# ESCUELA POLITÉCNICA NACIONAL

## FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

## IMPLEMENTACIÓN DE UNA PLATAFORMA DE ESTABILIZACIÓN PARA CONTROL DE POSICIÓN Y SEGUIMIENTO DE CAMINO DE UNA ESFERA.

**PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y CONTROL**

**ANGEL LEONEL CEDEÑO NIETO**

**angel.cedeno@epn.edu.ec**

**MARCO ANTHONY GORDÓN ALMEIDA**

**marco.gordon@epn.edu.ec**

**DIRECTOR: LUIS ALBERTO MORALES ESCOBAR, MSc.**

**luis.moralesec@epn.edu.ec**

**Quito, Noviembre 2016**

# DECLARACIÓN

Nosotros, Angel Leonel Cedeño Nieto y Marco Anthony Gordón Almeida, declaramos bajo juramento que el trabajo aquí descrito es de nuestra autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que hemos consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedemos nuestros derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

_____ _____

Angel Cedeño Nieto Marco Gordón Almeida

# CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Angel Leonel Cedeño Nieto y Marco Anthony Gordón Almeida, bajo mi supervisión.

_____

MSc. Luis Morales

# AGRADECIMIENTO

A Jesús, mi baluarte y guía en cada camino que he transitado en mi vida, quien ha sido mi fortaleza en cada batalla en la que he peleado por alcanzar mis sueños.

A mi familia, que, aunque lejos, son en mi vida una fuente de inspiración y ejemplo a seguir, especialmente a mi madre, Leyda, quien siempre ha sabido guiarme por el camino correcto y por quien he llegado a ser los que soy.

A mi prometida y futura esposa, Estefy, mi mayor inspiración para luchar día a día, mi compañera de vida y mi mejor amiga.

A mis grandes amigos, Allen, Manuel, Rocío que han sido una parte muy importante en mi vida y en la consecución de esta meta.

Al MSc. Luis Morales, director de proyecto de titulación, quien ha sabido aportar con sus conocimientos y experiencias las cuales han sido muy importantes y de mucha ayuda para culminar con éxito este proyecto.

Al MSc. Marco Herrera por su gran aporte y contribución en la realización de este proyecto, por sus acertados consejos que nos guiaron en las etapas más críticas.

A la Escuela Politécnica Nacional, hogar de la ciencia y el conocimiento en donde he adquirido mucha sabiduría y experiencias que me han enriquecido profundamente.

A mis compañeros, con quienes he compartido muchas etapas de aprendizaje, y de quienes llevare siempre un grato recuerdo.

Angel

# DEDICATORIA

A Jesús, el autor y consumador de mi fe, sea el honor, porque todo lo que soy y lo que he logrado es por él.

A Leyda, mi adorada Madre, a quien llevo muy profundamente en mi corazón, a Silvana, mi hermana, y a mis bellas sobrinas Erika y Lady que han sido un bálsamo en aquellos momentos cuando seguir avanzando es tan difícil.

A mi hermosa prometida, Estefy, mi ayuda idónea en el camino que hemos elegido caminar juntos.

Angel

# AGRADECIMIENTO

Agradezco infinitamente a mis padres, Cecilia y Jorge, cuyo apoyo y esfuerzo han sido fundamentales para la consecución de este y muchos otros objetivos de vida que he decido emprender. Sus valores, paciencia y sabiduría, los he recibido como un regalo que siempre llevaré inherente a mi vida y profundamente arraigado en mi personalidad.

A la Escuela Politécnica Nacional y sus docentes, por haber sido mi segundo hogar y mis maestros de vida. Juntos, el conocimiento y la entereza se forjan día a día en sus aulas, y de ella se edifican grandes seres humanos y profesionales de excelencia.

A mi director de proyecto de titulación, MSc. Luis Morales por toda la ayuda brindada durante el desarrollo de este proyecto. Su apoyo se ve reflejado en un trabajo de calidad y de excelentes resultados.

De forma general, agradezco a todos aquellos, amigos y familiares, quienes de una u otra manera me han apoyado y han compartido parte de este camino que hoy se edifica en una meta cumplida.

Marco Anthony

# DEDICATORIA

Todo mi esfuerzo y dedicación está dirigido especialmente a mis padres, Cecilia y Jorge. Gracias por guiarme y ayudarme a conseguir mis metas. Mi infinito amor y agradecimiento es para ustedes.

A mi hermana Gaby con quien he compartido momentos muy importantes y de quien he aprendido valiosas lecciones de vida.

A Pabli, mi hermano y mejor amigo, cuyo espíritu me alienta a continuar cada día y en cuyo recuerdo siempre permanecerán los mejores momentos que he vivido.

Marco Anthony

# CONTENIDO

**RESUMEN** ..... **i**

**PRESENTACIÓN** ..... **ii**

**CAPÍTULO 1** ..... **1**

**MARCO TEÓRICO** ..... **1**

**1.1 VISIÓN ARTIFICIAL** ..... **1**

1.1.1 FUNDAMENTOS ..... 1

1.1.1.1 Definición [1] ..... 1

1.1.1.2 Representación de una imagen [1] [2] ..... 1

1.1.1.3 Etapas del proceso de visión artificial [2] ..... 2

1.1.2 ADQUISICIÓN DE IMÁGENES ..... 3

1.1.2.1 Sensor óptico [1] ..... 3

1.1.2.2 Digitalización de imágenes [2] ..... 3

1.1.3 TÉCNICAS PARA EL PROCESAMIENTO DE IMÁGENES ..... 4

1.1.3.1 Brillo y contraste [1] ..... 4

1.1.3.2 Imágenes RGB y escala de grises [1] [2] ..... 5

1.1.3.3 Binarización [1] ..... 6

1.1.3.4 Complemento de una imagen [1] ..... 6

1.1.3.5 Segmentación y extracción de propiedades [1] [2] ..... 7

1.1.3.5.1 Contornos y formas ..... 7

1.1.3.5.2 Propiedad de centroide [4] ..... 8

**1.2 FILTRO DE KALMAN [5] [6]** ..... **8**

1.2.1 FUNDAMENTOS ..... 8

1.2.2 ALGORITMO DEL FILTRO DE KALMAN ..... 9

1.2.2.1 Fase de predicción ..... 10

1.2.2.2 Fase de corrección ..... 10

**1.3 CONTROLADORES PID** ..... **11**

1.3.1 DEFINICIÓN [7] [8] ..... 11

1.3.2 ARQUITECTURA ESTÁNDAR DEL CONTROLADOR PID [8] ..... 12

1.3.2.1 Acción de control proporcional ..... 12

1.3.2.2 Acción de control integral ..... 12
1.3.2.3 Acción de control derivativa ..... 13
1.3.3 CONTROLADOR PID DIGITAL [9] ..... 13
**1.4 COMUNICACIÓN SERIAL ..... 14**
1.4.1 DEFINICIÓN [10] ..... 14
1.4.2 FUNCIONAMIENTO ..... 15
1.4.2.1 Estándar RS-232 [10] [11] [12] ..... 15
1.4.2.1.1 Velocidad de transmisión ..... 15
1.4.2.1.2 Métodos de transmisión serial ..... 16
1.4.2.2 Estándar USB [13] [14] ..... 17
**1.5 COMUNICACIÓN BLUETOOTH [15] [16] ..... 19**
1.5.1 DEFINICIÓN ..... 19
1.5.2 FUNCIONAMIENTO ..... 19
1.5.3 ARQUITECTURA DEL HARDWARE ..... 20
1.5.4 ARQUITECTURA DEL SOFTWARE ..... 21
**1.6 SERVOMOTORES [17] ..... 22**
1.6.1 DEFINICIÓN ..... 22
1.6.2 MODO DE FUNCIONAMIENTO ..... 23
**1.7 MICROCONTROLADORES [19] [20] ..... 24**
1.7.1 DEFINICIÓN ..... 24
1.7.2 COMPONENTES INTERNOS ..... 24
**1.8 SENSOR, ACTUADORES Y MÓDULOS DE COMUNICACIÓN ..... 26**
1.8.1 SENSOR PARA LA ADQUISICIÓN DE IMÁGENES ..... 26
1.8.1.1 Cámara Logitech C615 [21] ..... 26
1.8.2 ACTUADORES DEL SISTEMA ..... 27
1.8.2.1 Servomotores Hitec HS-645MG [22] ..... 27
1.8.3 MÓDULO DE COMUNICACIÓN SERIAL ..... 28
1.8.3.1 Módulo USB-TTL [23] ..... 29
1.8.4 MÓDULO DE COMUNICACIÓN BLUETOOTH ..... 30
1.8.4.1 Módulo bluetooth HC-06 [25] ..... 31
1.8.5 MICROCONTROLADOR ..... 33
1.8.5.1 Microcontrolador ATmega 328 [26] ..... 33

1.8.5.2 Placa Arduino Uno [27] ..... 33

**CAPÍTULO 2 ..... 35**

**IMPLEMENTACIÓN DEL HARDWARE ..... 35**

**2.1 DESCRIPCIÓN DEL PROCESO Y SUS COMPONENTES ..... 35**

**2.2 SISTEMA MECÁNICO ..... 36**

2.2.1 PLATAFORMA Y ESFERA ..... 36

2.2.2 EJE CENTRAL DE PIVOTE ..... 37

2.2.3 SISTEMA DE TRANSMISIÓN DE MOVIMIENTO ..... 38

2.2.4 SERVOMOTORES Y SOPORTES ..... 39

2.2.5 CÁMARA Y ESTRUCTURA DE SOPORTE ..... 40

**2.3 COMPONENTES ELECTRÓNICOS ..... 41**

2.3.1 FUENTE DE ALIMENTACIÓN ..... 41

2.3.2 PLACA DE DISTRIBUCIÓN DE LA FUENTE DE ALIMENTACIÓN ..... 42

2.3.3 MÓDULOS DE COMUNICACIÓN ..... 42

**2.4 IMPLEMENTACIÓN FÍSICA ..... 43**

**CAPÍTULO 3. ..... 44**

**DESARROLLO DEL SOFTWARE ..... 44**

**3.1 PROCESAMIENTO DE IMÁGENES ..... 44**

3.1.1 ALGORITMO DE VISIÓN ARTIFICIAL ..... 44

**3.2 IMPLEMENTACIÓN DEL FILTRO DE KALMAN ..... 49**

3.2.1 JUSTIFICACIÓN ..... 49

3.2.2 MODELO DE LA ESFERA EN MOVIMIENTO [28] ..... 49

3.2.3 MATRICES PARA EL FILTRO DE KALMAN [6] ..... 51

**3.3 IMPLEMENTACIÓN DEL CONTROLADOR ..... 51**

3.3.1 ELECCIÓN DEL CONTROLADOR ..... 51

3.3.2 MODELO DINÁMICO DEL SISTEMA [29] [30] ..... 52

3.3.2.1 Modelo completo del sistema ..... 52

3.3.2.2 Modelo reducido del sistema ..... 59

3.3.3 ANÁLISIS DE CONTROLABILIDAD Y OBSERVABILIDAD [31] ..... 61

3.3.3.1 Controlabilidad ..... 61

3.3.3.2 Observabilidad ..... 62
3.3.4 DESARROLLO DEL CONTROLADOR ..... 63
3.3.5 ANÁLISIS DE ESTABILIDAD ..... 65
3.3.5.1 Estabilidad de Lyapunov [32] ..... 65
3.3.6 PRUEBAS DE SIMULACIÓN ..... 66
3.3.6.1 Implementación en Simulink ..... 66
3.3.6.2 Resultados obtenidos por Simulación ..... 67
3.3.7 REAJUSTE DE LOS PARÁMETROS DEL CONTROLADOR ..... 72
**3.4 INTERFACES DE USUARIO ..... 73**
3.4.1 DISEÑO DE LA INTERFAZ PRINCIPAL (PC) ..... 73
3.4.1.1 Lazo principal ..... 74
3.4.1.1.1 Control de posición (Lazo A) ..... 77
3.4.1.1.2 Recepción de datos (Lazo B) ..... 83
3.4.2 DISEÑO DE LA INTERFAZ SECUNDARIA (SMARTPHONE) ..... 94
3.4.2.1 Programación en Android Studio: Conceptos básicos [33] ..... 95
3.4.2.2 Desarrollo de la aplicación ..... 97
3.4.3 DISEÑO DE LA INTERFAZ AUXILIAR ..... 101
**3.5 IMPLEMENTACIÓN DEL PLOGRAMA EN ARDUINO ..... 102**
**CAPÍTULO 4 ..... 106**
**PRUEBAS Y RESULTADOS ..... 106**
**4.1 PRUBEAS – FILTRO DE KALMAN ..... 106**
4.1.1 PRUEBAS DE SEGUIMIENTO DE LA ESFERA ..... 106
4.1.2 PRUEBAS DE SEGUIMIENTO CON RESPUESTA AL RUIDO ..... 109
**4.2 PRUEBAS – SEÑAL DE CONTROL ..... 111**
**4.3 PRUEBAS – CONTROL DE POSICIÓN ..... 112**
**4.4 PRUEBAS – SEGUIMIENTO DE CAMINOS ..... 114**
**CAPÍTULO 5. ..... 138**
**CONCLUSIONES Y RECOMENDACIONES ..... 138**
**5.1 CONCLUSIONES ..... 138**

**5.2 RECOMENDACIONES ..... 139**

**REFERENCIAS BIBLIOGRÁFICAS ..... 140**

**ANEXOS ..... A1**

**ANEXO A. GUÍA DE CONFIGURACIONES Y PUESTA EN MARCHA ..... A1**

A.1 REQUERIMIENTOS MÍNIMOS DEL SISTEMA ..... A1

A.2 PREPARACIÓN DEL AMBIENTE DE TRABAJO ..... A1

**ANEXO B. PLANOS DE la implemntación DEL SISTEMA MECÁNICO ..... B1**

B.1 PLANO DEL SOPORTE PARA LA CÁMARA ..... B1

B.2 PLANO DEL SOPORTE PARA LOS SERVOMOTORES ..... B2

B.3 PLANO DEI EJE PIVOTE ..... B3

B.4 PLANOS DEL MONTAJE DEL SISTEMA ..... B4

**ANEXO C. HOJAS DE DATOS ..... C1**

C.1 MICROCONTROLADOR ATMEGA 328 ..... C1

C.2 MÓDULO USB-TTL (CHIP CP2102) ..... C2

C.3 MÓDULO DE COMUNICACIÓN BLUETOOTH HC-06 ..... C3

C.4 CÁMARA LOGITECH C615 ..... C4

# ÍNDICE DE FIGURAS

**Figura 1.1.** Mapa de colores en una imagen RGB [2]..... 2
**Figura 1.2.** Proceso de Visión Artificial [2] ..... 2
**Figura 1.3.** Sensor óptico de una cámara de video [3] ..... 3
**Figura 1.4.** Ajuste de brillo en una imagen ..... 4
**Figura 1.5.** Ajuste de contraste en una imagen ..... 5
**Figura 1.6.** Imagen RGB y Escala de grises [2]..... 6
**Figura 1.7.** (Izq.) Imagen en escala de grises (Der.) Imagen binarizada. ..... 6
**Figura 1.8.** Complemento de una imagen ..... 7
**Figura 1.9.** Técnica de extracción del contorno de un objeto [1] ..... 7
**Figura 1.10.** Centroide de una región contigua y no contigua. [4] ..... 8
**Figura 1.11.** Algoritmo del filtro de Kalman [5]..... 9
**Figura 1.12.** Lazo cerrado de control de un Planta. [7]..... 11
**Figura 1.13.** Arquitectura estándar del PID. [8]..... 12
**Figura 1.14.** Transmisión de datos síncrona. [12]..... 16
**Figura 1.15.** Transmisión de datos Asíncrona. [12] ..... 17
**Figura 1.16.** Tipos de conectores USB [14]..... 18
**Figura 1.17.** Redes Scatternet y Piconet. [15] ..... 20
**Figura 1.18.** Arquitectura de Hardware de Bluetooth. [15]..... 20
**Figura 1.19.** Arquitectura de Software de Bluetooth. [15] ..... 21
**Figura 1.20.** Partes constitutivas de un servomotor. [18]..... 22
**Figura 1.21.** PWM, Control de posición del servomotor. [17]..... 23
**Figura 1.22.** Diagrama de bloques general de un microcontrolador [20] ..... 24
**Figura 1.23.** Cámara Logitech C615 [21]..... 26
**Figura 1.24.** Servomotor Hitec Hs-645MG. [22]..... 27
**Figura 1.25.** Interfaz de conversión de niveles de voltajes. ..... 28
**Figura 1.26.** Modulo BTE13-007. [24]..... 29
**Figura 1.27.** Distribucion de pines del modulo BTE13-007. [24]..... 29
**Figura 1.28.** Enlace de Radiofrecuencia..... 31
**Figura 1.29.** Módulo Bluetooth HC-06 [25] ..... 31
**Figura 1.30.** Microcontrolador ATmega 328. [26] ..... 33
**Figura 1.31.** Placa Arduino Uno. [27]..... 34

**Figura 2.1.** Principales partes constitutivas del sistema ..... 35
**Figura 2.2.** Partes del sistema mecánico..... 36
**Figura 2.3.** Esfera y plataforma ..... 37
**Figura 2.4.** Eje central de pivote ..... 37
**Figura 2.5.** Corrección de movimiento de la plataforma en el eje yaw. ..... 38
**Figura 2.6.** Transmisión de movimiento a la plataforma ..... 38
**Figura 2.7.** Soporte para servomotores ..... 40
**Figura 2.8.** Estructura de soporte para la cámara. ..... 40
**Figura 2.9.** Esquema de conexiones del circuito de control. ..... 41
**Figura 2.10.** Fuente de alimentación (6V DC). ..... 42
**Figura 2.11.** Placa de distribución de la fuente de alimentación. ..... 42
**Figura 2.12.** Conexión de los módulos de comunicación ..... 43
**Figura 2.13.** Implementación física – Sistema completo ..... 43

**Figura 3.1.** Diagrama general del sistema. ..... 44
**Figura 3.2.** Imagen tomada sin procesamiento..... 45
**Figura 3.3.** Imagen recortada en la región de interés. ..... 46
**Figura 3.4.** Imagen aplicada la configuración de brillo y contraste. ..... 46
**Figura 3.5.** Imagen en escala de grises..... 47
**Figura 3.6.** Imagen en binario. ..... 48
**Figura 3.7.** Complemento de la imagen..... 48
**Figura 3.8.** Coordenadas del centro de la esfera..... 49
**Figura 3.9.** Movimiento de la esfera en el plano ..... 50
**Figura 3.10.** Representación de las variables en el eje X para la plataforma. ..... 52
**Figura 3.11.** Representación de las variables en el eje X para los servomotores 53
**Figura 3.12.** Composición de velocidades de la esfera ..... 54
**Figura 3.13.** Respuesta del Sistema sin Compensador..... 64
**Figura 3.14.** Respuesta del Sistema con Compensador. ..... 65
**Figura 3.15.** Lugar Geométrico de las Raíces, sistema compensado. ..... 66
**Figura 3.16.** Simulación Sistema Bola-Plataforma. ..... 66
**Figura 3.17.** Simulación Bola-Plataforma: Dinámica del Sistema..... 67

**Figura 3.18.** Simulación Bola-Plataforma: Controlador PD ..... 67
**Figura 3.19.** Referencia: Puntos ..... 68
**Figura 3.20.** Referencia, Posición Real, Control vs Tiempo: Puntos ..... 68
**Figura 3.21.** Referencia: Círculo ..... 69
**Figura 3.22.** Referencia, Posición Real, Control vs Tiempo: Círculo ..... 69
**Figura 3.23.** Referencia: Figura ocho ..... 70
**Figura 3.24.** Referencia, Posición Real, Control vs Tiempo: Ocho ..... 70
**Figura 3.25.** Referencia: Elipse ..... 71
**Figura 3.26.** Referencia, Posición Real, Control vs Tiempo: Elipse ..... 71
**Figura 3.27.** Lógica del Lazo Principal ..... 73
**Figura 3.28.** Lazo Principal: Bucle “infinito” ..... 74
**Figura 3.29.** Función setConfiguration ..... 75
**Figura 3.30.** Función CheckSerialPort ..... 76
**Figura 3.31.** Lazo Principal: Lazo Central ..... 76
**Figura 3.32.** Lazo Central: Control de Posición ..... 77
**Figura 3.33.** Función ballPosition ..... 78
**Figura 3.34.** Función kalmanFilter ..... 79
**Figura 3.35.** Función setpoints ..... 79
**Figura 3.36.** Función myPid ..... 80
**Figura 3.37.** Función commandSpoints ..... 81
**Figura 3.38.** Función dataPrepare ..... 81
**Figura 3.39.** Función storeData ..... 82
**Figura 3.40.** Lazo Central: Recepción de datos ..... 83
**Figura 3.41.** Función bluetoothFigureIs ..... 84
**Figura 3.42.** Función validarDato ..... 85
**Figura 3.43.** Función caminoTriángulo ..... 86
**Figura 3.44.** Función setCustomPid ..... 87
**Figura 3.45.** Función circle2points ..... 87
**Figura 3.46.** Función points2line ..... 88
**Figura 3.47.** Función caminoCírculo ..... 89
**Figura 3.48.** Función circle ..... 89
**Figura 3.49.** Función especiales ..... 89
**Figura 3.50.** Función caminoPuntos ..... 90

**Figura 3.51.** Función dibujarPunto..... 90
**Figura 3.52.** Interfaz de Control: Inicial..... 91
**Figura 3.53.** Función pararContinuar..... 91
**Figura 3.54.** Interfaz de Control: Graficas Posición vs Tiempo..... 92
**Figura 3.55.** Interfaz de Control: Graficas Control vs Tiempo..... 93
**Figura 3.56.** Interfaz de Control: Graficas Posición X vs Posición Y ..... 93
**Figura 3.57.** Interfaz de Control: Secciones ..... 94
**Figura 3.58.** Interfaz de Android Studio ..... 95
**Figura 3.59.** Clases y Objetos ..... 96
**Figura 3.60.** Ciclo de vida de una Actividad. [34]..... 97
**Figura 3.61.** Actividades de la aplicación BallOnPlate..... 97
**Figura 3.62.** Estructura lógica de la actividad Menú. ..... 98
**Figura 3.63.** Menú de opciones, aplicación BallOnPlate. ..... 98
**Figura 3.64.** Estructura lógica de la actividad Métodos de Control..... 99
**Figura 3.65.** Referencias de puntos y caminos..... 100
**Figura 3.66.** Ayuda, aplicación BallOnPlate..... 101
**Figura 3.67.** Aplicación Auxiliar ViewBallOnPlate..... 102
**Figura 3.68.** Trama de comunicación entre el computador y la placa Arduino. . 103
**Figura 3.69.** Diagrama de flujo – Programación en Arduino..... 104
**Figura 3.70.** Ángulos de los servomotores. ..... 104

**Figura 4.1.** Pruebas del filtro de Kalman con esfera estática (Eje X)..... 106
**Figura 4.2.** Pruebas del filtro de Kalman con esfera estática (Eje Y)..... 107
**Figura 4.3.** Pruebas de seguimiento de la esfera utilizando el filtro de Kalman (X vs Y). ..... 107
**Figura 4.4.** Pruebas de seguimiento de la esfera utilizando el filtro de Kalman (Eje X)..... 108
**Figura 4.5.** Pruebas de seguimiento de la esfera utilizando el filtro de Kalman (Eje Y)..... 108
**Figura 4.6.** Respuesta del filtro de Kalman ante oscilaciones de la plataforma (X vs Y). ..... 109
**Figura 4.7.** Respuesta del filtro de Kalman ante oscilaciones de la plataforma (Eje X)..... 109

**Figura 4.8.** Respuesta del filtro de Kalman ante oscilaciones de la plataforma (Eje Y)..... 110
**Figura 4.9.** Pruebas de la señal de control para un ángulo de 90 grados. ..... 111
**Figura 4.10.** Pruebas de la señal de control para un ángulo de 180 grados. .... 111
**Figura 4.11.** Control de posición de la esfera ..... 112
**Figura 4.12.** Posición de la esfera en el eje X ..... 112
**Figura 4.13.** Posición de la esfera en el eje Y ..... 113
**Figura 4.14.** Señal de control (Ángulos a cargar en los servomotores) ..... 114
**Figura 4.15.** Círculo – Radio 7 – Evolución del seguimiento (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador. ..... 116
**Figura 4.16.** Círculo – Radio 11 – Evolución del seguimiento (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador. ..... 117
**Figura 4.17.** Elipse – Evolución del seguimiento (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador. ..... 118
**Figura 4.18.** Figura 8 – Evolución del seguimiento (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador. ..... 119
**Figura 4.19.** Trébol 3 – Evolución del seguimiento (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador. ..... 120
**Figura 4.20.** Triángulo – Radio 15 – Evolución del seguimiento (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador. ..... 121
**Figura 4.21.** Triángulo – Radio 7 (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador..... 123
**Figura 4.22.** Triángulo – Radio 11 (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador..... 124
**Figura 4.23.** Triángulo – Radio 15 (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador..... 125
**Figura 4.24.** Cuadrado – Radio 7 (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador..... 126
**Figura 4.25.** Cuadrado – Radio 11 (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador..... 127

**Figura 4.26.** Cuadrado – Radio 15 (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador. ..... 128
**Figura 4.27.** Círculo – Radio 7 (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador. ..... 129
**Figura 4.28.** Círculo – Radio 11 (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador. ..... 130
**Figura 4.29.** Círculo – Radio 15 (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador. ..... 131
**Figura 4.30.** Figura 8 (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador. ..... 132
**Figura 4.31.** Trébol de 3 hojas (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador. ..... 133
**Figura 4.32.** Trébol de 5 hojas (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador. ..... 134
**Figura 4.33.** Elipse (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador. ..... 135

**Figura A.1.** Virtual Serial Port ..... A2
**Figura A.2.** Instalación de soporte para cámaras USB ..... A4
**Figura A.3.** Verificación de puertos seriales ..... A5
**Figura A.4.** Interfaz de control inicial ..... A5

**Figura B.1.** Plano del soporte para la cámara ..... B1
**Figura B.2.** Plano del soporte para los servomotores ..... B2
**Figura B.3.** Plano del eje pivote ..... B3
**Figura B.4.** Planos del montaje del sistema ..... B6

**Figura C.1.** Hoja de datos del microcontrolador ATmega 328 ..... C1
**Figura C.2.** Hoja de datos del módulo USB-TTL (chip CP2102) ..... C2
**Figura C.3.** Hoja de datos del módulo de comunicación bluetooth HC-06 ..... C3
**Figura C.4.** Hoja de datos de la cámara Logitech C615 ..... C4

# ÍNDICE DE TABLAS

**Tabla 1.1.** Niveles de voltaje del estándar RS-232 [11] ..... 15
**Tabla 1.2.** Pines del conector USB. [14] ..... 18
**Tabla 1.3.** Distribución de pines – Servomotor Hitec Hs-645MG. [22] ..... 28
**Tabla 1.4.** Características Servomotor Hitec Hs-645MG [22] ..... 28
**Tabla 1.5.** Descripcion de los pines del Modulo BTE13-007. [24] ..... 30
**Tabla 1.6.** Características técnicas del Módulo BTE13-007. [24] ..... 30
**Tabla 1.7.** Comandos AT para configurar el Módulo Bluetooth HC-06 [25] ..... 32
**Tabla 1.8.** Velocidades de transmisión del Módulo Bluetooth HC-06 [25] ..... 32
**Tabla 1.9.** Distribución de pines – Módulo Bluetooth HC 06 [25] ..... 32
**Tabla 1.10.** Distribución de pines Placa Arduino Uno [27] ..... 34
**Tabla 1.11.** Principales características de la Placa Arduino Uno. [27] ..... 34

**Tabla 2.1.** Características de la plataforma ..... 39

**Tabla 3.1.** Constantes del PID para cada camino. ..... 72

**Tabla 4.1.** Análisis en estado estable – control de posición. ..... 113
**Tabla 4.2.** Error cuadrático medio de cada figura. ..... 136

**Tabla A.1.** Suites de Programación ..... A3

# RESUMEN

En el campo de la Ingeniería de Control existen sistemas que son muy sencillos de controlar por la naturaleza de su comportamiento, teniendo así soluciones relativamente fáciles de implementar. Así también existen sistemas que por su comportamiento no lineal e inestable se han convertido en clásicos problemas de la Ingeniería de Control y son objeto de análisis e investigación. Este es el caso del sistema bola-plataforma, un sistema no lineal e inestable que tiene como objetivos posicionar y estabilizar la esfera en un punto del plano de una plataforma y realizar seguimiento de caminos y trayectorias, entre los más destacados.

El sistema implementado está compuesto de una plataforma donde rodará la bola, un eje central en el que descansa la plataforma el cual actúa como soporte de la mayor parte del peso de la estructura y como pivote para permitir el movimiento en todas las direcciones. Además, en la plataforma, está montada una estructura metálica que sostiene una cámara la cual actúa como sensor para la visualización de la posición de la esfera que es obtenida mediante el procesamiento de imágenes con técnicas de visión artificial. Los actuadores de la planta son servomotores, los cuales transmiten su movimiento angular a la plataforma mediante extensiones metálicas fijadas en su eje.

El funcionamiento del sistema inicia con la cámara tomando y enviando imágenes del plano de la plataforma hacia el computador, de estas imágenes se obtiene la posición actual de la esfera en el plano. Con esta información, a través de un software computacional, mediante un algoritmo de control, se calcula la señal que es enviada vía comunicación serial a una placa Arduino, la misma que se encarga de controlar los servomotores, los cuales generan el movimiento angular de la plataforma y por tanto el desplazamiento de la esfera, para nuevamente comenzar el ciclo.

# PRESENTACIÓN

El proyecto planteado está conformado por cinco capítulos que se describen a continuación.

El *Capítulo 1*, detalla todos los principios teóricos utilizados para la adquisición y procesamiento de la imagen, el filtrado de las mediciones y el diseño del controlador, así como los fundamentos sobre los dispositivos de comunicación y actuadores que se usan en el sistema.

El *Capítulo 2*, describe la implementación del sistema, sus partes constitutivas, así como la elección y características específicas de los dispositivos seleccionados en función de sus requerimientos.

El *Capítulo 3*, detalla el modelado matemático de la dinámica del sistema, y las consideraciones para la linealización y obtención del modelo reducido. Además, se detalla el diseño del controlador y sus respectivas pruebas en simulación. También se indica la estructura lógica y el diseño de las interfaces de usuario que permiten el control de la plataforma.

El *Capítulo 4*, muestra el análisis de las pruebas y resultados obtenidos, referentes a la implementación del filtro de Kalman, el seguimiento de los caminos y control de posición de la esfera en la plataforma.

El *Capítulo 5*, contiene las conclusiones y recomendaciones obtenidas a lo largo de la elaboración del presente trabajo.

# CAPÍTULO 1
# MARCO TEÓRICO

En este capítulo se detalla el fundamento teórico detrás de la realización de este proyecto. Se exponen los conceptos fundamentales de los temas más relevantes sobre los cuales se sustenta el desarrollo del presente trabajo.

## 1.1 VISIÓN ARTIFICIAL

### 1.1.1 FUNDAMENTOS

#### 1.1.1.1 Definición [1]

La visión artificial consiste en una serie de técnicas cuyo objetivo es la extracción de la realidad óptica en un entorno. La visión artificial requiere de un sensor y un computador que permitan obtener información mediante imágenes que posteriormente son manipuladas con un determinado fin.

#### 1.1.1.2 Representación de una imagen [1] [2]

Una imagen es la representación de un medio físico expuesto a una fuente de iluminación. Dicha representación contiene información como: brillo, matices, contrastes, formas y colores, que son el resultado de la luz reflejada por los objetos. Una imagen al ser digitalizada se la puede representar como una matriz de puntos, llamados pixeles, los cuales a su vez contienen información sobre la intensidad de los colores.

En una imagen a color o RGB, que por sus siglas en inglés representan los colores: rojo (red - R), verde (green - G) y azul (blue - B) como la imagen indicada en la Figura 1.1, cada pixel contiene tres valores, correspondientes a la intensidad de los colores primarios: rojo, verde y azul, a partir de los cuales se obtienen los demás colores y tonalidades.

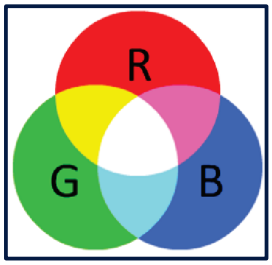



**Figura 1.1.** Mapa de colores en una imagen RGB [2]

El valor que toma cada pixel, es almacenado en una memoria, en registros de 8 bits, de modo que cada color puede adquirir un valor de 0 a 255 según la intensidad captada por el sensor óptico de la cámara.

### 1.1.1.3 Etapas del proceso de visión artificial [2]

La visión artificial comprende un proceso cuyo objetivo es obtener los mejores resultados posibles y dar un tratamiento eficiente a la imagen. Este proceso se muestra en la Figura 1.2.

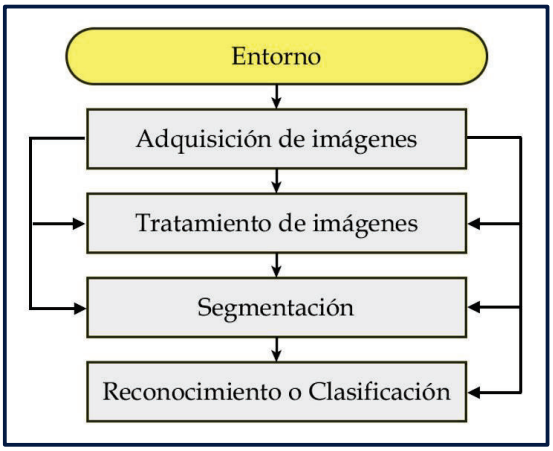



**Figura 1.2.** Proceso de Visión Artificial [2]

La primera etapa en el proceso de visión artificial es la adquisición de imágenes. Esta etapa comprende el sensor óptico y un sistema que digitalice la información obtenida.

*Etapa de tratamiento de imágenes*: En esta etapa se aplican filtros o transformadas para mejorar la imagen con el objetivo de eliminar ruido y enfatizar las zonas de importancia.

*Etapa de Segmentación*: Sirve para separar los objetos de interés en la imagen, de modo que en procesos siguientes se facilite su análisis.

*Etapa de Reconocimiento o Clasificación*: En esta etapa se realiza el análisis de aquello extraído en el proceso anterior, en busca de patrones que permitan identificar formas o colores específicos y obtener información de los mismos.

El proceso de visión artificial no es secuencial, así que se pueden alternar las diferentes etapas según convenga para la obtención de resultados satisfactorios.

### 1.1.2 ADQUISICIÓN DE IMÁGENES

#### 1.1.2.1 Sensor óptico [1]

Para realizar una captura de un evento, es necesario un sensor (Figura 1.3) que convierta la intensidad luminosa en una imagen digitalizada. Los equipos generalmente usados son cámaras y scanners.

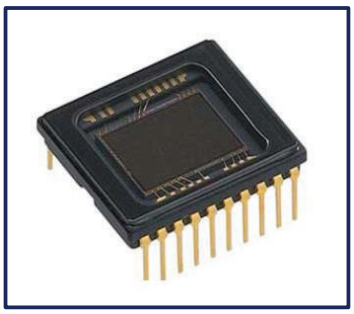


**Figura 1.3.** Sensor óptico de una cámara de video [3]

Las condiciones ambientales donde se captura el evento, tienen variables propias como: intensidad luminosa, contraste, brillo, etc. Estas variables pueden afectar la obtención de buenos resultados, por lo que el sensor óptico debe modificar sus parámetros para una correcta adquisición de información.

#### 1.1.2.2 Digitalización de imágenes [2]

Una vez que el sensor óptico toma la información, esta pasa a una tarjeta de adquisición de imágenes donde se discretiza la información obteniendo una imagen digital, formada por pixeles, cada uno con valores de intensidad.

En el proceso de discretización, es importante el tiempo de muestreo. Este tiempo define el parámetro de resolución de una cámara. Este valor viene dado en fotogramas por segundo (fps), es decir el número de capturas que puede tomar el sensor en un segundo.

Otro factor importante en la digitalización es la cuantización. Este término describe la precisión en la medición de la intensidad luminosa. Generalmente la cuantización se realiza en bytes. Es decir que cada color puede tomar valores intermedios entre 0 y 255.

### 1.1.3 TÉCNICAS PARA EL PROCESAMIENTO DE IMÁGENES

#### 1.1.3.1 Brillo y contraste [1]

El *brillo* en una imagen modifica la exposición de la misma a la luz. Con esta propiedad se puede ajustar la cantidad de luz recibida ya que en determinados ambientes la iluminación no es uniforme. La Figura 1.4 muestra los resultados al variar el brillo a una imagen.


**Figura 1.4.** Ajuste de brillo en una imagen

El contraste es una propiedad que permite resaltar los objetos de una imagen, incrementando la intensidad de los colores para de esta manera diferenciarlos del fondo. Esto se consigue cambiando la intensidad luminosa entre las zonas más oscuras y más claras de la imagen. La Figura 1.5 muestra los resultados al variar el contraste a una imagen.


**Figura 1.5.** Ajuste de contraste en una imagen

### 1.1.3.2 Imágenes RGB y escala de grises [1] [2]

Una imagen a color RGB está formada por una matriz de *m* x *n* x *3*, en donde cada pixel de la imagen contiene información de la mezcla de los tres colores básicos. Sin embargo, cuando una imagen contiene únicamente la información del brillo, se denomina imagen en escala de grises.

Una imagen en escala de grises está formada por una matriz de *m* x *n* que contiene la información de la intensidad luminosa de cada pixel. Por tanto, esta puede tomar valores entre 0 y 255, siendo 0 el valor correspondiente al color negro y 255 al color blanco.

De forma general, se puede transformar una imagen en escala de grises, utilizando la siguiente relación:

$$IEG\ (m,n) = A(Rojo(m,n)) + B(Verde(m,n)) + C(Azul(m,n)) \qquad (1.1)$$

donde:

$$\begin{aligned} \text{IEG} &: \text{Imagen en escala de grises} \\ \text{A, B, C} &: \text{Porcentaje de cada color} \\ Rojo &: Matriz\ (\text{mxn})\ \text{del color Rojo} \\ Verde &: Matriz\ (\text{mxn})\ \text{del color Verde} \\ Azul &: Matriz\ (\text{mxn})\ \text{del color Azul} \end{aligned}$$

En la Figura 1.6 se observa la estructura de las matrices para imágenes RGB y en escala de grises.

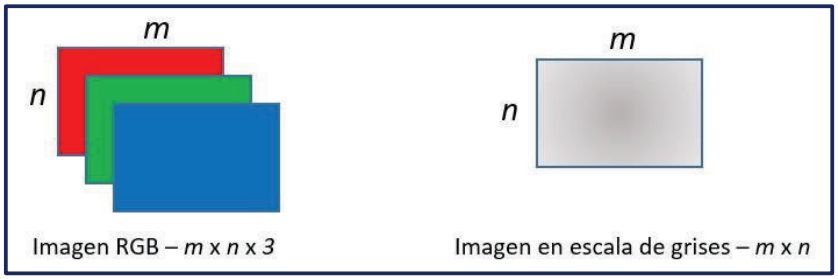



**Figura 1.6.** Imagen RGB y Escala de grises [2]

### 1.1.3.3 Binarización [1]

A partir de una imagen en escala de grises, la binarización consiste en crear una nueva imagen donde a cada pixel se le asigna 0 cuando los valores de intensidad sean menores a un umbral, y 1 cuando los valores sean mayores o iguales al valor de umbral, de modo que la imagen resultante es una imagen en blanco y negro sin tonalidades intermedias, como la indicada en la Figura 1.7.

$$Binarización\ (m,n) = \begin{cases} 0 & si\ \ Img\ (m,n) < umbral \\ 1 & si\ \ Img\ (m,n) \geq umbral \end{cases} \tag{1.2}$$

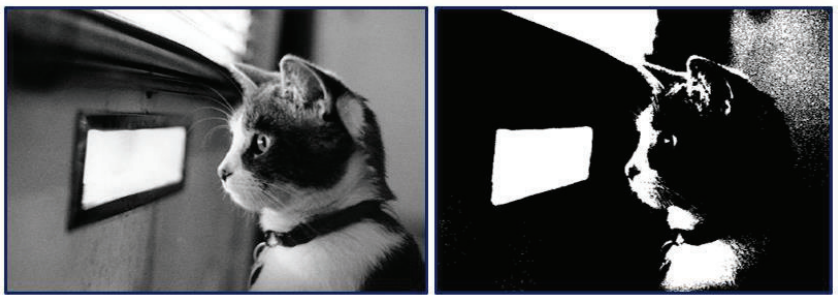


**Figura 1.7.** (Izq.) Imagen en escala de grises (Der.) Imagen binarizada.

### 1.1.3.4 Complemento de una imagen [1]

La propiedad de complemento, invierte el valor de los bits de una imagen en binario. Después de aplicado, se obtiene el negativo de la imagen original. Esta propiedad se observa en la Figura 1.8.

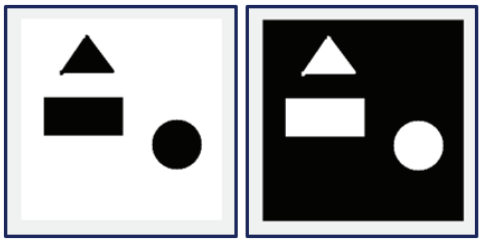


**Figura 1.8.** Complemento de una imagen

### 1.1.3.5 Segmentación y extracción de propiedades [1] [2]

La segmentación consiste en fragmentar una imagen o extraer de ella áreas de interés. La segmentación se logra mediante la utilización de parámetros como texturas, colores, contornos y formas.

Después de aplicado el método de segmentación, se obtiene una nueva imagen, en la que se puede identificar las características propias de lo que se deseaba extraer. Generalmente los algoritmos de segmentación se aplican únicamente a imágenes binarias (1 y 0).

#### *1.1.3.5.1 Contornos y formas*

El contorno de un objeto en una imagen, corresponde a todos los pixeles que cambian de valor respecto a un fondo. El análisis se realiza a cada pixel, comparando sus valores con los pixeles adyacentes. En una imagen binaria, un cambio de 1 a 0 o viceversa crea la idea de un contorno.

Se puede emplear la técnica de contorno para definir la silueta de un objeto y por ende extraer figuras u objetos, como se ve en la Figura 1.9.

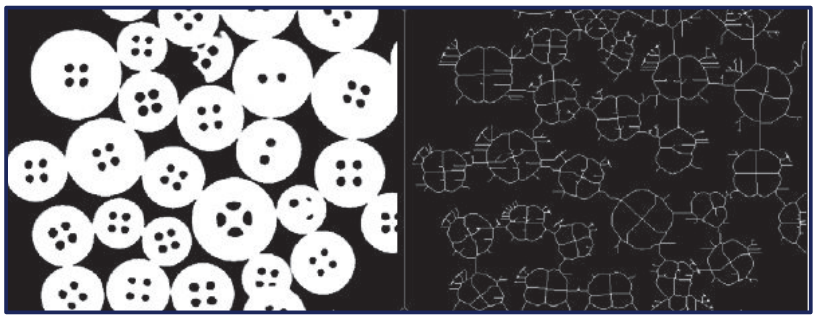


**Figura 1.9.** Técnica de extracción del contorno de un objeto [1]

*1.1.3.5.2 Propiedad de centroide [4]*

Esta propiedad es útil para obtener las coordenadas de la ubicación de un objeto en una imagen. El centroide o centro de masa de un objeto se puede obtener tanto para regiones contiguas como para regiones no contiguas.

Esta técnica de visión se basa en identificar una región mediante su contorno. Esta región se hace encajar en un recuadro, en cuyo punto, donde se cruzan las diagonales se localiza el centroide, ver Figura 1.10.


**Figura 1.10.** Centroide de una región contigua y no contigua. [4]

## 1.2 FILTRO DE KALMAN [5] [6]

### 1.2.1 FUNDAMENTOS

El Filtro de Kalman, es un algoritmo recursivo que permite estimar estados de un sistema dinámico de manera que se minimiza el error en la medida, incluso cuando el sistema está expuesto a ruido.

Este filtro fue desarrollado por Rudolf Kalman y su utilidad reside en que permite conocer estados pasados, presentes y futuros de un sistema. Existen diferentes variantes del filtro de Kalman, sin embargo, en el presente trabajo se tratará únicamente el filtro de Kalman en tiempo discreto.

Este algoritmo requiere de dos elementos principales, una medida de la variable o las variables del sistema, tomada por un sensor, y un modelo del sistema. Cada una de estas, la medida y el modelo, tienen una incertidumbre en su valor. La medida está sometida a ruido tanto del propio sensor como del ambiente en el que se toma. Por otro lado, el modelo del sistema también tiene incertidumbre ya que

el modelo matemático es solo una aproximación del sistema real. Estas incertidumbres se conocen como covarianzas. Tanto la covarianza del modelo, como la covarianza de la medida se pueden manipular según convenga, decidiendo cuál de estos dos valores tienen mayor peso al momento de obtener un valor estimado final. Por tanto, el estado estimado por el filtro Kalman fusiona estos dos valores para conseguir un valor final que sea más cercano al valor real.

### 1.2.2 ALGORITMO DEL FILTRO DE KALMAN

El algoritmo del filtro de Kalman comprende dos fases, la fase de predicción (a priori) y la fase de corrección (a posteriori). Dicho algoritmo es recursivo, es decir guarda los estados anteriores para generar nuevos valores. Por tanto, para la primera iteración del algoritmo, es necesario tener valores iniciales para algunas de las matrices.

La Figura 1.11 ilustra el proceso del algoritmo del filtro de Kalman en tiempo discreto:

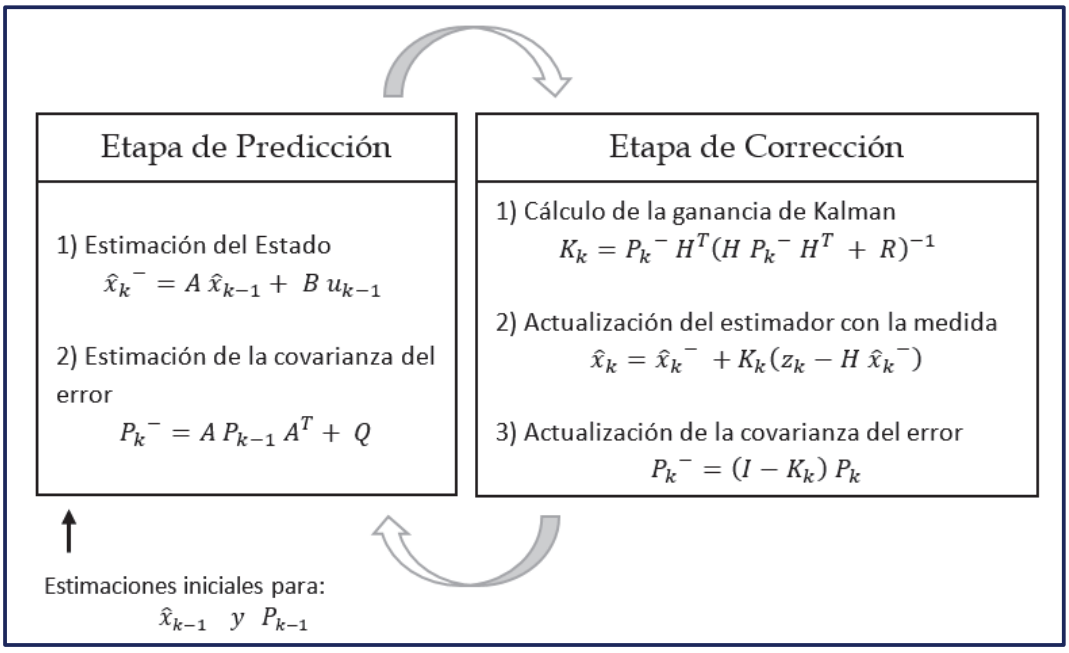



**Figura 1.11.** Algoritmo del filtro de Kalman [5]

#### 1.2.2.1 Fase de predicción

En esta fase se realiza el cálculo del estado de predicción y el cálculo de la covarianza estimada del error.

*Cálculo del estado de predicción:*

$$\hat{x}_k^{-} = A\,\hat{x}_{k-1} + B\,u_{k-1} \tag{1.3}$$

Donde:

$\hat{x}_k^{-}$ : $Estado\ actual\ estimado$
$A$ : $Matriz\ del\ modelo\ del\ sistema$
$\hat{x}_{k-1}$ : $Estado\ estimado\ anterior$
$B$ : $Matriz\ de\ control$
$u_{k-1}$ : $Estado\ anterior\ de\ control$

*Cálculo de la covarianza estimada del error:*

$$P_k^{-} = A\,P_{k-1}\,A^T + Q \tag{1.4}$$

Donde:

$P_k^{-}$ : $Covarianza\ del\ error$
$P_{k-1}$ : $Covarianza\ del\ error\ anterior$
$A^T$ : $Matriz\ transpuesta\ de\ A$
$Q$ : $Matriz\ de\ covarianza\ del\ modelo$

#### 1.2.2.2 Fase de corrección

En esta fase se realiza el cálculo de la ganancia de Kalman, el cálculo de la corrección en base al modelo y se calcula el estado actual de la covarianza del error.

*Cálculo de la ganancia de Kalman:*

$$K_k = P_k^{-}\,H^T(H\,P_k^{-}H^T + R)^{-1} \tag{1.5}$$

Donde:

$K_k$ : $Ganancia\ de\ Kalman$
$H$ : $Matriz\ de\ correlación\ de\ los\ valores\ medidos$
$H^T$ : $Matriz\ transpuesta\ de\ H$
$R$ : $Matriz\ de\ covarianza\ de\ la\ medición$

*Cálculo de la corrección en base al modelo (Salida del filtro de Kalman):*

$$\hat{x}_k = \hat{x}_k^{-} + K_k(z_k - H\,\hat{x}_k^{-}) \tag{1.6}$$

Donde:

$\hat{x}_k$ : $Salida\ del\ filtro\ de\ kalman\ (Estado\ estimado)$
$z_k$ : $Valor\ medido\ por\ el\ sensor$

*Cálculo del estado actual de la covarianza del error:*

$$P_k = (I - K_k)\,P_k^{-} \tag{1.7}$$

Donde:

$I$ : $Matriz\ identidad$

## 1.3 CONTROLADORES PID

### 1.3.1 DEFINICIÓN [7] [8]

Un controlador PID es un método de control que en función de la respuesta de la planta que se desea obtener y la respuesta real medida, calcula el error entre estas dos señales y permite tomar acciones de control de tal forma que el sistema responda de la manera esperada. Esto automáticamente lleva a la idea de realimentación, que no es más que medir la respuesta del sistema y usarla para determinar si está en el valor de consigna. La Figura 1.12, muestra el esquema general de un sistema con un lazo cerrado de control.

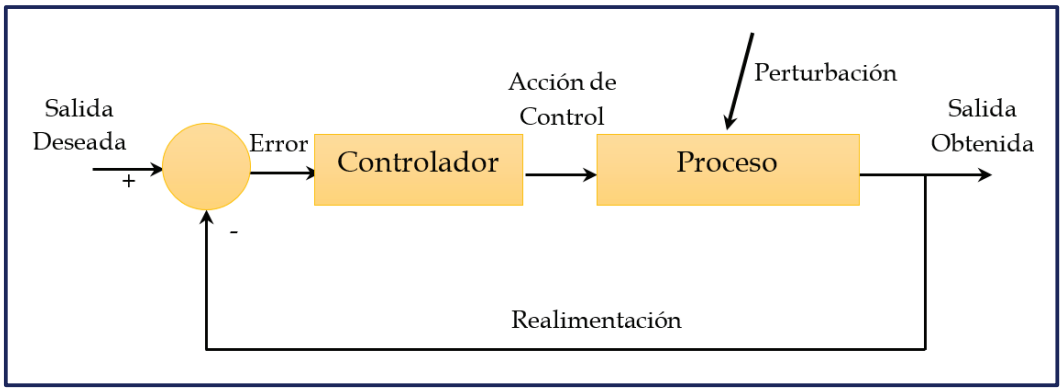



**Figura 1.12.** Lazo cerrado de control de un Planta. [7]

### 1.3.2 ARQUITECTURA ESTÁNDAR DEL CONTROLADOR PID [8]

El controlador PID consta de tres acciones de control que se manipulan independientemente mediante tres constantes: Ganancia Proporcional, Integral y Derivativa y su efecto en la salida del controlador es la suma de las tres. Estas acciones de control definirán el comportamiento de la respuesta obtenida tanto en la parte transitoria como en la parte estacionaria. La Figura 1.13 indica la arquitectura estándar del controlador PID.

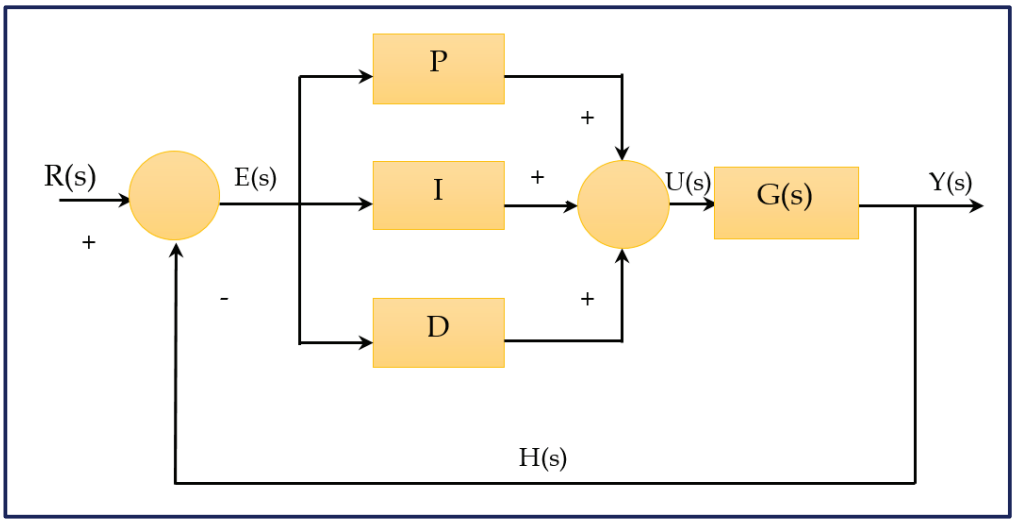



**Figura 1.13.** Arquitectura estándar del PID. [8]

#### 1.3.2.1 Acción de control proporcional

La salida del controlador es proporcional al error, es decir, la señal de control es un múltiplo del porcentaje de cambio en la medición, el término proporcional genera la salida del controlador:

$$U(s) = K_p E(s) \tag{1.8}$$

#### 1.3.2.2 Acción de control integral

La salida del controlador es proporcional al error acumulado, es decir, la acción de control es función del tiempo en que se ha mantenido el error. Esto implica que la señal de control U(s) tiene un valor diferente de cero incluso cuando el error sea cero, de esta manera se puede afirmar que la acción integral elimina el error en estado estacionario.

$$U(s) = \frac{K_i}{s} E(s) \tag{1.9}$$

#### 1.3.2.3 Acción de control derivativa

La salida del controlador es proporcional a la velocidad de cambio y dirección del error, es decir, que un cambio rápido en el error genera una gran acción de control. El término derivativo genera la salida del controlador:

$$U(s) = K_d s E(s) \tag{1.10}$$

El controlador PID es la suma de las tres acciones de control antes descritas, con lo que se obtiene la siguiente función de transferencia.

$$U(s) = K_p E(s) + \frac{K_i}{s} E(s) + K_d s E(s) \tag{1.11}$$

$$\frac{U(s)}{E(s)} = K_p + \frac{K_i}{s} + K_d s \tag{1.12}$$

Donde:

$U(s)$ : $Salida\ del\ Controlador, señal\ de\ control$
$E(s)$ : $Error\ entre\ el\ valor\ deseado\ y\ medido$
$K_p$ : $Ganancia\ Proporcional$
$K_i$ : $Ganancia\ Integral$
$K_d$ : $Ganancia\ Derivativa$

### 1.3.3 CONTROLADOR PID DIGITAL [9]

La ecuación (1.12), aunque se plantea en el dominio de la frecuencia sigue siendo de tiempo continuo. Para poder implementar este controlador en un dispositivo como un computador, es necesario discretizarlo ya que en cada iteración del lazo realizada en un periodo de tiempo, se obtiene un valor de la variable controlada y para ese valor se calcula la señal de control.

Para discretizar la función de transferencia del PID es necesario definir un periodo de muestreo al que se define como T, cabe recalcar que este valor es el tiempo que le toma al programa que ejecuta el controlador en realizar una iteración. Además, se debe definir el método con el que se aproximaran los términos: integral del error y derivada del error, por el hecho de que son funciones en tiempo continuo.

Para el termino derivativo se usará la aproximación de la derivada anterior:

$$\frac{de(t)}{dt} = \frac{e_k - e_{k-1}}{T} \tag{1.13}$$

Donde:

$e(t)$ : $Error\ en\ tiempo\ continuo$
$e_k$ : $Error\ actual$
$e_{k-1}$ : $Error\ anterior$
$T$ : $Tiempo\ de\ muestreo$

Para el termino integral se usará la aproximación Euler en atraso (Backward):

$$\int_0^t e(t)dt = \sum_{i=1}^{k} e(i)T = \sum_{i=1}^{k} Te_i \tag{1.14}$$

Donde:

$i$ : $Número\ de\ la\ iteración\ actual\ del\ lazo$
$e_i$ : $Error\ en\ esa\ iteración\ del\ lazo$
$k$ : $Tiempo\ actual$

Reemplazando las ecuaciones (1.11) y (1.12) en la función de transferencia del PID, tenemos:

$$\boldsymbol{u_k = K_p e_k + TK_i \sum_{i=1}^{k} e_i + K_d \frac{e_k - e_{k-1}}{T}} \tag{1.15}$$

La ecuación (1.15) ya es factible de implementar mediante un algoritmo en el computador.

## 1.4 COMUNICACIÓN SERIAL

### 1.4.1 DEFINICIÓN [10]

La comunicación serie es un método de trasmisión de datos que consiste en el envió de señales binarias de forma secuencial, es decir un pulso a continuación de otro, a diferencia de la comunicación en paralelo en la cual el envío de datos se da de forma simultánea. Aunque la comunicación en paralelo es mucho más rápida que la comunicación serie, esta última es mucho más sencilla y permite alcanzar mayores distancias.

### 1.4.2 FUNCIONAMIENTO

La comunicación serie entre dos equipos se implementa siguiendo interfaces como el RS-232, USB (Universal Serial Bus), etc.

#### 1.4.2.1 Estándar RS-232 [10] [11] [12]

RS-232 (Recommended Standard Number 232) es un estándar definido en las especificaciones ANSI, que indica un conjunto de normas y procedimientos para el intercambio de datos binarios entre dos equipos, un equipo terminal de datos (DTE) y un equipo de comunicación de datos (DCE).

El estándar especifica características eléctricas como niveles de voltajes, impedancias de cables, características mecánicas, tipos de conectores, descripción de pines y características funcionales de la interfaz.

Los Niveles de Voltajes que maneja el estándar RS-232 son los indicados en la Tabla 1.1:

**Tabla 1.1.** Niveles de voltaje del estándar RS-232 [11]

| Estado Lógico | Nivel de voltaje |
|---|---|
| 0 Lógico | 3 a 15 [V] |
| 1 Lógico | -3 a -15 [V] |

##### *1.4.2.1.1 Velocidad de transmisión*

La velocidad de transmisión (Baud Rate), es la velocidad a la cual una información es trasmitida en un canal de comunicación, es medida en baudios o en bits por segundo, por ejemplo 9600 Baudios.

Baudios es la cantidad de símbolos o codificaciones por segundo mientras que bit por segundo es la cantidad de estados de la señal en dicho intervalo de tiempo. Dependiendo del modo de transmisión los baudios pueden coincidir con los bits por segundos.

*1.4.2.1.2 Métodos de transmisión serial*

Existen dos métodos de transmitir datos que evitan errores de bits, la comunicación serial síncrona y la asíncrona.

La transmisión de datos síncrona consiste en sincronizar tanto al emisor como al receptor mediante una señal de reloj que indica básicamente el tiempo entre bits, esto hace que la información pueda ser leída sin errores, lo que se traduce en una comunicación mucho más rápida y eficiente ya que no se envía información adicional entre los datos que se quiere transmitir. La señal de reloj se la suele enviar usando una línea exclusiva para ella o codificada en la señal de datos, con la premisa de que si por alguna razón se pierde la señal de sincronismo la comunicación se da por terminada, en la Figura 1.14 se indica la sincronía entre la señal de reloj y la trama de datos.

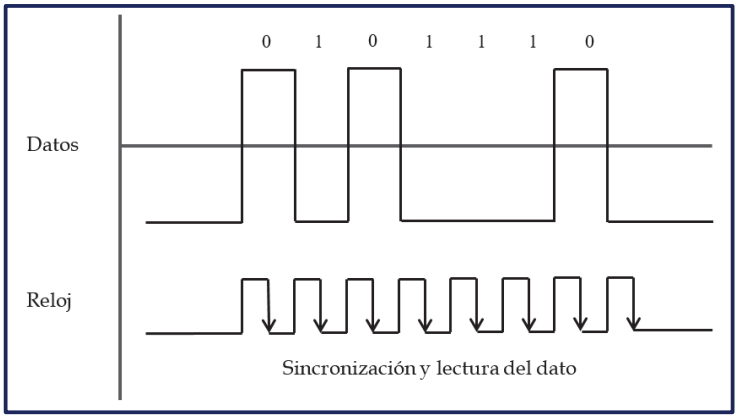



**Figura 1.14.** Transmisión de datos síncrona. [12]

La comunicación asíncrona añade información extra para determinar donde comienza y donde termina un dato de tal manera de poder obtener toda la trama sin errores. Los bits extras que se añaden son: Bit de Inicio, Bit de Paridad y Bit de Parada.

*Bit de Inicio y Bit de Parada***:** Estos bits cumplen con la función de delimitar el dato enviado de tal manera que quede “empaquetado”, de esta forma el receptor sabrá cuando empieza y cuando termina la trama. Cuando el canal está desocupado debe mantenerse en 1 lógico, así un cambio de estado a 0 lógico le indicará al receptor que un nuevo dato será enviado.

*Bit de Paridad*: El bit de paridad indica si la cantidad de estados lógicos “1” dentro de un dato recibido es par o impar. Esta información se utiliza para determinar en primera instancia la integridad del dato recibido.

*Bit de datos*: Indica la cantidad de bits que se envían por cada dato que pueden ir desde 5 a 9 bits.

La Figura 1.15, presenta la trama de una comunicación asíncrona e indica los bits extras que se añaden a los datos para que la información sea correctamente leída por el receptor.

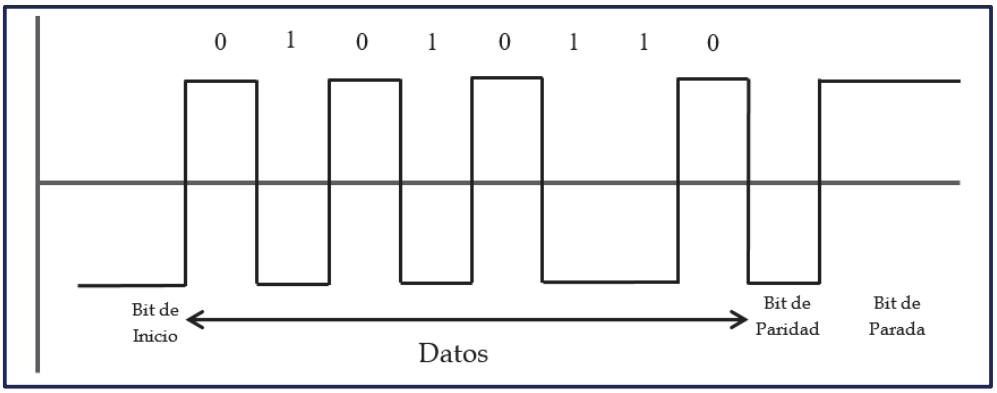



**Figura 1.15.** Transmisión de datos Asíncrona. [12]

### 1.4.2.2 Estándar USB [13] [14]

*USB* (Universal Serial Bus), es un estándar que define protocolos, tipos de conectores y cables para comunicación serial. USB permite alcanzar velocidades muy altas comparadas con el puerto serie RS-232, estas velocidades varían en las últimas versiones desde 60 Mbit/s (Versión 2.0) hasta 600 Mbit/s (Versión 3.0).

Esta interfaz además de permitir la comunicación serial, permite proveer de alimentación eléctrica a los dispositivos periféricos conectados mediante ella. Existen dos tipos de conectores USB, el tipo A y el tipo B, mostrados en la Figura 1.16:

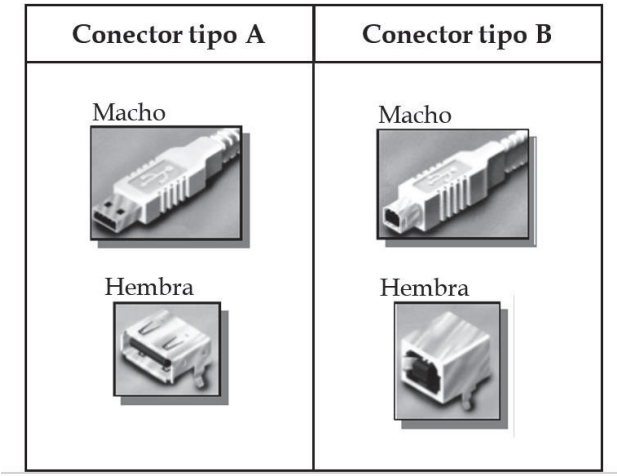



**Figura 1.16.** Tipos de conectores USB [14]

En la Tabla 1.2 se indican los nombres y las funciones de cada uno de los pines del conector USB.

**Tabla 1.2.** Pines del conector USB. [14]

| Pin | Señal | Descripción |
|---|---|---|
| 1 | Vcc | +5[V] |
| 2 | D- | Data- |
| 3 | D+ | Data+ |
| 4 | GND | Ground |

USB detecta los cambios de polaridad entre D- y D+ para obtener un “0” o “1” lógico (Señalización diferencial), con el objetivo de reducir los efectos del ruido electromagnético.

En el computador, el componente físico que realiza la comunicación serial es el UART (Universal Asynchronous Receiver and Transmitter) quien es el encargado del manejo de los puertos seriales para el envío y recepción de datos. Análogamente en el micro controlador el dispositivo físico que permite la comunicación serial es el USART (Universal Synchronous and Asynchronous serial Receiver and Transmitter).

En la comunicación asíncrona, se escriben los datos en el buffer de salida del UART y este se encarga de enviar bit a bit añadiendo los bits de inicio, parada y paridad. Por otro lado, cuando llegan los datos son almacenados en un buffer de entrada para su posterior lectura por el usuario.

# 1.5 COMUNICACIÓN BLUETOOTH [15] [16]

## 1.5.1 DEFINICIÓN

Bluetooth es un estándar global para la interconexión de dispositivos en redes inalámbricas de área personal (WPAN, Wireless Personal Area Network) mediante un enlace de radio-frecuencia, la tecnología bluetooth permite la transmisión de voz y de datos de manera simultánea en distancias que van hasta los 100 m. Entre sus principales características destacan la robustez, el bajo consumo, y el bajo costo.

## 1.5.2 FUNCIONAMIENTO

Bluetooth trabaja en la banda abierta ISM (Industrial Scientific Medical) que tiene un rango de frecuencia de 2400 a 2483.5 MHz, en el cual bluetooth incorpora la técnica “Adaptative Frecuency Hopping” o en sus siglas AFH, que permite realizar hasta 79 saltos de frecuencia de 83.5 canales disponibles (1 MHz por canal) con una rapidez de 1600 veces por segundo, con el objetivo de reducir la interferencia en ambientes en los que coexisten varias tecnologías inalámbricas.

Los dispositivos bluetooth pueden formar redes de tipo ad-hoc llamadas “Piconet” en las cuales pueden conectarse de 2 a 8 dispositivos al mismo tiempo, pudiendo además coexistir hasta 10 piconets en una misma área de cobertura. La red de piconets de denomina Scatternet.

Una Piconet es una red de corto alcance donde cada dispositivo comparte un mismo canal, es decir funcionan de forma síncrona, y siguen una misma frecuencia de salto. En una Piconet se establece un dispositivo como maestro y los demás como esclavos, el maestro define la frecuencia de salto y el reloj para la sincronización, los dispositivos esclavos hacen pequeños ajustes a su reloj nativo para sincronizarse con el maestro y así mantener la conexión.

Una Scatternet es una red de piconets que se forma cuando un dispositivo de una piconet (maestro o esclavo) participa como esclavo en otra piconet. En la Figura 1.17 se observa una Scatternet formada por varias piconets.

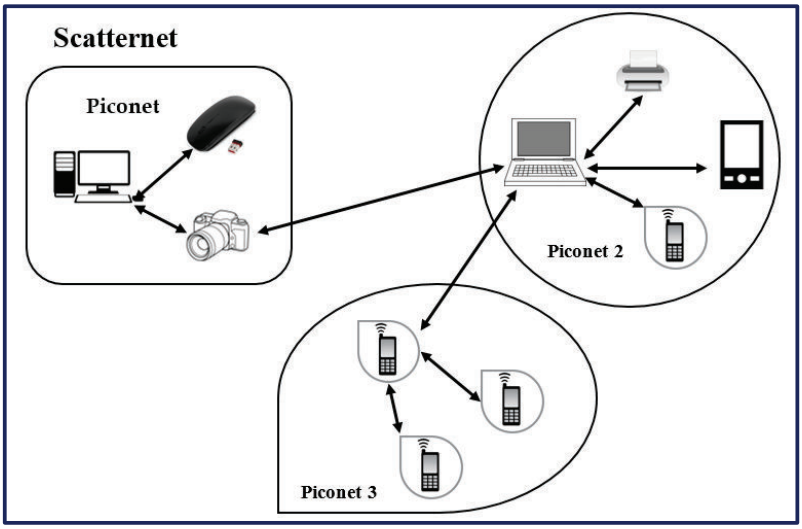



**Figura 1.17.** Redes Scatternet y Piconet. [15]

### 1.5.3 ARQUITECTURA DEL HARDWARE

Los dispositivos bluetooth se componen de dos elementos principales: un dispositivo de radio y un controlador digital como se indica en la Figura 1.18:

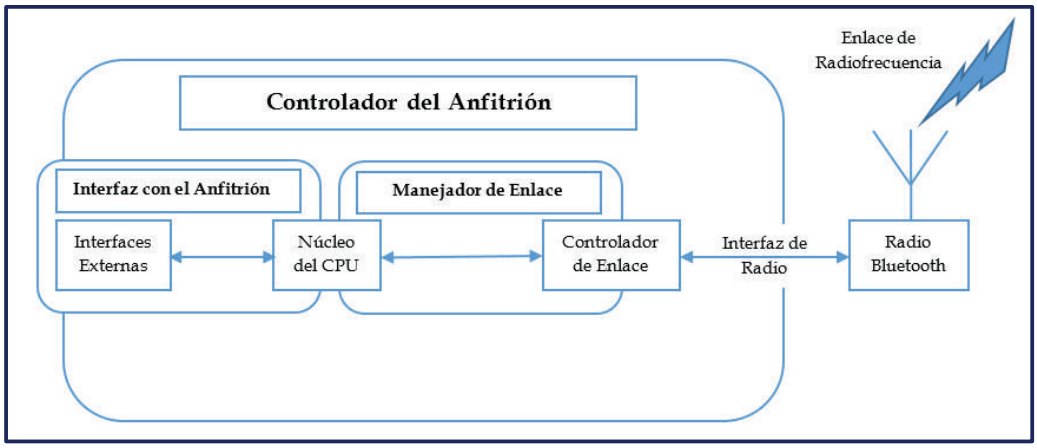



**Figura 1.18.** Arquitectura de Hardware de Bluetooth. [15]

*Dispositivo de Radio*. - Es un transceiver de radio frecuencia encargado de modular y transmitir la señal.

*Controlador Digital*. - Llamado también controlador de enlace banda-base (Link Controller), está compuesto por una CPU, un procesador de señales digitales, un software de gestión, y un subsistema de antena. Entre sus funciones se tiene:

Transferencia, codificación y cifrado de datos, así como el manejo de protocolos, y la interfaz con el dispositivo anfitrión.

La CPU para atender al anfitrión ejecuta un software llamado Link Manager que a su vez utiliza el protocolo LMP para poder comunicarse con otros dispositivos.

### 1.5.4 ARQUITECTURA DEL SOFTWARE

En la Figura 1.19 se observan los distintos niveles de protocolos en la estructura del software de la tecnología bluetooth y sus relaciones.

Los protocolos de alto nivel como RFCOMM, SDP, TSC interactúan entre si y se comunican con el controlador de banda base mediante el protocolo L2CAP.

La comunicación entre dispositivo anfitrión y el chip bluetooth se la realiza mediante una interfaz llamada HCI (Host Controller Interface).

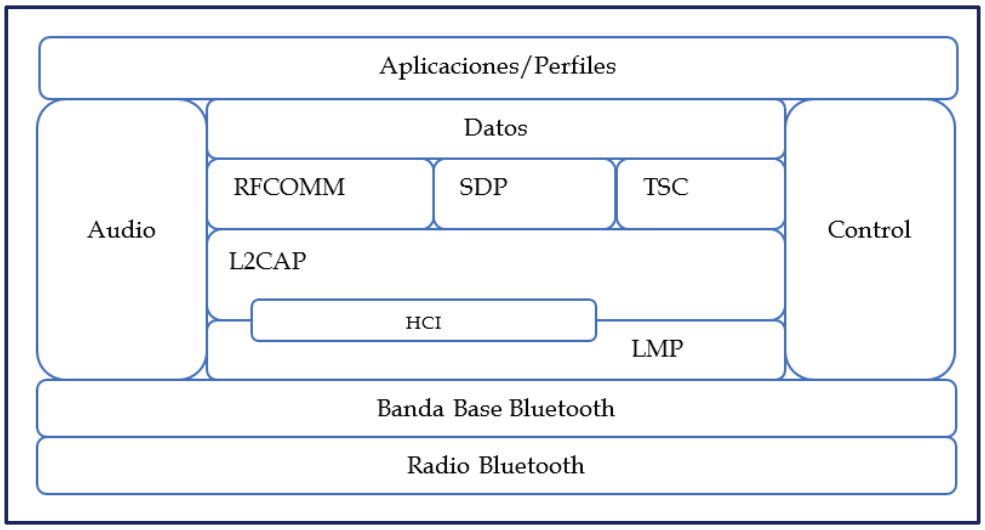



**Figura 1.19.** Arquitectura de Software de Bluetooth. [15]

*LMP*. - (Link Management Protocol). Configura y controla el enlace de radio entre dos dispositivos.

*L2CAP*.- (Logical Link Control and Adaptation Protocol). Se encarga de la Segmentación y re ensamblado de paquetes para su envío, permite manejar paquetes de hasta 64 kb.

*RFCOMM*. - (Radio Frequency Communication). Es un protocolo de transporte que permite emular 60 puertos serie RS-232 para conexiones simultaneas de dispositivos bluetooth.

*SDP*. - (Service Discovery Protocol). Es un protocolo usado para reconocer dispositivos bluetooth en un rango de comunicación, además de detectar que servicios soporta cada uno y que parámetros son necesarios para establecer la conexión.

*TSC*. - (Telephony Control Protocol Specification). Este protocolo es usado para configurar y controlar llamadas de voz y de datos entre dispositivos bluetooth.

## 1.6 SERVOMOTORES [17]

### 1.6.1 DEFINICIÓN

Los servomotores son dispositivos motorizados que en conjunto con aditamentos mecánicos y electrónicos permiten tanto el control de la velocidad como el control la posición angular del rotor dentro de un rango de operación definido por la construcción del mismo, este rango normalmente es de 0 a 180°.

Los Servomotores están formados básicamente por un motor eléctrico que puede ser de corriente directa o corriente alterna, un circuito de control de posición y un sistema de engranajes reductores para modificar la velocidad y el torque, como se puede observar en la Figura 1.20. En el presente trabajo se hablará únicamente de los motores de corriente continua.

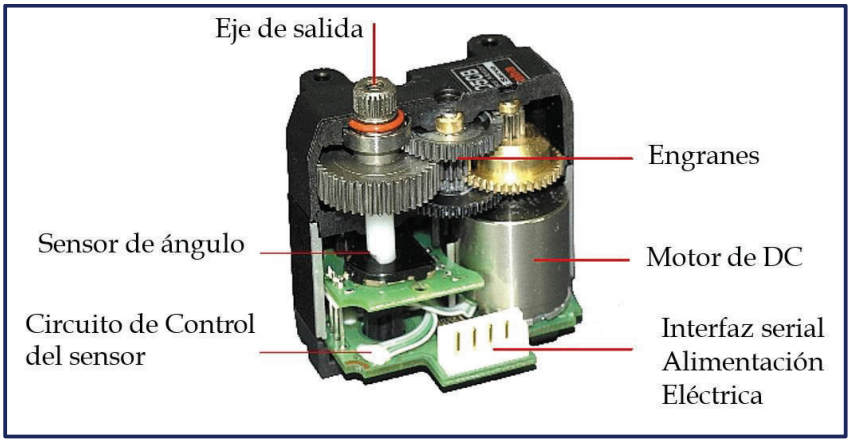



**Figura 1.20.** Partes constitutivas de un servomotor. [18]

El motor de DC hace girar su eje y genera el movimiento angular que es transferido al eje principal del servomotor a través del sistema de engranajes reductores, disminuyendo considerablemente la velocidad y aumentando el torque de salida del mismo.

Al eje principal esta acoplado un potenciómetro que cumple la función de un sensor de posición instantánea del eje y que junto con la tarjeta controladora cierran el lazo para el control de la posición angular del servomotor.

### 1.6.2 MODO DE FUNCIONAMIENTO

Para posicionar el eje del servomotor en un ángulo deseado se debe enviar un tren de pulsos con una frecuencia que puede estar entre 50 y 100 Hz, esto significa periodos de 10 a 20 ms.

La técnica usada para la generación del tren de pulsos es la PWM (Pulse-Width Modulation) o Modulación por ancho de Pulso, que básicamente es una señal cuadrada que permite variar su relación de trabajo (tiempo en alto) y por ende la cantidad de energía entregada.

La relación de trabajo de la PWM indica la posición angular a la que irá el eje del servomotor. El ancho del pulso tiene valores típicos para ir desde la posición de 0° hasta 180° que son 1 ms y 2 ms respectivamente, siendo la posición central o neutral 1,5 ms que corresponderían a 90° (Figura 1.21).

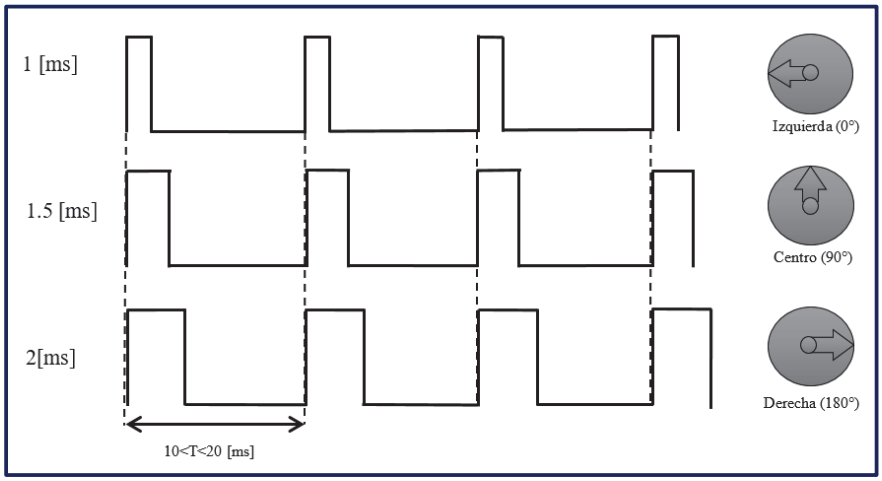



**Figura 1.21.** PWM, Control de posición del servomotor. [17]

Para que el servomotor mantenga una posición deseada el tren de pulsos debe enviarse continuamente caso contrario cualquier fuerza externa podría hacerlo perder la posición establecida.

## 1.7 MICROCONTROLADORES [19] [20]

### 1.7.1 DEFINICIÓN

Un microcontrolador es un circuito integrado que combina en su interior varias unidades similares a las de un computador, como son, unidad central de procesamiento (CPU), memorias y periféricos de entrada y salida.

Los microcontroladores están diseñados principalmente para el ámbito industrial, son de bajo costo, tamaño reducido y no procesan grandes cantidades de datos. Su fuente de alimentación es de voltaje continuo y trabajan con niveles de voltaje lógicos (datos binarios).

### 1.7.2 COMPONENTES INTERNOS

Un microcontrolador de forma general contiene una unidad central de procesamiento, memorias y periféricos de entrada y salida, todos ellos relacionados mediante un bus de comunicación. La Figura 1.22 muestra un diagrama de bloques general de los componentes de un microcontrolador:

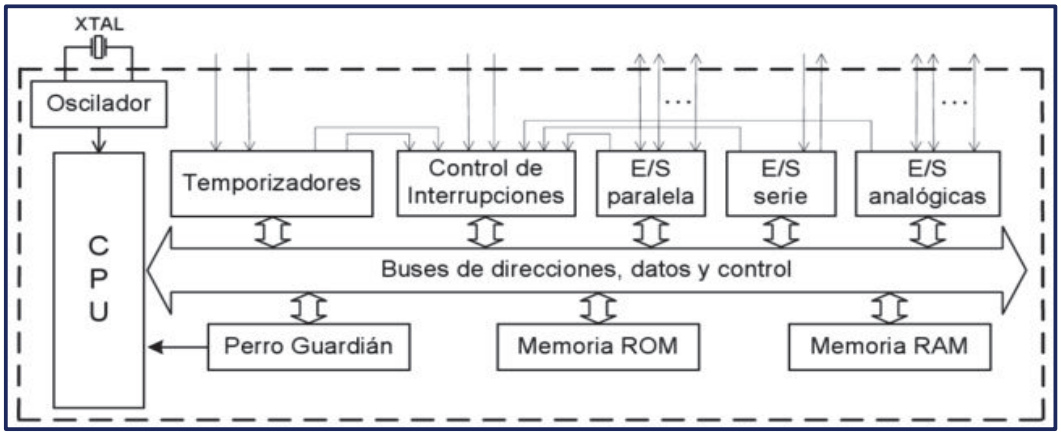



**Figura 1.22.** Diagrama de bloques general de un microcontrolador [20]

Un microcontrolador consta de un oscilador que es el encargado de generar pulsos a una determinada frecuencia. Estas señales permiten sincronizar todas las acciones y operaciones que se realizan dentro del mismo. Los osciladores son generalmente cristales de cuarzo que emiten señales con alta estabilidad en frecuencia.

La velocidad con la que se ejecutan las instrucciones en un microcontrolador depende directamente de la frecuencia de oscilación de dichos cristales. Así, por ejemplo, un cristal de oscilación de 16 MHz significa que el microcontrolador puede realizar 16 millones de ciclos de máquina o instrucciones, en un segundo.

Dentro de la unidad de procesamiento se encuentra la Unidad Aritmética Lógica o ALU, por sus siglas en inglés. Este encapsulado es el encargado de realizar las operaciones aritméticas básicas, las operaciones lógicas entre datos y las operaciones de desplazamiento de bits.

La CPU de un microcontrolador está formada, además, por una serie de registros tanto de propósito general, como específico, que permiten el procesamiento de datos; entre los más importantes están: Registro de Instrucciones, Registro de Direcciones de Datos y el Contador de programa.

El registro de instrucciones, es el encargado de almacenar la instrucción que en ese momento se encuentra realizando la CPU. El registro de Direcciones de Datos almacena la dirección en las que se encuentra determinada información en la memoria. El contador de programa, es el encargado de apuntar a la siguiente instrucción. Es decir, este contiene la dirección de la nueva instrucción, conforme el programa sigue ejecutándose.

Dentro del CPU también se encuentran las memorias. Un microcontrolador posee dos tipos de memorias: una memoria RAM (Ramdom Access Memory) y una memoria ROM (Read Only Memory). La memoria RAM, es una memoria de lectura y escritura, sin embargo, es del tipo volátil, por lo que se usa para almacenar datos temporales que eventualmente el programa los usa para manipular las variables. Por otro lado, la memoria ROM, es una memoria de solo lectura, no volátil, por lo que esta se usa para almacenar el programa que ejecutará el microcontrolador.

Es muy frecuente contar con una memoria tipo EEPROM, que es usada para almacenar datos constantes que usará el programa o datos de interés que el usuario desea mantener, aun cuando el microcontrolador se reinicie. Además, los microcontroladores tienen una memoria tipo FLASH que es usada para almacenar el programa principal.

Los circuitos de entrada y salida son los encargados de comunicarse con el exterior. Mediante estos se puede conectar al microcontrolador periféricos de entrada como un teclado o sensores y periféricos de salida como pantallas, relés o motores.

Un microcontrolador por lo general dispone de entradas y salidas, tanto analógicas como digitales. Además, posee conversores analógico-digital y viceversa. También permite establecer comunicación con otros dispositivos, para lo que usa puertos de comunicación serial, I2C, Ethernet, SPI, etc.

## 1.8 SENSOR, ACTUADORES Y MÓDULOS DE COMUNICACIÓN

A continuación, se detallan los elementos a utilizarse en el presente proyecto.

### 1.8.1 SENSOR PARA LA ADQUISICIÓN DE IMÁGENES

El sistema de control necesita una señal de realimentación para poder implementar el algoritmo del controlador. Esta señal se obtiene mediante visión artificial usando una cámara de video que toma imágenes de la plataforma y la esfera, permitiendo determinar su posición y así calcular la señal de control.

#### 1.8.1.1 Cámara Logitech C615 [21]

La cámara usada en el presente proyecto es la Logitech C615, que se muestra en la Figura 1.23.


**Figura 1.23.** Cámara Logitech C615 [21]

La cámara Logitech C615 posee las siguientes características técnicas:

- Resolución máxima de video: 1920x1080.
- Fotograma: 30 frames/s.
- Calidad de fotografía: 8 Mp.
- Enfoque automático de 20 pasos.
- Fuente de alimentación mediante conector USB.
- Plataformas de Hardware: Pc/Mac.
- Interfaz: USB 2.0 de alta velocidad.

### 1.8.2 ACTUADORES DEL SISTEMA

En todo sistema de control, para modificar cualquier variable de la planta o proceso es necesario poseer un actuador que modifique el valor de dicha variable en función de una señal de control previamente calculada. Para este caso, se requiere modificar la posición de la esfera en la plataforma, esto se logra a través de la variación del ángulo de giro de los servomotores que transfieren este desplazamiento angular al plano de la plataforma, el cual gira respecto al eje pivote.

#### 1.8.2.1 Servomotores Hitec HS-645MG [22]

Para este trabajo se usarán los servomotores Hitec Hs-645MG indicados en la Figura 1.24.


**Figura 1.24.** Servomotor Hitec Hs-645MG. [22]

Los Servomotor Hitec Hs-645MG poseen la distribución de pines mostrada en la Tabla 1.3:

**Tabla 1.3.** Distribución de pines – Servomotor Hitec Hs-645MG. [22]

| Color | Descripción |
|---|---|
| **Rojo** | Vcc (5 – 6 V) |
| **Negro** | Gnd |
| **Amarillo** | Control PWM |

Además, en la Tabla 1.4 se indican las principales características de los servomotores Hitec Hs-645MG.

**Tabla 1.4.** Características Servomotor Hitec Hs-645MG [22]

| Característica | Descripción |
|---|---|
| Modulación | Análoga (PWM) |
| Torque | 4.8[V]: 7.7[Kg/cm] |
| | 6.0[V]: 9.6[Kg/cm] |
| Velocidad | 4.8[V]: 0.24 s/60° sin carga |
| | 6.0[V]: 0.20 s/60° sin carga |
| Peso | 45.4[g] |
| Dimensiones | Largo: 40.4[mm] |
| | Ancho:19.6[mm] |
| | Alto :37.6[mm] |
| Tipo del Motor | Motor de 3-Polos |
| Sentidos de giro | Horario y anti horario |
| Rango de rotación | 180° |
| Periodo Máximo del ciclo | 20[ms] |
| Tiempo en alto | 1 a 2[ms] control de 0 a 180° |

### 1.8.3 MÓDULO DE COMUNICACIÓN SERIAL

Para realizar la conexión entre un computador y un micro controlador que manejan diferentes niveles de voltajes (Figura 1.25), es necesario un convertidor que traduzca la información de niveles TTL a niveles adecuados para el computador como por ejemplo RS-232 o USB.

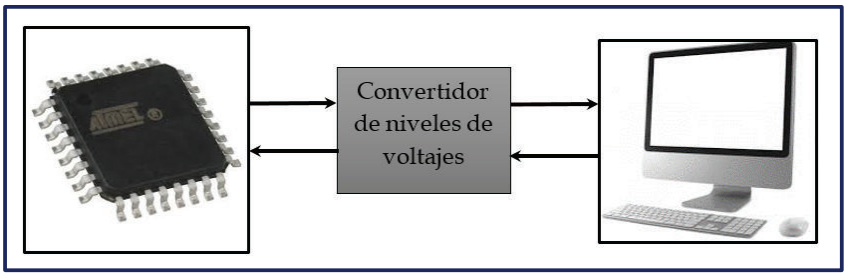



**Figura 1.25.** Interfaz de conversión de niveles de voltajes.

### 1.8.3.1 Módulo USB-TTL [23]

En el mercado existen módulos para establecer directamente una comunicación serial entre un dispositivo externo como un microcontrolador y un computador, estos módulos permiten implementar la interfaz de comunicación entre los dispositivos y adecuar los niveles de voltaje para la codificación de los estados logicos.

El módulo usado para la interfaz entre niveles lógicos TTL – USB es el BTE13-007 (Figura 1.26), que posee un conector USB tipo A y una velociadad 2.0 (hasta 60 MB/s), ademas del chip principal CP2102 para el manejo de los protocolos seriales y la conversion USB-RS232.

Este módulo crea un puerto COM virtual y la interfaz hacia la UART, que como se mencionó, es el dispositivo encargado del manejo de los puertos seriales en el computador.

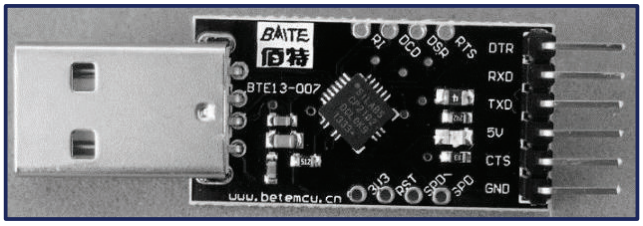



**Figura 1.26.** Modulo BTE13-007. [24]

El Módulo BTE13-007 tiene la siguiente distribución de pines mostrada en la Figura 1.27, los cuales se describen en la Tabla 1.5.


**Figura 1.27.** Distribucion de pines del modulo BTE13-007. [24]

**Tabla 1.5.** Descripcion de los pines del Modulo BTE13-007. [24]

| Pin | Descripción |
|---|---|
| DTR | Salida de control: Terminal de datos listo |
| RXD | Entrada de datos asíncrona (UART Receive) |
| TXD | Salida de datos asíncrona (UART Transmit) |
| VCC | Salida de voltaje 5 [V] |
| CTS | Entrada de control: Buffer vacío para el envío |
| GND | Referencia de datos, tierra de fuente de 5[V] |

De todos los pines descritos, normalmente se usan la fuente de alimentación, si la aplicación lo requiere (en caso de no tener alimentación independiente), y los pines de envío y recepción. En caso de no usar la fuente, obligatoriamente se deben referir las señales de voltaje enviadas o recibidas usando el pin de GND.

Además, el módulo BTE13-007 presenta las características mostradas en la Tabla 1.6.

**Tabla 1.6.** Características técnicas del Módulo BTE13-007. [24]

| Carácterística | Descripción |
|---|---|
| Voltaje de alimentación | 3.3 a 5 [V] |
| Velocidad Mínima | 300 bps |
| Velocidad Máxima | 1.5 Mbps |
| Bufer de Recepción | 576 bytes |
| Bufer de Envío | 640 bytes |
| Rango de Temperatura | -40 a 85 °C |
| Sistemas Operativos Compatibles | Windows, Mac y Linux |

### 1.8.4 MÓDULO DE COMUNICACIÓN BLUETOOTH

Para establecer una comunicación inalámbrica se necesita una interfaz que convierta las señales de voltajes (datos que se desean enviar) a señales de radiofrecuencia (espectro electromagnético), y además que sea capaz de leer estas señales cuando lleguen (datos recibidos) y convertirlas en información de tal forma que puedan ser interpretadas de nuevo por algún dispositivo. La Figura 1.28, muestra la idea general de esta comunicación.

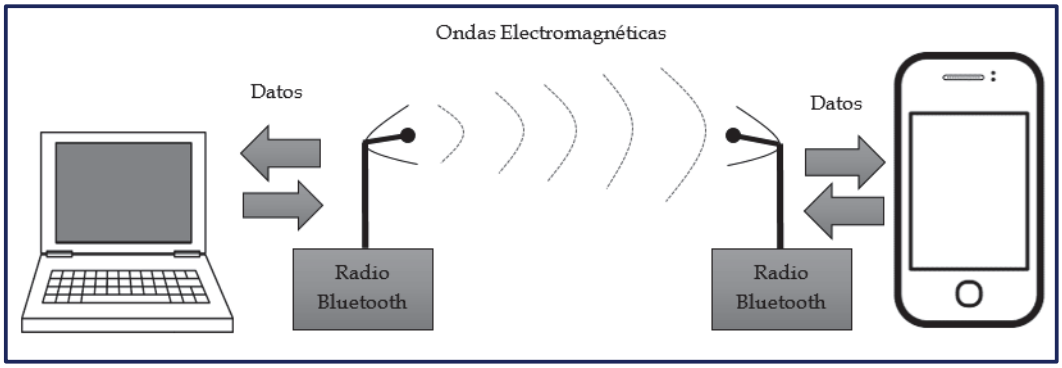



**Figura 1.28.** Enlace de Radiofrecuencia

En este caso los dispositivos que se necesitan comunicar de forma inalámbrica son un ordenador (PC) y un Smartphone. Como este último posee integrado un dispositivo bluetooth entonces solo se requiere un módulo bluetooth que se conecte con el ordenador y así poder establecer la comunicación.

### 1.8.4.1 Módulo bluetooth HC-06 [25]

El módulo bluetooth HC-06 (Figura 1.29) es un dispositivo que permite enviar y recibir datos vía inalámbrica desde cualquier dispositivo (Microcontrolador, Pc) siempre y cuando se puedan conectar entre ellos (Modulo bluetooth-Dispositivo).

Este módulo solo permite trabajar en modo esclavo, es decir, otro dispositivo debe asumir el papel de maestro, normalmente el dispositivo que inicia la comunicación se convierte en el maestro, lo que para este caso es útil ya que el maestro es el Smartphone quien inicia la comunicación cuando requiere enviar o recibir datos desde el ordenador.

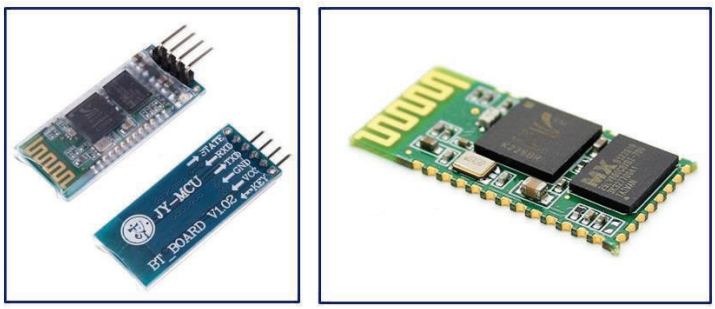



**Figura 1.29.** Módulo Bluetooth HC-06 [25]

La configuración del módulo HC-06, se la realiza mediante comandos AT (comandos usados para configurar módems), los cuales permiten cambiar parámetros como la velocidad de transmisión, el número de bits de datos, bit de paridad, nombre del dispositivo, entre otras cosas. En la Tabla 1.7 de detallan los comandos para su configuración.

Los parámetros que vienen por defecto son:

Usuario: linvor, Clave: 1234, Baudrate: 9600,8,N,1.

**Tabla 1.7.** Comandos AT para configurar el Módulo Bluetooth HC-06 [25]

| Nombre del Comando AT | Se envía | Se recibe |
|---|---|---|
| **Prueba de funcionamiento** | AT | OK |
| **Baud Rate** | AT+BAUD<Número> | OK<Baudrate> |
| **Cambiar nombre del dispositivo** | AT+NAME<Nombre> | OK Setname |
| **Cambiar clave de emparejamiento** | AT+PIN<Clave> | OK <Clave> |
| **Obtener versión del Firmware** | AT+VERSION | Linvor 1.8 |

La variable “Número” para configurar la velocidad de transmisión es un valor hexadecimal entre 1 y C que indica las velocidades en baudios listadas en la Tabla 1.8.

**Tabla 1.8.** Velocidades de transmisión del Módulo Bluetooth HC-06 [25]

| Velocidades de trasmisión del módulo bluetooth HC-06 | | | |
|---|---|---|---|
| 1 – 1200 | 2 – 2400 | 3 – 4800 | 4 – 9600 |
| 5 – 19200 | 6 – 38400 | 7 – 57600 | 8 – 115200 |
| 9 – 230400 | A – 460800 | B – 921600 | C – 1382400 |

Este módulo bluetooth HC-06 tiene cuatro pines para conectarse a otro dispositivo. En este punto hay que recalcar que, los niveles de voltaje que maneja este módulo son niveles TTL, es decir que para poder conectarlo al computador es necesario un conversor de TTL- USB como el usado para la comunicación serial entre Arduino y el ordenador. En la Tabla 1.9 se detallan los pines del módulo bluetooth HC-06.

**Tabla 1.9.** Distribución de pines – Módulo Bluetooth HC 06 [25]

| Pin | Descripción |
|---|---|
| **Vcc** | Fuente de alimentación del módulo |
| **Gnd** | Tierra |
| **TXD** | Salida de datos |
| **RXD** | Entrada de datos |

Además, el módulo bluetooth HC-06 posee las siguientes características:

- Voltaje de alimentación: 3.3 a 5 V.
- Banda de frecuencia de trabajo: 2400 a 2483.5 MHz.
- Frecuencia de saltos: 1600saltos/seg.
- Rango de Comunicación: 10 m
- Temperatura de Operación: -10 a 45 °C.
- Especificación Bluetooth: V1.1, V1.2, V2.0.
- Velocidad de transmisión: 1200bps a 1.3Mbps.

### 1.8.5 MICROCONTROLADOR

Para ejecutar la acción de control sobre los actuadores (servomotores), es necesario un microcontrolador. Este se encargará de recibir, vía comunicación serial los valores de los ángulos que deben girar los ejes de los servomotores y comandar la señal (PWM) que los controla.

#### 1.8.5.1 Microcontrolador ATmega 328 [26]

En base a estas características, es necesario un microcontrolador que posea: 1 puerto serial y 2 salidas PWM. El microcontrolador usado en el presente proyecto es el ATmega 328 de la marca Atmel (Figura 1.30).

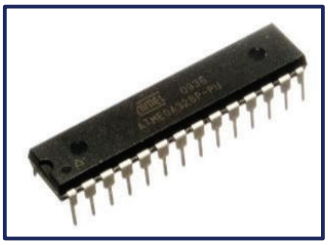


**Figura 1.30.** Microcontrolador ATmega 328. [26]

#### 1.8.5.2 Placa Arduino Uno [27]

En el mercado se puede encontrar microcontroladores como el ATmega 328 incorporados en una placa que facilita el acceso a sus pines, como es el caso de la placa Arduino Uno. La compañía Arduino además, facilita la programación de estos, con un entorno de desarrollo (IDE), cuyo software permite la programación de

microcontroladores combinando un lenguaje de alto nivel (basado en lenguaje C) y un lenguaje de bajo nivel (Ensamblador).

La placa que se usa en este proyecto es la *Arduino Uno***.** La Figura 1.31 muestra una vista de la placa y sus pines disponibles:

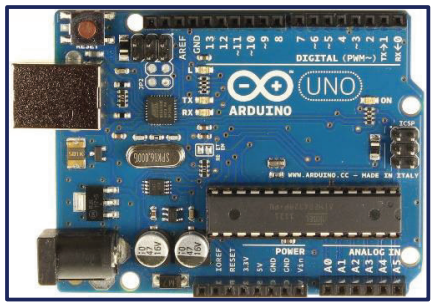



**Figura 1.31.** Placa Arduino Uno. [27]

En la Tabla 1.10 se muestra la distribución de pines de la placa Arduino Uno:

**Tabla 1.10.** Distribución de pines Placa Arduino Uno [27]

| Descripción | Pines |
|---|---|
| Pines analógicos | A0-A5 |
| Pines digitales | 0-13 |
| Comunicación serial | 0(RX) y 1(TX) |
| Salidas PWM | 3,5,6,9,10,11 |

En la Tabla 1.11, se presentan las características más relevantes de la placa Arduino Uno.

**Tabla 1.11.** Principales características de la Placa Arduino Uno. [27]

| Característica | Descripción |
|---|---|
| Microcontrolador | ATmega 328P |
| Voltaje de funcionamiento | 5V |
| Voltaje de Entrada (Recomendado) | 7 – 12 V |
| Voltaje de Entrada (Límites) | 6 – 20 V |
| Pines Digitales (I/O) | 14 (6 PWM) |
| Pines Analógicos | 6 |
| Corriente DC por pin (I/O) | 40 mA |
| Memoria Flash | 32 KB |
| SRAM | 2 KB |
| EEPROM | 1 KB |
| Puertos Seriales | 1 |
| Cristal de Oscilación | 16 MHz |
| Timers disponibles | 3 |

# CAPÍTULO 2
# IMPLEMENTACIÓN DEL HARDWARE

En esta sección se detallan las características más relevantes en la implementación de la plataforma, así como también se describe, de manera específica, cada uno de los componentes que forman parte de la plataforma, tanto mecánicos como electrónicos.

## 2.1 DESCRIPCIÓN DEL PROCESO Y SUS COMPONENTES

El sistema implementado está constituido por una cámara de video que actúa como sensor para la visualización de la posición de la esfera sobre la plataforma. La información obtenida por la cámara llega a un computador, el mismo que procesa esta información y calcula la señal de control. Mediante comunicación serial y a través de una placa Arduino, se toma acción sobre los servomotores que funcionan como actuadores transmitiendo su movimiento angular a la plataforma mediante extensiones metálicas fijadas entre el eje del motor y la parte inferior de la plataforma. La Figura 2.1 muestra de forma general las principales partes constitutivas del sistema.

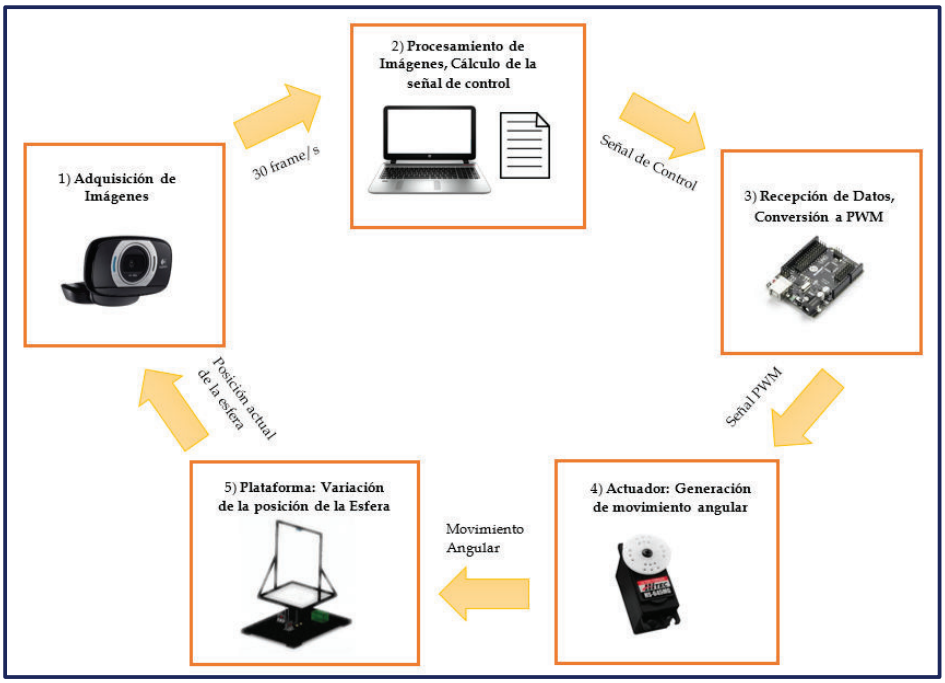



**Figura 2.1.** Principales partes constitutivas del sistema

## 2.2 SISTEMA MECÁNICO

El sistema está compuesto de una plataforma donde rodará la bola, misma que descansa sobre un eje central que, además, funciona como pivote para permitir el movimiento en todas las direcciones. El sistema consta de dos grados de libertad, por lo tanto, se compone de dos servomotores sujetos mediante soportes a la base. Adicionalmente, el sistema tiene extensiones para transmitir el movimiento angular de los servomotores a la plataforma. La cámara está montada en una estructura metálica paralela a la plataforma. La Figura 2.2 detalla cada una de las partes y piezas del sistema mecánico.

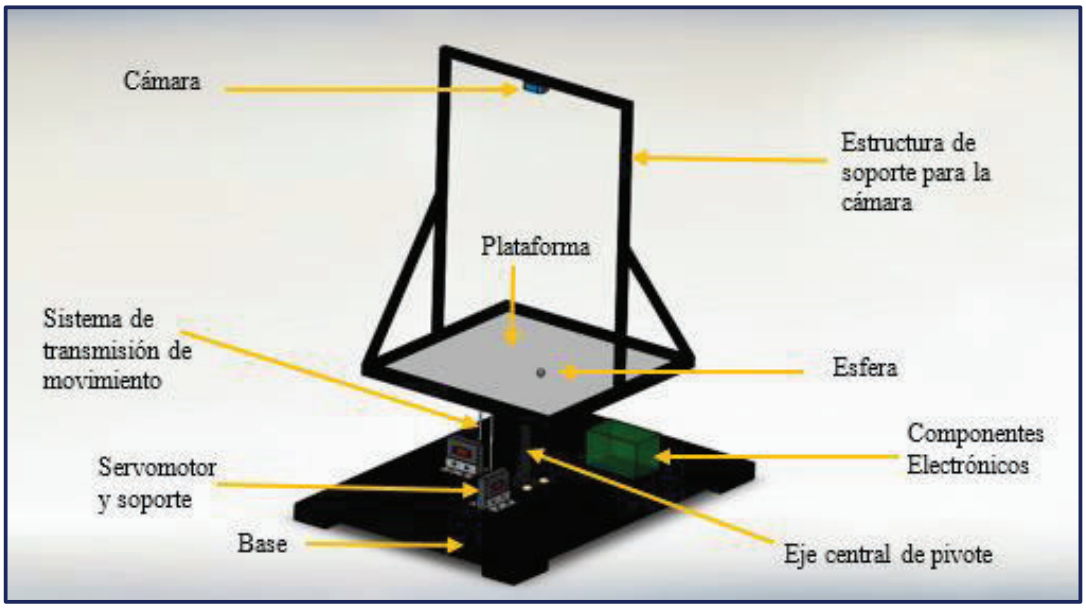



**Figura 2.2.** Partes del sistema mecánico

### 2.2.1 PLATAFORMA Y ESFERA

Con el objetivo de optimizar la ubicación de la esfera sobre la plataforma, se escogió colores fácilmente diferenciables, que además permiten tener un mayor contraste entre la esfera y la plataforma.

La esfera es sólida, de acero, de color negro, con un diámetro de 2.5 cm y una masa de 500 gramos. La plataforma principal, es de madera, de dimensiones 60 x 60 cm y de color blanco. En la Figura 2.3 se muestra la esfera y la plataforma principal.

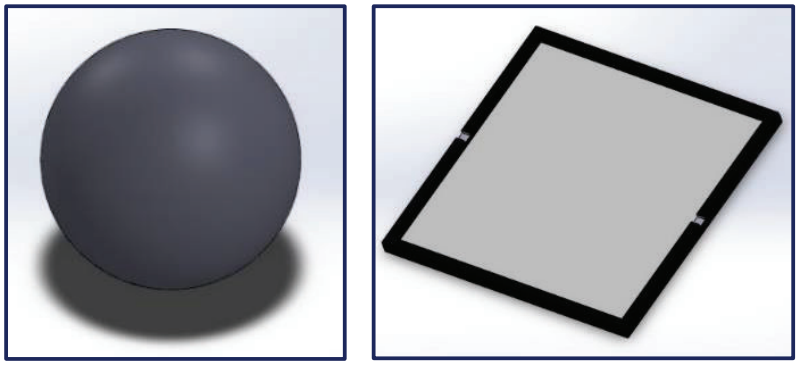


**Figura 2.3.** Esfera y plataforma

### 2.2.2 EJE CENTRAL DE PIVOTE

El eje central debe permitir todos los movimientos a manera de pivote y además es quien soporta la mayor cantidad del peso de la plataforma. El material de construcción es de acero y se sujeta a la base. La estructura implementada se muestra en la Figura 2.4.

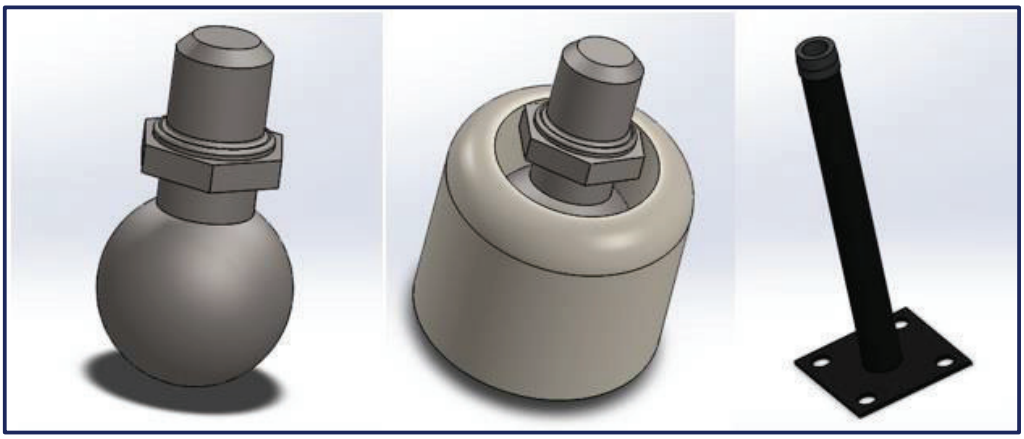


**Figura 2.4.** Eje central de pivote

Además, para evitar que la plataforma gire en sentido del eje yaw (movimiento no deseado) y cambie así la acción de los servomotores, se limitó ese movimiento con el soporte mostrado en la Figura 2.5.

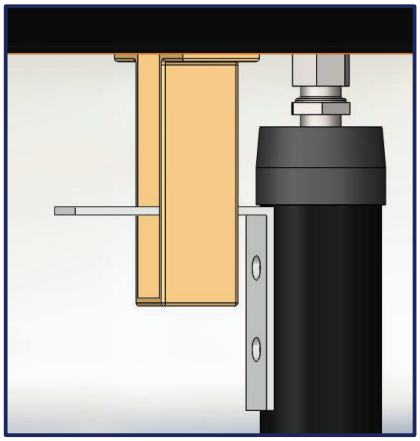


**Figura 2.5.** Corrección de movimiento de la plataforma en el eje yaw.

### 2.2.3 SISTEMA DE TRANSMISIÓN DE MOVIMIENTO

Para transmitir el movimiento de los servomotores hacia la plataforma se ha utilizado una varilla limitada por cauchos, que permiten al sistema ceder ligeramente, ya que la acción de un eje, influye en la posición del otro. El sistema que transmite el movimiento angular de los servomotores a la plataforma, se muestra en la Figura 2.6.

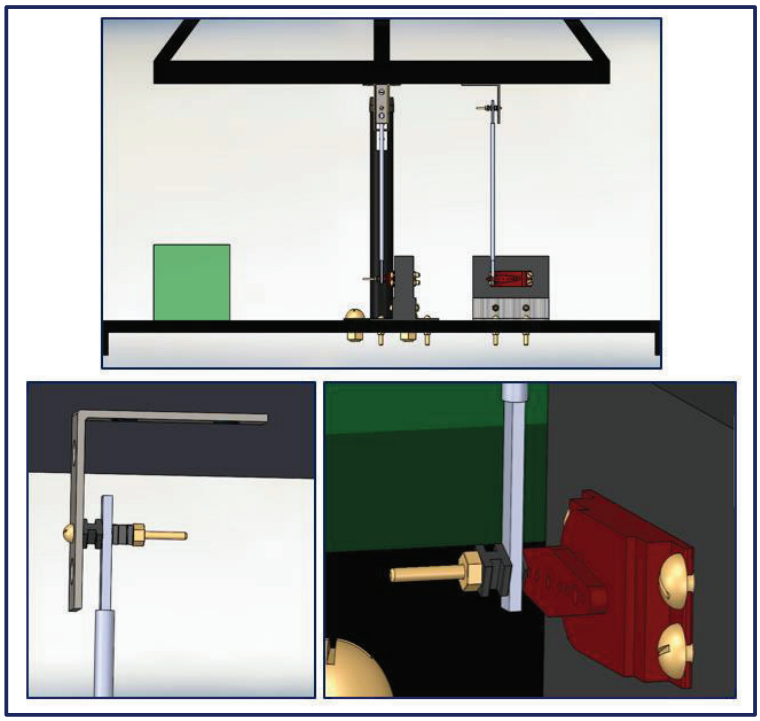


**Figura 2.6.** Transmisión de movimiento a la plataforma

### 2.2.4 SERVOMOTORES Y SOPORTES

Los actuadores de la planta deben permitir variaciones pequeñas de movimiento y además ser rápidos en respuesta. Se escogió servomotores para transferir el desplazamiento angular al plano de la plataforma, el cual gira respecto al eje pivote.

Los servomotores se especifican según el torque que se requiere (que es función del peso de la plataforma que van a mover y la distancia al eje central), para este caso se necesita mover la plataforma que tiene las características que se muestran en la Tabla 2.1:

**Tabla 2.1.** Características de la plataforma.

| Característica | Descripción |
|---|---|
| **Dimensiones** | Largo : 60[cm] |
| | Ancho: 60[cm] |
| | Espesor: 9[mm] |
| **Masa Total** | 6 [Kg] |
| **Plataforma** | Madera |
| **Soporte** | Aluminio |

En los cálculos de dimensionamiento, se incluye también el peso de la estructura de soporte de la cámara. Así como también el peso de la esfera.

La relación entre la masa total y la distancia al eje del servomotor es:

$$\gamma = \frac{6\ Kg}{2.5\ cm} = 2.4\frac{kg}{cm}$$

Por lo tanto, se ha decidido utilizar servomotores Hitec Hs-645MG, que permiten un $\gamma = 9.6\ \mathrm{Kg/cm}$ (a 6V) y adicionalmente, poseen engranaje metálico, para asegurar que el peso conjunto de la plataforma sea movido sin problemas y que la fuerza ejercida en el eje del servomotor no vaya a romper los engranajes internos.

Debido al peso de la plataforma, los motores tienen alto torque y por tanto requieren un soporte que los mantengan fijos e inmóviles para evitar vibraciones. La Figura 2.7 muestra las estructuras de soporte para los servomotores, implementadas en madera y sujetas a la base.

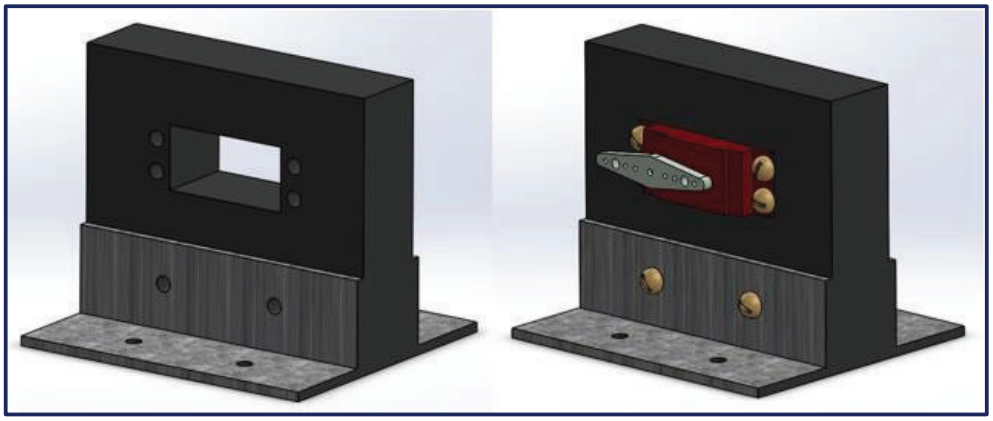


**Figura 2.7.** Soporte para servomotores

### 2.2.5 CÁMARA Y ESTRUCTURA DE SOPORTE

La cámara usada como sensor es la Logitech C615. Está cámara se ha escogido por su buena resolución y capacidad de tomar 30 fotogramas por segundo.

Se ha decidido montar la cámara sobre la plataforma para que se mueva paralela a ella. Esto permite evitar errores en la medida producto de la inclinación de la plataforma ya que, de no ser así, sería necesario compensar la posición de la esfera en el eje z. La estructura de soporte para la cámara y su montaje sobre la plataforma se muestran en la Figura 2.8.

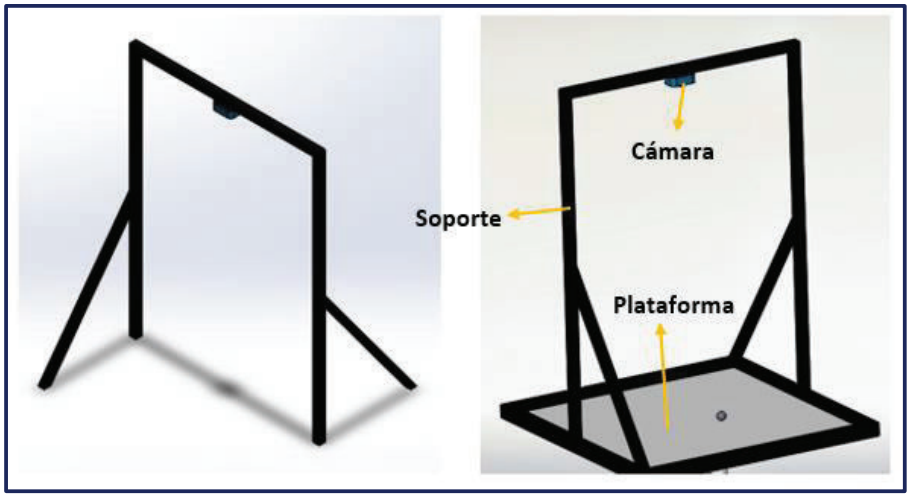



**Figura 2.8.** Estructura de soporte para la cámara.

## 2.3 COMPONENTES ELECTRÓNICOS

En la plataforma también se encuentra una caja que contiene los componentes electrónicos necesarios para el funcionamiento del sistema. La Figura 2.9 detalla todos los componentes y las conexiones que funcionan de interfaz entre el software y el hardware.

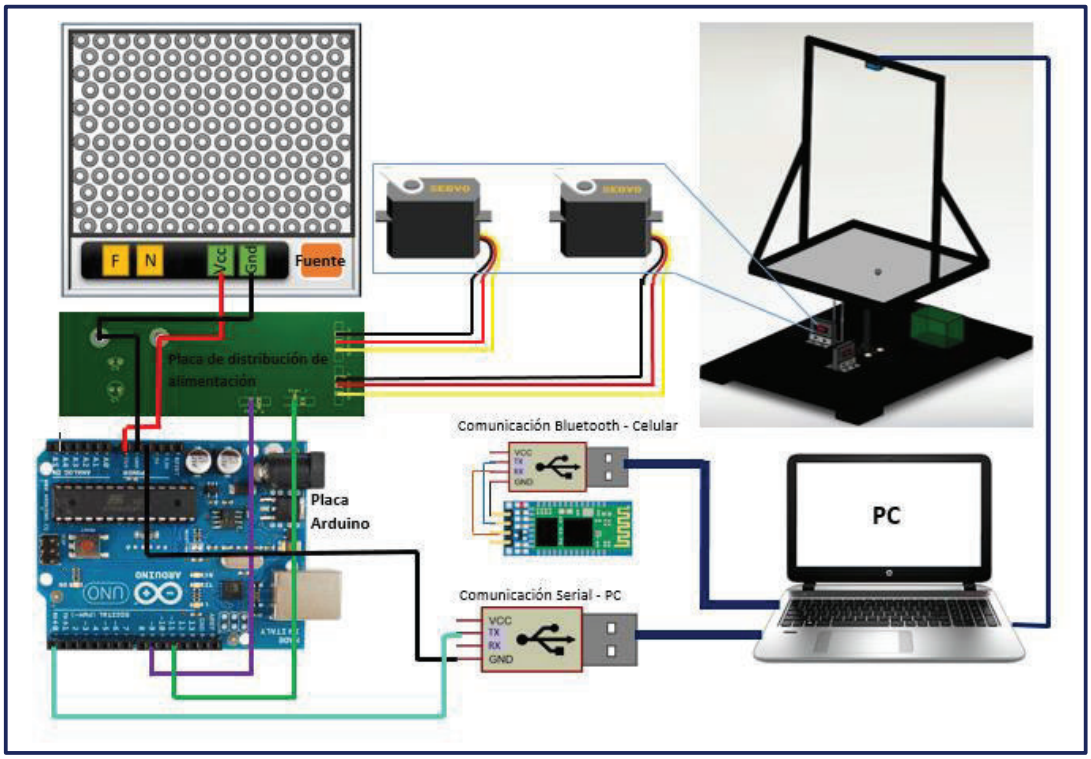



**Figura 2.9.** Esquema de conexiones del circuito de control.

### 2.3.1 FUENTE DE ALIMENTACIÓN

La fuente de alimentación tiene un voltaje de salida de 6V DC. Este voltaje es necesario para que los servomotores trabajen a su máximo torque y velocidad, según las especificaciones del fabricante. De igual manera, se utiliza la misma fuente para alimentar a la placa Arduino.

La corriente máxima proporcionada por la fuente es de 7 A, suficiente para suplir a los servomotores y al sistema de control. La fuente utilizada se muestra en la Figura 2.10 y se conecta a una toma de voltaje alterno de 120 V.


**Figura 2.10.** Fuente de alimentación (6V DC).

### 2.3.2 PLACA DE DISTRIBUCIÓN DE LA FUENTE DE ALIMENTACIÓN

Con el objetivo de distribuir mejor los cables y conexiones, se diseñó una placa para conectar de forma directa los servomotores y la fuente de alimentación. A esta placa llega la fuente de alimentación, y los pines de los servomotores además que se conecta la señal de control.

Para evitar posible ruido electromagnético en la alimentación del circuito, se colocaron dos capacitores; un electrolítico de 47 uF para ruidos de baja frecuencia y un cerámico de 0.01 uF para ruidos de alta frecuencia. El diseño de la placa de distribución se muestra en la Figura 2.11.

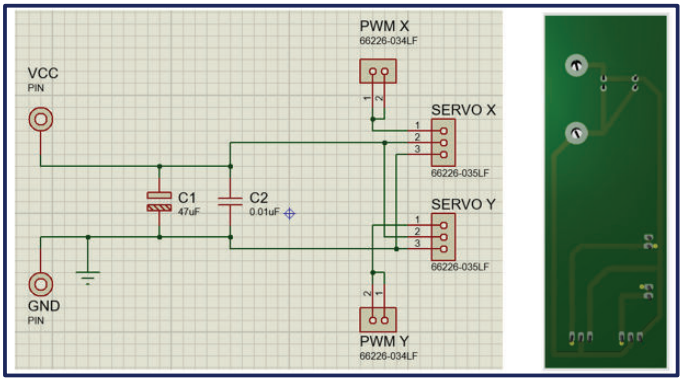



**Figura 2.11.** Placa de distribución de la fuente de alimentación.

### 2.3.3 MÓDULOS DE COMUNICACIÓN

Para establecer la comunicación serial entre el PC y la placa Arduino, se ha utilizado el módulo BTE 13-007, que es un módulo estándar para comunicación serial.

Para comunicar al PC con un Smartphone, se utilizó un módulo de comunicación serial conectado a un módulo bluetooth. Esta conexión se realiza ya que el módulo bluetooth trabaja con niveles de voltaje TTL en sus pines y es necesario un módulo serial adicional para poder comunicar al PC con el módulo bluetooth. La Figura 2.12 muestra esta conexión.

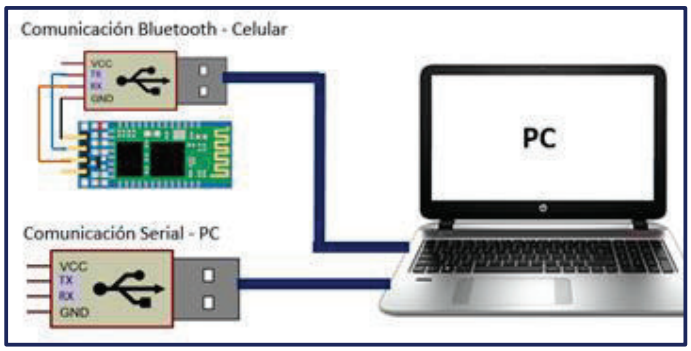



**Figura 2.12.** Conexión de los módulos de comunicación

## 2.4 IMPLEMENTACIÓN FÍSICA

En la Figuras 2.13 se muestra la implementación física de la plataforma con todos sus componentes.

**Figura 2.13.** Implementación física – Sistema completo

# CAPÍTULO 3.
# DESARROLLO DEL SOFTWARE

En este Capítulo se detallará el proceso del diseño del sistema de control, que permite a la esfera seguir un camino o situarse en una posición determinada. Además, se explica el proceso del desarrollo de las interfaces de usuario, que permitirán controlar el sistema, esto es, definir rutas o caminos, así como establecer puntos específicos como referencias de posición para la esfera en la plataforma.

El diagrama de la Figura 3.1, indica de manera general como está constituido el sistema, la parte física: plataforma, servomotores, cámara, placa Arduino y la parte lógica que esta implementada en el computador: Procesamiento de las imágenes, filtro de Kalman y controlador:

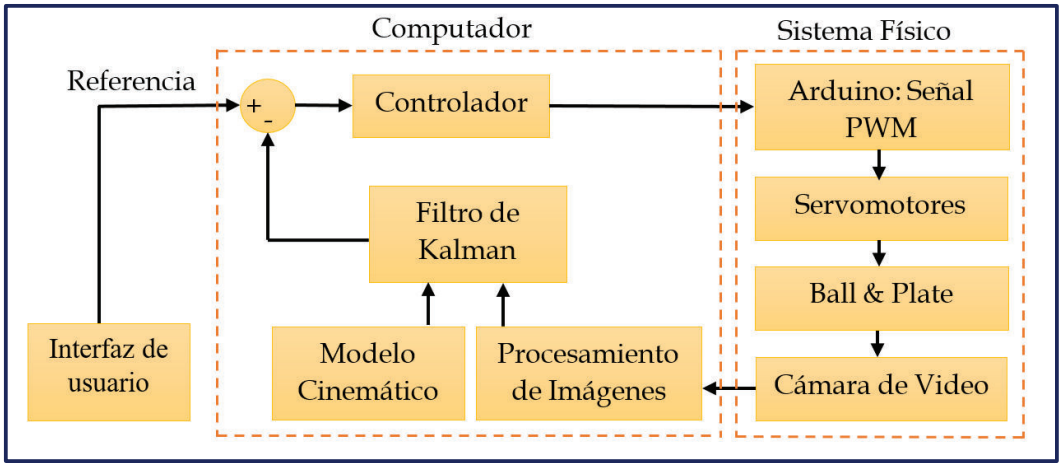



**Figura 3.1.** Diagrama general del sistema.

## 3.1 PROCESAMIENTO DE IMÁGENES

### 3.1.1 ALGORITMO DE VISIÓN ARTIFICIAL

La visión artificial es una parte muy importante del proyecto porque de esta depende la realimentación del sistema, sin embargo, también es crítica ya que está ligada al tiempo de ejecución del lazo principal, el mismo que debe ser lo más rápido posible para poder controlar la esfera.

La cámara de video utilizada permite tomar 30 fotogramas por segundo (30 fps), que son equivalentes a 33 milisegundos entre cada captura. Con el objetivo de mantener este número de fps, se requiere un algoritmo de visión lo más simplificado posible.

El primer recurso usado es aprovechar los colores binarios. Como se presentó en el diseño del hardware, la esfera es de color negro, y la plataforma es de color blanco. Esta acción simplifica cualquier segmentación por color. La Figura 3.2 presenta una captura del cuadro inicial tomado por la cámara, antes de realizar el procesamiento de la imagen.


**Figura 3.2.** Imagen tomada sin procesamiento.

Otro recurso utilizado es limitar la región de interés. Con esto se consigue eliminar cualquier objeto exterior a la plataforma, reducir el área a analizar al mínimo y fijar el área de interés como un cuadrado. La resolución usada por la cámara es de 640 x 480 pixeles antes del recorte y 460 x 460 pixeles después del recorte. La Figura 3.3 muestra el resultado obtenido después de realizado el recorte.


**Figura 3.3.** Imagen recortada en la región de interés.

También se ha aprovechado las propiedades de la cámara como son el brillo y el contraste. Estas propiedades mejoran la imagen al eliminar sombras, además de resaltar los colores blanco y negro creando una mayor diferenciación entre la esfera y la plataforma.

Estas propiedades simplifican el algoritmo de visión, ya que, de no ser así, se deberían aplicar filtros para conseguir resultados similares.

Los valores utilizados son: Brillo = 181 y Contraste = 252. La Figura 3.4 muestra el resultado obtenido después de realizada la configuración de brillo y contraste.


**Figura 3.4.** Imagen aplicada la configuración de brillo y contraste.

Todas las acciones mencionadas anteriormente no necesitan ejecutarse en cada lazo del bucle principal, solo se configuran una vez al inicio del programa.

En el lazo principal, el primer paso del algoritmo de visión consiste en capturar una imagen. La función de Matlab usada es *getsnapshot.*

La imagen a capturar está en RGB, por lo que la siguiente acción a tomar es transformar la imagen a escala de grises. La función que ofrece Matlab es *rgb2gray.* La Figura 3.5 muestra el resultado obtenido después de transformar la imagen RGB a escala de grises.

Como valores predeterminados, Matlab utiliza las siguientes constantes, referidas a la ecuación (1.1):

$$IEG\ (m,n) = \ 0.2989(Rojo(m,n)) + 0.5870(Verde(m,n)) + 0.1140(Azul(m,n))$$


**Figura 3.5.** Imagen en escala de grises.

Para poder extraer el centroide de la esfera, se requiere primero binarizar la imagen. Se utilizó un umbral del 40% de 255 (102), esto por seguridad, para eliminar la posibilidad de encontrar sobras que no hayan sido eliminadas por las configuraciones anteriores. La función usada es *im2bw.* Esta función tiene las características de la ecuación (1.2). La Figura 3.6 muestra el resultado obtenido al binarizar la imagen.

$$Binarización\ (m,n) = \begin{cases} 0 & si\ \ Img\ (m,n) < 102 \\ 1 & si\ \ Img\ (m,n) \geq 102 \end{cases}$$

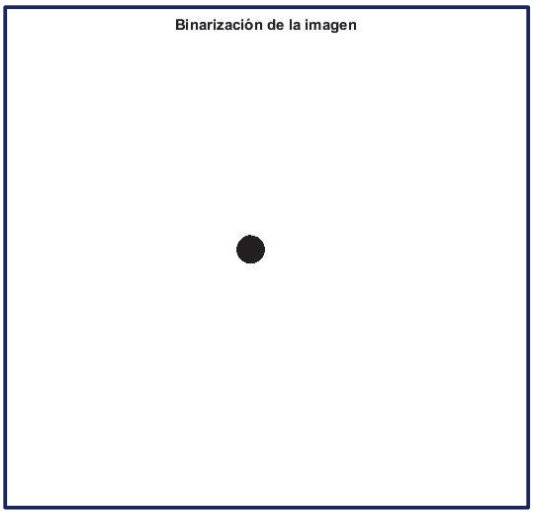



**Figura 3.6.** Imagen en binario.

Como se observa, la imagen es ideal para extraer el centroide de la esfera. Sin embargo, por características del comando de Matlab (regionprops) que ubica el centroide, la imagen debe tener fondo negro y el objeto, bordes blancos. Para conseguir esto, se obtiene el complemento de la imagen. La Figura 3.7 muestra el resultado obtenido después de aplicar el complemento a la imagen previa.


**Figura 3.7.** Complemento de la imagen.

Finalmente, se extrae el centroide de la esfera, para ello se ha usado la función *regionprops* que permite obtener las coordenadas x,y del centro de la esfera. En la Figura 3.8, se muestran algunos resultados obtenidos.

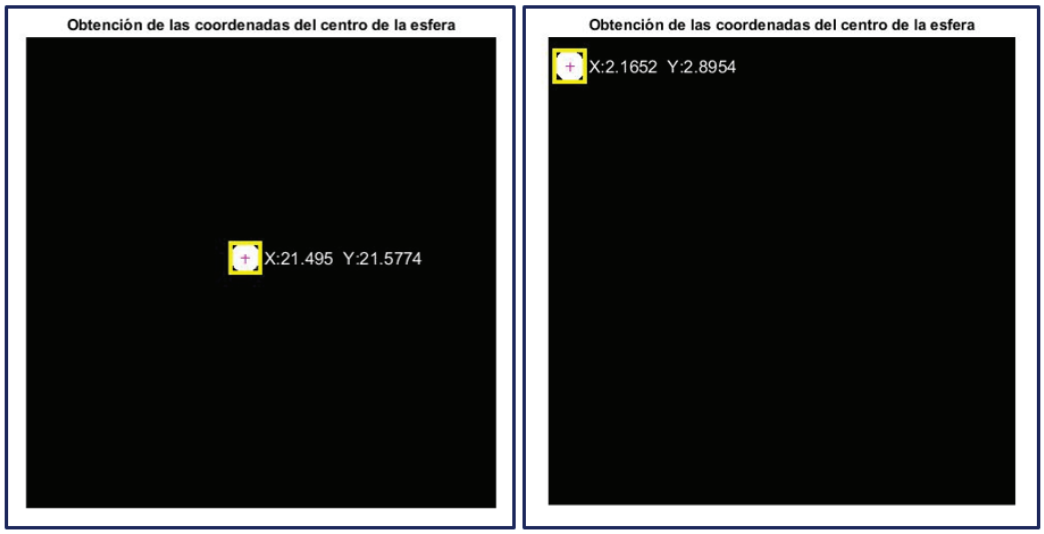



**Figura 3.8.** Coordenadas del centro de la esfera.

## 3.2 IMPLEMENTACIÓN DEL FILTRO DE KALMAN

### 3.2.1 JUSTIFICACIÓN

En el desarrollo del proyecto, debido a la implementación de la cámara montada sobre la plataforma se presentaron inconvenientes en la señal de realimentación del sistema. Debido a la acción de los servomotores, la cámara se agita, mayormente en el eje X, por lo que la posición medida por la cámara oscila dando valores erróneos, provocando que el controlador oscile de igual manera y el sistema no se estabilice.

### 3.2.2 MODELO DE LA ESFERA EN MOVIMIENTO [28]

Para implementar el filtro de Kalman se utiliza el modelo de un cuerpo con movimiento rectilíneo uniformemente variado (MRUV) sobre un plano x,y cuyo diagrama se muestra en la Figura 3.9.

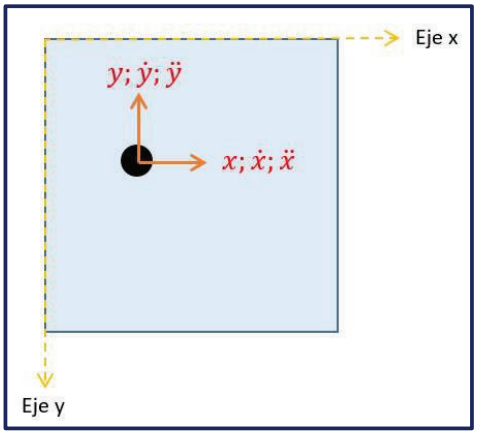



**Figura 3.9.** Movimiento de la esfera en el plano

Por lo tanto, las ecuaciones que describen el movimiento son:

$$x_k = x_{k-1} + \dot{x}_{k-1}T + \frac{1}{2}\ddot{x}T^2 \tag{3.1}$$

$$y_k = y_{k-1} + \dot{y}_{k-1}T + \frac{1}{2}\ddot{y}T^2 \tag{3.2}$$

$$\dot{x}_k = \dot{x}_{k-1} + \ddot{x}\,T \tag{3.3}$$

$$\dot{y}_k = \dot{y}_{k-1} + \ddot{y}\,T \tag{3.4}$$

Las ecuaciones anteriores escritas en forma matricial son:

$$\begin{bmatrix} x_k \\ y_k \\ \dot{x}_k \\ \dot{y}_k \end{bmatrix} = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \dot{x}_{k-1} \\ \dot{y}_{k-1} \end{bmatrix} + \begin{bmatrix} T^2/2 & 0 \\ 0 & T^2/2 \\ T & 0 \\ 0 & T \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix} \tag{3.5}$$

Matriz A Matriz B

La realimentación del sistema es únicamente de posición, por lo tanto solo se mide x,y, de modo que la matriz de correlación de valores H se define como:

$$\hat{z}_k = H\,\hat{x}_k \tag{3.6}$$

$$\rightarrow \quad H = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{bmatrix} \tag{3.7}$$

### 3.2.3 MATRICES PARA EL FILTRO DE KALMAN [6]

Una vez que se determinó el modelo, se definen las matrices de covarianza de la siguiente manera:

*Matriz de covarianza de la medición:*

$$R = \begin{bmatrix} {\sigma_x}^2 & 0 \\ 0 & {\sigma_y}^2 \end{bmatrix} \tag{3.8}$$

Usando el método heurístico se determinó el valor de la covarianza de la medición:

$$\sigma_x = \sigma_y = 11.6$$

*Matriz de covarianza del modelo:*

$$Q = \begin{bmatrix} T^4/4 & 0 & T^3/2 & 0 \\ 0 & T^4/4 & 0 & T^3/2 \\ T^3/2 & 0 & T^2 & 0 \\ 0 & T^3/2 & 0 & T^2 \end{bmatrix} [cov_{modelo}] \tag{3.9}$$

De igual manera, usando el método heurístico se determinó el valor de la covarianza del modelo:

$$cov\_modelo = 2.8$$

La matriz de control se la considera con valores constantes y además se define el valor de T=1:

$$u_k = \begin{bmatrix} 0.005 \\ 0.005 \end{bmatrix}$$

## 3.3 IMPLEMENTACIÓN DEL CONTROLADOR

### 3.3.1 ELECCIÓN DEL CONTROLADOR

De entre todas las posibilidades de controladores que existen, en función del grado de complejidad y de los requerimientos de la planta se ha escogido controlar el sistema con un PID, que presenta mucha robustez frente a perturbaciones y ruidos y que además brinda ventajas tales como: su diseño e implementación son relativamente sencillos además permite el reajuste de sus parámetros por el método heurístico que en otros controladores es mucho más complicado.

### 3.3.2 MODELO DINÁMICO DEL SISTEMA [29] [30]

Para el diseño del controlador es necesario conocer un modelo matemático que represente la dinámica del sistema con el fin de determinar el controlador adecuado y además de poder corroborar su funcionamiento mediante simulación, para luego llevar estos valores a la planta física.

#### 3.3.2.1 Modelo completo del sistema

Para la obtención del modelo dinámico del sistema, se toma en cuenta las siguientes consideraciones:

- El modelo mecánico, tanto del sistema que transmite el movimiento hacia la plataforma, como la plataforma en sí, se consideran ideales, sin pérdidas de movimiento y de masa despreciable.
- Se desprecia la fricción entre la esfera y la plataforma.

La Figura 3.10 muestra el movimiento de la esfera sobre la plataforma y la representación de las variables en el eje X.

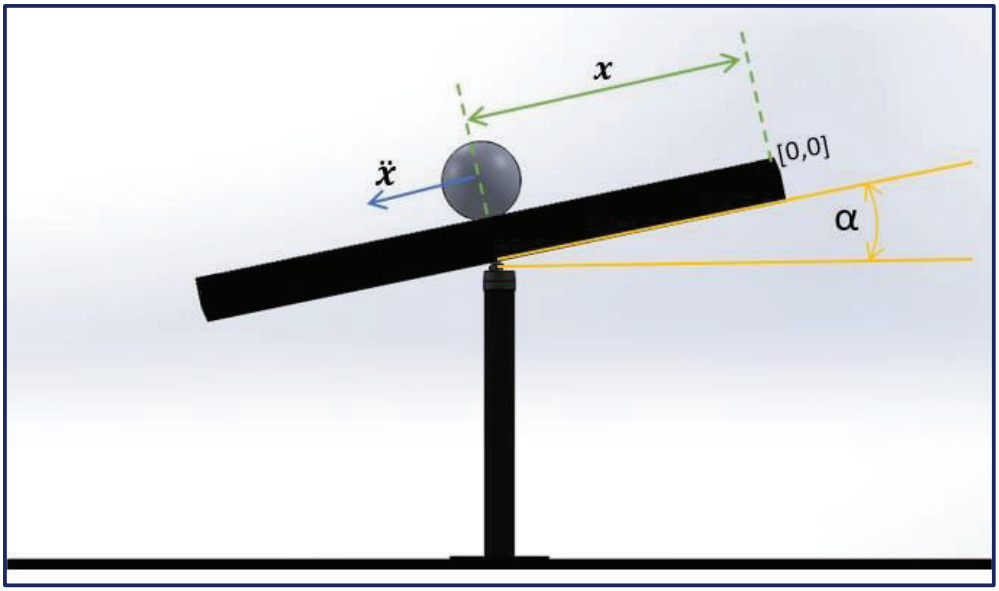



**Figura 3.10.** Representación de las variables en el eje X para la plataforma.

A continuación, se detalla las variables y su representación física tanto para el eje X como para el eje Y:

$\boldsymbol{\alpha} \rightarrow Ángulo\ de\ inclinación\ de\ la\ plataforma\ en\ el\ eje\ x$

$\dot{\boldsymbol{\alpha}} \rightarrow Velocidad\ angular\ de\ la\ plataforma\ en\ el\ eje\ x$

$\boldsymbol{\beta} \rightarrow Ángulo\ de\ inclinación\ de\ la\ plataforma\ en\ el\ eje\ y$

$\dot{\boldsymbol{\beta}} \rightarrow Velocidad\ angular\ de\ la\ plataforma\ en\ el\ eje\ y$

$\boldsymbol{x} \rightarrow Desplazamiento\ de\ la\ esfera\ en\ el\ eje\ x$

$\dot{\boldsymbol{x}} \rightarrow Velocidad\ lineal\ de\ la\ esfera\ en\ el\ eje\ x$

$\ddot{\boldsymbol{x}} \rightarrow Aceleración\ lineal\ de\ la\ esfera\ en\ el\ eje\ x$

$\boldsymbol{y} \rightarrow Desplazamiento\ de\ la\ esfera\ en\ el\ eje\ y$

$\dot{\boldsymbol{y}} \rightarrow Velocidad\ lineal\ de\ la\ esfera\ en\ el\ eje\ y$

$\ddot{\boldsymbol{y}} \rightarrow Aceleración\ lineal\ de\ la\ esfera\ en\ el\ eje\ y$

$\boldsymbol{r_e} \rightarrow Radio\ de\ la\ esfera$

$\boldsymbol{m_e} \rightarrow Masa\ de\ la\ esfera$

$\boldsymbol{I_e} \rightarrow Momento\ de\ inercia\ de\ la\ esfera$

$\boldsymbol{r_s} \rightarrow Radio\ del\ aspa\ del\ servomotor\ medido\ desde\ el\ eje\ hasta\ el\ conector$

$\boldsymbol{H} \rightarrow Distancia\ desde\ el\ conector\ hasta\ el\ pivote$

$\boldsymbol{\theta_x} \rightarrow Angulo\ del\ servomotor\ del\ eje\ x$

$\boldsymbol{\theta_y} \rightarrow Angulo\ del\ servomotor\ del\ eje\ y$

La Figura 3.11 muestra la representación de las variables en el eje X para el movimiento de los servomotores.

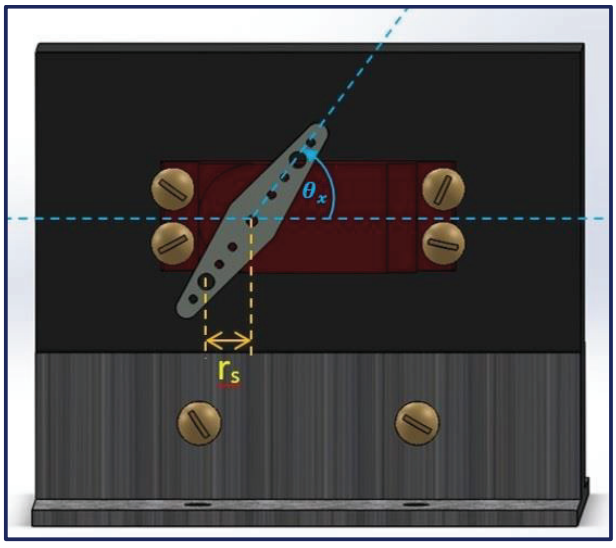



**Figura 3.11.** Representación de las variables en el eje X para los servomotores

Para obtener el modelo se usa la ecuación de Euler-Lagrange donde el Lagrangiano se obtiene de la diferencia de la energía cinética y potencial del sistema.

$$L(t) = E_{c_{sys}}(t) - E_{p_{sys}}(t) \tag{3.10}$$

*Energía Cinética del Sistema:*

La energía cinética total del sistema es igual a la suma de la energía cinética de la esfera cuando rota en la plataforma, más la energía cinética de la esfera producto de la rotación de la plataforma:

$$\boldsymbol{E_{c_{sys}}} = E_{c_{esfera}} + E_{c_{esfera-plataforma}} \tag{3.11}$$

*Energía Cinética de la Esfera:*

La esfera se desplaza a lo largo de la plataforma con una velocidad angular y una velocidad lineal, definidas como se muestra en la Figura 3.12:

*Composición de velocidades de la esfera:*

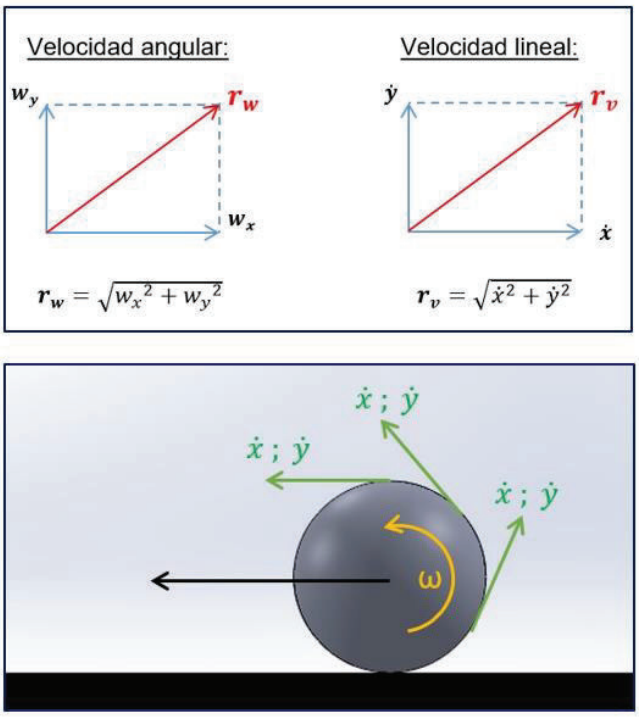



**Figura 3.12.** Composición de velocidades de la esfera

La energía cinética de la esfera es igual a la suma de la energía cinética de rotación y la energía cinetica de traslación:

$$\boldsymbol{E_{c_e}} = E_{c_{re}} + E_{c_{te}} \tag{3.12}$$

$$\boldsymbol{E_{c_e}} = \frac{1}{2} I_e (r_w)^2 + \frac{1}{2} m_e (r_v)^2 \tag{3.13}$$

$$\boldsymbol{E_{c_e}} = \frac{1}{2} I_e \left( {w_x}^2 + {w_y}^2 \right) + \frac{1}{2} m_e (\dot{x}^2 + \dot{y}^2) \tag{3.14}$$

Como la velocidad tangencial de la esfera es:

$$\begin{aligned} \dot{x} = w_x r_e \qquad & \therefore \ \boldsymbol{w_x^2} = \left(\frac{\dot{x}}{r_e}\right)^2 \\ \dot{y} = w_y r_e \qquad & \therefore \ \boldsymbol{w_y^2} = \left(\frac{\dot{y}}{r_e}\right)^2 \end{aligned} \tag{3.15}$$

$$\rightarrow \quad \boldsymbol{E_{c_e}} = \frac{1}{2} I_e \left[ \left(\frac{\dot{x}}{r_e}\right)^2 + \left(\frac{\dot{y}}{r_e}\right)^2 \right] + \frac{1}{2} m_e (\dot{x}^2 + \dot{y}^2)$$

$$\boldsymbol{E_{c_e}} = \frac{1}{2} \left( m_e + \frac{I_e}{{r_e}^2} \right) (\dot{x}^2 + \dot{y}^2) \tag{3.16}$$

*Energía Cinética de la esfera sobre la plataforma:*

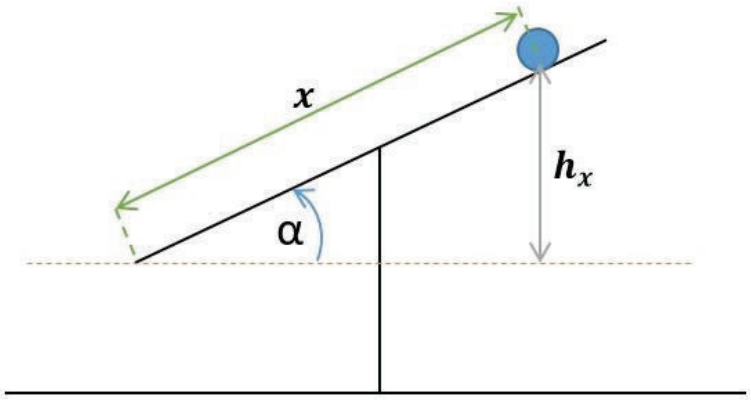
De forma similar, la esfera describe un movimiento de rotación y traslación cuando la plataforma describe un ángulo.

$$\boldsymbol{E_{c_{ep}}} = E_{cr_{ep}} + E_{ct_{ep}} \tag{3.17}$$

$$\boldsymbol{E_{c_{ep}}} = \frac{1}{2} I_{ep} (r_\phi)^2 + \frac{1}{2} m_e (r_{Vt})^2 \tag{3.18}$$

$$\boldsymbol{E_{c_{ep}}} = \frac{1}{2} I_{ep} \left( \dot{\alpha}^2 + \dot{\beta}^2 \right) + \frac{1}{2} m_e \left( x\dot{\alpha} + y\dot{\beta} \right)^2 \tag{3.19}$$

*Energía Potencial del sistema:*




$$sen\ \alpha = \frac{h_x}{x}$$

$$\rightarrow h_x = x\ sen\ \alpha$$

$$\rightarrow h_y = y\ sen\ \beta$$

$$\boldsymbol{E_{p_{sys}}} = m_e g h \tag{3.20}$$

$$\boldsymbol{E_{p_{sys}}} = m_e g[x\ sen(\alpha) + y\ sen(\beta)] \tag{3.21}$$

Por lo tanto, el Langrangiano del Sistema según la ecuación (3.10), es resultado de las siguientes operaciones entre las ecuaciones $(3.16) + (3.19) - (3.21)$:

$$\boldsymbol{\mathcal{L}} = \frac{1}{2}\left(m_e + \frac{I_e}{{r_e}^2}\right)(\dot{x}^2 + \dot{y}^2) + \frac{1}{2} I_{ep}\left(\dot{\alpha}^2 + \dot{\beta}^2\right)$$

$$+\frac{1}{2} m_e\left(x\dot{\alpha} + y\dot{\beta}\right)^2 - m_e g\left(x\ sen(\alpha) + y\ sen(\beta)\right) \tag{3.22}$$

*Análisis en el eje X:*

Ecuación de Lagrange del movimiento:

$$\boldsymbol{\frac{d}{dt}\left[\frac{\partial \mathcal{L}}{\partial \dot{x}}\right] - \frac{\partial \mathcal{L}}{\partial x} = 0} \tag{3.23}$$

Utilizando la ecuación (3.22) y derivando se obtiene:

$$\frac{\partial \mathcal{L}}{\partial \dot{x}} = \left(m_e + \frac{I_e}{{r_e}^2}\right)\dot{x} \tag{3.24}$$

$$\frac{\partial \mathcal{L}}{\partial x} = m_e x\dot{\alpha}^2 + m_e y\dot{\alpha}\dot{\beta} - m_e g\ sen(\alpha) \tag{3.25}$$

$$\frac{d}{dt}\left[\frac{\partial \mathcal{L}}{\partial \dot{x}}\right] = \left(m_e + \frac{I_e}{{r_e}^2}\right)\ddot{x} \tag{3.26}$$

Reemplazando las ecuaciones (3.24) (3.25) y (3.26) en (3.23):

$$\therefore \left(m_e + \frac{I_e}{{r_e}^2}\right)\ddot{x} - m_e x\dot{\alpha}^2 - m_e y\dot{\alpha}\dot{\beta} + m_e g\ sen(\alpha) = 0$$

Como el momento de inercia de una esfera sólida respecto a su eje es:

$$\boldsymbol{I_e} = \frac{2}{5} m_e {r_e}^2 \tag{3.27}$$

$$\rightarrow \boxed{\frac{\boldsymbol{7}}{\boldsymbol{5}}\boldsymbol{\ddot{x} - x\dot{\alpha}^2 - y\dot{\alpha}\dot{\beta} + g\ sen(\alpha) = 0}} \tag{3.28}$$

*Análisis en el eje Y:*

Ecuación de Lagrange del movimiento:

$$\boldsymbol{\frac{d}{dt}\left[\frac{\partial \mathcal{L}}{\partial \dot{y}}\right] - \frac{\partial \mathcal{L}}{\partial y} = 0} \tag{3.29}$$

Utilizando la ecuación (3.22) y derivando se obtiene:

$$\frac{\partial \mathcal{L}}{\partial \dot{y}} = \left(m_e + \frac{I_e}{{r_e}^2}\right)\dot{y} \tag{3.30}$$

$$\frac{\partial \mathcal{L}}{\partial y} = m_e y\dot{\beta}^2 + m_e x\dot{\alpha}\dot{\beta} - m_e g\ sen(\beta) \tag{3.31}$$

$$\frac{d}{dt}\left[\frac{\partial \mathcal{L}}{\partial \dot{y}}\right] = \left(m_e + \frac{I_e}{{r_e}^2}\right)\ddot{y} \tag{3.32}$$

Reemplazando las ecuaciones (3.30) (3.31) y (3.32) en (3.29):

$$\therefore \left(m_e + \frac{I_e}{{r_e}^2}\right)\ddot{y} - m_e y\dot{\beta}^2 - m_e x\dot{\alpha}\dot{\beta} + m_e g\ sen(\beta) = 0$$

Como el momento de inercia de una esfera sólida respecto a su eje es la ecuación (3.27):

$$\rightarrow \boxed{\frac{\boldsymbol{7}}{\boldsymbol{5}}\boldsymbol{\ddot{y} - y\dot{\beta}^2 - x\dot{\alpha}\dot{\beta} + g\ sen(\beta) = 0}} \tag{3.33}$$

*Modelo del Sistema en variables de estado:*

Las ecuaciones (3.28) y (3.33) describen el movimiento de la esfera ante un cambio en el ángulo de inclinación de la plataforma. Despejando la aceleración de (3.28) y (3.33) se tiene:

$$\rightarrow \boxed{\ddot{x} = \frac{5}{7}\left(x\dot{\alpha}^2 + y\dot{\alpha}\dot{\beta} - g\ sen(\alpha)\right)} \tag{3.34}$$

$$\rightarrow \boxed{\ddot{y} = \frac{5}{7}\left(y\dot{\beta}^2 + x\dot{\alpha}\dot{\beta} - g\ sen(\beta)\right)} \tag{3.35}$$

Utilizando estas ecuaciones se plantea el sistema en variables de estado, quedando:

$$x_1(t) = x(t) \qquad x_2(t) = \dot{x}_1(t) = \dot{x}(t)$$

$$\dot{x}_2(t) = \ddot{x}(t) \qquad x_3(t) = \alpha(t)$$

$$x_4(t) = \dot{x}_3(t) = \dot{\alpha}(t) \qquad \dot{x}_4(t) = \ddot{\alpha}(t)$$

$$x_5(t) = y(t) \qquad x_6(t) = \dot{x}_5(t) = \dot{y}(t)$$

$$\dot{x}_6(t) = \ddot{y}(t) \qquad x_7(t) = \beta(t)$$

$$x_8(t) = \dot{x}_7(t) = \dot{\beta}(t) \qquad \dot{x}_8(t) = \ddot{\beta}(t)$$

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} x_2 \\ \frac{5}{7}\left(x\dot{\alpha}^2 + y\dot{\alpha}\dot{\beta} - g\ sen(\alpha)\right) \\ x_4 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} [\alpha] \tag{3.36}$$

$$\begin{bmatrix} \dot{x}_5 \\ \dot{x}_6 \\ \dot{x}_7 \\ \dot{x}_8 \end{bmatrix} = \begin{bmatrix} x_6 \\ \frac{5}{7}\left(y\dot{\beta}^2 + x\dot{\alpha}\dot{\beta} - g\ sen(\beta)\right) \\ x_8 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} [\beta] \tag{3.37}$$

### 3.3.2.2 Modelo reducido del sistema

Como el sistema es no lineal, para trabajar posteriormente en el diseño del controlador, se puede reducir el sistema a un sistema lineal tomando en cuenta lo siguiente:

Como el ángulo de inclinación de la plataforma no excede los 8 grados, y considerando que los movimientos de la plataforma son lentos, se puede decir que:

$$\dot{\alpha} \cong 0 \quad ^{\wedge} \quad \dot{\beta} \cong 0$$

$$\therefore \ \dot{\alpha}^2 \cong 0 \ ^{\wedge} \ \dot{\beta}^2 \cong 0 \ ^{\wedge} \ \dot{\alpha}\dot{\beta} \cong 0$$

Reemplazando las anteriores aproximaciones en (3.34) y (3.35) resultan las siguientes ecuaciones:

$$\ddot{x} = -\frac{5}{7} g sen(\alpha) \tag{3.38}$$

$$\ddot{y} = -\frac{5}{7} g sen(\beta) \tag{3.39}$$

A continuación, se linealiza las ecuaciones (3.38) y (3.39) en el punto $\alpha = 0 \ ^{\wedge} \beta = 0$ que es cuando la plataforma está en posición horizontal y mantiene a la esfera en equilibrio. Se linealiza utilizando la serie de Taylor:

$$f(x) = \sum_{n=0}^{\infty} \frac{f^n(a)}{n!} (x-a)^n \tag{3.40}$$

$$donde: a \ es \ el \ punto \ de \ linealización \ y \ n \ es \ el \ orden \ de \ la \ n - ésima \ derivada$$

Para la primera derivada (n=1):

$$f(x) = f(a) + f'(a)\,(x-a) \tag{3.41}$$

$$\therefore \ \ddot{x} = \left[-\frac{5}{7} g sen(0)\right] + \left[-\frac{5}{7} g cos(0)(\alpha - 0)\right]$$

$$\rightarrow \ \ddot{x} = -\frac{5}{7} g\alpha \quad y \ si \ la \ gravedad \ \ g = -980 \left[\frac{cm}{s^2}\right]$$

$$\rightarrow \ \ddot{x} = 700\,\alpha \tag{3.42}$$

Esta ecuación relaciona la aceleración de la esfera ante un cambio en el ángulo de la plataforma, sin embargo, la acción de control se va a realizar en los servomotores, por tanto, se debe relacionar al ángulo de la plataforma con el ángulo del servomotor.

La relación entre el ángulo del servomotor y el ángulo de la plataforma es del tipo lineal cuando los ángulos rondan el valor de 90 ± 60 grados (en el servomotor). Para la obtención de la constante que relaciona estos dos ángulos se realizaron pruebas utilizando un giroscopio y se obtuvo que la relación entre los ángulos es de aproximadamente 1/10.

$$\alpha = \frac{1}{10}\theta_x \qquad \rightarrow \ddot{x} = 700\left(\frac{1}{10}\theta_x\right)$$

$$\rightarrow \boldsymbol{\ddot{x} = 70\,\theta_x} \tag{3.43}$$

De forma análoga para el eje Y se tiene:

$$\boldsymbol{\ddot{y} = 70\,\theta_y} \tag{3.44}$$

Aplicando la transformada de Laplace para obtener la función de trasferencia en el dominio de la variable compleja $s$:

$$\mathcal{L}\{f''\} = s^2\mathcal{L}\{f\} - sf(0) - f'(0) \tag{3.45}$$

Considerando además condiciones iniciales nulas, se tiene:

$$\boxed{\frac{\boldsymbol{X(s)}}{\boldsymbol{\theta_x(s)}} = \frac{\boldsymbol{70}}{\boldsymbol{s^2}}} \tag{3.46}$$

$$\boxed{\frac{\boldsymbol{Y(s)}}{\boldsymbol{\theta_y(s)}} = \frac{\boldsymbol{70}}{\boldsymbol{s^2}}} \tag{3.47}$$

Donde $X(s)\ y\ Y(s\ )$ son las posiciones en cada eje de la esfera en la plataforma, en centímetros y $\theta_x(s)\ y\ \theta_y(s)$ son los ángulos de los servomotores en grados.

A partir de estas funciones de transferencia (3.46) y (3.47) se obtiene el modelo en variables de estado que servirá para el análisis de la Controlabilidad del sistema.

La representación en variables de estado de un doble integrador viene dada de la siguiente forma:

$$\dot{x}\,(t) = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ k \end{bmatrix} u(t) \tag{3.48}$$

$$y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} x(t) \tag{3.49}$$

Donde $k$ es la ganancia del doble integrador ($k/s^2$), $x(t)$ es el vector de estados y $\dot{x}\,(t)$ es la evolución de los estados. Por lo tanto, el sistema Bola-plataforma queda representado así:

$$\begin{bmatrix} \dot{\boldsymbol{x}} \\ \ddot{\boldsymbol{x}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{1} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \dot{\boldsymbol{x}} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{70} \end{bmatrix} \boldsymbol{\alpha} \tag{3.50}$$

$$\begin{bmatrix} \dot{\boldsymbol{y}} \\ \ddot{\boldsymbol{y}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{1} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{y} \\ \dot{\boldsymbol{y}} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{70} \end{bmatrix} \boldsymbol{\beta} \tag{3.51}$$

De (3.50) y (3.51) se obtienen las matrices que servirán para los posteriores cálculos. Las matrices que definen el sistema en cada eje son idénticas:

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ 70 \end{bmatrix}, \qquad C = \begin{bmatrix} 1 & 0 \end{bmatrix}, \qquad D = [0]$$

### 3.3.3 ANÁLISIS DE CONTROLABILIDAD Y OBSERVABILIDAD [31]

#### 3.3.3.1 Controlabilidad

La controlabilidad es una característica de los sistemas que permite determinar si la evolución de sus estados puede ser modificada por las entradas al sistema.

En otras palabras, un sistema es controlable si existe una señal de entrada que permita llevar al sistema del estado $x_1$ al estado $x_2$ en tiempo finito.

La Controlabilidad se puede demostrar mediante el criterio de Controlabilidad de Kalman que se enuncia así: Un sistema LTI representado por la ecuación de estados:

$$\dot{x}\,(t) = A\,x(t) + Bu(t)$$

donde $\mathrm{A} \in \mathrm{R}^{\mathrm{nxn}}, \mathrm{B} \in \mathrm{R}^{\mathrm{nxp}}$ es controlable si y solo si, la matriz de Controlabilidad C tiene un rango $\mathrm{n}$. La matriz C está dada por:

$$\boldsymbol{C} = [B \mid AB \mid \ldots\ldots \mid A^{n-1}B] \tag{3.52}$$

Y para este caso de estudio, mediante (3.52):

$$\boldsymbol{C} = \begin{bmatrix} 0 & 70 \\ 70 & 0 \end{bmatrix} \quad \rightarrow \quad rango(C) = 2$$

La matriz A es de dimensiones $2x2$, por lo tanto $n = 2$, y el rango de la matriz C es 2 por lo tanto el sistema es Controlable.

### 3.3.3.2 Observabilidad

La observabilidad es una característica de los sistemas que mide la capacidad de poder estimar los valores históricos de un estado teniendo como información los datos de la entrada y la salida.

En otras palabras, un estado inicial $x(0)$ de un sistema es observable si existe un tiempo finito tal que en el intervalo de $[0, \mathrm{t}]$ conociendo la entrada y la salida, se puede determinar el estado $x(0)$.

La Observabilidad se puede demostrar usando el criterio de Observabilidad de Kalman: Un sistema LTI representado por las ecuaciones:

$$\dot{x}\,(t) = A\,x(t) + Bu(t)$$

$$y(t) = B\,x(t) + Du(t)$$

Donde $\mathrm{A} \in \mathrm{R}^{\mathrm{nxn}}, \mathrm{B} \in \mathrm{R}^{\mathrm{nxp}}, \mathrm{C} \in \mathrm{R}^{\mathrm{qxn}}, \mathrm{D} \in \mathrm{R}^{\mathrm{qxp}}$ es observable si y solo si la matriz de Observabilidad O tiene un rango $\mathrm{n}$. La matriz O está dada por:

$$\boldsymbol{O} = \begin{bmatrix} C \\ CA \\ . \\ . \\ CA^{n-1} \end{bmatrix} \tag{3.53}$$

Y para este caso de estudio, mediante (3.53):

$$\boldsymbol{O} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \rightarrow \quad rango(O) = 2$$

Como el rango de A es $n = 2$, y el rango O es 2 por lo tanto el sistema es observable.

### 3.3.4 DESARROLLO DEL CONTROLADOR

Una vez que se ha determinado que el sistema es controlable, el objetivo es determinar un controlador que sea capaz de estabilizar al sistema y logre los resultados deseados en lo que respecta al seguimiento de caminos o control de posición de la esfera en la plataforma.

Para determinar el controlador se han tomado las siguientes consideraciones:

- El controlador ha sido determinado mediante un algoritmo iterativo basado en el método heurístico, es decir, es un algoritmo computacional sugerido por [7], que toma valores de un rango previamente definido para las constantes del controlador y prueba que valores hacen que el sistema genere el comportamiento deseado tanto para el régimen transitorio como para el régimen permanente en la respuesta del sistema.

- Las especificaciones de diseño que se plantearon son las siguientes:

$$Mp \leq 1\%$$
$$ts \leq 0.04[s]$$
$$Ess = 0$$

Donde:

$Mp$ : $Máximo\ sobre\ pico$

$ts$ : $Tiempo\ de\ establecimiento$

$Ess$ : $Error\ de\ posición\ en\ estado\ estable$

Después de varias pruebas y ajustes el algoritmo arrojo los siguientes valores para las constates buscadas:

$$Kp = 1 \qquad Ki = 1 \qquad Kd = 1.5$$

Dando como resultado la siguiente función de transferencia del controlador PID:

$$\boldsymbol{C(s) = \frac{1.5s^2 + s + 1}{s}} \tag{3.54}$$

Mediante simulación, se obtiene el siguiente comportamiento en régimen transitorio y estacionario del controlador dado por la ecuación 3.54. En la Figura 3.13 se muestra la respuesta de la planta ante una entrada escalón sin compensador y en la Figura 3.14 la respuesta de la planta incluido el controlador que se determinó previamente del cual se obtuvieron los siguientes resultados.

$$Mp = 0.5429\% \quad ts = 0.0349[s] \quad Ess = 0$$

La respuesta del sistema sin controlador es una señal que crece indefinidamente, como ya se había dicho antes, en presencia de cualquier perturbación que varíe el ángulo de la plataforma la posición de la esfera incrementaría sin detenerse, en cambio, con el controlador se puede observar que la posición de la esfera llega al valor deseado manteniéndose en él, y manteniendo un error en estado estacionario de cero, como se esperaba.

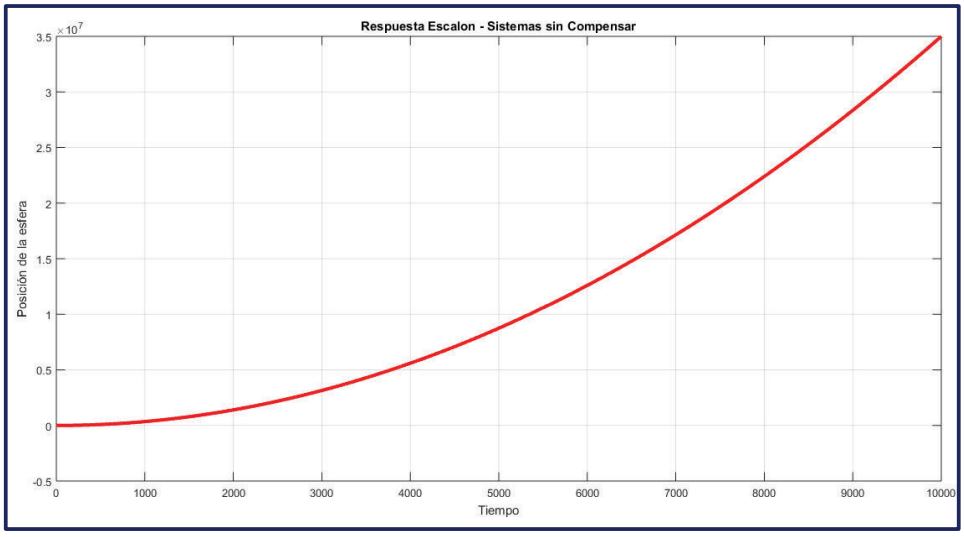



**Figura 3.13.** Respuesta del Sistema sin Compensador

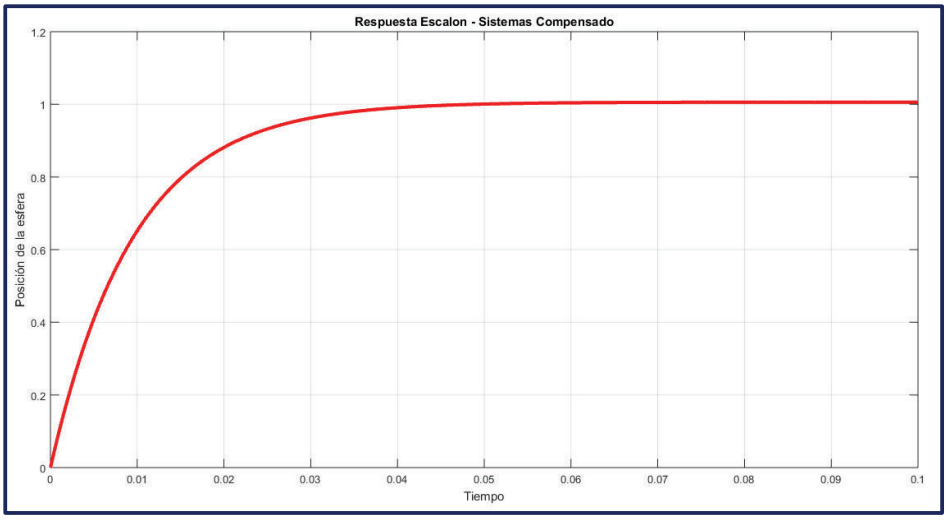



**Figura 3.14.** Respuesta del Sistema con Compensador.

### 3.3.5 ANÁLISIS DE ESTABILIDAD

El sistema bola-plataforma tiene una naturaleza inherentemente inestable, pero con el controlador adecuado se puede llevar al sistema a la estabilidad:

#### 3.3.5.1 Estabilidad de Lyapunov [32]

La estabilidad según el primer método de Aleksandr Lyapunov, llamado también Método indirecto de estabilidad de Lyapunov, dice que un sistema es estable si los polos de su ecuación característica se ubican en el semiplano izquierdo del plano complejo $s$.

Determinando la función de transferencia de todo el conjunto planta-controlador en lazo cerrado y calculando sus raíces se tiene:

$$GT\ (s) = \frac{105s^2 + 70s + 70}{s^3 + 105s^2 + 70s + 70}$$

$$s_1 = -104.34$$

$$s_{2,3} = -0.33 \pm 0.75i$$

En la Figura 3.15 se muestra el lugar geométrico de las raíces del sistema compensado en donde se ve la posición de los polos del sistema en lazo cerrado, corroborando así su estabilidad.

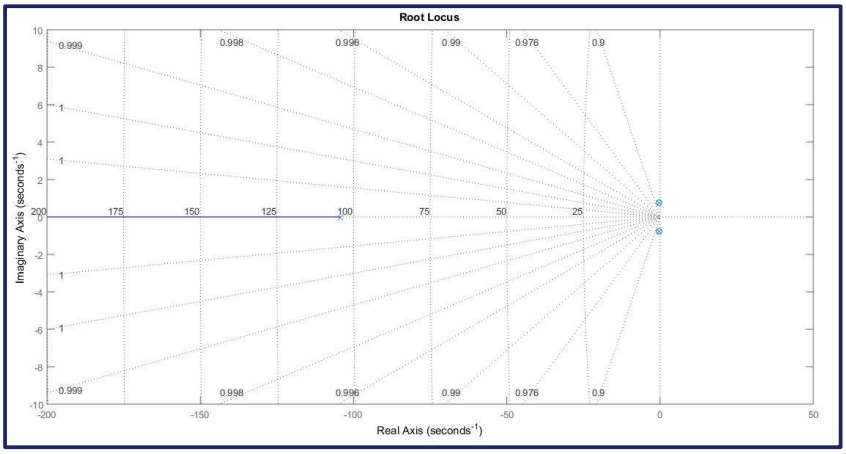



**Figura 3.15.** Lugar Geométrico de las Raíces, sistema compensado.

### 3.3.6 PRUEBAS DE SIMULACIÓN

#### 3.3.6.1 Implementación en Simulink

Para probar el controlador diseñado de una manera aún más precisa, y con una amplia variedad de caminos se ha implementado una simulación con la herramienta Simulink de Matlab, en la cual se ha usado el modelo completo de la planta, representado por (3.34) y (3.35). La Figura 3.16 muestra el diagrama de simulación del sistema bola-plataforma.

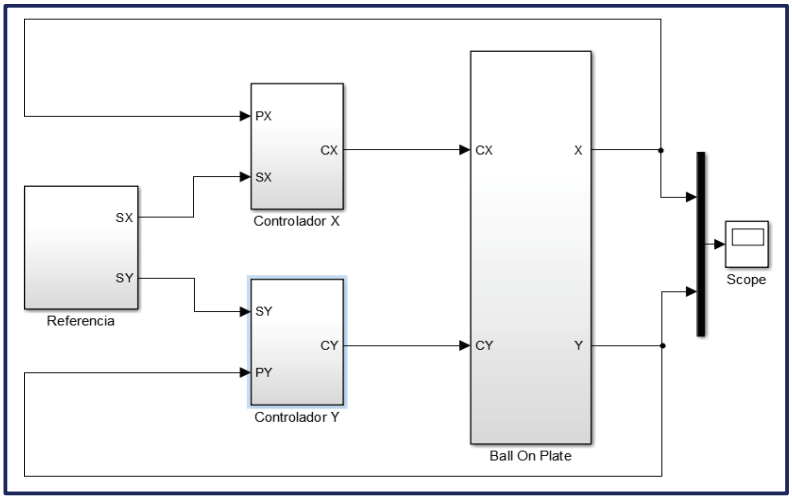



**Figura 3.16.** Simulación Sistema Bola-Plataforma.

La simulación consta de tres Bloques principales: Un bloque que simula al sistema completo incluido las no linealidades, un bloque que simula el controlador, y un bloque que simula la generación de caminos o referencias.

El bloque del modelo dinámico del sistema, mostrado en la Figura 3.17, incluye dos funciones que representan la dinámica de la planta en cada eje.

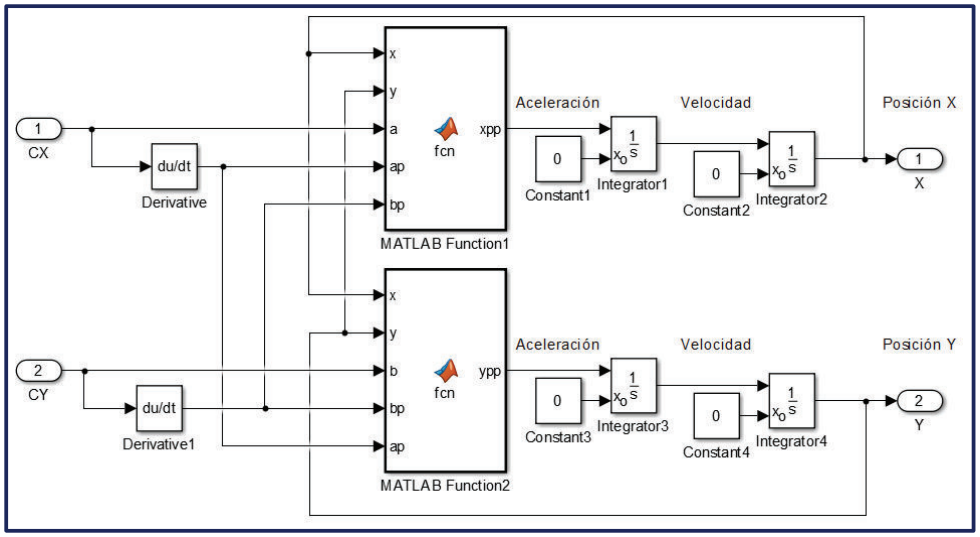



**Figura 3.17.** Simulación Bola-Plataforma: Dinámica del Sistema

En la Figura 3.18 se muestra diagrama del controlador en el eje X. De forma análoga, se realiza el controlador para el eje Y.

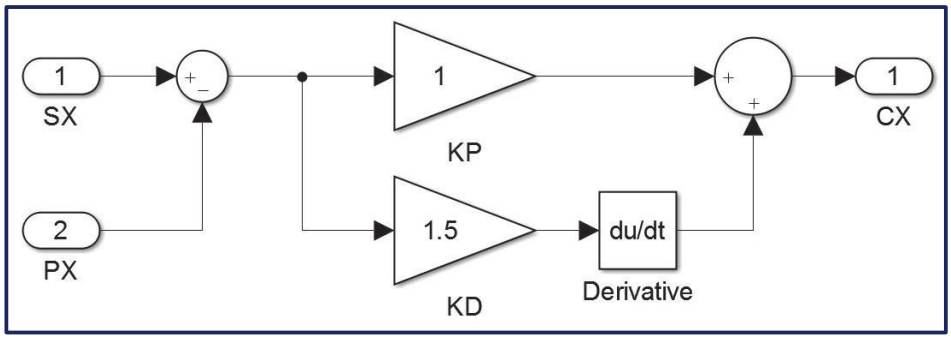



**Figura 3.18.** Simulación Bola-Plataforma: Controlador PD

### 3.3.6.2 Resultados obtenidos por Simulación

A continuación, se muestran los resultados obtenidos mediante simulación. Sin embargo, es importante mencionar que para la simulación no se consideró la acción integral ya que provocaba inestabilidad de los resultados por el hecho de la que planta ya posee un doble integrador.

Con esta simulación se probaron cuatro caminos: Punto, Círculo, Número Ocho, Elipse que arrojaron los siguientes resultados:

*Puntos:* En la Figura 3.19, en la primera grilla se indica el punto deseado $P_2$ con color rojo y el punto de partida $P_1$ con color verde, que en este caso es el centro de la plataforma, en la segunda grilla se muestra con una línea verde el camino seguido por la esfera para ir desde el punto $P_1$ hasta el punto $P_2$

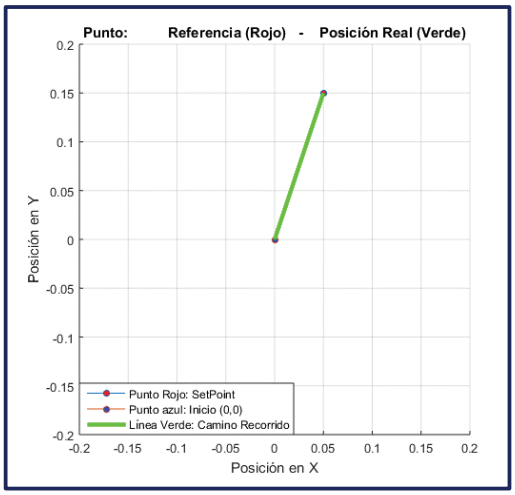



**Figura 3.19.** Referencia: Puntos

En la Figura 3.20 se muestran: Referencia, Posición Real, y señal de Control versus tiempo, para cada eje.

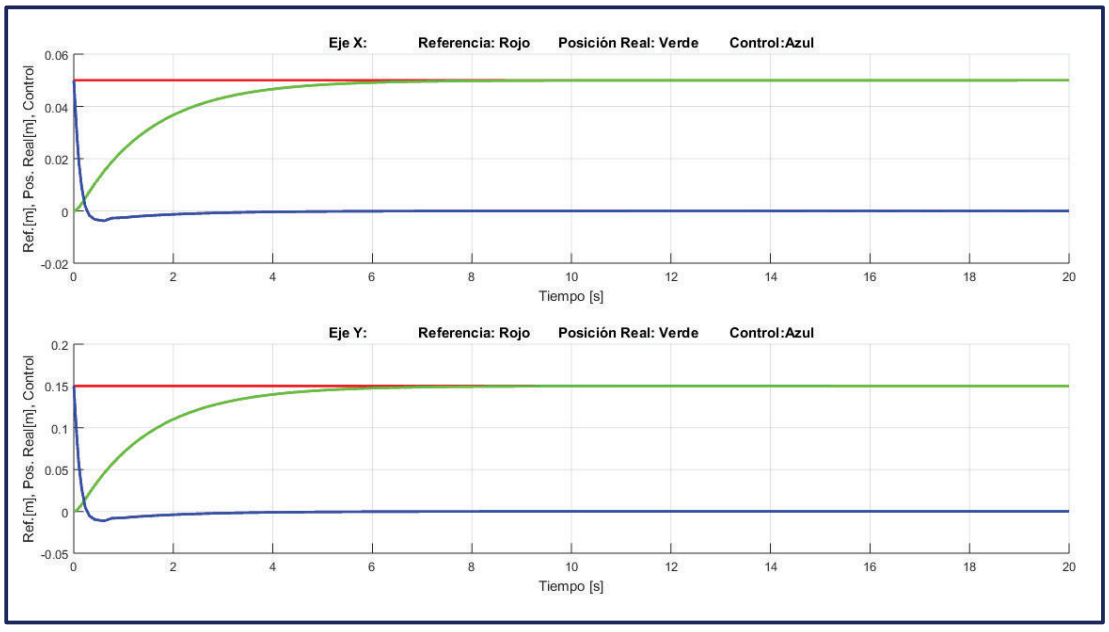



**Figura 3.20.** Referencia, Posición Real, Control vs Tiempo: Puntos

*Círculo:* En la Figura 3.21, en color rojo se encuentra el circulo que se desea realizar y en verde el circulo realizado por el sistema con el controlador diseñado. El circulo nace desde el centro de la plataforma y empieza su recorrido un poco fuera de la trayectoria, pero con el pasar del tiempo alcanza el valor deseado.

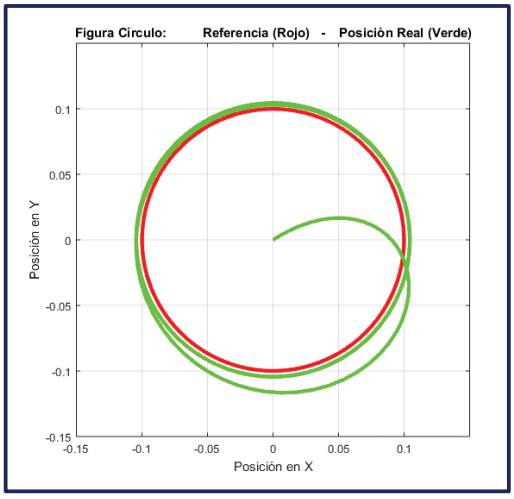



**Figura 3.21.** Referencia: Círculo

En la Figura 3.22 se muestran: Referencia, Posición Real, y señal de Control versus tiempo, para cada eje.

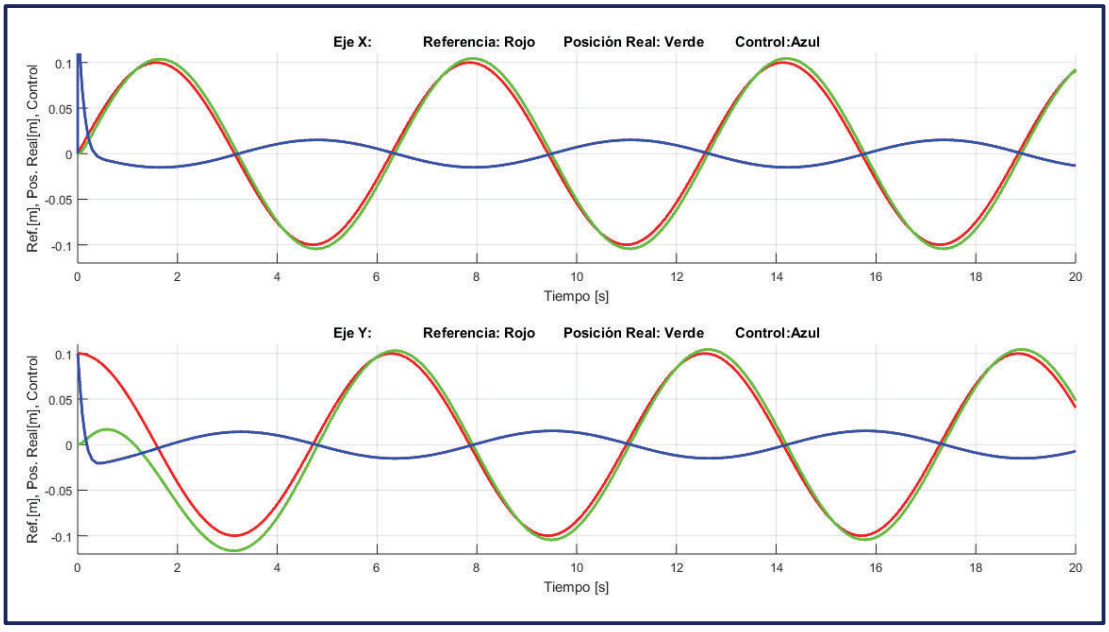



**Figura 3.22.** Referencia, Posición Real, Control vs Tiempo: Círculo

*Figura ocho:* En la Figura 3.23, en el primer recuadro se tiene la figura deseada que en este caso es el número ocho, y en segundo recuadro se encuentra la figura que el sistema realiza, y como se observa la figura nace desde el centro para luego, después de un tiempo llegar al valor deseado y mantenerse en él.

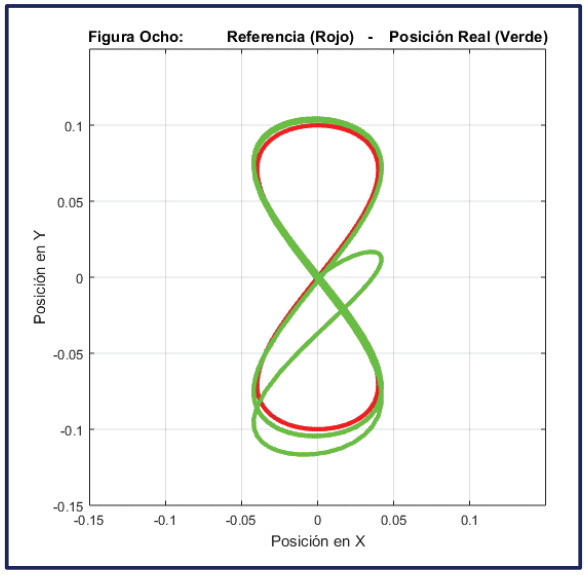



**Figura 3.23.** Referencia: Figura ocho

En la Figura 3.24 se muestran: Referencia, Posición Real, y señal de Control versus tiempo, para cada eje.

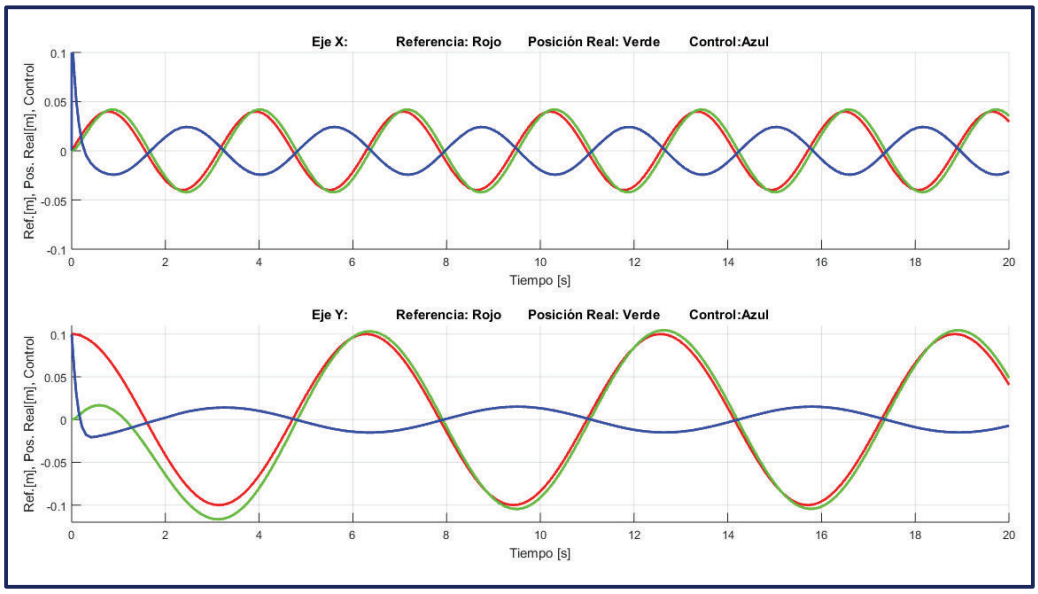



**Figura 3.24.** Referencia, Posición Real, Control vs Tiempo: Ocho

*Elipse:* En la Figura 3.25, de forma análoga a los ejemplos anteriores, la figura en rojo es el resultado deseado y en verde el resultado obtenido.

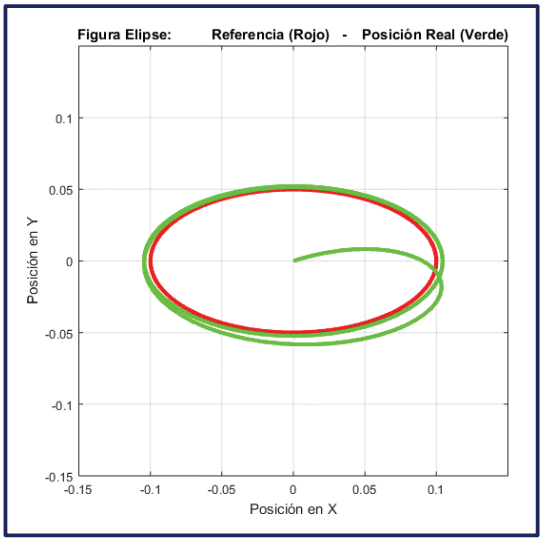



**Figura 3.25.** Referencia: Elipse

En la Figura 3.26 se muestran: Referencia, Posición Real, y señal de Control versus tiempo, para cada eje.

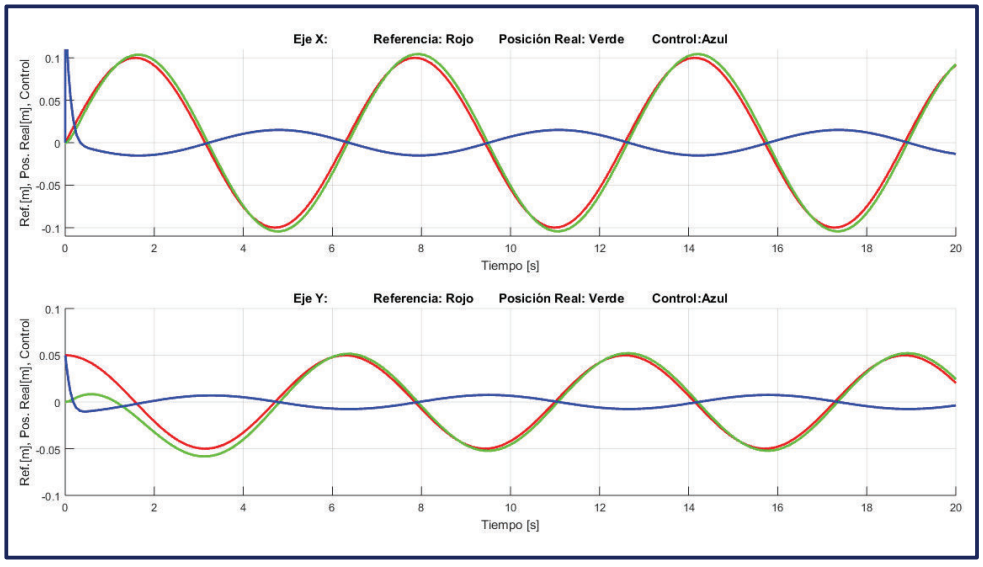



**Figura 3.26.** Referencia, Posición Real, Control vs Tiempo: Elipse

Además, se implementaron otras figuras como triangulo y cuadrados de diferentes radios del círculo circunscrito usado para obtener los vértices. Los resultados de estas figuras implementadas en el sistema real, se muestran en el capítulo 4.

### 3.3.7 REAJUSTE DE LOS PARÁMETROS DEL CONTROLADOR

Con los valores obtenidos del controlador (3.54) se tiene un punto de partida para poder realizar un ajuste más fino y de manera manual a las constantes del controlador, de tal manera que mediante ensayos se han llegado a obtener los valores adecuados para el control del cada camino realizado por el sistema.

Estos ajustes se los realiza porque para cada camino difiere la cantidad de puntos y por ende la distancia entre ellos, es decir, para un mismo tiempo, la distancia entre puntos puede aumentar o disminuir, lo que se refleja en la velocidad con la que la esfera sigue el camino y en consecuencia una ligera variación de las constantes del controlador para obtener óptimos resultados.

Por otro lado, como se indicó anteriormente, matemáticamente, el sistema por la presencia del doble integrador debería ser capaz de mantener el error en estado estacionario en cero, pero físicamente por la presencia de otras condiciones no consideradas como la fricción entre la esfera y la plataforma, se ha incluido la acción integral de tal manera que pueda corregir las incertidumbres propias de la planta inherentes a su construcción física.

En la Tabla 3.1 se indican los valores de las constantes proporcional, integral y derivativa que se obtuvieron mediante el método heurístico para cada camino:

**Tabla 3.1.** Constantes del PID para cada camino.

| Camino | KP | | KI | | KD | |
|---|---|---|---|---|---|---|
| Valores Diseñados | 1 | | 1 | | 1.5 | |
| | $x$ | $y$ | $x$ | $y$ | $x$ | $y$ |
| Puntos | 1.8 | 1.95 | 0.4 | 0.4 | 1.5 | 1.5 |
| Triangulo 7 [cm] | 1.34 | 1.455 | 0.2 | 0.2 | 1.49 | 1.575 |
| Triangulo 11 [cm] | 1.12 | 1.245 | 0.09 | 0.09 | 1.55 | 1.49 |
| Triangulo 15 [cm] | 1.02 | 1.15 | 0.1 | 0.1 | 1.44 | 1.54 |
| Cuadrado 7 [cm] | 1.4 | 1.5 | 0.11 | 0.11 | 1.54 | 1.60 |
| Cuadrado 11 [cm] | 1.17 | 1.26 | 0.11 | 0.11 | 1.45 | 1.69 |
| Cuadrado 15 [cm] | 1.13 | 1.07 | 0.11 | 0.11 | 1.39 | 1.48 |
| Circulo de radio 7 [cm] | 1.42 | 1.62 | 0.3 | 0.3 | 0.85 | 0.9 |
| Circulo de radio 11 [cm] | 1.39 | 1.52 | 0.3 | 0.3 | 0.83 | 0.88 |
| Circulo de radio 15 [cm] | 1.22 | 1.33 | 0.3 | 0.3 | 0.84 | 0.86 |
| Número Ocho | 1 | 0.705 | 0.12 | 0.13 | 1.36 | 1.48 |
| Trébol de 3 foliolos | 0.74 | 0.72 | 0.11 | 0.11 | 1.68 | 1.80 |
| Trébol de 5 foliolos | 0.39 | 0.62 | 0.22 | 0.24 | 1.83 | 1.86 |
| Elipse | 1.182 | 1.16 | 0.22 | 0.19 | 0.8 | 0.755 |

## 3.4 INTERFACES DE USUARIO

### 3.4.1 DISEÑO DE LA INTERFAZ PRINCIPAL (PC)

La interfaz de control que corre en el computador está elaborada con Guide. Guide es una herramienta de la suite de programación Matlab que permite crear "Interfaces de Usuario" mediante un lenguaje gráfico y programático, para el manejo interactivo de aplicativos de diferentes índoles y ramas de la ciencia.

Una vez creado un proyecto en Guide, se generan dos archivos, (.m y .fig) los cuales llevan el nombre que se le haya dado a la aplicación, que en nuestro caso es Interfaz_de_control_ball_on_plate.

En el archivo .m se generan automáticamente tres funciones muy importantes que se usan para correr la aplicación.

*Función Principal:* Esta función entre otras cosas, crea una estructura para asignar las funciones OpeningFcn y OutputFcn a su respectivo callback dentro de la interfaz gráfica principal.

*Función OpeningFcn:* En esta función es donde se realiza las configuraciones iniciales referentes a la primera vez que se muestra la interfaz gráfica al usuario. En ella se pone en modo visible los elementos que se desean en primera instancia.

*Función OutputFcn:* En esta función se crea e inicializa todas las variables locales usadas, así como las variables globales que se usaran a nivel de toda la aplicación. Esta función, además, aloja el Lazo Principal donde se ejecuta todo el control de la esfera sobre la plataforma que tiene la estructura mostrada en la Figura 3.27.

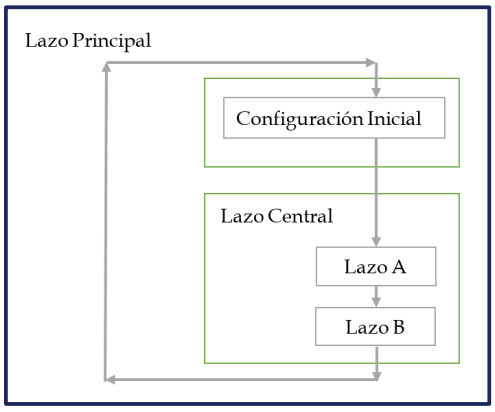



**Figura 3.27.** Lógica del Lazo Principal

### 3.4.1.1 Lazo principal

Después de haberse ejecutado la función OutputFcn, y habiéndose creado e inicializado las variables, el programa entra en un bucle “infinito” que tiene el siguiente comportamiento:

El bucle es continuo y el flujo del programa solo sale momentáneamente de él para atender a los callback generados por eventos en la interfaz de usuario como presionar un botón, o cerrar la aplicación definitivamente.

En la primera iteración configura la cámara creando un objeto de video, que posteriormente permitirá tomar imágenes y determinar la posición de la esfera.

La Figura 3.28 describe el diagrama de flujo del bucle principal del programa.

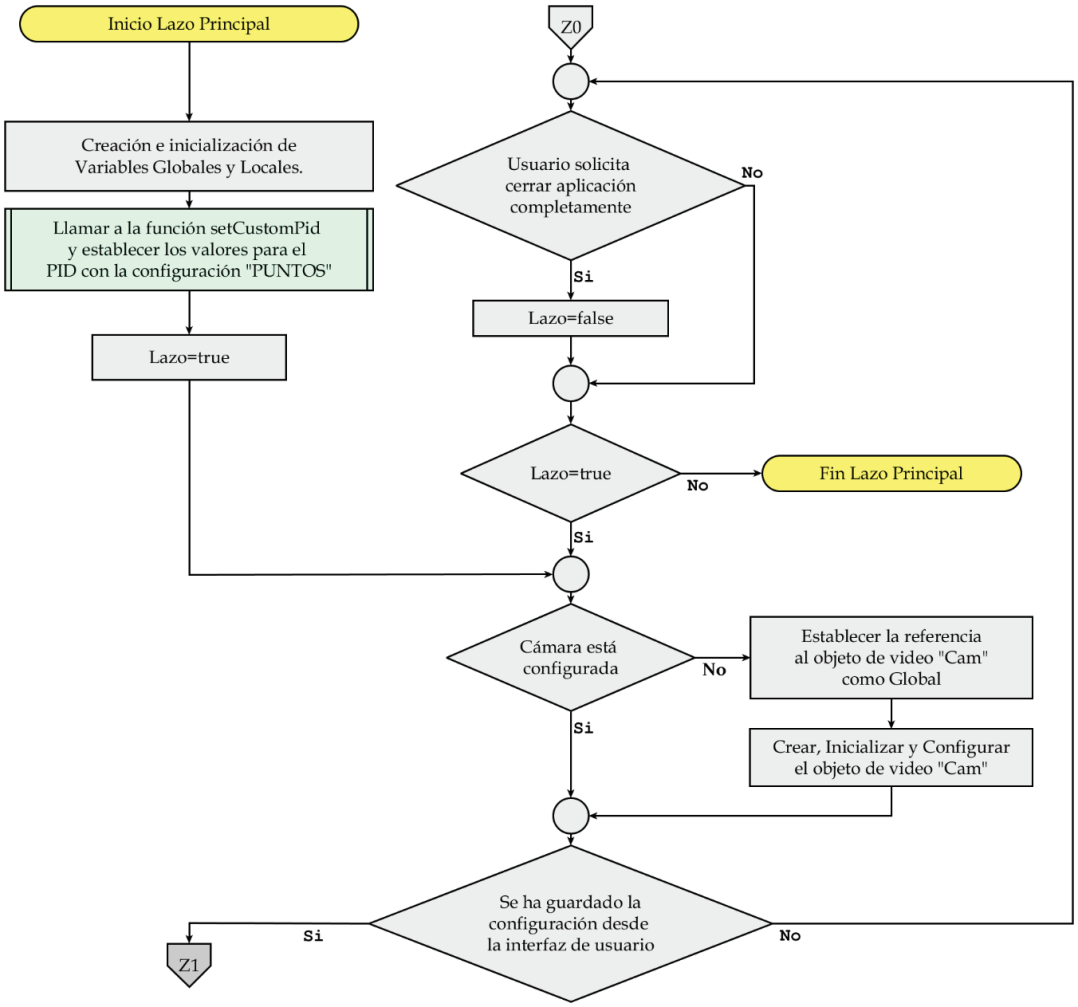



**Figura 3.28.** Lazo Principal: Bucle “infinito”

Desde la segunda iteración en adelante no realiza ninguna tarea, y espera por el ingreso de datos por parte del usuario: los números de puertos serie para la conexión con los módulos Arduino y bluetooth, y sus velocidades de transmisión.
Este evento se produce cuando el usuario presiona el botón “Guardar Configuración”. En la Figura 3.29 se detalla las acciones realizadas después de presionado el botón.

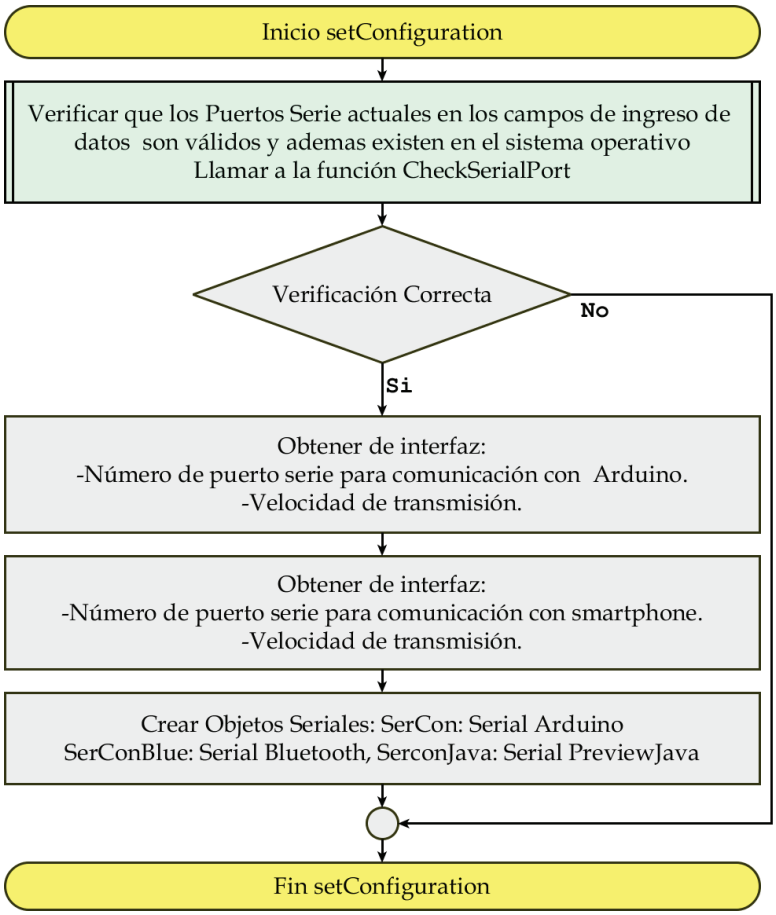



**Figura 3.29.** Función *setConfiguration*

La función *setConfiguration* en primera instancia invoca a la función *CheckSerialPort*, cuyo diagrama de flujo se muestra en la Figura 3.30. Esta función determina si los puertos especificados por el usuario realmente existen en el sistema operativo, luego si esta función no retorna ningún problema se crean los objetos seriales y se cambia el valor de la variable que permite entrar en el bucle central.

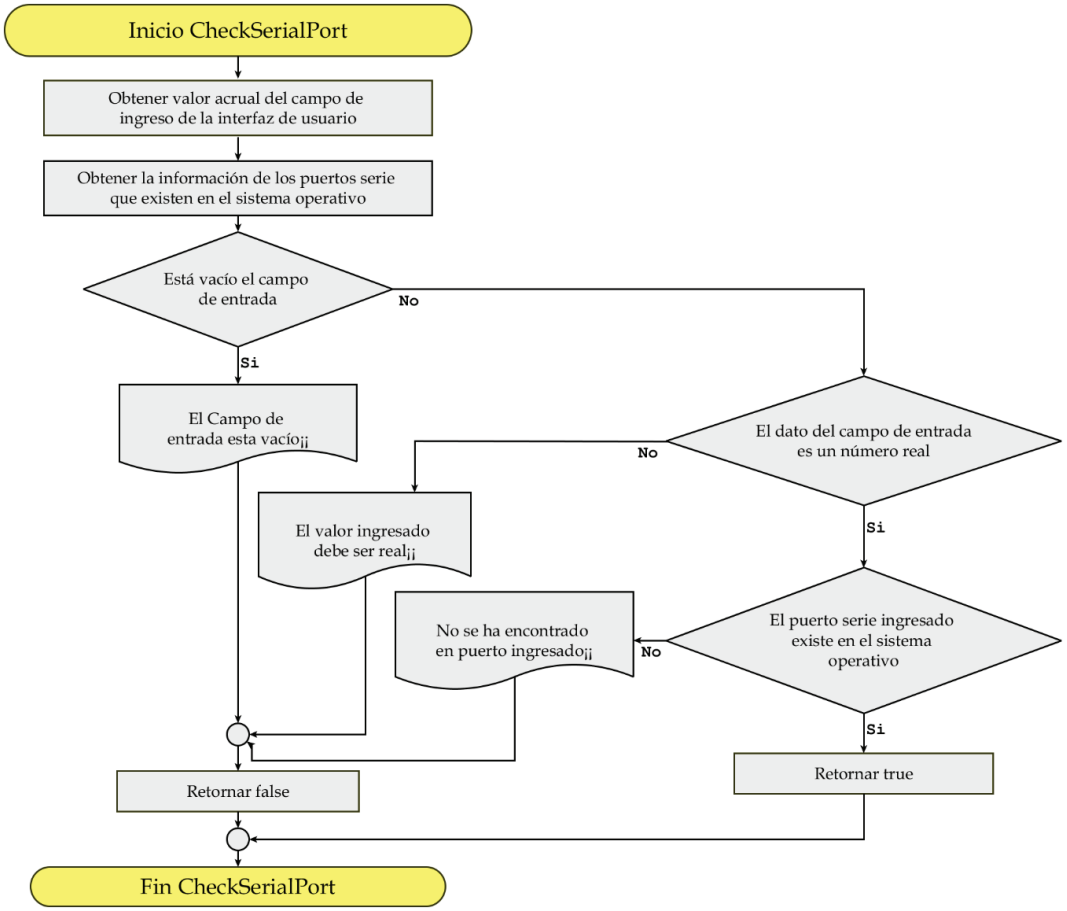



**Figura 3.30.** Función *CheckSerialPort*

El bucle central consta de dos lazos secundarios, Lazo A y Lazo B, cuya estructura se muestra en la Figura 3.31.

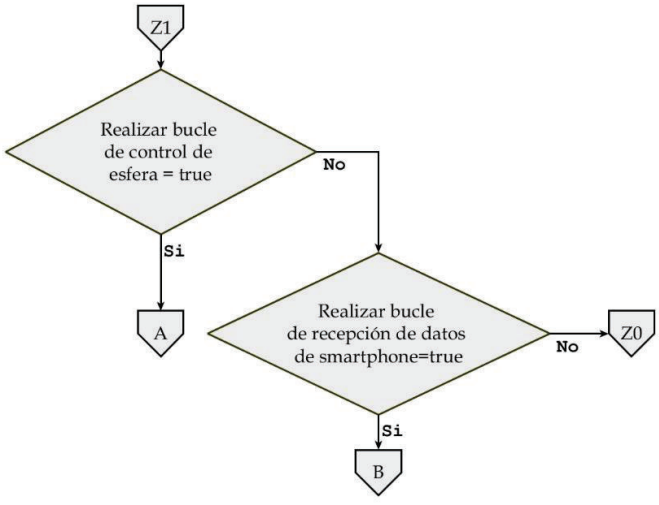



**Figura 3.31.** Lazo Principal: Lazo Central

*3.4.1.1.1 Control de posición (Lazo A)*

Este lazo secundario realiza el control de la esfera en la plataforma, su diagrama de flujo se muestra en la Figura 3.32.

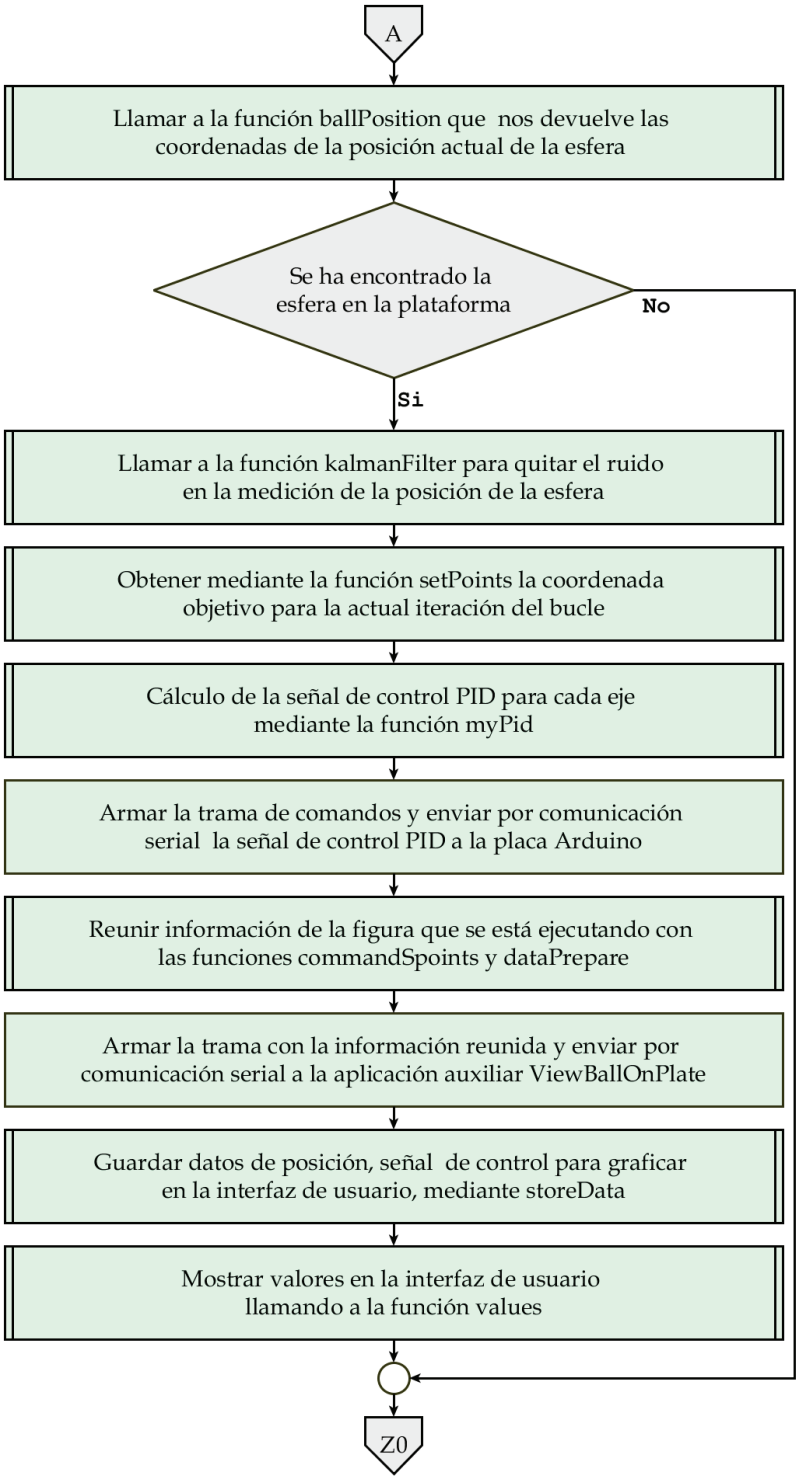



**Figura 3.32.** Lazo Central: Control de Posición

A continuación, se detalla cada una de las funciones utilizadas por el lazo de control de posición:

***Determinar Posición de la esfera*** *- Función ballPosition:*

El lazo A, primero llama a la función *ballPosition,* cuyo diagrama de flujo se muestra en la Figura 3.33. Esta función captura una imagen de la cámara usando el objeto de video creado al inicio, y utilizando técnicas de visión artificial permite determinar las coordenadas de la posición de la esfera. Si la esfera no se encontró en la imagen impide que se pase a los demás bloques para así evitar errores de cálculo y que se detenga la aplicación.

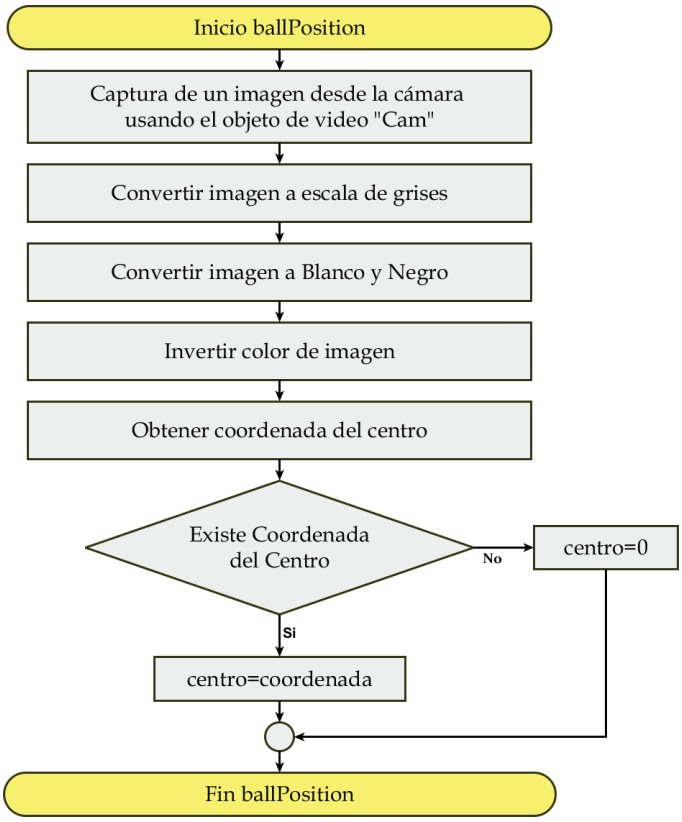



**Figura 3.33.** Función *ballPosition*

***Filtro de Kalman*** *- Función kalmanFilter:*

En segundo lugar, se llama a la función “kalmanFilter”, cuyo diagrama de flujo se muestra en la Figura 3.34. Esta función devuelve la posición estimada con base en la medida obtenida desde el observador que en nuestro caso es la cámara de video y el modelo matemático que rige el movimiento de la esfera en la plataforma. La

estimación del filtro de Kalman pretende eliminar el ruido que se cuela en la medición de la señal ya sea por vibraciones de la estructura que sostiene a la cámara o por perturbaciones que generen ruido de alta frecuencia como por ejemplo el ruido que podrían causar los servomotores.

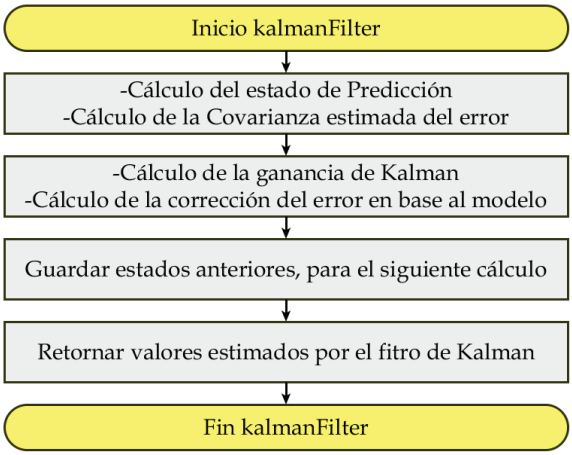



**Figura 3.34.** Función *kalmanFilter*

***Generación periódica de Puntos*** *- Función setPoints:*

En tercer lugar, se llama a la función *setPoints*, cuyo diagrama de flujo se muestra en la Figura 3.35. Esta función recibe el vector que contiene los puntos del camino escogido y devuelve un único valor de ese vector que será la referencia actual al que se llevará la esfera en esa iteración del lazo.

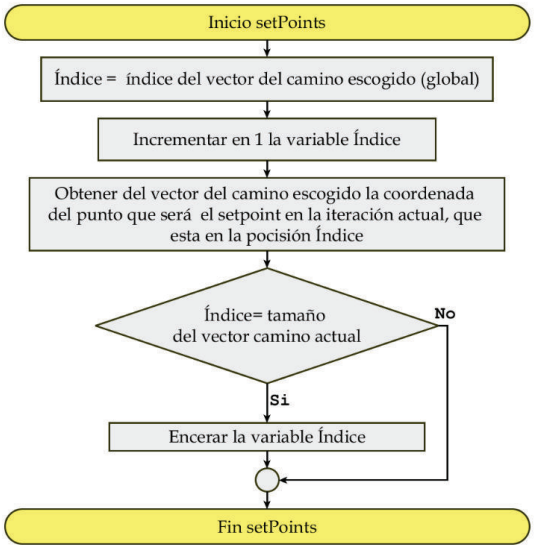



**Figura 3.35.** Función *setpoints*

***Controlador PID*** *- Función myPid:*

En cuarto lugar, con los valores de posición actual (filtrada) y referencia actual, se llama a la función *myPid,* cuyo diagrama de flujo se muestra en la Figura 3.36. Esta función devuelve el cálculo de la señal de control requerida para posicionar la esfera donde se desea:

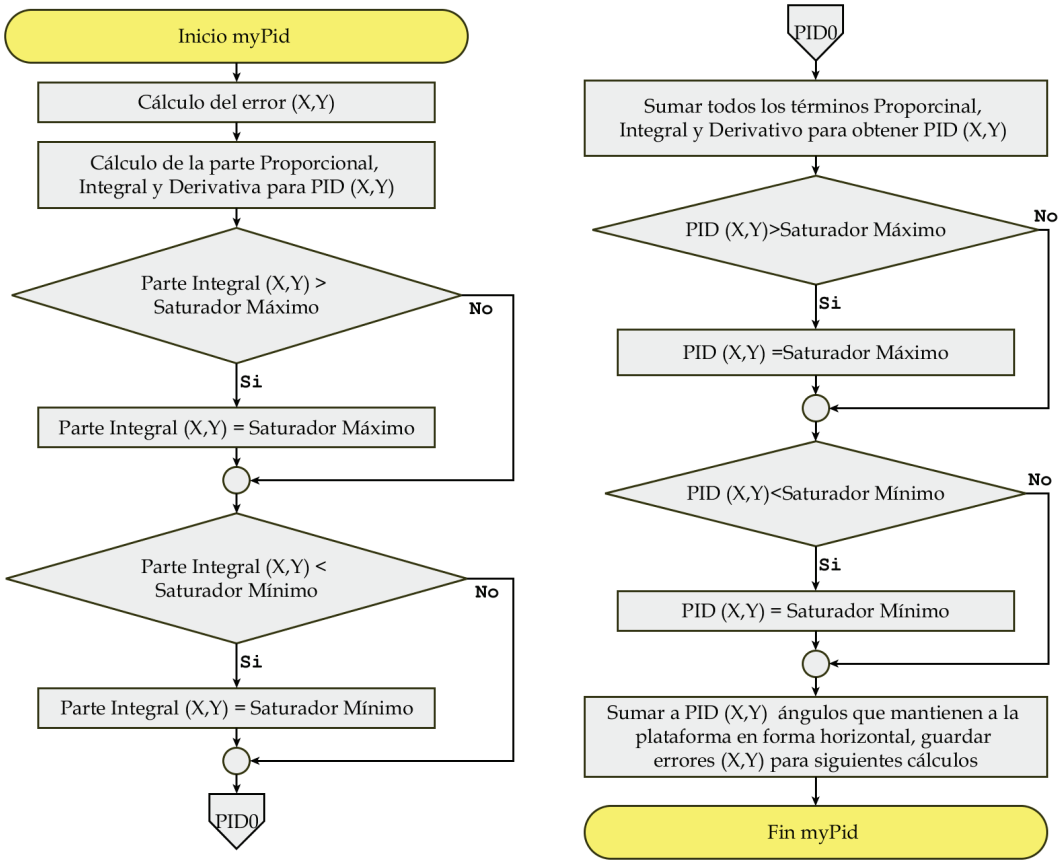



**Figura 3.36.** Función *myPid*

Después de tener listo el dato de la señal de control, se arma una pequeña trama y se envía esos datos mediante comunicación serial a la placa Arduino que transformará esa información en una PWM para accionar a los servomotores.

En seguida, se llama a dos funciones para armar una trama y enviar datos a la aplicación auxiliar (ViewBallOnPlate) que permite visualizar el camino o punto establecido y la posición actual de la esfera.

La primera función *commandSpoints*, cuyo diagrama de flujo se muestra en la Figura 3.37, regresa un número que corresponde a la figura actual, y en caso de ser un punto devuelve dicha coordenada.

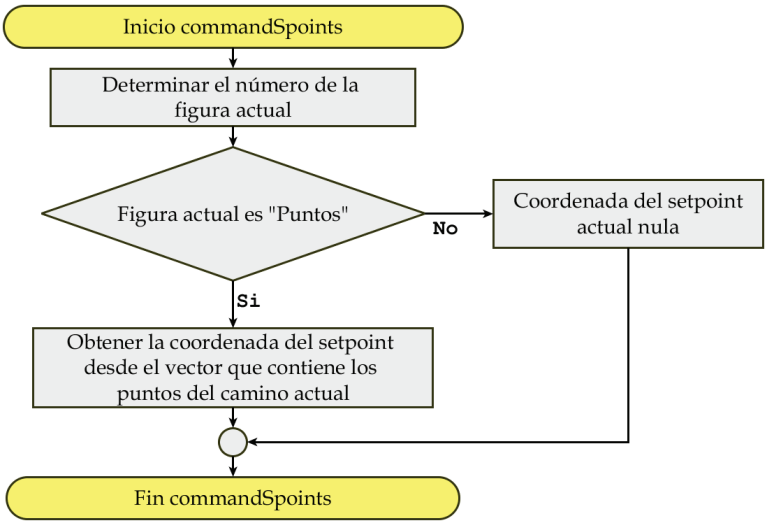



**Figura 3.37.** Función *commandSpoints*

La segunda función prepara los datos de la posición actual. La Figura 3.38 muestra el diagrama de flujo de la función *dataPrepare*. Con los datos retornados por las dos funciones se arma una trama para enviar la información mediante comunicación serial usando un puerto virtual a la aplicación auxiliar que permite visualizar el camino y la posición actual de la esfera.

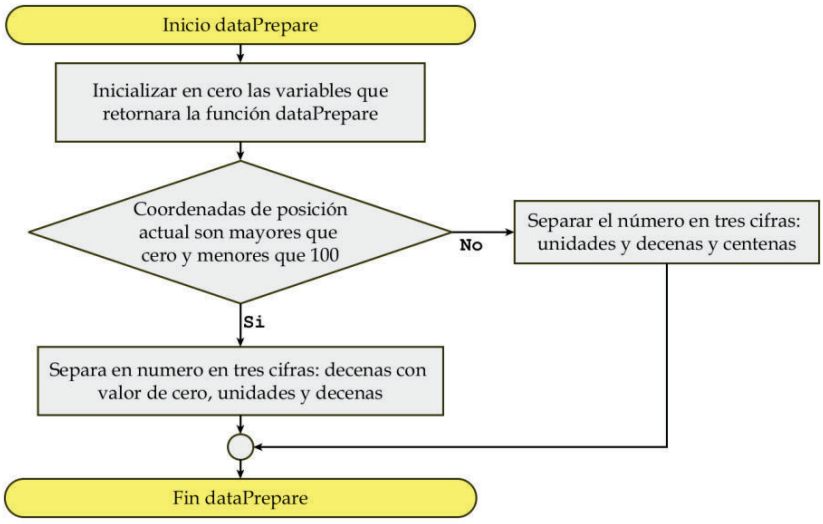



**Figura 3.38.** Función *dataPrepare*

***Almacenar Datos*** *- Función storeData:*

Con la función *storeData,* cuyo diagrama de flujo se muestra en la Figura 3.39, se almacenan los valores que se van generando: posición actual, señal de control y referencia, para luego ser graficados.

Esta función utiliza tres vectores de una longitud constante previamente definida para almacenar los valores que tiene las tres variables en cada iteración del lazo, lo que indica que llegado el momento esa capacidad se agota, por lo cual, una vez llegado al valor máximo de almacenamiento se desplaza la información contenida en los vectores una posición hacia atrás de tal manera que el nuevo dato quede en la última posición, y con esto se logra que la información se vaya actualizando sin cambiar la longitud de los vectores de almacenamiento.

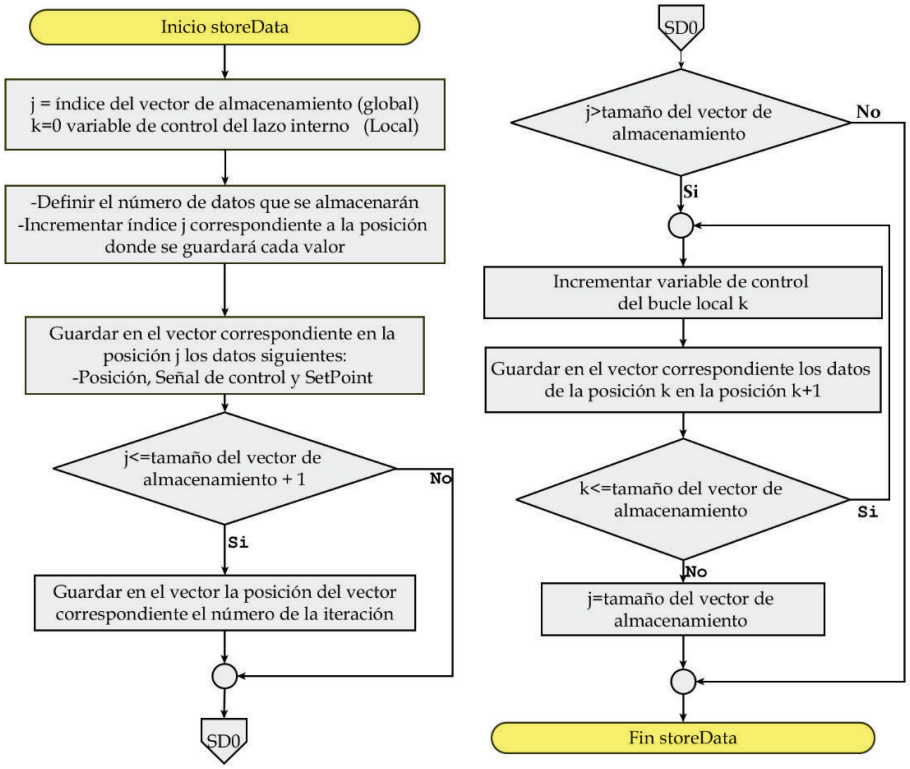



**Figura 3.39.** Función *storeData*

Por último, se llama a la función *values* que muestra en la interfaz los valores actuales de posición y referencia. Y si no se cambia la lógica desde la interfaz de usuario el ciclo vuelve a comenzar.

*3.4.1.1.2 Recepción de datos (Lazo B)*

En este lazo se recibe la trama de comandos enviada desde el Smartphone al computador. En la Figura 3.40 se muestra el diagrama de flujo para la recepción de datos.

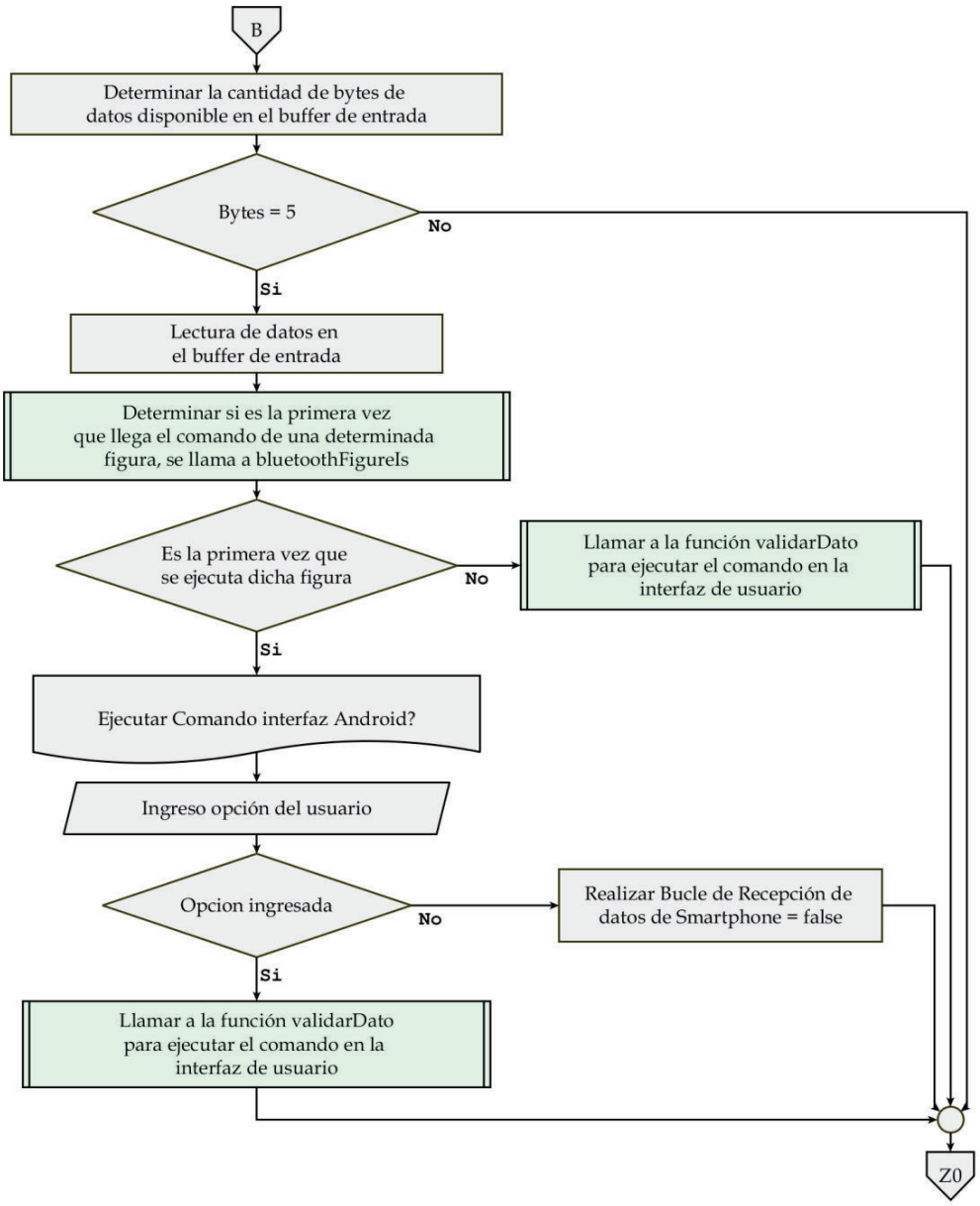



**Figura 3.40.** Lazo Central: Recepción de datos

Primero se verifica la cantidad de bytes el buffer de entrada, este debe ser igual a cinco, que corresponde con la cantidad de bytes que se envían en cada trama desde la interfaz en el Smartphone.

Si los bytes recibidos son cinco, se determina la figura que se ha recibido mediante la función *bluetoothFigureIs,* cuyo diagrama de flujo se muestra en la Figura 3.41.

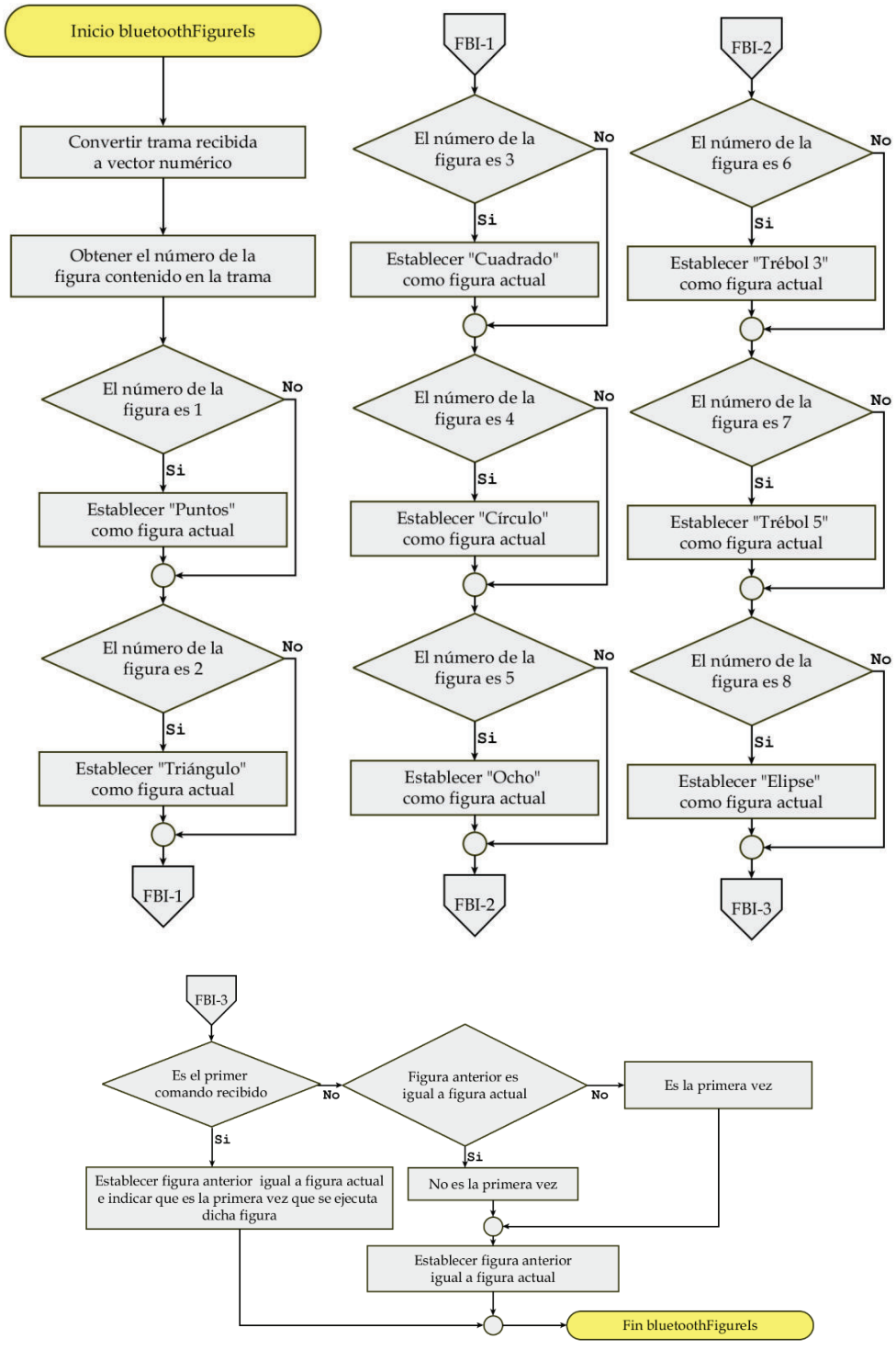



**Figura 3.41.** Función *bluetoothFigureIs*

A continuación se muestra un mensaje al usuario pidiendo confirmación para ejecutar el comando recibido, si la respuesta es “si” se llama a la función “validarDato”, cuyo diagrama de flujo se muestra en la Figura 3.42. Esta función determina la figura y el tamaño solicitado (solo para triángulos, cuadrados y circulos) o el punto deseado y según eso llama al callback de cada función requerida.

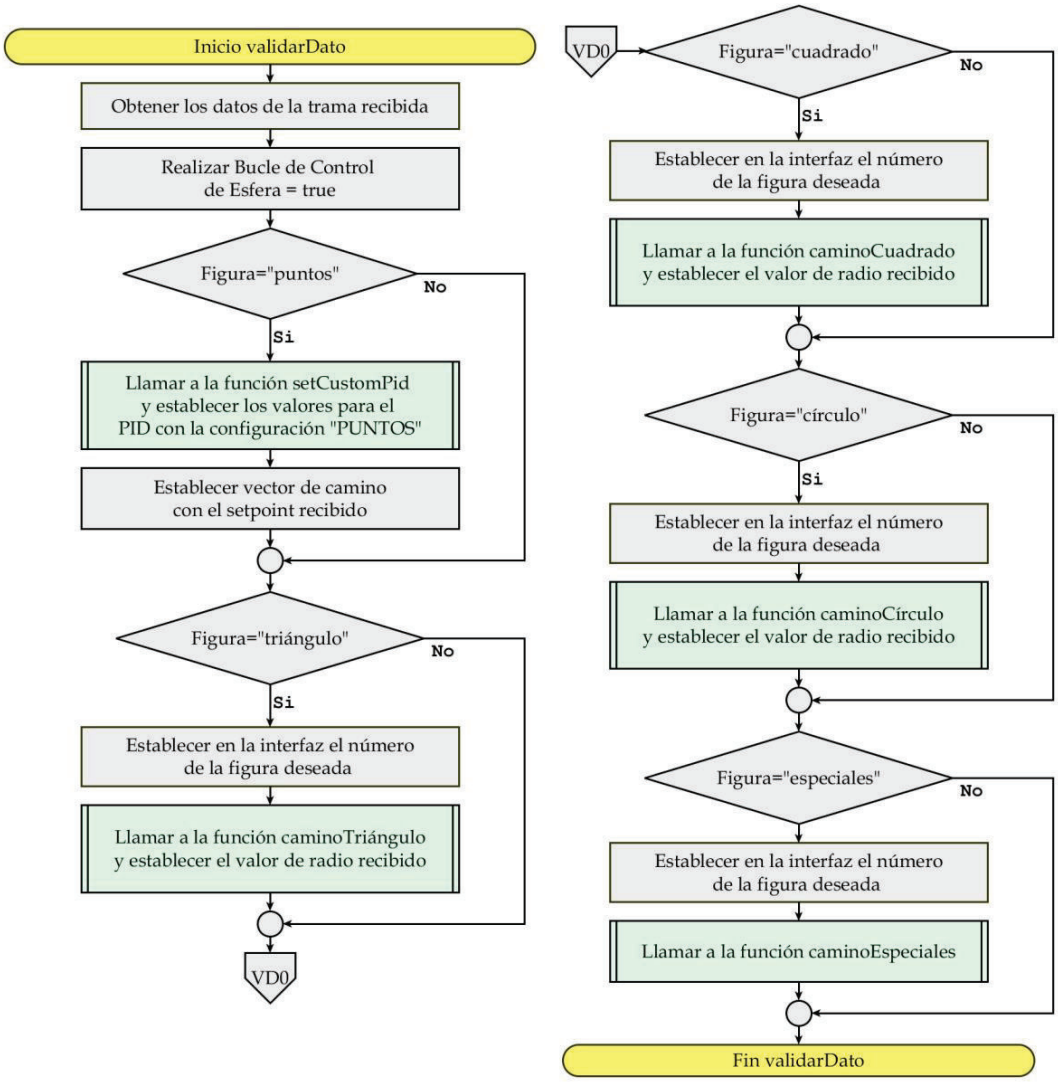



**Figura 3.42.** Función *validarDato*

Las funciones que responden cuando desde la interfaz de usuario se presionan los botones para establecer una figura como camino actual: Triangulo, Cuadrado, Circulo, Especiales o Puntos se indican a continuación:

***Triángulos y Cuadrados*** - *Función caminoTriángulo* y *caminoCuadrado*

La función *caminoTriángulo*, cuyo diagrama de flujo se muestra en la Figura 3.43, tiene la misma lógica y estructura que la función *caminoCuadrado*, por ello solo se indica una de ellas.

Esta función es el callback que se ejecuta cuando en la interfaz se presiona el botón "Triangulo", la misma que toma el valor del radio actual del campo "Pop-up-menú" y genera el vector Camino, análogamente sucede con el callback del botón caminoCuadrado.

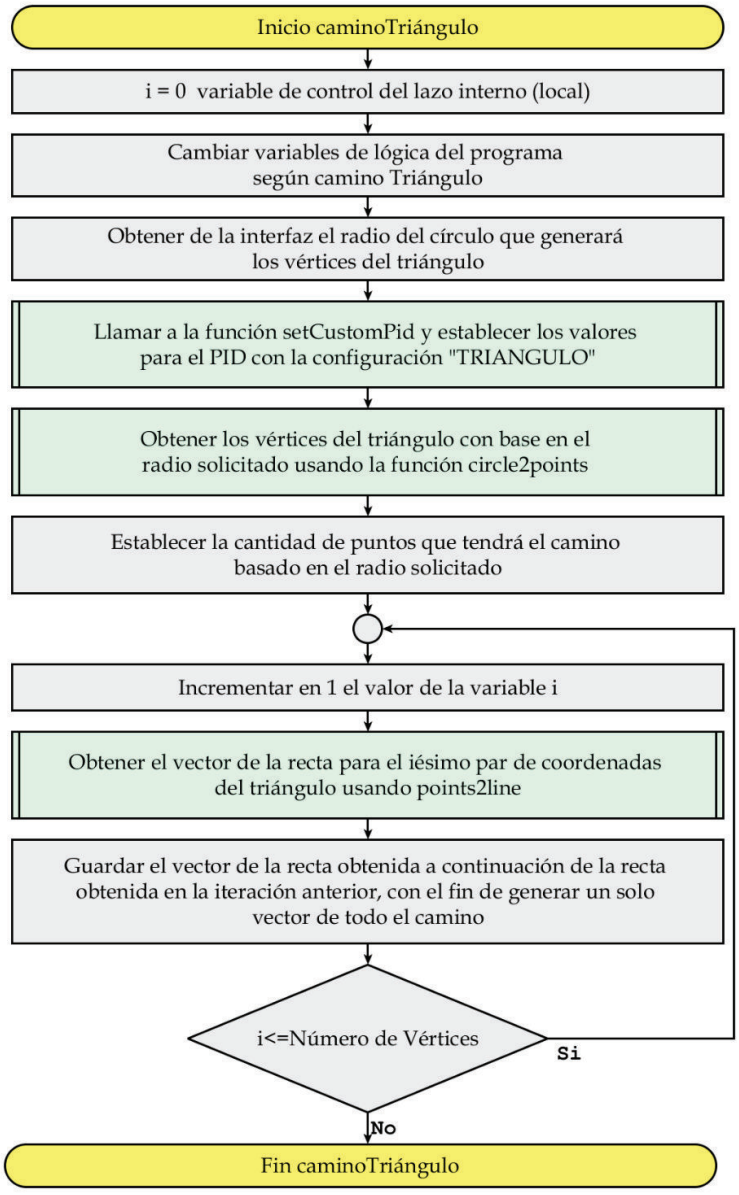



**Figura 3.43.** Función *caminoTriángulo*

Las instrucciones en las funciones *caminoTriángulo o caminoCuadrado* invocan a la función *setCustomPid,* cuyo diagrama de flujo se muestra en la Figura 3.44. Esta función establece los valores del PID que se utilizarán para realizar el camino, esta última contiene los valores para la configuración del PID para todas las figuras de este proyecto:

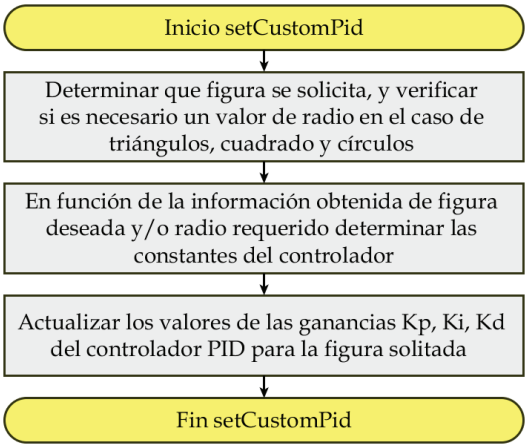



**Figura 3.44.** Función *setCustomPid*

Además, usando la función *circle2points*, cuyo diagrama de flujo se muestra en la Figura 3.45, se obtiene los vértices del triángulo o cuadrado solicitado:

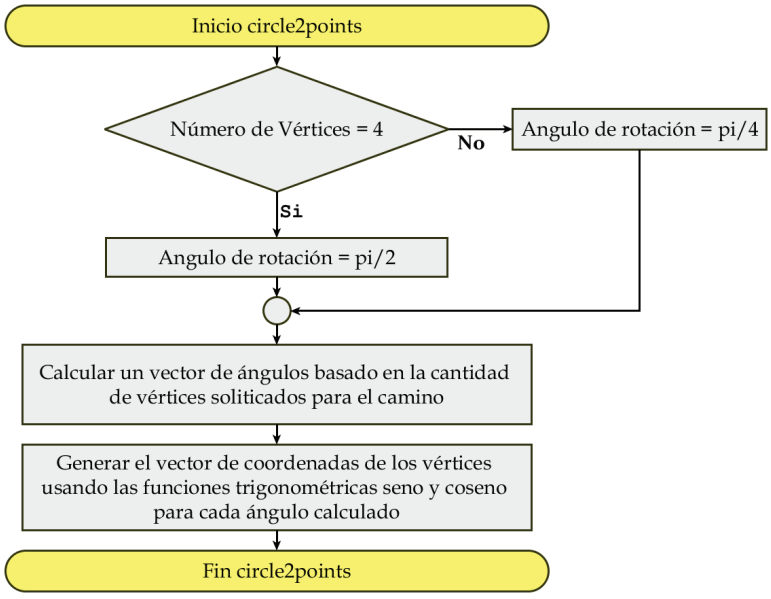



**Figura 3.45.** Función *circle2points*

Mediante la función *points2line,* cuyo diagrama de flujo se muestra en la Figura 3.46, se obtiene el vector completo que contiene los puntos del camino deseado.

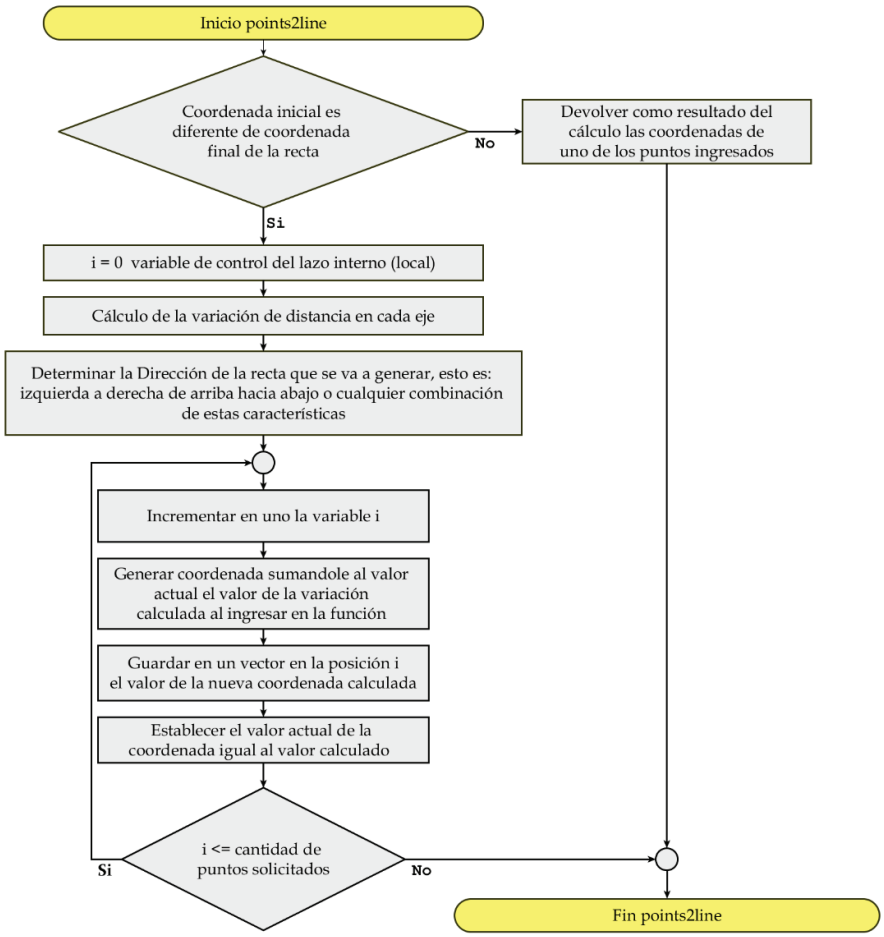



**Figura 3.46.** Función points2line

***Círculos y Especiales*** - *Función caminoCirculo y caminoEspeciales:*

La función *caminoCírculo* tiene la misma lógica y estructura que la función *caminoEspeciales*, y llaman a las funciones *circle* y *especiales* respectivamente para obtener el vector de puntos del camino solicitado según sea el valor del radio o el tipo de figura especial deseada.

Son cuatro figuras especiales que se pueden realizar en este proyecto: número Ocho, trébol de tres foliolos, trébol de cinco foliolos y elipse.

Las cuatro figuras especiales tienen tamaño fijo, no se permite al usuario modificar sus valores, ni el radio ni el número de puntos. Las Figuras 3.47, 3.48 y 3.49 detallan los diagramas de flujo para generar cada tipo de camino.

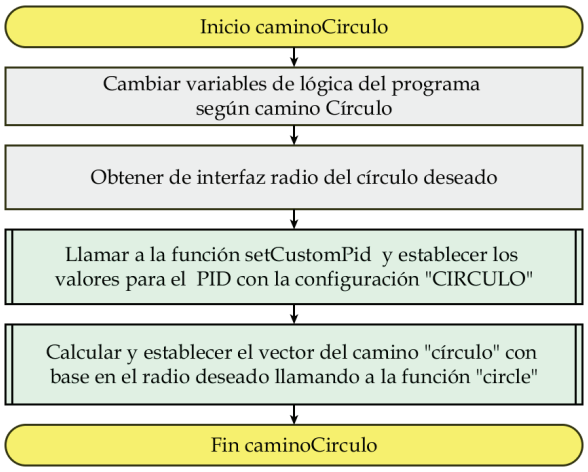



**Figura 3.47.** Función *caminoCírculo*

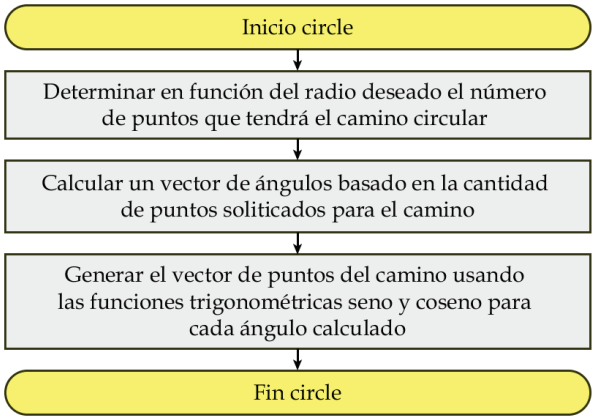



**Figura 3.48.** Función *circle*

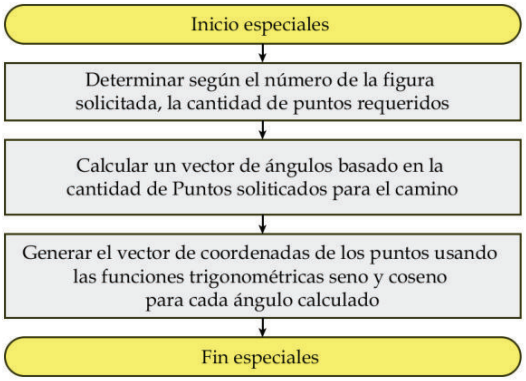



**Figura 3.49.** Función *especiales*

***Puntos*** - *Función caminoPuntos:*

La función callback del botón “Puntos”, habilita al usuario para poder definir el punto deseado dando clic sobre un eje de coordenadas, para lograr esto establece una función receptora del evento “presionar clic” llamada *ButtonDownFnc* e integrándola en la función *dibujarPunto*. Las Figuras 3.50 y 3.51 detallan el diagrama de flujo de las funciones encargadas de generar las referencias para el control de posición.

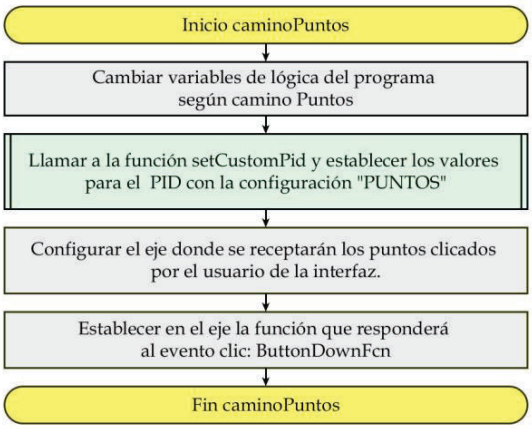



**Figura 3.50.** Función *caminoPuntos*

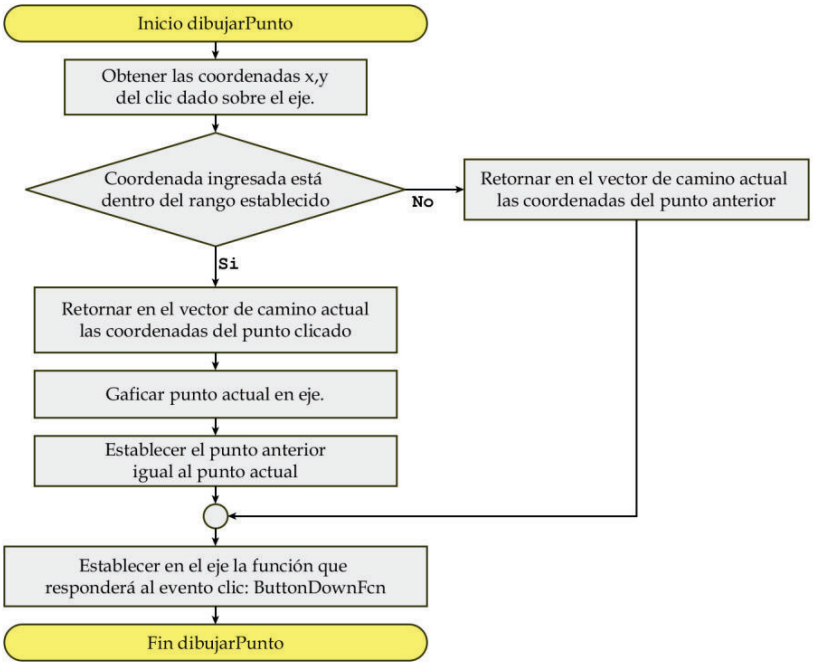



**Figura 3.51.** Función *dibujarPunto*

Cuando se corre el programa de la interfaz de control, y se muestra por primera vez tan solo están habilitados el botón “Guardar configuración” y los campos de ingreso del número de puertos serie para la conexión con la placa Arduino y para la conexión vía bluetooth con el Smartphone, así como los Pup-up-menú para escoger la velocidad de transmisión, como se observa en la Figura 3.52.

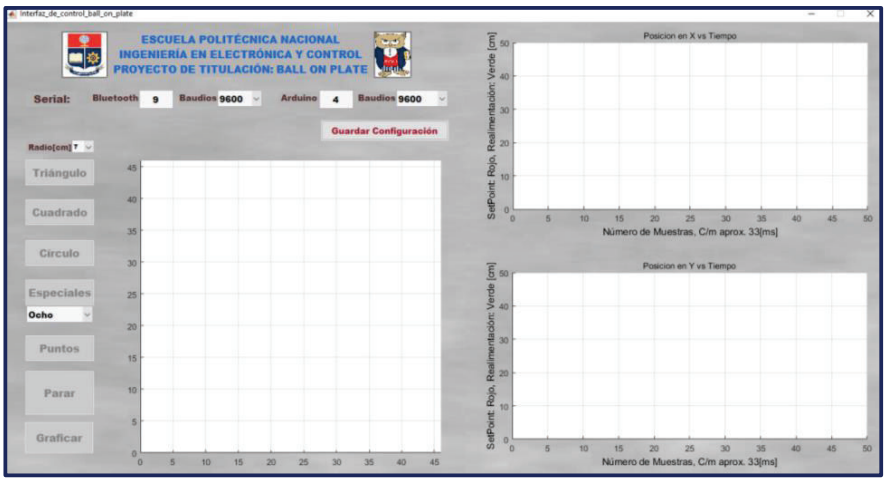



**Figura 3.52.** Interfaz de Control: Inicial

Una vez establecida la configuración y si no ocurre ningún problema con el hardware (cámara y módulos de comunicación serial alámbrica o inalámbrica) este botón se oculta mostrando los campos que indican los valores actuales de posición y referencia. La función *pararContinuar* es el callback asociado al botón Parar y tiene la lógica mostrada en la Figura 3.53.

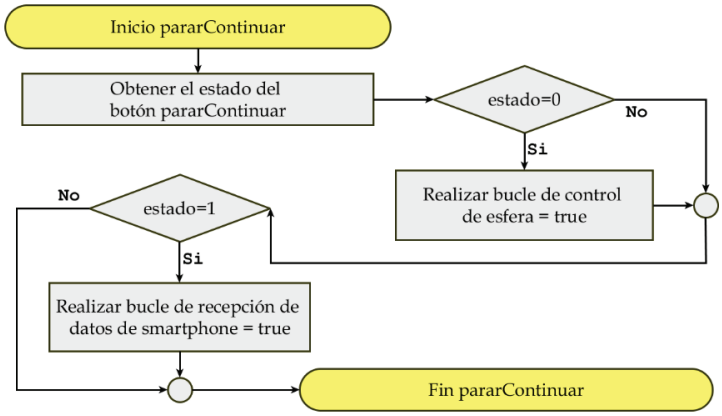



**Figura 3.53.** Función *pararContinuar*

Cuando se quiere ver los gráficos obtenidos, se presiona el botón Parar, esta acción detiene la ejecución del camino o punto que se esté realizando en ese momento y se habilita el botón Graficar. También cambia el flujo del programa para habilitar el Lazo B para la recepción de comandos enviados desde el celular.

El botón “Parar”, una vez presionado cambia a “Continuar”, de tal manera que con el mismo botón se puede detener y continuar la ejecución de la figura referencia actual o en su defecto escoger una nueva.

Además, el botón “Parar” deshabilita todos los botones de Caminos para evitar que mientras se están mostrando las gráficas se pueda iniciar la ejecución de otra figura, ya que eso podría provocar un mal funcionamiento del controlador por el aumento excesivo del tiempo de lazo.

La Figura 3.54, muestra una captura de la interfaz una vez que se ha establecido la configuración inicial y se ha detenido una figura en ejecución.

En ella se puede observar las gráficas: Posición en X vs Unidad de tiempo y Posición en Y vs Unidad de tiempo:

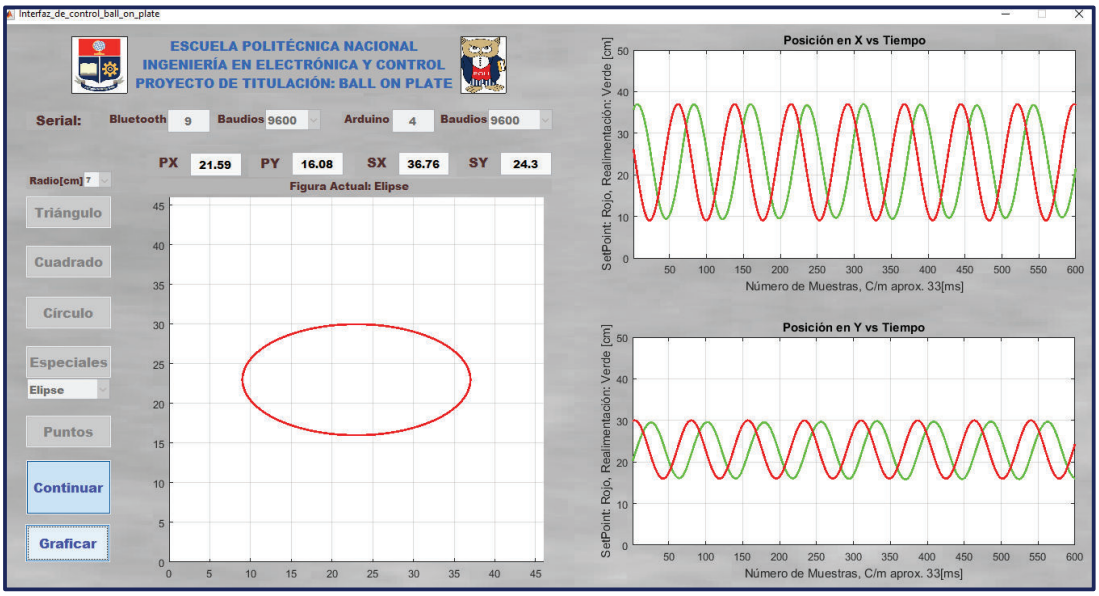



**Figura 3.54.** Interfaz de Control: Graficas Posición vs Tiempo

La Figura 3.55 muestra que al dar clic por segunda vez en el botón graficar se observa solamente: Señal de control X vs Unidad de tiempo y Señal de control Y vs Unidad de tiempo:

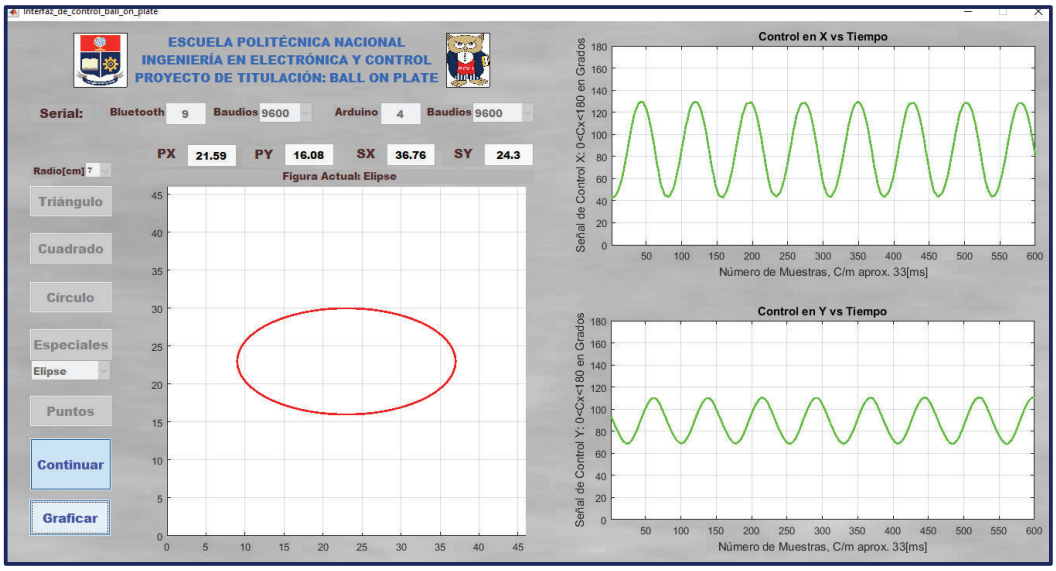



**Figura 3.55.** Interfaz de Control: Graficas Control vs Tiempo

La Figura 3.56 muestra que al dar clic por tercera vez en el botón graficar, se observa: Posición en X vs Posición en Y:

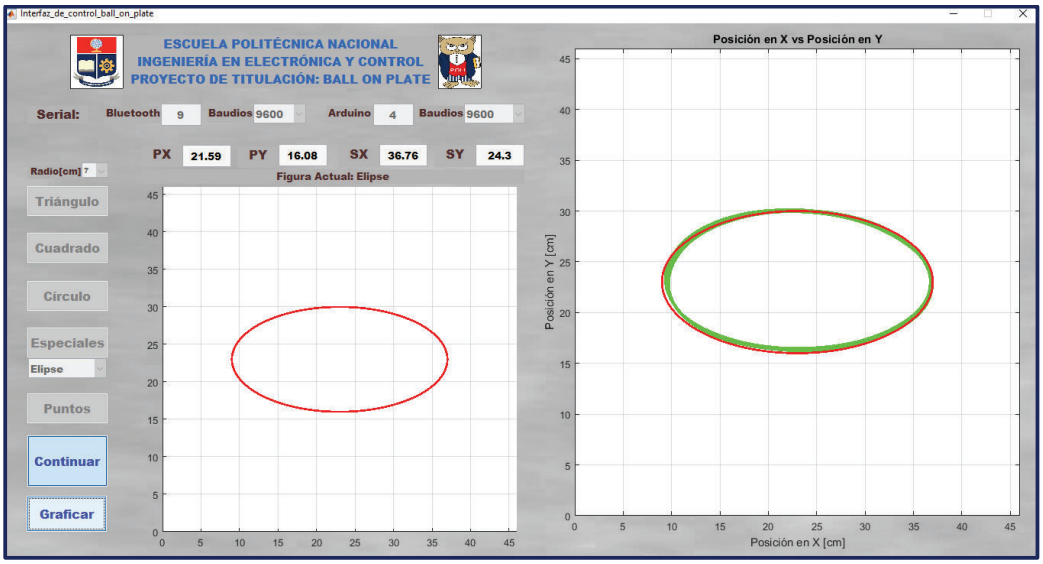



**Figura 3.56.** Interfaz de Control: Graficas Posición X vs Posición Y

Cuando se da clic por cuarta vez en el botón graficar se reinicia la cuenta y se vuelve a la primera gráfica.

De forma general, la Interfaz está compuesta de las secciones especificadas en la Figura 3.57:

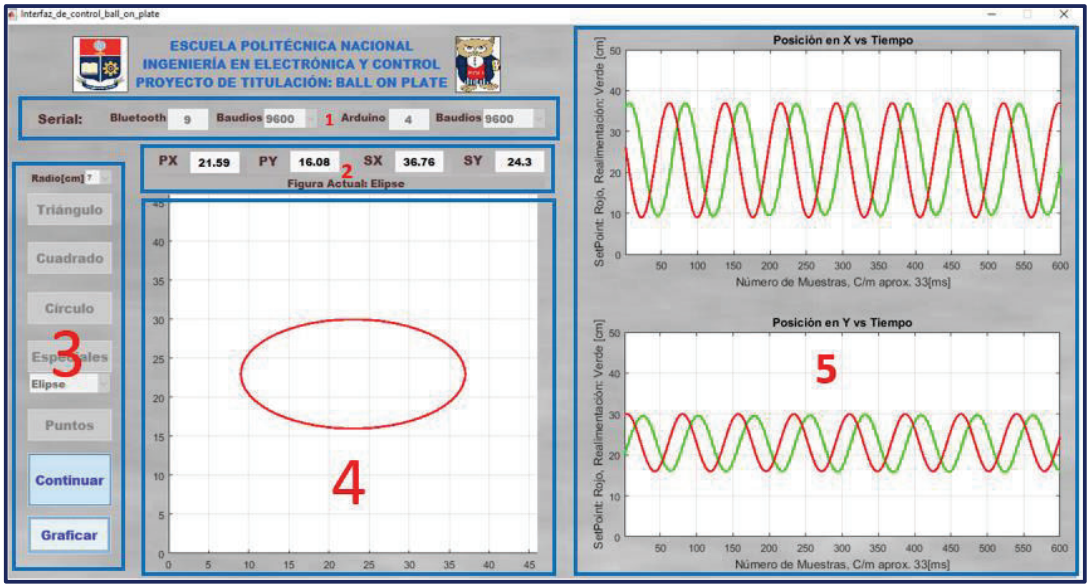



**Figura 3.57.** Interfaz de Control: Secciones

1. Campos de ingreso de número de puerto serial y velocidad de transmisión.
2. Datos actuales: Referencia y Posición.
3. Botones para seleccionar Caminos y Puntos, y gráficas.
4. Eje que indica la figura actual y permite recibir las coordenadas deseadas mediante clic del mouse.
5. Ejes para graficar: Posición y Control.

### 3.4.2 DISEÑO DE LA INTERFAZ SECUNDARIA (SMARTPHONE)

La interfaz móvil está diseñada con un entorno de desarrollo integrado que habilita la creación de aplicativos para la plataforma Android, llamado Android Studio de la compañía Google. La interfaz de Android Studio se indica en la Figura 3.58.

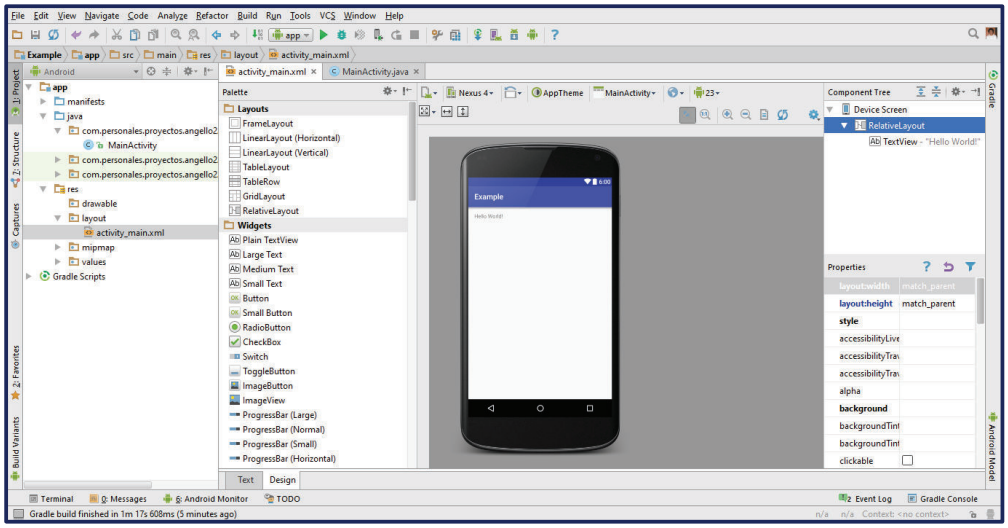


**Figura 3.58.** Interfaz de Android Studio

### 3.4.2.1 Programación en Android Studio: Conceptos básicos [33]

Para programar en Android Studio se utiliza el lenguaje de programación Java, que se basa en el paradigma de la programación orientada a objetos, por lo que es necesario definir ciertos conceptos necesarios para el desarrollo de la aplicación deseada.

*Objetos:* Un objeto es una entidad dentro de la programación que posee unas características y un comportamiento que le permite realizar tareas específicas. Las características o atributos de un objeto se denominan datos y al comportamiento de estos, se los denomina métodos.

*Clases:* Una clase es una abstracción de la realidad, es decir, permite abstraer las características y los comportamientos asociados a ella. Una clase es el molde para construir objetos, es por ello que a un objeto se le llama instancia de clase.

En la Figura 3.59, se muestra un ejemplo para aclarar estas ideas.

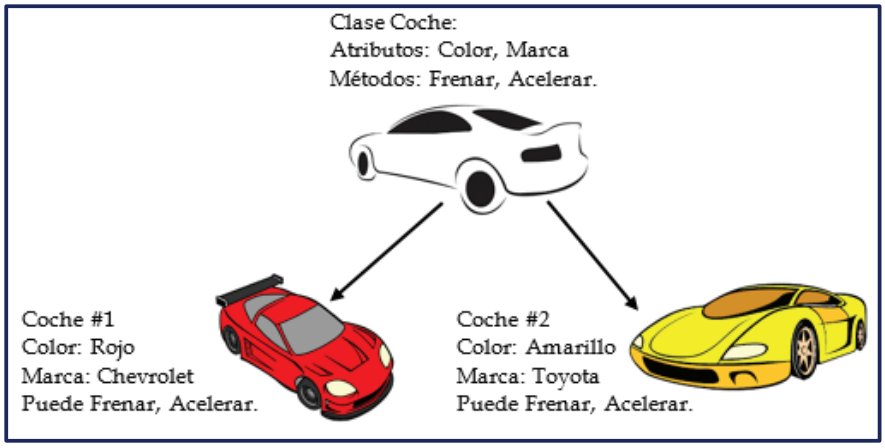



**Figura 3.59.** Clases y Objetos

*Herencia.* - La herencia es una característica de la programación orientada a objetos que permite a una clase derivar o heredar características o comportamientos de otras clases existentes. La clase de la que se hereda se denomina clase padre o superclase y la clase que hereda se denomina clase hija o subclase.

*Encapsulación*. – Es ocultar y delimitar el acceso a los datos propios de un objeto de tal manera que solo se puedan acceder a ellos mediante los métodos definidos para dicho objeto, es decir, no hay otra forma de interactuar con un objeto que no sea por medio de los comportamientos asociados a él.

*Polimorfismo.* – Es la cualidad que tienen los objetos de responder de distinta forma al mismo llamado, esto permite nombrar a varias acciones o métodos de una clase con el mismo nombre, tan solo variando ciertos parámetros como son el tipo o número de variables.

*Actividad* (Activity). – Cada aplicación está conformada por una o varias actividades, que se pueden definir como las pantallas que se muestran al usuario cada vez que son invocadas y traídas al primer plano, se conforman de una parte grafica que contiene información de cómo se ve la pantalla y una parte lógica que contiene el código o algoritmo que determina su comportamiento.

El ciclo de vida de una actividad desde que iniciada hasta que es eliminada, y el momento cuando está corriendo (momento cuando ejecuta el algoritmo del programador) se indican en la Figura 3.60:

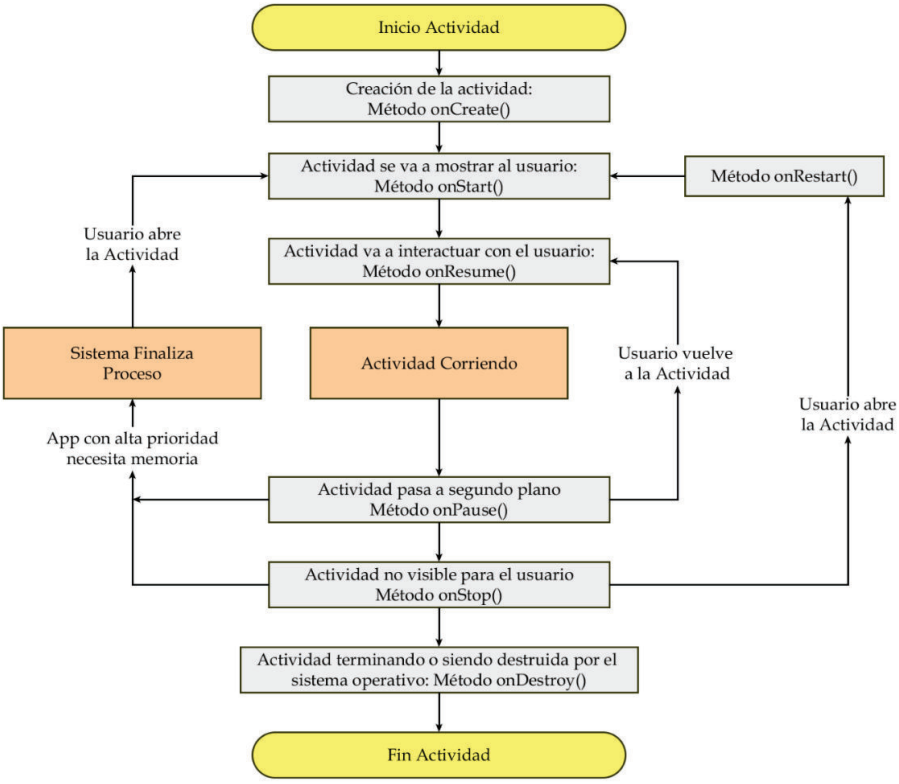



**Figura 3.60.** Ciclo de vida de una Actividad. [34]

### 3.4.2.2 Desarrollo de la aplicación.

La función principal de la aplicación es enviar tramas de comandos que permitan establecer caminos y puntos de referencia, a través de una conexión bluetooth. La aplicación móvil tiene varias actividades que se indican en la Figura 3.61. Una vez que corre la aplicación se tiene una pantalla que muestra un menú, el mismo que consta de las siguientes opciones: Referencias de puntos y caminos, Acerca de, Ayuda y Salir.

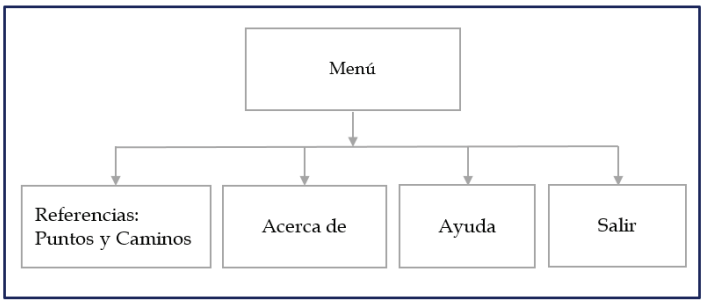



**Figura 3.61.** Actividades de la aplicación BallOnPlate.

La actividad Menú tiene la estructura lógica mostrada en la Figura 3.62. Una vez que la aplicación esta en dicha actividad, se solicita al usuario que encienda la radio bluetooth o en su defecto que autorice a la aplicación para encenderla.

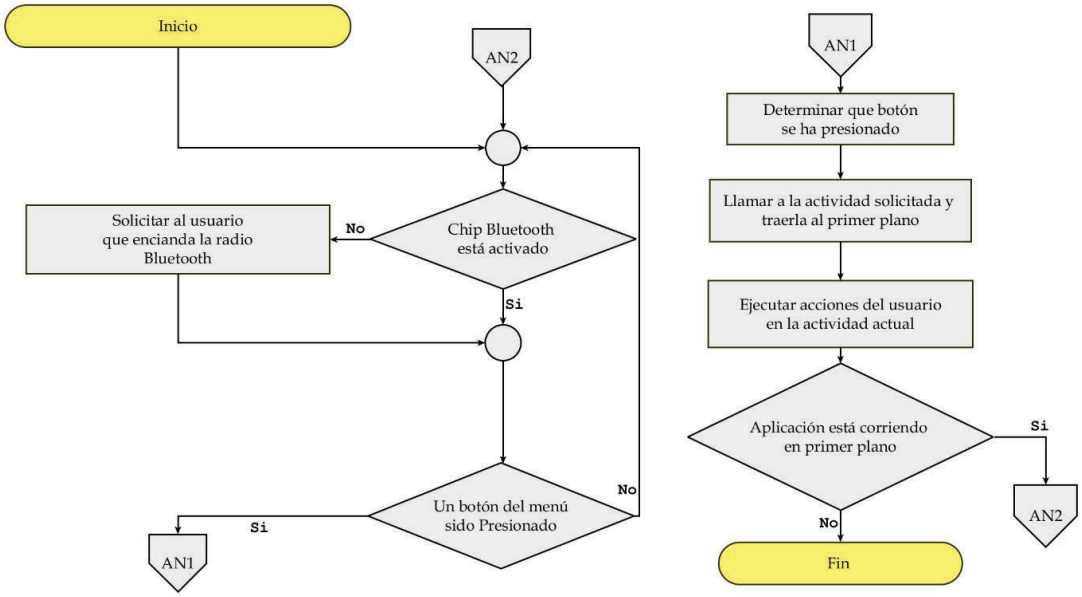



**Figura 3.62.** Estructura lógica de la actividad Menú.

La actividad menú posee cuatro botones, como se indica en la Figura 3.63, estos botones están enlazados de tal manera que cuando se presionan llevan al usuario a la actividad solicitada.


**Figura 3.63.** Menú de opciones, aplicación BallOnPlate.

La primera opción, llamada Métodos de Control, es la actividad que contiene los botones para seleccionar el camino o punto deseado, el cual una vez seleccionado se dibuja en la grilla superior de la pantalla, además de la conexión con el modulo bluetooth. En la Figura 3.64 se muestra como está estructurada la lógica de dicha actividad.

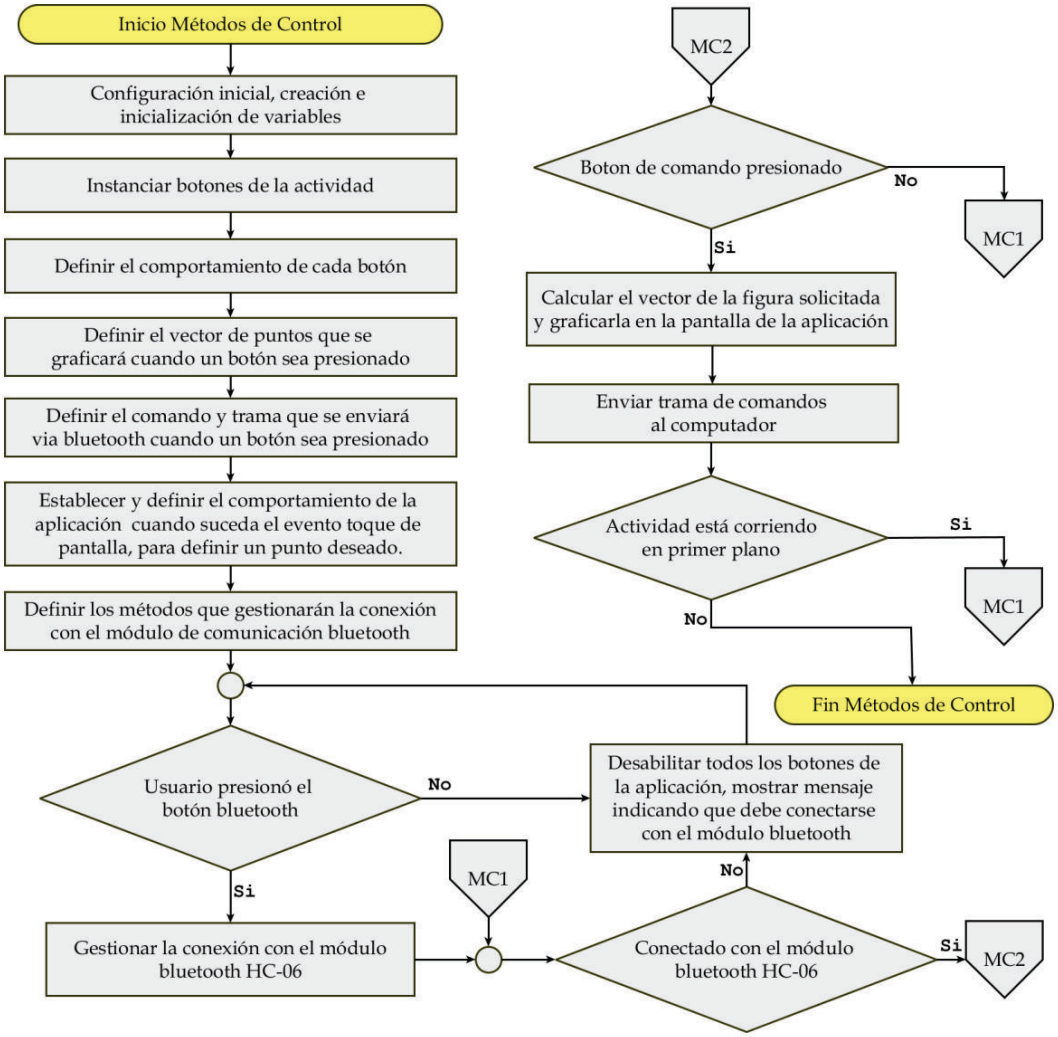



**Figura 3.64.** Estructura lógica de la actividad Métodos de Control.

Cuando la actividad está corriendo o en primer plano están activos los fragmentos de código que se ejecutan cuando ocurren los eventos de “botón presionado” y “toque de pantalla”, denominados “Listeners”.

En primer lugar, se habilita el dispositivo bluetooth del teléfono, luego se espera por el evento clic del botón “Bluetooth” para establecer la conexión con el módulo HC-06, después se mantiene a la escucha para atender cuando el usuario presiona algún botón para establecer algún camino o punto, y por ultimo enviar dicha información al computador.

En la Figura 3.65, se indica el resultado final de la implementación de la actividad Métodos de Control, en la cual se observa la grilla donde se grafican las referencias de puntos o caminos seleccionados por el usuario, los botones con los que el usuario seleccionara la figura deseada. Los puntos deseados como referencia son obtenidos del evento “toque de pantalla” el cual se habilita cuando el usuario pulse el botón “Puntos” de la aplicación.

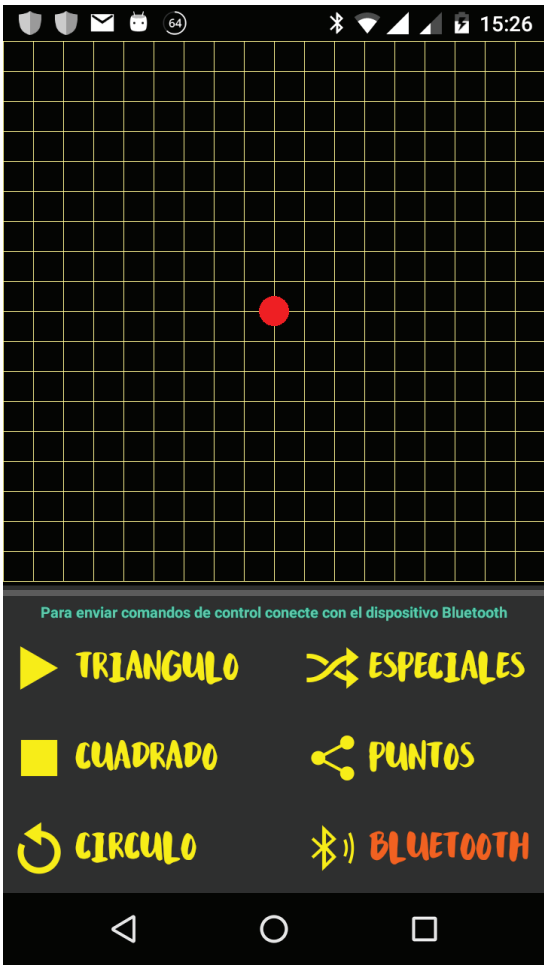



**Figura 3.65.** Referencias de puntos y caminos.

La segunda opción es la actividad que muestra una breve información de los desarrolladores de la aplicación. La tercera actividad muestra información importante acerca de la aplicación y la plataforma (Figura 3.66), y la cuarta opción permite salir del aplicativo.


**Figura 3.66.** Ayuda, aplicación BallOnPlate

### 3.4.3 DISEÑO DE LA INTERFAZ AUXILIAR

La aplicación auxiliar también está diseñada mediante programación orientada a objetos, en el entorno de programación NetBeans y su funcionamiento es muy básico: recibe los datos enviados desde Matlab y los grafica de tal manera que se puedan visualizar la figura deseada y la posición actual de la esfera en línea.

Esta aplicación ha sido implementada para evitar el aumento del tiempo de la ejecución del lazo que controla la esfera, ya que si se usa alguna herramienta en Guide para graficar la referencia y la posición actual mientras se está realizando el control de la esfera, esto produce un incremento considerable del tiempo de ejecución del lazo de control que repercute en el buen funcionamiento del sistema.

Por el contrario, enviando por comunicación serial los datos y graficándolos en la aplicación auxiliar tan solo se consume el tiempo del envío serial el cual es menor que 1 ms lo que no afecta en nada al funcionamiento de la plataforma a la hora de realizar una figura.

En la Figura 3.67 se observa el funcionamiento de la aplicación auxiliar y se puede ver lo sencilla que es:

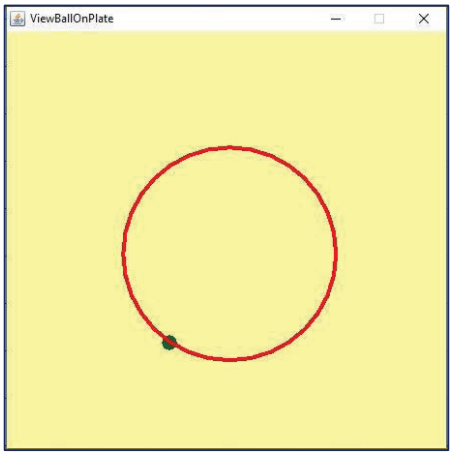



**Figura 3.67.** Aplicación Auxiliar ViewBallOnPlate.

## 3.5 IMPLEMENTACIÓN DEL PLOGRAMA EN ARDUINO

El programa implementado en la placa Arduino Uno recibe del computador, por comunicación serial, el valor del ángulo en grados a cargar en los servomotores. Finalmente, este dato se convierte en un valor de salida de la PWM. Para este fin se utilizó el puerto serial número 0 y los Timers 1 y 2 para generar la señal de control.

La trama que llega desde el computador está formada por tres bytes, un encabezado y los ángulos de giro tanto para el eje X como para el eje Y.

El encabezado tiene un valor constante de 255 y sirve como referencia para posteriormente separa los datos siguientes. El segundo dato de la trama corresponde al ángulo a cargar en el servomotor del eje X. Este ángulo sexagesimal puede contener valores que van desde 0 hasta 180 es decir, la inclinación del aspa del servomotor se desplazará como máximo 180 grados. El tercer dato de la trama corresponde al ángulo a cargar en el servomotor del eje Y. De igual manera este ángulo varía entre 0 y 180 y tiene las mismas consideraciones que en el eje X. La Figura 3.68 muestra la trama de comunicación con el computador y los valores de cada dato.

| *Encabezado* | *Dato Servo X* | *Dato Servo Y* |
|---|---|---|
| 255 | 0 *a* 180 | 0 *a* 180 |

**Figura 3.68.** Trama de comunicación entre el computador y la placa Arduino.

Con el fin de evitar posibles problemas generados por las interrupciones seriales, se decidió ubicar todo el programa en el lazo principal (void loop). Dentro de este lazo, se verifica constantemente el buffer de entrada, preguntando por la llegada de un dato. Una vez leído el encabezado (255) se sabe que el siguiente dato corresponde al ángulo del servo en el eje X y el siguiente dato corresponde al ángulo en el eje Y.

Se ha utilizado la librería de servomotores proporcionada por el IDE de Arduino. En esta librería basta con cargar el valor en grados que se desea para generar la señal PWM.

En la Figura 3.69 se detalla el diagrama de flujo del programa implementado en la placa Arduino.

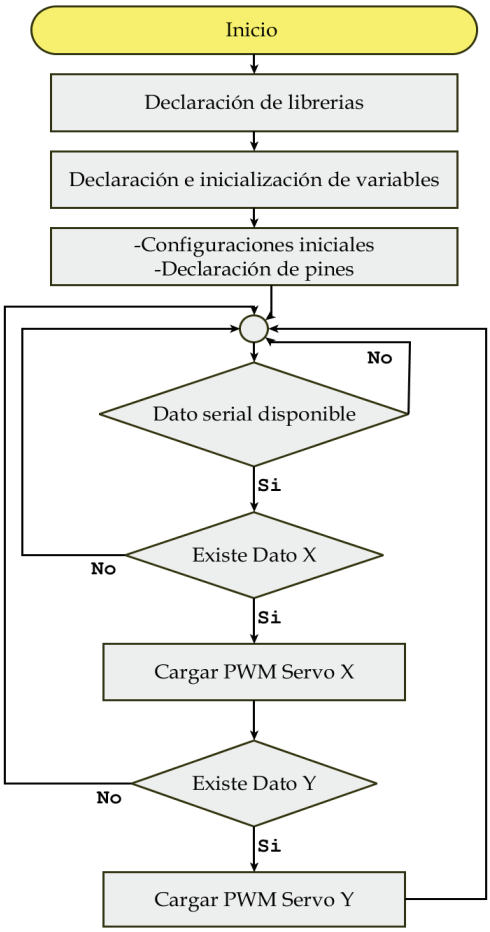



**Figura 3.69.** Diagrama de flujo – Programación en Arduino.

Es importante especificar que, debido a la ubicación del servomotor en los soportes (posición horizontal), el valor del ángulo para el cual la plataforma se mantiene horizontal es de 90 grados para los servomotores. En la Figura 3.70 se muestran tres posiciones del eje del servomotor como ejemplo.

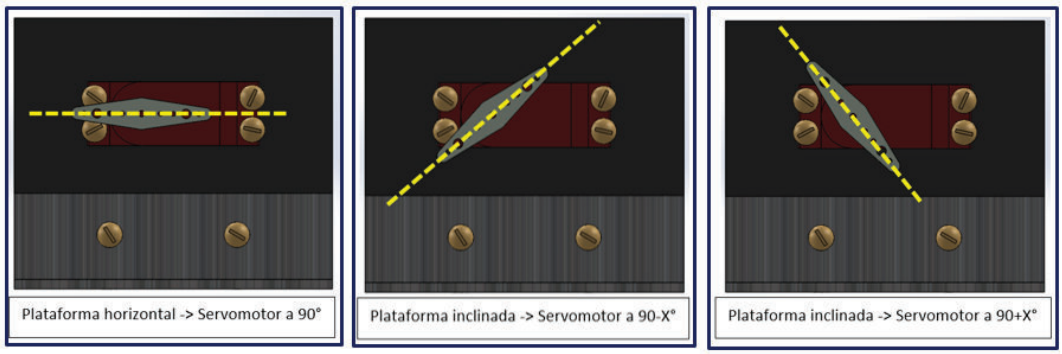



**Figura 3.70.** Ángulos de los servomotores.

Como se observa en la Figura 3.70, para realizar cualquier desplazamiento del aspa del servomotor, se tiene como referencia la posición horizontal (90 grados). Es decir que los ángulos calculados por el controlador tienen una estructura de: 90 más un limite o 90 menos un límite. Los límites representan la saturación del PID. Cada camino de referencia, dependiendo de la necesidad de inclinación de la plataforma, tiene su propio valor, siendo el límite máximo de 90 grados, pues el rango de funcionamiento, como se mencionó anteriormente, está entre 0 y 180 grados.

# CAPÍTULO 4
# PRUEBAS Y RESULTADOS

## 4.1 PRUBEAS – FILTRO DE KALMAN

### 4.1.1 PRUEBAS DE SEGUIMIENTO DE LA ESFERA

Esta prueba tiene el objetivo de comprobar que la salida del filtro siga de manera similar al valor tomado por la cámara, cuando no hay presencia de ruido en la medida.

La primera prueba que se realizó, es para comprobar que tan “ruidosa” es la medida de la cámara. Se dejó estática a la esfera en una posición y se obtuvieron los siguientes resultados mostrados en las Figuras 4.1 y 4.2.

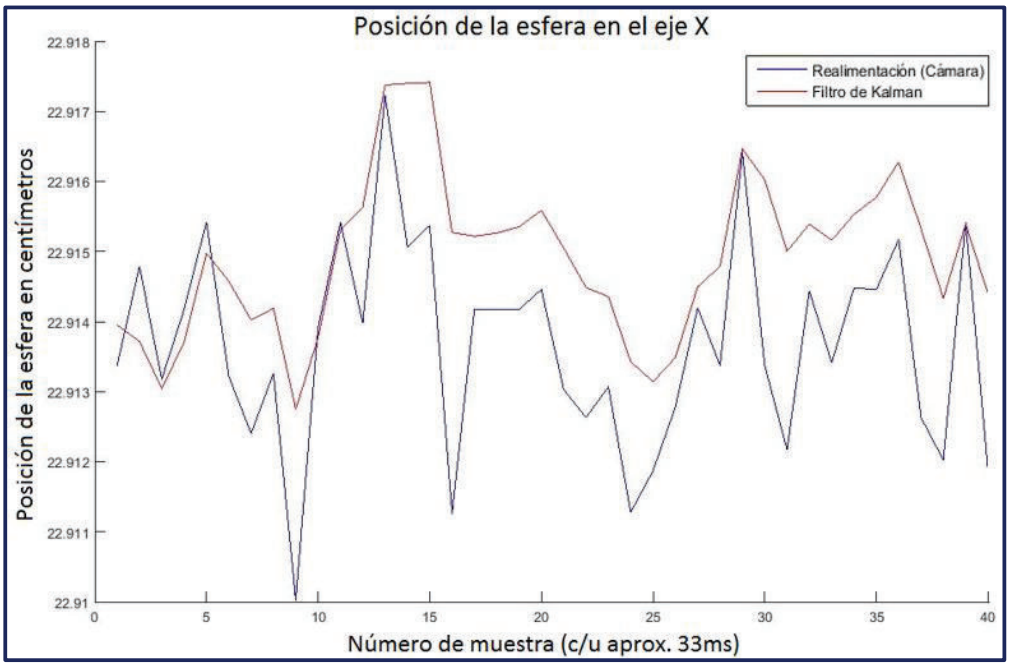



**Figura 4.1.** Pruebas del filtro de Kalman con esfera estática (Eje X).

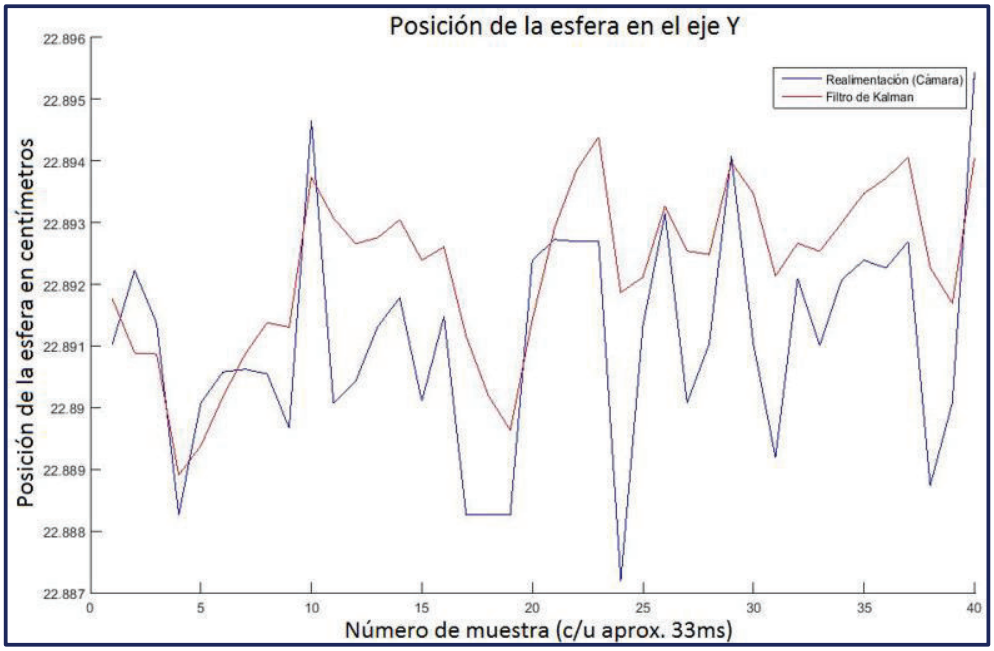



**Figura 4.2.** Pruebas del filtro de Kalman con esfera estática (Eje Y).

De los resultados se puede notar que la cámara no presenta gran cantidad de ruido cuando el sistema y la medida se mantienen estáticos. La diferencia entre el pico más alto y el pico más bajo, no supera el 1 milímetro.

Se realizaron también pruebas de seguimiento cuando no hay acción oscilatoria de la plataforma para comprobar que el filtro no se retrase de la medida actual. Estos resultados se muestran en las Figuras 4.3, 4.4 y 4.5.

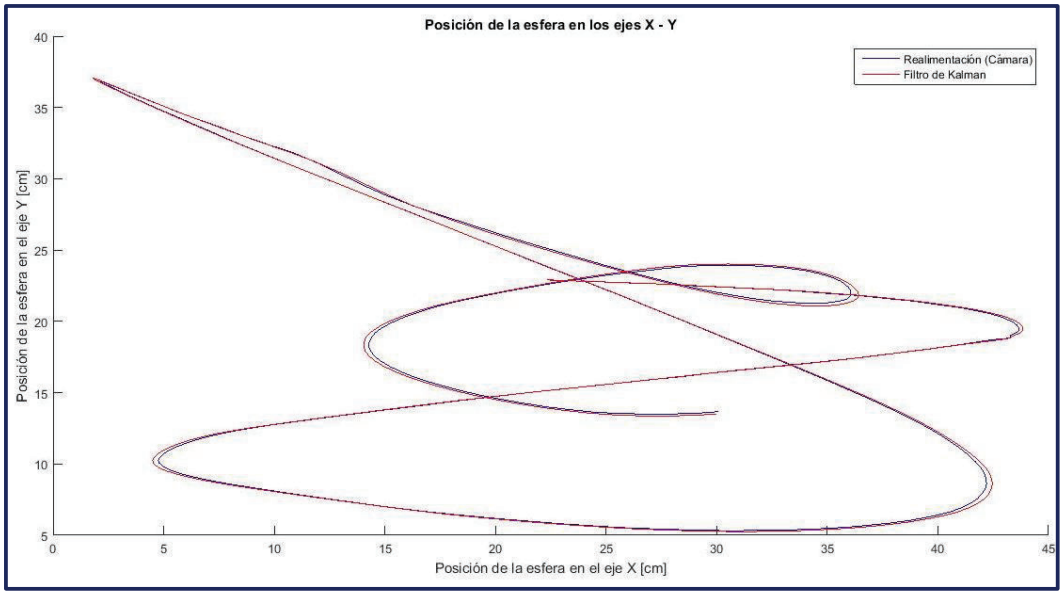



**Figura 4.3.** Pruebas de seguimiento de la esfera utilizando el filtro de Kalman (X vs Y).

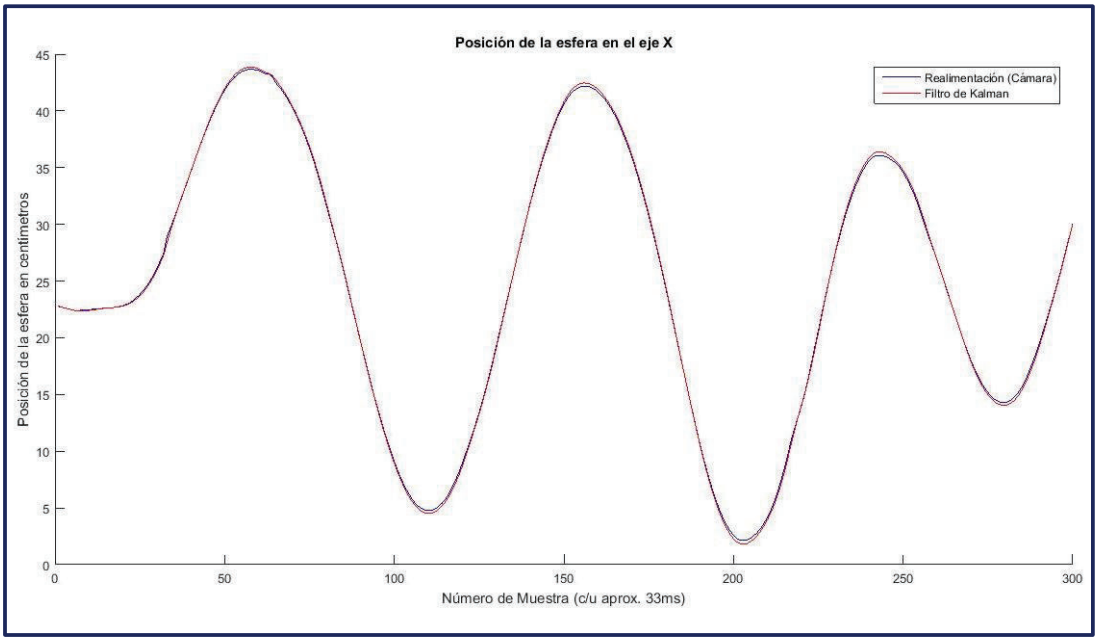



**Figura 4.4.** Pruebas de seguimiento de la esfera utilizando el filtro de Kalman (Eje X).

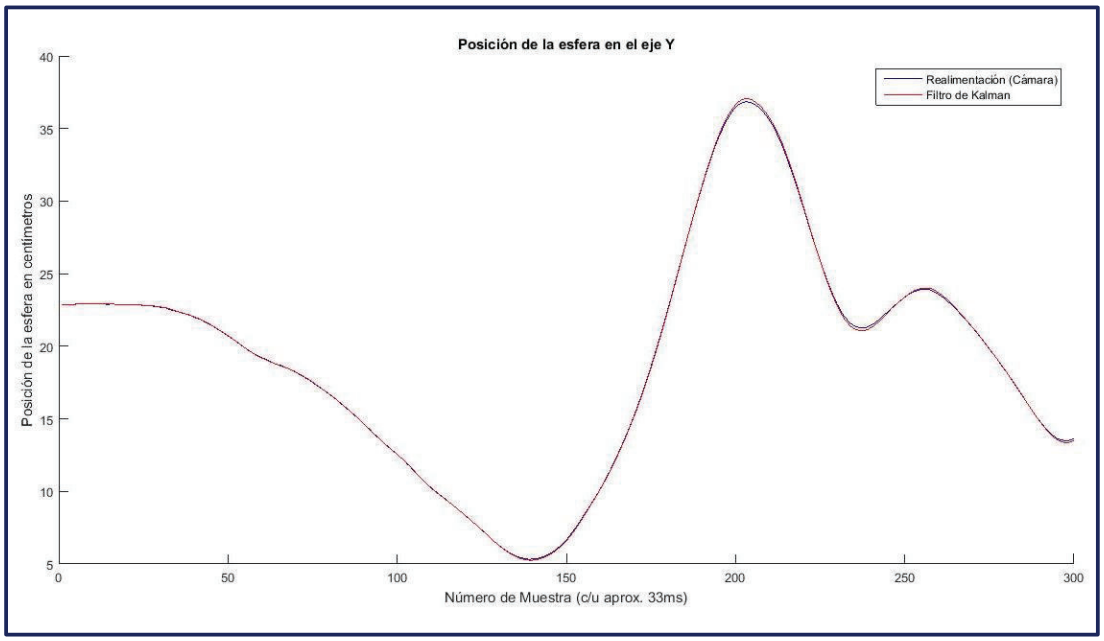



**Figura 4.5.** Pruebas de seguimiento de la esfera utilizando el filtro de Kalman (Eje Y).

De estos resultados se concluye que la salida del filtro de Kalman sigue de buena manera a la señal de referencia, sin presentar retardos considerables en el seguimiento.

### 4.1.2 PRUEBAS DE SEGUIMIENTO CON RESPUESTA AL RUIDO

Para poder visualizar los efectos del filtro de Kalman al momento de atenuar el ruido provocado por los servomotores, se realizaron pruebas de seguimiento de la esfera con oscilación de la plataforma. La mayor oscilación se obtiene en el eje X que es donde descansa mayormente el peso de la plataforma y donde está montada la cámara. Se obtuvieron los resultados mostrados en las Figuras 4.6, 4.7 y 4.8.

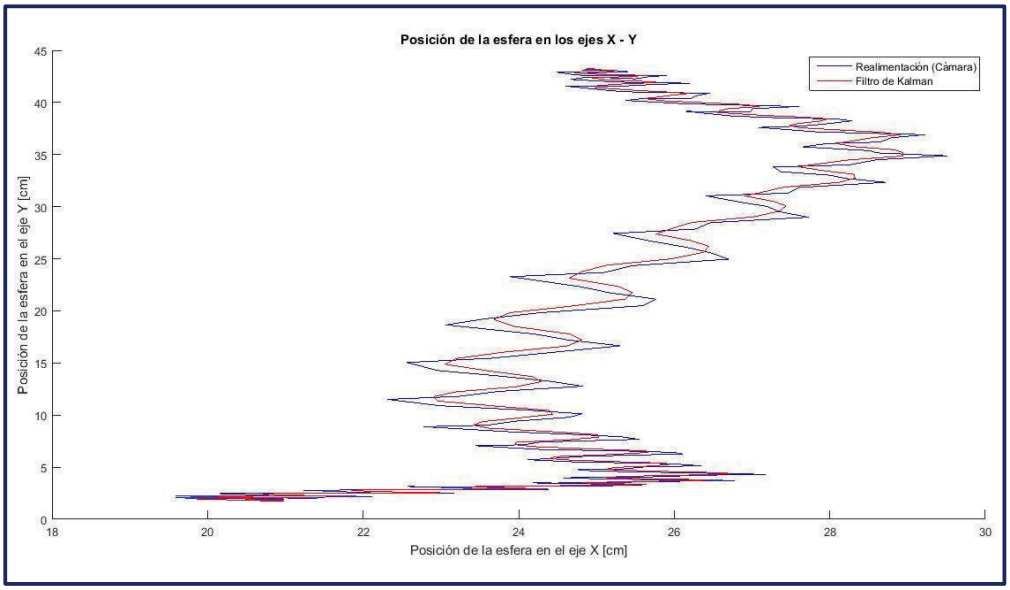



**Figura 4.6.** Respuesta del filtro de Kalman ante oscilaciones de la plataforma (X vs Y).

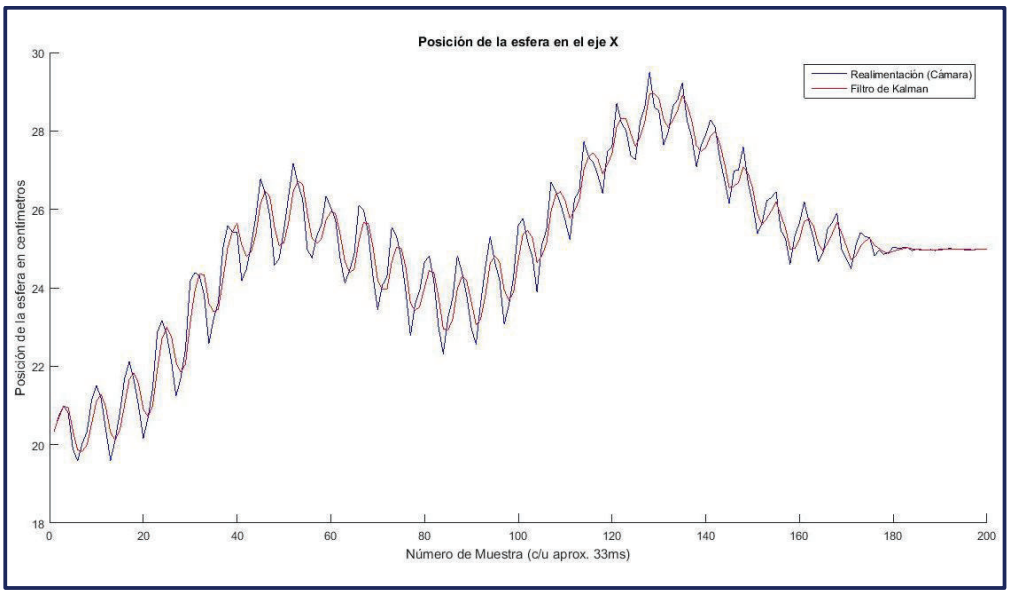



**Figura 4.7.** Respuesta del filtro de Kalman ante oscilaciones de la plataforma (Eje X).

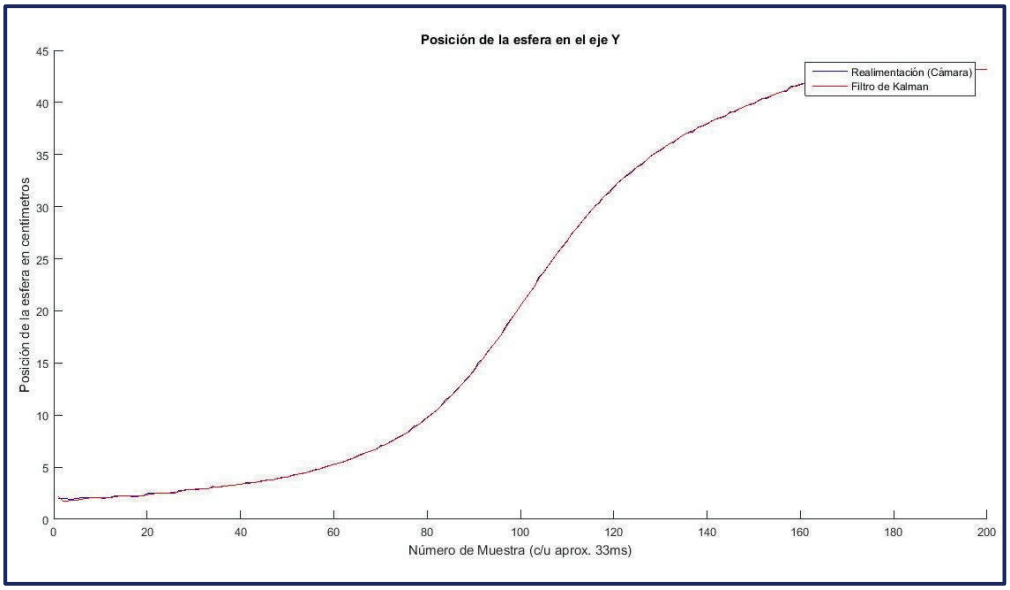



**Figura 4.8.** Respuesta del filtro de Kalman ante oscilaciones de la plataforma (Eje Y).

Cuando se observa los resultados en el eje X, donde se produce la oscilación, de color azul se muestra el valor medido por la cámara, y entre una muestra y otra se presentan variaciones de hasta 2 cm. Estas variaciones altas, producen que el controlador tome acciones correctivas hacia un sentido y hacia el otro bruscamente. Esto provoca que la oscilación se incremente aún más y hace imposible el seguimiento de caminos. Con ayuda del filtro de Kalman, esta variación se reduce hasta un 50%, lo que disminuye las oscilaciones paulatinamente y permite realizar el seguimiento de caminos sin problema.

También se observa que en el eje (eje Y) donde no se presentan las oscilaciones, el seguimiento es normal, sin modificaciones.

Se podría atenuar aún más el ruido producido por las oscilaciones, esto aumentando la covarianza del modelo, sin embargo, al realizar esto, el seguimiento de la esfera se retrasa y se observa que cuando la esfera realiza curvas, el filtro se pasa antes de recuperar el seguimiento, esto debido a que se dio más confianza al modelo que a la medida, por lo que, en cambios repentinos de dirección como esquinas, no responde de manera adecuada.

## 4.2 PRUEBAS – SEÑAL DE CONTROL

De acuerdo al manual de funcionamiento de los servomotores, el periodo de la PWM debe estar entre 15 ms y 20 ms. Se eligió usar 20ms de periodo.

Se realizaron pruebas para comprobar la señal de control de los servomotores. Los resultados obtenidos son los esperados y se muestran en las Figuras 4.9 y 4.10.

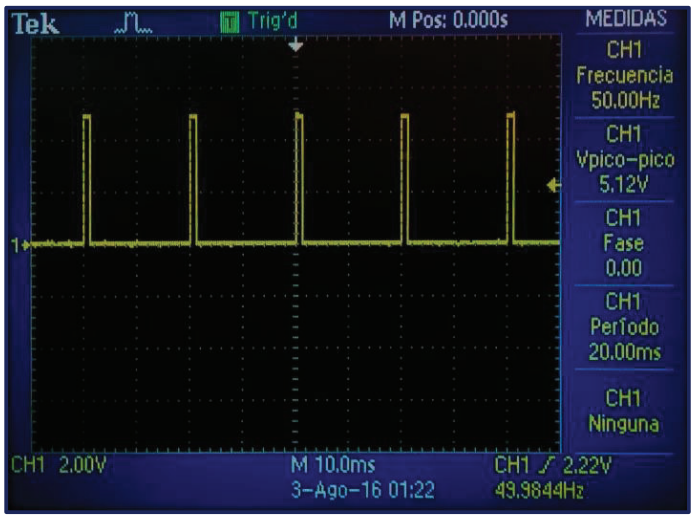



**Figura 4.9.** Pruebas de la señal de control para un ángulo de 90 grados.

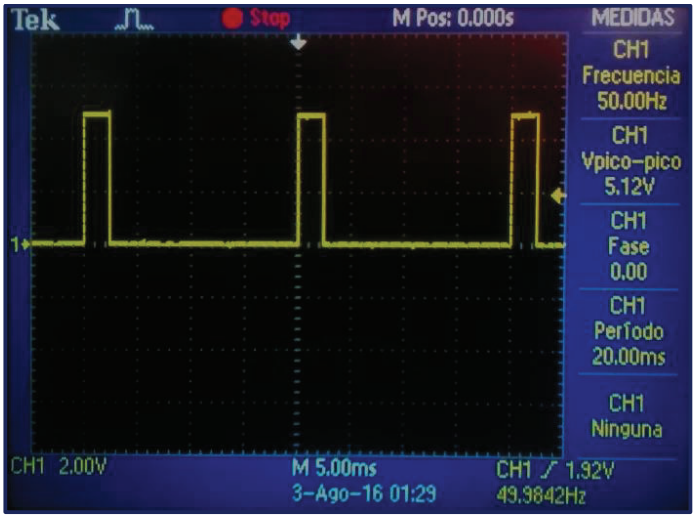



**Figura 4.10.** Pruebas de la señal de control para un ángulo de 180 grados.

## 4.3 PRUEBAS – CONTROL DE POSICIÓN

En esta prueba, se puede establecer el valor de referencia y se observa la gráfica de como la esfera llega al valor indicado. En la interfaz se muestra un recuadro de color azul (24 x 24 cm) que establece el área permitida donde el usuario puede seleccionar la posición de la esfera. Esta restricción se consideró por seguridad, para evitar puntos en los extremos de la plataforma donde la esfera puede caer de la misma. En la Figura 4.11 se observa un cuadrado de color azul, que delimita el área antes mencionada.

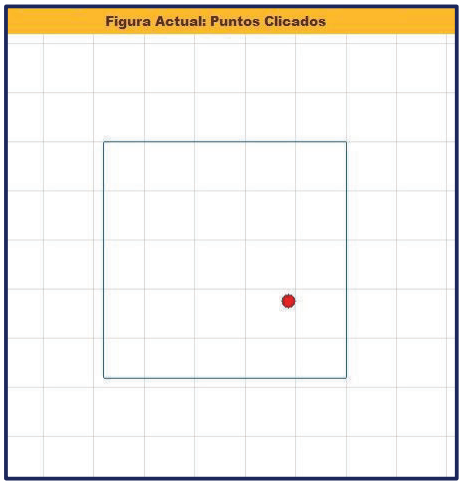



**Figura 4.11.** Control de posición de la esfera

A continuación, en las Figuras 4.12 y 4.13 se muestra la respuesta del sistema ante una entrada de referencia de 29.2 cm para el eje x y 27.2 cm para el eje y.


**Figura 4.12.** Posición de la esfera en el eje X


**Figura 4.13.** Posición de la esfera en el eje Y

De los datos obtenidos en las gráficas anteriores, se obtuvieron los resultados mostrados en la Tabla 4.1, correspondientes a la respuesta del sistema ante la referencia antes mencionada.

**Tabla 4.1.** Análisis en estado estable – control de posición.

| Parámetros | Eje X | Eje Y |
|---|---|---|
| Referencia | 29.2 [cm] | 27.2 [cm] |
| Valor inicial | 16.12 [cm] | 15.82 [cm] |
| Valor de establecimiento | 28.93 [cm] | 27.3 [cm] |
| Tiempo de establecimiento | 1.225 [s] | 1.075 [s] |
| Máximo sobre pico | 0 | 0 |
| Error Absoluto | 0.27 [cm] | 0.1 [cm] |
| Error Relativo | 2.04 % | 0.88 % |

De la prueba anterior, se puede observar que el error absoluto es de 0.27 cm en el eje x y 0.1 cm en el eje y, es decir, respecto a la referencia, una vez que la esfera se detiene, el error no sobrepasa los 3 milímetros, siendo este un buen resultado.

A continuación, en la Figura 4.14 se muestra la señal de control, es decir, los ángulos calculados por el controlador que fueron enviados hacia los servomotores durante el movimiento de la esfera.

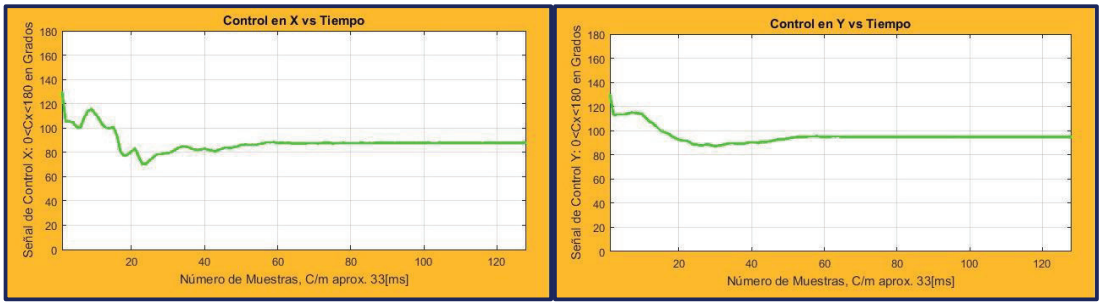



**Figura 4.14.** Señal de control (Ángulos a cargar en los servomotores)

## 4.4 PRUEBAS – SEGUIMIENTO DE CAMINOS

En las figuras siguientes, se tomará en cuenta las siguientes consideraciones para una correcta interpretación de resultados:

En cada una de las gráficas se debe tener en cuenta la diferenciación de colores:

| | |
|---|---|
| **Color Rojo:** | **Referencia de camino** |
| **Color Verde:** | **Salida (Seguimiento de la esfera)** |

Se presentarán cuatro secciones en cada figura sobre los resultados obtenidos. Cada una de ellas se detalla a continuación:

(a) Muestra la referencia deseada. Es decir, el camino o figura que se realiza.

(b) Muestra, de color rojo el camino de referencia, y de color verde el camino real que ha seguido la esfera. Esta gráfica, de la posición X-Y cm, superpone el camino seguido y la referencia, para poder visualizar la efectividad del controlador.

(c) Muestra el seguimiento de la esfera por cada eje. Esta gráfica permite visualizar de mejor manera como cambia la referencia y la señal de realimentación. En el eje X se grafica el número de muestras (≈33ms c/u) y en el eje Y se grafica la posición en centímetros.

(d) Muestra la salida del controlador, es decir el valor del ángulo que se carga en cada instante en los servomotores. En el eje X se grafica el número de muestras (≈33ms c/u) y en el eje Y el ángulo que se carga en el servomotor en grados.

Además, es importante mencionar que en las secciones (c) y (d) de cada figura, se especifica en el eje X el número de muestras tomadas para cada figura. Dada la configuración del programa, una vez que se presiona el botón “Parar” en la interfaz, este guarda un número determinado de muestras correspondientes a las últimas entradas de realimentación de la cámara. En 4 vectores se guardan las coordenadas de referencia y de realimentación de la esfera. Estos vectores contienen la referencia en x, la referencia en y, la realimentación en x y la realimentación en y, con cuyos valores se realizan las gráficas mostradas y el análisis de resultados.

A continuación, en las Figuras de la 4.15 a la 4.20 se mostrará los primeros resultados obtenidos apenas se inicia la realización de una figura. Estas figuras tienen el objetivo de mostrar la evolución del error desde que la esfera parte en reposo en una posición inicial hasta que la misma empieza a superponerse en su camino de referencia. En la sección (b) de estas figuras, se marcará con un círculo de color azul el punto desde el cual inició el movimiento la esfera. Además, estas figuras ponen en evidencia la acción del controlador que lleva a la esfera desde una posición inicial hasta que su movimiento se superpone y sincroniza con el cambio constante de la referencia. Se escogió seis figuras a manera de ejemplo en las que se puede ver de color verde, como evoluciona el movimiento de la esfera sobre la plataforma respecto al camino de referencia.

- ***Círculo – Radio 7 - Evolución del seguimiento de camino***

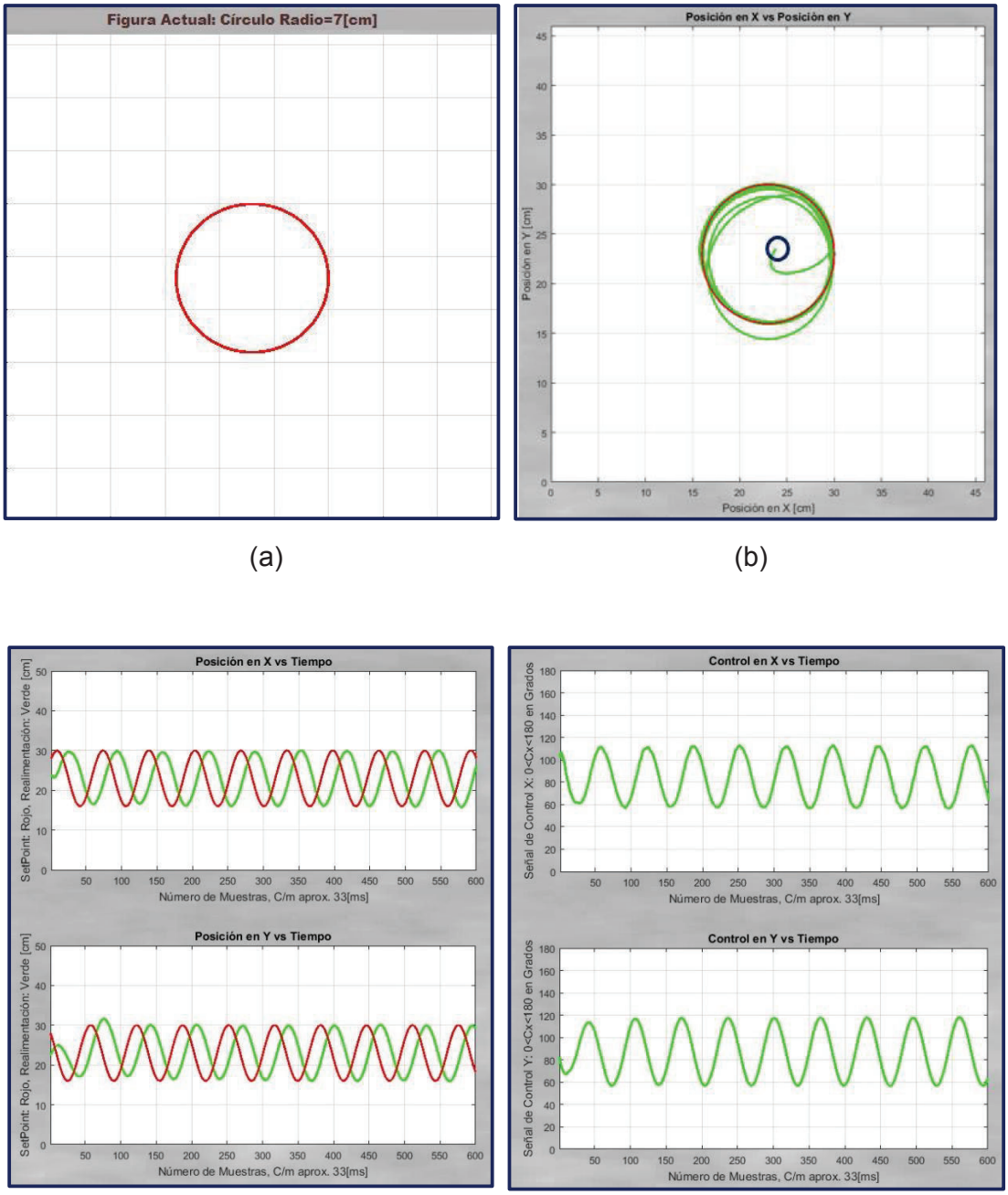
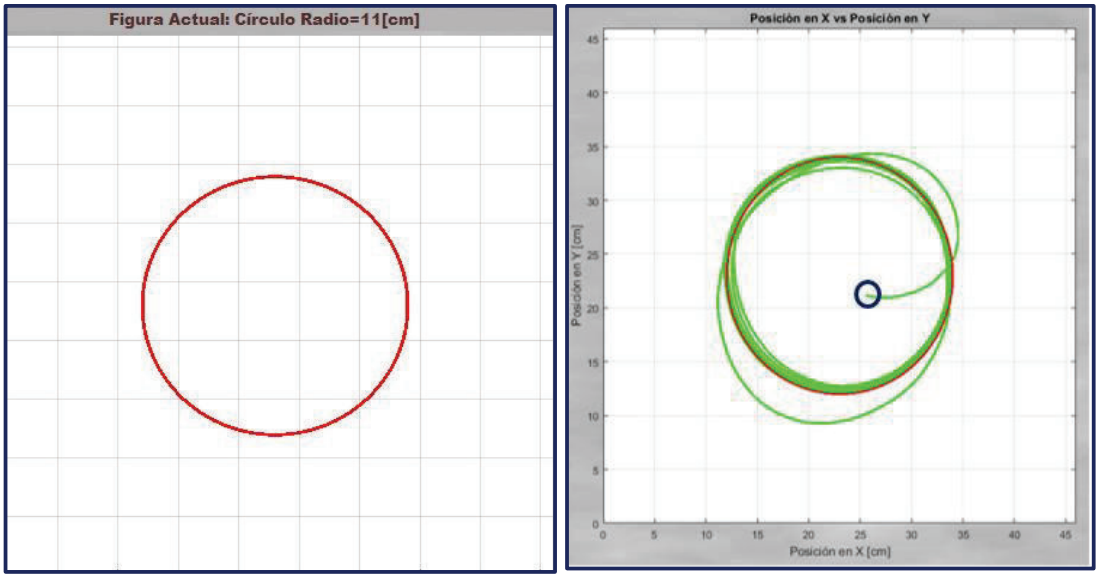
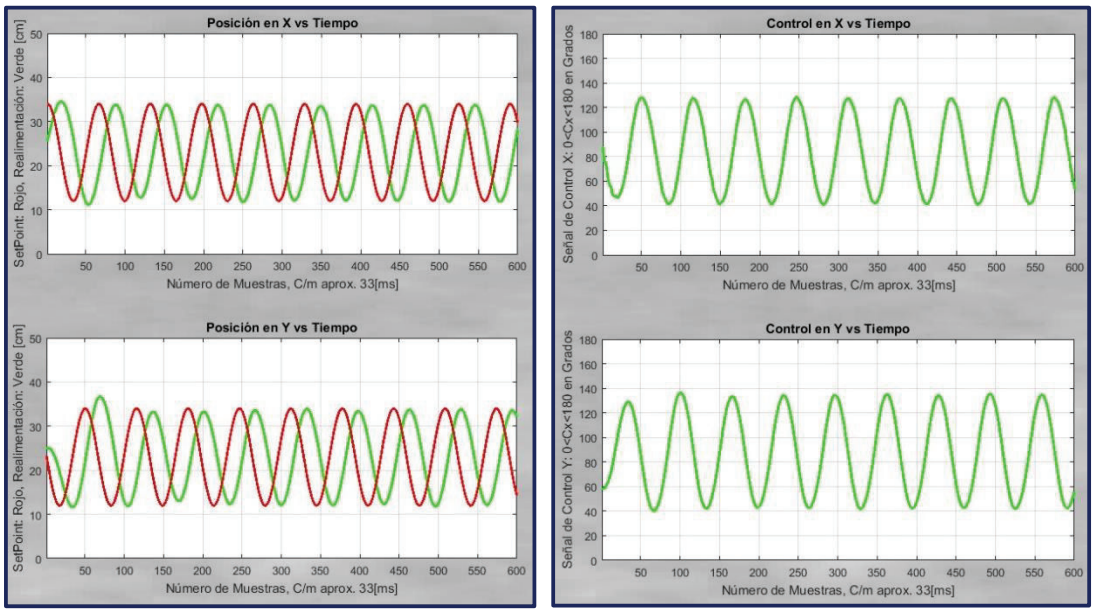



**Figura 4.15.** *Círculo – Radio 7 – Evolución del seguimiento* (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador.

- ***Círculo – Radio 11 - Evolución del seguimiento de camino***




(a) (b)




(c) (d)

**Figura 4.16.** *Círculo – Radio 11 – Evolución del seguimiento* (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador.

- ***Elipse - Evolución del seguimiento de camino***

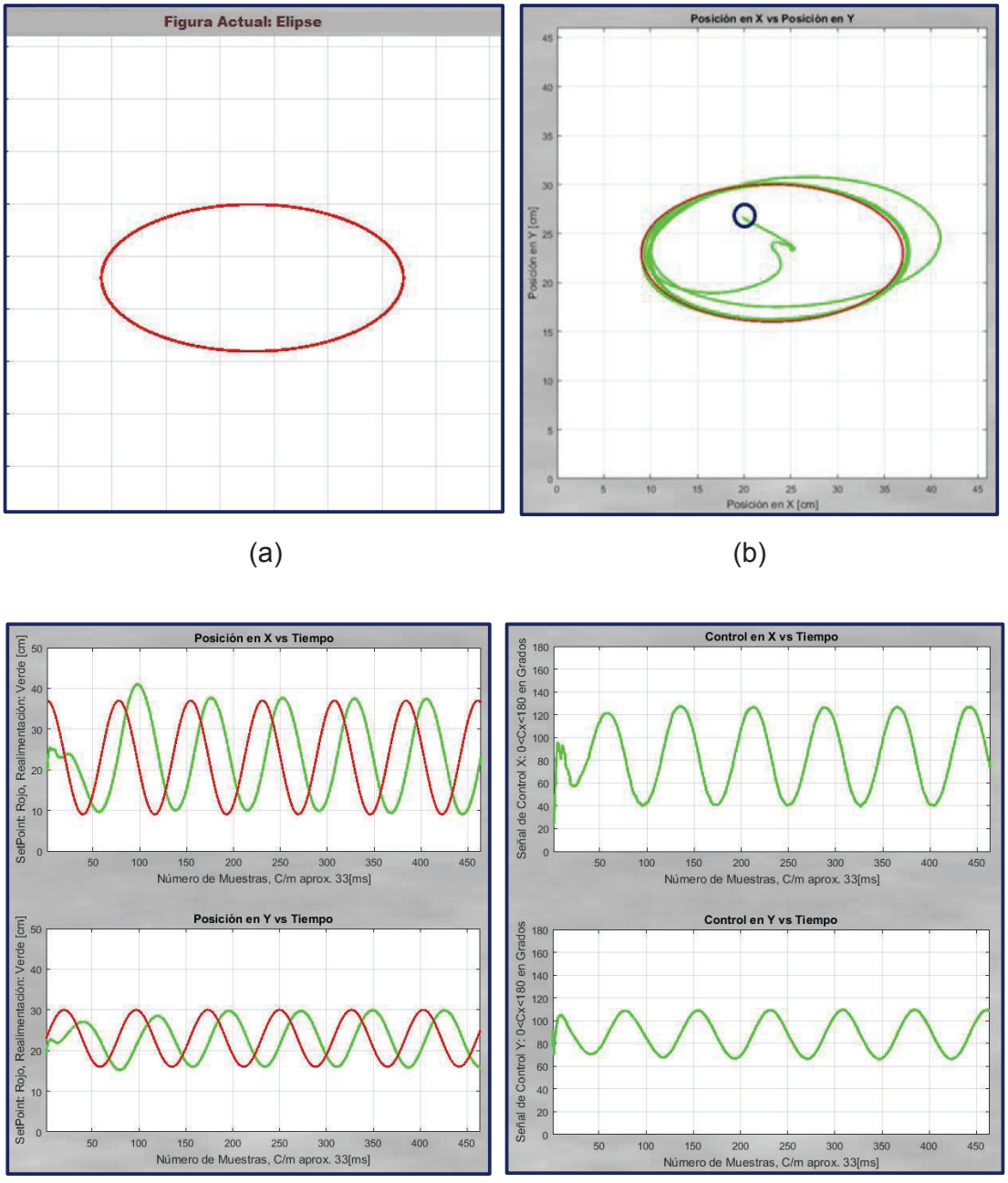
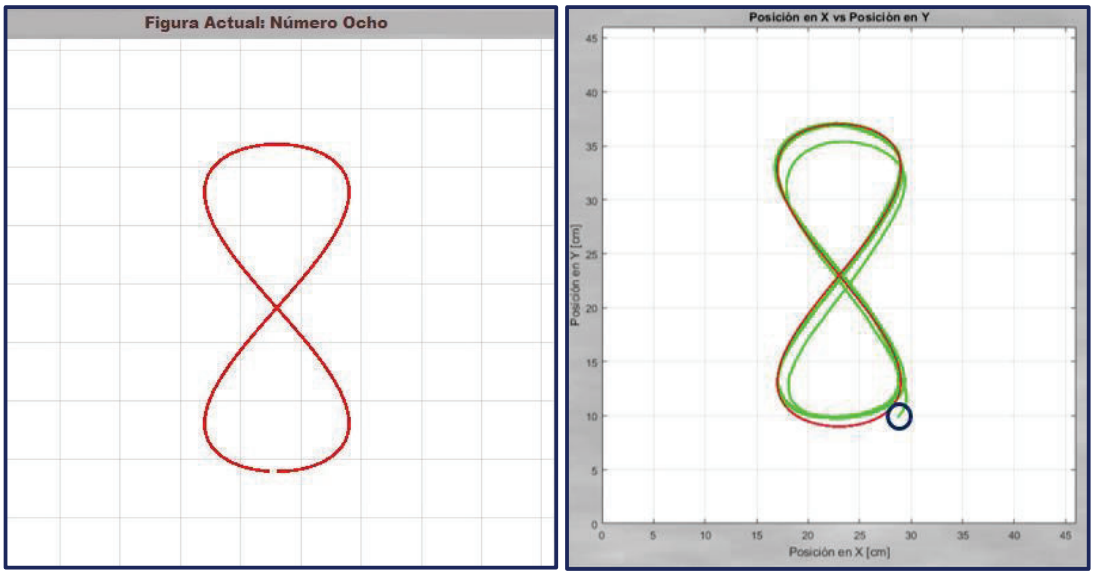
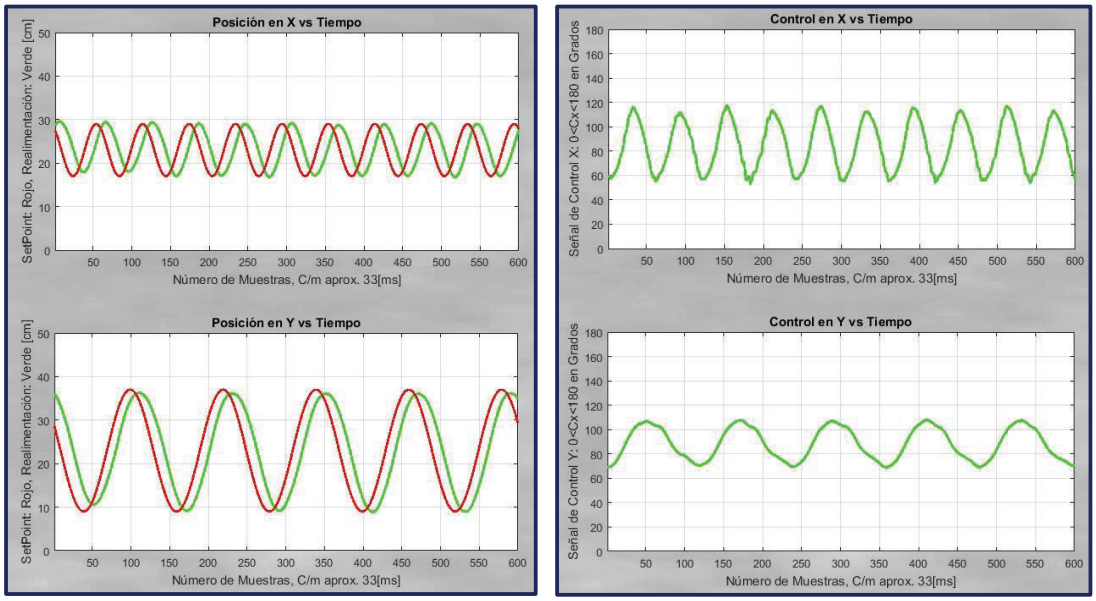



**Figura 4.17.** Elipse – *Evolución del seguimiento* (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador.

- ***Figura 8 - Evolución del seguimiento de camino***



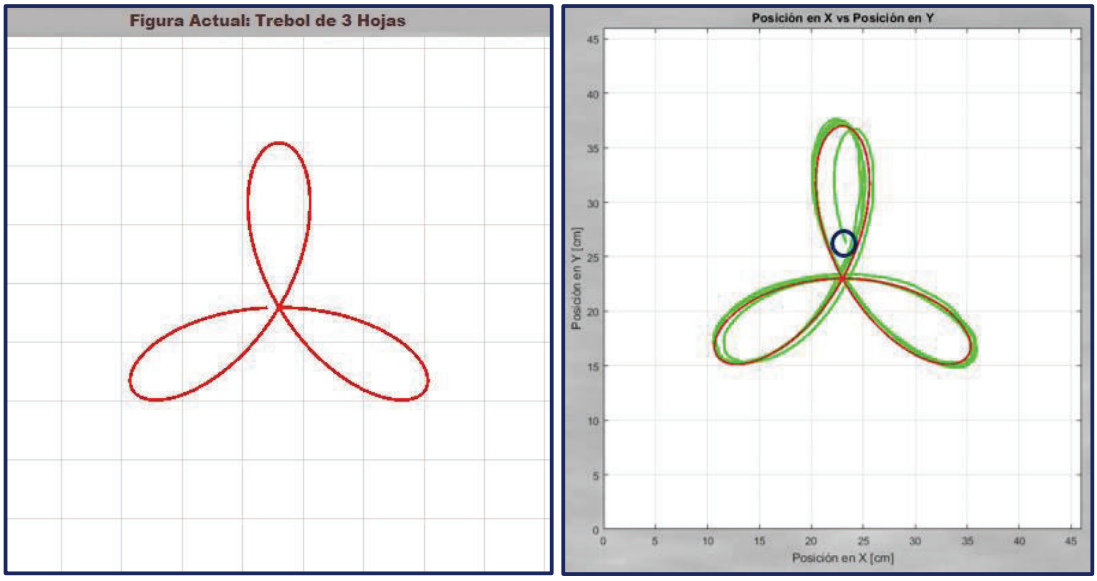

(a) (b)



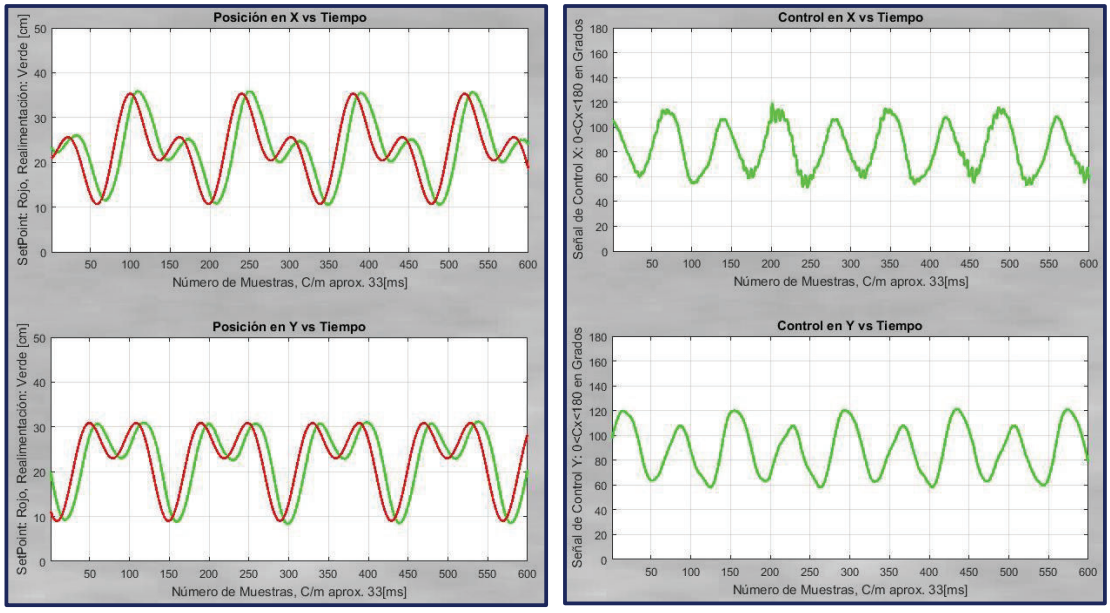

(c) (d)

**Figura 4.18.** Figura 8 – *Evolución del seguimiento* (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador.

- ***Trébol 3 - Evolución del seguimiento de camino***



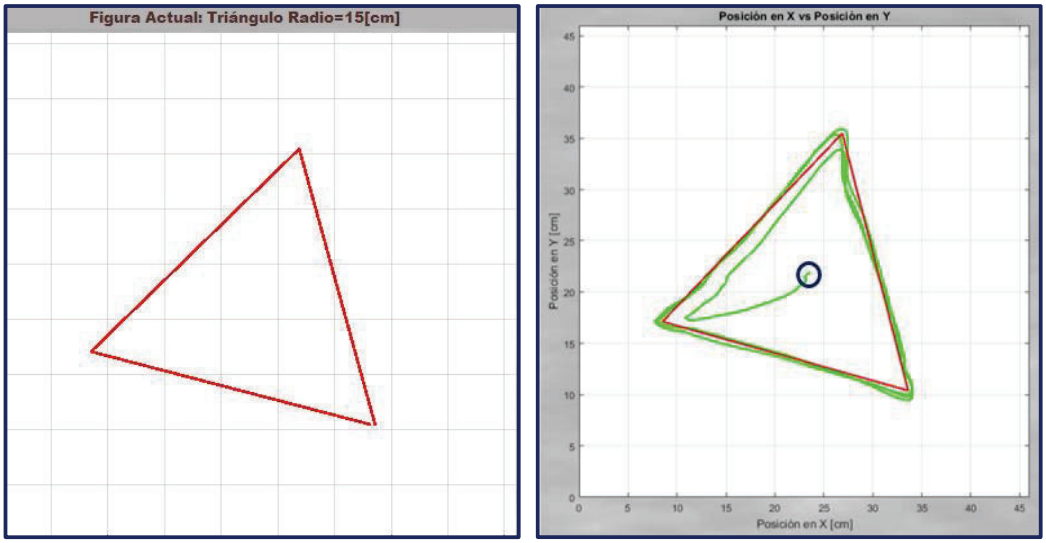

(a) (b)



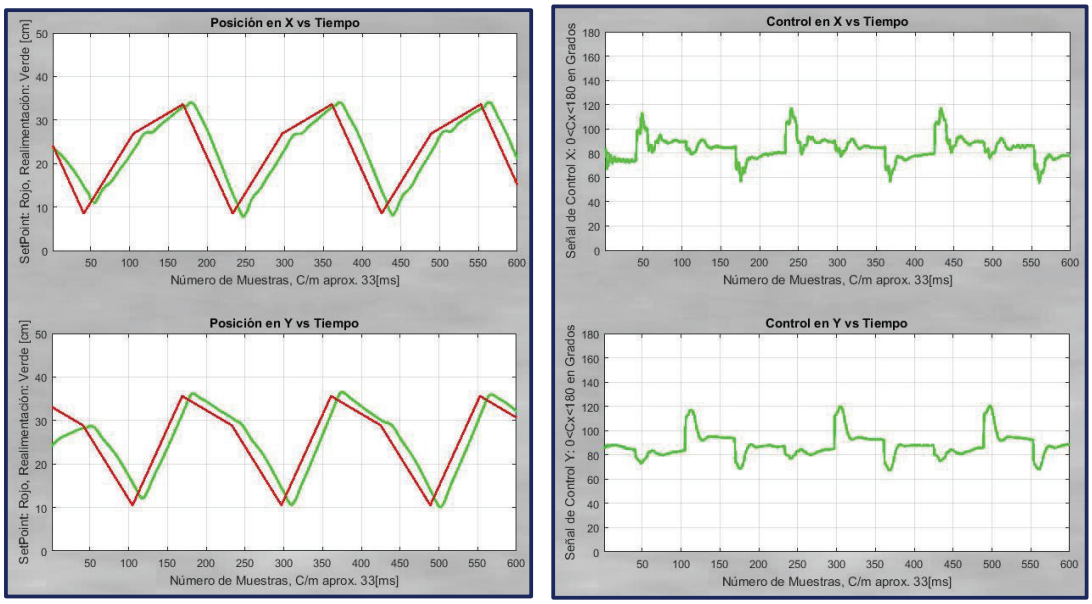

(c) (d)

**Figura 4.19.** Trébol 3 – *Evolución del seguimiento* (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador.

- ***Triángulo – Radio 15 - Evolución del seguimiento de camino***




(a) (b)




(c) (d)

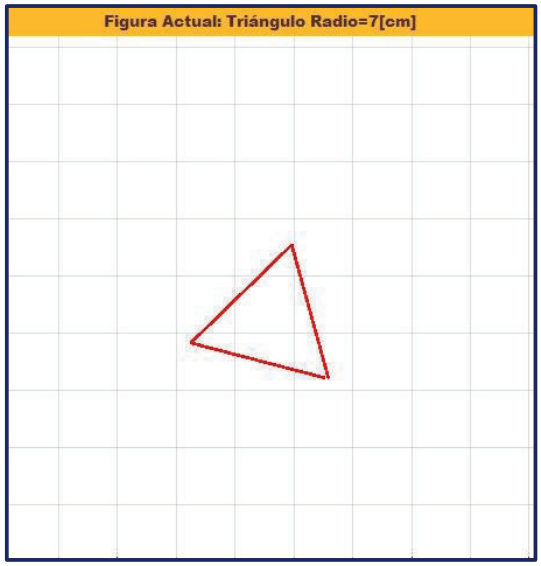
**Figura 4.20.** *Triángulo – Radio 15 – Evolución del seguimiento* (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador.

Una vez que ha transcurrido el tiempo suficiente, la esfera describe la figura lo más cercana posible a su referencia y de no haber ninguna perturbación, esta se mantiene en su camino.

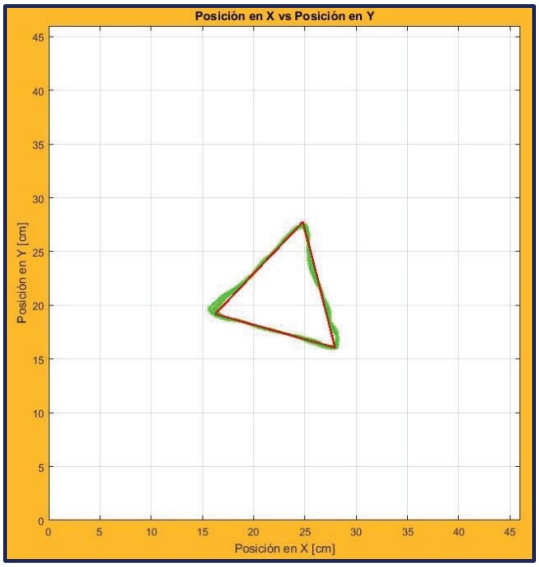
Es en este estado estable en el que vamos a realizar el análisis del seguimiento de camino, pues de esta manera podemos comparar la figura descrita con la figura de referencia.

A continuación, en las Figuras, de la 4.21 hasta la 4.33, se muestran los resultados del seguimiento de caminos en estado estable, por lo que no se observará el punto de partida de la esfera, sino las últimas muestras a partir de su superposición más óptima respecto a la referencia. Además, para poder medir el error del seguimiento de caminos, se analiza el error cuadrático medio tanto para el eje x como para el eje y.

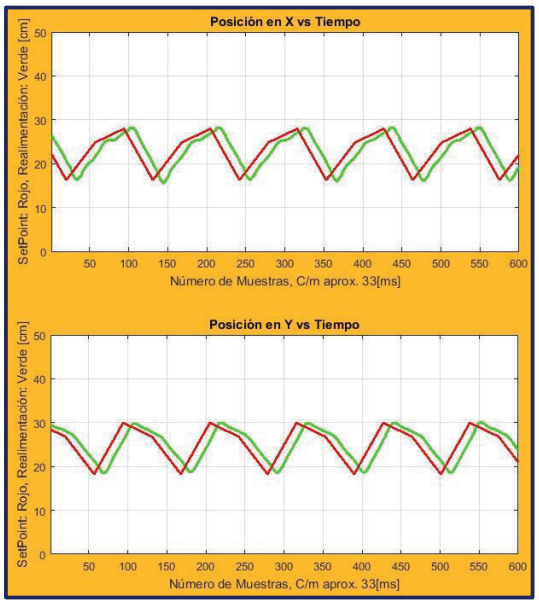
- ***Triangulo – Radio 7***




(a)




(b)

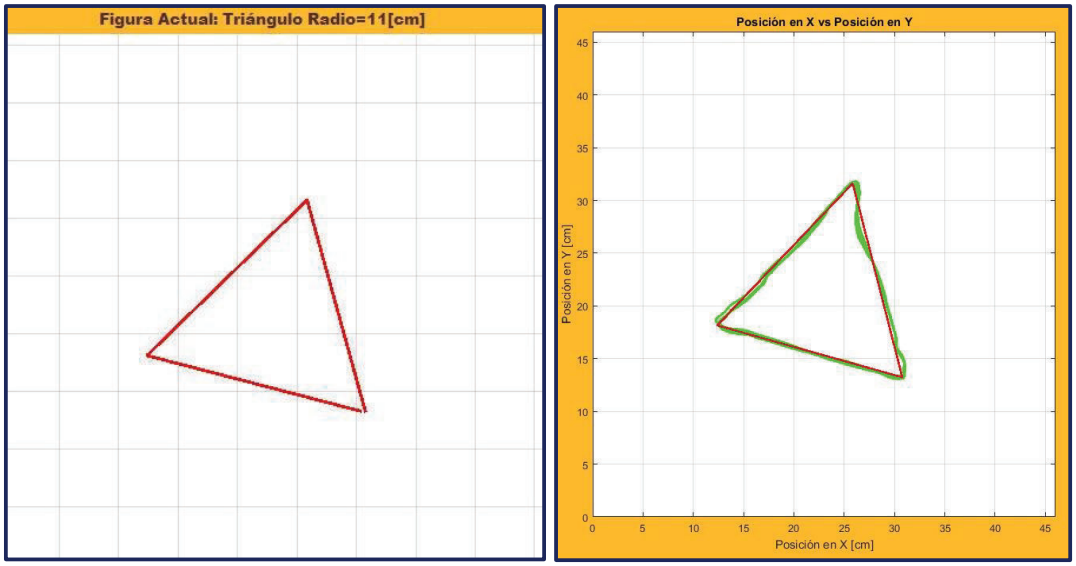



(c)

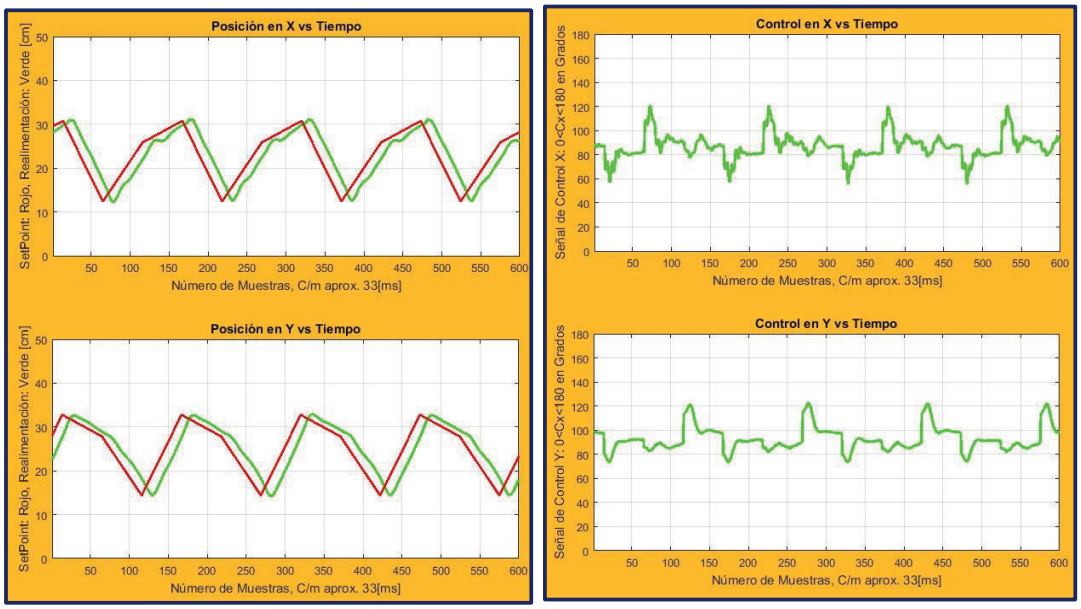

(d)

**Figura 4.21.** *Triángulo – Radio 7* (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador.

- ***Triangulo – Radio 11***




(b) (b)




(c) (d)

**Figura 4.22.** *Triángulo – Radio 11* (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador.

- ***Triangulo – Radio 15***

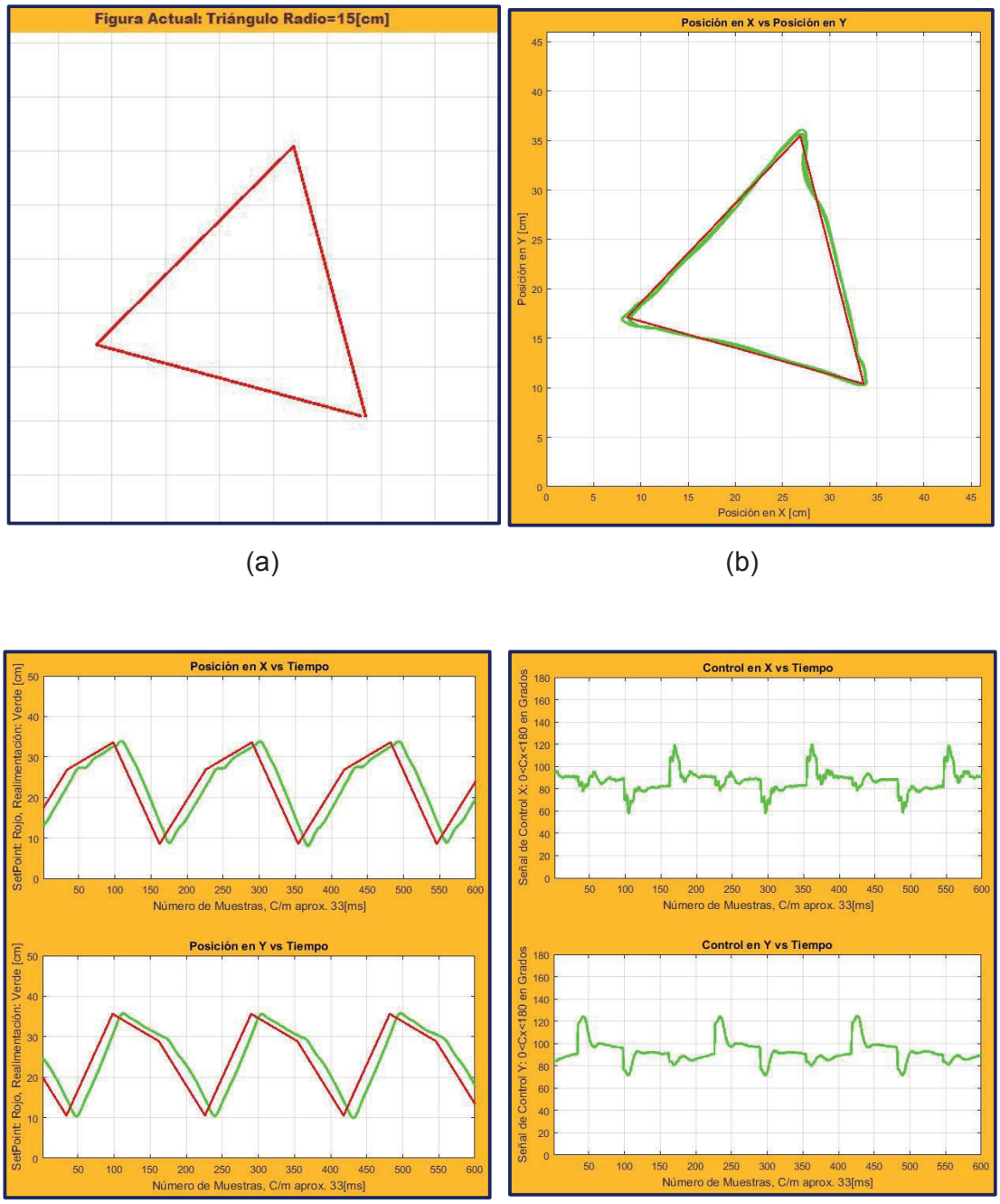
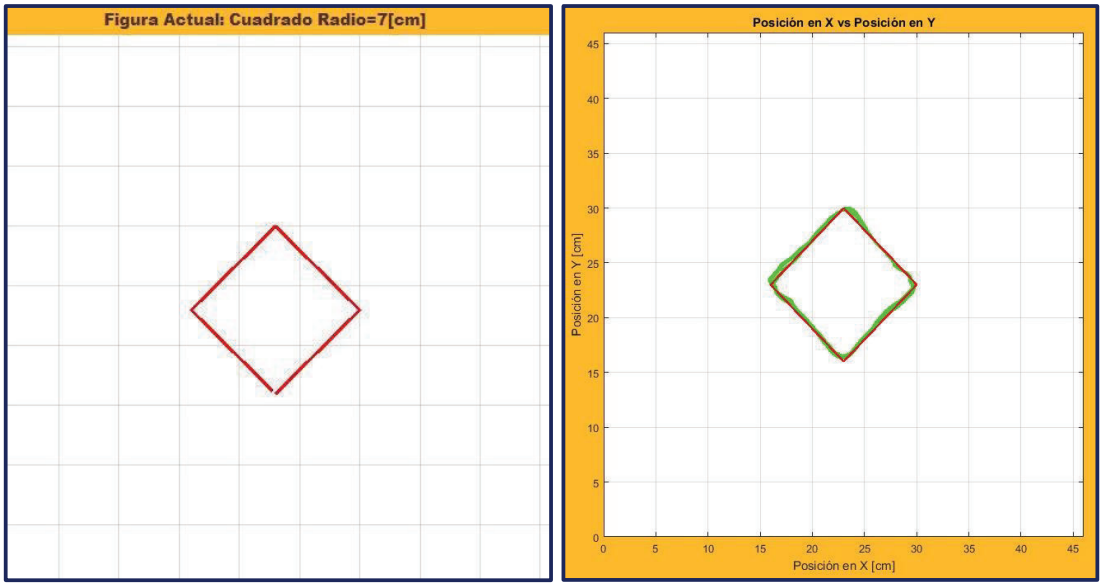
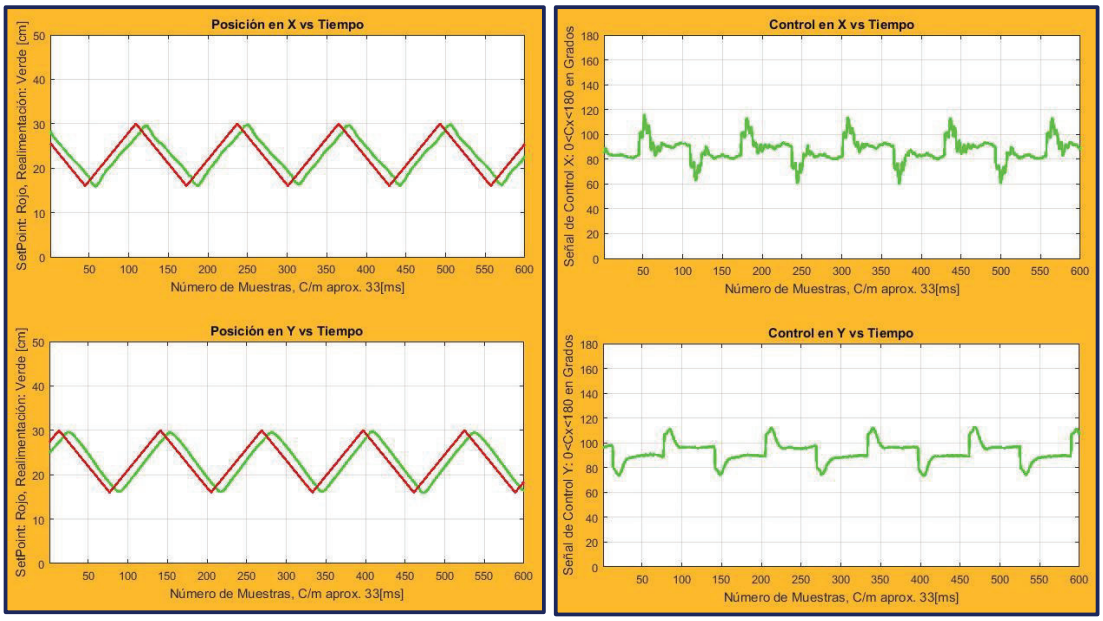



**Figura 4.23.** *Triángulo – Radio 15* (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador.

- ***Cuadrado – Radio 7***



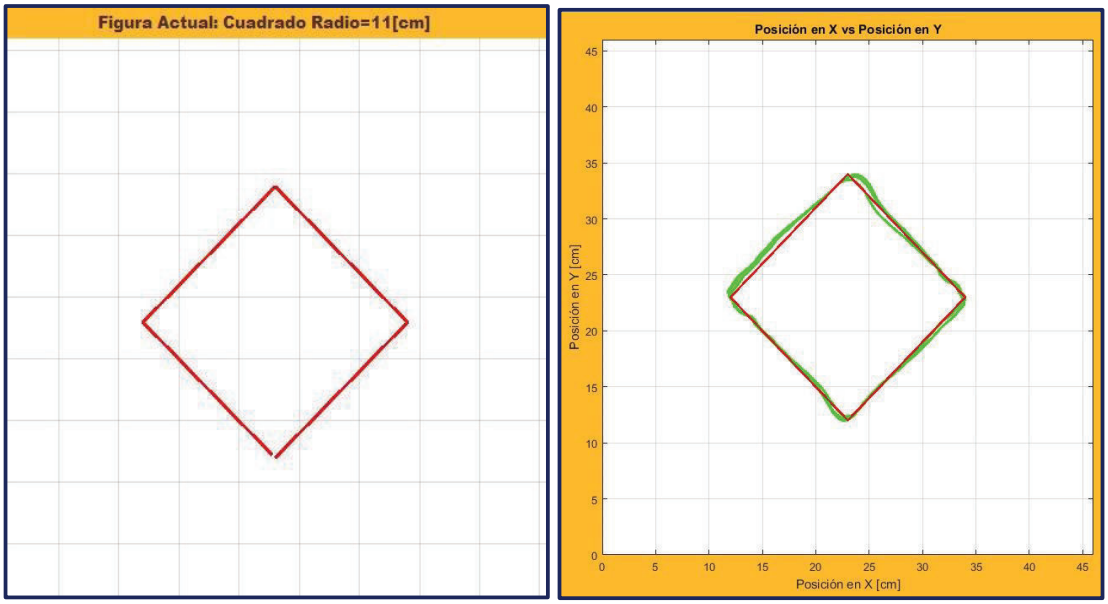

(a) (b)



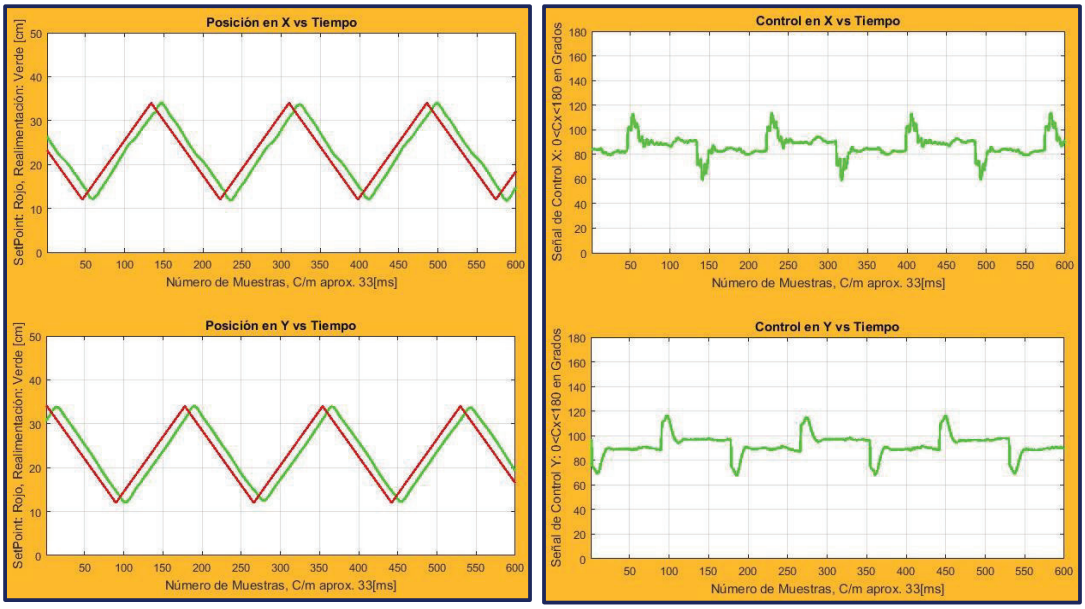

(c) (d)

**Figura 4.24.** *Cuadrado – Radio 7* (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador.

- ***Cuadrado – Radio 11***



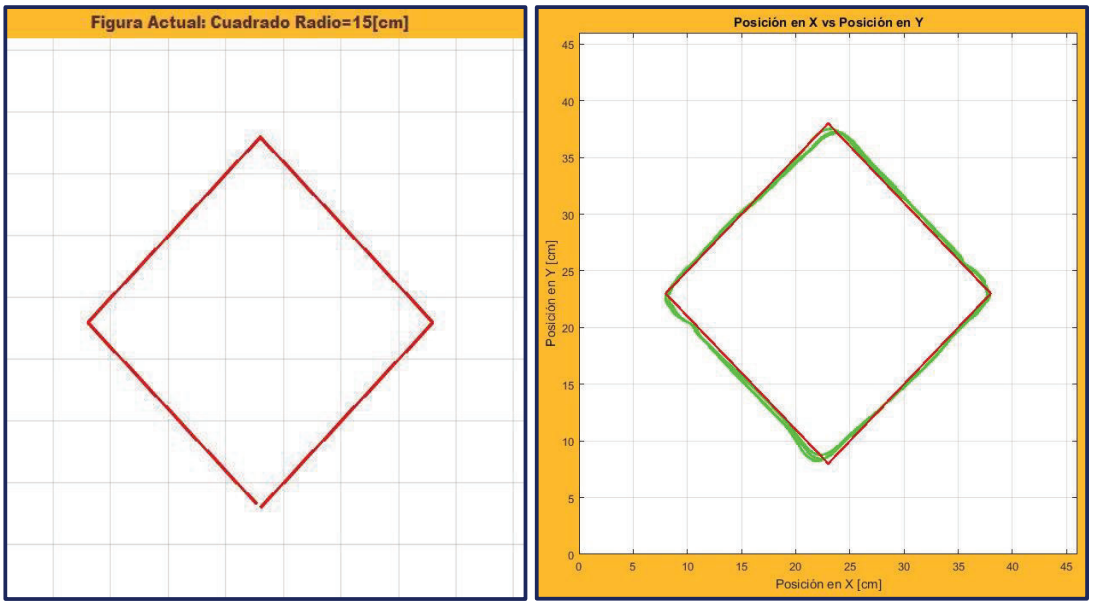

(a) (b)



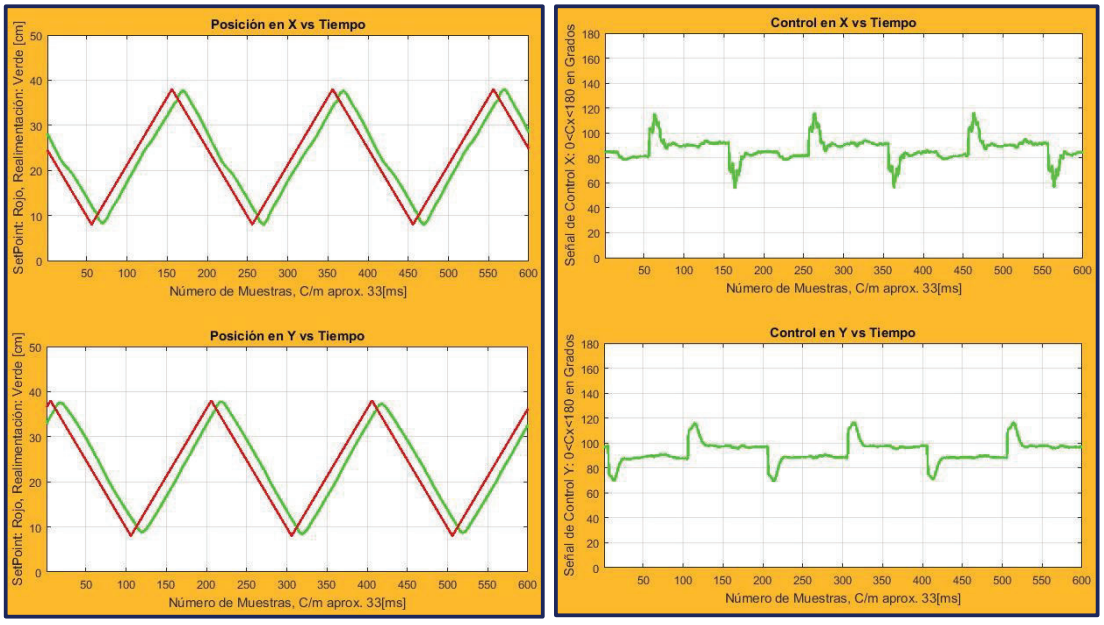

(c) (d)

**Figura 4.25.** *Cuadrado – Radio 11* (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador.

- ***Cuadrado – Radio 15***



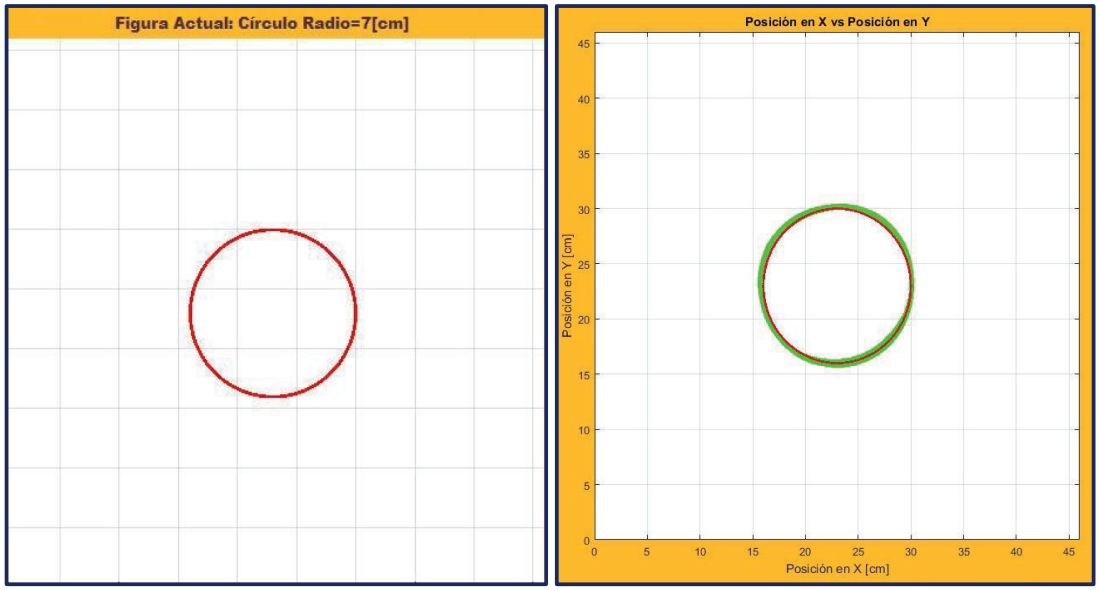

(a) (b)



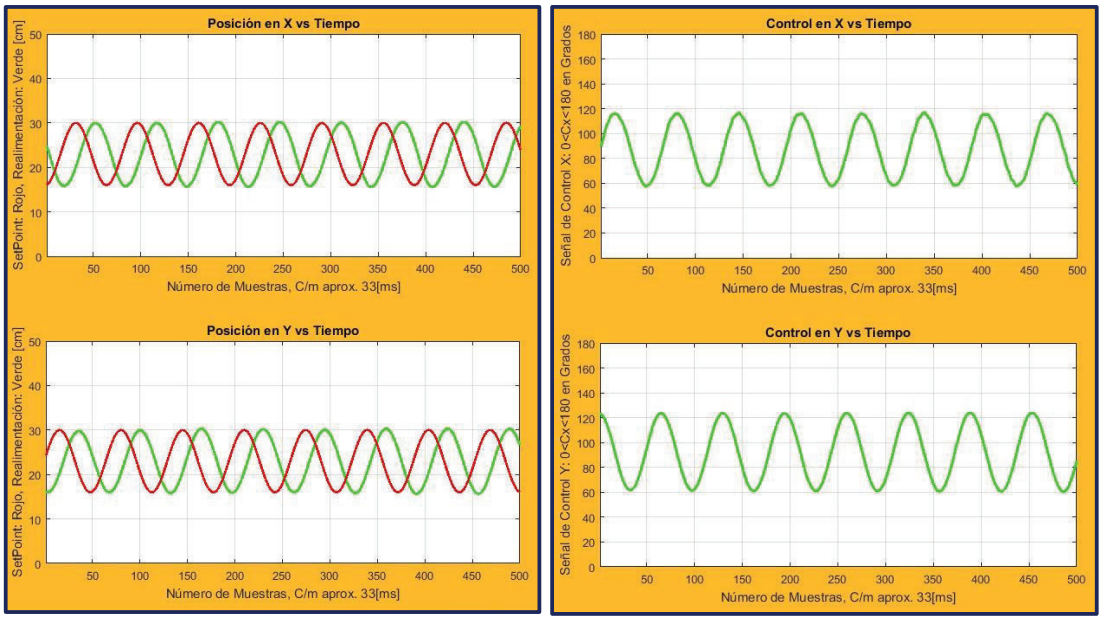

(c) (d)

**Figura 4.26.** *Cuadrado – Radio 15* (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador.

- ***Círculo – Radio 7***



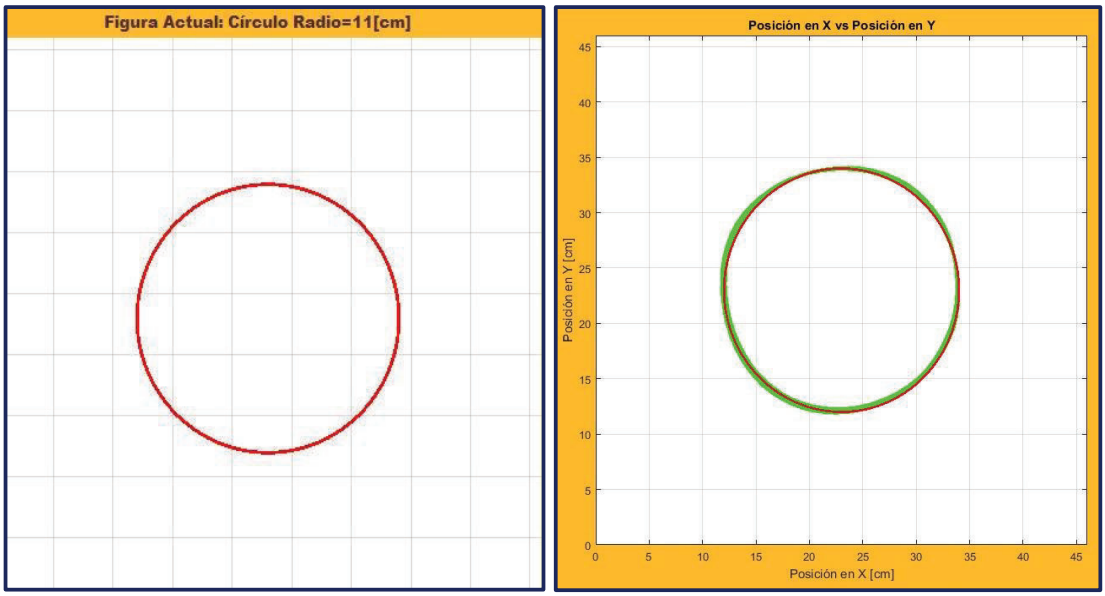

(a) (b)



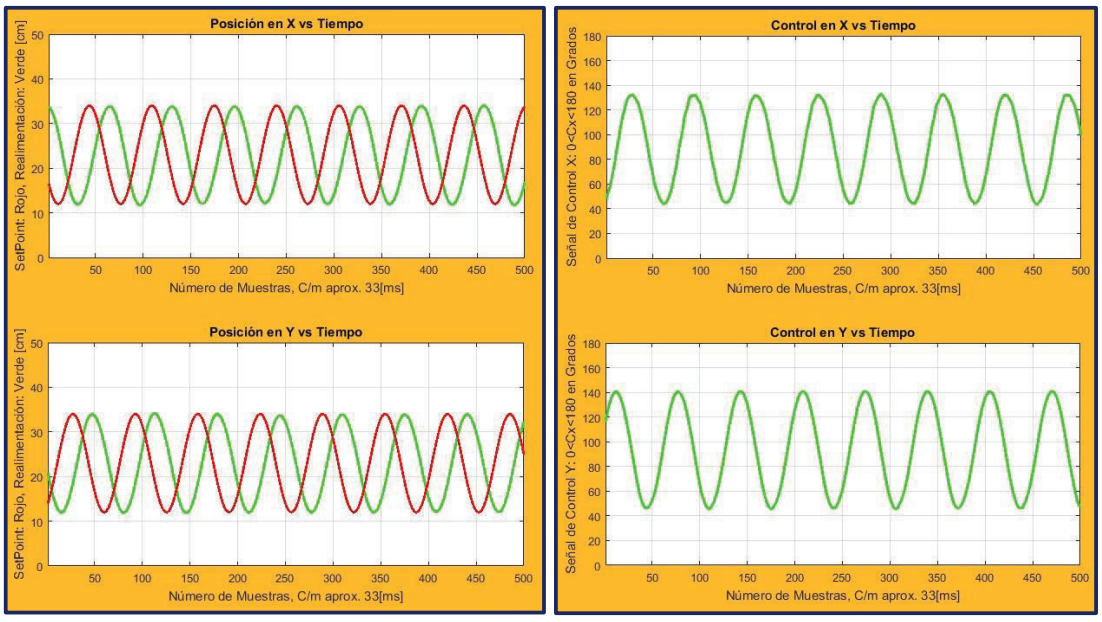

(c) (d)

**Figura 4.27.** *Círculo – Radio 7* (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador.

- ***Círculo – Radio 11***



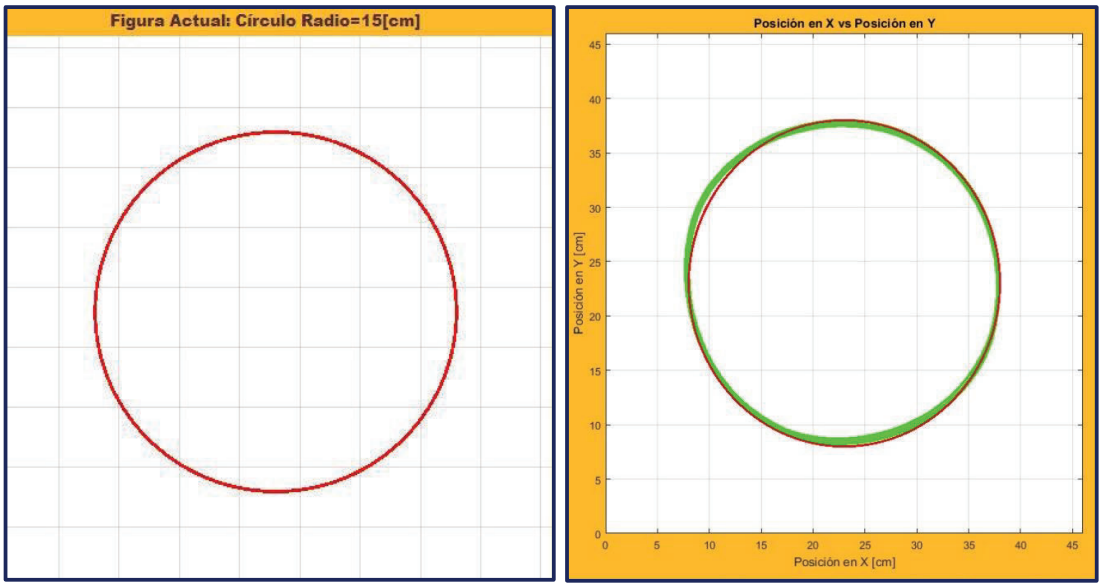

(a) (b)



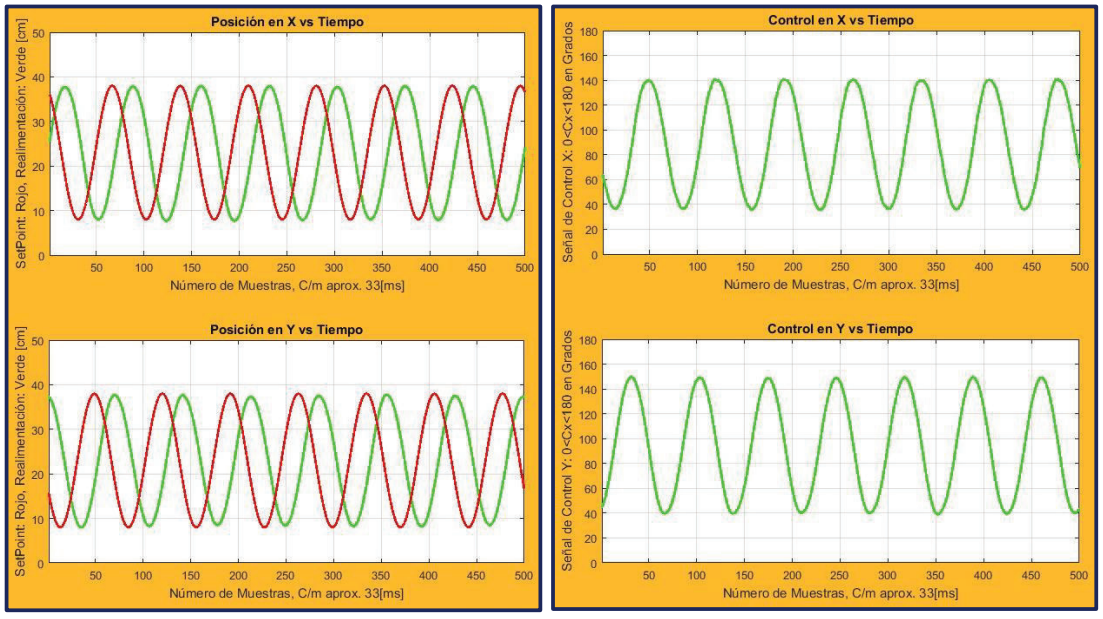

(c) (d)

**Figura 4.28.** *Círculo – Radio 11* (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador.

- ***Círculo – Radio 15***



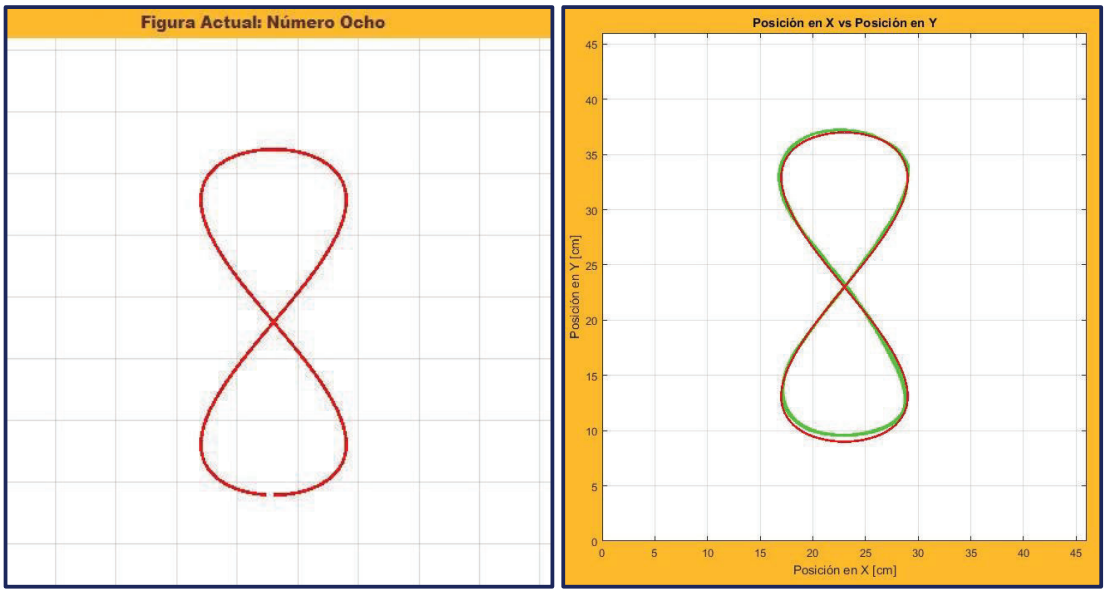

(a) (b)



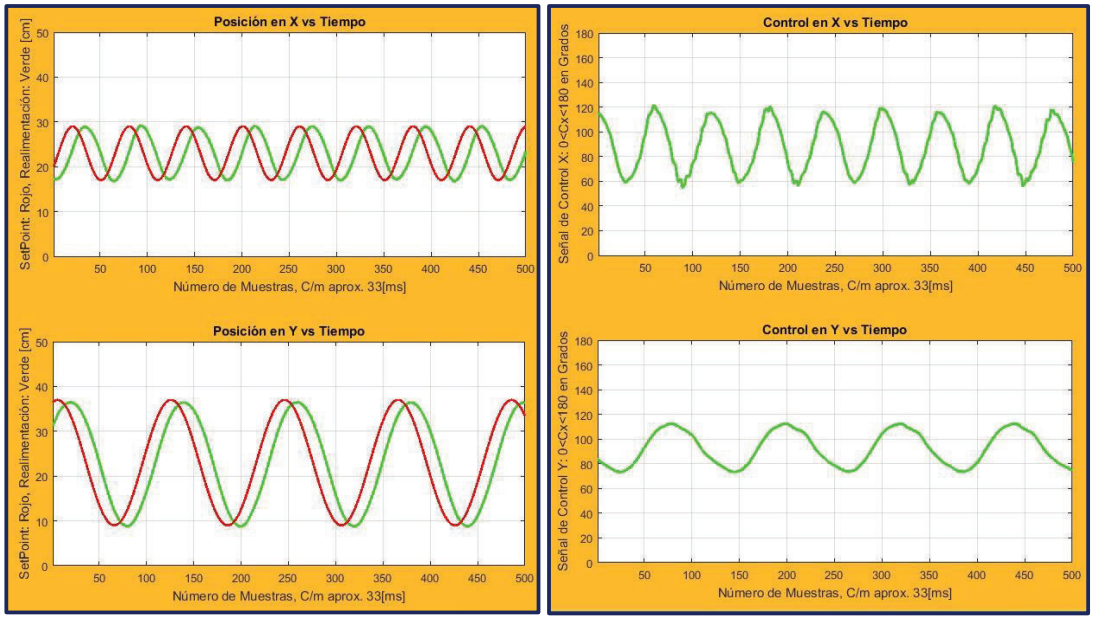

(c) (d)

**Figura 4.29.** *Círculo – Radio 15* (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador.

- ***Figura 8***




(a) (b)




(c) (d)

**Figura 4.30.** *Figura 8* (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador.

- ***Trébol de 3 foliolos***

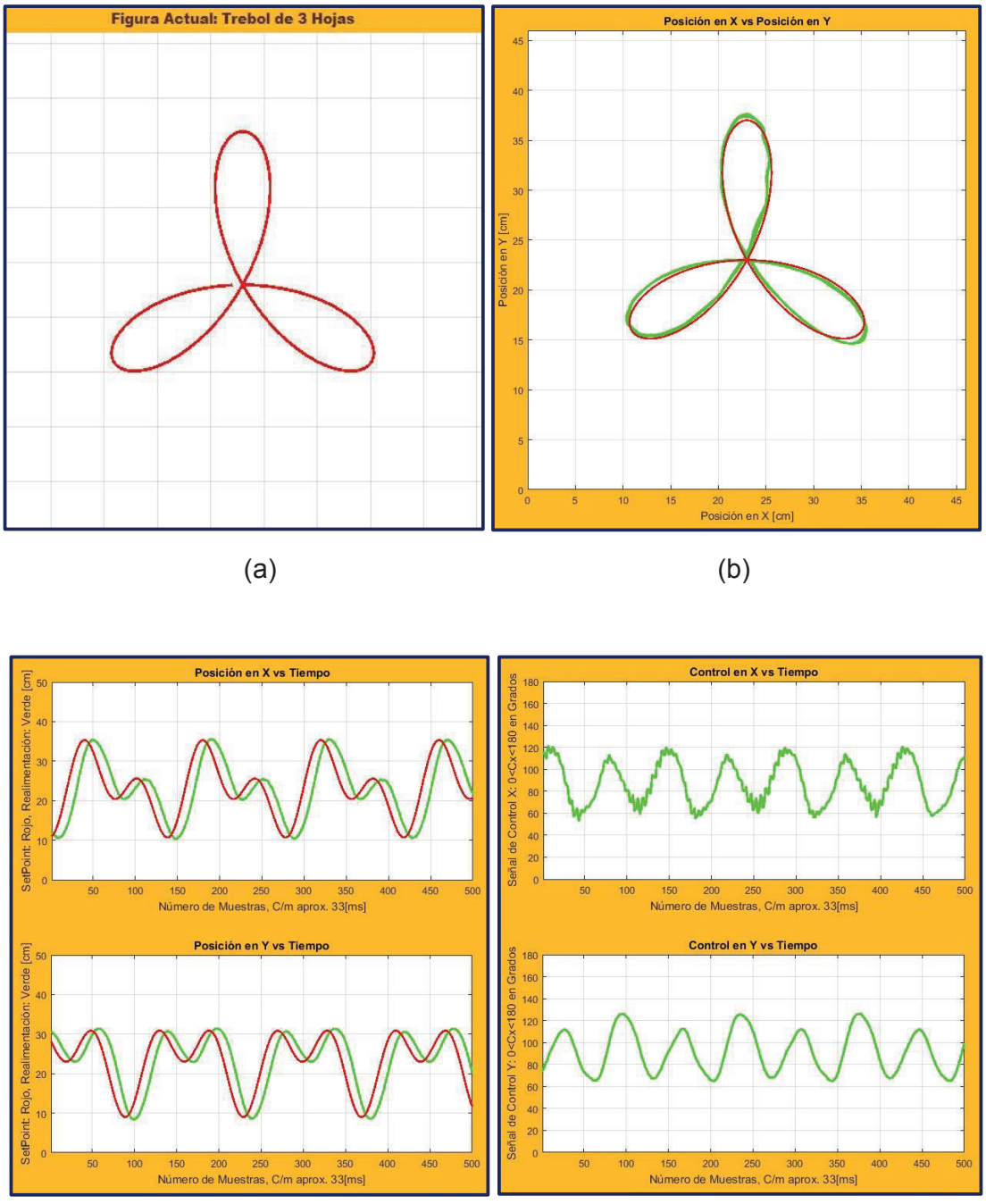
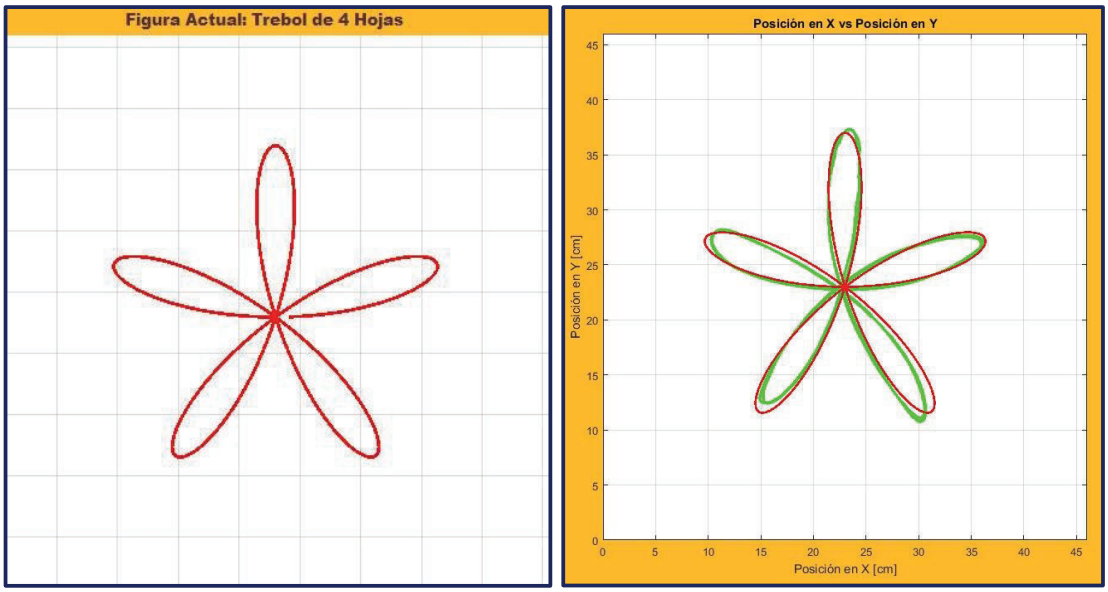
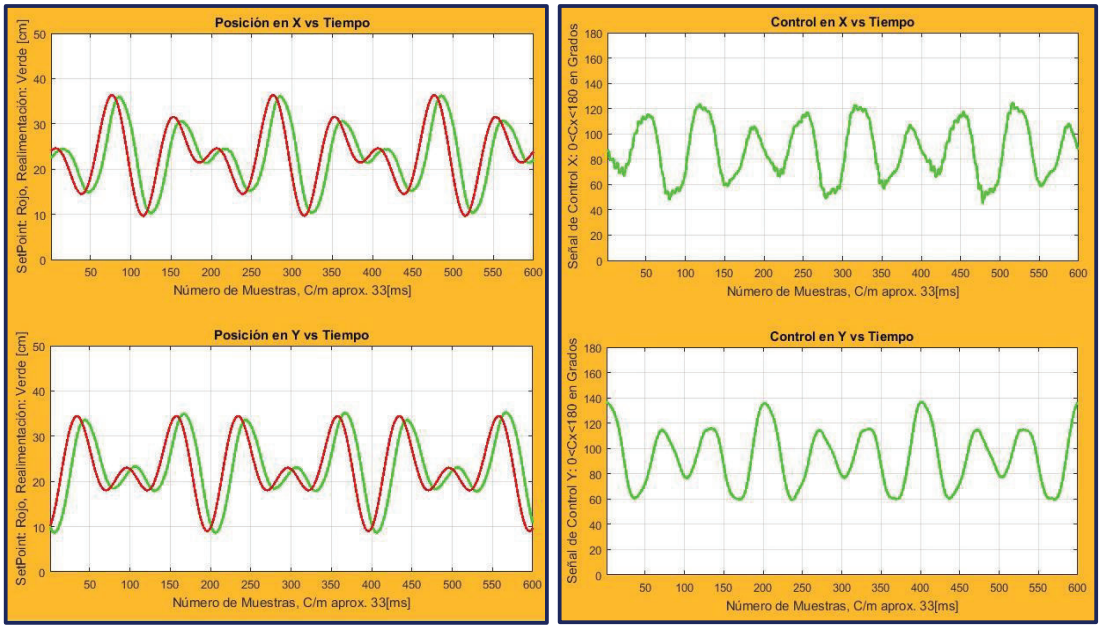



**Figura 4.31.** *Trébol de 3 hojas* (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador.

- ***Trébol de 5 foliolos***




(a) (b)




(c) (d)

**Figura 4.32.** *Trébol de 5 hojas* (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador.

- ***Elipse***

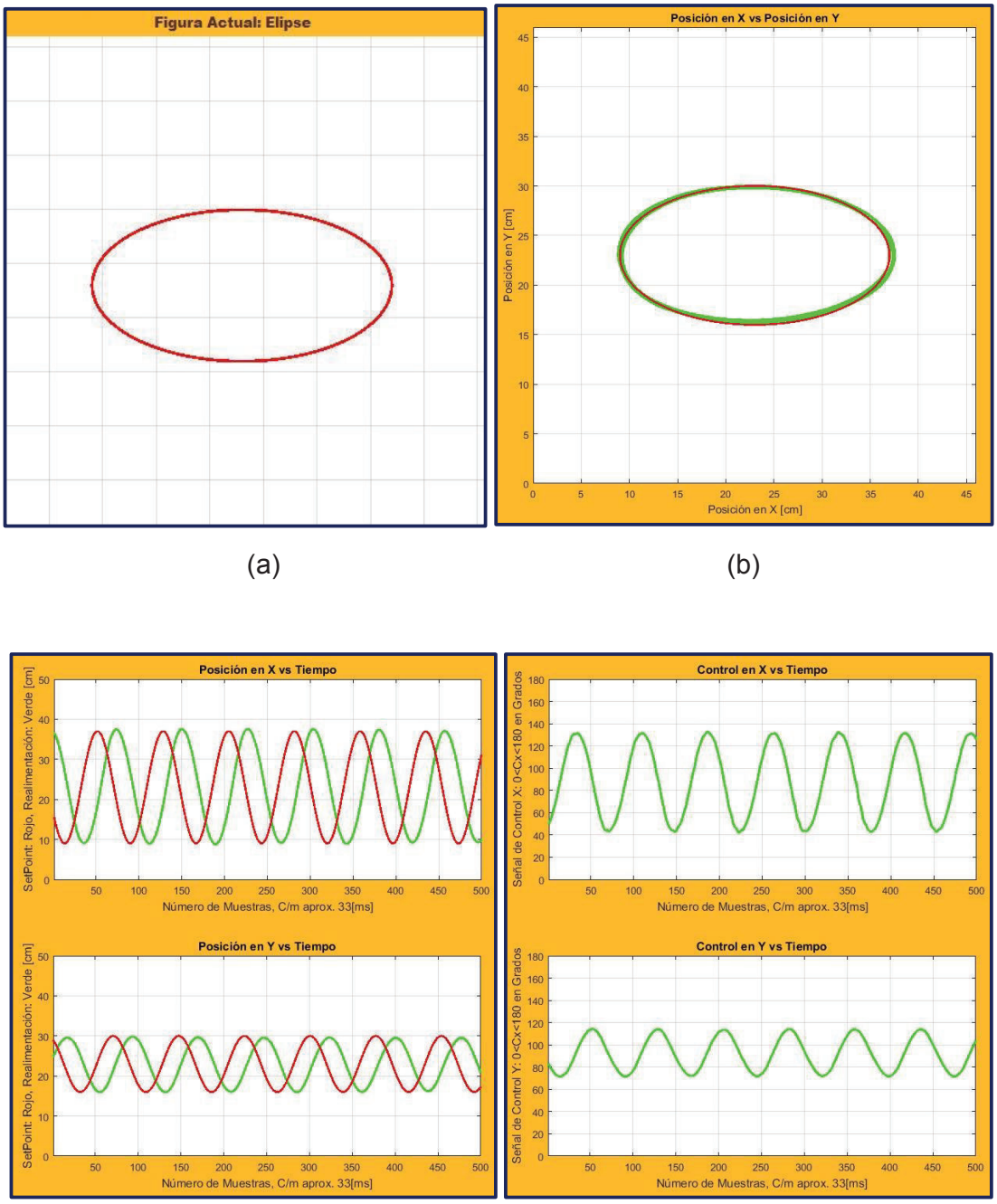



**Figura 4.33.** *Elipse* (a) Referencia (b) Seguimiento en el plano XY (c) Seguimiento por cada eje (d) Salida del controlador.

El objetivo del análisis de los resultados de seguimiento de caminos está enfocado en determinar la similitud de la forma de la figura obtenida respecto a la referencia. Esto se considera debido a la gran cantidad de puntos que conforman cada camino, así como también de la rapidez de cambio de la referencia, por lo cual no se podría realizar un análisis punto a punto como en el caso del control de posición.

Por lo tanto, para analizar los resultados se ha escogido el error cuadrático medio como parámetro de comparación. La fórmula de cálculo es la siguiente:

$$ECM = \frac{1}{n}\sum_{i=1}^{n}(Referencia_i - Realimentación_i)^2 \tag{4.1}$$

En la Tabla 4.2 se muestra el error cuadrático medio correspondiente a cada figura:

**Tabla 4.2.** Error cuadrático medio de cada figura.

| Figura | | Error Cuadrático Medio (ECM) [cm] | |
|---|---|---|---|
| | | Eje X | Eje Y |
| Triángulo | Radio 7 | 0.11 | 0.09 |
| | Radio 11 | 0.14 | 0.18 |
| | Radio 15 | 0.22 | 0.19 |
| | | | |
| Cuadrado | Radio 7 | 0.18 | 0.15 |
| | Radio 11 | 0.32 | 0.20 |
| | Radio 15 | 0.20 | 0.44 |
| | | | |
| Círculo | Radio 7 | 0.18 | 0.27 |
| | Radio 11 | 0.04 | 0.04 |
| | Radio 15 | 0.13 | 0.08 |
| | | | |
| Figura 8 | | 0.16 | 0.20 |
| | | | |
| Trébol de 3 hojas | | 0.16 | 0.22 |
| | | | |
| Trébol de 5 hojas | | 0.82 | 1.06 |
| | | | |
| Elipse | | 0.17 | 0.09 |

El error cuadrático medio está medido en centímetros y nos dice que tan similar es el camino descrito por la esfera, respecto a la referencia.

Para figuras cuya referencia tiene componentes sinusoidales, como el caso de círculos, la elipse, la figura 8, entre otras, se obtienen mejores resultados y por tanto menor error, esto debido a que la referencia tiene un cambio continuo y no brusco.

Por otro lado, en figuras como triángulos o cuadrados, existe una mayor dificultad para el seguimiento del camino, pues las esquinas representan un cambio brusco en la señal de referencia, ya que tiene que cambiar el sentido de movimiento de la esfera rápidamente.

Sin embargo, de forma general, se observa que, con excepción del trébol de 5 foliolos, el ECM no sobrepasa los 0.32 cm para el eje x y los 0.44 cm para el eje y.

La figura más complicada, tanto por su referencia (densidad de puntos) como por su complejidad de forma, es el trébol de 5 hojas. Esta figura representó una prueba de las limitaciones que se pueden obtener con un controlador PID. Los foliolos (pétalos) de la figura representan cambios muy pequeños en el controlador, por lo que se requiere de mayor precisión en el control.

En figuras con esquinas o que requieren cambios bruscos de la dirección de la esfera, se observa en la Tabla 3.1 que los valores de la constante derivativa son más altos que en las otras figuras, esto se debe a que para poder completar una de las esquinas de un cuadrado por ejemplo, se tiene que anticipar el movimiento, por lo que la constante tiene valores más altos, mismos que permitieron obtener mejores resultados. Esta característica se observa de igual manera en el trébol de 5 foliolos, pues para conseguir que la esfera se frene y pueda realizar de mejor manera la forma de los pétalos, se incrementó el valor de la constante derivativa. Un valor alto en la constate derivativa si bien ayuda a realizar de mejor manera una figura complicada, puede llegar a producir oscilaciones y sacar al sistema de su estabilidad haciendo más difícil su control.

# CAPÍTULO 5.
# CONCLUSIONES Y RECOMENDACIONES

## 5.1 CONCLUSIONES

- Se implementó la plataforma para estabilizar la esfera de la forma descrita en el trabajo. El sistema mecánico se construyó con la finalidad de que su comportamiento sea controlable. La plataforma y todos sus componentes mecánicos produjeron resultados satisfactorios en el control de la esfera, permitiendo a la misma ser capaz de posicionarse en la plataforma, así como de seguir caminos establecidos.
- En el control de posición de la esfera en un punto dado, se obtuvieron errores de posición en estado estable no que superan el 3% y tiempos de establecimiento de máximo 1.4 segundos. En el seguimiento de caminos, el error cuadrático medio por cada eje no supera el valor de 4 milímetros. Por esta razón se puede concluir que los resultados obtenidos en el presente proyecto son satisfactorios.
- Los mejores resultados se obtuvieron para figuras cuya señal de referencia es del tipo sinusoidal, como son los círculos, la figura 8 y la elipse. Esto se debe a que el seguimiento de una señal sinusoidal significa para los servomotores un movimiento amplio y continuo, además que no hay cambios bruscos en la seña de control.
- Las técnicas de visión artificial para la detección de objetos, y para este caso la determinación de las coordenadas de la posición de la esfera en la plataforma, son más eficientes cuando se usa un algoritmo enfocado en extracción de objetos por colores que por forma, siempre y cuando la diferenciación entre los colores de la plataforma y la esfera sea notoria.
- La atenuación de las oscilaciones de la señal de control gracias al filtro de Kalman se redujeron en más del 50%, permitiendo al controlador actuar de mejor manera. Los resultados del filtro de Kalman son satisfactorios ya que atenúan el ruido y además no retrasan a la señal de realimentación respecto al movimiento real de la esfera.

- El controlador PID ha sido escogido para el sistema de control porque presenta grandes ventajas como su alta robustez y relativa facilidad en el diseño e implementación, además presenta una gran versatilidad en el momento de la calibración manual de sus constantes. Esto es posible ya que la señal de control generada por el PID es función básicamente de la señal de error entre la entrada y la salida, es decir, no considera los estados dinámicos de la planta que produjeron tal salida, esta ventaja ha sido la que más se ha explotado de tal manera de obtener resultados óptimos.

## 5.2 RECOMENDACIONES

- Como el sistema Bola-Plataforma es un sistema de control que por sus características no lineales es ideal para el estudio de diferentes controladores. Se recomienda implementar otros controladores como son: Modo Deslizante, Fuzzy e incluso controladores mediante Redes Neuronales.
- En este proyecto, se han realizado solo el seguimiento de caminos es decir que la figura deseada se dibuje correctamente y sin errores, pero sin considerar el tiempo en que se completa la misma, por ello se recomienda implementar seguimientos de trayectorias es decir que el camino este parametrizado en el tiempo, con lo cual sería necesario implementar un controlador para la velocidad de la esfera.
- Respecto a la construcción física de la plataforma, se recomienda reemplazar el eje pivote que sostiene a la plataforma por un eje que no presente movimiento en el eje Yaw para evitar causar perturbaciones y problemas al controlador.
- Para trabajos futuros se recomienda implementar el sistema de control y el procesamiento de las imágenes en un sistema embebido como por ejemplo usando la plataforma Raspberry Pi, lo cual eliminaría el uso de un computador y su dependencia de él.

# REFERENCIAS BIBLIOGRÁFICAS


[1] A. Gonzáles Marcos, F. J. Martínez de Pisón Ascacíbar, A. V. Pernía Espinoza, F. Alba Elías, M. Castejón Limas, J. Ordieres Meré y E. Vergara González, "Técnicas y algoritmos básicos de visión artificial", Logroño, 2006, pp. 11-36.

[2] J. Vélez Serrano, A. B. Moreno Díaz, Á. Sánchez Calle y J. L. Esteban Sánchez-Marín, "Visión por computador", Segunda ed., 2003, pp. 19-58.

[3] Visión Online. (2016). La visión artificial y su influencia en España. [En línea]. Available: http://www.visiononline.es/es/actualidad-en-vision-artificial/la-vision-artificial-y-su-influencia-en-espana.

[4] MathWorks - Matlab. (1994-2016). PDF Documentation for Image Processing Toolbox. [En línea]. Available: http://www.mathworks.com/help/pdf_doc/images/index.html.

[5] G. Welch y G. Bishop, *An Introduction to the Kalman Filter,* Dept. of Computer Science, Univerity of North Carolina at Chapel Hill, USA, 2006.

[6] Y. Bulut, D. Vines-Cavanaugh y D. Bernal, *Process and Measurement Noise Estimation for Kalman Filtering,* Northeastern University at Boston, USA, 2010.

[7] K. Ogata, "Ingeniería de control moderna", Quinta ed., Madrid, 2010.

[8] T. Álamo Cantarero, *Diseño del Controlador PID,* Dept. de Ingeniería de Sistemas y Automática, Universidad de Sevilla, España.

[9] K. Ogata, "Sistemas de Control en Tiempo Discreto", Segunda ed., Prentice Hall, 1996.

[10] National Instruments Corporation. (2014). Comunicación Serial: Conceptos Generales. [En línea]. Available: http://digital.ni.com/public.nsf/allkb/039001258CEF8FB686256E0F005888D1.

[11] N. G. Forero Saboya, *Normas de Comunicación en Serie: RS-232, RS-422 y RS-485,* Universidad Distrital de Bogotá, Colombia, 2012.

[12] maxEmbedded. (2013). Serial Communication. [En línea]. Available: http://maxembedded.com/2013/09/serial-communication-introduction/#modes.

[13] CCM. (2016). USB (Bus de serie universal). [En línea]. Available: http://es.ccm.net/contents/407-usb-bus-de-serie-universal.

[14] Hawlett-Packard Company, Intel Corporation, Microsoft Corportaion, Renesas Corporation, ST-Ericsson, Texas Instruments, *Universal Serial Bus 3.1 Specification,* Revisión 1.0, 2013.

[15] C. I. Rodríguez Saucedo. (2012). Eficiencia y seguridad en Bluetooth y Zigbee. [En línea]. Available: http://www.ptolomeo.unam.mx:8080/xmlui/handle/132.248.52.100/229.

[16] Bluetooth. (2016). Bluetooth technology basics. [En línea]. Available: https://www.bluetooth.com/what-is-bluetooth-technology/bluetooth-technology-basics.

[17] Naylamp Mechatronics. (2016). Uso de servomotores con arduino. [En línea]. Available: http://www.naylampmechatronics.com/blog/33_Tutorial-uso-de-servomotores-con-arduino-.html.

[18] Futaba. (2004). Parts for Robot. [En línea]. Available: https://www.futaba.co.jp/en/robot/5350/index.html.

[19] S. F. Barret y D. J. Pack, "Microcontrollers Fundamentals for Engineers and Scientists", Primera ed., Morgan&Claypool Publishers, 2006.

[20] F. E. Valdés Pérez y R. Pallás Areny, "Microcontroladores, fundamentos y aplicaciones con PIC", 2007.

[21] Arp. (2011). Logitech C615 HD Webcam. [En línea]. Available: http://www.arp.ch/logitech-c615-hd-webcam-960-001056-5098630?area=004_003_001.

[22] ServoDatabase. (2009). Hitec HS-645MG - High Torque MG Servo. [En línea]. Available: http://www.servodatabase.com/servo/hitec/hs-645mg.

[23] Silicon Labs. (2016). SINGLE-CHIP USB TO UART BRIDGE - CP2102/9. [En línea]. Available: https://www.silabs.com/Support%20Documents/TechnicalDocs/CP2102-9.pdf.

[24] Siliceo. (2012). USB serie modulo RS232 UART TTL Cable puerto COM chip CP2102. [En línea]. Available: http://tienda.siliceo.es/es/usb-serie-ttl-serial-rs232/27-usb-serie-modulo-rs232-uart-ttl-cable-puerto-com-chip-cp2102.html.

[25] Olimex. (2006). Product Data Sheet - Module HC-06. [En línea]. Available: https://www.olimex.com/Products/Components/RF/BLUETOOTH-SERIAL-HC-06/resources/hc06.pdf.

[26] Atmel. (2015). ATMEL 8-BIT MICROCONTROLLER. [En línea]. Available: http://www.atmel.com/Images/Atmel-8271-8-bit-AVR-Microcontroller-

ATmega48A-48PA-88A-88PA-168A-168PA-328-328P_datasheet_Complete.pdf.

[27] Arduino. (2016). Arduino UNO. [En línea]. Available: https://www.arduino.cc/en/Main/ArduinoBoardUno.

[28] M. Rogério Fernandes, *Rastreamento de Objetos usando Filtro de Kalman,* Universidade Tecnológica Federal do Paraná, 2015.

[29] M. Nokhbeh y D. Khashabi, *Modelling and Control of Ball-Plate System,* Amirkabir University of Technology, Irán, 2011.

[30] G. Andrews, C. Colasuonno y A. Herrmann, *Ball on Plate Balancing System,* Rensselaer Polytechnic Institute, NY, USA, 2004.

[31] R. J. Mantz y P. E. Troncoso, Control Moderno: Controlabilidad y observabilidad, Universidad Nacional de la Plata, Buenos Aires, 2016.

[32] C. R. Marco Gamero, *Propuesta de resolución del sistema matricial en el segundo método de estabilidad de Lyapunov,* Universidad Nacional Experimental Politécnica Antonio José de Sucre, Guayana, 2005, pp. 167-171.

[33] F. Morero, "Introduccion a la OOP", Grupo EIDOS, 2000.

[34] Developers - Android. (2016). Starting an Activity. [En línea]. Available: https://developer.android.com/training/basics/activity-lifecycle/starting.html.

# ANEXOS

## ANEXO A. GUÍA DE CONFIGURACIONES Y PUESTA EN MARCHA

### A.1 REQUERIMIENTOS MÍNIMOS DEL SISTEMA

- Sistema operativo de 64 bits.
- Memoria RAM de 4 GB.
- Espacio mínimo en disco para instalación de programas:  20GB.
- Puertos Seriales USB: 3.

### A.2 PREPARACIÓN DEL AMBIENTE DE TRABAJO

En el CD Proporcionado para el Laboratorio se encuentran los siguientes archivos y carpetas:

Carpeta Documentos:

- Tesis – Plataforma de Estabilización.
- Manual de Usuario y Puesta en marcha.
- Listado de componentes.
- Plan de Proyecto de Titulación.

Carpeta Ejecutables:

- Carpeta Principal_BOP: Contiene el archivo Principal_BOP.exe
- Carpeta Auxiliar_BOP: Contiene el archivo Auxiliar_BOP.exe
- Carpeta Móvil_BOP: Contiene el archivo Móvil_BOP.apk

Carpeta Librerias:

- RXTXcomm x64
    - ✓ RXTXcomm.jar
    - ✓ RxtxSerial.dll

Carpeta Instaladores

- Virtual Serial Port
- NetBeans
- Android Studio

- Matlab 2015b
- Kit de Desarrollo Java

Carpeta CódigoFuente

- BallOnPlateMat – Matlab
- BallOnPlateJava – NetBeans
- BallOnPlate – Android Studio

En primer lugar, se requiere de un puerto virtual que será emulado con Virtual Serial Port y serán del tipo pareja o “pair” y tendrán los nombres COM220 y COM221. Es necesario que el puerto virtual tenga los nombres que se han especificado antes, ya que con ese identificador se han incluido en las dos interfaces, y de no ser así, los dos programas que usan el puerto virtual darán errores y no correrán. La Figura A.1 muestra el programa para simular puertos virtuales.

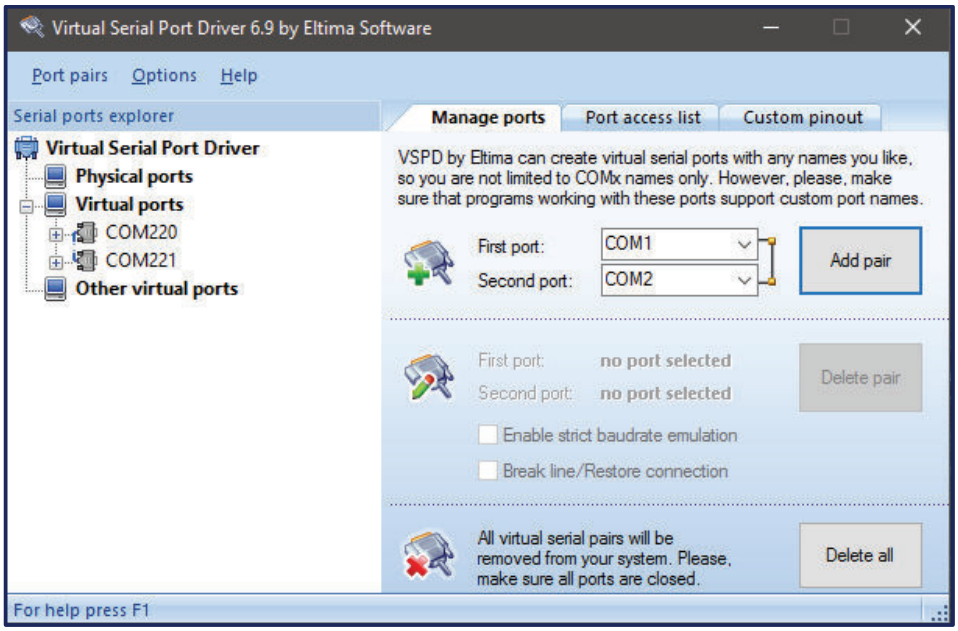



**Figura A.1.** Virtual Serial Port

En segundo lugar, para usar la aplicación auxiliar se requiere tener instalado el Kit de desarrollo Java en el computador. Para ello se instala el JDK de Java mediante el archivo *JDK-8u91-x64* ubicado en la carpeta Kit de Desarrollo Java.

Una vez instalado el JDK se debe agregar a la variable de entorno del sistema operativo *PATH* la ruta donde se encuentra el compilador de Java.

Propiedades del Sistema – Variables de entorno – Path:

$$Agregar\ <JAVA\_HOME>\backslash bin$$

Luego de la carpeta RXTXcomm x64 copiar los archivos rxtxSerial.dll y RXTXcomm.jar a las siguientes ubicaciones:

$$RXTXcomm.jar\ \text{---}>\ <JAVA\_HOME>\backslash jre\backslash lib\backslash ext$$

$$rxtxSerial.dll\ \text{---}>\ <JAVA\_HOME>\backslash jre\backslash bin$$

Donde <*JAVA_HOME*>\ es la dirección de la instalación del JDK.

Nota: Ya sea que se usen los archivos ejecutables o se corran las aplicaciones desde el código fuente usando las suites de programación adecuada, los pasos anteriores deben ser hechos correctamente para poder usar la plataforma.

Para poder abrir los archivos de código fuente se necesitan las siguientes suites de Programación con las versiones que se indican en la Tabla A.1. las cuales además se encuentran en el CD proporcionado.

**Tabla A.1.** Suites de Programación

| IDE | Versión | Página del Desarrollador (Descarga) | Licencia |
|---|---|---|---|
| Matlab | 2015b | http://www.mathworks.com/ | Pagada |
| NetBeans | 8.1 | https://netbeans.org/downloads/ | Gratuita |
| Android Studio | 2.1.2 | https://developer.android.com/studio/index.html | Gratuita |

La instalación de Matlab se la realiza normalmente, pero una vez instalado se necesita descargar un archivo de soporte de la cámara de video mediante la aplicación de descarga que proporciona Matlab, llamada: Get Hardware Support Packages. La Figura A.2 muestra una captura de pantalla de la pestaña para instalación del soporte de hardware de la cámara.

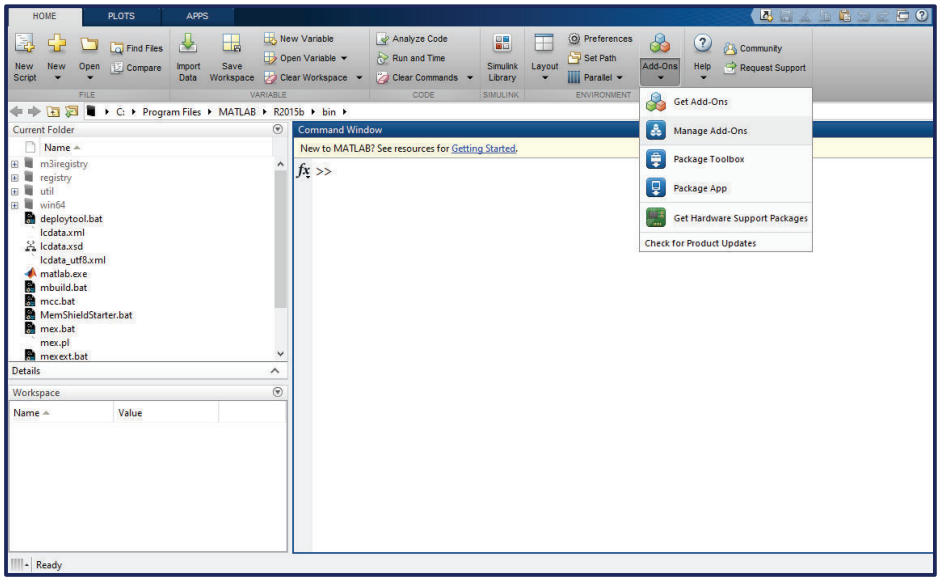



**Figura A.2.** Instalación de soporte para cámaras USB

Una vez abierta la aplicación de descarga, ir a la pestaña *Install From Internet* y finalmente instalar *OS Generic Video Interface*.

La instalación de NetBeans y Android Studio se realiza normalmente como si se tratase de cualquier otra aplicación.

A continuación, se detallan aspectos importantes a considerar para el correcto funcionamiento del sistema.

1. Los puertos seriales que se pide sean ingresados en el momento de la configuración inicial son los puertos generados por el sistema operativo cuando se conectan los módulos USB-TTL, para poder ver esta información se debe ir al administrador de dispositivos y en el apartado Puertos (COM y LPT) como se observa en la Figura A.3.

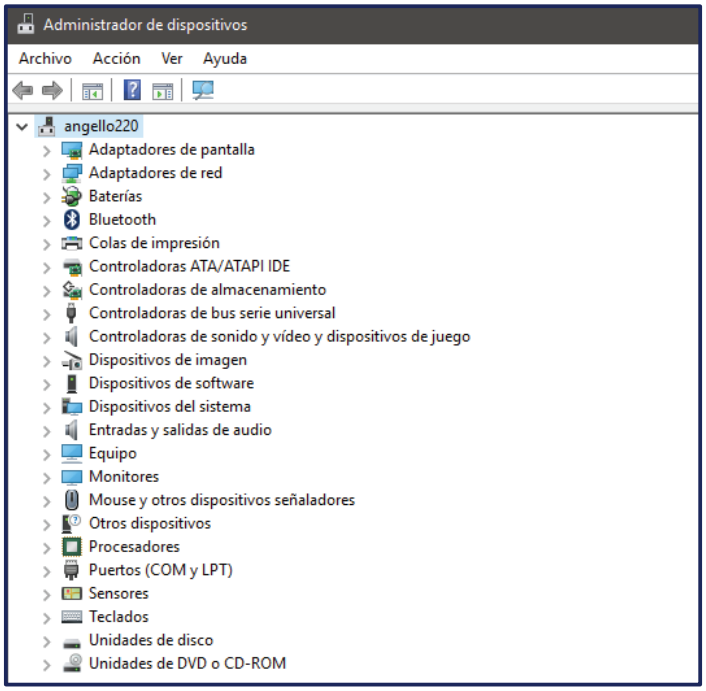



**Figura A.3.** Verificación de puertos seriales

2. La velocidad de transmisión usada en todas las comunicaciones es de 9600 Baudios.
3. Una vez ingresada la información de los puertos y velocidades correctamente presionando el botón Guardar Configuración se está listo para poder realizar el control de posición y seguimiento de caminos con la plataforma. La pantalla que se muestra en ese momento se visualiza en la Figura A.4.

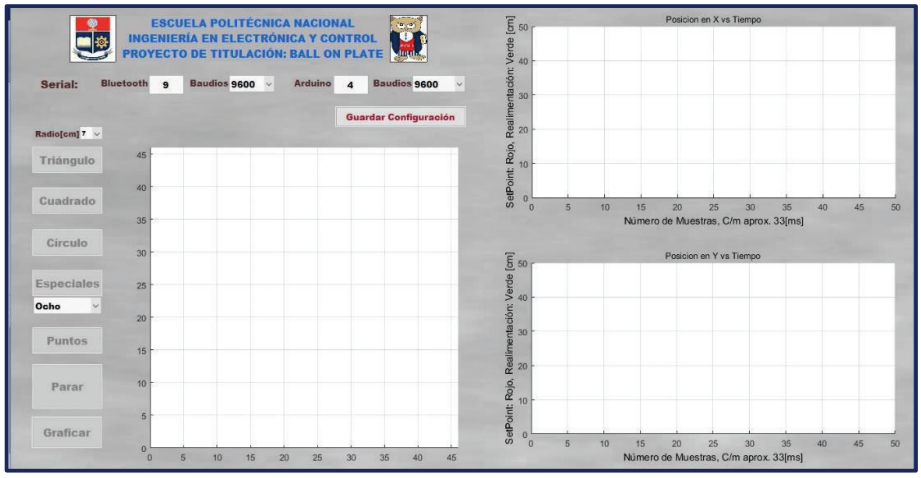



**Figura A.4.** Interfaz de control inicial

4. Para poder establecer una referencia de puntos o caminos los botones deben estar habilitados, esto se consigue alternando con el botón “Parar/Continuar”.
5. Cuando se pare la ejecución de un comando, se activa el botón “Graficar” y con ello se puede alternar entre las diferentes graficas que presenta la interfaz, como son posición, referencia, y control.
6. La aplicación para Smartphone ha sido desarrollada y probada en el sistema operativo Andrioid 6.0, en un Smartphone cuya pantalla tiene las siguientes dimensiones: 720x1280 pixeles, 5.0 pulgadas. Un cambio de estos parámetros podría afectar su normal funcionamiento.

# ANEXO B. PLANOS DE LA IMPLEMNTACIÓN DEL SISTEMA MECÁNICO

## B.1 PLANO DEL SOPORTE PARA LA CÁMARA

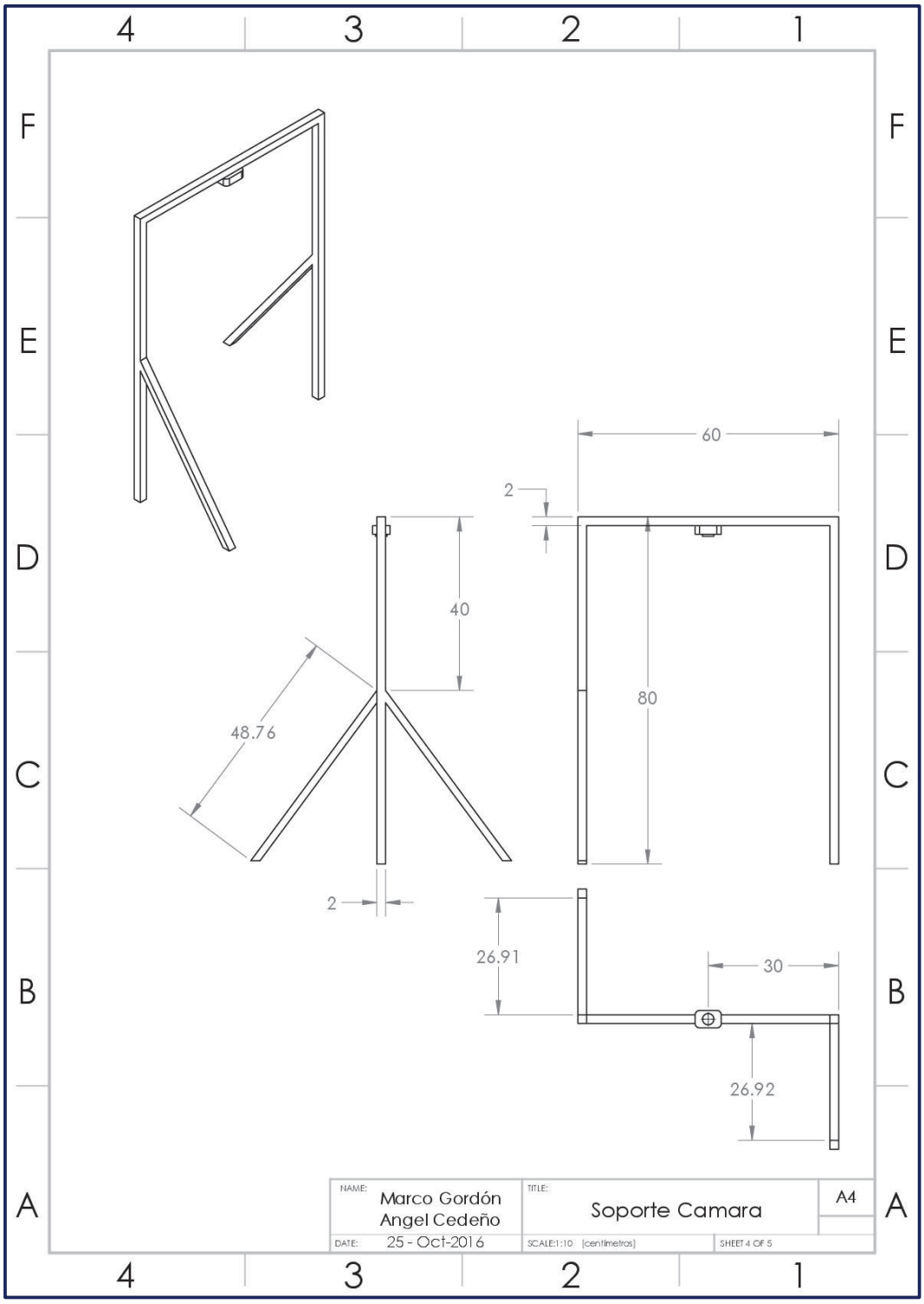



**Figura B.1.** Plano del soporte para la cámara

## B.2 PLANO DEL SOPORTE PARA LOS SERVOMOTORES

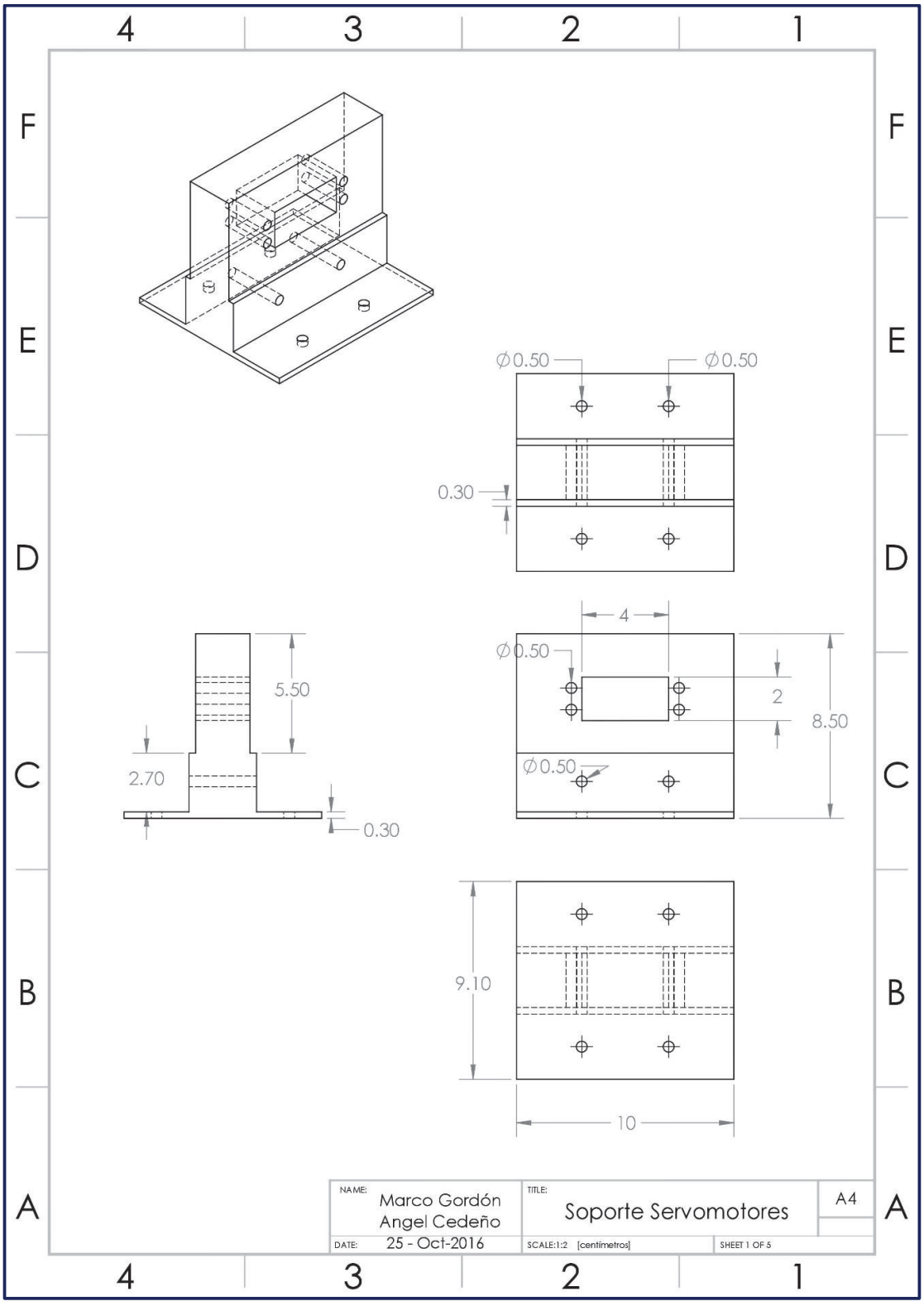



**Figura B.2.** Plano del soporte para los servomotores

## B.3 PLANO DEL EJE PIVOTE

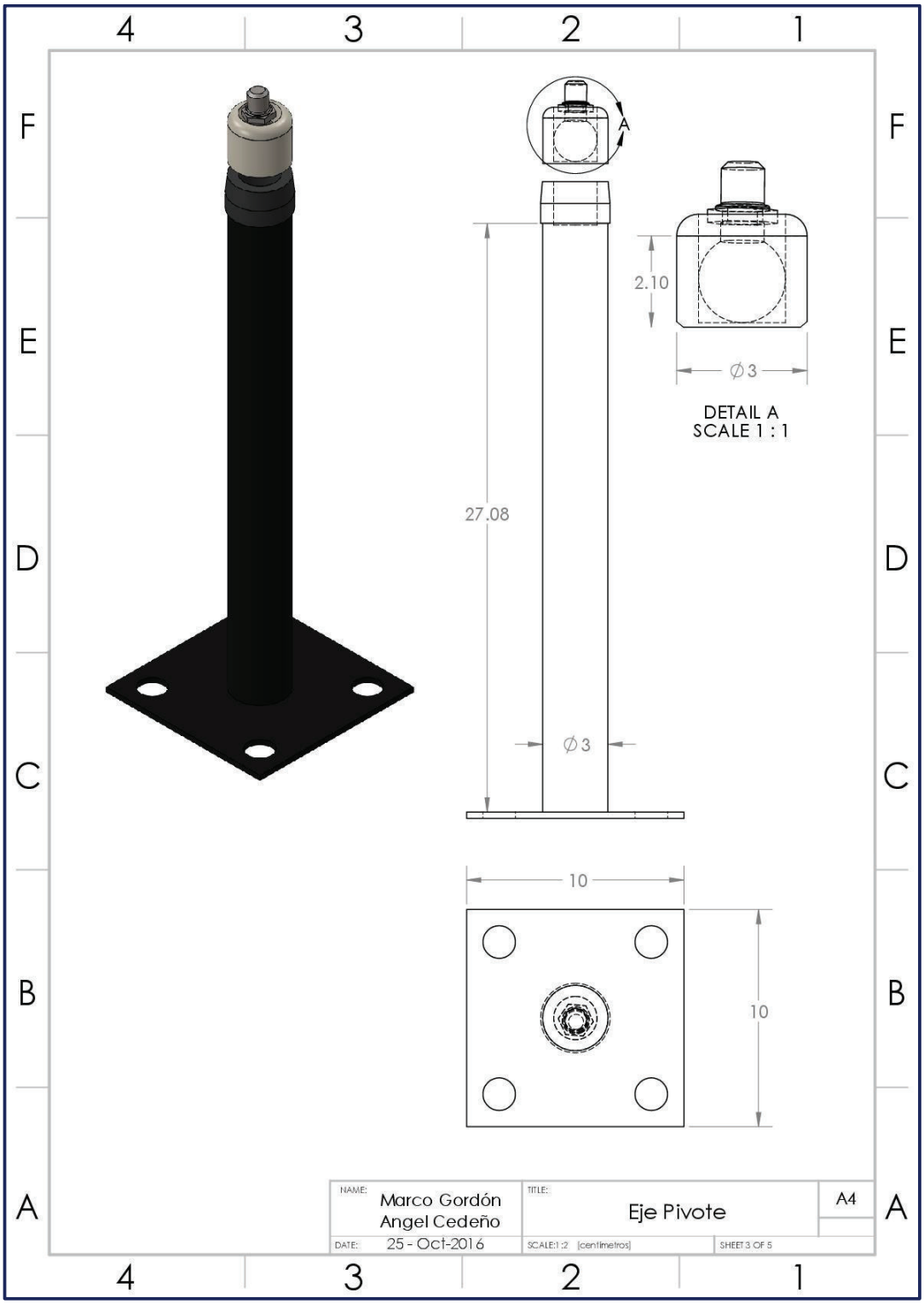
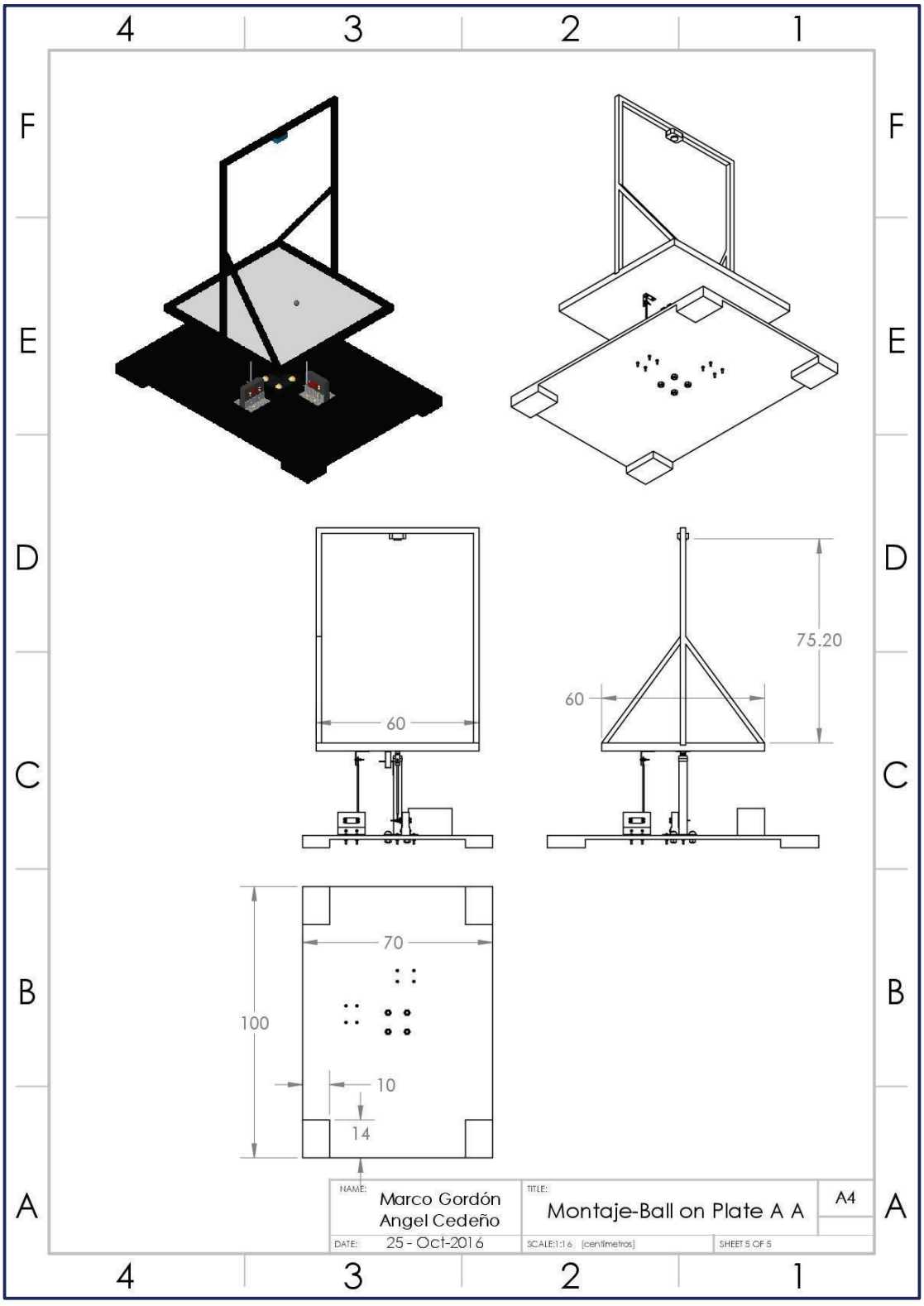
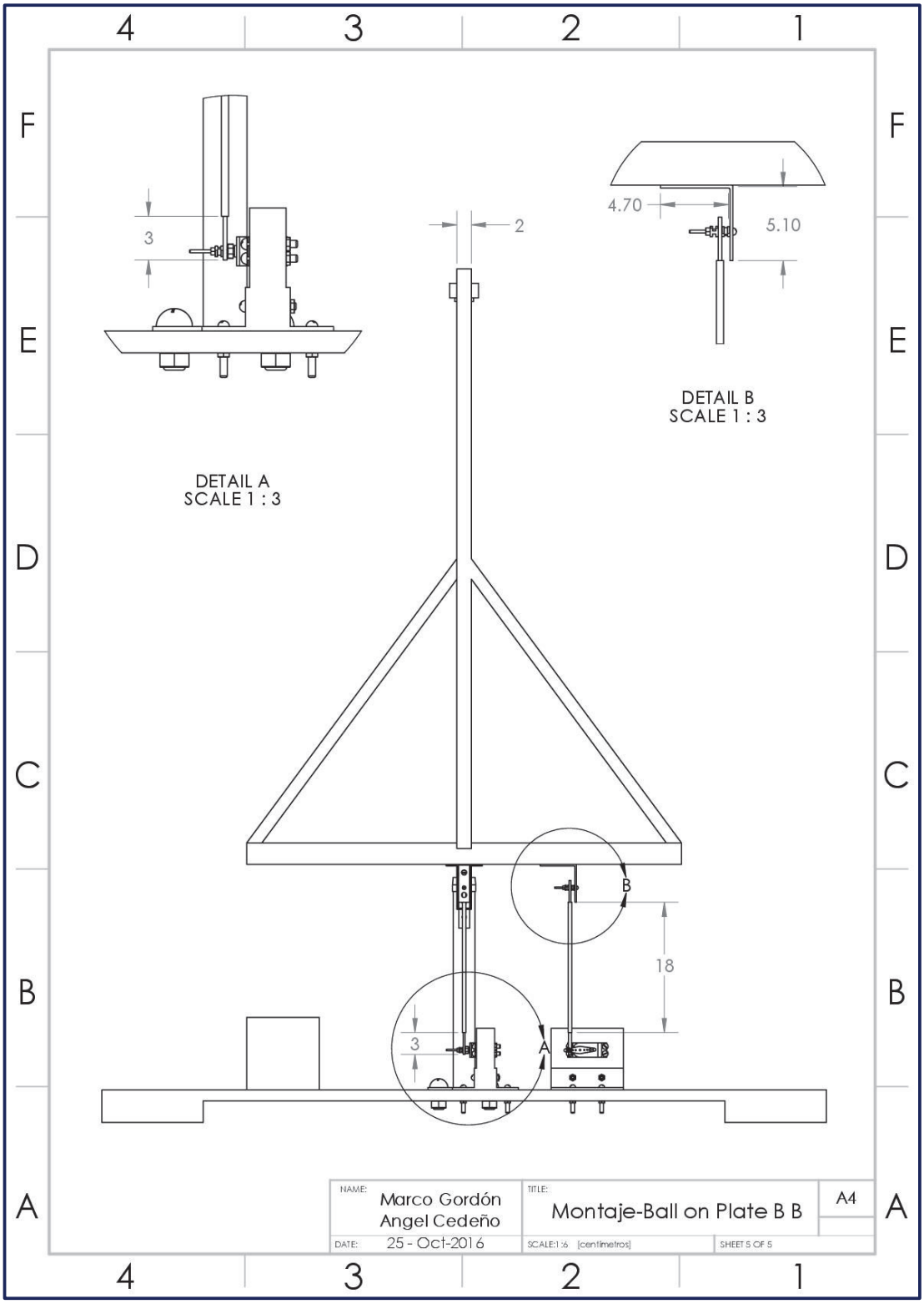



**Figura B.3.** Plano del eje pivote

## B.4 PLANOS DEL MONTAJE DEL SISTEMA




4
3
2
1
F
E
D
C
B
A
3
2
4.70
5.10
DETAIL A
SCALE 1 : 3
DETAIL B
SCALE 1 : 3
B
18
3
A
NAME: Marco Gordón
Angel Cedeño
DATE: 25 - Oct-2016
TITLE: Montaje-Ball on Plate B B
A4
SCALE:1:6 [centímetros]
SHEET 5 OF 5

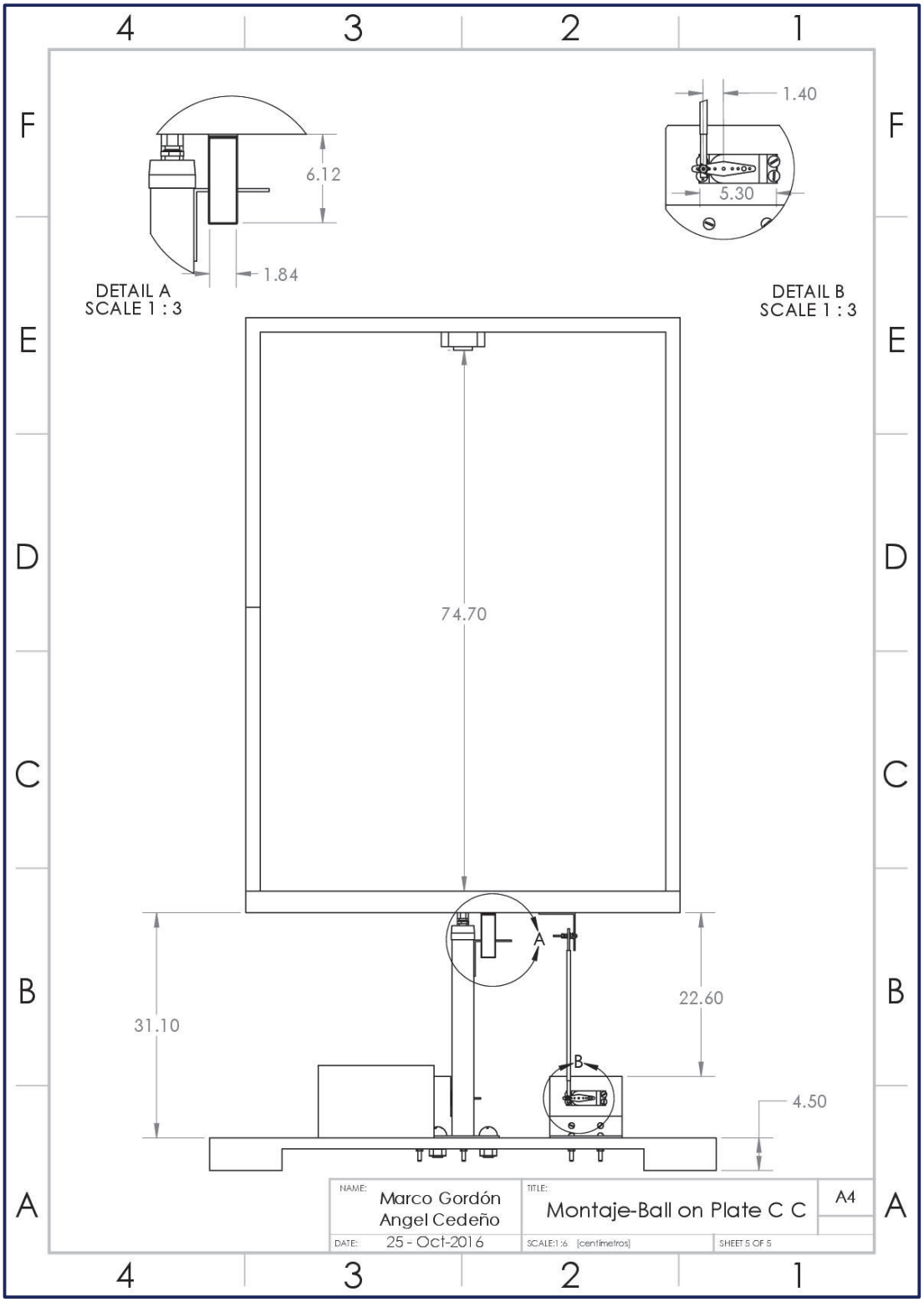
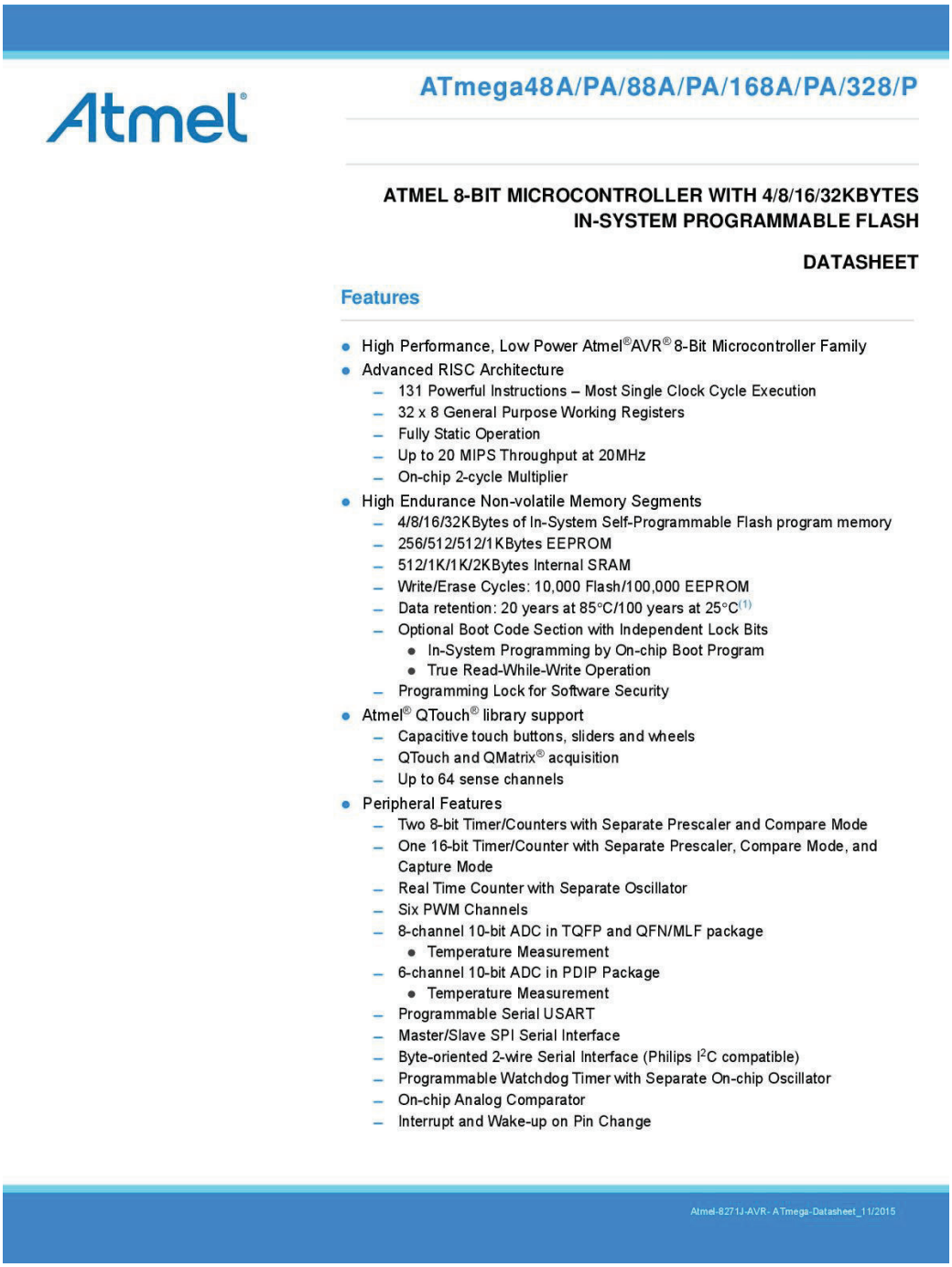



**Figura B.4.** Planos del montaje del sistema

# ANEXO C. HOJAS DE DATOS

## C.1 MICROCONTROLADOR ATMEGA 328



Atmel

ATmega48A/PA/88A/PA/168A/PA/328/P

ATMEL 8-BIT MICROCONTROLLER WITH 4/8/16/32KBYTES IN-SYSTEM PROGRAMMABLE FLASH

DATASHEET

Features

- High Performance, Low Power Atmel®AVR® 8-Bit Microcontroller Family
- Advanced RISC Architecture
  - 131 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 20 MIPS Throughput at 20MHz
  - On-chip 2-cycle Multiplier
- High Endurance Non-volatile Memory Segments
  - 4/8/16/32KBytes of In-System Self-Programmable Flash program memory
  - 256/512/512/1KBytes EEPROM
  - 512/1K/1K/2KBytes Internal SRAM
  - Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
  - Data retention: 20 years at 85°C/100 years at 25°C(1)
  - Optional Boot Code Section with Independent Lock Bits
    - In-System Programming by On-chip Boot Program
    - True Read-While-Write Operation
  - Programming Lock for Software Security
- Atmel® QTouch® library support
  - Capacitive touch buttons, sliders and wheels
  - QTouch and QMatrix® acquisition
  - Up to 64 sense channels
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Six PWM Channels
  - 8-channel 10-bit ADC in TQFP and QFN/MLF package
    - Temperature Measurement
  - 6-channel 10-bit ADC in PDIP Package
    - Temperature Measurement
  - Programmable Serial USART
  - Master/Slave SPI Serial Interface
  - Byte-oriented 2-wire Serial Interface (Philips $I^2C$ compatible)
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
  - Interrupt and Wake-up on Pin Change

Atmel-8271J-AVR- ATmega-Datasheet_11/2015

**Figura C.1.** Hoja de datos del microcontrolador ATmega 328

Para mayor información puede recurrir a la página web de la referencia [26].

## C.2 MÓDULO USB-TTL (CHIP CP2102)

SILICON LABS

# CP2102/9

## SINGLE-CHIP USB TO UART BRIDGE

**Single-Chip USB to UART Data Transfer**
- Integrated USB transceiver; no external resistors required
- Integrated clock; no external crystal required
- Internal 1024-byte programmable ROM for vendor ID, product ID, serial number, power descriptor, release number, and product description strings
  - EEPROM (CP2102)
  - EPROM (One-time programmable) (CP2109)
- On-chip power-on reset circuit
- On-chip voltage regulator
  - 3.3 V output (CP2102)
  - 3.45 V output (CP2109)
- 100% pin and software compatible with CP2101

**USB Function Controller**
- USB Specification 2.0 compliant; full-speed (12 Mbps)
- USB suspend states supported via SUSPEND pins

**Asynchronous Serial Data BUS (UART)**
- All handshaking and modem interface signals
- Data formats supported:
  - Data bits: 5, 6, 7, and 8
  - Stop bits: 1, 1.5, and 2
  - Parity: odd, even, mark, space, no parity
- Baud rates: 300 bps to 1 Mbps
- 576 Byte receive buffer; 640 byte transmit buffer
- Hardware or X-On/X-Off handshaking supported
- Event character support
- Line break transmission

**Virtual COM Port Device Drivers**
- Works with existing COM port PC Applications
- Royalty-free distribution license
- Windows 8/7/Vista/Server 2003/XP/2000
- Mac OS-X/OS-9
- Linux

**USBXpress™ Direct Driver Support**
- Royalty-Free Distribution License
- Windows 7/Vista/XP/Server 2003/2000
- Windows CE

**Example Applications**
- Upgrade of RS-232 legacy devices to USB
- Cellular phone USB interface cable
- USB interface cable
- USB to RS-232 serial adapter

**Supply Voltage**
- Self-powered: 3.0 to 3.6 V
- USB bus powered: 4.0 to 5.25 V

**Package**
- RoHS-compliant 28-pin QFN (5x5 mm)

**Ordering Part Numbers**
- CP2102-GM
- CP2109-A01-GM

**Temperature Range: –40 to +85 °C**

**Note:** For newer designs, the CP2102N devices offer compatible footprints and are recommended for use instead of the CP2102/9. See the Silicon Labs website (www.silabs.com/usbxpress) for more information.

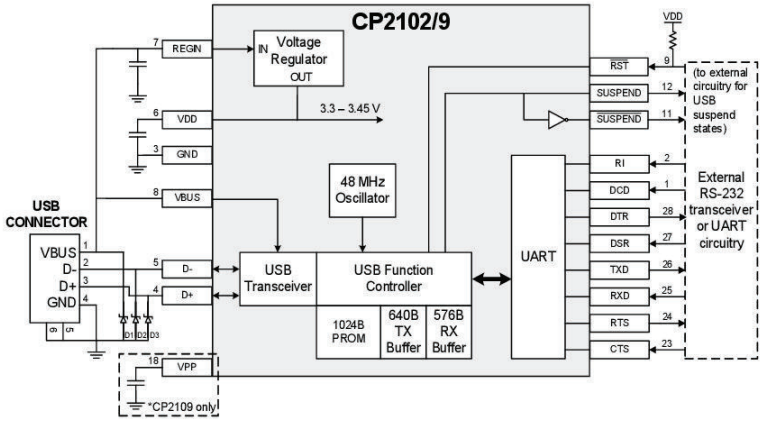



**Figure 1. Example System Diagram**

**Figura C.2.** Hoja de datos del módulo USB-TTL (chip CP2102)

Para mayor información puede recurrir a la página web de la referencia [23].

## C.3 MÓDULO DE COMUNICACIÓN BLUETOOTH HC-06



### 2. Feature

- Wireless transceiver
  - Sensitivity (Bit error rate) can reach -80dBm.
  - The change range of output's power: -4 - +6dBm.
- Function description (perfect Bluetooth solution)
  - Has an EDR module; and the change range of modulation depth: 2Mbps - 3Mbps.
  - Has a build-in 2.4GHz antenna; user needn't test antenna.
  - Has the external 8Mbit FLASH
  - Can work at the low voltage (3.1V~4.2V). The current in pairing is in the range of 30～40mA. The current in communication is 8mA.
  - Standard HCI Port (UART or USB)
  - USB Protocol: Full Speed USB1.1, Compliant With 2.0
  - This module can be used in the SMD.
  - It's made through RoHS process.
  - The board PIN is half hole size.
  - Has a 2.4GHz digital wireless transceiver.
  - Bases at CSR BC04 Bluetooth technology.
  - Has the function of adaptive frequency hopping.
  - Small (27mm×13mm×2mm)
  - Peripherals circuit is simple.
  - It's at the Bluetooth class 2 power level.
  - Storage temperature range: -40 ℃ - 85℃， work temperature range: -25 ℃ - +75℃
  - Any wave inter Interference: 2.4MHz， the power of emitting: 3 dBm.
  - Bit error rate: 0. Only the signal decays at the transmission link, bit error may be produced. For example, when RS232 or TTL is being processed, some signals may decay.
- Low power consumption
- Has high-performance wireless transceiver system
- Low Cost

www.wavesen.com Phone: 020-84083341 Fax: 020-84332079 QQ:1043073574
Address: Room 527, No.13, Jiangong Road, Tianhe software park, Tianhe district, Guangzhou Post: 510660
Technology consultant: support@wavesen.com Business consultant:sales@wavesen.com
Complaint and suggestion: sunbirdit@hotmail.com

**Figura C.3.** Hoja de datos del módulo de comunicación bluetooth HC-06

Para mayor información puede recurrir a la página web de la referencia [25].

## C.4 CÁMARA LOGITECH C615

Logitech®

# HD Webcam C615

Logitech

### Package specifications

| | Primary pack | Master shipper carton |
|---|---|---|
| Part # WER | 960-000735 | n/a |
| Bar Code | 5099206028036 (EAN-13) | 50992060280315 (SCC-14) |
| Part # Central | 960-000736 | n/a |
| Bar Code | 5099206028043 (EAN-13) | 50992060280414 (SCC-14) |
| Part # EER | 960-000737 | n/a |
| Bar Code | 5099206028050 (EAN-13) | 50992060280513 (SCC-14) |
| Weight | 226.8 g | 1.960 kg |
| Lenght | 20.95 cm | 46.50 cm |
| Width | 7.62 cm | 11.60 cm |
| Height / depth | 15.24 cm | 27.80 cm |
| Volume | 2.433dm3 | 0.02230m3 |
| 1 primary pack | 1 | n/a |
| 1 intermediate pack | 0 | n/a |
| 1 master shipper carton | 8 | 1 |
| 1 pallet EURO | 432 | 54 |
| 1 container 20ft | 10080 | 1260 |
| 1 container 40ft | 21280 | 2660 |
| 1 container 40ft HQ | 23408 | 2926 |

### Technical Specifications

- Full HD 1080p video capture (up to 1920 x 1080 pixels) with recommended system
- HD video calling (1280 x 720 pixels) with recommended system
- Logitech Fluid Crystal™ Technology*
- Autofocus
- Photos: up to 8 megapixels (software enhanced)
- Built-in mics with automatic noise reduction
- Hi-Speed USB 2.0 certified
- Universal clip fits laptops, LCD or CRT monitors

Logitech webcam software:
- Logitech Vid™ HD
- Automatic low-light correction
- Video and photo capture
- MAGIX™ photo and video editing**
- 1-click Facebook®, Twitter™ and YouTube™ HD upload (registration required)
- Logitech Video Effects™: fun filters, avatars, face accessories, video masks and mask maker***

* Requires installation of included software.
** Windows-based computers only. Email registration required. Additional terms and conditions apply.
*** Windows-based computers only. Intel® Pentium® 4 (2.8 GHz) recommended.

### Package Contents

- Webcam with 90-cm (3-foot) cable
- 90-cm (3-foot) extension cable
- Tripod-ready base
- Logitech webcam software with Logitech Vid™ HD (for PC and Mac®)
- User documentation
- 2-year manufacturer's guarantee and full product support

### System Requirements

- Windows® XP (SP2 or higher), Windows Vista®, Windows® 7 (32-bit or 64-bit) or Mac OS® X 10.5 or higher
- 1 GHz
- 512 MB RAM or more
- 200 MB hard drive space
- Internet connection
- USB 1.1 port (2.0 recommended)

For HD 720p video calling (on Logitech Vid™ HD) and Full HD 1080p video recording:
- 2.4 GHz Intel® Core™2 Duo
- 2 GB RAM
- 200 MB hard drive space
- USB 2.0 port
- 1 Mbps upload speed or higher
- 1280 x 720 screen resolution



**Figura C.4.** Hoja de datos de la cámara Logitech C615

Para mayor información puede recurrir a la página web de la referencia [21].