual se usa para detallar aquella área cerca al porcentaje de la señal de control máxima (50%) que centra a la válvula; en este lugar de trabajo el movimiento realizado por el sistema no corresponde a una función de movimiento continua por lo cual no son deseados para la operación del sistema, por tanto para el trabajo con sistemas neumáticos se recomienda operar por fuera de esta zona.

Otra razón por la que no se hace necesario tomar todos los valores de porcentaje de apertura para el análisis, es que debido a la estructura del sistema actual

donde el elemento de suministro (unidad de mantenimiento) da un caudal nominal de 730 L/min y el caudal nominal de la válvula proporcional es de 1400 L/min (Ver anexo F), se llega a un punto donde por más que se aumente el área de apertura de la válvula el caudal no puede aumentar muchos más.

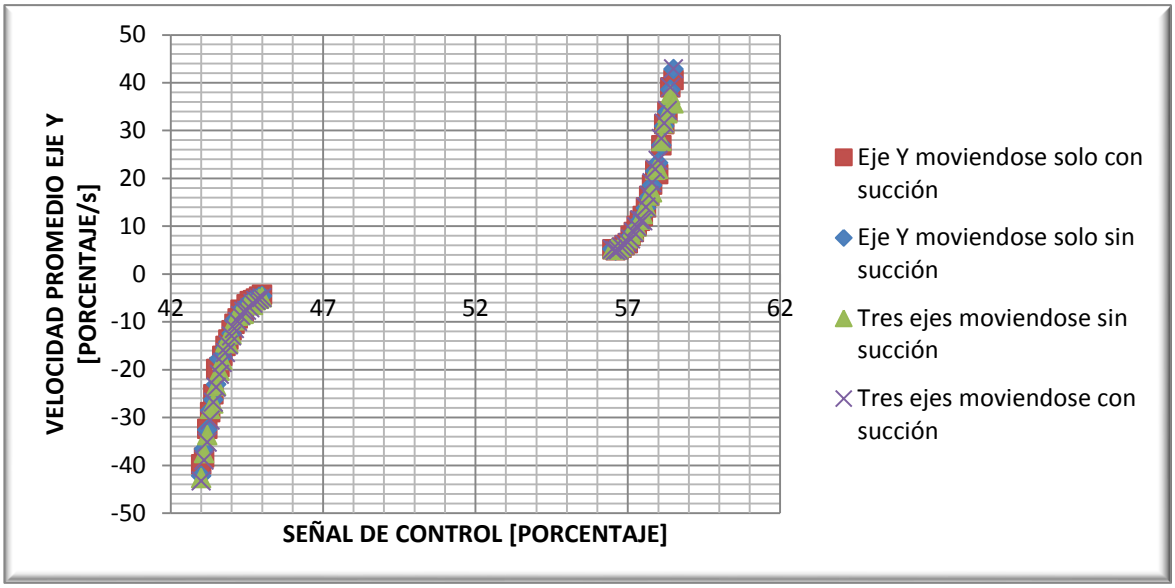
Uno de los problemas que surgen para este método de control es conocer en qué momento puede ser válido la toma de los datos para su uso posterior, pues hay una infinidad de factores que podrían hacer dudar al lector sobre qué tan confiable es el dato obtenido para cada señal de control. Por ejemplo, para el caso en el que solo se mueve un eje a la vez o si el robot se encuentra sujetando un elemento, *¿es la velocidad promedio diferente?*, para resolver esta duda se tomaron los datos de las velocidades promedios para cada válvula proporcional en el rango de señales de control (recuerde esta se da en porcentaje) fuera de la zona de velocidad intermitente y debajo del valor donde el caudal no aumenta debido a la configuración, para diferentes condiciones de trabajo tales como: movimiento de un eje a la vez, con y sin un elemento sujetado, los tres ejes moviéndose a la vez, con y sin un elemento sujetado. La conclusión obtenida para cada válvula fue la misma, que para este rango de señales de control (o las aperturas de forma directa) dichos valores de velocidades promedio no diferían de forma significativa. Lo mismo ocurre para el valor del tiempo inicial y del recorrido final, por tanto se válida que dichos datos pueden ser tomados en cualquier situación, por lo cual el proceso inicial que realiza este método de control para obtener los datos de las tres variables para cada señal de control lo hace con los tres ejes moviéndose a la vez, con el fin de crear un sistema de calibración que no dure un tiempo mayor a los 10 minutos.

Figura 20. Velocidad promedio para la válvula proporcional del eje X en diferentes condiciones



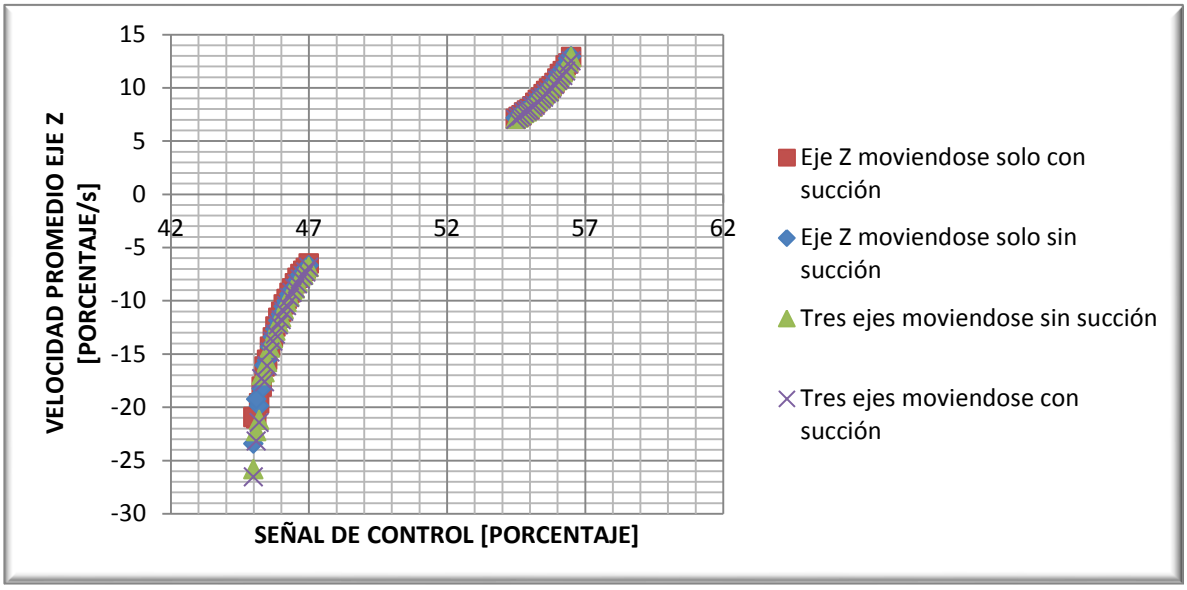
Fuente: Proceso de experimentación en el robot neumático

Figura 21. Velocidad promedio para la válvula proporcional del eje Y en diferentes condiciones



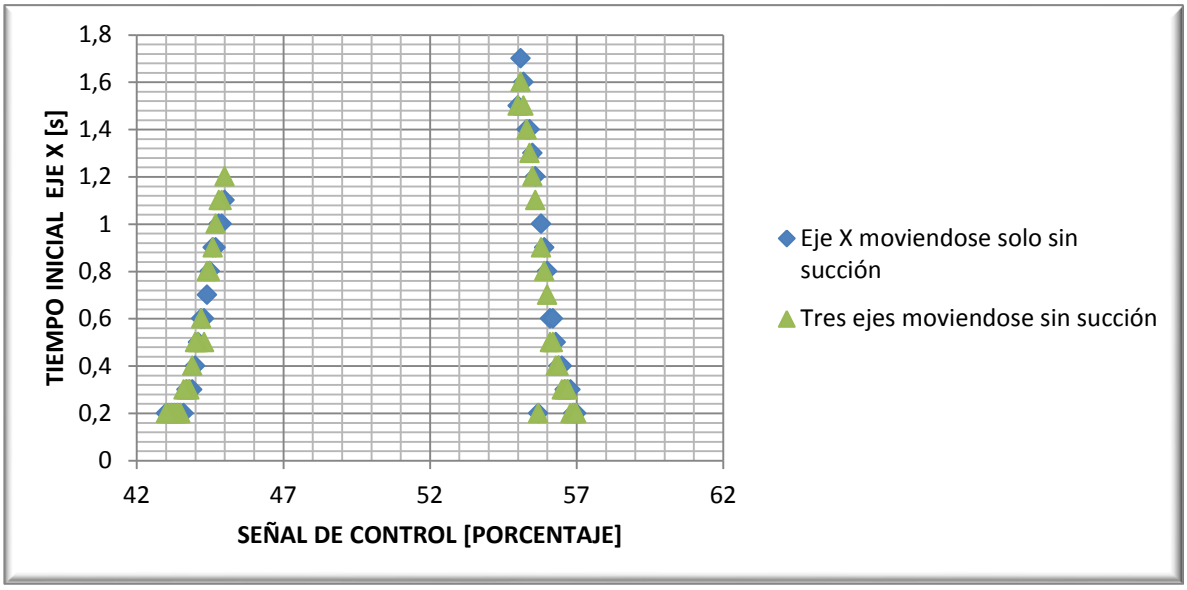
Fuente: Proceso de experimentación en el robot neumático

Figura 22. Velocidad promedio para la válvula proporcional del eje Z en diferentes condiciones



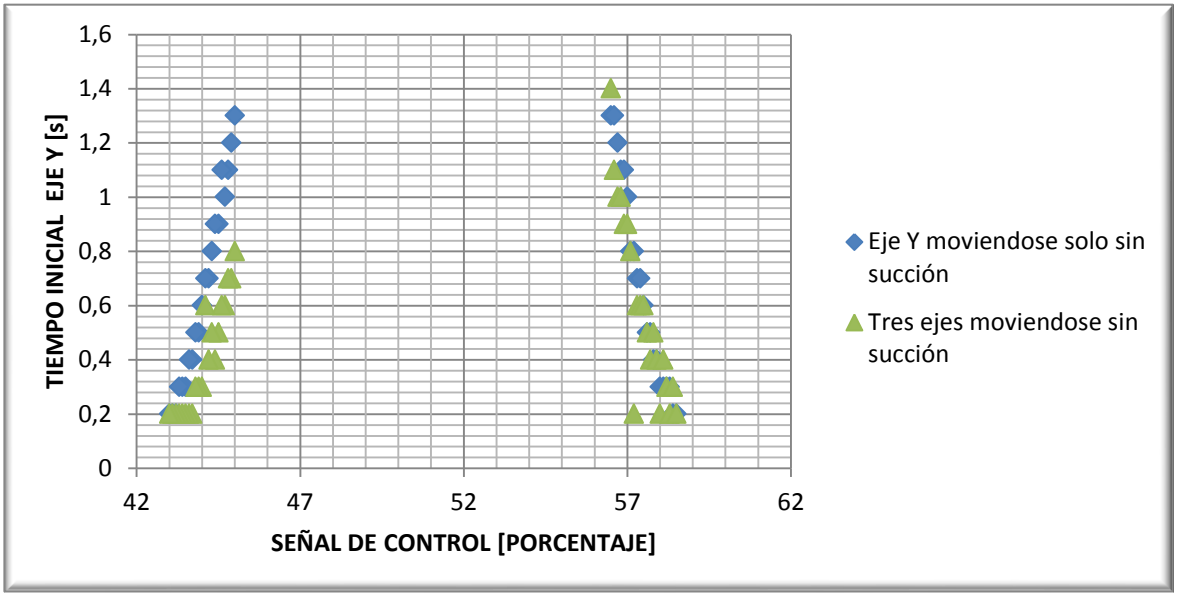
Fuente: Proceso de experimentación en el robot neumático

Figura 23. Tiempo inicial eje X para varios estados



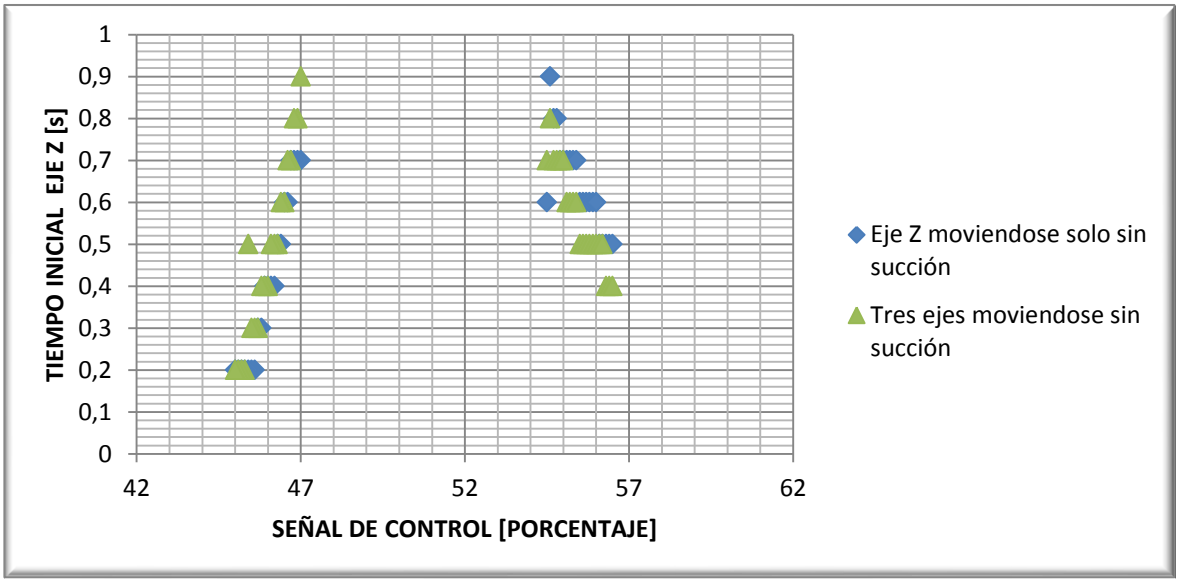
Fuente: Proceso de experimentación en el robot neumático

Figura 24. Tiempo inicial eje Y para varios estados



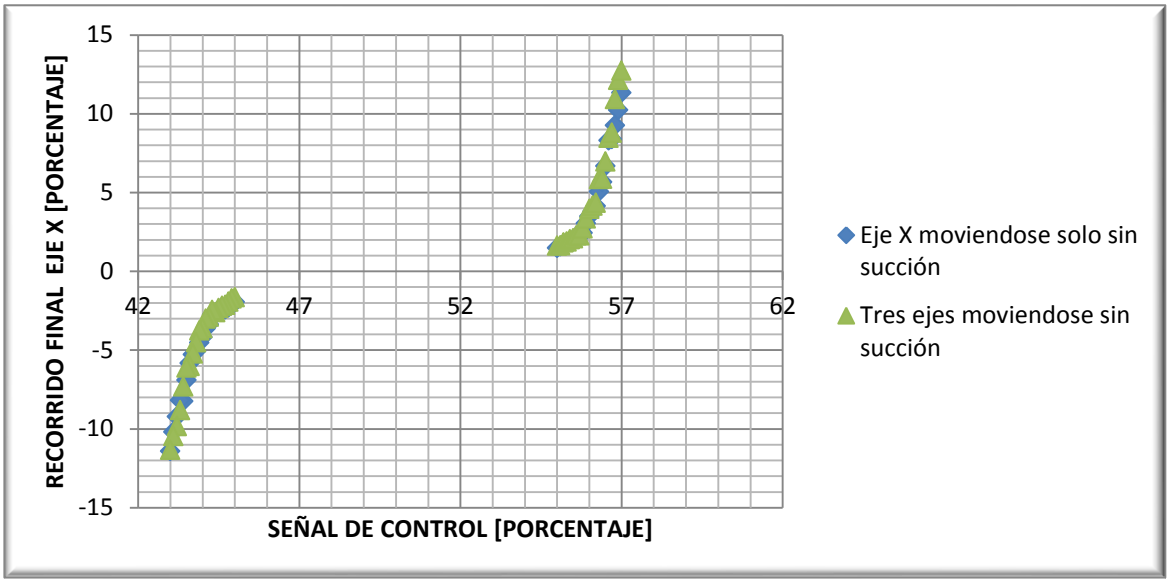
Fuente: Proceso de experimentación en el robot neumático

Figura 25. Tiempo inicial eje Z para varios estados



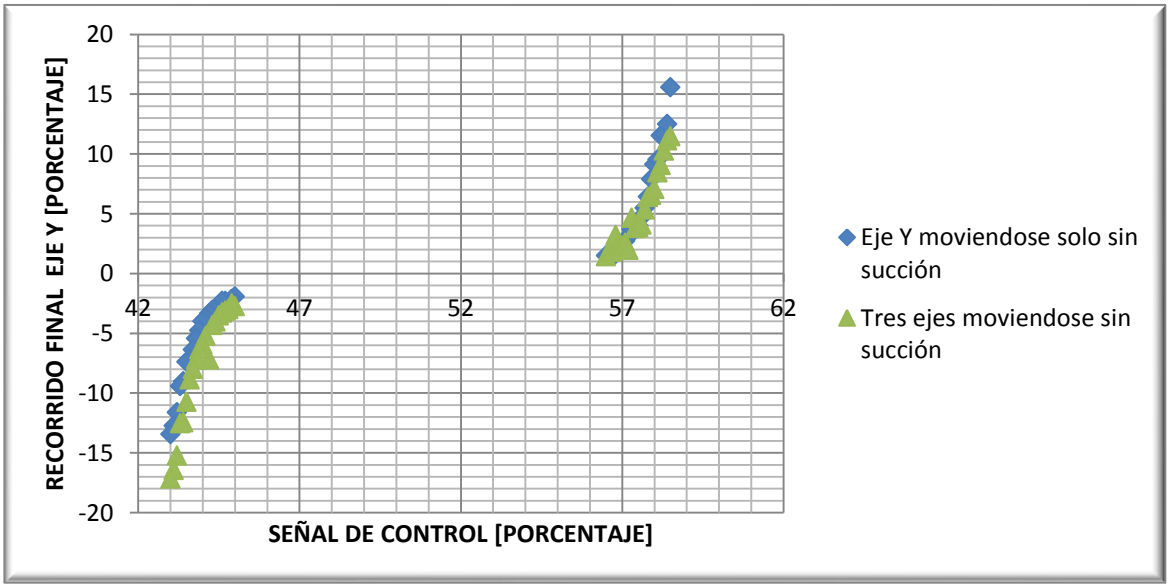
Fuente: Proceso de experimentación en el robot neumático

Figura 26. Recorrido final eje X para varios estados



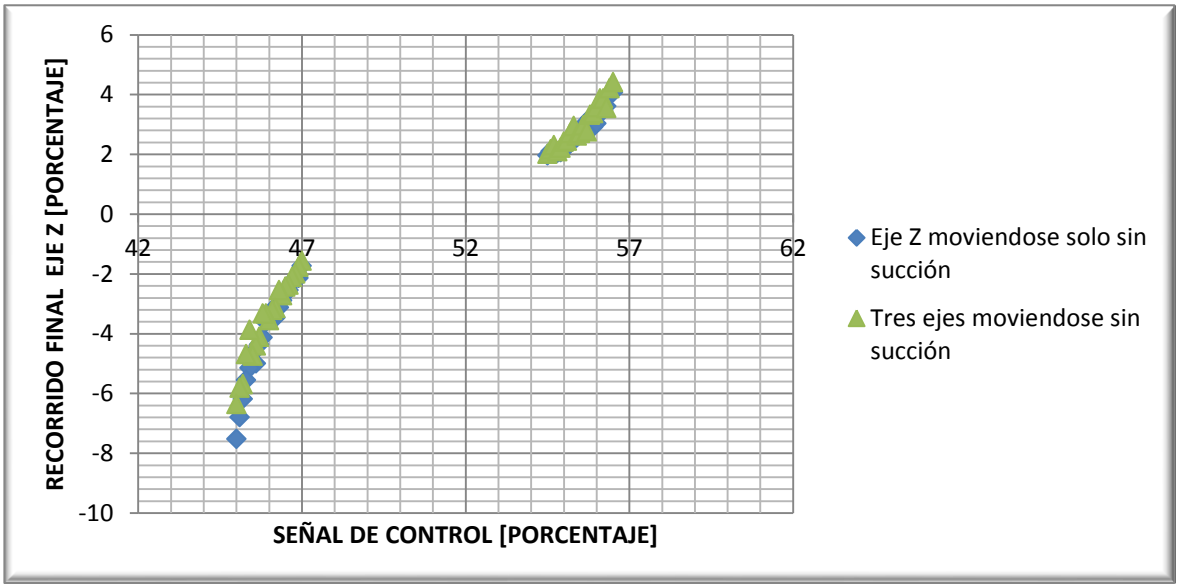
Fuente: Proceso de experimentación en el robot neumático

Figura 27. Recorrido final eje Y para varios estados



Fuente: Proceso de experimentación en el robot neumático

Figura 28. Recorrido final eje Z para varios estados



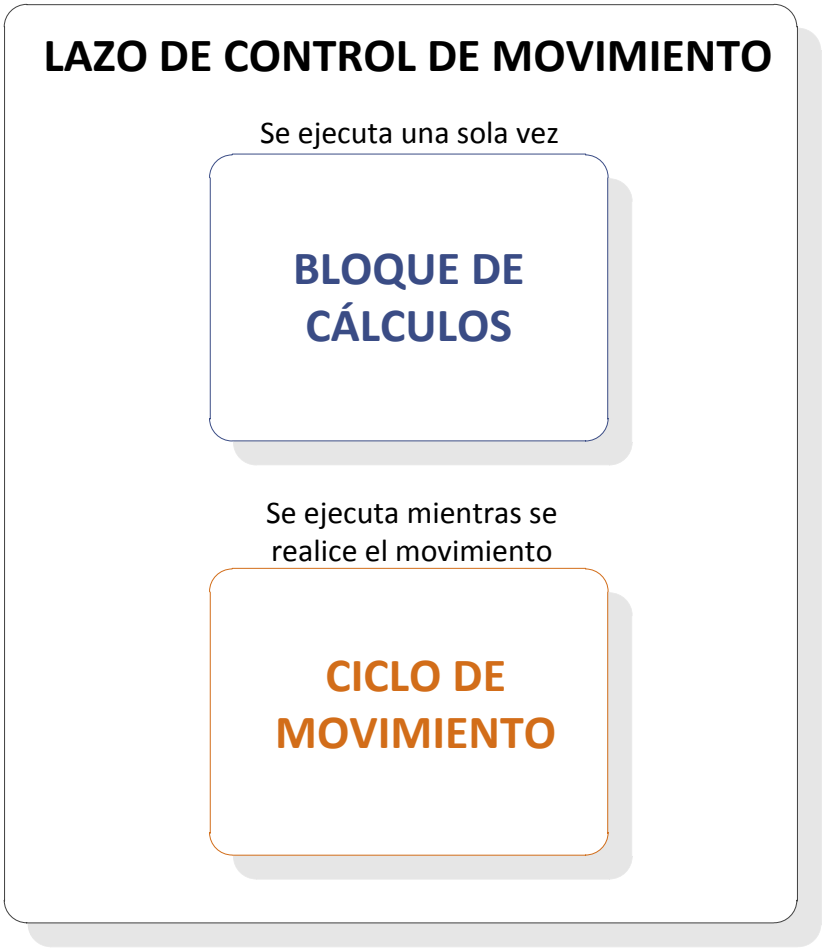
Fuente: Proceso de experimentación en el robot neumático

Una vez el sistema realiza la calibración y obtiene los datos de las tres características para cada porcentaje de la señal de control dentro de los rangos establecidos, los cuales son: 55% a 58% y 46% a 43,1 % en el eje X, 56,5% a 58,5% y 45% a 42,9% en el eje Y, 54,5% a 57,4% y 47% a 44,1% en el eje Z. El sistema puede ejecutar cualquiera de los ciclos de control creados para el movimiento del robot.

- **Lazo de control de posición: Tiempo como parámetro de trabajo (Velocidades Variables).**

El lazo de control de posición se divide en dos procesos, el primero de ellos es la zona de cálculos previos antes de ejecutar el movimiento, éste realiza una sola vez. Entre tanto el segundo es el ciclo de ejecución, en éste se ubican los algoritmos que deben ejecutarse mientras el robot se encuentra en movimiento, aquí se ubican los cálculos que permiten el inicio y final del desplazamiento del robot (Figura 28).

Figura 29. Procesos del lazo de control



.

El bloque de cálculos para este sistema se define en las Figuras desde la número 30 hasta la 37.

El ciclo de movimiento para este sistema se define en las Figuras desde la número 38 hasta la 40.

Figura 30. Bloque de cálculos 1

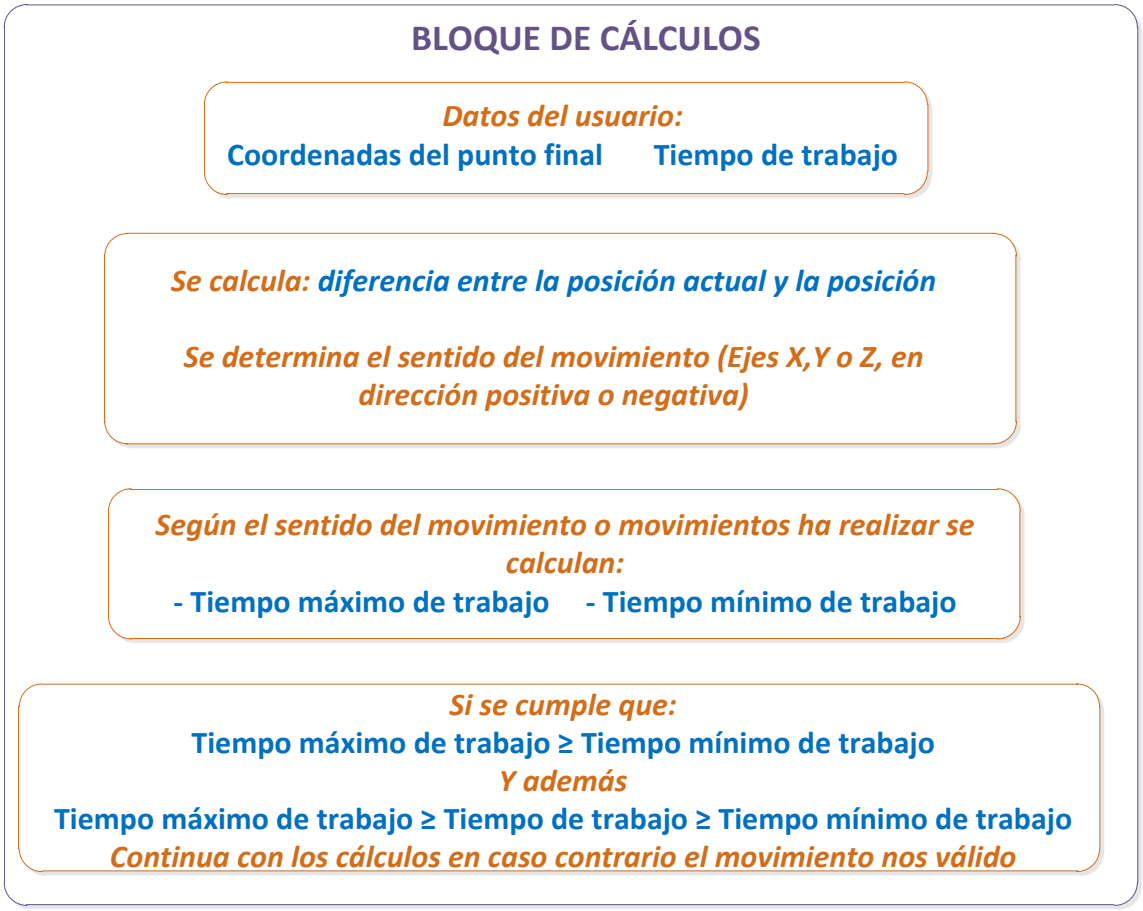


Figura 31.Bloque de cálculos 2

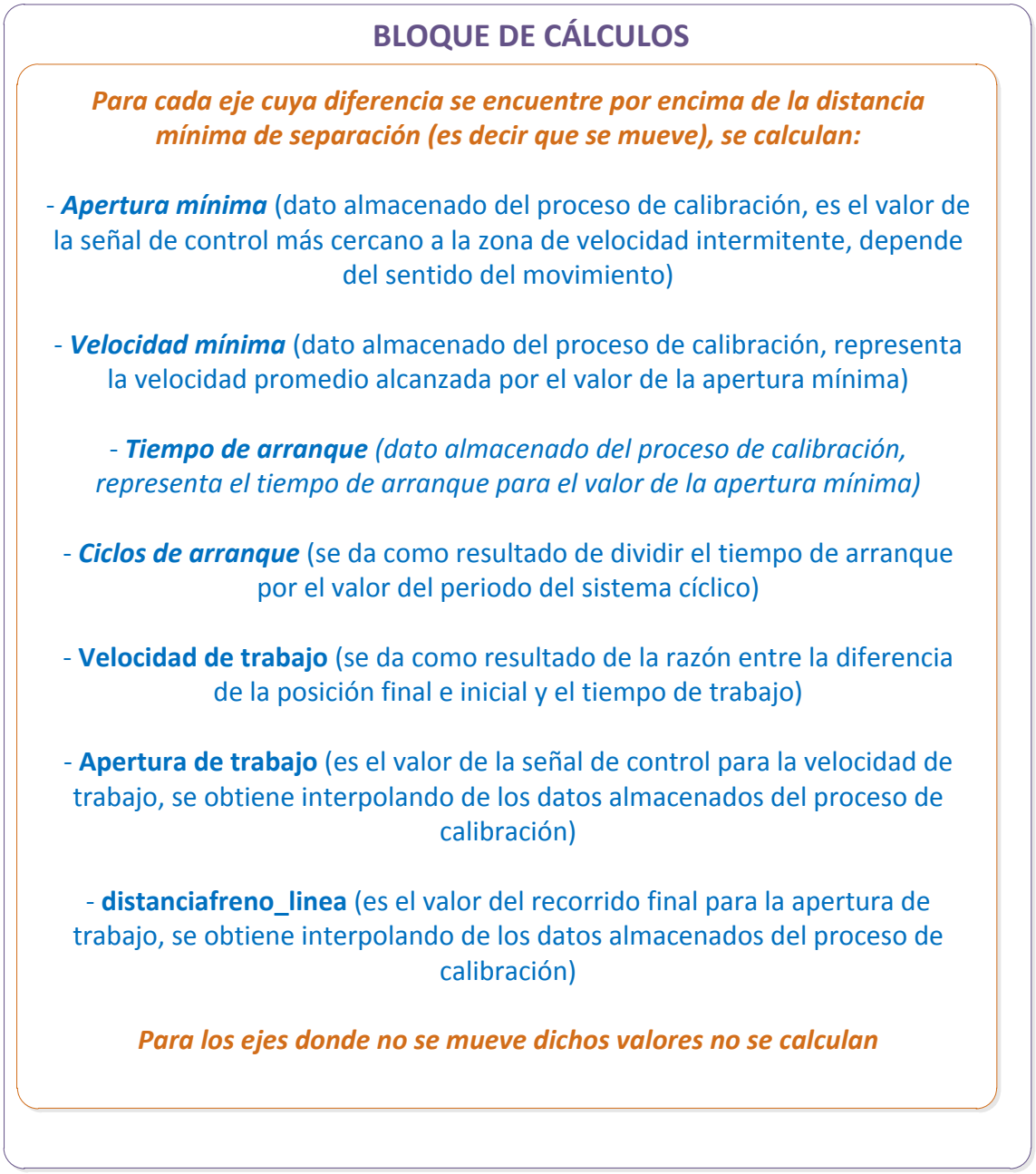


Figura 32. Bloque de cálculos 3

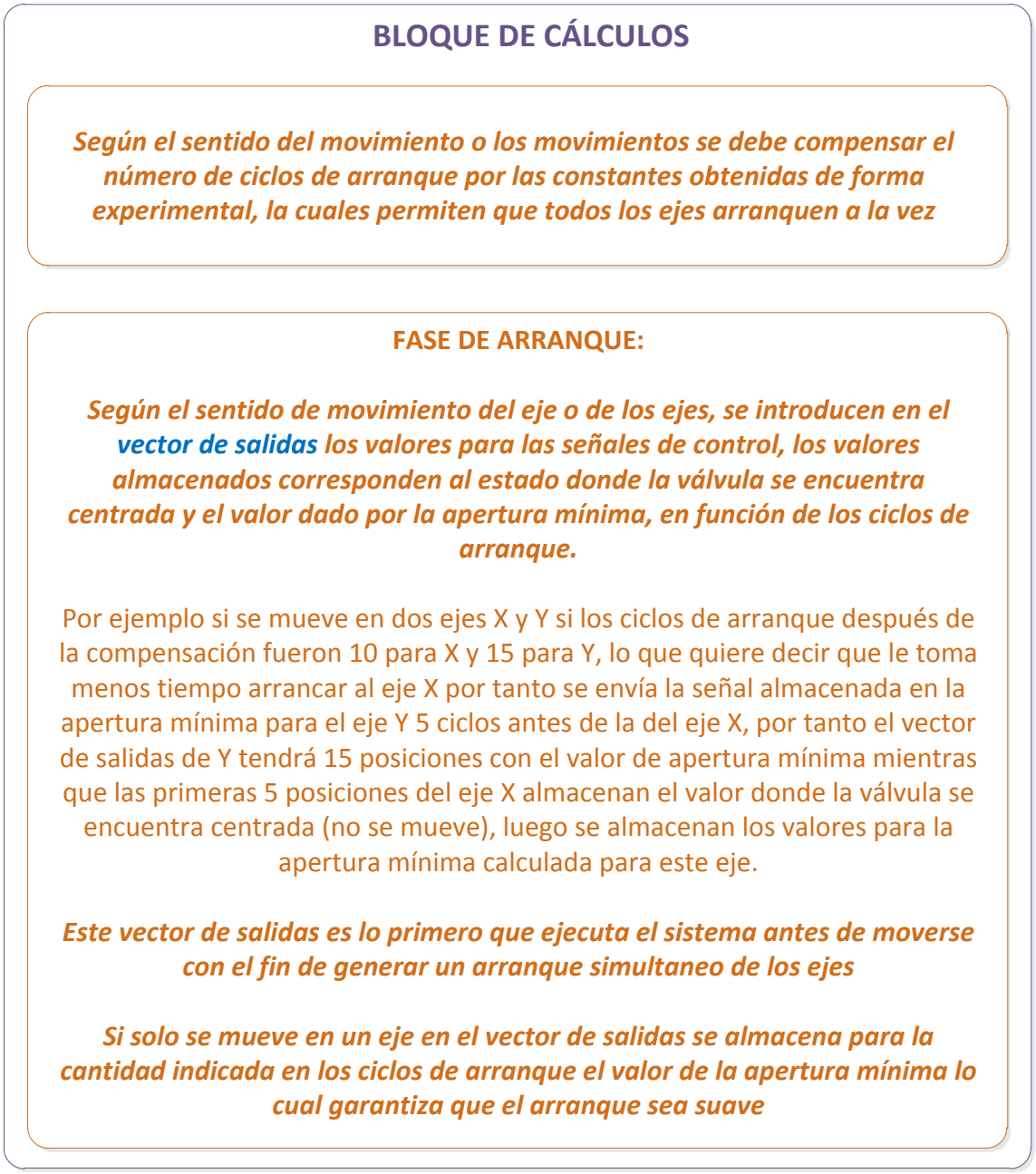


Figura 33. Bloque de cálculos 4

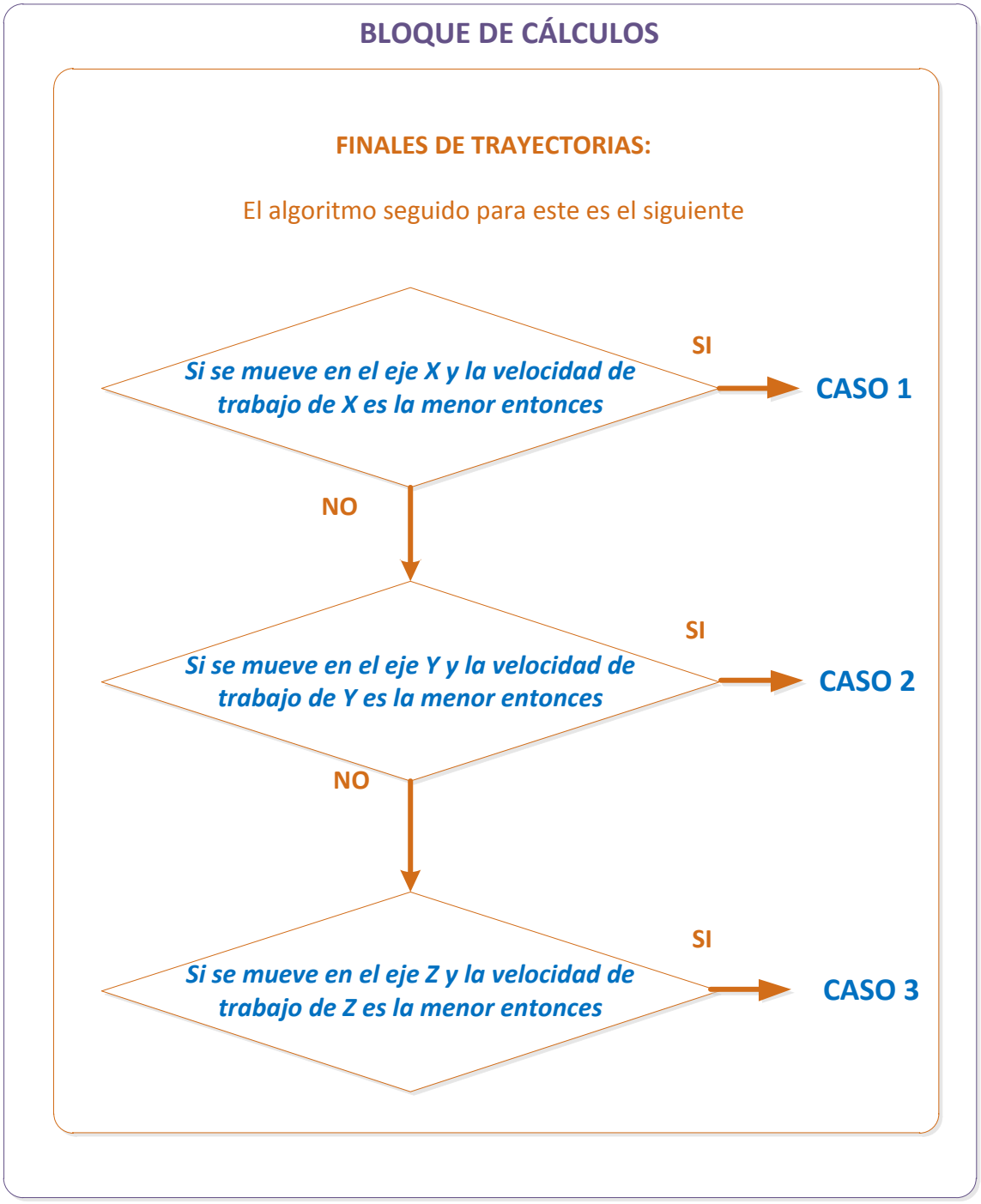


Figura 34.Bloque de cálculos 5

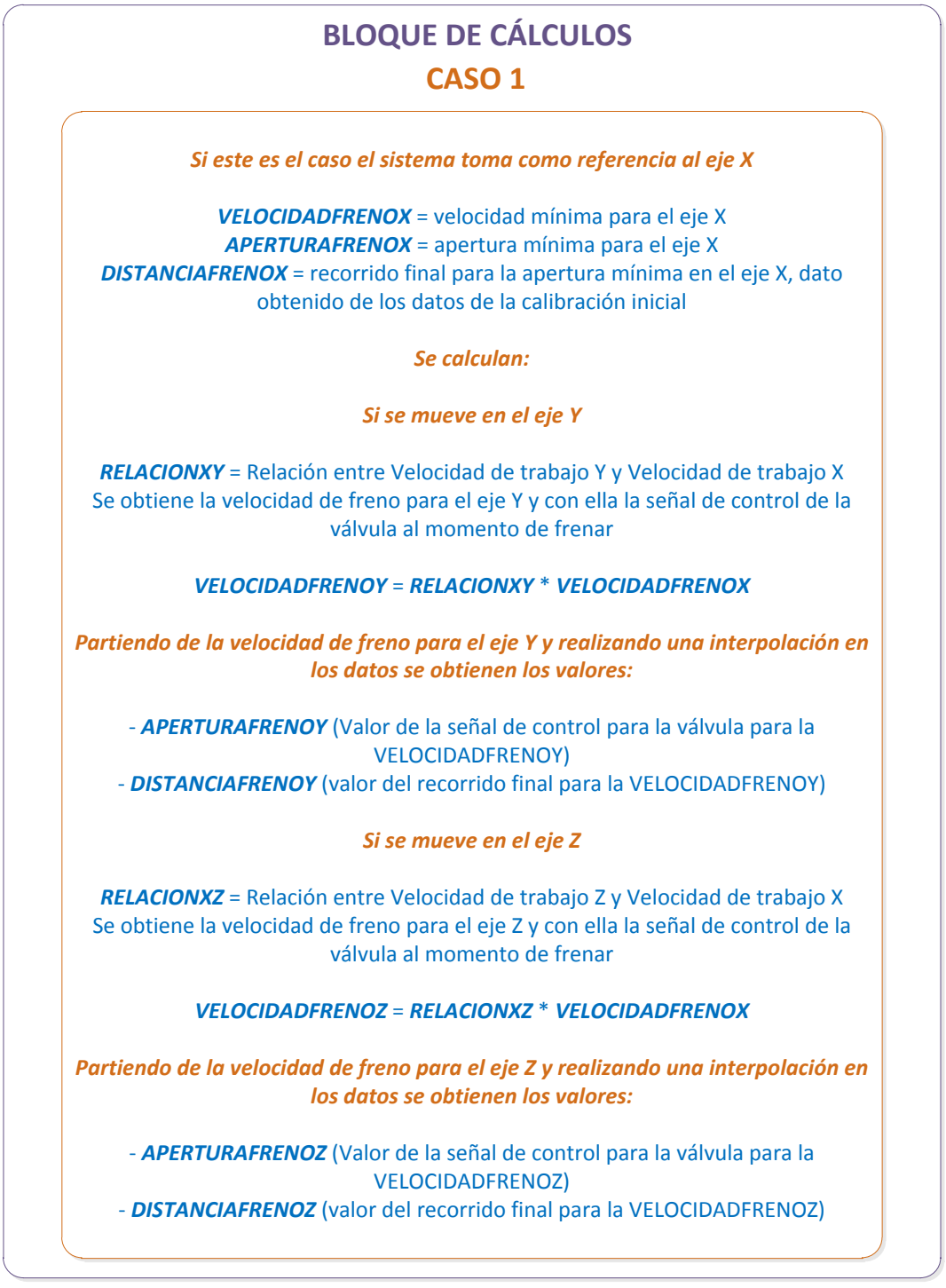


Figura 35. Bloque de cálculos 6

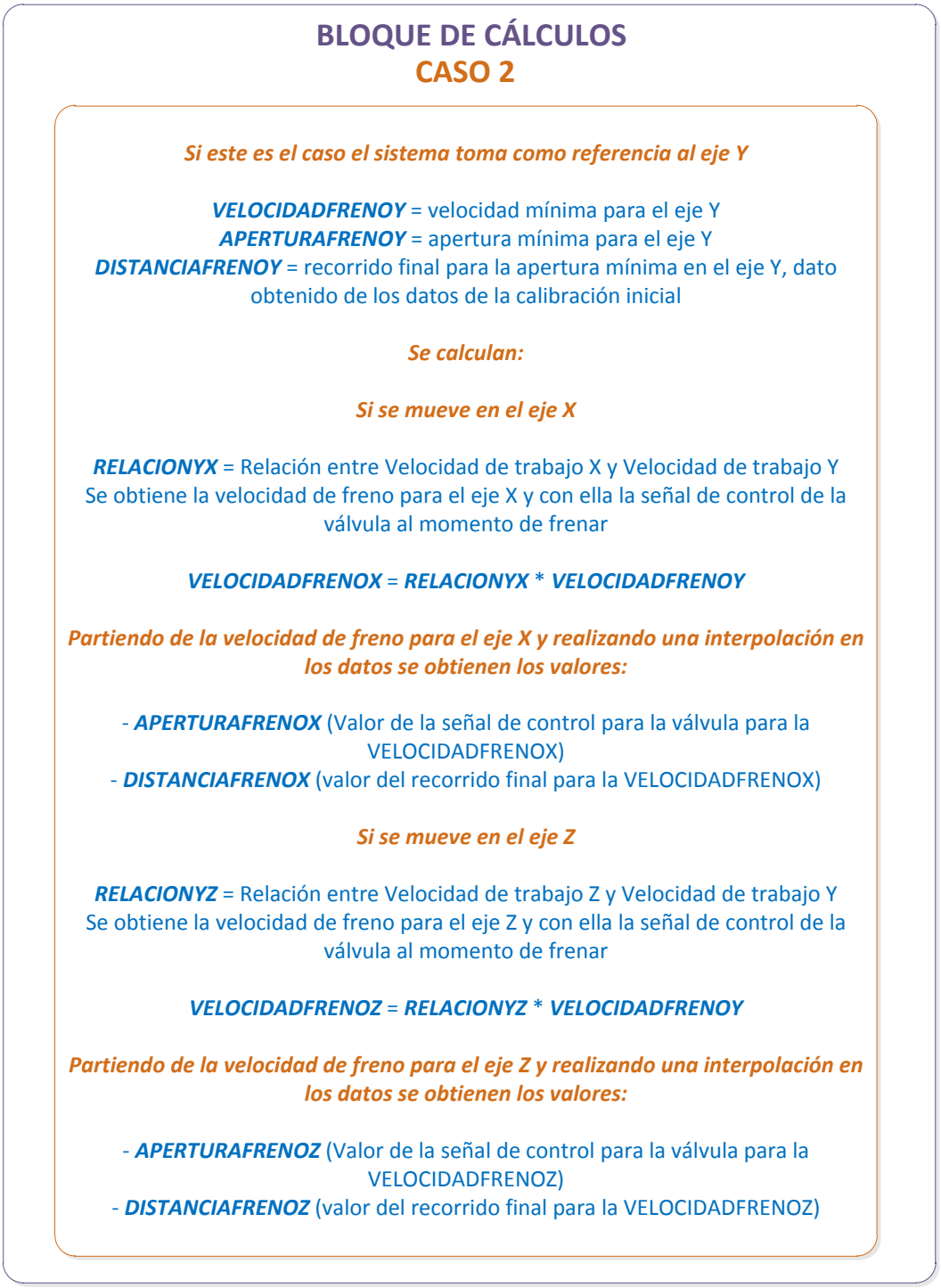


Figura 36. Bloque de cálculos 7

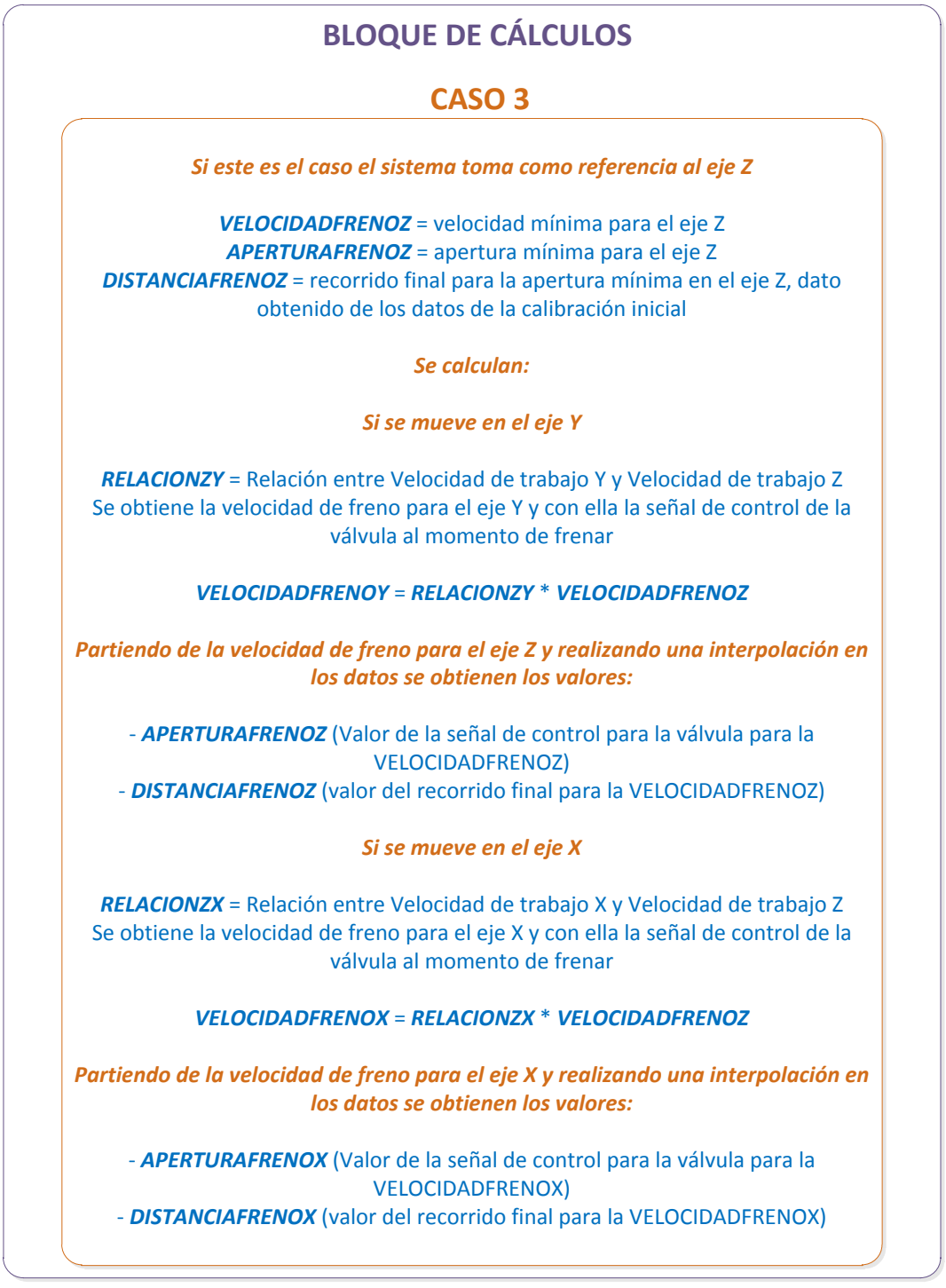


Figura 37. Bloque de cálculos 8

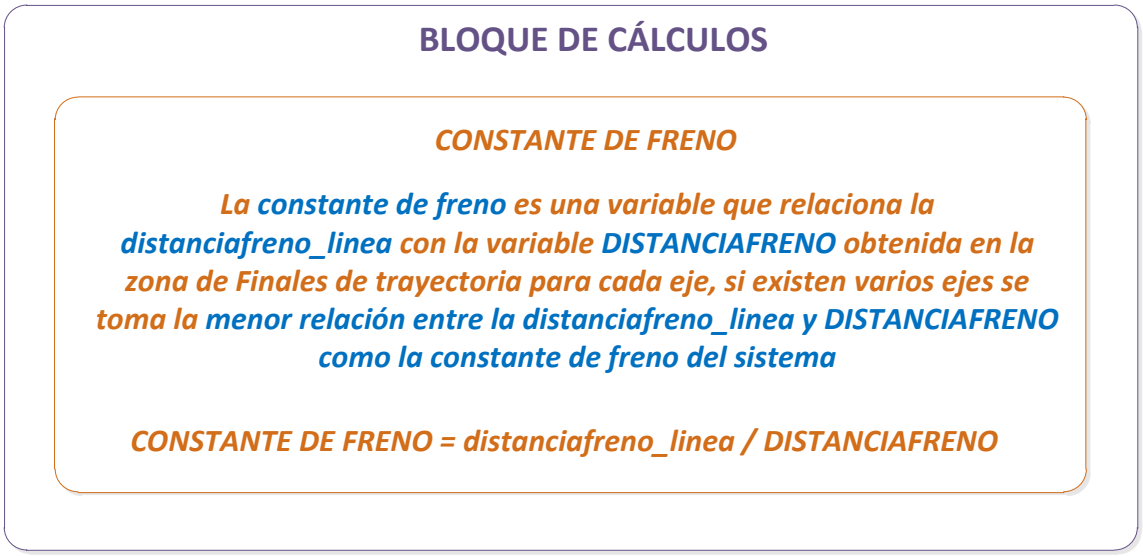


Figura 38. Ciclo de movimiento 1

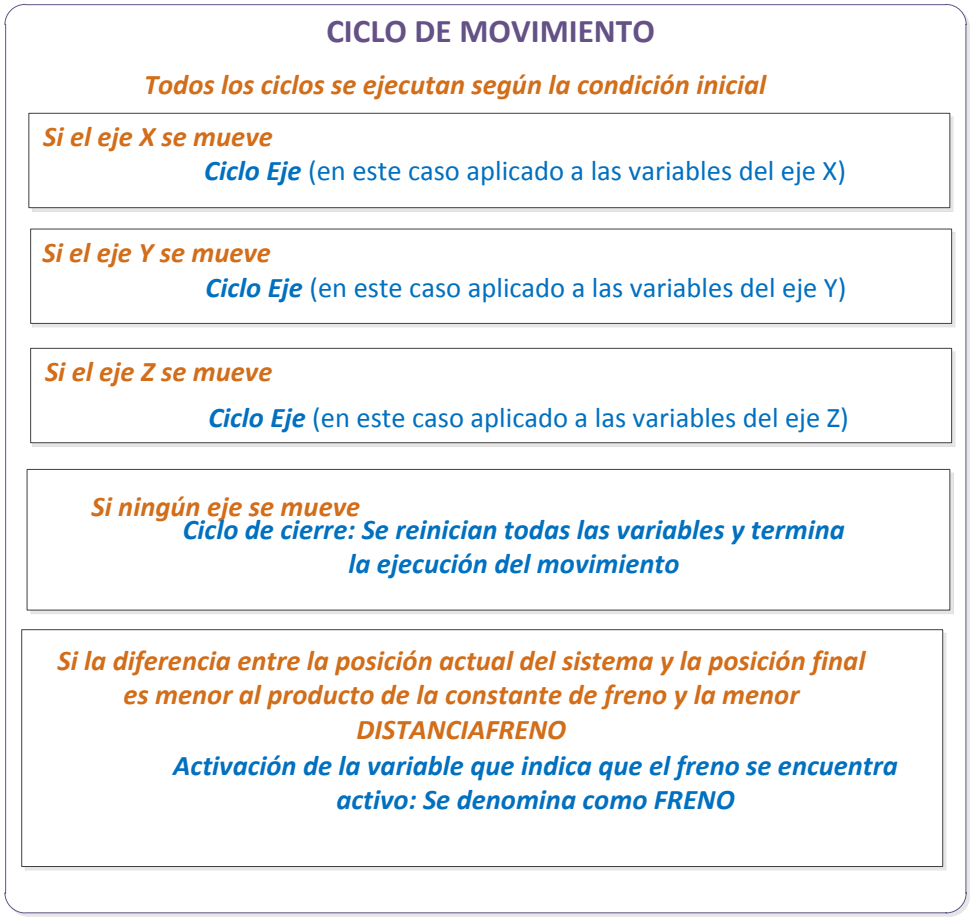
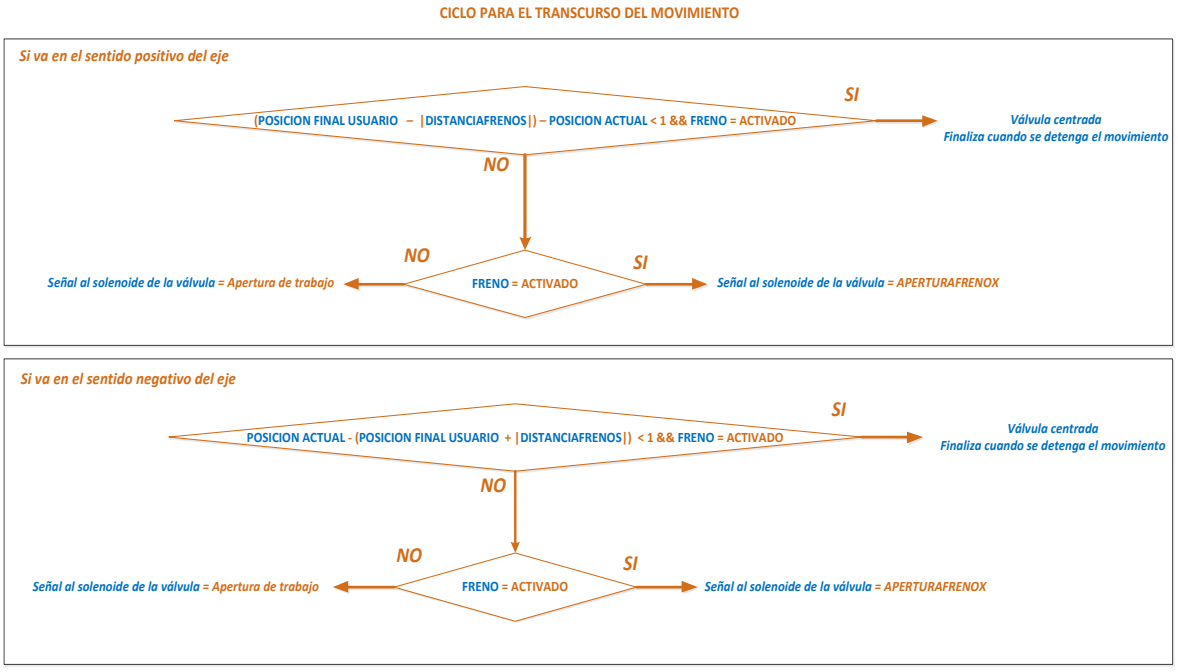


Figura 39. Ciclo de movimiento 2

Figura 40. Ciclo de movimiento 3



- **Lazo de control de posición: Velocidad única.**

Al igual que para el lazo de posición con el tiempo como parámetro, este sistema de control se basa en dos procesos como los ilustrados en la Figura 28.
El proceso del bloque de cálculos se describe en la Figura 41.
El proceso del ciclo de movimiento se presenta en las Figuras 42 y 43.

Figura 41. Bloque de cálculos segundo control

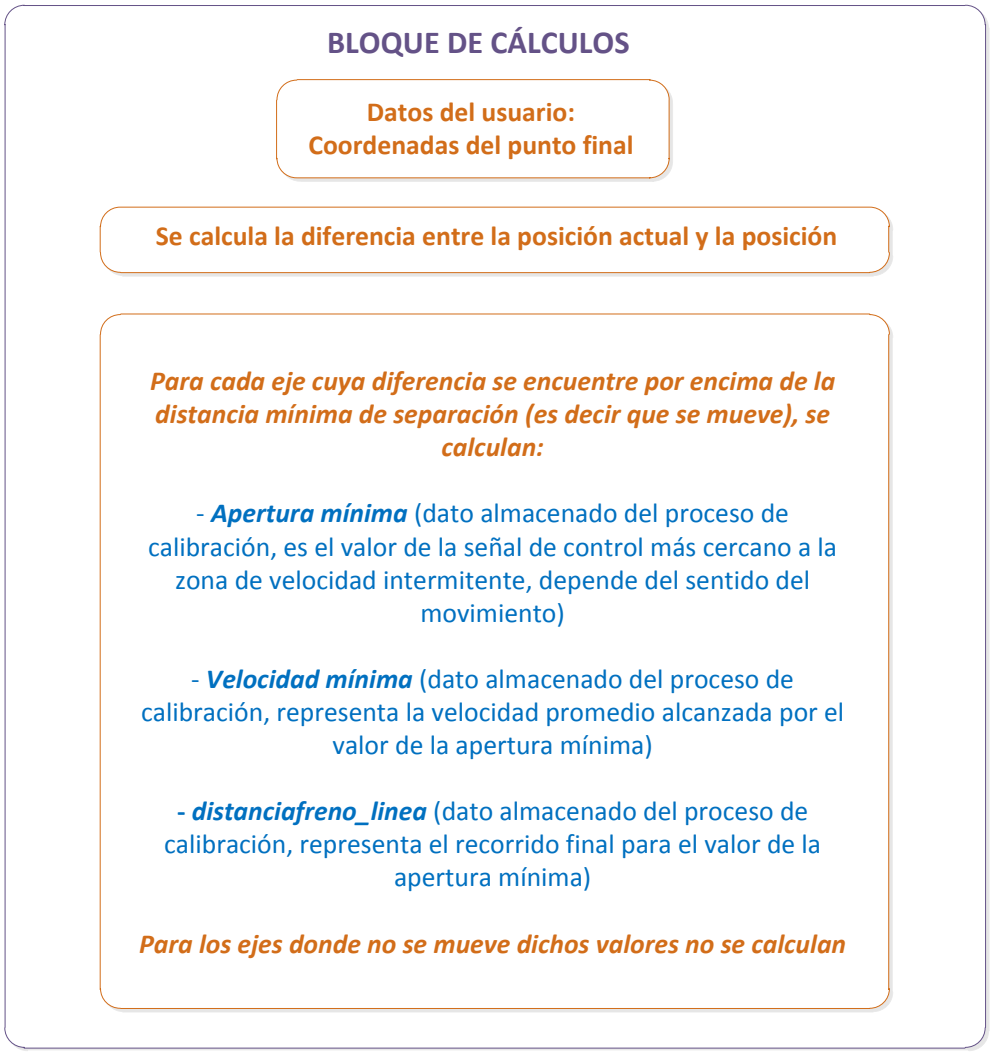
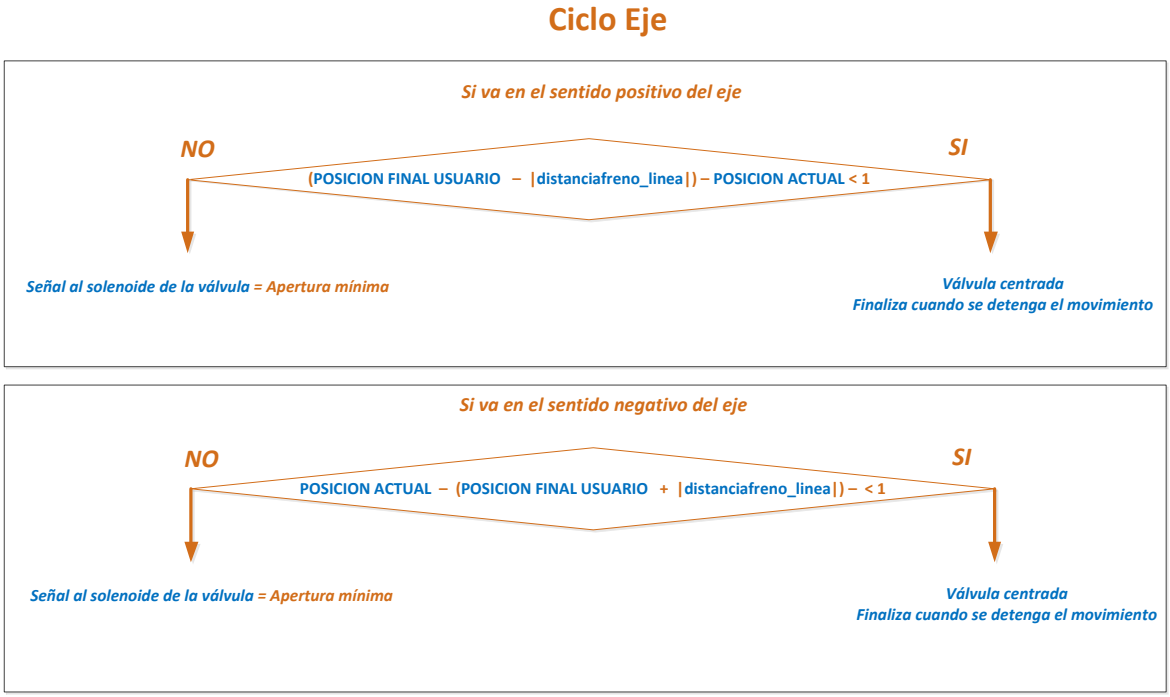


Figura 42. Ciclo de movimiento segundo control 1

Figura 43. Ciclo de movimiento segundo control 2



6. ROBOT NEUMÁTICO: INTERFAZ – VENTANA DE DATOS – PROGRAMA EN EL AUTÓMATA

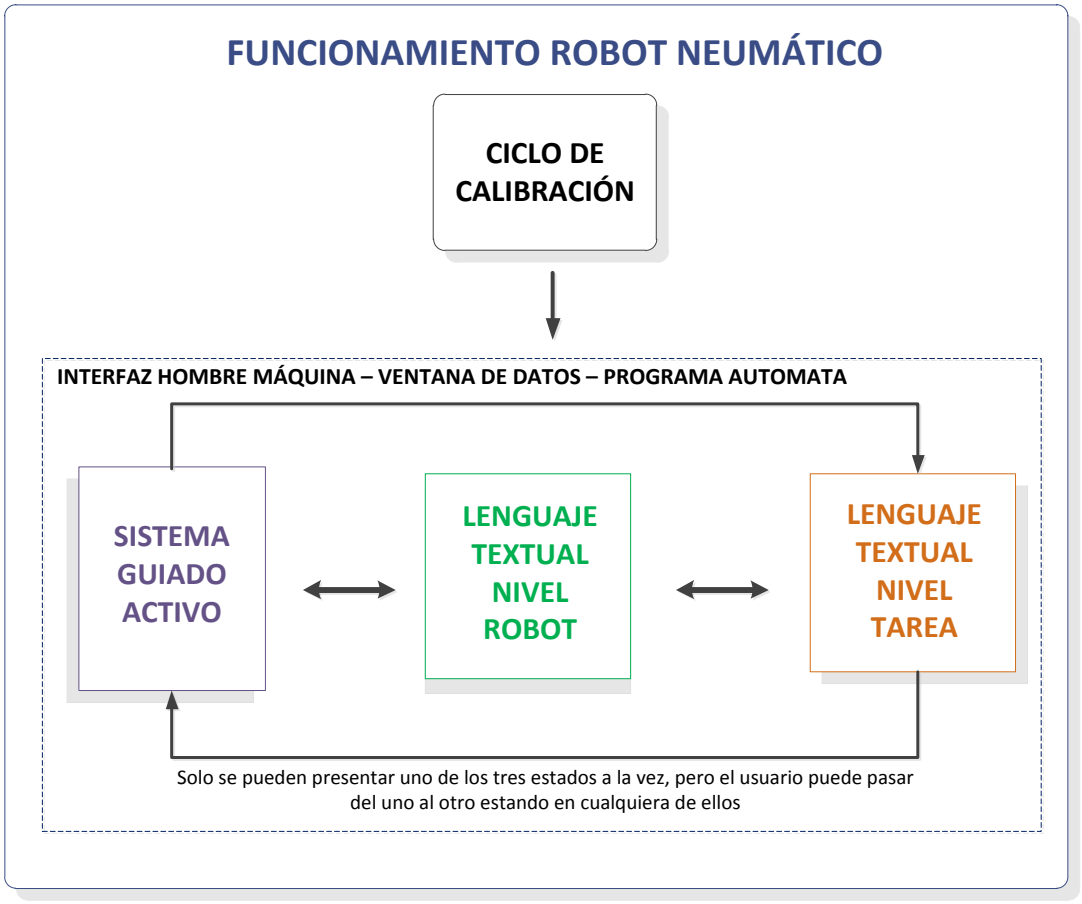
Figura 44. Robot neumático cartesiano sistema final

Fuente: Banco de trabajo, Laboratorio de Automatización Industrial

Para cada lenguaje de programación en el robot neumático se cuenta con un conjunto interfaz – ventana de datos – programa en el autómata, que cumple con las necesidades presentadas según la forma de trabajo. Los cuales son desarrollados usando los programas ***STEP 7 TIA PORTAL V11*** (programación para el autómata) y ***WINCC PROFESSIONAL*** (Creación y aplicación de interfaces hombre – máquina en computadores) parte del paquete software ofrecido por la empresa SIEMENS® para el uso de sus autómatas programables.

Por ende, en el presente capítulo se describe el proceso del cómo funcionan los grupos (interfaz – ventana de datos – programa autómata) en cada lenguaje de programación, además se incluye el proceso inicial de calibración que requiere el robot para obtener los datos necesarios que posibilitan el control de movimiento.

Figura 45. Estructura de trabajo Robot neumático

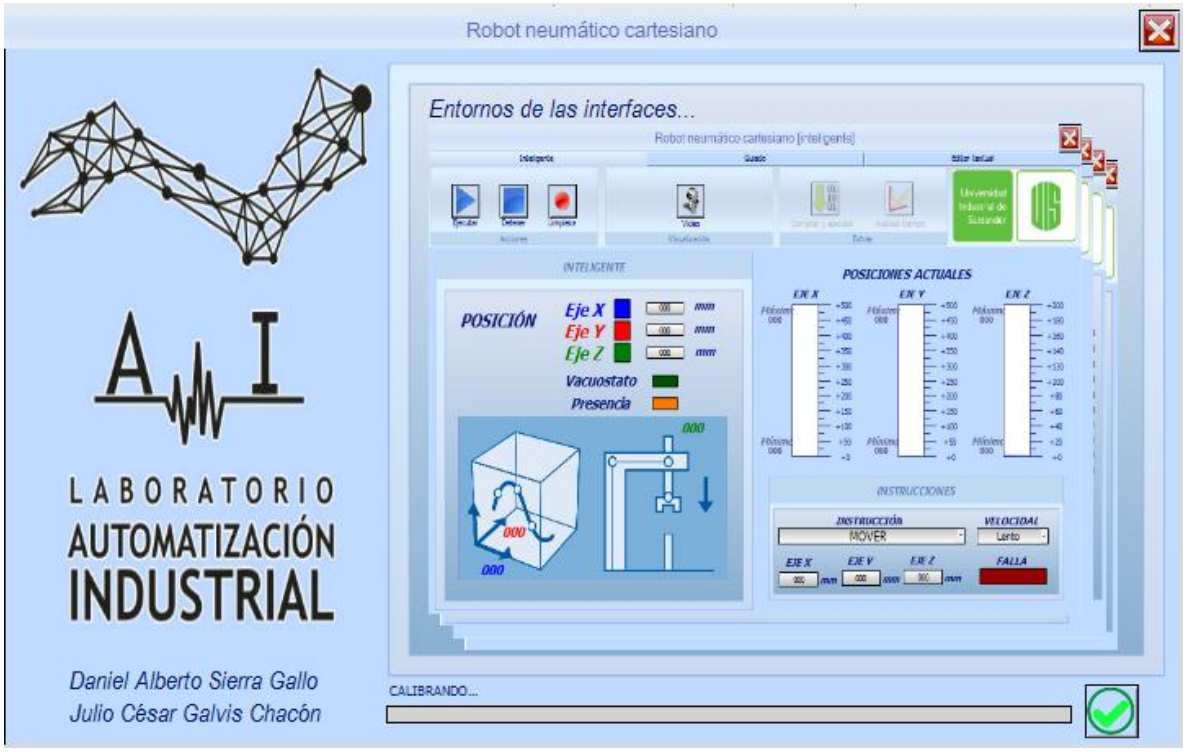


6.1 SISTEMA INICIAL DE CALIBRACIÓN

Al iniciar el sistema por primera vez realiza, como su primera actividad, el proceso de calibración, el cual consiste en tomar los datos de la velocidad promedio, el recorrido final y el tiempo de arranque (especificados para los lazos de control de movimiento en el capítulo 3) para cada apertura de las válvulas proporcionales en el sistema de trabajo.

La interfaz mostrada para este proceso se presenta en la Figura 44, en la cual la barra inferior presenta al usuario el progreso del estado de la calibración.

Figura 46. Interfaz inicial: proceso de calibración

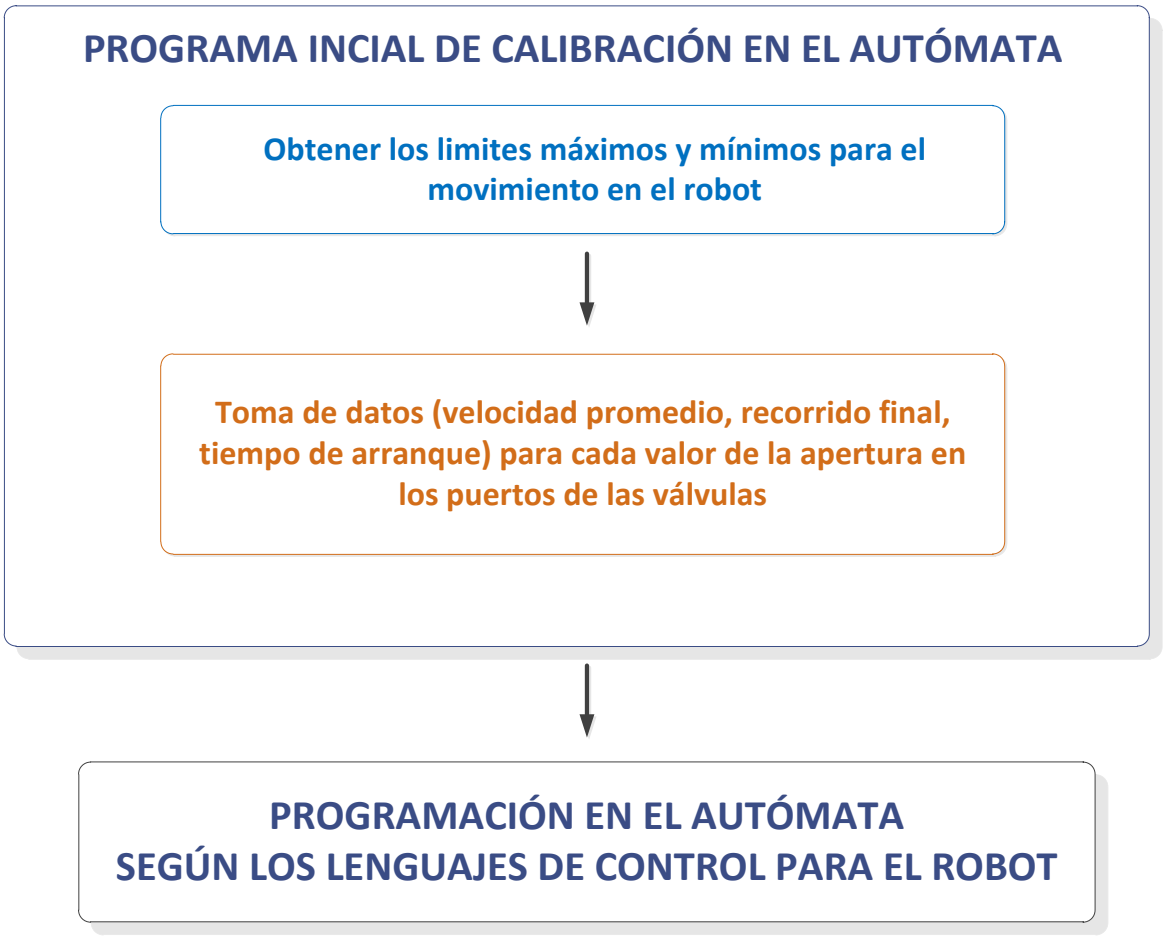


Fuente: Banco de trabajo, Laboratorio de Automatización Industrial

La ventana de datos para esta parte del proceso se basa en una sola variable, el contador de ciclos (elemento presente dentro de la programación del autómata encargado de registrar el número de veces que se han tomado los datos para cada apertura de las válvulas), el estado de dicha variable se muestra en la barra de progreso de la interfaz, al llegar al valor máximo de pasadas (60 veces), la programación dentro del autómata continua hacia las funciones presentadas para cada lenguaje de programación, al igual que lo hace la interfaz en el computador, iniciando en el conjunto interfaz – ventana de datos – programa autómata para el sistema guiado.

La programación dentro del autómata para este proceso de calibración se presenta en la Figura 47.

Figura 47. Programa inicial de calibración en el autómata



6.2 SISTEMA GUIADO ACTIVO

Este sistema de programación se basa en el control del robot por medio del uso de un joystick como el ilustrado en la Figura 48.

Figura 48. Joystick sistema guiado activo

Fuente: Banco laboratorio automatización industrial

Los movimientos realizados por el robot son memorizados para luego ser reproducidos de forma automática.

Dentro del sistema de control programado en el autómata del robot se cuenta con tres estados de trabajo para este lenguaje de programación.

***Movimiento libre**:* En este modo de trabajo el usuario puede mover libremente el robot sin que los desplazamientos o acciones sean memorizados.

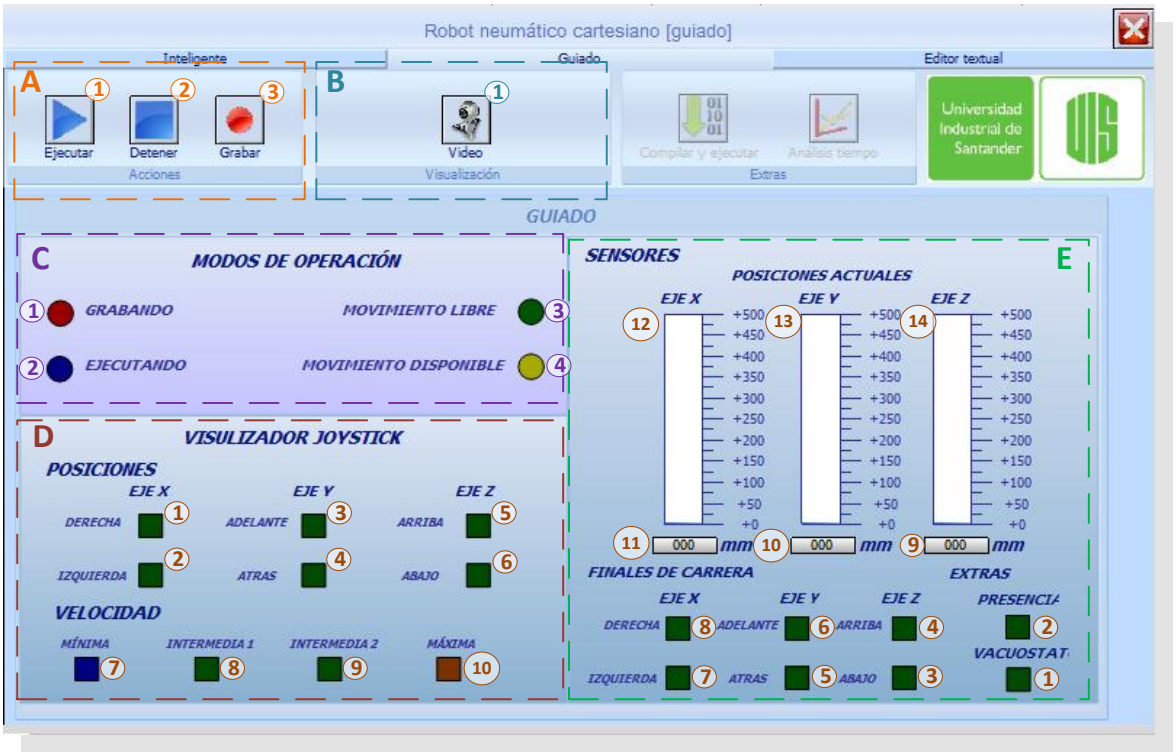
***Estado de grabación**:* Cuando el sistema se encuentra en esta etapa, todos los movimientos y acciones que el robot realice por las instrucciones dadas por el usuario son memorizadas.

***Reproducción de los movimientos almacenados**:* En este modo de trabajo el robot realiza todas las acciones almacenadas previamente en el estado de grabación.

- **Interfaz para el lenguaje de programación**

La estructura de la interfaz se muestra en la Figura 49.

Figura 49. Estructura interfaz del lenguaje guiado activo



Fuente: Banco laboratorio automatización industrial

Los elementos ubicados en la interfaz se dividen en cinco secciones, las cuales se describen a continuación:

❖ ***Sección A: Acciones*** Son aquellos eventos que ejecutan las tareas principales del lenguaje guiado activo, a este grupo pertenecen.

○ ***Elemento número 1:*** *Botón Ejecutar*, cuando este se pulsa, si la variable GRABACION_HMI se encuentra desactivada y MOVIMIENTO_LISTO_HMI se encuentra activa (Lo que quiere decir que los datos almacenados son los suficientes para ser ejecutados), como resultado se desactiva MOVIMIENTO_LIBRE_HMI y se activa EJECUCIONMOVIMIENTO_HMI, si no se

cumple las condiciones la interfaz muestra al usuario un mensaje que dicta *“No se puede ejecutar, debido a que no existe la cantidad suficiente de datos”*.

La condición estipulada en el autómata programable establece que si la variable EJECUCIONMOVIMIENTO_HMI (para el autómata es EJECUCIONMOVIMIENTO) se encuentra activa y tanto MOVIMIENTO_LIBRE_HMI (para el autómata es MOVIMIENTO_LIBRE) como GRABACION_HMI (para el autómata es GRABACION) se encuentran desactivadas el sistema pasa al modo “*Reproducción de los movimientos almacenados”*, ocasionando que el robot comience a ejecutar las acciones previamente almacenadas.

- ***Elemento número 2****: Botón Detener* Al pulsarlo se activa la variable modopausa_HMI, la cual dentro del autómata representa a la variable modopausa, encargada de detener por completo la ejecución y el almacenamiento de los movimientos, dejando al sistema en el modo movimiento libre.

- ***Elemento número 3****: Botón Grabar* Si al pulsarlo la variable EJECUCIONMOVIMIENTO_HMI no se encuentra activa (Lo que indica que no se está reproduciendo un movimiento almacenado con anterioridad) como resultado se activa la variable GRABACION_HMI (GRABACION en el autómata programable) y se desactiva MOVIMIENTO_LIBRE_HMI (MOVIMIENTO_LIBRE en el autómata programable), en caso que no se cumpla la condición el usuario recibe un mensaje que dicta *“No se puede realizar el proceso de grabación ya que se está ejecutando el movimiento grabado con anterioridad”*.

❖ ***SECCIÓN B: VISUALIZACIÓN***

○ ***Elemento número 1****: Botón Video* Cuando este se pulsa, se abre en la interfaz el programa que permite una vigilancia remota en tiempo real del estado del robot.

Figura 50. Visualización remota robot en el sistema guiado

Fuente: Banco laboratorio automatización industrial

❖ ***SECCIÓN C: MODOS DE OPERACIÓN*** Consiste en cuatro elementos de visualización.

○ ***Elemento número 1****: Led Grabación* Este led refleja el estado de la variable GRABACION_HMI, cuando se encuentra activa este parpadea (debido a que el sistema se encuentra en el modo grabación).

- ***Elemento número 2**: Led Ejecutando* Este led refleja el estado de la variable EJECUCIONMOVIMIENTO_HMI, cuando se encuentra activa este parpadea (debido a que el sistema se encuentra en el proceso de ejecución de los movimientos memorizados).

- ***Elemento número 3**: Led Movimiento libre* Este led refleja el estado de la variable MOVIMIENTO_LIBRE_HMI, cuando se encuentra activa este parpadea (debido a que el sistema se encuentra en el modo movimiento libre).

- ***Elemento número 4**: Led Movimiento disponible* Este led refleja el estado de la variable MOVIMIENTO_LISTO_HMI, cuando se encuentra activa este parpadea (lo cual indica que la cantidad de datos almacenada es la mínima suficiente para recrear un movimiento).

❖ **SECCIÓN D: VISUALIZADOR DEL JOYSCTICK** Refleja el estado de los pulsadores ubicados dentro del joystick, cuando el botón es pulsado en el mando principal el led virtual asociado a la variable se activa, en caso contrario se desactiva.

- ***Elemento número 1**: Led asociado al pulsador que indica hacia la derecha en el joystick* cuyo resultado se almacena en la variable conXDER_HMI (dentro del autómata se reconoce como conXDER, por medio de la misma el estado del botón es transmitido al programa de control ubicado dentro del autómata programable).

- ***Elemento número 2**: Led asociado al pulsador que indica hacia la izquierda en el joystick* cuyo resultado se almacena en la variable conXIZQ_HMI (dentro del autómata se reconoce como conXIZQ, por medio de la misma el estado del botón es transmitido al programa de control ubicado dentro del autómata programable).

- ***Elemento número 3****: Led asociado al pulsador que indica hacia adelante en el joystick* cuyo resultado se almacena en la variable conYADE_HMI (dentro del autómata se reconoce como conYADE, por medio de la misma el estado del botón es transmitido al programa de control ubicado dentro del autómata programable).

- ***Elemento número 4:*** *Led asociado al pulsador que indica hacia atrás en el joystick*

*C*uyo resultado se almacena en la variable conYATR_HMI (dentro del autómata se reconoce como conYATR, por medio de la misma el estado del botón es transmitido al programa de control ubicado dentro del autómata programable).

- ***Elemento número 5****: Led asociado al pulsador que indica hacia arriba en el joystick* cuyo resultado se almacena en la variable conZARR_HMI (dentro del autómata se reconoce como conZARR, por medio de la misma el estado del botón es transmitido al programa de control ubicado dentro del autómata programable).

- ***Elemento número 6****: Led asociado al pulsador que indica hacia arriba en el joystick* cuyo resultado se almacena en la variable conZABJ_HMI (dentro del autómata se reconoce como conZABJ, por medio de la misma el estado del botón es transmitido al programa de control ubicado dentro del autómata programable).

- ***Elemento número 7****: Led asociado al pulsador que indica la velocidad mínima en el joystick* cuyo resultado se almacena en la variable conVmin_HMI (dentro del autómata se reconoce como conVmin, por medio de la

misma el estado del botón es transmitido al programa de control ubicado dentro del autómata programable).

- ***Elemento número 8****: Led asociado al pulsador que indica la velocidad intermedia 1 en el joystick* cuyo resultado se almacena en la variable conV1_HMI (dentro del autómata se reconoce como conV1, por medio de la misma el estado del botón es transmitido al programa de control ubicado dentro del autómata programable).

- ***Elemento número 9****: Led asociado al pulsador que indica la velocidad intermedia 2 en el joystick* cuyo resultado se almacena en la variable conV2_HMI (dentro del autómata se reconoce como conV2, por medio de la misma el estado del botón es transmitido al programa de control ubicado dentro del autómata programable).

- ***Elemento número 10****: Led asociado al pulsador que indica la velocidad máxima en el joystick* cuyo resultado se almacena en la variable conVMax_HMI (dentro del autómata se reconoce como conVMax, por medio de la misma el estado del botón es transmitido al programa de control ubicado dentro del autómata programable).

- ***SECCIÓN E: SENSORES DEL ROBOT***

- ***Elemento número 1****: SENSOR DE PRESENCIA*

Indica el estado del sensor de presencia, cuando se encuentra activo el led cambia su color con el fin de mostrar el estado actual. El valor se almacena en SENSORPRESENCIA_HMI (dentro del autómata se reconoce como sensorpresensia, por medio de esta variable se transmite el valor al programa de control ubicado dentro del autómata programable).

- ***Elemento número 2**: VACUOSTATO*

Indica el estado del vacuostato, cuando se encuentra activo el led cambia su color con el fin de mostrar el estado actual. El valor se almacena en VACUOSTATO_HMI (dentro del autómata se reconoce como vacuostato, por medio de esta variable se transmite el valor al programa de control ubicado dentro del autómata programable).

- ***Los elementos del 3 al 8**, representan los estados de los finales de carrera, donde las direcciones son: 3, Abajo eje Z; 4, Arriba eje Z; 5, Atrás eje Y; 6, Adelante eje Y; 7, Izquierda eje X; 8, Derecha eje X. Las variables en la cuales se almacena el valor del estado y la relación con el autómata se presenta a continuación, el orden representa el descrito al inicio de este párrafo: Z_ABAJO_HMI (z_abajo), Z_ARRIBA_HMI (z_arriba), Y_ATRAS_HMI (y_atras), Y_ADELANTE_HMI (y_adelante), X_IZQUIERDA_HMI (x_izquierda), X_DERECHA (x_derecha).*

- ***Los elementos del 9 al 11**, representan el valor de la posición para los diferentes ejes ubicados como números enteros. **Los elementos del 12 al 14** representan la posición solo que en este caso se hace por medio de una barra.*

La posición que representa cada elemento, la variable en la que se almacena el valor en la interfaz y el autómata programable se describe a continuación:

Posición en el eje X, elementos 11 y 12, variable en interfaz POTENCIOMETROX_HMI, en el autómata es POTENCIOMETROX.

Posición en el eje Y, elementos 10 y 13, variable en interfaz POTENCIOMETROY_HMI, en el autómata es POTENCIOMETROY.

Posición en el eje Z, elementos 9 y 14, variable en interfaz POTENCIOMETROZ_HMI, en el autómata es POTENCIOMETROZ.

- **Ventana de datos**

La interfaz ofrecida al usuario para este sistema comparte con el autómata las siguientes variables:

Tabla 1. Lista de variables modo guiado 1

Variable en la interfaz	Variable en el autómata programable	Descripción
INTELIGENTE_HMI	INTELIGENTE	Variable booleana que al encontrarse activa indica que se encuentra en el lenguaje de programación textual nivel tarea
GUIADO_HMI	GUIADO	Variable booleana que al encontrarse activa indica que se encuentra en el lenguaje de programación guiado activo
TEXTUAL_HMI	TEXTUAL	Variable booleana que al encontrarse activa indica que se encuentra en el lenguaje de programación textual nivel robot
GRABACION_HMI	GRABACION	Variable booleana que al estar activa introduce el modo de trabajo al estado de grabación
MOVIMIENTOLISTO_HMI	MOVIMIENTOLISTO	Variable booleana que al activarse indica que existen la cantidad de acciones suficientes para ser reproducidas luego de forma automática
MOVIMIENTO_LIBRE_HMI	MOVIMIENTO_LIBRE	Variable booleana que mientras permanezca activa asegura que el sistema se encuentre en el modo de trabajo de movimiento libre
EJECUCIONMOVIMIENTO_HMI	EJECUCIONMOVIMIENTO	Variable booleana que mientras permanezca activa asegura que el sistema se encuentra en el modo de trabajo de reproducción de los movimientos almacenados
POTENCIOMETROX_HMI	POTENCIOMETROX	Número real que establece el valor de la posición en el eje X
POTENCIOMETROY_HMI	POTENCIOMETRY	Número real que establece el valor de la posición en el eje Y
POTENCIOMETROZ_HMI	POTENCIOMETROZ	Número real que establece el valor de la posición en el eje Z

Tabla 2. Lista de variables modo guiado 2

Variable en la interfaz	Variable en el autómata programable	Descripción
conZARR_HMI	conZARR	Indica el estado del pulsador en la dirección hacia arriba en el joystick, este dato es transmitido por medio de la misma al autómata programable, él cual tiene programados los eventos según el valor de esta variable
conZABJ_HMI	conZABJ	Indica el estado del pulsador en la dirección hacia abajo en el joystick, este dato es transmitido por medio de la misma al autómata programable, él cual tiene programados los eventos según el valor de esta variable
conYADE_HMI	conYADE	Indica el estado del pulsador en la dirección hacia adelante en el joystick, este dato es transmitido por medio de la misma al autómata programable, él cual tiene programados los eventos según el valor de esta variable
conYATR_HMI	conYATR	Indica el estado del pulsador en la dirección hacia atrás en el joystick, este dato es transmitido por medio de la misma al autómata programable, él cual tiene programados los eventos según el valor de esta variable
conXDER_HMI	conXDER	Indica el estado del pulsador en la dirección hacia la derecha en el joystick, este dato es transmitido por medio de la misma al autómata programable, él cual tiene programados los eventos según el valor de esta variable
conXIZQ_HMI	conXIZQ	Indica el estado del pulsador en la dirección hacia la izquierda en el joystick, este dato es transmitido por medio de la misma al autómata programable, él cual tiene programados los eventos según el valor de esta variable

Tabla 3. Lista de variables modo guiado 3

Variable en la interfaz	Variable en el autómata programable	Descripción
conVMin_HMI	conVMin	Indica el estado del pulsador para la velocidad mínima en el joystick, este dato es transmitido por medio de la misma al autómata programable, él cual tiene programados los eventos según el valor de esta variable
conV1_HMI	conV1	Indica el estado del pulsador para la velocidad intermedia 1 en el joystick, este dato es transmitido por medio de la misma al autómata programable, él cual tiene programados los eventos según el valor de esta variable
conV2_HMI	conV2	Indica el estado del pulsador para la velocidad intermedia 2 en el joystick, este dato es transmitido por medio de la misma al autómata programable, él cual tiene programados los eventos según el valor de esta variable
conVMax_HMI	conVMax_HMI	Indica el estado del pulsador para la velocidad máxima en el joystick, este dato es transmitido por medio de la misma al autómata programable, él cual tiene programados los eventos según el valor de esta variable
conSUCCION_HMI	conSUCCION	Indica el estado del pulsador para la acción de succión en el joystick, este dato es transmitido por medio de la misma al autómata programable, él cual tiene programados los eventos según el valor de esta variable

Tabla 4. Lista de variables modo guiado 4

Variable en la interfaz	Variable en el autómata programable	Descripción
X_DERECHA_HMI	x_derecha	Indica el estado del final de carrera para la posición derecha en el eje X en el joystick, este dato es transmitido por medio de la misma al autómata programable, él cual tiene programados los eventos según el valor de esta variable
X_IZQUIERDA_HMI	x_izquierda	Indica el estado del final de carrera para la posición izquierda en el eje X en el joystick, este dato es transmitido por medio de la misma al autómata programable, él cual tiene programados los eventos según el valor de esta variable
Y_ADELANTE_HMI	y_adelante	Indica el estado del final de carrera para la posición adelante en el eje Y en el joystick, este dato es transmitido por medio de la misma al autómata programable, él cual tiene programados los eventos según el valor de esta variable
Y_ATRAS_HMI	y_atrás	Indica el estado del final de carrera para la posición atrás en el eje Y en el joystick, este dato es transmitido por medio de la misma al autómata programable, él cual tiene programados los eventos según el valor de esta variable
Z_ARRIBA_HMI	z_arriba	Indica el estado del final de carrera para la posición arriba en el eje Z en el joystick, este dato es transmitido por medio de la misma al autómata programable, él cual tiene programados los eventos según el valor de esta variable
Z_ABAJO_HMI	z_abajo	Indica el estado del final de carrera para la posición abajo en el eje Z en el joystick, este dato es transmitido por medio de la misma al autómata programable, él cual tiene programados los eventos según el valor de esta variable

Tabla 5. Lista de variables modo guiado 5

Variable en la interfaz	Variable en el autómata programable	Descripción
SENSORPRESENCIA_HMI	sensorpresensencia	Indica el estado del sensor de presencia, este dato es transmitido por medio de la misma al autómata programable, él cual tiene programados los eventos según el valor de esta variable
VACUOSTATO_HMI	vacuostato	Indica el estado del vacuostato, este dato es transmitido por medio de la misma al autómata programable, él cual tiene programados los eventos según el valor de esta variable

- **Programa en el autómata**

La forma como trabaja el sistema interno de programación en el autómata para los diferentes modos de trabajo se especifica en las Figuras 51, 52, 53, 54.

Figura 51. Sistema guiado activo: programa en el autómata 1

Figura 52. Sistema guiado activo: programa en el autómata 2

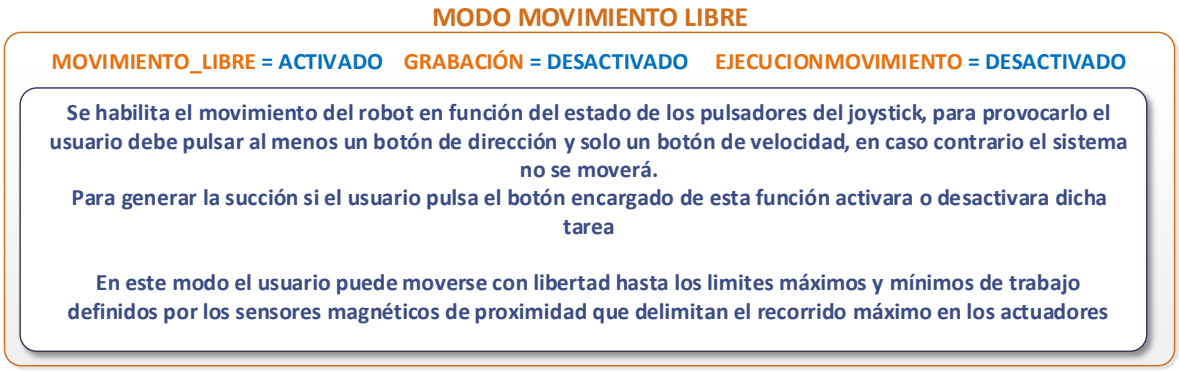


Figura 53. Sistema guiado activo: programa en el autómata 3

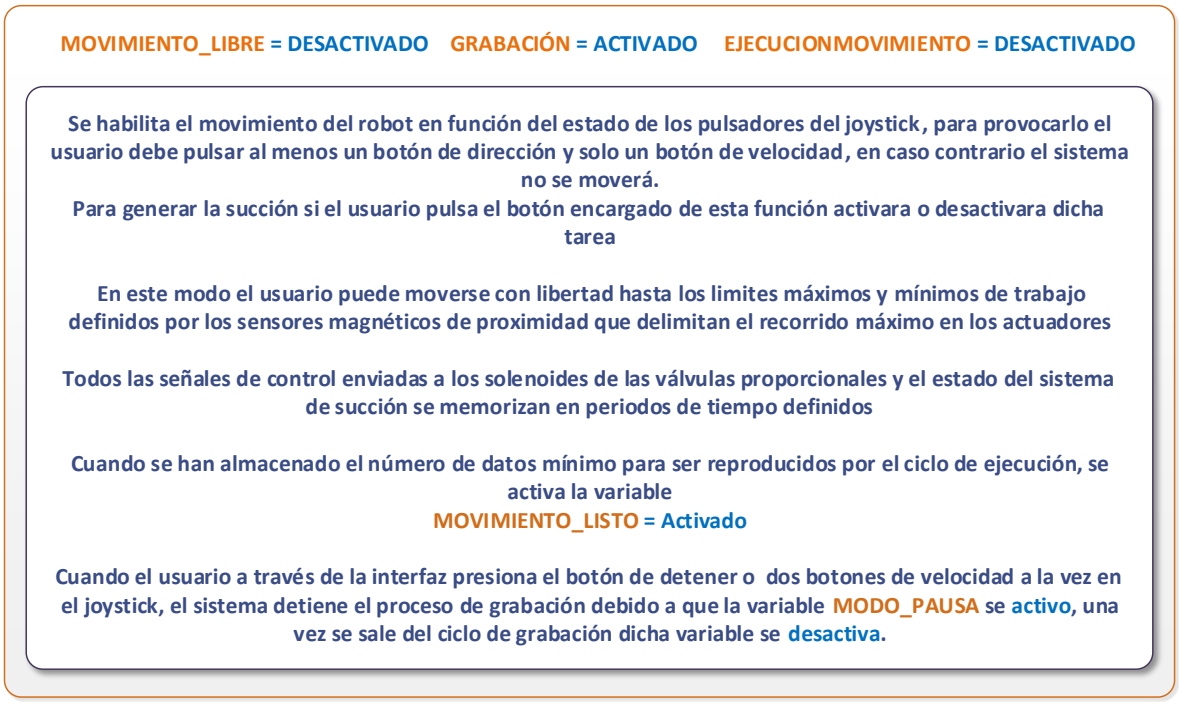
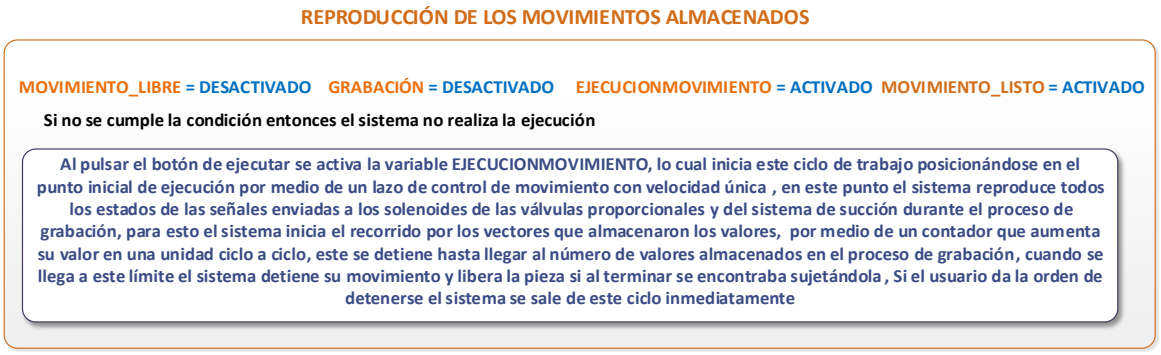


Figura 54. Sistema guiado activo: programa en el autómata 4

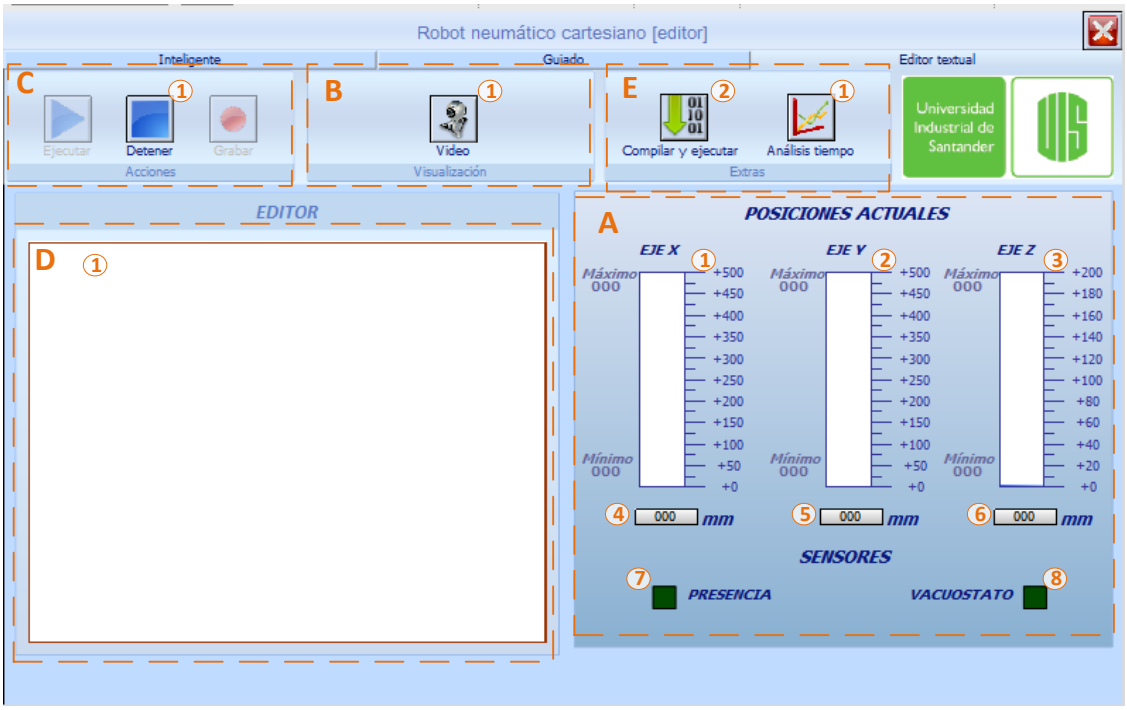


6.3 LENGUAJE TEXTUAL NIVEL ROBOT

En el lenguaje textual nivel robot, el usuario le especifica la secuencia de acciones que debe realizar la máquina para cumplir con la tarea deseada.

- **Interfaz para el lenguaje**

Figura 55. Estructura lenguaje textual nivel robot



Fuente: Banco de trabajo, Laboratorio de Automatización Industrial.

La interfaz para el lenguaje textual nivel robot se divide en 5 secciones, la cuales se describen a continuación:

- ***SECCIÓN A: SENSORES DEL SISTEMA***

Esta sección de la interfaz se encuentra dedicada a ilustrar al usuario el estado de los sensores presentes en el robot cartesiano. Los elementos que la componen son:

- ***Elemento número 1****: Barra de posición Eje X* Esta barra muestra la posición en milímetros en el eje X a la cual se encuentra el efector final en función del punto de origen de coordenadas, acompañada de los indicadores que ilustran los límites máximo y mínimo de trabajo.

- ***Elemento número 2:*** *Barra de posición Eje Y* Realiza la misma función que el elemento número 1 solo que en este caso se enfoca para el eje Y del sistema robótico.

- ***Elemento número 3:*** *Barra de posición Eje Z* Realiza la misma función que el elemento número 1 solo que en este caso se enfoca para el eje Z del sistema robótico.

- ***Elemento número 4:*** *Valor de posición X* Ilustra al usuario de la posición en milímetros del efector final en el eje X por medio de un número entero.

- ***Elemento número 5:*** *Valor de posición Y* Ilustra al usuario de la posición en milímetros del efector final en el eje Y por medio de un número entero.

➢ ***Elemento número 6:*** *Valor de posición Z* Ilustra al usuario de la posición en milímetros del efector final en el eje Z por medio de un número entero.

➢ ***Elemento número 7:*** *Indicador sensor de presencia* Muestra al usuario el estado del sensor de presencia, si este se encuentra activo debido a la cercanía de un elemento el led virtual se activa e inicia a parpadear.

➢ ***Elemento número 8:*** *Indicador vacuostato* Muestra al usuario el estado del vacuostato (sensor que indica si el efector final ha sujetado un elemento), si se activa el led virtual parpadea.

❖ ***SECCIÓN B: MONITOREO EXTERNO***

➢ ***Elemento número 1:*** *Video* Al pulsar este botón el sistema activa el programa que permite visualizar en tiempo real el estado del robot desde la interfaz.

Figura 56. Monitoreo externo editor textual nivel robot

Fuente: Banco de trabajo, Laboratorio de Automatización Industrial.

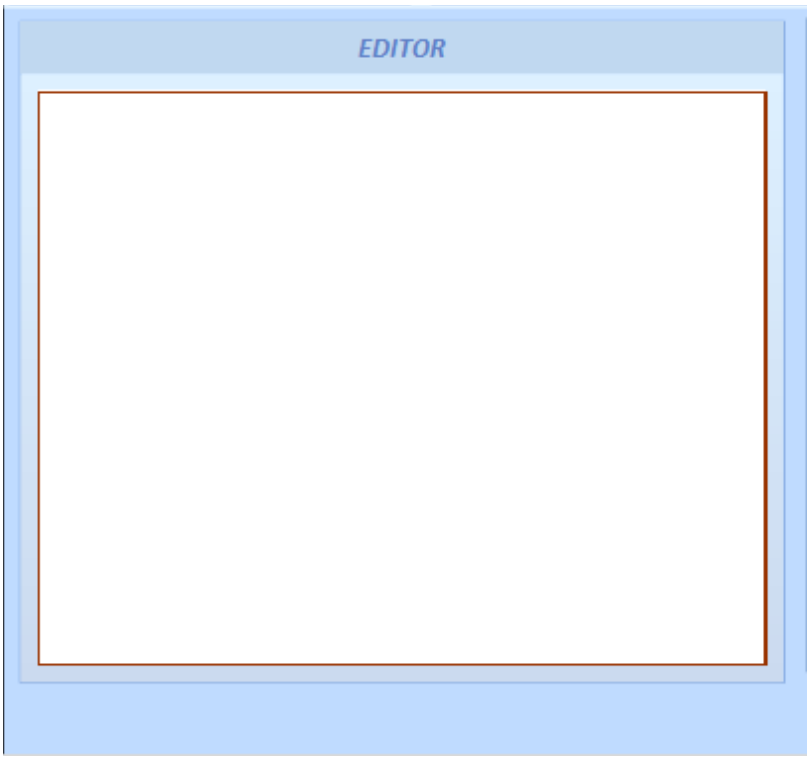
❖ ***SECCIÓN C: OPERACIONES EXTERNAS***

➢ ***Elemento número 1:*** *Botón Detener* Al pulsar este botón se detiene el proceso propio del lenguaje textual nivel robot, si este se está ejecutando.

❖ ***SECCIÓN D: EDITOR TEXTUAL***

➢ ***Elemento número 1:*** *EDITOR* En esta zona el usuario ingresa el código a ejecutar.

Figura 57. Editor textual para el lenguaje nivel robot



Fuente: Banco de trabajo, Laboratorio de Automatización Industrial.

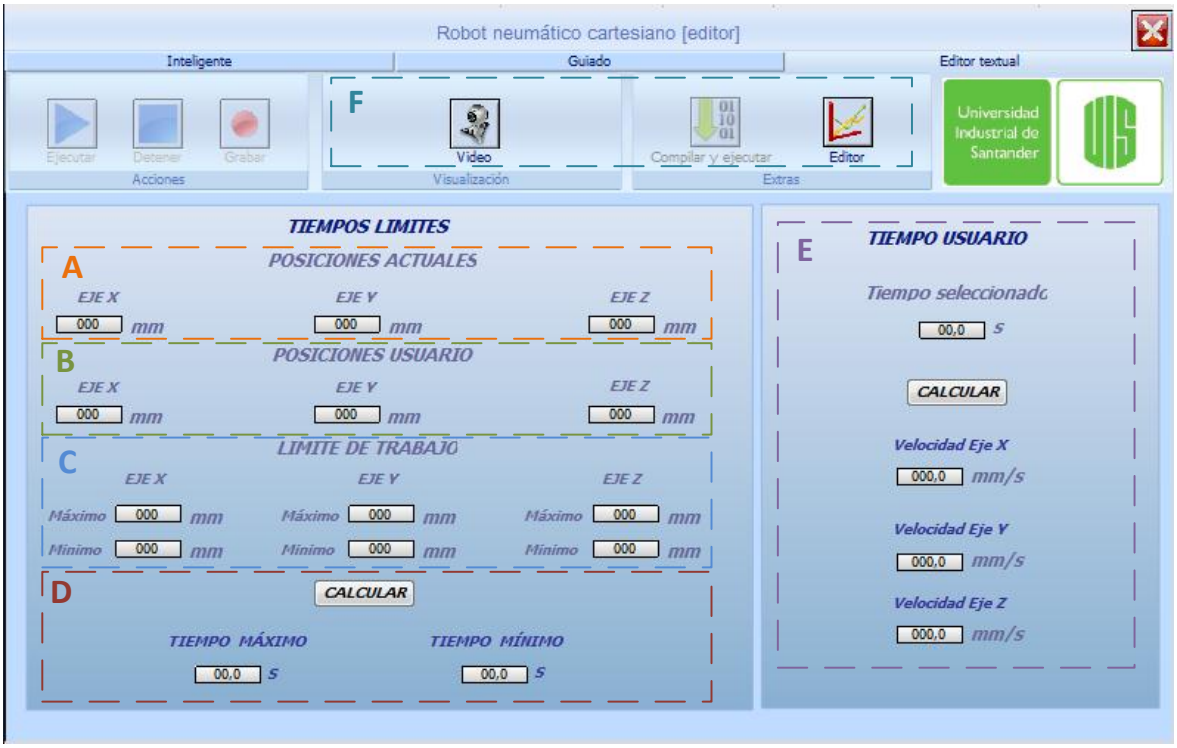
❖ ***SECCIÓN E: FUNCIONES DEL LENGUAJE***

➢ ***Elemento número 1****: SISTEMA DE ANÁLISIS* Al pulsar este botón emerge la ventana para que el usuario pueda realizar los cálculos pertinentes a los movimientos del robot.

La interfaz presentada se muestra en la Figura 58, el objetivo de la misma es permitir al usuario el cálculo del tiempo mínimo y máximo según las posiciones del punto final e inicial de la trayectoria a recorrer. Lo anterior es importante ya que los parámetros para la instrucción de movimiento dentro del lenguaje textual nivel robot son la posición final del movimiento y el tiempo de trabajo.

Además de los límites de trabajo en el tiempo, el operador puede calcular las velocidades promedio que alcanza el sistema en los diferentes ejes, lo cual le permite tener una percepción de la precisión del movimiento, pues a valores más cercanos a la velocidad mínima se hace más óptima.

Figura 58. Interfaz para los cálculos



Fuente: Banco de trabajo, Laboratorio de Automatización Industrial.

La estructura de la interfaz se describe a continuación:

- **SECCIÓN A:**

En esta parte el usuario introduce los valores de las coordenadas para el punto inicial del movimiento.

- **SECCIÓN B:**

Se introducen los valores para las coordenadas del punto final del movimiento.

- **SECCIÓN C:**

Se muestra al usuario las coordenadas mínimas y máximas de trabajo para cada eje.

- **SECCIÓN D:**

En ella se muestra el valor del tiempo mínimo y máximo, calculado como resultado de pulsar el botón CALCULAR, ubicado en esta zona.

- **SECCIÓN E:**

Al pulsar el botón CALCULAR, se muestran al usuario los valores de las velocidades promedios que alcanza el sistema como resultado del tiempo de trabajo introducido por él, y la separación del punto final e inicial.

- **SECCIÓN F:**

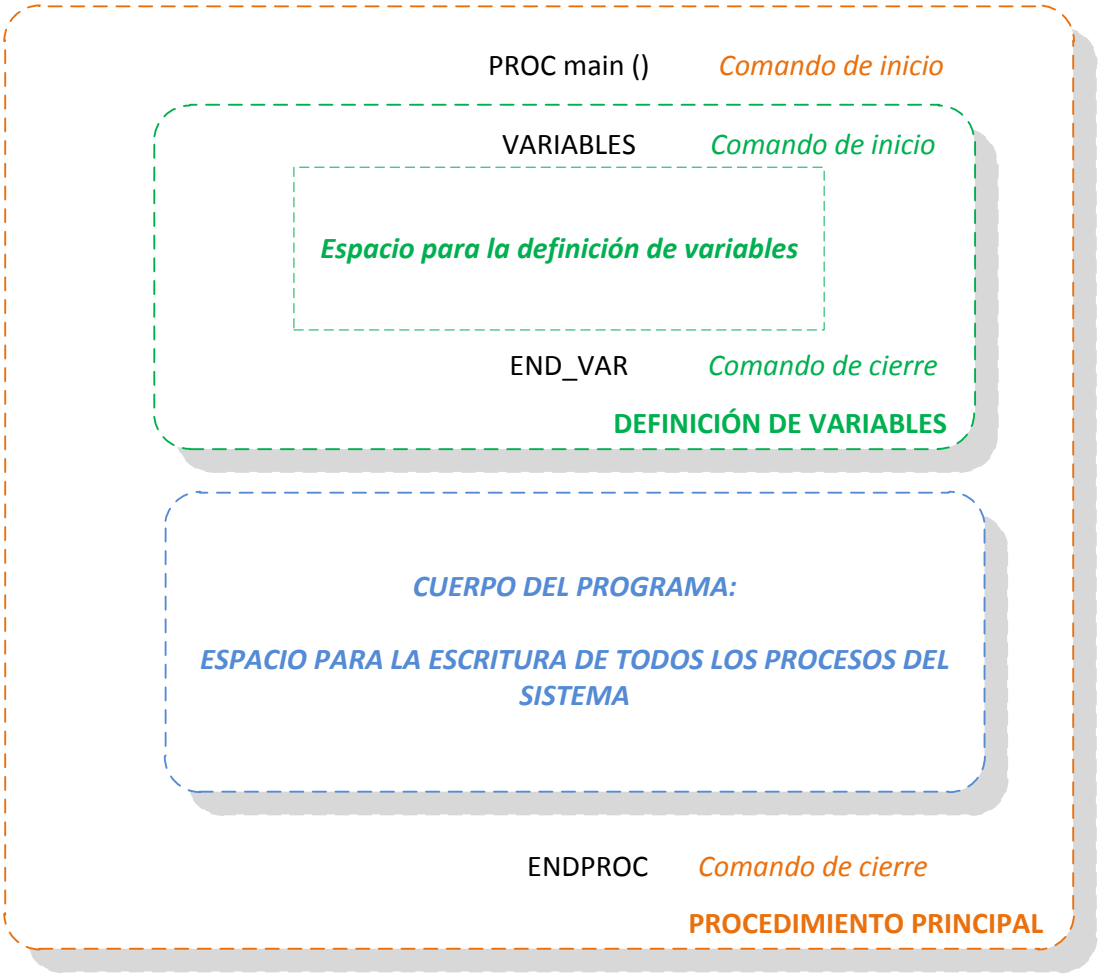
En esta zona se muestran dos elementos de trabajo, el primero es el botón video, el cual al pulsarlo se inicia el programa que permite la vigilancia remota en tiempo real del robot neumático cartesiano, el segundo es el botón EDITOR, el cual al pulsarlo vuelve el sistema a la interfaz principal de trabajo para el lenguaje textual nivel robot.

- ***Elemento número 2:*** *COMPILAR Y EJECUTAR* Al pulsar este botón se ejecuta la función principal (script de Visual Basic) encargada del proceso principal del lenguaje textual nivel robot.

Dentro de la función principal se encuentra todo el proceso del lenguaje, esta se realiza en un script de Visual Basic y dentro de ella se llaman a otras funciones.

Antes de describir a la función es importante resaltar la estructura de un programa escrito en el lenguaje textual nivel robot para este proyecto, esto se presenta en la Figura 59.

Figura 59. Estructura código escrito en el lenguaje textual nivel robot



Como se observa en la Figura 59. Un programa realizado en el lenguaje textual nivel robot se basa en el uso de un *procedimiento principal*, el cual se ejecuta por una sola vez y su activación se presenta cuando el usuario pulsa el botón de "*compilar y ejecutar*" en la interfaz de trabajo. Para declararlo correctamente en la zona del "editor textual" (***SECCIÓN D: EDITOR TEXTUAL***) se debe iniciar con el comando "PROC main ()" y se debe cerrar con la palabra "ENDPROC", cualquier proceso ubicado fuera del espacio entre los dos elementos indicará un error al usuario ya que solo se puede declarar un solo procedimiento.

El procedimiento principal se divide en dos regiones importantes, la primera de ellas, es la declaración de variables, en este espacio el usuario define todos los elementos que va a usar dentro del procedimiento principal, valor que no se encuentre declarado no se reconoce cuando se ejecuta el programa.

Para iniciar esta zona se usa el comando “VARIABLES” y se debe cerrar con la palabra “END_VAR”.

Dentro de las variables que se pueden encontrar tenemos:

Variables numéricas: Almacenan números reales, pueden ser modificadas como proceso del programa.

Variables booleanas: Almacenan datos del tipo lógico (dos estados, verdadero o falso), pueden ser modificadas como proceso del programa.

Variables string: Almacenan cadenas de caracteres con una longitud máxima de 100, pueden ser modificadas como proceso del programa.

Constantes numéricas: Almacenan números reales, solo pueden ser definidas en la zona de variables.

Constantes booleanas: Almacenan datos del tipo lógico (dos estados, verdadero o falso), solo pueden ser definidas en la zona de variables.

Constantes string: Almacenan cadenas de caracteres con una longitud máxima de 100, solo pueden ser definidas en la zona de variables.

Variables de posición: Almacenan el conjunto de tres coordenadas, valores enteros que indican la posición en milímetros de un punto en el espacio del

sistema robótico, sus valores pueden ser modificados dentro del programa fuera de la zona de variables.

Matrices unidimensionales y bidimensionales: Almacenan números reales en sus espacios, una misma variable puede tener hasta 99 espacios por dimensión, los valores almacenados pueden ser modificados fuera de la zona de variables.

La segunda zona del procedimiento principal ubicada entre los comandos "END_VAR" y "ENDPROC" es el cuerpo del programa, aquí el usuario ingresa todos los procesos a realizar.

Los procesos se pueden dividir en tres grupos: **asignaciones**, **instrucciones** y **ciclos de control**.

Una **asignación** consiste en un proceso que modifica el valor interno de una variable.

Las **asignaciones** se caracterizan porque su estructura se basa en el nombre de la variable a la cual se desea cambiar su valor, seguido del elemento ":=", continuando con el proceso que se le desea asignar y finalizando con el carácter punto y coma (;).

Una **instrucción** es una orden la cual determina una actividad que el programa o el robot deben realizar, dentro de las instrucciones presentadas para el lenguaje textual nivel robot se encuentran:

MENSAJE, este comando es seguido de una variable o constante tipo string, como resultado envía un cuadro de mensaje con el contenido almacenado en el elemento de naturaleza string.

MOVER, los parámetros de entrada para esta instrucción son: una variable de posición y una variable, constante numérica o número real positivo que indique el tiempo de ejecución del proceso, si el análisis dentro de la función programada dentro del robot indica que el movimiento es posible, el sistema lleva al efector final a la posición indicada, en caso contrario la interfaz presenta al usuario un mensaje de error.

SUCCIÓN, esta instrucción no tiene parámetros de entrada, cuando se encuentra en el programa se da la orden al robot de activar el proceso de succión.

SOLTAR, esta instrucción no tiene parámetros de entrada, cuando se encuentra en el programa se da la orden al robot de desactivar el proceso de succión.

Un ciclo de control es un elemento que realiza una serie de instrucciones bajo ciertas condiciones programadas por el usuario, entre los que se usan en el programa se presentan:

Ciclo If, acompañado por una sentencia del tipo lógico, si la evaluación de la misma da un resultado afirmativo, el sistema realiza las instrucciones ubicadas dentro del mismo, en caso contrario continua con las instrucciones ubicadas luego del elemento de cierre o del elemento que indica el camino opcional en este caso se usa el comando Else para indicar dicha separación.

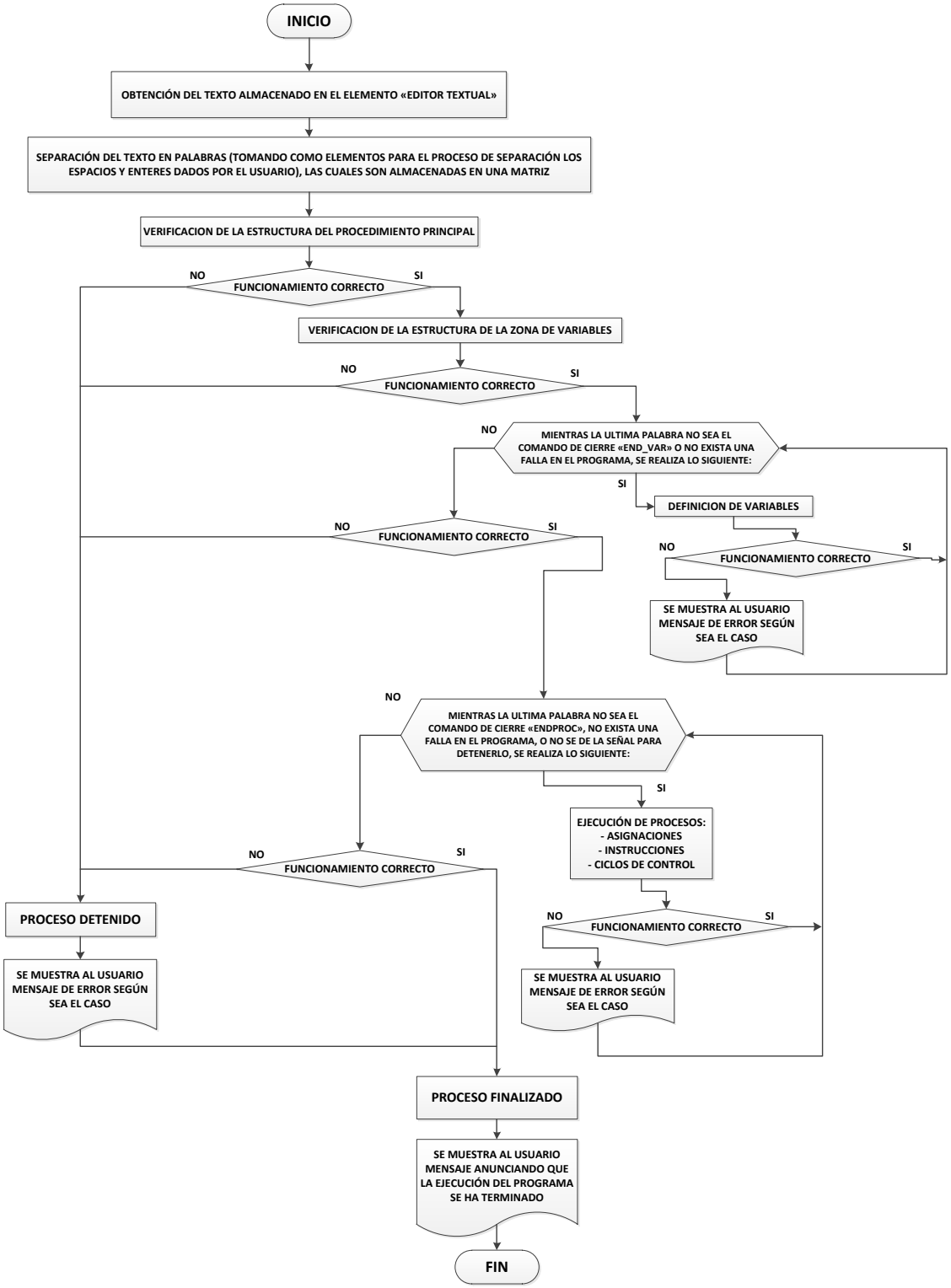
Ciclo While, acompañado por una sentencia del tipo lógico, mientras la evaluación de la misma de afirmativo realizará todas las acciones ubicadas entre su inicio y el elemento de cierre Endwhile, en caso de que la evaluación de un resultado negativo continuará con los procesos que siguen después del comando EndWhile.

Ciclo For, acompañado de tres elementos cuya naturaleza representa una cantidad numérica, los cuales indican un valor inicial, un valor final y un paso de aumento o disminución, se realizan todas las acciones ubicadas entre el inicio dado por el elemento FOR y su comando para el cierre ENDFOR, siempre y cuando la suma almacenada ciclo a ciclo, la cual arranca con el valor inicial y poco a poco se le adiciona el valor del paso, no supera al valor final (si la suma va en ascenso) o no es menor a este (si la suma va en descenso).

Una vez resumidas las características con las cuales se define un programa en el lenguaje textual nivel robot del proyecto, se puede continuar con el proceso de la forma como la función principal desarrollada en un script de Visual Basic permite la ejecución del código escrito por el usuario en la zona del "*editor textual*".

El proceso general de la función principal se describe de manera global en la Figura 60.

Figura 60. Proceso general de la función principal



- ***Primer proceso y segundo proceso:*** *Obtención del texto almacenado en el elemento "Editor Textual" / Separación del texto en palabras siendo almacenadas en una matriz.*

Para este proceso el sistema toma todo el texto que se encuentra en el Editor Textual (SECCIÓN D: EDITOR TEXTUAL, en la interfaz del lenguaje textual nivel robot), y lo separa en datos, tomando como elementos de separación los espacios introducidos por la barra espaciadora y el botón enter, los valores obtenidos son almacenados en una matriz, cuyo espacio máximo de trabajo son 2'000.000 de elementos, al terminar la separación obtiene el valor de datos totales que se guardaron.

- ***Tercer Proceso:*** *Verificación de la estructura del procedimiento principal*.

El sistema verifica que los primeros datos almacenados sean "PROC", "main" y "()", los cuales definen el inicio del procedimiento principal, de no ser así se envía un mensaje al usuario especificando que el principio del procedimiento no se encuentra correctamente definido y se detiene el proceso de ejecución del programa.

Si el inicio se encuentra correctamente definido, el sistema continua revisando el último de los datos almacenados, el cual debe corresponder con el comando "ENDPROC", encargado del cierre del procedimiento principal, de no ser así entonces se envía un mensaje al usuario indicando esta situación y se detiene el proceso de ejecución del programa.

Si el final se define correctamente, continua al siguiente proceso.

- ***Cuarto Proceso:*** *Verificación de la estructura de la zona de variables.*

En este caso el sistema revisa que el siguiente comando después del inicio del procedimiento sea la palabra “VARIABLES” la cual indica que se ha iniciado correctamente la zona de definición de variables, en caso contrario se envía un mensaje al usuario de la ausencia del elemento y finaliza el proceso de ejecución.

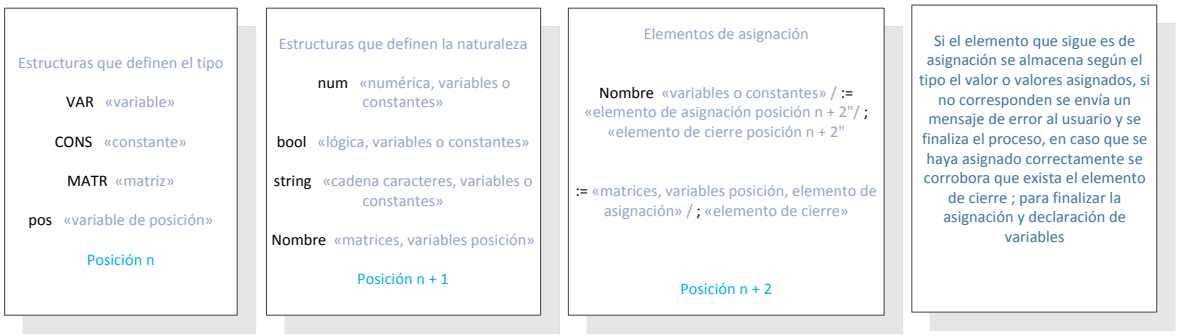
Si no existe falla alguna en la descripción anterior, continua verificando que exista el elemento de cierre “END_VAR” si existe el sistema continua al siguiente proceso en caso contrario envía un mensaje al usuario de la falta del comando y finaliza el proceso de ejecución.

- ***Quinto proceso:*** *Ciclo repetitivo para la definición de variables*.

En este proceso el sistema inicia un ciclo mientras, el cual realiza las funciones que determinan el proceso de definición de variables siempre y cuando no se encuentre el elemento de cierre “END_VAR” o suceda una falla en la definición de variables la cual finaliza el proceso de ejecución.

El proceso de definición de variables se basa en la verificación de la estructura ilustrada en la Figura 61, en caso de no cumplirla se envía un mensaje al usuario del elemento que falta para que se correcta a modo que pueda revisar y corregir su error, el proceso se finaliza para que el código introducido por el operador pueda ser modificado.

Figura 61. Definición de variables



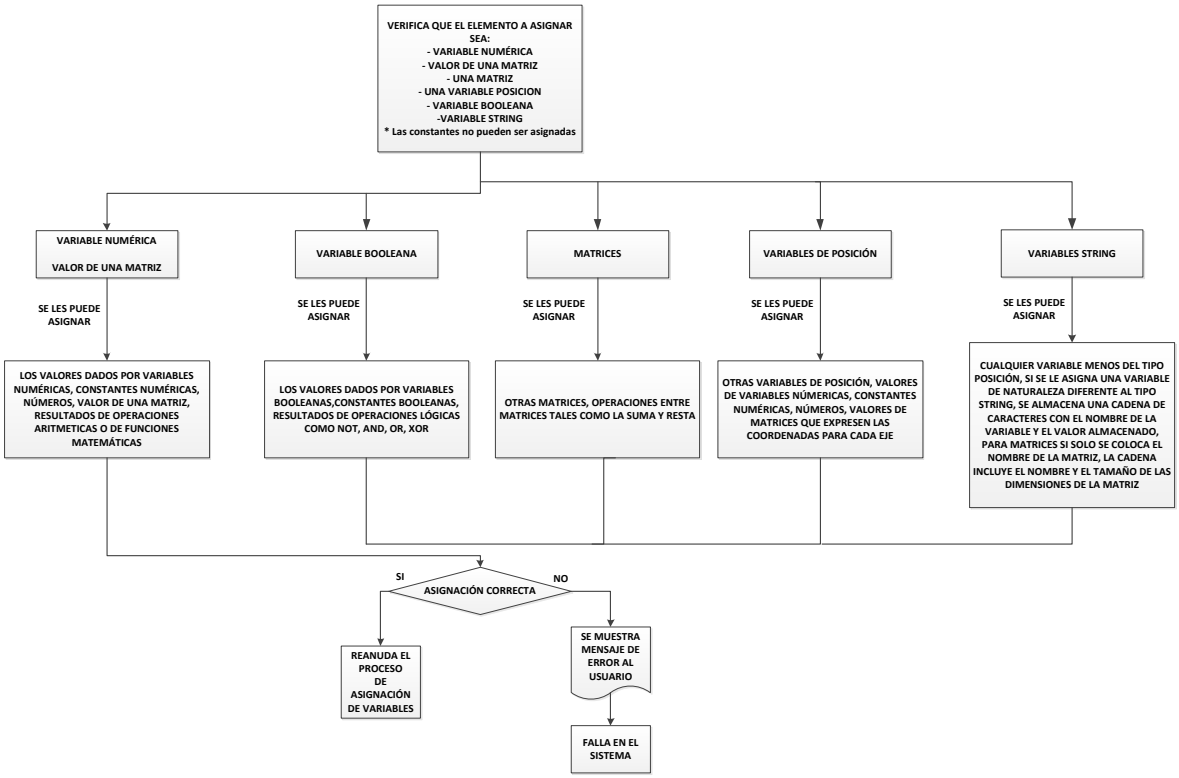
- **Sexto proceso:** ciclo repetitivo para la ejecución de asignaciones, instrucciones y ciclos de control.

En este proceso el sistema verifica en primer lugar si todos los ciclos de control se encuentran correctamente definidos en caso de no ser así, se envía un mensaje al usuario del error y se finaliza el proceso.

Si el resultado de la comprobación anterior es positivo y se puede continuar con el código, cada vez que inicia este proceso, pregunta si el usuario ha activado la variable la cual indica que se debe detener la ejecución, si es así, el proceso se da por finalizado.

Si no existe señal que detenga al programa, el sistema comienza una verificación si el elemento es una asignación, cuya estructura se resumen en la Figura 60.

Figura 62. Estructura de una asignación



Si no es una asignación se verifica que sea una instrucción, existen cuatro clases de instrucciones:

- ***MOVER***, esta instrucción viene acompañada de una variable posición, la cual contiene las coordenadas del punto final, el otro parámetro que se introduce en la instrucción es el tiempo de trabajo, estos datos son enviados al autómata programable del robot el cual determina si el movimiento se puede realizar de ser así se ejecuta el movimiento en caso contrario se envía un mensaje al usuario indicándole de la falla, una vez termine el proceso la instrucción MOVER deja de realizarse y continua el proceso de ejecución del programa.

La estructura de la instrucción MOVER se define así: el comando MOVER, seguido de la VARIABLE DE POSICIÓN, finalizando con el TIEMPO DE TRABAJO.

- ***SUCCIÓN***, esta instrucción no necesita un parámetro externo, cuando se encuentra en el proceso de ejecución el programa activa la salida digital conectada al robot que permite generar la succión en la ventosa de trabajo o efector final.

- ***SOLTAR***, esta instrucción no necesita un parámetro externo, cuando se encuentra en el proceso de ejecución el programa desactiva la salida digital conectada al robot que permite generar la succión en la ventosa de trabajo o efector final.

- ***MENSAJE***, esta instrucción viene acompañada de una variable o constante tipo string, si no es así el sistema envía un error al usuario, la función que cumple es emitir un mensaje con la información contenida en la variable.

Si no es un proceso de asignación o una instrucción se verifica que sea alguno de los ciclos de control, cada uno opera de la siguiente manera:

- ***Ciclo If***, su estructura se basa en el inicio del ciclo con el comando “If”, seguido de una instrucción lógica, luego de la palabra “Then”, si el resultado en la evaluación de la instrucción lógica es verdadero entonces realiza todas las operaciones ubicadas hasta la palabra “Else” o el elemento de cierre “Endif”, en caso de ser lo contrario y existe el comando “Else” dentro de este ciclo, se realizan las instrucciones ubicadas entre él y el elemento de cierre “Endif”.

Para ciclos if anidados, lo que quiere decir que se encuentra uno dentro de otros, se hace uso de un contador el cual al encontrar el comando “If” aumenta en una unidad lo cual le indica el nivel de trabajo en el que se encuentra, es decir que el sistema entiende en que piso de ciclos If va y su objetivo es detenerse hasta encontrar el elemento de cierre del nivel del If en el que va.

- ***Ciclo For***, su estructura se basa en el comando “FOR” seguido de variables del tipo numérico que indican el valor inicial de la cuenta, el valor final, el paso de aumento, su elemento de cierre es la palabra “ENDFOR”.

Para este ciclo el sistema posee un contador el cual almacena el valor de la cuenta producida por la suma entre el valor del ciclo anterior del for y el paso, si el paso es negativo es decir desciende, el sistema for hace que se realicen todos los procesos que se encuentran entre el inicio del ciclo y el elemento de cierre, de forma cíclica hasta que el valor acumulado deje de ser mayor al valor final, en el caso que el paso es positivo realiza la mismas función solo que en este caso se deja de ejecutar hasta que la suma no sea menor o igual al valor final.

Si se tienen ciclos For anidados, se usa un contador para indicar el nivel de ciclos For en el cual va, por tanto el sistema tiene programado que según el piso en el

que se encuentre si el ciclo no se ha terminado no puede continuar y disminuir en uno el contado, para analizar el ciclo más externo.

- ***Ciclo While***, Su estructura se inicia con el comando “While” seguido de una expresión lógica, su elemento de cierre es “Endwhile”, este ciclo realiza todos los procesos ubicados entre la palabra de apertura y cierre mientras la evaluación de la condición lógica sea verdadera.

Si existen ciclos While anidados, el sistema aumenta un contador cada vez que inicia un nuevo ciclo While esto le permite conocer el nivel en el que está operando, la condición programada es que no puede continuar con el ciclo While exterior hasta que la evaluación de la condición lógica del proceso del piso superior no sea falsa.

- ***Séptimo Proceso:*** *Final de la función principal*

Ya sea que se haya presentado un error durante la ejecución o el programa se realizó satisfactoriamente, el sistema envía un mensaje al usuario indicado que llegó al final del programa, de esta manera el operador entiende cuando puede volver a iniciar un nuevo programa.

- **Ventana de datos**

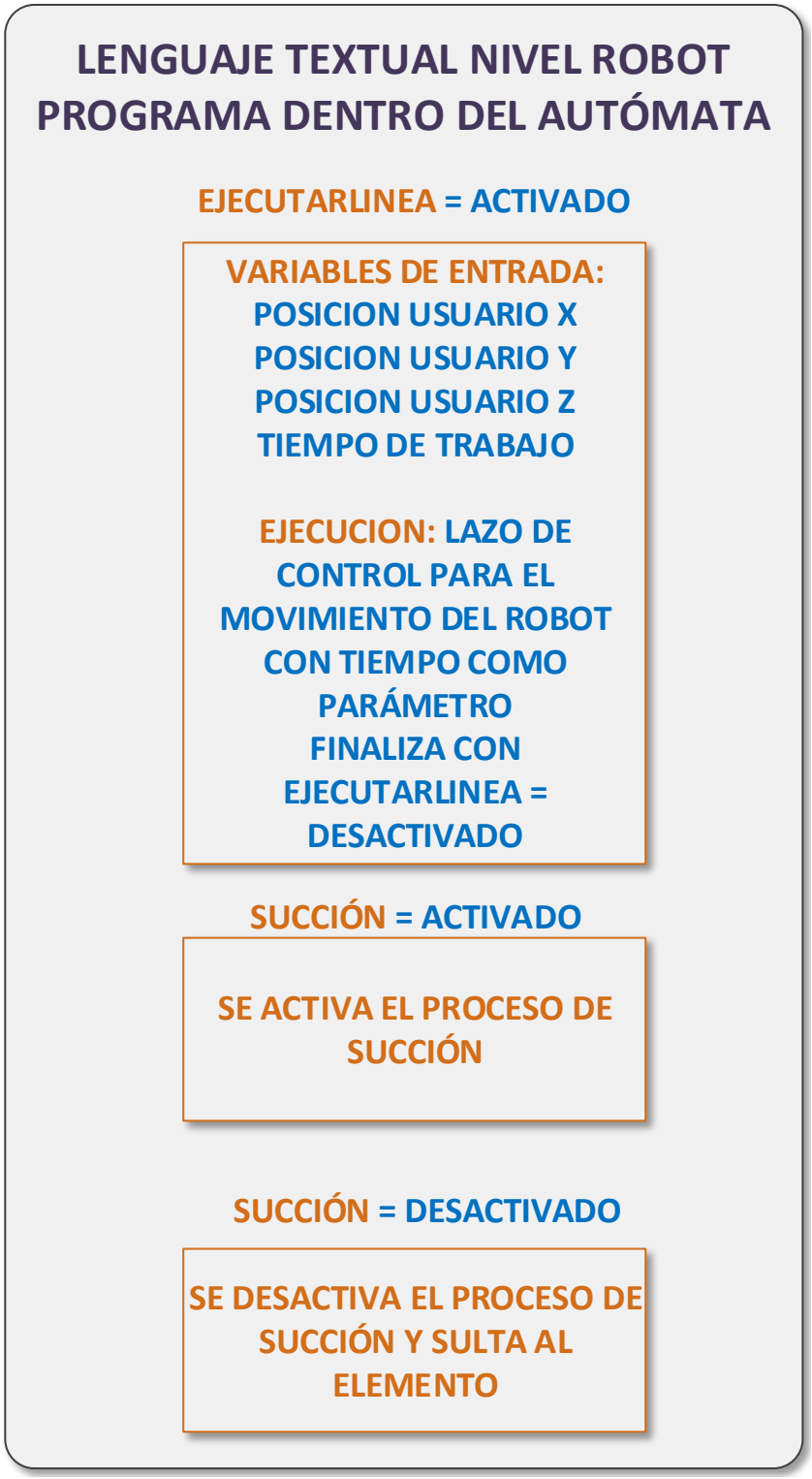
Tabla 6. Lenguaje textual nivel robot: Ventana de datos 1

Variable en la interfaz	Variable en el autómata	Descripción
POSICION USUARIO X_HMI	POSICION USUARIO X	Esta variable almacena el valor ingresado por el usuario para la coordenada en el eje X, se da en milímetros
POSICION USUARIO Y_HMI	POSICION USUARIO Y	Esta variable almacena el valor ingresado por el usuario para la coordenada en el eje Y, se da en milímetros
POSICION USUARIO Z_HMI	POSICION USUARIO Z	Esta variable almacena el valor ingresado por el usuario para la coordenada en el eje Z, se da en milímetros
TIEMPODETRABAJO_HMI	TIEMPODETRABAJO	Esta variable almacena el valor ingresado por el usuario para indicar el tiempo de trabajo.
SUCCION_HMI	SUCCION	Esta variable almacena el estado en el cual se debe encontrar el sistema de succión

- **Programa en el autómata**

El programa en el autómata para este lenguaje de programación se presenta en la Figura 63.

Figura 63. Lenguaje textual nivel robot: programa en el autómata



6.4 LENGUAJE TEXTUAL NIVEL TAREA

En el lenguaje textual nivel tarea el usuario ingresa la orden en una sola instrucción.

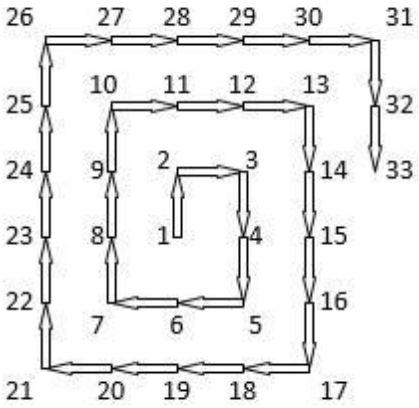
Para el caso específico de este proyecto el lenguaje de programación se enfoca en tres tareas: MOVER, SUJETAR ELEMENTO, MOVER Y SOLTAR.

MOVER: Al indicar esta tarea, el sistema robótico debe moverse a la posición dada y con la velocidad especificada.

MOVER Y SOLTAR: Esta tarea solo se realiza si el robot se encuentra sujetando una pieza, pues esta se decodifica en moverse a la posición dada por el usuario con la velocidad especificada, una vez llega a esta zona el robot libera al objeto.

SUJETAR ELEMENTO: Esta tarea solo se realiza si el robot no tiene sujeto elemento alguno, el sistema se mueve a la posición especificada, activa la succión, si el objeto de trabajo es agarrado en el primer intento se cumple con la tarea y se mantiene sujeto por el robot, en caso contrario el sistema inicia moviéndose por una serie de puntos que describen una espiral cuadrada, succionando en cada punto que se detiene con el objetivo de sujetar al elemento presente en esta zona (Figura 64).

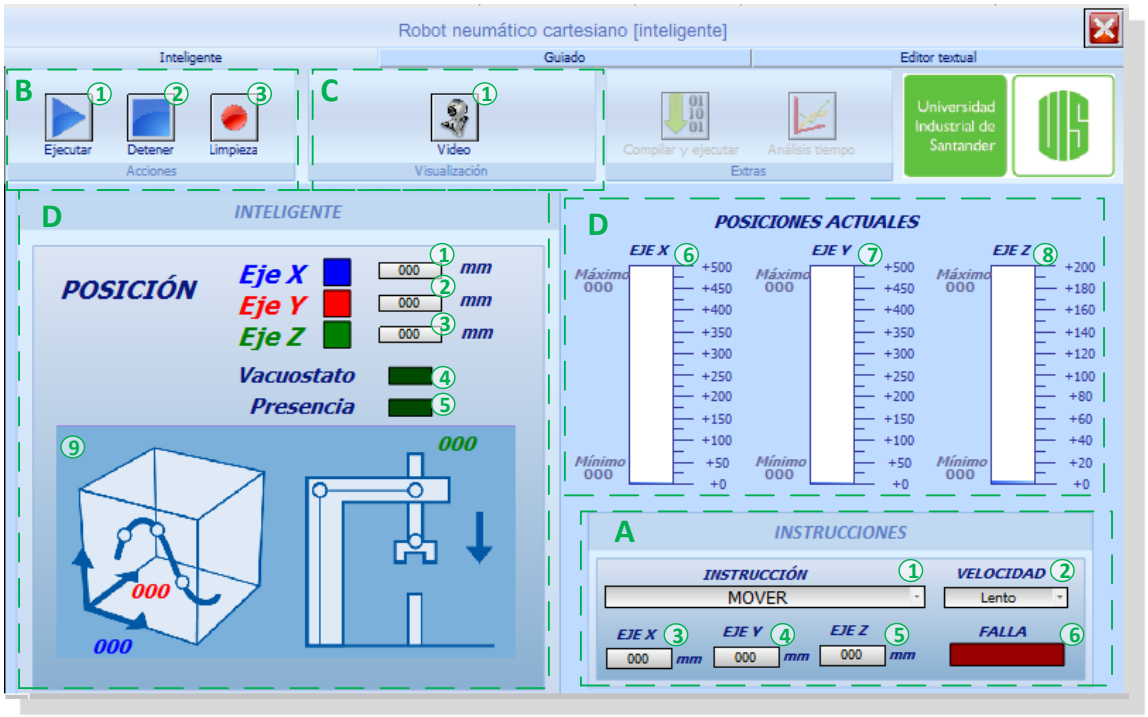
Figura 64. Espiral cuadrada



- **Interfaz para el lenguaje**

La interfaz presente para este lenguaje se representa en la Figura 65.

Figura 65. Interfaz lenguaje textual nivel tarea



Fuente: Banco laboratorio automatización industrial

La interfaz se divide en 4 secciones**.**

❖ ***Sección A: Instrucciones*** En este punto se especifican las tareas y los parámetros de trabajo

○ ***Elemento 1:*** *Instrucción* Se presenta un menú desplegable con las tres instrucciones de trabajo “MOVER”, “MOVER Y SOLTAR” y “SUJETAR ELEMENTO”. La variable en la interfaz asociada para conocer el número que caracteriza al estado es MODOSINTELIGENTE_HMI (MODOSINTELIGENTE en el Autómata).

○ ***Elemento 2:*** *Velocidad* Se despliega un menú con dos opciones las cuales son “Rápido” y “Lento”, donde el primer comando indica que el tiempo en el que se realiza el movimiento debe ser el menor posible por tanto la velocidad es la máxima, el segundo contrario ordena que el tiempo debe ser el máximo por tanto la velocidad es la más baja posible. La variable en la interfaz asociada para almacenar el número que identifica la velocidad escogida es MOV_inteligente_HMI (en el autómata es MOV_inteligente)

○ ***Elemento 3, 4 y 5:*** *Posiciones dadas por el usuario* En esta zona el usuario especifica las coordenadas de la posición a la cual desea llegar. Las variables que almacenan dichos valores son POSICION INTELIGENTE USUARIO X_HMI (en el autómata POSICION INTELIGENTE USUARIO X), POSICION INTELIGENTE USUARIO Y_HMI (en el autómata POSICION INTELIGENTE USUARIO Y), POSICION INTELIGENTE USUARIO Z_HMI (en el autómata POSICION INTELIGENTE USUARIO Z).

○ ***Elemento 6:*** *Led falla* Cuando el sistema ingresa a falla, el indicador parpadea a una frecuencia fija. La variable asociada al estado de este elemento es FALLA_INTELIGENTE_HMI (en el autómata FALLA_INTELIGENTE).

❖ ***Sección B: Acciones*** Se asocia con los proceso principales de este lenguaje de programación

○ ***Elemento 1: Ejecutar*** Al pulsar este botón se da inicio a la tarea junto con los parámetros especificados. Cuando se pulsa se activa la variable EJECUTARINTELIGENTE_HMI (en el autómata es EJECUTARINTELIGENTE_HMI)

Al pulsarlo, el sistema puede entrar en falla ya sea porque el movimiento programado no se puede realizar, o porque se da la orden de sujetar cuando el objeto ya se encuentra succionado por el robot, también cuando se ejecuta la tarea de soltar y no hay ningún elemento sujeto a la ventosa del robot, por último se puede especificar que el sistema va a falla cuando al terminar la búsqueda de un objeto en el estado “SUJETAR ELEMENTO” no ha encontrado alguno de ellos.

○ **Elemento 2: Detener** Al pulsarlo se detiene la tarea que se esté ejecutando. La variable que se activa al pulsarlo es DETENERINTELIGENTE_HMI (en el autómata es DETENERINTELIGENTE).

○ **Elemento 3: Limpieza** Al pulsarlo se activa la variable LIMPIEZAINTELIGENTE_HMI, la cual se encarga de reiniciar las variables desactivando la variable que representa la Falla del sistema.

❖ **Sección C: VISUALIZACIÓN**

○ **Elemento 1: Video** Cuando este se pulsa, se abre en la interfaz el programa que permite una vigilancia remota en tiempo real del estado del robot.

Figura 66. Lenguaje textual nivel tarea: Visualización

Fuente: Banco laboratorio automatización industrial

❖ **Sección D: Sensores**

○ **Elemento 1 y 6: Posición eje X** Estos elementos representan la posición del eje X, están asociados a la variable POTENCIOMETROX_HMI (en el autómata POTENCIOMETROX)

○ **Elemento 2 y 7: Posición eje Y** Estos elementos representan la posición del eje Y, están asociados a la variable POTENCIOMETROY_HMI (en el autómata POTENCIOMETROY)

○ **Elemento 3 y 8: Posición eje Z** Estos elementos representan la posición del eje Z, están asociados a la variable POTENCIOMETROZ_HMI (en el autómata POTENCIOMETROZ)

- **Elemento 4: Vacuostato** Este elemento representa el estado del sensor vacuostato, el cual al activarse indica si se encuentra sujeto un elemento. La variable asociada a este led virtual es VACUOSTATO_HMI (en el autómata VACUOSTATO)

- **Elemento 5: Sensor de presencia** Este elemento almacena el estado del sensor de presencia, el cual se activa cuando se presenta un elemento cerca a la ventosa plana. La variable asociada a este led virtual es PRESENCIA_HMI (en el autómata PRESENCIA)

- **Elemento 9: Diagrama de posiciones** En este se muestra en tiempo real de forma esquemática la posición del robot neumático cartesiano en los tres ejes.

- **Ventana de datos**

Los datos presentes en estas tablas permiten comunicar las acciones de la interfaz dadas por el usuario con los procesos internos del autómata programable. El valor de las variables pueden ser modificadas por el autómata o por la interfaz.

Tabla 7. Lenguaje textual nivel tarea: ventana de datos 1

Variable en la interfaz	Variable en el autómata	Descripción
MODOSINTELIGENTE_HMI	MODOSINTELIGENTE	Esta variable almacena el número que idientifica la tarea ha realizar (1 "MOVER" , 2 "SUJETAR ELEMENTO", 3 "MOVER Y SOLTAR")
MOV_inteligente_HMI	MOV_inteligente	Esta variable almacena el número que identifica la velocidad a la cual se va a mover (1 "Rápido", 2 "Lento")
POSICION INTELIGENTE USUARIO X_HMI	POSICION INTELIGENTE USUARIO X	Esta variable almacena el valor ingresado por el usuario para la coordenada en el eje X, se da en milímetros
POSICION INTELIGENTE USUARIO Y_HMI	POSICION INTELIGENTE USUARIO Y	Esta variable almacena el valor ingresado por el usuario para la coordenada en el eje Y, se da en milímetros
POSICION INTELIGENTE USUARIO Z_HMI	POSICION INTELIGENTE USUARIO Z	Esta variable almacena el valor ingresado por el usuario para la coordenada en el eje Z, se da en milímetros
FALLA_INTELIGENTE_HMI	FALLA_INTELIGENTE	Esta variable indica si el sistema se encuentra en falla, si se encuentra en falla esta se encuentra activa, en caso contrario se presenta desactivada

Tabla 8. Lenguaje textual nivel tarea: ventana de datos 2

Variable en la interfaz	Variable en el autómata	Descripción
EJECUTARINTELIGENTE_HMI	EJECUTARINTELIGENTE	Variable que al activarse ejecuta los protocolos almacenados dentro del autómata para cada tarea
DETENERINTELIGENTE_HMI	DETENERINTELIGENTE	Variable que al activarse hace que los procesos en ejecución se detengan y terminen
POTENCIOMETROX_HMI	POTENCIOMETROX	Esta variable almacena el valor de la posición actual del robot en el eje X
POTENCIOMETROY_HMI	POTENCIOMETROY	Esta variable almacena el valor de la posición actual del robot en el eje Y
POTENCIOMETROZ_HMI	POTENCIOMETROZ	Esta variable almacena el valor de la posición actual del robot en el eje Z
VACUOSTATO_HMI	VACUOSTATO	Esta variable almacena el estado del vacuostato presente en el robot, el cual al activarse indica que un elemento es sujetado por el robot
PRESENCIA_HMI	PRESENCIA	Esta variable almacena el estado del sensor de presencia del robot, el cual al activarse indica que se encuentra un objeto cerca de la ventosa plana

- **Programa en el autómata**

En esta sección se muestra el funcionamiento interno de los algoritmos programados dentro del autómata y el uso de las variables de la ventana de datos en la estructura de los mismos.

La estructura del funcionamiento se resume en las Figuras 67 a la 70.

Figura 67. Lenguaje textual nivel tarea: programa autómata 1

Figura 68. Lenguaje textual nivel tarea: programa autómata 2

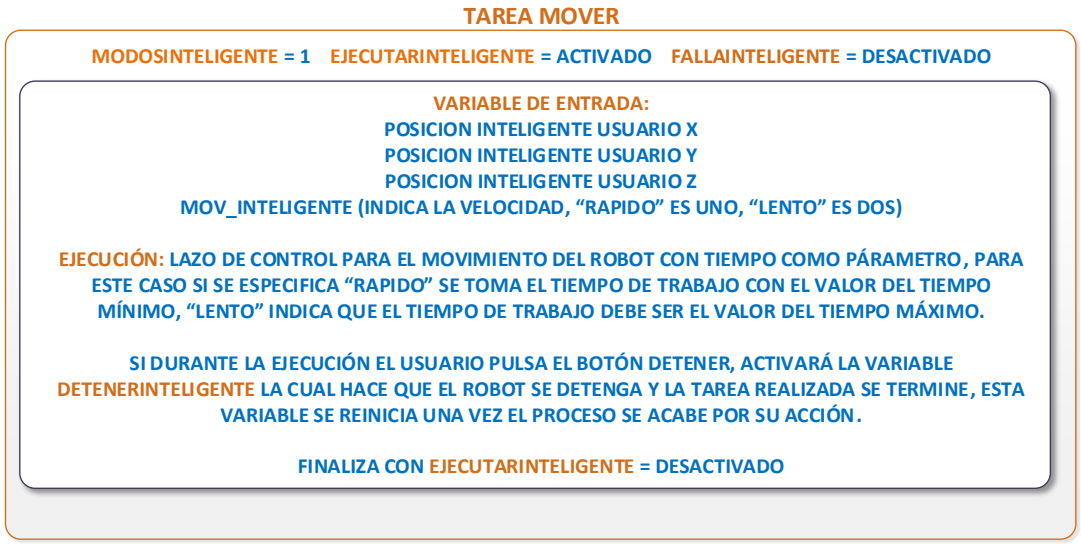


Figura 69. Lenguaje textual nivel tarea: programa autómata 3

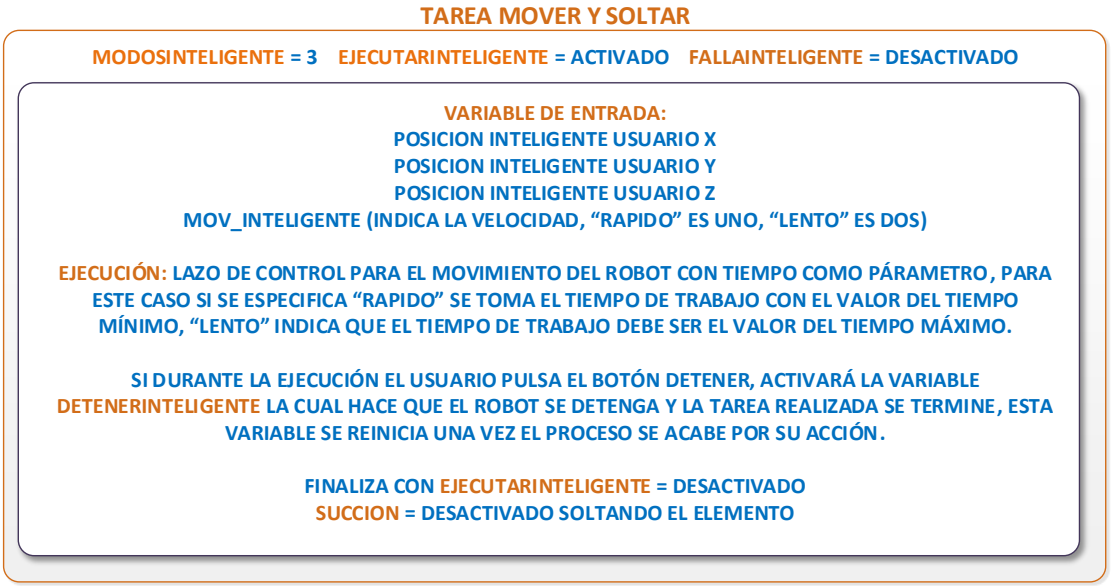
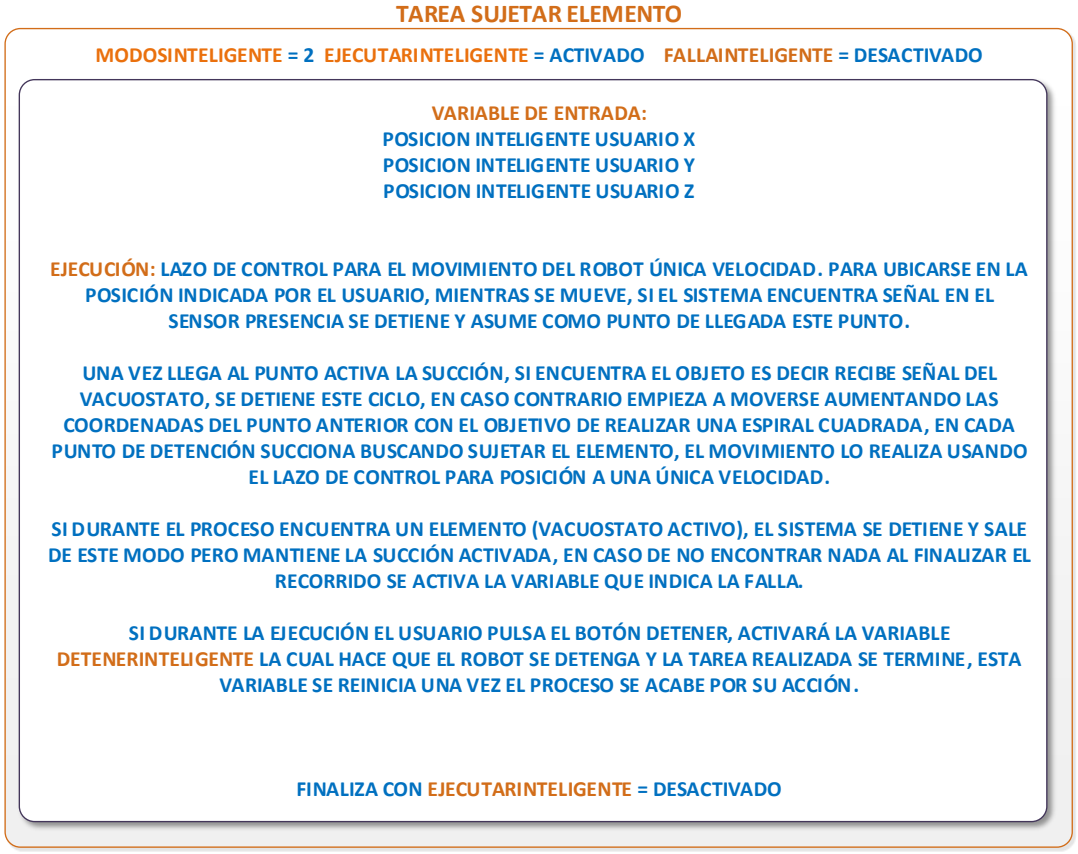


Figura 70. Lenguaje textual nivel tarea: programa autómata 4



6.5 LENGUAJES DE PROGRAMACIÓN DESARROLLADOS EN EL PROYECTO: DEMOSTRACIÓN COMO MODELOS VÁLIDOS PARA EL APRENDIZAJE

El objetivo principal del proyecto es desarrollar un banco que permite el aprendizaje de los lenguajes de programación para sistemas robóticos, por esta razón se hace importante validar los modelos de los lenguajes desarrollados para el robot neumático cartesiano.

En la actualidad no existe un estándar legal que gobierne sobre las estructuras que debe poseer un sistema de programación para robots industriales, por esta razón para la creación de los lenguajes se toma como referencia la teoría que ha sido aceptada a nivel internacional en el tema de robótica.

En la presente sección se demuestra como los sistemas desarrollados para el proyecto cumplen con los requerimientos establecidos en los lenguajes de programación de sistemas robóticos (ver Anexo C).

- **Programación por medio de un sistema guiado activo**

  - Desde el punto de vista del control de movimiento del robot.

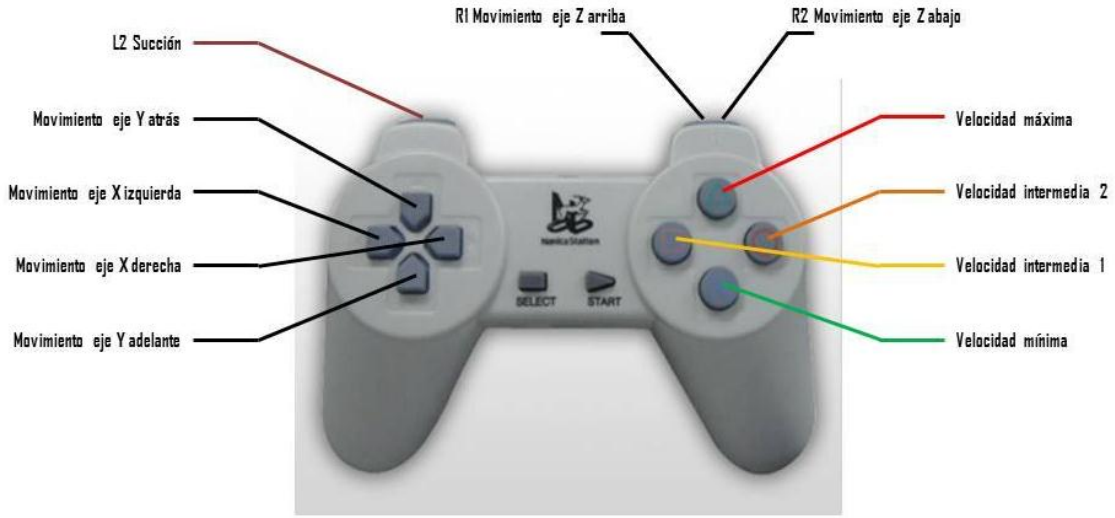
Posee joystick basado en una serie de once pulsadores, los cuales por medio de señales digitales indican al robot la instrucción que desea el operador. Cuenta con un juego de seis botones que determinan la dirección en la cual desea moverse ya sea hacia la derecha, izquierda, atrás, adelante, arriba y abajo, cuatro pulsadores que indican la velocidad de trabajo, que va desde una velocidad mínima hasta una máxima con dos estados intermedios, uno mayor al otro.

Para realizar el movimiento el usuario debe presionar al menos un botón encargado de la dirección y solo un pulsador de velocidad, ya que si especifica

más de dos o ninguno el sistema no se moverá. Otra razón por la cual el efector final no tendrá movimiento sucede si el operario presiona dos botones que indiquen direcciones opuestas como en el caso que apriete el pulsador que determina hacia la derecha y hacia la izquierda simultáneamente, ya que este movimiento es físicamente imposible de realizar.

Además de las acciones de movimiento el joystick le permite al usuario efectuar la segunda tarea del robot la cual es la de sujetar elementos, para esto por medio de uno de los pulsadores el operario puede activar o desactivar la acción de succión, si se encuentra desactivada la acción de succionar y el usuario pulsa el botón determinado para esto, el robot comenzará a succionar lo que se encuentre debajo de su efector final. Si el usuario vuelve a presionar el botón la acción se detendrá.

Figura 71. Esquema de trabajo del joystick

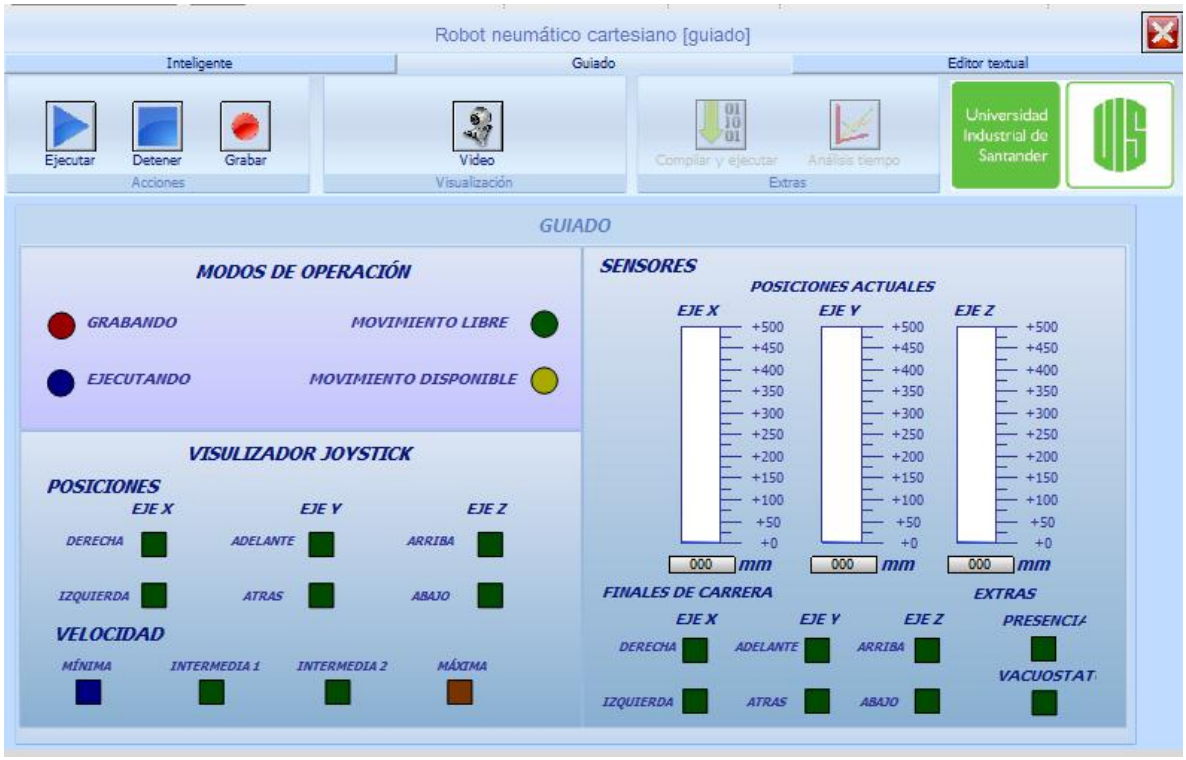


Fuente: NANICASTATION®

- Desde el punto de vista del entorno de programación

El usuario cuenta con la siguiente interfaz de trabajo.

Figura 72. Interfaz del sistema guiado activo



Fuente: Banco laboratorio automatización industrial

En la interfaz, el usuario encuentra los comandos necesarios para un sistema de programación guiado, ya que cuenta con un sistema de grabación y otro de ejecución de los movimientos memorizados. Además puede observar la forma de trabajo por medio de leds luminosos ubicados en la sección de modos de operación. En la parte de visualización del joystick el operario puede confirmar los pulsadores que está activando desde el control de mando.

Como parte integral de la interfaz se dedica una sección para la visualización del sistema sensorial con el cual cuenta el robot, en este segmento se ilustra la ubicación del robot dada por los sensores de posición (potenciómetros), los límites máximos y mínimos de trabajo dados por los finales de carrera, el estado del sensor de presencia, el cual al activarse indica que existe un elemento cerca del

efector final, el valor actual del vacuostato, si este se encuentra activo indica que un elemento está siendo succionado por el actuador final.

- Desde el punto de vista de modelado del entorno

El robot neumático representa la posición del efector final a partir de un sistema de coordenadas cartesianas donde cada dimensión depende de cada uno de los tres actuadores, lo cual representa que para recrear un movimiento en un solo eje el desplazamiento se debe a un solo actuador, para moverse en dos ejes de forma simultanea se usan dos actuadores, el mismo caso aplica para un movimiento tridimensional el cual se genera por el cambio de la posición de los tres actuadores simultáneamente.

La representación que tiene el robot de los objetos de trabajo no depende ni de la masa, peso o las dimensiones, solo se basa en el lugar donde se encuentran debajo de su efector final localizado por medio del sensor de presencia.

- Desde el punto de vista del tipo de datos

Los datos presentes en este modo de programación van enfocados hacia la parte sensorial del sistema, ya que se tiene una representación por medio de números enteros de la ubicación del actuador final expresada en milímetros, además cuenta una serie de leds virtuales que representan el estado de los sensores (presencia, vacuostato, finales de carrera), de los pulsadores del control de mando y de los modos de trabajo.

Lo cual indica que la naturaleza de los datos para este sistema de programación corresponde al tratamiento de señales del tipo digital y analógico.

- Desde el punto de vista del manejo de entradas y salidas

La interfaz del lenguaje guiado ofrecido por el robot neumático cuenta con un sistema de monitoreo, que le permite al usuario desde el computador donde ejecuta la interfaz la visualización de los movimientos realizados por el autómata

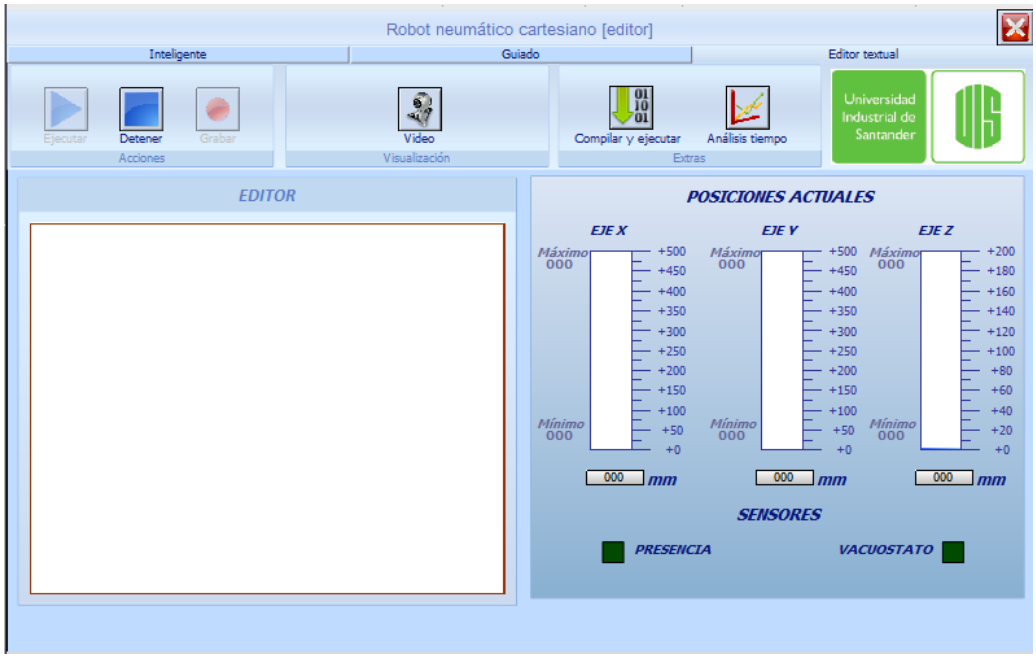
en tiempo real, sin necesidad de encontrarse frente a él, lo cual facilita al usuario el trabajo ya que puede interactuar con la interfaz y el control de mando sin dejar su puesto de trabajo.

- Desde el punto de vista del control del flujo del programa

En el sistema guiado desarrollado en este proyecto no se presenta el uso de ciclos de flujo del programa basados en las instrucciones de un lenguaje de programación de computadoras (if, While, for) ya que las dos operaciones de trabajo no las requieren, pues una ellas se encarga de memorizar los movimientos los cuales son determinados por el usuario al usar el mando de control y la otra de ejecutarlos nuevamente sin necesidad de la presencia del usuario.

- **Programación textual nivel robot**
  - Desde el punto de vista del entorno de programación

Figura 73. Entorno de programación textual enfocado al robot



Fuente: Banco laboratorio automatización industrial

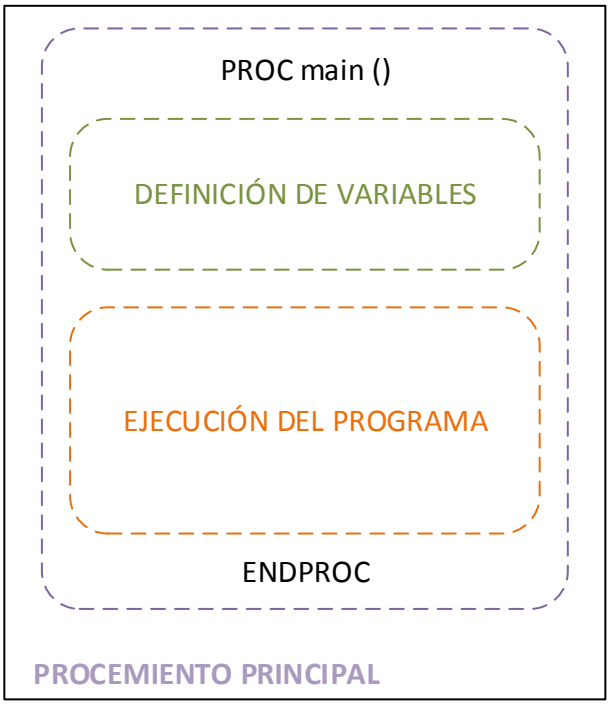
En este lenguaje el usuario cuenta con un editor textual por medio del cual puede dar las instrucciones al robot neumático.

La estructura de un programa por medio del editor textual establece las siguientes características:

El programa es escrito en un procedimiento, el cual es el ciclo principal de trabajo.

Un procedimiento se divide en dos partes, en la primera de ellas el usuario define todas las variables de trabajo, en la segunda parte se especifican todas las funciones, instrucciones y ciclos de flujo del programa que desea realizar.

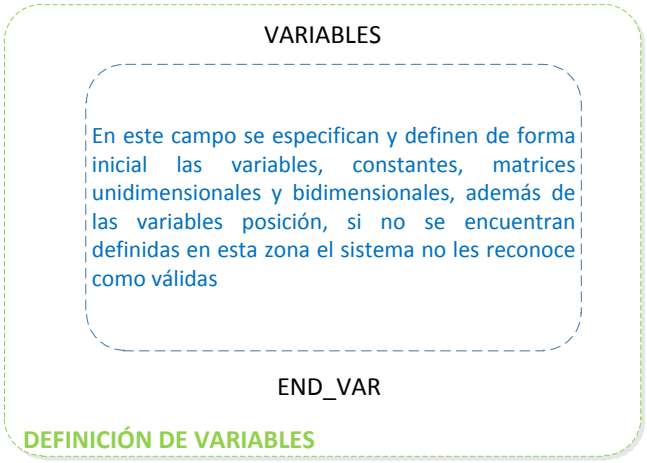
Figura 74. Estructura de un procedimiento lenguaje textual enfocado al robot



Para iniciar el procedimiento principal se usa la instrucción “PROC main ()”, para cerrarlo se usa la palabra “ENDPROC”, si el usuario escribe partes del código fuera de la zona delimitada por estos dos comandos; al momento de ejecutar el código, el sistema informará de este error y no se ejecuta ningún proceso.

La definición de variables se hace dentro de los comandos “VARIABLES” y “END_VAR” donde el primero permite iniciar la definición y el segundo determina el cierre de definición de variables.

Figura 75. Definición de las variables en el lenguaje textual enfocado al robot



Dentro de la zona de definición e instrucciones el usuario cuenta con cuatro posibilidades de trabajo, la primera es la asignación de valores, la segunda es la asignación de funciones, la tercera es la ejecución de instrucciones, la cuarta corresponde al uso de un ciclo de control del flujo del programa:

La asignación de valores consiste en almacenar un resultado dentro de una variable previamente definida, este dato puede darse de procesos aritméticos tales como la suma, resta, multiplicación, división, potenciación u operaciones de suma y resta entre elementos matriciales, también el almacenamiento de pequeños fragmentos textuales.

La asignación de funciones se basa en guardar un resultado dentro de una variable definida con anterioridad, donde el valor obtenido proviene del uso de operaciones definidas tales como la evaluación del valor absoluto de un elemento, el logaritmo natural, las potencias del número Euler, además de propiedades trigonométricas tales como el seno, coseno y tangente de un ángulo.

Cuando se hace referencia al uso de instrucciones, son aquellas que el sistema cumple a partir de unos parámetros de entrada, entre estas encontramos las siguientes:

SUCCION, cuando el usuario escribe este comando el sistema que realiza la succión se activa.

SOLTAR, cuando se usa esta instrucción el sistema desactiva el proceso de succión.

MOVER, el robot mueve su efector final según la posición y el tiempo de trabajo especificados.

MENSAJE, esta instrucción envía un mensaje al usuario con el valor almacenado dentro de una variable.

Los ciclos de control de flujo del programa estipulados en este lenguaje corresponden a las estructuras If, For, While presentadas en los lenguajes de programación para computadoras.

- Desde el punto de vista del modelado del entorno

El robot neumático representa la posición del efector final a partir de un sistema de coordenadas cartesianas donde el movimiento en cada dimensión está determinado por el desplazamiento de cada actuador por separado, lo cual representa que para recrear un movimiento en un solo eje el desplazamiento se debe a un solo actuador, para moverse en dos ejes de forma simultanea se usan dos actuadores, de la misma manera se aplica para un movimiento tridimensional el cual se genera por el cambio de la posición de los tres actuadores simultáneamente.

La representación que tiene el robot de los objetos de trabajo no depende de las propiedades físicas del objeto, solo se basa de la posición donde pueden ser detectados por el sensor de presencia para ser succionados por el efector final.

- Desde el punto de vista del tipo de datos

En este lenguaje de programación el sistema se basa en el manejo de tipo diferentes de datos. Los cuales se describen a continuación:

Variables numéricas: Este tipo de datos representan cantidades numéricas, tales como valores enteros, decimales, cualquier elemento que pertenezca al conjunto de los reales, una de las propiedades importantes de las mismas es que pueden ser modificadas como parte de las operaciones del procedimiento principal, para crearlas en la zona de variables se usa el comando VAR seguido de la palabra num y el nombre dado por el usuario para que identifique al parámetro numérico.

Figura 76. Variables numéricas lenguaje textual enfocado al robot

Constantes numéricas: Al igual que las variables numéricas representan elementos pertenecientes al orden de los reales, una diferencia primordial es que su valor solo puede ser definido en la zona de variables y no puede ser modificado durante el proceso de ejecución del procedimiento principal. Se definen con el comando CONS, seguido de la palabra num y el nombre de la variable.

Figura 77. Constantes numéricas lenguaje textual enfocado al robot

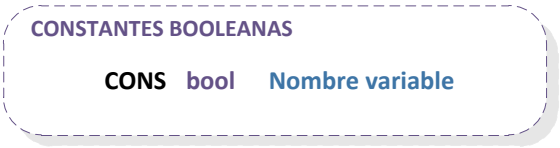
Variables booleanas: Son variables del tipo lógico, cuya magnitud se puede encontrar en dos estados, Verdadero o Falso. Estas pueden su valor puede ser modificado como resultado de un proceso del procedimiento principal. Su

definición se hace dentro de la zona de variables por medio del comando VAR seguido de la palabra bool y el nombre indicado por el usuario.

Figura 78. Variables booleanas lenguaje textual enfocado al robot

Constantes booleanas: Al igual que las variables booleanas la magnitud almacenada obedece a una naturaleza de dos estados, la diferencia con estas es que su valor no puede ser modificado como resultado de un proceso en el procedimiento principal, su valor solo puede ser definido en la zona de variables. Para definirla se usa el comando CONS seguido de la palabra bool y el nombre dado por el usuario.

Figura 79. Contantes booleanas lenguaje textual enfocado al robot



Matrices unidimensionales: Son elementos que pueden almacenar una serie de datos numéricos dados por el usuario, en este caso se asigna a una sola dimensión la capacidad de almacenamiento, estos valores pueden ser asignados tanto en la zona de variables y también como resultado de un proceso en el procedimiento principal. Para definirlas dentro de la zona de variables se usa el comando MATR seguido del nombre de la matriz, el elemento de apertura "(", luego la cantidad de datos máxima que permite almacenar, terminando con el elemento de cierre ")".

Figura 80. Matrices unidimensionales lenguaje textual enfocado al robot

Matrices bidimensionales: Son elementos que pueden almacenar una serie de datos numéricos dados por el usuario, estos pueden ser almacenados en dos dimensiones, los valores pueden ser asignados dentro de la zona de variables o como resultado de un proceso dentro del procedimiento principal. Para definirlas dentro de la zona de variables se usa el comando MATR, seguido del nombre de la matriz, el elemento de apertura "(", la máxima cantidad de datos almacenados en la dimensión principal (máxima cantidad de filas), el elemento de separación ",", la máxima cantidad de datos almacenados para la dimensión secundaria (máxima cantidad de columnas), finalizando con el elemento de cierre ")".

Figura 81. Matrices bidimensionales lenguaje textual enfocado al robot

Variables tipo string: Estas variables pueden almacenar cadenas de caracteres con una longitud máxima de 100 elementos, además dentro del proceso en el procedimiento pueden almacenar el resultado de las variables y constantes numéricas, booleanas, valores de matrices unidimensionales y bidimensionales, el tamaño de las matrices. En la zona de variables son definidas por medio del comando VAR seguido de la palabra string y el nombre dado por el usuario.

Figura 82. Variables tipo string lenguaje textual enfocado al robot

Constantes tipo string: Estas variables pueden almacenar solo cadenas de caracteres con una longitud máxima de 100 elementos, solo pueden ser asignadas en la zona de variables para hacerlo se inicia con el parámetro CONS seguido de la palabra string y el nombre dado por el usuario.

Figura 83. Constantes tipo string lenguaje textual enfocado al robot

Variables tipo posición: Son variables que almacenan las coordenadas de las posiciones a las cuales desea llegar el usuario, debido a que el sistema obedece a una naturaleza tridimensional esta variable debe guardar tres elementos numéricos que indican la posición en el espacio expresada en milímetros.

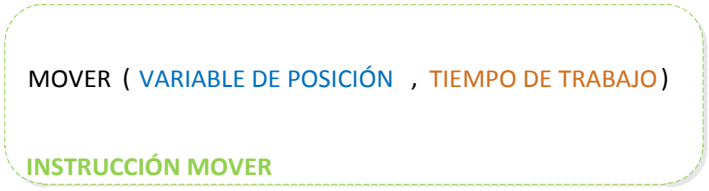
Este tipo de variables pueden ser modificadas como parte de un proceso dentro del procedimiento principal fuera de la zona de variables. Para declararlas dentro de la definición de variables se usa el comando pos seguido del nombre de la variable, el elemento de apertura “(“, la cantidad numérica que indica el valor del eje X, el elemento de separación “,”, la cantidad numérica que determina el valor del eje Y, el elemento de separación “,”, la cantidad numérica que dicta el valor del eje Z, finalizando con el elemento de cierre “)”.

Figura 84. Variables tipo posición lenguaje textual enfocado al robot

- Desde el punto de vista del control de movimiento del robot.

El movimiento del robot se hace por medio de la instrucción MOVER, las cuales posee como argumentos de entrada la posición a la cual se desea llegar usando una variable de posición y el tiempo de trabajo.

Figura 85. Instrucción mover lenguaje textual enfocado al robot



Debido al sistema de control presentado para el movimiento del robot, el valor del tiempo de trabajo posee dos límites uno mínimo y otro máximo, por esta razón la interfaz del editor textual enfocado al robot cuenta con una ventana adicional diseñada estrictamente para indicar el cálculo del valor máximo y mínimo del tiempo como función de las coordenadas del punto inicial y final del desplazamiento, además le informa al usuario si un movimiento es posible o no. Como dato adicional, permite calcular la velocidad promedio que logra cada eje en función del tiempo especificado por el usuario, de esta manera el operador obtiene una percepción de la precisión lograda por el sistema.

Figura 86. Interfaz adicional para el cálculo de los límites del tiempo

Fuente: Banco laboratorio automatización industrial

Las instrucciones de SUCCION y SOLTAR permiten al usuario activar o desactivar la actividad de succión para sujetar elementos.

Figura 87. Instrucción SUCCION para el lenguaje textual enfocado al robot

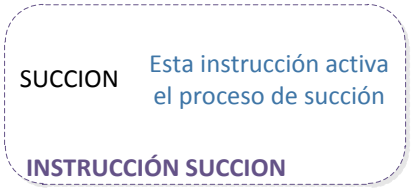
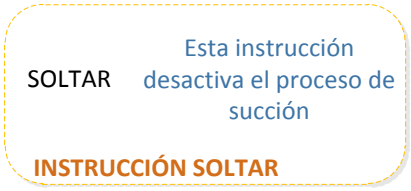


Figura 88. Instrucción SUCCION para el lenguaje textual enfocado al robot



- Desde el punto de vista del manejo de entradas y salidas

La interfaz del lenguaje textual enfocado al robot cuenta con un sistema de monitoreo, el cual le permite al usuario la operación desde la interfaz si necesidad de visualizar directamente al banco.

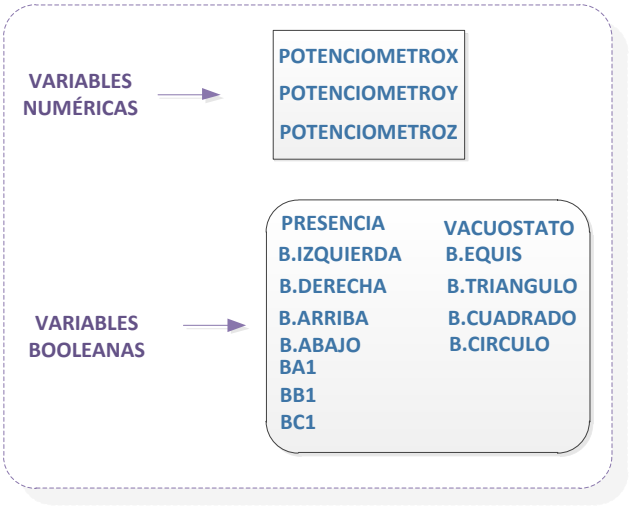
Además el editor textual le permite al usuario el uso de los sensores internos del robot dentro del código desarrollado por él.

Por medio de los comandos POTENCIOMETROX, POTENCIOMETROY, POTENCIOMETROZ, se obtiene el valor de la posición en cada una de las dimensiones, estos deben ser asignados a variables numéricas.

Los comandos PRESENCIA, VACUOSTATO, permiten obtener el estado del sensor de presencia y el sensor que indica si un elemento se encuentra sujeto por el sistema, estos deben asignarse a variables booleanas.

Los comandos B.ARRIBA, B.ABAJO, B.DERECHA, B.IZQUIERDA, BA1, BB1, BC1, B.EQUIS, B.TRIANGULO, B.CIRCULO, B.CUADRADO, permiten obtener los estados de los pulsadores del joystick, el cual se habilita para este lenguaje, estos comandos deben asignarse a variables booleanas.

Figura 89. Sensores internos para el lenguaje textual enfocado al robot



- Desde el punto de vista del control del flujo del programa

El editor textual usado en el lenguaje enfocado al robot presente en este proyecto consta con tres ciclos de flujo del programa los cuales son:

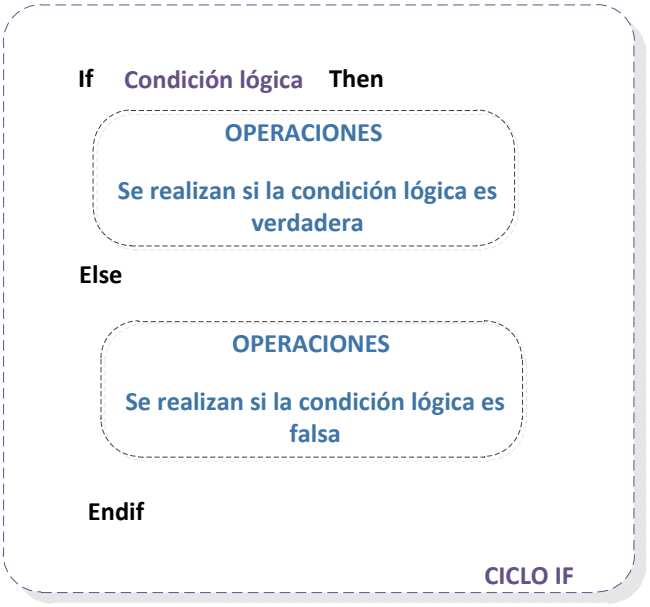
Ciclo If: Este ciclo corresponde a la estructura presentada en los lenguajes de computadora tradicionales, la cual opera de la siguiente manera:

El ciclo se inicia con el comando If seguido por una condición lógica, luego se ubica la palabra Then, a partir de esta instrucción se colocan todas las operaciones a realizar por el programa, el ciclo se cierra con el comando Endif, en algunos casos el usuario puede hacer uso del elemento Else el cual se ubica en el espacio formado entre las palabras Then y Endif.

Si la evaluación de la condición de tipo lógica ubicada al lado derecho del comando If tiene como resultado un valor verdadero se realizan las operaciones que se encuentran entre la palabra Then y el elemento de cierre Endif, si entre ellas se encuentra el comando Else el sistema solo realiza las instrucciones que se presenten entre las palabras Then y Else. Si el resultado obtenido es falso el

sistema no realiza ninguna operación ubicada entre el elemento Then y Endif, en caso que se presente el valor Else, el lenguaje realiza las operaciones indicadas entre las palabras claves Else y Endif.

Figura 90. Ciclo if lenguaje textual enfocado al robot



Ciclo For: Este ciclo permite repetir un proceso un número de veces determinadas por el usuario, el sistema parte de un valor inicial indicado por el usuario, aumenta en la cantidad señalada por el operador y se realiza mientras la cantidad almacenada sea menor o igual a la magnitud final indicada.

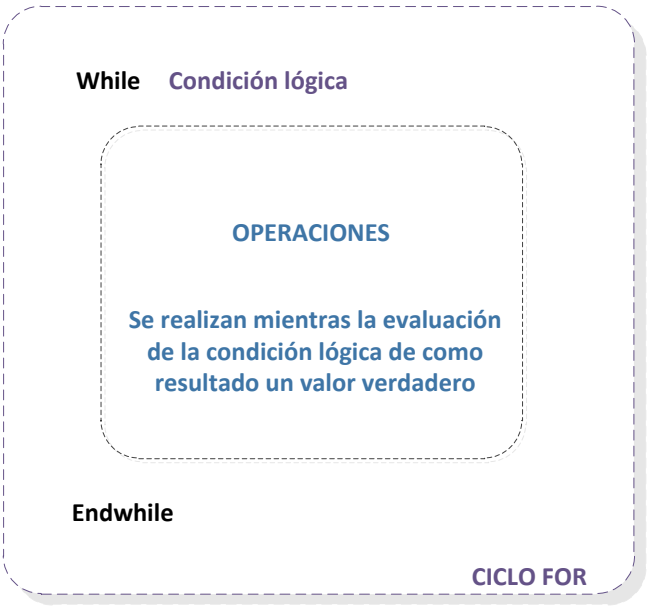
La estructura de trabajo es la siguiente: Se inicia con el comando FOR luego continua el valor inicial, la magnitud final y el paso, cada uno puede indicarse por medio de una cantidad, variable o constante numérica. Luego de estos valores siguen las operaciones a realizar por el sistema, el cierre de esta operación se da por medio del elemento ENDFOR, el ciclo repite las operaciones localizadas entre el valor final y elemento de cierre hasta que la cuenta almacenada deje de ser menor o igual al valor final.

Figura 91. Ciclo For lenguaje textual enfocado al robot

Ciclo While: Este ciclo realiza las operaciones especificadas en su interior mientras la evaluación de la condición lógica que lo rige dé como resultado un valor verdadero.

Su estructura inicia con el comando While seguido de una condición lógica, el sistema realizará las operaciones que se encuentren entre la instrucción lógica y la palabra Endwhile, siempre que la evaluación de la declaración lógica sea verdadera.

Figura 92. Ciclo While lenguaje textual enfocado al robot

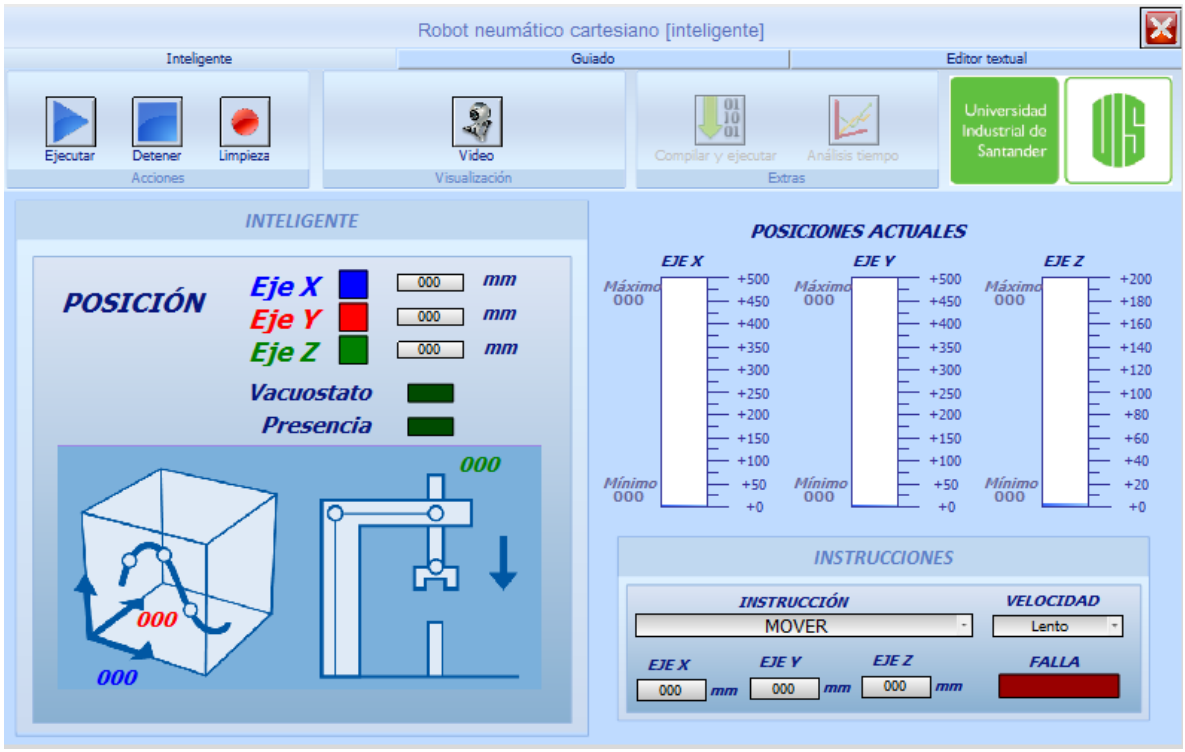


Ciclos anidados: El editor textual le permite al operador el uso de ciclos anidados dentro de otros, esto quiere decir que dentro de un ciclo ya sea If, For o While pueden ubicarse como operaciones internas ciclos if, for y While.

- **Programación textual nivel tarea**

  - Desde el punto de vista del entorno de programación

Figura 93. Interfaz general lenguaje de programación nivel tarea



Fuente: Banco laboratorio automatización industrial

La interfaz cuenta con bloque dedicado a las instrucciones, otro dedicado a las acciones del sistema y el último dedicado a la visualización del estado de los sensores.

En la sección referente a las instrucciones el usuario puede seleccionar la instrucción de trabajo a partir de un menú desplegable entre las opciones que tiene para elegir se encuentran: MOVER (Función en la cual el robot mueve el efector final a la posición indicada en la sección de instrucciones, a la velocidad determinada en el menú desplegable el cual puede estar en la opción de rápido o lento), SUJETAR ELEMENTO (Función en la cual si el robot no tiene ningún elemento sujeto, inicia su recorrido a la posición dada en la sección de

instrucciones, al llegar allí inicia a succionar, si no encuentra el elemento, el sistema inicia un recorrido en forma de espiral cuadrada con separaciones de 12 mm con el objetivo de encontrar el elemento), MOVER Y SOLTAR(Función en la cual el efector final se lleva a la posición indicada, una vez llega a la ubicación el elemento se libera).

En la sección de acciones el usuario puede encontrar tres botones los cuales tienen las siguientes funciones:

Al pulsar el botón ejecutar el sistema realiza la instrucción seleccionada con anterioridad por el usuario, en algunos casos puede entrar en falla, lo cual se indica creando un parpadeo en el led ubicado en la sección de instrucciones.

Los motivos por los cuales el sistema ingresa en falla son:

Si al seleccionar la instrucción MOVER o MOVER Y SOLTAR el movimiento no se puede realizar debido a que la posición final se encuentra fuera de los límites de trabajo o la relación que existe entre las diferencias de las distancias no corresponde a un movimiento que dé como resultado un segmento cercano a una línea.

Si al dar ejecutar a la instrucción SUJETAR ELEMENTO, ya se encuentra un elemento sujeto por el sistema o al terminar la búsqueda no encuentra ningún elemento.

Si al ejecutar la instrucción MOVER Y SOLTAR no hay sujeto un elemento el sistema ingresa en falla.

En caso que el sistema se encuentre en falla, el usuario pulsa el botón de limpieza, se retira el estado de falla y el robot queda listo para la ejecución de una nueva instrucción.

Si el usuario desea detener la ejecución de una instrucción cuenta con el botón detener.

El entorno de programación ofrece al usuario la posibilidad de conocer el estado de los sensores digitales asociados al evento de presencia de un elemento bajo el efector final y la sujeción de piezas (PRESENCIA Y VACUOSTATO), además de conocer la posición de la herramienta del robot.

- Desde el punto de vista del modelado del entorno

El sistema robótico se concentra en la posición de los elementos, desprecia su masa y dimensiones geométricas, solo se enfoca en el centro del área que necesita para poder sujetar al elemento. Dicha posición se da en función de un sistema absoluto de coordenadas, cuyos ejes están ligados de forma directa a cada uno de los actuadores del sistema de forma independiente, lo que quiere decir que mover un actuador provoca el desplazamiento en un solo eje.

- Desde el punto de vista del tipo de datos

En el lenguaje enfocado a la tarea, las variables presentadas son: la instrucción de trabajo (MOVER, MOVER Y SOLTAR, SUJETAR ELEMENTO), el valor de la velocidad (Rápido, Lento), la posición final (Posición en milímetros para los tres ejes de trabajo), el estado de los sensores digitales (Sensor de presencia y Vacuostato). El valor de estas variables determina el proceso que realiza el robot.

- Desde el punto de vista del control de movimiento del robot.

En este lenguaje el usuario cuenta con tres formas de movimiento:

***Instrucción MOVER**:* En esta el usuario configura la posición final a la cual desea llegar ubicando las coordenadas del punto en la sección de instrucciones, además de esto configura la magnitud de la velocidad expresada en dos posibles estados Rápido y Lento, si el usuario selecciona el parámetro rápido el sistema se

moverá a este punto a la mayor velocidad posible en caso de seleccionar lento la velocidad será la mínima posible para el recorrido.

***Instrucción SUJETAR ELEMENTO**:* En esta forma de movimiento el usuario configura un punto final, el parámetro de la magnitud de la velocidad no influye ya que todo el movimiento el sistema lo realiza a la mínima velocidad posible para obtener la mejor precisión posible. Una vez el robot llega al punto indicado por el usuario o detecta una señal en el sensor de presencia (lo que indica que se ha pasado cerca al objeto buscado), el movimiento se detiene activando la función de succión el sistema espera por tres segundos si no encuentra respuesta en el sensor que indica que ha sujeta el elemento inicia un proceso de búsqueda realizando una trayectoria en forma de espiral cuadrada, deteniéndose en cada punto con el objetivo de activar la succión por 3 segundos con el fin de sujetar el elemento que se encuentra por esta zona, para realizar cada uno de los segmentos el sistema sube 12 mm relativos a la altura inicial desde donde inició la búsqueda, si el robot logra sujetar al elemento la búsqueda se detiene quedando preparado para una nueva instrucción, si al terminar el sistema no encuentra ningún objeto se indica falla en la interfaz.

***Instrucción MOVER Y SOLTAR**:* Esta función realiza el movimiento de la misma forma como lo hace la instrucción MOVER, el usuario ubica las coordenadas del movimiento e indica la velocidad (Como función de los dos estados rápido o lento), la diferencia primordial radica en que este proceso no se puede ejecutar si el robot no se encuentra sujetando un elemento, ya que al llegar al punto de destino la actividad de la succión se desactivará soltando al objeto.

- Desde el punto de vista del manejo de entradas y salidas

El lenguaje textual nivel tarea presenta al usuario la oportunidad de visualización remota del robot neumático, lo que le permite al operador programar al sistema

robótico sin la necesidad de retirar su vista de la interfaz presentada para el lenguaje.

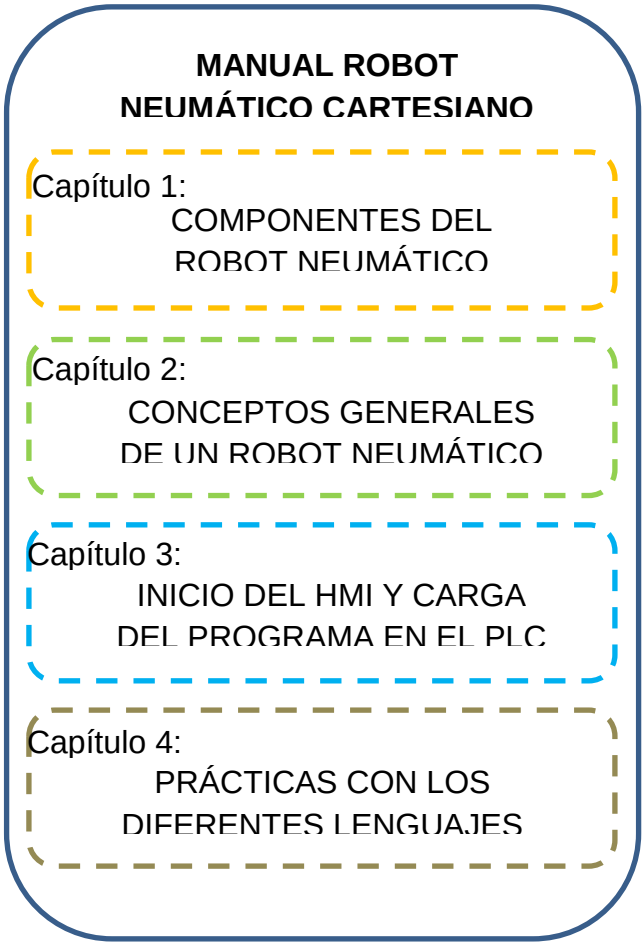
- Desde el punto de vista del control del flujo del programa

En este lenguaje el flujo del programa lo determina el usuario cada vez que ejecuta una instrucción puesto que tiene la oportunidad de detenerla en cualquier momento, para este sistemas de programación no se usan los ciclos if, for y While dados en los lenguajes de computadoras.

7. ROBOT NEUMÁTICO: MANUAL DE USUARIO

El manual del robot neumático cartesiano contiene cuatro capítulos, los capítulos 1 y 2 contienen teoría acerca del robot, de los componentes que conforman el robot neumático desarrollado en este proyecto y de los conceptos generales de un robot industrial. En el capítulo 3 se enseña al estudiante la forma de iniciar la interfaz hombre máquina y la carga del programa en el PLC 315F-2 PN/DP; por su parte, el capítulo 4 contiene 3 prácticas que muestran el uso de la interfaz HMI para cada uno de los lenguajes de programación.

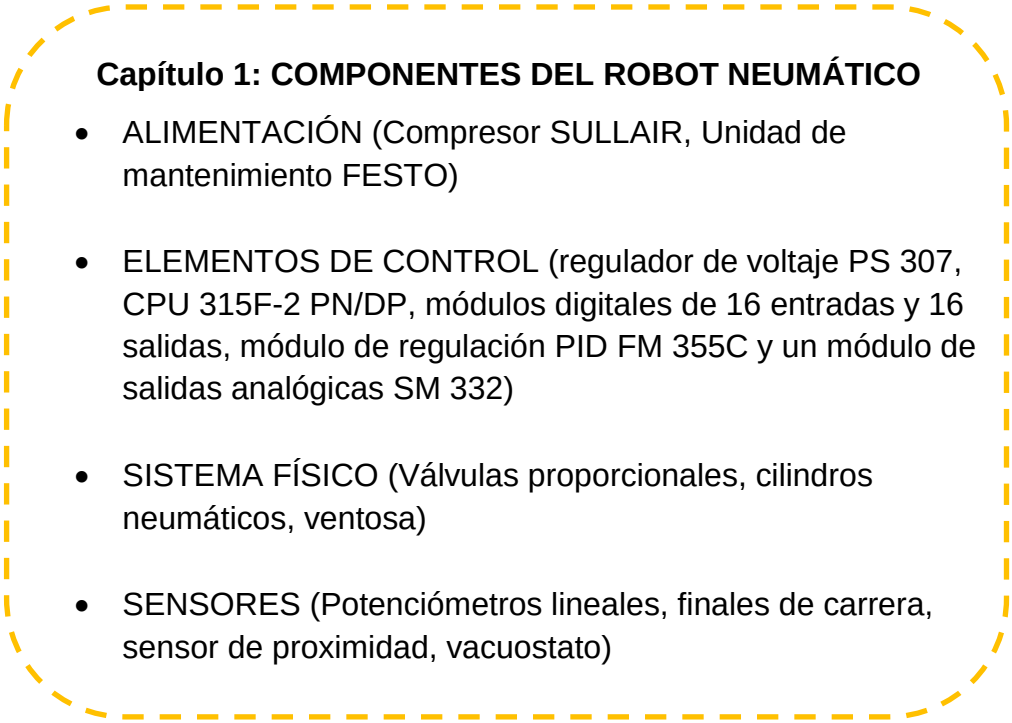
Figura 94. Estructura Manual de usuario



7.1 MANUAL DEL USUARIO: CAPÍTULO 1

En este capítulo se describen los componentes que permiten a la parte operativa del robot neumático cartesiano, la correcta ejecución de su función principal. Entre estos elementos están, los elementos de alimentación, control, sistema físico y sensores, éstos se agrupan en subsistemas para una mejor visualización de la estructura del robot.

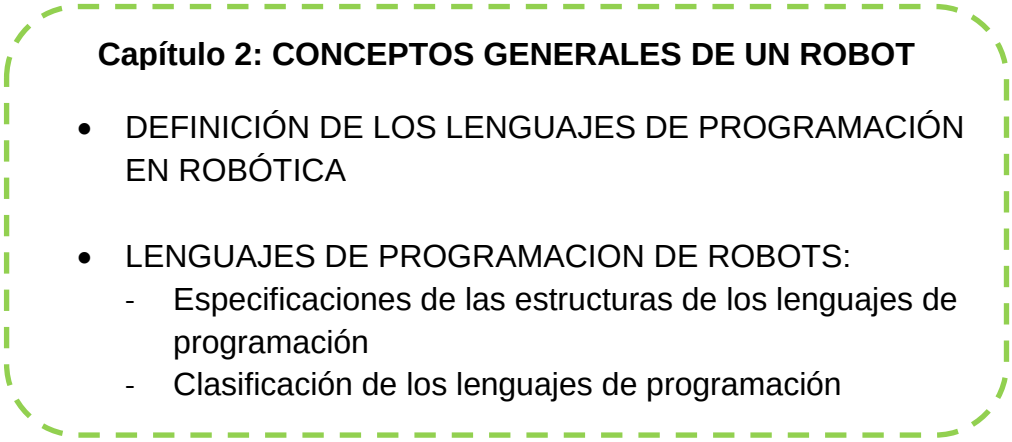
Figura 95. Capítulo 1 manual robot neumático cartesiano



7.2 MANUAL DEL USUARIO: CAPÍTULO 2

Aquí se da una definición acerca de los lenguajes de programación en robótica y su importancia en la industria moderna debido a su versatilidad y flexibilidad a los continuos cambios que se dan en las líneas de producción. En la segunda parte de este capítulo se dan las especificaciones generales que se han ido aceptando a través del tiempo para las interfaces hombre máquina en robótica y las diferentes clasificaciones de los lenguajes de programación.

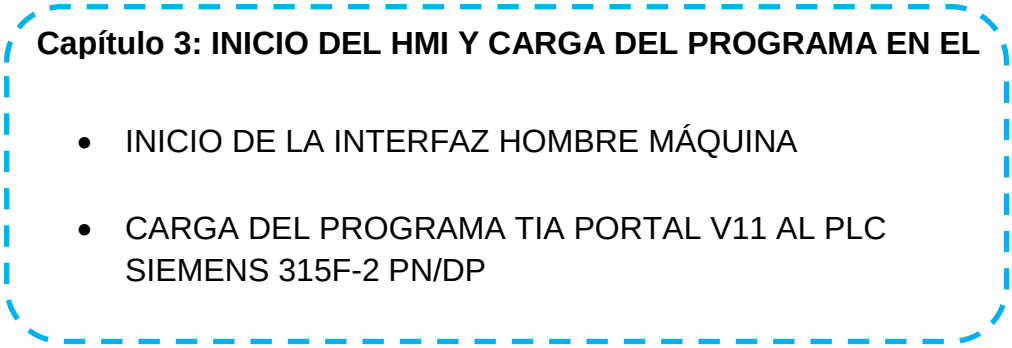
Figura 96. Capítulo 2 manual robot neumático cartesiano



7.3 MANUAL DEL USUARIO: CAPÍTULO 3

En este capítulo se muestra el paso a paso del inicio de la interfaz de usuario en el programa TOTALLY INTEGRATED AUTOMATION PORTAL V11 de Siemens que se ejecuta en el runtime SIMATIC WinCC RT y la carga del programa en el PLC 315F-2 PN/DP.

Figura 97. Capítulo 3 manual robot neumático cartesiano



7.4 MANUAL DEL USUARIO: CAPÍTULO 4

Se presentan unas prácticas para el aprendizaje de los lenguajes de programación en robótica, estas prácticas están desarrolladas con el propósito que el estudiante conozca cada uno de los paneles de operación, campos de entrada y salida de datos, gráficos y botones con los que cuentas las tres interfaces.

En la práctica 1 se programa la búsqueda de un elemento en el área de trabajo del robot para ser llevado a un lugar especificado en la descripción de la actividad.

En la práctica 2 se plantea grabar una rutina de movimiento de algunos cubos para luego verificar que los movimientos hechos anteriormente concuerdan con la grabación y mostrar al estudiante la practicidad de este tipo de programación en la robótica.

En la práctica 3 se realiza la codificación de un programa que muestra todas las funciones que tiene el lenguaje de programación textual y los bucles If, For y While también considerando la programación anidada. De igual forma, se muestran las instrucciones definidas para el robot neumático cartesiano, en cada paso se muestra un mensaje para que se pueda observar cada uno de los procesos que se llevan a cabo.

Figura 98. Capítulo 4 manual robot neumático cartesiano

Capítulo 4: PRÁCTICAS PARA EL APRENDIZAJE DE LOS LENGUAJES DE PROGRAMACIÓN EN SISTEMAS ROBÓTICOS

- **PRÁCTICA 1:** MODO DE PROGRAMACIÓN INTELIGENTE (sistema de programación textual enfocado a la tarea)
- **PRÁCTICA 2:** MODO DE PROGRAMACIÓN GUIADA (Activa)
- **PRÁCTICA 3:** MODO DE PROGRAMACIÓN POR MEDIO DEL EDITOR TEXTUAL (Lenguaje de programación textual enfocada al robot)

8. CONCLUSIONES

Con la implementación de un modelo predictivo de control para la posición en sistemas neumáticos, basado en la velocidad promedio, recorrido final y el tiempo de arranque obtenidos de la curva desplazamiento - tiempo del elemento deslizante de un cilindro neumático para las diferentes aperturas de los puertos en la válvula proporcional que controla el fluido que llega al actuador; se crea una forma válida para el movimiento de un sistema neumático compuesto por válvulas proporcionales como elementos de preactuación y un conjunto de cilindros neumáticos sin vástago o con vástago pasante como actuadores. Lo anterior, brinda una nueva y más practica alternativa sobre los métodos de control para elementos neumáticos, aplicable a los sistemas fundamentados en el uso de autómatas programables industriales, micro controladores o cualquier otro sistema que tenga la capacidad de ser programado y posea un conjunto de módulos especializados en la lectura de entradas digitales/analógicas, además de la capacidad de modificar salidas digitales/analógicas.

Con la creación de tres sistemas de programación para el robot neumático cartesiano del laboratorio de Automatización Industrial de la Escuela de Ingeniería Mecánica, basados en los requerimientos presentados en la bibliografía publicada sobre sistemas robóticos industriales, se permite un modelo válido para que los estudiantes realicen prácticas sobre un sistema real para el aprendizaje de los lenguajes de programación de robots; permitiendo complementar la formación del profesional y brindarle la posibilidad de obtener un mejor desenvolvimiento en el campo de las líneas automatizadas de producción.

Con el desarrollo para los lenguajes de programación en robótica de una serie de interfaces Hombre – Máquina, aplicadas en un computador y comunicadas al autómata programable, se logró dotar al robot neumático cartesiano con tres modos de interfaces de alto nivel.

Con la inclusión de un joystick para el manejo del robot cartesiano neumático, se logra adaptar una nueva interfaz al sistema, pues puede ser usado dentro de la programación de un código escrito en el lenguaje textual nivel robot.

Con la adición de una Cámara Web a las interfaces del robot cartesiano, se implementa un sistema de monitoreo remoto, que permite controlar al robot sin la necesidad de despegarse de la interfaz de trabajo presentada en el computador.

De acuerdo a lo anterior, se cumple con todos los objetivos propuestos para la realización del presente proyecto de grado, logrando con esto un aporte de bajo costo para el aprendizaje de sistemas robóticos.

9. RECOMENDACIONES

Se recomienda referirse al “MANUAL DEL USUARIO” creado para el robot neumático cartesiano, antes de hacer uso del mismo.

De otra parte, se recomienda la inclusión de nuevas herramientas para el robot neumático cartesiano tales como pinzas u otros elementos que le permitan al robot nuevas formas de realizar sus tareas.

Entretanto, se invita a la implementación de nuevos sensores tales como caudalímetros y sensores de presión con miras a realizar un proyecto de investigación enfocado a la generación de trayectorias en robots basados en sistemas neumáticos.

Finalmente, se propone la inclusión de sensores de visión para el efector final con el fin de crear una forma de reconocimiento de alto nivel para el estado de los elementos del robot, además del cambio de los módulos y CPU del controlador por series más potentes, con el objetivo de establecer modelos de inteligencia artificial en el transporte de elementos en robot neumáticos.

BIBLIOGRAFÍA

BARRIENTOS, Antonio et al. Introducción. En: Fundamento de Robótica. Segunda edición. Madrid: Concepción Fernández Madrid, 1997. p. 15-48.

---------. Programación de Robots. En: Fundamento de Robótica. Segunda edición. Madrid: Concepción Fernández Madrid, 1997. p. 219-254.

BUITRAGO, Jhon Edisson y CEDIEL Nelson Gustavo. Robot Cartesiano Neumático para el Laboratorio de Sistemas Mecatrónicos de la Escuela de Ingeniería Mecánica. Diseño y Construcción. Trabajo de grado Ingeniero Mecánico. Bucaramanga. Universidad Industrial De Santander. 2010. 164 p.

DAVID, Roberto Fabio y LIZARAZO Rubén Darío. Guía de estudio de los modos de marchas y paradas (Gemma) para el Manipulador Cartesiano del Laboratorio de Automatización Industrial: Diseño y Construcción. Trabajo de grado Ingeniero Mecánico. Bucaramanga. Universidad Industrial De Santander. 2012. 155 p.

MESEGUER, Alejandro R. Sintonía experimental de controladores. En: Montaje, programación y puesta en marcha de un robot neumático de escritura paralela. Trabajo de grado Ingeniero de Telecomunicaciones. Cartagena. Universidad Politécnica de Cartagena. 2008. 30 p.

NOURI, Bashir M, et al. Modelling a pneumatic servo positioning system with friction. En Proceedings American Control Conference, Junio de 2000. Vol. 2. p. 1067-1071.

SIEMENS AG. System Manual: SIMATIC HMI, WinCC V7.0 SP1 MDM - WinCC: Scripting (VBS, ANSI-C, VBA). Industry Sector. NÜRNBERG. Noviembre de 2008. 2532 p. [En línea]. http://support.automation.siemens.com/WW/llisapi.dll?func=cslib.csinfo&objId=375

72697&nodeid0=10805591&load=content&lang=en&siteid=cseus&aktprim=0&objaction=csview&extranet=standard&viewreg=WW

ANEXOS

ANEXO A. DEFINICIÓN DE ROBOT INDUSTRIAL

En esta sección se muestran los diferentes conceptos sobre lo que define a un robot industrial según diferentes organizaciones, las definiciones son tomadas del libro “FUNDAMENTOS DE ROBÓTICA”[10]:

Para la Asociación Francesa de Normalización (AFNOR), un robot industrial se define como un “*Manipulador automático servocontrolado, reprogramable, polivalente, capaz de posicionar y orientar piezas, útiles o dispositivos especiales, siguiendo trayectorias variables reprogramables, para la ejecución de tareas variadas. Normalmente tiene la forma de uno o varios brazos terminados en una muñeca. Su unidad de control incluye un dispositivo de memoria y ocasionalmente de percepción del entorno. Normalmente su uso es el de realizar una tarea de manera cíclica, pudiéndose adaptar a otra sin cambios permanentes en su material*”.

Para la Federación Internacional de Robótica, “*por robot industrial de manipulación se entiende a una máquina de manipulación automática, reprogramables y multifuncional con tres ejes o más que pueden posicionar y orientar materias, piezas, herramientas o dispositivos especiales para la ejecución de trabajos diversos en las diferentes etapas de la producción industrial, ya sea en una posición fija o en movimiento*”.

[10] BARRIENTOS, Antonio et al. Introducción. En: Fundamento de Robótica. Segunda edición. Madrid: Concepción Fernández Madrid, 1997. Pág. 220.

ANEXO B. LOS LENGUAJES DE PROGRAMACIÓN EN ROBOTS

La información obtenida en el presente anexo es tomada del capítulo “PROGRAMACIÓN DE ROBOTS”, del libro “FUNDAMENTOS DE ROBÓTICA”.

Dentro de las líneas de producción automatizadas se requieren equipos que tengan la capacidad para adaptarse al cambio en las tareas exigidas por el proceso. Una respuesta práctica a esta necesidad nace con el uso de robots industriales, cuya característica principal es la de ser máquinas “flexibles” es decir que pueden ser programadas, lo cual permite que se adecuen al sistema de producción de una forma más eficiente.

Programar un robot industrial consiste en introducir dentro de su información de trabajo una serie de acciones que deberán ser ejecutadas durante el desarrollo de su parte en el proceso de producción. Según el tipo de robot la forma como se realiza este traspaso de información puede verse modificada, por esta razón cuando se mencionan los sistemas de programación de robots se referencian como lenguajes de programación.

Un lenguaje de programación para robots es aquel conjunto de reglas y acciones que le permiten al usuario comunicar las instrucciones que desea que el sistema realice, permitiéndole explotar todas las características y posibilidades del robot industrial.

ANEXO C. ESPECIFICACIONES DE LAS ESTRUCTURAS DE LOS LENGUAJES DE PROGRAMACIÓN

Los lenguajes de programación de robots en la actualidad no obedecen a una norma o estándar estipulado legalmente, pero debido a las necesidades comunes que surgen al pretender establecer un enlace entre el usuario y el robot, se presentan algunas características que pueden ayudar a definir de forma general a los lenguajes. Dichas propiedades son especificadas en los textos sobre programación de sistema robóticos, para efectos de este proyecto se toman como referencia los requerimientos nombrados en la sección “REQUERIMIENTOS DE UN SISTEMA DE PROGRAMACIÓN DE ROBOTS” en el capítulo “PROGRAMACIÓN DE ROBOTS” del libro “FUNDAMENTOS DE ROBÓTICA”.

La naturaleza de la proporción de los requerimientos para un sistema de programación de robots puede variar según el lenguaje, ya que en algunos casos pude darse importancia a unos aspectos sobre otros o no presentarse todas las características.

Los requerimientos presentes son los siguientes:

- ***Entorno de programación:*** Un entorno de programación debe realizarse en pro de las necesidades que presenta cada lenguaje, por esta razón una de las características fundamentales en los sistemas de programación es la de ser de tipo interpretado, lo cual permite una verificación del paso a paso de las instrucciones, permitiendo al usuario la programación del robot de una forma eficiente y sencilla.

- ***Modelado del entorno:*** El modelado del entorno se entiende como la representación que hace el robot de los elementos sobre los cuales va a operar,

ya sea por medio de las características propias del objeto (masa, volumen, entre otras) o por las relativas al espacio de trabajo (posición, orientación del objeto).

Algunos de los modelos permiten crear relaciones entre los artículos de trabajo, las cuales se pueden clasificar en: Unión rígida, si el movimiento de uno de los objetos implica el del otro. Independientes, si al mover un componente no afecta de forma directa al otro. Unión no rígida, el movimiento de un elemento implica el del otro pero no se da en el sentido contrario.

- ***Tipo de datos:*** Un lenguaje de programación para robots debe contar con un sistema de variables que le permiten el trabajo y el uso de las instrucciones creadas dentro de sus procesos (tales como valores booleanos, enteros, reales, entre otros), entre las variables desarrolladas debe presentarse una clase en especial que permita la representación de la posición según el sistema de coordenadas usado en el sistema robótico.

- ***Manejo de entradas y salidas:*** La inclusión de elementos o aplicaciones que le permitan al sistema robótico acciones externas a su proceso normal le permiten la adecuación correcta en una planta de trabajo, tales como sensores con funciones especiales, sistemas de comunicación con otros dispositivos o aplicaciones que permitan una vigilancia remota del sistema robótico.

Además se debe garantizar que sus sensores internos ya sean de tipo digital o analógico se encuentren disponibles para uso externo del usuario fuera de los lazos internos de control propios del robot.

- ***Control del movimiento del robot:*** En los sistemas de programación de robots es indispensable el desarrollo de instrucciones o comandos que permitan el movimiento del robot en función de una serie de parámetros determinados por el usuario tales como la posición final del movimiento, el tiempo de duración del movimiento y la velocidad del robot. Los

parámetros introducidos por el usuario van acorde al sistema de control del robot industrial por lo cual estos pueden ser diferentes según la morfología del mismo.

- ***Control del flujo de ejecución del programa:*** En un lenguaje de programación en especial aquellos que usen editores textuales como medio para dar las instrucciones de trabajo al robot, requieren de una serie de bucles que permitan modificar el flujo de ejecución del programa, como ejemplos de estos ciclos se pueden tratar las estructuras If, For y While usadas en lenguajes de programación para computadoras.

ANEXO D. CLASIFICACIÓN DE LOS LENGUAJES DE PROGRAMACIÓN DE ROBOTS

En esta sección se dan a conocer los diferentes tipos de lenguajes de programación para sistemas robóticos especificados en el capítulo “PROGRAMACIÓN DE ROBOTS”, del libro “FUNDAMENTOS DE ROBÓTICA” enfocando la información en las tres clases seleccionadas para la solución del proyecto de grado.

- **Programación por medio de un sistema guiado activo**

Cuando se habla de un sistema guiado, es aquel donde el usuario tiene la posibilidad de mover el robot con una mayor libertad, cada uno de estos movimientos son grabados por el autómata para ser repetidos posteriormente de forma automática.

Existen dos tipos de sistemas guiados:

El sistema guiado pasivo, este tipo de configuración se presenta en dos formas, en primer lugar si los actuadores del robot se encuentran des energizados y el operario puede mover el efector final del robot a los puntos deseados con total libertad donde cada uno de estos movimientos es memorizado por el autómata para su posterior reproducción, se habla de un sistema guiado pasivo directo, en la segunda forma de trabajo el usuario no mueve el robot sino a una maqueta o representación a escala del mismo, lo cual facilita el trabajo para el operador, en este paso se habla de un sistema guiado pasivo por medio de maniquí.

El enfoque de la solución del proyecto va dirigida a otro tipo de sistema de programación guiado, la configuración física del robot no permite el uso de una programación del tipo pasivo ya que el sistema no habilita el movimiento del efector final si sus actuadores no son actuados, por esta razón se elige que el usuario pueda controlar los movimientos del robot por medio de un sistema de

mando (joystick) el cual modificara de forma indirecta la señal de actuación que llega a los actuadores provocando que el efector se mueva en la dirección y a la velocidad deseada por el operario, cada uno de los movimientos es almacenado por el autómata y luego son reproducidos de forma automática, a esta manera de programación se le conoce como sistema guiado activo.

- **Programación textual nivel robot**

Cuando se habla de un lenguaje de programación textual nivel robot se entiende que las instrucciones dadas para controlarlo se dan por medio de un lenguaje escrito, en el cual las palabras o caracteres describen una función específica de trabajo.

Este tipo de lenguaje se enfoca a las actividades propias que debe realizar el robot para cumplir con las subtareas que desea el usuario, como por ejemplo si desea sujetar un elemento, el operador debe especificar los movimientos necesarios para hacerlo además de la instrucción de succionar al encontrar el elemento debajo del efector final.

- **Programación textual nivel tarea**

En este tipo de lenguaje el programa se reduce a una sola sentencia la cual le especifica al robot que debe hacer, el sistema robótico por definición tiene configurados todos los procedimientos que debe efectuar para realizar la tarea.

Para el caso específico de este proyecto el lenguaje de programación se enfoca en tres tareas: MOVER, SUJETAR ELEMENTO, MOVER Y SOLTAR.

ANEXO E. SISTEMAS DE CONTROL

El siguiente anexo toma los contenidos presentados para el regulador todo – nada y el controlador PID en el blog TIPOS DE CONTROL[11].

- Regulador Todo - Nada

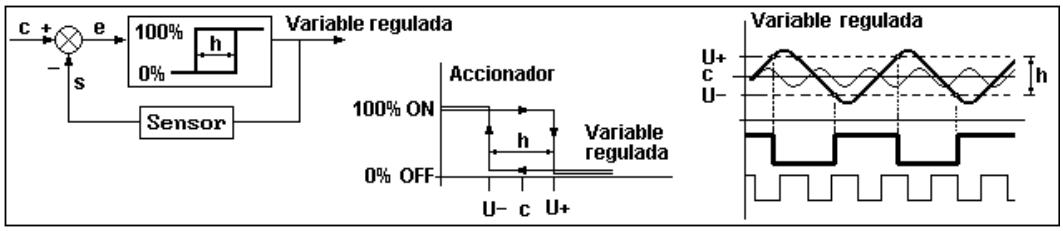
Es la regulación más simple y económica, interesante en numerosas aplicaciones en las que puede admitirse una oscilación continua entre dos límites, siempre y cuando se trate de procesos de evolución lenta. Como ejemplos podemos citar la regulación de nivel, de presión o de temperatura, todos ellos problemas relativamente sencillos de lógica digital que no tratamos en este tema. Numerosos reguladores incorporan esta función básica, que además ofrece la máxima rapidez de respuesta y en ocasiones se recurre a este tipo de control cuando el error es grande, y se pasa de forma automática a otro tipo de regulación cuando el error se aproxima a cero.

En la siguiente figura se puede ver un diagrama de bloques y una representación de su funcionamiento: Gracias a la existencia de una histéresis (h), el número de conmutaciones se reduce notablemente. Sin histéresis, el accionador se activaría y desactivaría con demasiada frecuencia (gráfica con línea fina). La histéresis es como una oposición a experimentar cualquier cambio y generalmente será un efecto perjudicial, por ejemplo, al descender una temperatura después de haber alcanzado un máximo, el sensor pudiera mantener el mismo nivel de señal hasta que la temperatura real descienda más de 8 grados, por ejemplo. Sin embargo, este efecto no es perjudicial en el tipo de regulador que tratamos: Su respuesta es de tipo todo-nada, de forma que se conecta cuando la variable regulada ha descendido hasta un valor (-U) por debajo del punto de consigna "c" y solo se

[11] http://www.oocities.org/es/jeeesusmeeerino/procesos/tipos/tipos.html

desconecta cuando llega a otro valor (+U) por encima del punto de consigna. Así se establece un margen de variación en el que mantiene su estado el actuador.

Figura 99. Regulador todo o nada

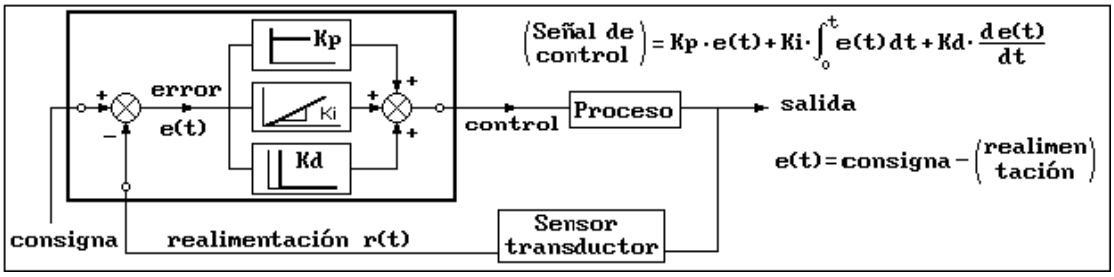


Fuente: blog TIPOS DE CONTROL

- Regulador PID

Un regulador proporcional-integral-derivativo o PID tiene en cuenta el error, la integral del error y la derivada del error. La acción de control se calcula multiplicando los tres valores por una constante y sumando los resultados. Los valores de las constantes, que reciben el nombre de constante proporcional, integral y derivativa, definen el comportamiento del regulador.

Figura 100. Regulador PID



Fuente: blog TIPOS DE CONTROL

ANEXO F. COMPONENTES DEL MANIPULADOR Y ROBOT NEUMÁTICOS CARTESIANOS

El robot neumático cartesiano tiene como función principal, permitirle al usuario el transporte de piezas por medio de tres lenguajes de programación en sistemas robóticos. Para lograr esto, la parte operativa del sistema está configurada en base a dos tareas fundamentales en las cuales se puede dividir la actividad principal. La primera de ellas consiste en la sujeción de piezas por medio de una ventosa de succión; la segunda trata sobre el movimiento del efector final (ventosa de succión), el cual, se encuentra restringido a tres grados de libertad traslacionales cuyos ejes forman un sistema de coordenadas ortogonales.

A continuación se describen los procesos y componentes que permiten a la parte operativa del robot neumático, la correcta ejecución de su función principal.

- **Compresor neumático SULLAIR**

Debido a que es un robot neumático, su fuente de alimentación debe proveer de aire comprimido a todos los actuadores para que estos funcionen correctamente, para suplir esta necesidad el robot neumático cartesiano cuenta con un compresor de aire de tornillo rotativo SULLAIR modelo ES-6 10 H cuyos datos técnicos son:

Potencia de trabajo: 10 HP
Presión máxima de trabajo: 125 psi
Presión mínima de trabajo: 80 psi
Alimentación máxima de caudal: 1080 l/min
Temperatura de salida del aire: [60°C - 80°C]
Tipo de control: ON-OFF

Este aire es llevado hasta el robot neumático cartesiano por medio de tuberías y mangueras neumáticas con una llave de globo para controlar el paso de aire a la unidad de mantenimiento que prepara el aire para ser usado en los cilindros.

Figura 101. Compresor SULLAIR ES - 6 10H

Fuente: SULLAIR, Modelo: ES-6 S-energy® (5–10hp•4–7kW).

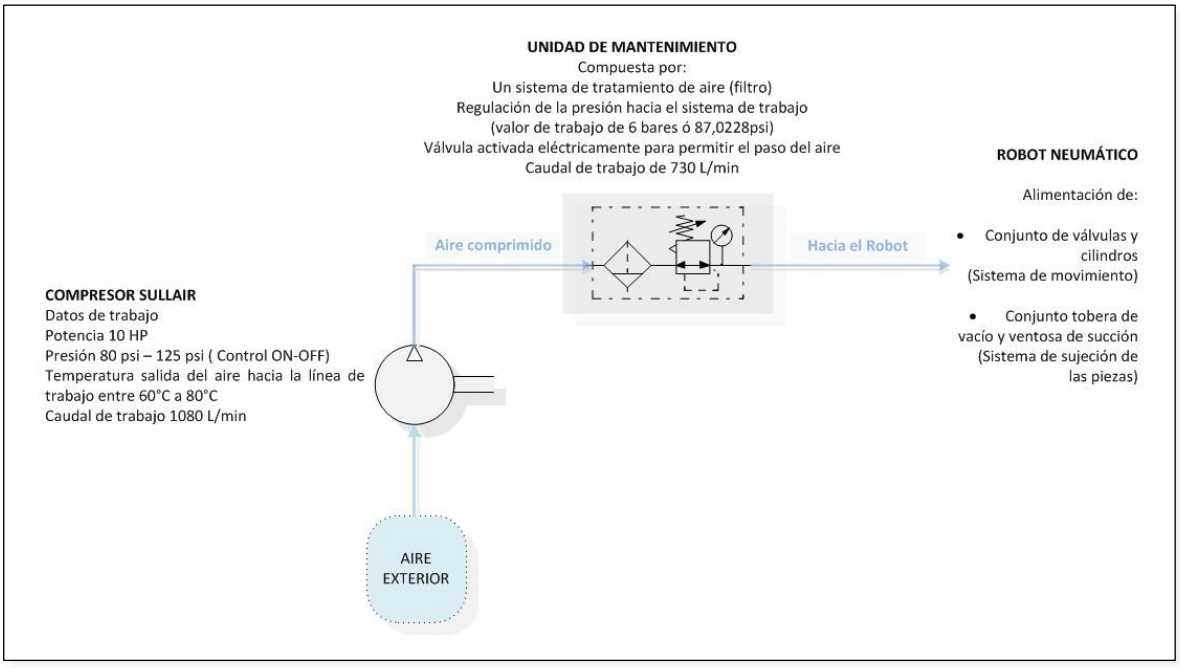
- **Unidad de mantenimiento**

El ingreso del aire desde el compresor al sistema robótico se hace por medio de una unidad de mantenimiento cuya referencia es la siguiente: LFR-1/4-D-MINI-KD de la empresa FESTO®, dentro de ella vienen incluidos diferentes elementos para el tratamiento del aire (secador de membrana, filtro), regulación de la presión a la salida de la unidad (el objetivo de la regulación es el de mantener en un valor constante en la presión de suministro del sistema para los diferentes elementos neumáticos) y el paso del aire hacia el sistema (cuenta con una válvula accionada eléctricamente que al activarse permite la salida del flujo del aire hacia el sistema neumático). Los parámetros de trabajo para los elementos que conforman el robot neumático gracias al uso de esta unidad se establecen en los siguientes valores: presión de suministro de 6 bares (87,0228 psi), con un caudal de 730 l/min (dato obtenido del catálogo suministrado por el proveedor).

Figura 102. Unidad de mantenimiento

Fuente: Catálogo de Combinaciones de unidades de mantenimiento LFR-K/LFRS-K serie D, FESTO®

Figura 103. Esquema general: tratamiento y generación aire comprimido



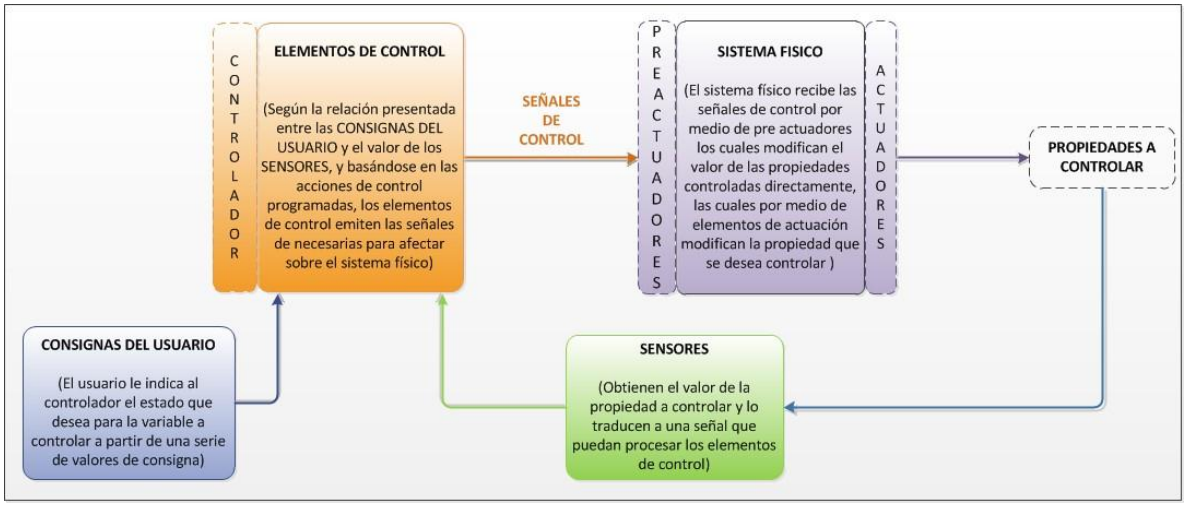
- Lazos de control:

Con el objetivo de cumplir las dos tareas fundamentales de trabajo, se establecen una serie de lazos de control según el tipo de lenguaje de programación por medio del cual son dirigidas las instrucciones al sistema robótico.

El objetivo principal de un sistema de control es la modificación de algunas propiedades del sistema para “controlar de forma indirecta” la magnitud o valor de otra.

La distribución y operación de un sistema de control se puede resumir en la Figura 104. Sistema de control (Distribución y operación).

Figura 104. Sistema de control (Distribución y operación)



➢ Descripción general de los elementos presentes en los sistemas de control del robot neumático cartesiano.

- **Elementos de control**:

Para realizar el control de las actividades del robot neumático cartesiano se usa un autómata programable industrial, en este caso se presenta un modelo de la empresa SIEMENS®, gama media alta S7-300.

Cuando se usa un sistema basado en un autómata programable, se tienen los siguientes elementos: Una regulación de voltaje, una Unidad Central de Proceso (CPU), Módulos según las necesidades del usuario o sistema, en este caso se cuenta con dos paquetes de 16 entradas/ 16 salidas digitales cada uno, un módulo de regulación PID y uno con cuatro salidas analógicas.

Debido a que el controlador es un dispositivo electrónico, su potencia de trabajo es muy baja, por este motivo posee una fuente de regulación que transforma el potencial de 125 Voltios de corriente alterna de la línea eléctrica del laboratorio a un potencial de 24 voltios de corriente directa. Para los modelos S7-300, la empresa de SIEMENS® ha desarrollado una serie de reguladores conocidos con la nomenclatura PS, para el robot neumático la regulación del voltaje se hace con el modelo PS 307 de 5 amperios. Este regulador alimenta no solo al autómata sino a los módulos de trabajo, los sensores digitales y los actuadores del robot neumático.

Figura 105. Regulador de voltaje para el autómata (PS 307)

Fuente: Siemens®

El autómata programable posee una unidad de proceso conocida como CPU, es la parte principal del controlador ya que a partir de los datos obtenidos por los módulos (los cuales recogen la información de los sensores) y las decisiones que

le han sido programadas por el usuario (por medio de instrucciones ya sean de tipo lógico usadas en sistemas de control para eventos discretos o en lenguaje de alto nivel dadas en sistemas de control cuasi continuos de datos muestreados), realiza las acciones necesarias para actuar sobre los pre actuadores (se envía la señal de control a los pre actuadores por medio de los módulos adicionados al autómata programable) modificando de esta manera al sistema físico y por ende a la magnitud de la propiedad controlada.

La CPU usada en el robot neumático cartesiano pertenece a los modelos gama media alta ofrecidos por la empresa SIEMENS®, la referencia es CPU 315F-2 PN/DP versión V3.2, cuyas características de trabajo son las siguientes: Memoria de trabajo de 512 Kb; 0,05 ms/1000 instrucciones; comunicación vía Ethernet. Se escoge debido a la capacidad de memoria ya que es la suficiente para almacenar los bloques de programación que contienen el código necesario para la ejecución de los lazos de control de los diferentes modos de trabajo y la forma de comunicación es la más rápida ofrecida en estos modelos.

Figura 106. CPU 315F - 2 PN/DP

Fuente: Siemens®

Los módulos adicionados al autómata programable para controlar el robot neumático son los siguientes:

Dos módulos de 16 entradas y 16 salidas digitales cada uno, cuyo potencial de trabajo en cada uno de los puertos (16 entrada / 16 salida) es de 24 Voltios de corriente directa, ofrecido por la empresa SIEMENS® correspondiente a la gama media alta S7 300.

Figura 107. Módulo de 16 entradas y 16 salidas digitales

Fuente: Yocon Technology Co.,ltd. Modelo: 6es7 323 - 1bl00 - 0aa0: sm323 16 di 24v dc/16do 24v 0.5a dc.

Cuando se hace referencia a una entrada digital, se trata que por medio de ella se pueda obtener la información transmitida por sensores del tipo todo o nada, como finales de carrera, pulsadores, entre otros.

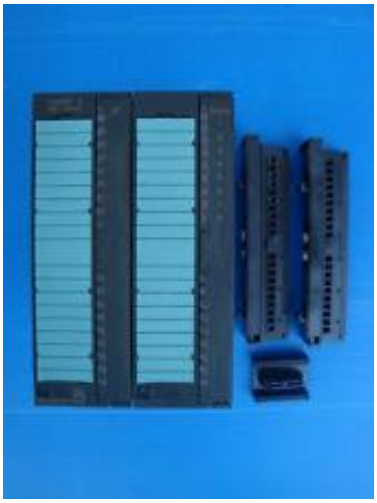
Debido a que este módulo trabaja con entradas digitales de tensión, se tiene que cuando una señal de 24 voltios es recibida por uno de sus puertos se entenderá con el valor de “1” lo cual nos dice que se encuentra activo el sensor si es del tipo normalmente abierto o se encuentra desactivado si el del tipo normalmente cerrado.

Una salida digital, consiste en que por medio de ella se pueda controlar los pre actuadores o actuadores del tipo todo o nada, como relés, solenoides de válvulas neumáticas y demás.

Debido a que este módulo trabaja con salidas digitales, cuando el sistema obtiene como resultado la activación de una salida, genera una diferencia de potencial de 24 Voltios de corriente directa con el fin de energizar el elemento conectado a ella, por el contrario si la orden es de desactivación, la diferencia de potencial es de 0 Voltios generando que se des energice el dispositivo acoplado a ella.

Un módulo de regulación PID modelo FM 355C y otro que consta de 4 salidas del tipo analógico cuya resolución es de 12 bits modelo SM 332 son usados dentro de los lazos de control cuasi continuos realizados dentro del API (por sus siglas en español para autómata programable industrial), ambos son ofrecidos por la empresa SIEMENS® para la gama media alta S7 300.

Figura 108. Módulo de regulación PID FM 355C



Fuente: SIEMENS, SIMATIC PID CONTROL FM355 C 6ES7 355-0VH10-0AE0 + 3 CONNECTORS.

Figura 109. Módulo salidas analógicas SM 332

Fuente: Siemens, Módulos de salidas analógicas SM 332. 6ES7332-5HD01-0AB0.

El modelo FM 355C consta con cuatro canales de lectura analógicos, cada uno con la posibilidad de tener su propio lazo de control integrado dentro del módulo, y cuenta con cuatro canales de salida analógicos.

La entrada analógica permite la lectura de variables físicas continuas, debido a que el controlador en esencia es un sistema digital, la lectura de este tipo de magnitudes se hace de forma muestreada, es decir, que a partir de una unidad mínima de lectura, se toma un límite máximo y mínimo, la separación entre ellos se divide por esta mínima unidad de tal manera que el sistema queda escalado a un número finito de partes.

La salida analógica permite que un valor o resultado obtenido por el autómata programable se convierta en una señal de tensión o intensidad que puede llegar a un pre actuador o actuador, de la misma manera que en las entradas analógicas esta señal emitida tiene una unidad mínima de trabajo por lo cual la magnitud de salida se da como función de los múltiplos de la división más pequeña de trabajo.

- **Sistema Físico:**

Como sistema físico se considera la parte neumática presente en los diferentes lazos de control del sistema robótico, en este caso contamos con los siguientes elementos que se describen a continuación.

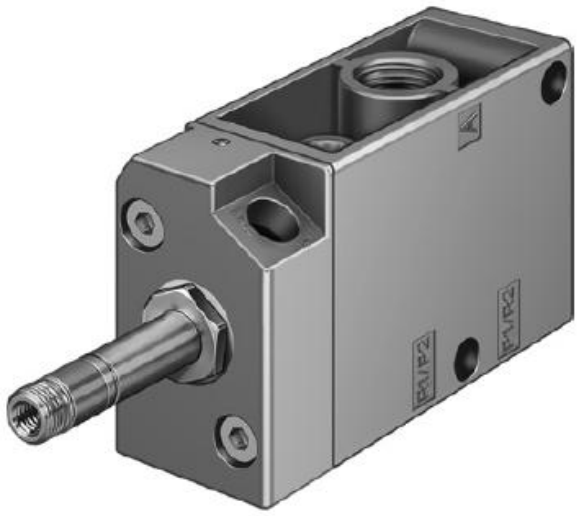
- **PRE-ACTUADORES:**

Como el fluido de trabajo es aire comprimido, pero debido a que se usa un sistema con un autómata programable como controlador, cuyas señales son de tensión e intensidad eléctrica, para él es imposible modificar de forma directa las propiedades del gas, por tanto para suplir esta necesidad, el robot neumático cuenta con elementos capaces de traducir las señales enviadas por el API en cambios de las propiedades del fluido de trabajo.

Electroválvula MFH-3-1/4:

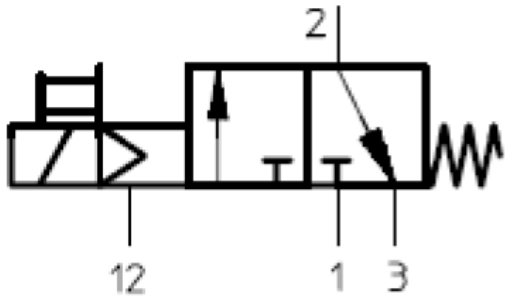
Es una válvula de 3/2 vías para abrir o cerrar el paso de aire en la vía de succión.

Figura 110. Electroválvula MFH - 3 - 1/4



Fuente: Festo, MFH-3-1/4 Festo Solenoid valve.

Figura 111. Interior conexiones electroválvula MFH - 3 - 1/4



Fuente: Festo, Pneumatic Schematic.

Descripción general de los lazos de control del robot neumático cartesiano.

Las dos tareas fundamentales que conforman la función principal del robot neumático cartesiano son:

Sujeción de piezas:

El controlador envía una señal de activación a una salida digital con el fin de energizar el solenoide de la electroválvula MFH-3-1/4 acoplado a ella, esto permite que la posición de la válvula conmute haciendo que el aire comprimido llegue a la tobera aspiradora de vacío, la cual genera una presión de vacío en la ventosa con el fin de crear un sistema que puede succionar elementos.

El vacuostato ayuda en el conocimiento de la sujeción de los elementos de la siguiente manera: al encontrarse interconectado entre la tobera de vacío y la ventosa de succión, le permite conocer cuándo un elemento es sujetado por el sistema. Si un elemento es atrapado por la ventosa de succión el sensor envía una señal de activación a la entrada digital del autómata programable a la cual se encuentra acoplado, en caso de que nada se encuentre sujeto por la ventosa no envía señal alguna.

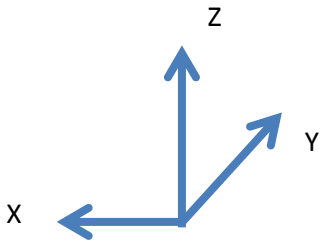
La señal de activación de la salida es impartida por el usuario en su programación y no depende de la señal dada por el vacuostato.

Movimiento en el espacio tridimensional:

El objetivo de este lazo de control es la posición del efector final (ventosa), especificada a partir de tres ejes ortogonales. La posición en cada uno de los ejes es independiente ya que cada una es controlada por un actuador diferente.

Los ejes de trabajo se establecen de la forma como lo indica la Figura 112.

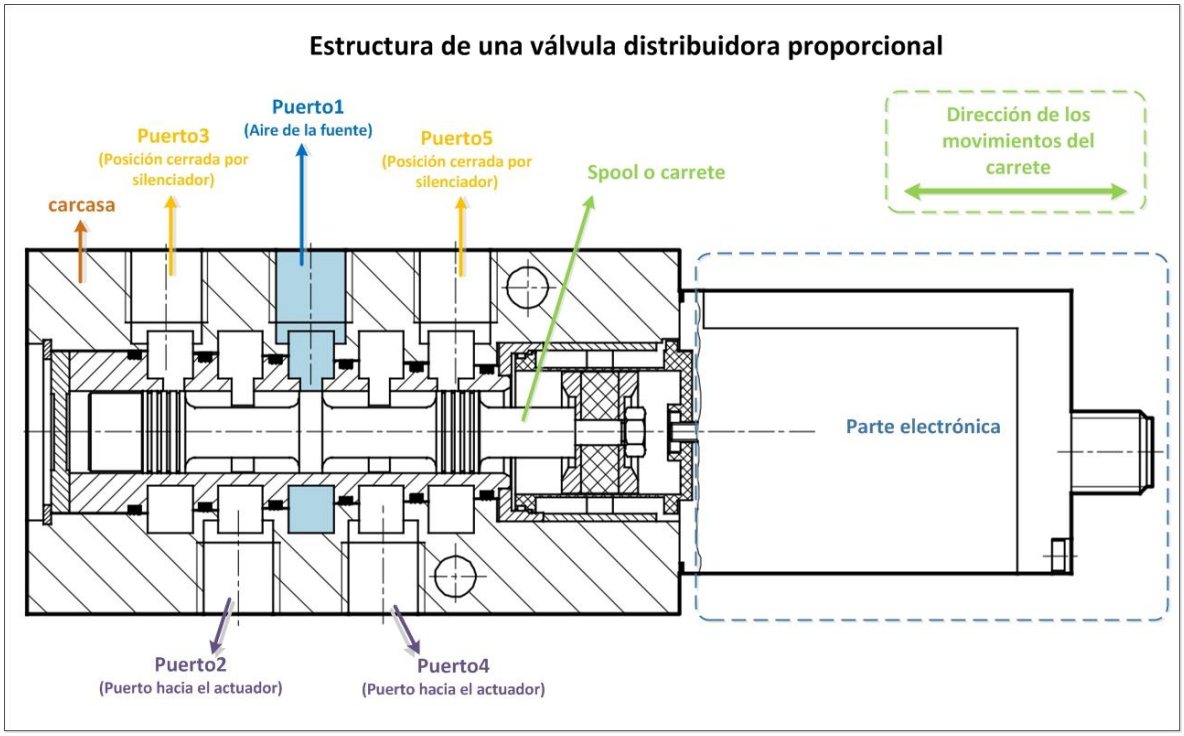
Figura 112. Ubicación de los ejes en el robot neumático cartesiano



La posición en los **ejes X** y **Y** es controlada en cada uno por medio de un cilindro de doble efecto sin vástago modelo **DGPL-25-500-PPV-A-B-KF**, mientras que para el **eje Z** depende de un cilindro de doble efecto con vástago pasante tipo **DNCM-32-200-POT1-S20-A,** los dos modelos corresponden a productos de la empresa FESTO®.

El caudal que llega a cada uno de los actuadores es controlado por medio de **tres válvulas distribuidoras proporcionales** modelo **MPYE-5-1/4-420-B** (Ofrecidas por la empresa FESTO®), una para cada eje.

Figura 113. Estructura de una válvula distribuidora proporcional



Fuente: FESTO®

La Figura 113 ilustra el interior de la válvula proporcional, el funcionamiento del elemento consiste en modificar la posición del carrete en función de la señal de intensidad eléctrica que llega al sistema electrónico, lo cual permite modificar el área de apertura entre el puerto de alimentación y los puertos que van hacia el actuador.

La forma de trabajo de la válvula se puede describir de la siguiente manera:

El rango de intensidad en la forma de trabajo va desde 4 mA hasta 20 mA, esto quiere decir que el rango intermedio se encuentra en los 12 mA.

Cuando el controlador (API) a partir de una de sus salidas analógicas envía una señal en el rango entre 12 mA a 20 mA, la relación de apertura entre el puerto que suple de aire (puerto 1) y el puerto 4 que va hacia el actuador se va incrementando conforme tiende al valor de 20 mA, llenando de esta manera una

de las cámaras del actuador, en contra posición el puerto 2 que trae el aire de la cámara opuesta del actuador, lo lleva al medio exterior a través del silenciador acoplado en el puerto 3.

Cuando la señal enviada desde el controlador se encuentra en el rango de los 4 mA a 12 mA, la relación en la apertura entre el puerto 1 y el puerto 2 que va hacia el actuador se incrementa conforme se aleja de 12 mA y se acerca al valor de 4 mA, de esta manera se llena una de las cámaras del actuador, en oposición el puerto 4 que trae el fluido desde la cámara opuesta, lo lleva al medio exterior por medio del silenciador acoplado en el puerto 5.

La válvula se centra cuando el controlador envía una señal de 12 mA o cercana a este valor, este rango es conocido como la zona muerta de la válvula, cuando el carrete se encuentra en esta zona la relación de apertura de la válvula se reduce a cero generando así que el actuador se detenga si se encontraba en movimiento y no se mueva.

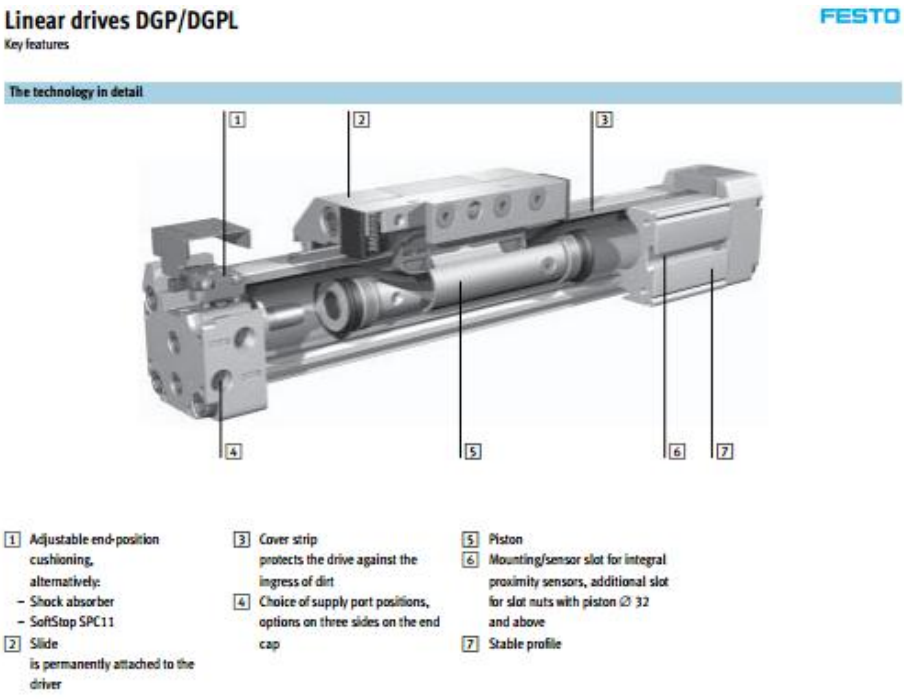
- **ACTUADORES:**

Los actuadores se encargan de realizar los movimientos del robot y la succión de los elementos a manipular, para llevar a cabo estos procesos se usan cilindros neumáticos de la empresa FESTO y una ventosa para proporcionar el agarre en el actuador final principal.

Cilindros:

Para los ejes X e Y se usan dos cilindros neumáticos **DGPL-25-500-PPV-A-B-KF.** Este cilindro cuenta con una carrera de 500 mm y un diámetro de pistón de 25mm, tiene regulador de caudal y amortiguamiento en cada uno de los extremos para evitar choques que puedan afectar la funcionalidad del elemento.

Figura 114. Cilindro neumático DGPL-25-500-PPV-A-B-KF



Fuente: Festo, DGP/DGPL rodless drive units.

Para el eje Z usa un cilindro neumático **DNCM-32-200-POT1-S20-A.** Este cilindro cuenta con una carrera de 200 mm y un diámetro de pistón de 32mm, tiene regulador de caudal y amortiguamiento en cada uno de los extremos para evitar choques que puedan afectar la funcionalidad del elemento.

Figura 115. Cilindro neumático DNCM-32-200-POT1-S20-A

Fuente: Festo, Cylinders with desplacement enconder

Ventosa:

La ventosa es el elemento final de actuación el cual tiene contacto directo con los elementos a manipular, esta va conectada al cilindro neumático del eje Z y proporciona el agarre del sistema físico:

Figura 116. Ventosa FESTO

Fuente: Festo®.

- **Sensores:**

El robot neumático cartesiano cuenta con dos tipos de sensores:

Potenciómetros:

Encoder de desplazamiento analógico para la unidad de accionamiento neumática lineal, para determinar la posición actual por medio de una señal que varía desde los 0 hasta los 10 voltios, con una carrera de medición de 500mm en los ejes X e Y y de 200mm en el eje Z.

Figura 117. Potenciómetro lineal

Fuente: Festo®.

Sensor de proximidad magnético:

Para tener un control de eventos discretos y poder determinar los límites de la zona de trabajo se usan finales de carrera de la empresa FESTO ubicados en todos los extremos del espacio de trabajo del robot neumático cartesiano.

Figura 118. Sensor de proximidad magnético FESTO

Fuente: Festo®, Sensor de proximidad.

Sensor de proximidad o presencia:

El sensor de proximidad está ubicado en el extremo del vástago del eje Z junto con la ventosa, este elemento envía una señal de presencia o ausencia al controlador el cual determina que acción se debe realizar en caso de encontrar o no algún objeto cercano al efector final del robot.

Figura 119. Sensor de proximidad o presencia

Fuente: Automatizando S.A.S. Sensor de proximidad capacitivo.

Vacuostato:

Es un switch que es activado cuando la presión que está siendo censada pasa un punto de consigna, es usado para indicar que hay un elemento que está siendo sujetado por la ventosa.

Figura 120. Vacuostato

Fuente: Festo®, Vacusotato PEN.

ANEXO G. AUTOMATAS PROGRAMABLES: CONCEPTOS BÁSICOS

Se explica la descripción y los pasos a seguir para poner en funcionamiento un autómata programable SIMATIC S7-300. Se comienza por la descripción física del autómata, como lo son su configuración: (componentes o módulos que conforman el autómata) y disposición mecánica.

CONFIGURACION

Los SIMATIC S7-300 tienen una configuración modular, es posible componerlo de forma personalizada utilizando la gama de módulos disponibles para los S7-300. Para configurar un autómata programable S7-300 y ponerlo en funcionamiento se utilizan una serie de componentes que están relacionados en la tabla 9.

Tabla 9. Descripción de componentes PLC siemens S7-300

Componente	Función	Tipos
PERFIL SOPORTE	Sirve para soportar los módulos de un S7-300 formando una fila o bastidor	483 mm 530 mm **830 mm** 2000 mm
FUENTE DE ALIMENTACION PS	Convierte la tensión de red (AC 120/230 V) en una tensión DC 24 V para el funcionamiento del S7-300 y para alimentar los circuitos de carga a DC 24 V.	**PS 307, 2 A** PS 307, 5 A PS 307, 10 A
CPU	Ejecuta el programa de usuario, alimenta con 5 V el bus posterior del S7-300, se comunica vía interface MPI, con otras Estaciones de una red MPI.	CPU 314C-2 PN/DP CPU 315-2 PN/DP CPU 317-2 PN/DP CPU 319-3 PN/DP

Fuente: Manual Siemens S7-300

Tabla 10. (Continuación)

Componente	Función	Tipos
Micro Memory Card SIMATIC	sirve como memoria de carga, y es la memoria que almacena el programa ejecutable y permite guardar una copia de seguridad	MMC max. 8MB
Módulos de señales SM	Se encargan de adaptar los diferentes niveles de señal del proceso al nivel interno del S7-300.	E digitales S digitales E/S digitales E analógicas S analógicas E/S analógicas
Módulos de función FM	Se utiliza en tareas de procesamiento de señales del proceso de tiempo crítico y altos requerimientos de memoria, por Ejemplo posicionamiento o regulación. *trae funciones integradas como por ejemplo contadores, Posicionamiento, control PID.*	FM 350-1 Contadores FM 350-2 Contadores FM 351 Posicionamiento FM 352 Cam. Control óptico FM 353 Posicionamiento motores paso a paso FM 354 Posicionamiento Servo Motores FM 355 Control lazo cerrado FM 357 Posicionamiento
Procesador de comunicaciones **CP**	Alivian a la CPU de tareas de comunicación, por ejemplo para el enlace a la red PROFIBUS-DP	CP 340 CP 341 CP 342-2 **CP 342-5** CP 343-5
SIMATIC TOP connect	Es utilizado para cablear los módulos digitales	SM 321 SM 322 SM 323

Fuente: Manual Siemens S7-300

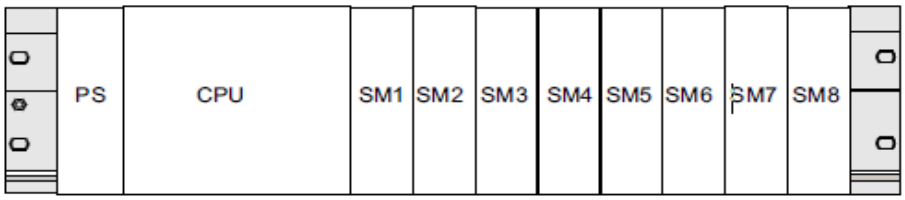
Tabla 11. (Continuación)

Componente	Función	Tipos
Interfaces **IM**	Sirve para cablear los módulos de señales de un S7-300, en varios bastidores	IM 360 IM 361 IM 365
Cable de PROFIBUS con conector a bus	Interconecta estaciones de una subred MPI o PROFIBUS-DP.	Para tendido subterráneo Para servicios móviles Con envolvente PE (Ind. Alimentos) Para tendido en girnalda
Cable PG	Enlaza un PG/PC con la CP	RS232
Repetidor RS 485	Se utiliza para amplificar las señales en una subred MPI o PROFIBUS-DP así como para acoplar segmentos de una subred MPI o PROFIBUS-DP.	
Unidad de programación PG o PC con el paquete STEP 7	Se utiliza para configurar, parametrizar, Programar y probar el S7-300.	PC/PG con Windows 95 o NT 4.0

Fuente: Manual Siemens S7-300

Los módulos del S7-300 se pueden disponer en uno o varios bastidores, la disposición en un bastidor se realiza como se muestra en la Figura 121, en ella se puede observar las direcciones o los números de SLOT donde se debe colocar cada componente

Figura 121. Disposición de los módulos en un bastidor



Fuente: Manual Siemens S-7300

A la hora de colocar los módulos en un bastidor es necesario tener en cuenta lo siguiente:

- La fuente de alimentación (PS) deberá colocarse siempre como primer módulo en la parte izquierda del perfil soporte.
- La CPU (módulo central) se colocará siempre, como segundo módulo, a la derecha de la fuente de alimentación.
- A la derecha de la CPU pueden montarse máximo 8 módulos (SM, FM, CP).
- El número de módulos (SM, FM, CP) está limitado por el consumo de corriente tomado del bus posterior.

Para la disposición de los S7-300 en varios bastidores es necesario tener en cuenta lo siguiente:

- La interface IM ocupa siempre el puesto o SLOT 3, a la izquierda del primer módulo.
- En cada bastidor puede utilizarse hasta 8 módulos (SM, FM, CP).