

# Facultad de Informática

Grado en Ingeniería Informática

## Trabajo de Fin de Grado

Ingeniería de computadores

Desarrollo del control de equilibrio y cinemática de un robot autobalanceado

Christian Mares

Julio 2019

Tutor: Luis Gardeazabal



# Resumen

---

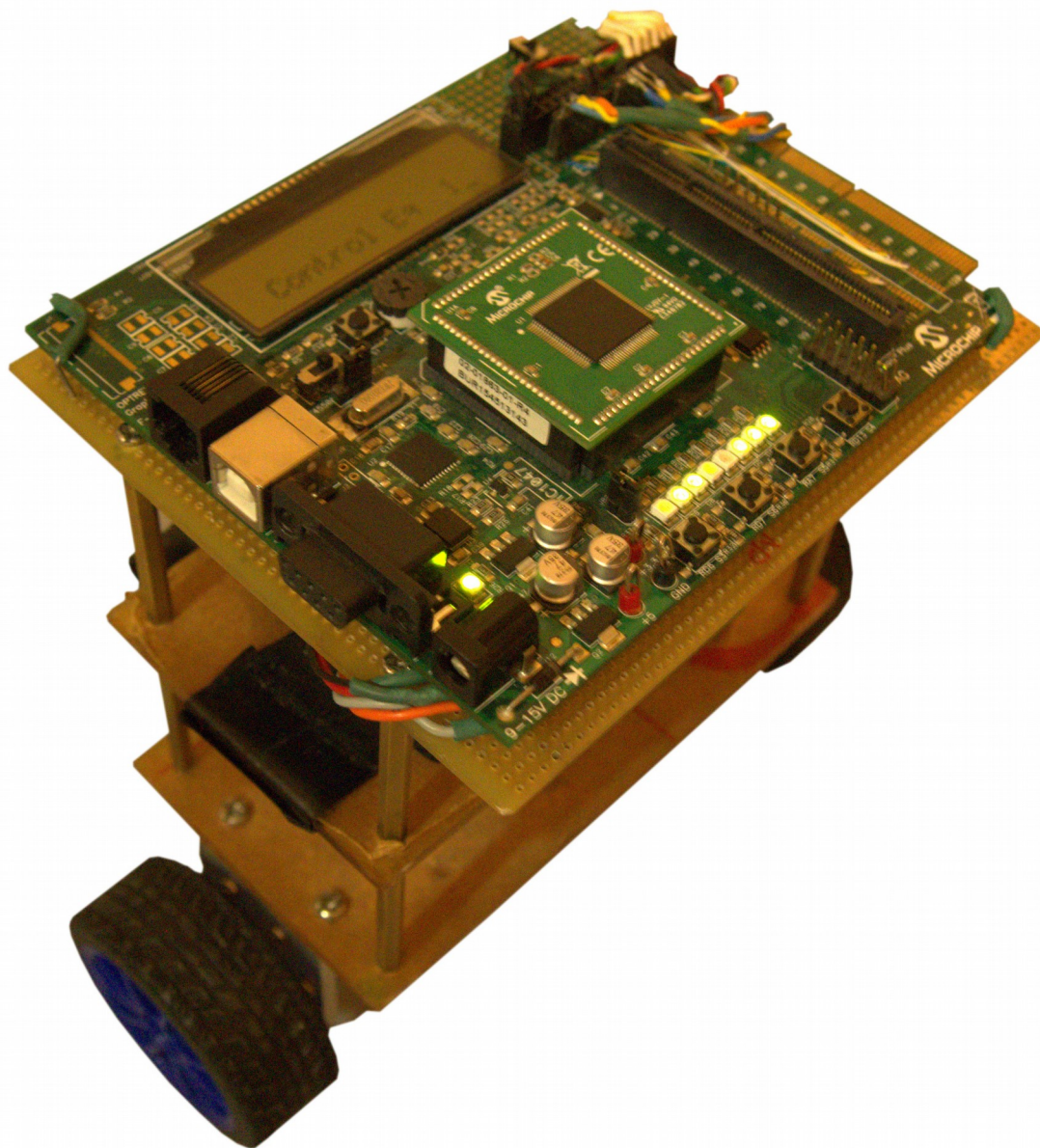
Este proyecto se ha llevado a cabo con la intención de mejorar y completar el sistema de control de un prototipo de robot balancín desarrollado en proyectos anteriores. Se espera poder utilizar los resultados en futuros desarrollos de aplicaciones que involucren la tecnología de robots autobalanceados.

Este trabajo ha consistido en el estudio de las características del prototipo y en el desarrollo de un sistema de control de equilibrio y cinemática. El sistema de control de equilibrio sirve para la conservación del equilibrio y la posición en superficies de cualquier tipo. El sistema de control de cinemática permite controlar acciones del robot relacionadas con desplazamiento, pero también aumenta la efectividad del control de equilibrio, haciendo posible la estabilidad en superficies inclinadas.

Para la realización de los sistemas de control se ha escogido un control proporcional, integral y derivativo (PID), muy extendido para el control de una variable en procesos industriales y comerciales. Ambos sistemas de control funcionan conjuntamente en un PID en cascada en el que uno (el control de cinemática, denominado externo) controla el valor deseado del otro (el control de equilibrio, denominado interno).

Al final del desarrollo se han cumplido satisfactoriamente todas las metas propuestas al inicio del proyecto, dentro del alcance definido. Se ha obtenido un control de equilibrio robusto y estable y un control de cinemática efectivo, sin sacrificar características del equilibrio. El robot es capaz de permanecer inmóvil en la misma posición, tanto en planos horizontales como inclinados, compensar perturbaciones externas, recuperar su posición y desplazarse y maniobrar sobre la superficie de apoyo.

El control PID de cinemática en cascada ha sido la gran clave para la implementación de las características avanzadas de equilibrio, haciendo posible la estabilidad en planos inclinados y otras funcionalidades mixtas de equilibrio y locomoción. Gracias a este se han completado la mayoría de las metas del proyecto. Por otro lado, el control PID de equilibrio es la base del control de cinemática, por lo que su correcta programación y calibrado han resultado fundamentales, tomando gran parte del desarrollo.



# Índice

---

|   |           |
|---|-----------|
| Resumen.....  | III       |
| <b>1. Introducción, Objetivos y Alcance.....</b>                    | <b>1</b>  |
| 1.1 Introducción.....   | 2         |
| 1.2 Objetivos.....  | 2         |
| 1.2.1 Tareas a resolver.....  | 3         |
| 1.3 Alcance.....  | 4         |
| 1.4 Planificación inicial.....                                      | 5         |
| 1.5 Planificación final.....  | 6         |
| <b>2. Antecedentes.....</b>   | <b>7</b>  |
| 2.1 Estado del arte.....  | 8         |
| 2.2 Trabajos anteriores.....  | 10        |
| 2.2.1 Diseño original del robot.....                                | 11        |
| <b>3. Desarrollo del prototipo.....</b>                             | <b>13</b> |
| 3.1 Hardware utilizado.....   | 14        |
| 3.1.1 Introducción.....   | 14        |
| 3.1.2 Componentes.....  | 14        |
| 3.1.3 Desarrollo del diseño.....                                    | 16        |
| 3.2 Diseño del software.....  | 17        |
| 3.2.1 Introducción.....   | 17        |
| 3.2.2 Dispositivos programados.....                                 | 17        |
| 3.2.2.1 SENSORES: I2C, Módulo de sensores y fusión de sensores..... | 17        |
| 3.2.2.2 PWM y Motores.....  | 20        |
| 3.2.2.3 Input Capture[6] y Codificadores.....                       | 21        |
| 3.2.3 Sistema de control: PID.....                                  | 22        |
| 3.2.3.1 Introducción.....   | 22        |
| 3.2.3.2 Implementación[9].....                                      | 23        |
| 3.2.3.3 PID de equilibrio.....                                      | 26        |
| 3.2.3.4 PID de velocidad.....                                       | 27        |
| 3.2.4 Calibración del PID.....                                      | 28        |
| 3.2.4.1 Distintos métodos.....                                      | 28        |
| 3.2.4.2 Método manual.....  | 28        |
| 3.2.4.3 Calibrador automático.....                                  | 29        |
| 3.2.5 Aspectos misceláneos de la programación.....                  | 32        |
| 3.3 Puesta en marcha.....   | 34        |
| <b>4. Resultados.....</b>   | <b>37</b> |
| 4.1 Sistemas de control.....  | 38        |
| 4.1.1 Equilibrio.....   | 38        |
| 4.1.2 Velocidad.....  | 38        |
| 4.2 Tareas resueltas.....   | 39        |
| 4.2.1 Objetivos de estabilidad.....                                 | 39        |
| 4.2.2 Objetivos de cinemática.....                                  | 39        |
| <b>5. Conclusiones y trabajo futuro.....</b>                        | <b>41</b> |

## Lista de tablas

|                                     |   |
|-------------------------------------|---|
| Tabla 1: Planificación inicial..... | 5 |
| Tabla 2: Planificación final.....   | 6 |

## Lista de figuras

|  |    |
|--|----|
| Figura 1: Segway.....  | 8  |
| Figura 2: Monociclo eléctrico.....   | 8  |
| Figura 3: Hoverboard.....  | 9  |
| Figura 4: Onewheel.....  | 9  |
| Figura 5: iBOT.....  | 9  |
| Figura 6: Honda U3-X.....  | 9  |
| Figura 7: Honda UNI-CUB $\beta$ .....  | 9  |
| Figura 8: Robot balancín.....  | 10 |
| Figura 9: Disposición original de los componentes.....   | 16 |
| Figura 10: Posición final del sensor en el robot.....  | 18 |
| Figura 11: Ejes del módulo de sensores.....  | 19 |
| Figura 12: Obtención del ángulo del acelerómetro.....  | 19 |
| Figura 13: Demostración de la fusión de sensores.....  | 20 |
| Figura 14: Esquema del proceso del PID.....  | 23 |
| Figura 15: Integración rectangular.....  | 24 |
| Figura 16: Oscilaciones del robot superando la frecuencia en amplitud.....   | 30 |
| Figura 17: Estabilización de las oscilaciones en altas frecuencias debido al efecto de una alta ganancia en el término derivativo..... | 31 |
| Figura 18: Diagrama de flujo del bucle principal.....  | 32 |
| Figura 19: Estructura del menú LCD.....  | 33 |

# 1

---

## Introducción, Objetivos y Alcance

---

# 1.1 Introducción

---

En este trabajo se lleva a cabo el estudio de las características de un prototipo de robot balancín e implementación del control de equilibrio y cinemática. El prototipo es continuación del trabajo previo realizado en [1] y [2] que se basa en el modelo del péndulo invertido. A lo largo del documento se detalla el desarrollo y evolución del mismo.

El diseño físico y funcionamiento desarrollado están basados en los mismos principios que los de los conocidos *Segways*, que se emplean para el transporte individual en situaciones que requieren recorrer continuamente distancias medias, o los *Hoverboards* que habitualmente tienen un uso recreativo.

El prototipo desarrollado es una versión reducida de los dispositivos anteriores, manteniendo los aspectos clave de la disposición de sus elementos que lo caracteriza como un péndulo invertido. Sustenta todos sus componentes sobre dos ruedas coaxiales y mantiene el equilibrio haciéndolas girar para situar la base bajo el centro de masa, manteniendo la posición erguida.

El problema del equilibrio no es trivial, ya que un péndulo invertido, por naturaleza, no es estable y tiende a salir de su posición de equilibrio inestable cuando se produce una mínima perturbación en el sistema. Por este motivo, se requiere un sistema de control que proporcione una constante corrección de su estado. Para mantener una posición de equilibrio estable y duradera se requiere un sistema de control fiable, bien calibrado y con un conocimiento preciso de la inclinación para poder contrarrestar las perturbaciones rápidamente y recuperar lo antes posible la posición de equilibrio.

A lo largo de este documento se describen las características que condicionan el funcionamiento del prototipo, el desarrollo y las modificaciones llevadas a cabo sobre éste para implementar un sistema de control de equilibrio robusto y el sistema de control de cinemática.

## 1.2 Objetivos

---

El Objetivo a cumplir en este proyecto ha sido desarrollar el sistema de control para el robot. Esto significa controlar acciones del prototipo a voluntad, como el movimiento a una determinada velocidad a lo largo de una distancia dada permitan un control automático o manual. Además, se busca la mayor versatilidad del robot, por eso se ha puesto gran énfasis en la estabilidad en general de éste y en particular en la habilidad de sostenerse en planos inclinados.

Para poder controlar el robot a voluntad hace falta un control del equilibrio robusto, que corrija la inclinación de éste rápidamente y con la menor cantidad de oscilaciones posible. Una vez conseguido esto, se ha trabajado en características avanzadas del equilibrio, como la recuperación de la posición frente a interferencias externas o la estabilidad en planos inclinados al igual que en características del movimiento, como recorrer distancias concretas a velocidades definidas o la habilidad de realizar giros.

En el siguiente apartado, tareas a resolver, se detalla cada uno de estos objetivos.



## 1.2.1 Tareas a resolver

### ♦ **Equilibrio**

La primera tarea a resolver ha sido conseguir el mantenimiento del equilibrio del robot, que ofrezca una buena estabilidad y que le permita permanecer erguido indefinidamente, minimizando el máximo posible las oscilaciones del sistema cuando se encuentre en equilibrio y libre de influencias externas.

### ♦ **Recuperación de caídas**

Cuando el robot se desequilibra, por ejemplo, porque recibe un impulso del exterior, este tiene que ser capaz de recuperarse dentro de unos márgenes razonables. Si el impulso es lo bastante débil como para que el robot pueda acelerar y desplazar su base más rápido que su centro de masa, entonces debe ser capaz de recuperarse, evitando la caída y volviendo a un punto de equilibrio estable. En este proceso, el robot debería realizar el mínimo de oscilaciones hasta recuperar el punto de equilibrio.

Las características de equilibrio y recuperación de caídas descritas sugieren que el sistema de control ha de tener una respuesta rápida y robusta, que no deje tiempo a que una perturbación de ninguna clase saque al robot de su posición de equilibrio por más de una fracción de segundo. Aunque no se trata del objetivo central, es fundamental para el resto del proyecto que el control de equilibrio cumpla estas condiciones. Sin él, es imposible implementar ninguna de las otras características que se describirán a continuación.

### ♦ **Recuperación de la posición inicial**

Esta es la capacidad de retornar a la posición en la que se encontraba el robot antes de, por ejemplo, ser empujado. Para ello, primero el robot debe recuperar el equilibrio y seguidamente desplazarse en la dirección opuesta al impulso que causó el desequilibrio. Esto supone una característica de control mixta de estabilidad y movimiento.

La solución ideal sería poder realizar la recuperación a medida que recobra el equilibrio, de forma que se aproxime directamente a la posición original. Esta solución resulta a priori difícil de conseguir, pues para ello hay que conocer muy bien los fundamentos del desplazamiento rectilíneo del robot.

Para conseguir la recuperación de la posición inicial se va a hacer uso de los codificadores incorporados en los motores del robot. Con ellos se puede medir el desplazamiento del robot por la rotación de las ruedas. Además, hay que desarrollar la capacidad de desplazamiento del robot de forma autónoma.

### ♦ **Desplazamiento y maniobrabilidad**

Consiste en la capacidad del robot para avanzar en línea recta en ambos sentidos, así como de rotar sobre una de sus ruedas, tanto en reposo como en movimiento, o sobre su propio eje.

El objetivo es que el robot se incline ligeramente y comience a acelerar suavemente, manteniendo un ángulo estable a una velocidad constante. Para ello se planea desarrollar un sistema de control en cascada, que aproveche la habilidad del sistema de control de equilibrio para mantener un cierto grado de inclinación del robot, hasta alcanzar la velocidad deseada. Esto supone la necesidad de un conocimiento preciso de la velocidad del robot.

Una buena maniobrabilidad implica una buena capacidad para hacer giros. Es por tanto imprescindible desarrollar, al menos, la capacidad de rotación sobre el propio eje mientras se mantiene el equilibrio. De esta manera se puede alcanzar cualquier posición en el mismo plano, a través de desplazamientos rectos en cualquier dirección. A parte de girar sobre si mismo, el robot también podría corregir su trayectoria a medida que se desplaza, controlando la velocidad de cada rueda por separado, siempre y cuando mantengan, en conjunto, el impulso suficiente para conservar el ángulo de inclinación.

#### ♦ Estabilidad en planos inclinados y cambios del centro de masa

Para mantenerse en equilibrio en un plano inclinado es necesario un impulso adicional. Sin éste, la gravedad tira del robot cuesta abajo y comienza a acelerar hasta que vuelve a una superficie horizontal. Para compensar el efecto de la gravedad, hay que añadir un impulso adicional en el sentido ascendente del plano.

Este problema es similar a un cambio repentino del centro de masa, como ocurre al apoyar un objeto sobre la parte superior del robot. Cuando esto sucede, el ángulo en el que se encuentra el equilibrio se ve alterado y el robot busca el reposo en una posición no estable, acelerando hasta perder el equilibrio.

Se espera resolver ambos problemas con el control de la velocidad. Como resulta difícil medir tanto la inclinación del plano sobre el que se encuentra, como calcular el centro de masa del robot, se va a utilizar la velocidad a la que el robot se desplaza, a causa de estas influencias, para calcular el ángulo de inclinación en que se encuentra la posición de equilibrio.

## 1.3 Alcance

---

Dadas las múltiples opciones que se pueden acometer en un proyecto de este tipo y de la limitación de tiempo, el proyecto se va a limitar al desarrollo del marco de control del robot. Los resultados deben servir para facilitar la implementación de cualquier aplicación que implique la movilidad o estabilidad del prototipo, pero no para el desarrollo de ninguna aplicación en concreto que haga el uso del control del robot como sería, por ejemplo, el control remoto.

A lo largo del desarrollo del proyecto se podrá modificar el prototipo como se desee, siempre que no implique, en lo posible, la adición de nuevos elementos como sensores y actuadores. Se puede cambiar la disposición de cualquier elemento y la adición de cableado y otras dependencias que resulten. No está contemplada la adición de una antena Bluetooth o sensores de distancia, que serían necesarios para una aplicación de control remoto o automático.

Tampoco se realizará ninguna modificación que tenga como resultado la alteración del modelo físico del robot. Es decir, para mantener el modelo del péndulo invertido no se realizará ninguna modificación que sitúe el centro de masa, ni ninguna cantidad de masa, por debajo del eje de las ruedas, ya que esto ayudaría a mantener el equilibrio haciendo un efecto de contrapeso. De la misma manera, no se añadirá ningún elemento que ayude a mantener el equilibrio a través de un efecto giroscópico, como podría obtenerse al hacer rotar rápidamente una masa cilíndrica.

## 1.4 Planificación inicial

La Tabla 1 resume la planificación propuesta al inicio del proyecto y que por imprevistos y retrasos acontecidos a lo largo del desarrollo se ha ido modificando para llegar a la planificación real.

| PT    | Nombre de tarea                                  | 2018 |    |    |    |    |    |    |    |    |     |     |     | 2019 |    |
|-------|--|------|----|----|----|----|----|----|----|----|-----|-----|-----|------|----|
|       |  | M1   | M2 | M3 | M4 | M5 | M6 | M7 | M8 | M9 | M10 | M11 | M12 | M1   | M2 |
| PT 0  | Seguimiento y control del proyecto               |      |    |    |    |    |    |    |    |    |     |     |     |      |    |
| PT 1  | Investigación preliminar                         |      |    |    |    |    |    |    |    |    |     |     |     |      |    |
| T 1.1 | Conceptos teóricos: PID                          |      |    |    |    |    |    |    |    |    |     |     |     |      |    |
| T 1.2 | Conceptos teóricos: Péndulo invertido            |      |    |    |    |    |    |    |    |    |     |     |     |      |    |
| T 1.3 | Prototipo: Características físicas               |      |    |    |    |    |    |    |    |    |     |     |     |      |    |
| T 1.4 | Prototipo: Componentes internos                  |      |    |    |    |    |    |    |    |    |     |     |     |      |    |
| H 1   | Finalización del aprendizaje                     |      |    | H1 |    |    |    |    |    |    |     |     |     |      |    |
| PT 2  | Puesta en marcha del robot                       |      |    |    |    |    |    |    |    |    |     |     |     |      |    |
| T 2.1 | Software controlador de periféricos del Pic      |      |    |    |    |    |    |    |    |    |     |     |     |      |    |
| T 2.2 | Desarrollo de interfaz LCD y RS232               |      |    |    |    |    |    |    |    |    |     |     |     |      |    |
| T 2.3 | Programación de motores                          |      |    |    |    |    |    |    |    |    |     |     |     |      |    |
| T 2.4 | Programación del módulo de sensores (SSC7150)    |      |    |    |    |    |    |    |    |    |     |     |     |      |    |
| H 2   | Prototipo funcional                              |      |    |    |    | H2 |    |    |    |    |     |     |     |      |    |
| PT 3  | Desarrollo del prototipo                         |      |    |    |    |    |    |    |    |    |     |     |     |      |    |
| T 3.1 | Implementación de la fusión de sensores          |      |    |    |    |    |    |    |    |    |     |     |     |      |    |
| T 3.2 | Implementación del PID de equilibrio             |      |    |    |    |    |    |    |    |    |     |     |     |      |    |
| T 3.3 | Implementación del PID de SP                     |      |    |    |    |    |    |    |    |    |     |     |     |      |    |
| H 3   | Prototipo equilibrado                            |      |    |    |    |    |    |    |    | H3 |     |     |     |      |    |
| PT 4  | Desarrollo de la cinemática                      |      |    |    |    |    |    |    |    |    |     |     |     |      |    |
| T 4.1 | Lecturas de codificadores y velocímetro/odómetro |      |    |    |    |    |    |    |    |    |     |     |     |      |    |
| T 4.2 | Implementar funciones de equilibrio avanzadas    |      |    |    |    |    |    |    |    |    |     |     |     |      |    |
| T 4.3 | Implementar funciones de cinemática              |      |    |    |    |    |    |    |    |    |     |     |     |      |    |
| H 4   | Prototipo final                                  |      |    |    |    |    |    |    |    |    |     |     |     | H4   |    |

Tabla 1: Planificación inicial

El paquete de trabajo 1 consiste en la adquisición de conocimientos necesarios para llevar a cabo el desarrollo del prototipo. Incluye el estudio de los conceptos teóricos pertinentes y del mismo robot.

El paquete de trabajo 2 consiste en la programación preliminar del robot. El resultado debería ser un robot con todo su software controlador preparado y funciones básicas para el control de actuadores, lectura de sensores y la base de una interfaz para el manejo de las funciones que se vayan a añadir.

El paquete de trabajo 3 consiste en el comienzo del desarrollo del sistema de control y filtrado de la señal del sensor para obtener un control de equilibrio básico y un sistema de calibrado del punto de equilibrio. La finalidad de este sistema de calibrado es mejorar la estabilidad sin necesidad de ajustar manualmente el punto de equilibrio y, potencialmente, su empleo en las funciones avanzadas de equilibrio y cinemática.

El paquete de trabajo 4 consiste en el final del desarrollo del prototipo, con la programación de las funciones avanzadas de equilibrio y cinemática.

A medida que el proyecto ha avanzado, se ha considerado que el sistema de calibrado del punto de equilibrio no tiene sentido en sí mismo y para la consecución de ningún otro objetivo. En lugar de ello, se ha comenzado el desarrollo del control de velocidad, que, a medida que ha avanzado el desarrollo, se ha observado que puede aportar a más funcionalidades que simplemente la de locomoción. Por ejemplo, puede utilizarse para el ajuste del punto de equilibrio o la recuperación de la posición.

## 1.5 Planificación final

La Tabla 2 resume la planificación resultante tras los cambios en los objetivos y retrasos sufridos a medida que ha avanzado el proyecto.

| PT    | Nombre de tarea                                  | 2018 |    |    |    |    |    |    |    |    |     |     |     | 2019 |    |    |    |    |    |    |
|-------|--|------|----|----|----|----|----|----|----|----|-----|-----|-----|------|----|----|----|----|----|----|
|       |  | M1   | M2 | M3 | M4 | M5 | M6 | M7 | M8 | M9 | M10 | M11 | M12 | M1   | M2 | M3 | M4 | M5 | M6 | M7 |
| PT0   | Seguimiento y control del proyecto               |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    |    |
| PT1   | Investigación preliminar                         |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    |    |
| T 1.1 | Conceptos teóricos: PID                          |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    |    |
| T 1.2 | Conceptos teóricos: Péndulo invertido            |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    |    |
| T 1.3 | Prototipo: Características físicas               |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    |    |
| T 1.4 | Prototipo: Componentes internos                  |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    |    |
| H 1   | Finalización del aprendizaje                     |      |    |    | H1 |    |    |    |    |    |     |     |     |      |    |    |    |    |    |    |
| PT2   | Puesta en marcha del robot                       |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    |    |
| T 2.1 | Software controlador de periféricos del Pic      |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    |    |
| T 2.2 | Desarrollo de interfaz LCD y RS232               |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    |    |
| T 2.3 | Programación de motores                          |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    |    |
| T 2.4 | Programación del módulo de sensores (SSC7150)    |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    |    |
| H 2   | Prototipo funcional                              |      |    |    |    |    |    |    |    | H2 |     |     |     |      |    |    |    |    |    |    |
| PT3   | Desarrollo del prototipo                         |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    |    |
| T 3.1 | Implementación de la fusión de sensores          |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    |    |
| T 3.2 | Implementación del PID de equilibrio             |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    |    |
| T 3.3 | Calibración del PID de equilibrio                |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    |    |
| H 3   | Prototipo equilibrado                            |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    | H3 |    |    |
| PT4   | Desarrollo de la cinemática                      |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    |    |
| T 4.1 | Lecturas de codificadores y velocímetro/odómetro |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    |    |
| T 4.2 | Implementación del PID de velocidad              |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    |    |
| T 4.3 | Calibración del PID de velocidad                 |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    |    |
| H 4   | Prototipo controlable                            |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    | H4 |
| PT5   | Desarrollo de procesos de calibrado              |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    |    |
| T 5.1 | Programación del oscilómetro                     |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    |    |
| T 5.2 | Desarrollo y ajuste de calibrador de equilibrio  |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    |    |
| T 5.3 | Desarrollo y ajuste de calibrador de velocidad   |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    |    |
| H 5   | Prototipo final                                  |      |    |    |    |    |    |    |    |    |     |     |     |      |    |    |    |    |    | H5 |

Tabla 2: Planificación final

Los paquetes de trabajo 1 y 2 permanecen idénticos a los originales. Los paquetes de trabajo 3 y 4 de la planificación inicial conservan la idea general, pero la estructura de tareas que los componen cambia. El desarrollo del prototipo pasa a consistir en la implementación de la fusión de sensores y el desarrollo del PID de equilibrio, tomando en consideración también el tiempo que puede llegar a llevar encontrar la calibración ideal. El desarrollo de la cinemática se centra en el PID de velocidad (con su calibración), para el que el velocímetro es esencial.

Además se añade una tercera fase de desarrollo a través del paquete de trabajo 5 para la implementación de procesos de calibrado automáticos, basándose en los procesos de calibrado manuales llevados a cabo en el curso de los paquetes 3 y 4.

# 2

---

## Antecedentes

---

## 2.1 Estado del arte

La tecnología de robots auto-balanceados está muy extendida entre los medios de transporte personales eléctricos y se pueden encontrar infinidad de ellos en el mercado. Entre las múltiples alternativas de transporte personal eléctrico como patinetes, bicicletas y motos eléctricas, esta tecnología ofrece nuevas posibilidades en cuestión de control. El hecho de utilizar solo una o dos ruedas en paralelo, junto con la posibilidad de controlar el vehículo simplemente con la inclinación del cuerpo abre nuevas posibilidades. Permite eliminar cualquier forma de control manual, como el manillar, haciéndolos generalmente más compactos, lo cual es beneficioso no solo para su almacenamiento sino también para su manejo en espacios reducidos o entre viandantes.

Fuera de los modernos diseños de transporte personal, existen pocas instancias en las que se ha implementado la tecnología de auto-balanceo. Cabe notar que existen alternativas a la combinación de Unidades de Medición Inercial (IMU: *Inertial Measurement Unit*), como emplear giroscopios que ayudan a mantener el equilibrio a través de inercia. Este método se puede encontrar en algunos medios de transporte de dos ruedas, pero en este apartado solo se analizan diseños que empleen la misma tecnología que la del prototipo.

### ♦ El *Segway*

El *Segway* es el ejemplo clásico de la fusión de la tecnología de auto-balanceo con el transporte personal. Su base compacta y manillar que se extiende hacia arriba le permiten ser utilizado entre viandantes y en superficies que de otra manera solo serían accesibles para estos. Fue el primer transporte de este tipo en ser comercializado y aunque no consiguió todo el éxito que en un principio se esperaba, ha abierto todo un mundo de posibilidades en cuestión de transporte personal, inspirando numerosos diseños nuevos.



Figura 1: *Segway*

### ♦ El monociclo eléctrico (monociclo auto-balanceado)



Figura 2: Monociclo eléctrico

Este diseño consiste en un vehículo de una sola rueda que se mantiene erguido en la dirección de movimiento. El usuario debe mantenerlo en equilibrio de lado a lado y con la inclinación que este ejerza a través del plano horizontal puede controlar su velocidad y hacer giros. Hay una amplia variedad de gamas y diámetros de rueda que afectan a sus especificaciones, como la distancia y la velocidad máxima que puede alcanzar, pero por lo general son todoterreno y pueden ser utilizados sobre terrenos irregulares y poco firmes como suelos de tierra o gravilla e incluso escalar pequeños obstáculos como escalones. Otra ventaja con

la que cuentan las diferentes variantes de este diseño, por lo general, es que al carecer de manillar y sillín resultan muy compactos y pueden ser guardados en un maletero o en una habitación sin ocupar demasiado espacio.

Una variante extrema de este diseño es el RYNO. Con sus grandes dimensiones parece más una motocicleta de una sola rueda que un monociclo. El usuario se sienta sobre la rueda y maneja el vehículo con un manillar.

#### ♦ El *Hoverboard* (o scooter auto-balanceado)

El *Hoverboard* se asemeja al *segway*, pero sin manillar. Es mucho más compacto y sus ruedas son de un diámetro mucho menor. Es más transportable, pero también reduce su velocidad máxima y la duración de su batería. Además, su baja elevación no le permite moverse sobre terrenos irregulares. Por ello, es habitualmente considerado un juguete antes que un medio de transporte serio.



Figura 3: *Hoverboard*

#### ♦ El *Onewheel*

El *Onewheel* es similar al monociclo auto-balanceado, pero basado en el concepto del skateboard. Consiste en una tabla con una única rueda en el centro. Al igual que el monociclo, solo se mantiene en equilibrio en la dirección en la que avanza y el usuario debe mantenerlo equilibrado de lado a lado. La potencia de sus motores y su ancha rueda permiten utilizarlo incluso sobre arena.



Figura 4: *Onewheel*

#### ♦ Diseños no comerciales:

##### ○ *iBOT*

El *iBOT* es una silla de ruedas capaz de alinear sus ruedas verticalmente y erguirse sobre dos de ellas, elevando la altura del usuario a la misma que la de una persona en pie. Mantiene las dos ruedas de cada lado unidas por un mismo chasis capaz de girar 360° respecto al resto de la silla mientras se mantiene en equilibrio. Esto le permite subir escaleras, alternando entre un par de ruedas y el siguiente.



Figura 5: *iBOT*

##### ○ Vehículos de concepto de Honda



Figura 6: Honda U3-X

El Honda U3-X y el Honda UNI-CUB y UNI-CUB  $\beta$  son tres iteraciones sucesivas de un nuevo concepto de vehículo de transporte personal desarrollado por la marca de automóviles Honda. El primero en ser mostrado públicamente fue el U3-X en 2009. Consiste en un aparato compacto estilo monociclo que se sitúa entre las piernas del usuario y es capaz de moverse omnidireccionalmente. Utiliza la tecnología *Omni-Traction* de Honda que consiste en una serie de pequeñas ruedas alineadas circularmente para formar otra rueda de mayor diámetro. Con las ruedas pequeñas es capaz de desplazarse lateralmente y combinando el movimiento de la rueda grande con el de las pequeñas consigue su característico desplazamiento omnidireccional.

Los modelos UNI-CAB y UNI-CAB  $\beta$  son diseños sucesores del U3-X, presentados entre 2012 y 2013. Mejoran varios aspectos del anterior, convirtiéndolo en un medio de transporte más práctico, con una mayor superficie y sillín que mejoran su ergonomía y una segunda rueda, que también incluye la tecnología *Omni-Traction* de Honda y que se sitúa en posición perpendicular a la principal, permitiendo realizar giros.



Figura 7: Honda UNI-CUB  $\beta$

- **Mesa auto-balanceada [3]**

Un ingeniero japonés ha diseñado una mesa con dos patas y una rueda en el extremo de cada una, capaz de mantenerse en equilibrio y servir mesas en un restaurante. La mesa simplemente se mantiene erguida y se desplaza en la dirección que se le indica a través de un mando a distancia.

- ♦ **Robots balancines**

Como con todas las tecnologías económicas y novedosas, los pequeños robots balancín se han extendido ampliamente por Internet. Diseños virtualmente idénticos al del prototipo de este proyecto pueden encontrarse en infinidad de videos en Internet y pueden adquirirse como kit para montar y programar. El programa suele consistir en simple PID que puede encontrarse en Internet y ser compilado y cargado en el controlador que traiga, una vez se haya terminado la construcción. Además, muchos diseños tienden a concentrar tanto peso como es posible bajo el eje de las ruedas, ayudando a mantener el equilibrio y alejando el modelo físico del robot del péndulo invertido.

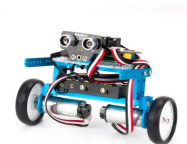


Figura 8: Robot balancín

El prototipo a desarrollar en este proyecto está basado en uno de estos kits. En los apartados 2.2.1: “Diseño original del robot” y 3.1.3: “Desarrollo del diseño ” se estudia más a fondo sus similitudes y qué elementos han sido alterados o sustituidos para hacer funcional el diseño.

## 2.2 Trabajos anteriores

---

El prototipo objeto de estudio y desarrollo de este proyecto tiene su origen en un trabajo previo [1][2]. Este proyecto consistió en la construcción inicial del robot y en el estudio del péndulo invertido para desarrollar una versión inicial del control de equilibrio, control por codificadores y control remoto por adaptador Bluetooth.

Para la implementación del sistema de control se estudió la teoría del péndulo invertido y se implementó un PID en el que la lectura del sensor de inclinación sirve como entrada del mismo y la potencia enviada a los motores como salida. De esta manera, se intenta mantener un ángulo de  $0^\circ$  con la vertical. Para ello, se implementó una fusión de sensores entre giroscopio y acelerómetro similar a la realizada en este proyecto.

A continuación se llevó a cabo la implementación del control a través de codificadores. El objetivo principal era programar un odómetro para utilizarlo en la recuperación de la posición en el caso de una perturbación externa.

La última fase del proyecto consistió en la implementación de un control remoto a través de un adaptador Bluetooth. Aunque idealmente el control se pensaba realizar a través de una aplicación externa con un dispositivo Android o similar, a fin de simplificar, se empleó una Explorer16 con un microcontrolador Pic24 y un módulo Bluetooth idénticos a los del robot. Al disponer de solo cuatro pulsadores para su control, solo se podían manejar cuatro acciones diferentes. Esto permitió simplificar el protocolo de comunicación y las acciones a realizar se reducían a avanzar, retroceder y girar en ambas direcciones a una velocidad continua.



Al final del proyecto se consiguió un control del equilibrio capaz de mantener el robot erguido durante más de diez minutos. Esto se debía, en parte, a que la compensación frente a perturbaciones externas estaba basada entorno a la calibración del PID. La calibración del PID puede escogerse teniendo en mente diferentes objetivos ya que las distintas aplicaciones que requieren de un sistema de control varían en los requisitos que deben satisfacer. Por ejemplo algunas requieren garantizar que no haya *overshoot*. Si se permite realizar oscilaciones amplias al robot, éste pasa más tiempo inclinado en sentido opuesto al del impulso externo, contrarrestándolo. Desgraciadamente, también implica sacrificar la robustez del equilibrio, ya que el robot experimenta oscilaciones muy amplias que son pronas a la desestabilización. También implica una peor respuesta en el estado de equilibrio inestable, ya que reacciona con menor agresividad, causando potencialmente oscilaciones en este estado y un desplazamiento gradual a través de la superficie de apoyo.

Respecto al control por codificadores y control remoto por Bluetooth, tanto el odómetro como el protocolo de comunicación y su traducción en acciones se completó con éxito. Desgraciadamente fue imposible la implementación de una acción única para ambas funciones, la del movimiento rectilíneo. Debido a ello resultó imposible implementar la función de recuperación de la posición y las acciones de avanzar y retroceder para el control remoto.

En el proyecto actual, se busca mejorar el control de equilibrio conseguido en el trabajo previo, cumplir las metas que no se pudieron cumplir, mejorándolas y ampliándolas en lo posible para proporcionar un control de estabilidad y cinemática apropiados para la implementación de aplicaciones como el control remoto previamente propuesto.

A continuación se estudia el diseño del prototipo inicial que dio como producto el proyecto anterior.

## 2.2.1 Diseño original del robot

El primer diseño del prototipo fue desarrollado con la idea de conseguir un robot balancín capaz de sostenerse en equilibrio sobre dos ruedas, basándose en diseños populares como el del *Segway*, *Hoverboards* e infinidad de robots balancín disponibles en el mercado para construir y programar personalmente.

Este primer diseño partió, de hecho, de uno de estos diseños disponibles en el mercado. Consiste en un kit con los materiales básicos para construir y programar un robot balancín completo, con chasis, ruedas, batería, unidades de control y sensores. Todo en él era original, excepto la placa del chasis del nivel superior, que tuvo que ser sustituida por otra más ancha para acomodar al Pic24 sobre la Explorer16, en lugar del Arduino como controlador. Posteriormente se sustituyeron los motores y los puentes en H debido a que los motores iniciales no daban la potencia necesaria y los puentes en H no podían suministrar potencia suficiente a los nuevos motores.

Tras la finalización del proyecto, se produjeron algunos cambios. La unidad de medición inercial original del kit fue más tarde sustituida por una de la marca Microchip. Una pequeña placa con características similares a la original (tres sensores, I2C, etc.) pero con la ventaja de que era más compatible con la Explorer16 y que incluía un chip con lógica de fusión de sensores y otros filtros, proporcionando lecturas precisas sin necesidad de más procesamiento por software.

El estado final del robot, al comenzar este nuevo proyecto, era el descrito hasta el momento, con sus elementos ordenados de la siguiente manera. En la base del chasis, sujetos con escuadras, se encontraban los motores directamente conectados a las ruedas; en el primer nivel del chasis, sobre los motores estaba ubicada la batería. En el segundo nivel, los puentes en H; y en el último nivel, en la parte superior del robot, la placa de prototipado con el microcontrolador.

El módulo de sensores no se encontraba fijado a ninguna parte del robot. En lugar de ello, se conectaba por el Pictail plus, un conector de la placa que proporcionaba una fijación provisional poco fiable. Suficiente para programar un prototipo, pero demasiado susceptible a vibraciones.

En el apartado 3.1.3, “Desarrollo del diseño ” se estudian diversos problemas con el diseño y posibles mejoras que se han llevado a cabo sobre el prototipo durante su desarrollo, llevándolo al estado en que se encuentra actualmente.

# 3

---

## Desarrollo del prototipo

---

## 3.1 Hardware utilizado

---

### 3.1.1 Introducción

La estructura del robot está formada por tres placas de plástico (metacrilato) apiladas verticalmente y separadas por barras metálicas de 5cm. Entre placa y placa se albergan los diferentes componentes electrónicos. A continuación se describen los componentes que forman el robot según su situación en los distintos niveles. Desde la placa inferior hasta la superior (Figura 9).

### 3.1.2 Componentes

#### ♦ Motores: GB37Y3530

El robot lleva dos motores que impulsan las ruedas en la base del robot, van sujetos por dos escuadras en la parte inferior de la primera placa.

Los motores son de corriente continua de 12v, que pueden girar a 251 rpm en vacío. Llevan un codificador con resolución de 64 cuentas por revolución. Según la documentación disponible [4], posee una caja de engranajes incorporada con una relación de 43,7:1. Esto proporcionaría una resolución del codificador de 2797 cuentas por cada vuelta completa del eje. En la práctica se ha comprobado que esto no es cierto y que la relación de la caja de engranajes es de 30:1, por lo que la resolución del codificador es de 1920 cuentas por vuelta completa del eje. El motor es capaz de generar un par de torsión de hasta 18kg a un consumo de 7A.

#### ♦ Batería:

La batería que alimenta el robot es de litio (li-on) recargable de 12v y 5200mA, pesa 310g y ocupa un espacio de 20x60x110 mm. Debido a su tamaño, ocupa una de las placas y no deja espacio para más componentes en dicha placa.

#### ♦ Puente en H

El puente H permite controlar la alimentación del motor, proporcionando hasta  $\pm 24V$  con control PWM. Se utilizan dos puentes en H, uno para cada motor y tienen las siguientes características:

- Voltaje operativo: 6-24 v
- Corriente de salida máxima: 20 A
- Frecuencia del PWM: CD – 20 kHz
- Medidor analógico de corriente de salida
- Dimensiones: 63,5x57,15x12,7 mm
- Peso: 37g

Este componente se añadió como una modificación del diseño original del robot, ya que el puente H original no tenía potencia suficiente para soportar los cambios constantes de polaridad y terminaban sobrecalentándose peligrosamente y sufriendo daños.

♦ **Unidad de procesamiento y control:**

◦ **Microcontrolador: PIC24FJ128GA010** [5]

El controlador empleado en el proyecto pertenece a la familia de los Pic24 de la compañía Microchip®. Esta es una familia de microcontroladores de 16 bits. Concretamente, el modelo FJ128GA010 cuenta con infinidad de periféricos, entre ellos, módulo UART (*Universal Asynchronous Receive & Transmit*), PMP (*Parallel Master Port*), I2C (*Inter-integrated Circuit*), *Output Compare* e *Input Capture*.

El UART es un receptor/transmisor en serie configurable que puede comunicarse a cualquier frecuencia mientras el oscilador lo permita, transmitiendo en paquetes 8 o 9 bits, bits paridad y control de flujo en full-duplex. Se ha utilizado para la transmisión de información y recepción de comandos desde un terminal (un PC) para expandir los controles del robot.

El PMP[6] es un módulo de entrada/salida de 8 bits diseñado para comunicarse con dispositivos transmitiendo datos en paralelo, como LCDs, memorias externas u otros microcontroladores. En el proyecto ha sido únicamente utilizado para controlar el LCD.

El módulo I2C es un dispositivo de comunicación en serie que se emplea para conectar con dispositivos periféricos, como memorias EEPROM, pantallas, sensores, etc. Se ha empleado en la comunicación con las unidades de medición inercial que proporcionan información acerca de la inclinación del robot.

El Output Compare (OC) es un módulo capaz de generar pulsos y trenes de pulsos en coordinación con un temporizador. Se ha empleado para implementar el PWM que controla los motores.

El Input Capture (IC) detecta señales digitales y captura el momento en que esto sucede. Se ha empleado para la lectura de los codificadores de los motores, haciendo posible la implementación del odómetro/velocímetro.

◦ **Periféricos (programación, RS232, LCD, IMU): Explorer16**[7] y **SSC7150**

La Explorer16 es una placa de prototipado diseñada para desarrollar aplicaciones para una gran variedad de familias de microcontroladores de 16 y 32 bits, entre ellas la del Pic24. Incluye infinidad de dispositivos, puertos y conexiones externas para aprovechar todas las entradas y salidas del controlador. También es el soporte principal para la programación del controlador durante el desarrollo de la aplicación. En el proyecto se ha empleado como placa de control del robot, conteniendo todos los dispositivos y sus conexiones.

El LCD[8][9] se ha empleado en combinación con los pulsadores para la navegación a través de menús de opciones para la configuración del robot y para mostrar algunos datos.

El puerto RS232 se ha empleado en la comunicación con el PC para recibir comandos y realizar acciones más complejas de lo que permiten los pulsadores, como la introducción de parámetros. También se ha empleado para mostrar información acerca del estado del robot y para la recopilación de los valores de un conjunto de variables internas del programa con el fin

de poder crear gráficas para mejorar aspectos de la programación y de la calibración del PID (control Proporcional Integral y Derivativo).

El SSC7150[10][11][12][13] (módulo de sensores) es una pequeña placa que contiene tres IMUs repartidas en dos chips, en combinación con un tercer chip que realiza funciones de fusión de sensores, añadiendo funciones de cancelación del desvío en el giroscopio, sensores combinados de 9 ejes, calibración, etc.

### 3.1.3 Desarrollo del diseño

A lo largo del desarrollo del proyecto el diseño del robot ha evolucionado en aspectos sutiles pero que han tenido efectos considerables en el funcionamiento de este. Desde su primera concepción en el proyecto original hasta el diseño actual, hay varios dispositivos y disposiciones que han sido alterados con el fin de mejorar su funcionamiento.

Las características principales del robot, sin embargo, siguen siendo las mismas. Aunque es un robot balancín basado en el modelo físico del péndulo invertido, este tipo de robot no está pensado para transportar personas ni objetos. Es una versión reducida que sirve como prueba de concepto del robot balancín. Con una envergadura de  $19,4 \times 12 \times 20,8 \text{ cm}$  (alto\ fondo\ ancho), es capaz de mantener el equilibrio, recuperarse de empujones, desplazarse y realizar giros, pero apenas puede soportar ningún peso adicional.

#### ♦ Diseño Actual:

El estado del robot al comenzar este proyecto era el descrito en el apartado 2.2.1, “Diseño original del robot”, con sus elementos ordenados de la siguiente manera (Figura 9). En primer nivel del chasis, sujetos con escuadras, se encontraban los motores directamente conectados a las ruedas, sobre este primer nivel del chasis, estaba ubicada la batería. En el segundo nivel, los puentes en H y por último, en la parte superior del robot, la placa Explorer16 con el microcontrolador.

El módulo de sensores no se encontraba fijado a ninguna parte del robot. En lugar de ello se conectaba por el Pictail plus, un conector de la placa que proporcionaba una fijación provisional poco fiable, aunque suficiente para programar un prototipo.

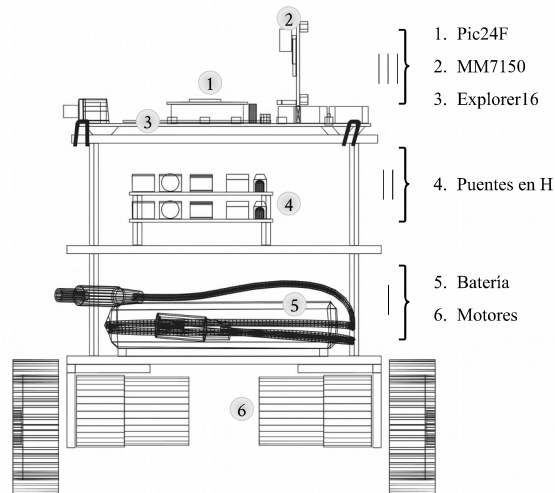


Figura 9: Disposición original de los componentes

La disposición original de los elementos del robot resultaba problemática por dos aspectos. En primer lugar, el peso del robot no estaba repartido de manera óptima y el centro de gravedad se encontraba demasiado bajo. Ya que es imposible tener el centro de masa por debajo de los motores debido a sus características y a que en tal caso el modelo físico dejaría de ser el de un péndulo invertido, la mejor disposición posible tendría el centro de masa lo más alto posible. Esto mejora la controlabilidad del robot, ya que cuanto más elevado esté el centro de masa, más tiempo de recorrido tiene hasta el suelo, ganando por tanto tiempo de reacción y mejorando la estabilidad del sistema. Es por esto que se ha hecho lo posible para elevar el centro de gravedad. Los elementos más pesados son la batería y los motores. Los motores se conectan

directamente con las ruedas a través de su eje y moverlos implicaría modificaciones demasiado complicadas. La batería, en cambio, no importa donde se encuentre. Por ello, se ha movido del primer nivel del chasis al segundo, intercambiándola con los puentes en H.

En segundo lugar, la mala fijación del sensor era problemática, ya que es un componente cuya posición y fijación al resto del robot son claves para obtener mediciones de la inclinación lo más precisas posibles. En su configuración original el sensor estaba casi suelto y podía vibrar y descolocarse fácilmente con las sacudidas que sufre el robot. Ello hacía casi imposible mantener el equilibrio para el robot. Para solucionarlo se ha fijado con dos tonillos a la cara inferior de la tercera placa del chasis, quedando situado en el segundo nivel (junto a la batería).

## 3.2 Diseño del software

---

### 3.2.1 Introducción

Antes de poder desarrollar cualquier aplicación con el robot, hace falta diseñar u obtener el software que controla los dispositivos empleados en éste. Hay una amplia variedad de dispositivos necesarios para hacer funcionar la aplicación. El prototipo utiliza un sensor de inclinación (basado en I2C), un PWM y varias señales de control para la conservación del equilibrio. Para la lectura de los codificadores y con ello ser capaz de controlar la velocidad y la aceleración de las ruedas, se utiliza el *Input Capture*. El resto de periféricos del microcontrolador se utiliza como interfaz para controlar y configurar el robot, comprobar su estado y obtener datos para estudiar su comportamiento. Todo esto sirve principalmente para el desarrollo y *debugging* del software, resultando una parte esencial para estos procesos.

### 3.2.2 Dispositivos programados

Los dos módulos principales del robot son el sensor de inclinación (giroscopio + acelerómetro) y el control de los motores. Entre ellos es suficiente para mantener en equilibrio el robot, su función más importante y básica.

#### 3.2.2.1 SENSORES: I2C, Módulo de sensores y fusión de sensores

El sensor de inclinación proporciona una información imprescindible para el robot. Mide la inclinación del robot en cada momento, permitiéndole calcular la potencia a enviar a los motores para mantener el equilibrio. Da la posición absoluta del chip con respecto al suelo en radianes. Este sensor se forma a partir de la fusión de un giroscopio y de un acelerómetro. En principio, con un giroscopio o un acelerómetro sería suficiente para obtener la inclinación, pero estos dispositivos tienen un error acumulativo e individualmente son inapropiados para una aplicación real.

El giroscopio funciona midiendo la cantidad de giro, en cada eje, por unidad de tiempo. De esta manera, con el sumatorio de cada lectura consecutiva del sensor se puede tener la posición relativa del sensor. Sus desventajas son que sólo da una posición relativa, por lo que cada vez que se inicializa parte de un nuevo centro, según la posición en la que se encuentre. El otro problema es que al tratarse de un sistema físico, tiene unos márgenes de tolerancia, las lecturas no son del todo precisas y se acumula un error por lo que se genera una desviación en la posición relativamente rápida.

El acelerómetro, por otro lado, mide las fuerzas inerciales que experimenta el robot, como sus propios movimientos o la gravedad. La forma de calcular la inclinación con este sensor es observando el efecto de la gravedad, puesto que ésta siempre proviene de la misma dirección. Así que, combinando trigonométricamente la medición de cada eje cómo las componentes de la misma fuerza, es posible obtener el vector de la dirección en que la tierra atrae a la unidad. Usando este vector siempre fijo, es posible calcular la inclinación en términos absolutos, es decir, con el “0°” siempre en la misma posición. La desventaja de este método para obtener la inclinación es que tiene un tiempo de reacción lento y es susceptible a otras fuerzas, que pueden causar interferencias en el cálculo de la inclinación.

Los sensores empleados se encuentran todos en un mismo módulo (placa) de sensores de la marca Microchip, el SSC7150. Este incluye tres tipos de sensor (giroscopio, acelerómetro y magnetómetro) y un coprocesador (chip integrado) de fusión de sensores para obtener datos de sensores virtuales como el inclinómetro tridimensional y sensor de posición *quaternion*. El inclinómetro que ofrece el módulo sería ideal para esta aplicación, pero sólo tiene una precisión de 1°. Tras probarlo se comprobó que esta precisión no alcanza para conseguir suficiente estabilidad en el equilibrio. Por ello, se decidió en las primeras pruebas del PID que habría que implementar un sensor de inclinación con fusión de sensores por software.

La fusión de sensores consiste en la combinación de dos o más sensores en uno solo, con la intención de mejorar la precisión y la fiabilidad de los datos. Se combinan características de unos con las de otros para compensar los errores o desventajas propias de cada uno. Cómo se

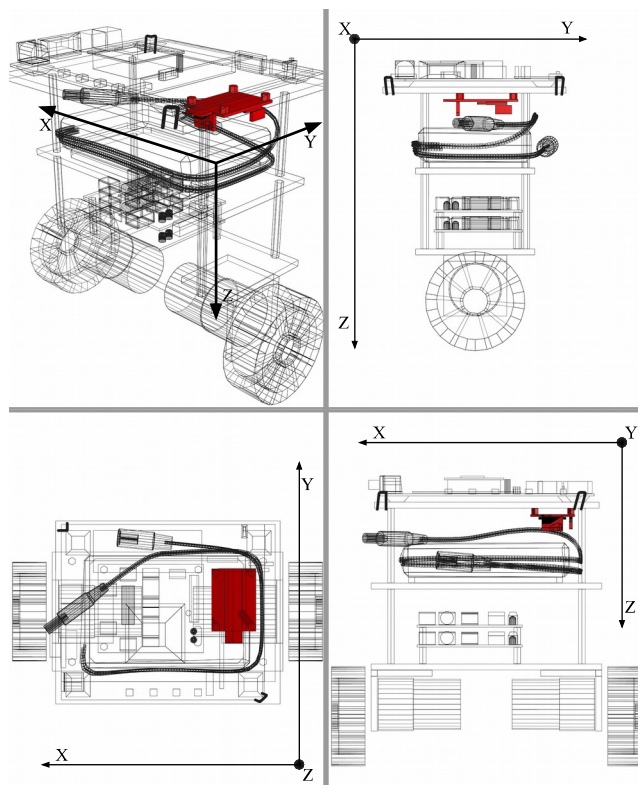


Figura 10: Posición final del sensor en el robot

ha mencionado, el giroscopio es rápido para detectar cambios de inclinación, pero carece de precisión y de la capacidad para determinar su posición respecto al suelo. El acelerómetro, en cambio, calcula su posición con respecto al suelo por lo que puede medir la inclinación con suma precisión, pero es lento y poco fiable hasta que se estabiliza. Con la fusión de sensores se ha conseguido medir la inclinación absoluta del robot con una precisión de 0.1°, aproximadamente.

La técnica más extendida para la fusión de sensores es el filtro de Kalman. No obstante, se trata de una técnica matemáticamente compleja que consume recursos de CPU. Se decidió que estudiarla y comprenderla supondría demasiado tiempo en el desarrollo del proyecto. En lugar de ello se emplea un filtro complementario. Esta es una

técnica simple en la que la información de cada sensor se combina en una suma ponderada, asignándole un peso a cada uno. Al giroscopio se le da la gran mayoría de la influencia en el cálculo de la inclinación, pues reacciona de forma más rápida a los cambios de posición. De esta



manera, el acelerómetro corrige paulatinamente la posición calculada a medida que el giroscopio acumula error.

Para poder combinar los datos de ambos sensores hace falta procesarlos primero por separado ya que deben dar lecturas en las mismas unidades. El módulo de sensores se ha situado en el robot con sus ejes alineados con los motores para que queden en paralelo (X) o perpendicular (Y,Z) con el eje sobre el que éste pivota cuando se balancea, simplificando la interpretación de la información. En las figuras 10 y 11 se pueden ver los ejes del sensor tal y como los indica el fabricante y la alineación de estos respecto al robot. Queda por tanto el eje Z como vertical y los ejes X e Y en el plano horizontal, con el X en paralelo con el eje de las ruedas y el Y en perpendicular.

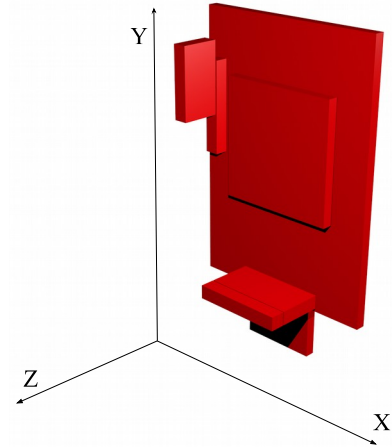


Figura 11: Ejes del módulo de sensores

Del giroscopio interesa tan solo el eje alineado con las ruedas, ya que es el eje sobre el que se pivota. Este es el eje X del sensor. Los demás no proporcionan ninguna información relevante para el giroscopio. Este sensor da en cada lectura la cantidad de cambio en el eje por unidad de tiempo desde la última lectura en grados por segundo. La posición dada por el giroscopio es por tanto:

$$incl = \sum_{n=0}^m lectura_n * \Delta t_n$$

Donde  $\Delta t_n$  es el tiempo en segundos desde la lectura  $n - 1$ .

Del acelerómetro son los ejes vertical y perpendicular al de pivote (Y y Z) del robot los que dan la información relevante, debido a que estos ejes definen el plano sobre el que se reparte la influencia de la gravedad en todo momento. Si se alinea un único eje del módulo en perpendicular con el suelo como, por ejemplo, el Z en la Figura 10, y se deja en reposo sin influencias externas, se obtendría (aproximadamente) -1 como valor de la lectura en dicho eje y 0 en el resto. Si se sitúa el eje a un ángulo  $\alpha$  con respecto al suelo y manteniendo uno de los otros en paralelo al suelo (como el X) como en la Figura 12, la magnitud de la lectura se reparte entre los dos ejes no paralelos al suelo de tal manera que:

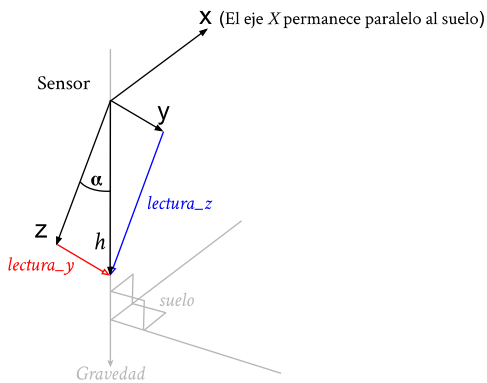


Figura 12: Obtención del ángulo del acelerómetro

$$lectura_y = \sin(\alpha) \cdot h \text{ y } lectura_z = \cos(\alpha) \cdot h$$

Una vez establecido esto, se puede calcular la posición del sensor por:

$$\arcsen(lectura_y/h)$$

Donde la  $h$  es la hipotenusa del triángulo rectángulo formado por  $lectura_y$  y  $lectura_z$  (Figura 12). El ángulo se calcula con respecto al eje Z ya que este queda alineado verticalmente respecto al suelo cuando el robot se encuentra en equilibrio.

La siguiente gráfica muestra la posición medida por cada sensor por separado y la medida tras la fusión de sensores.

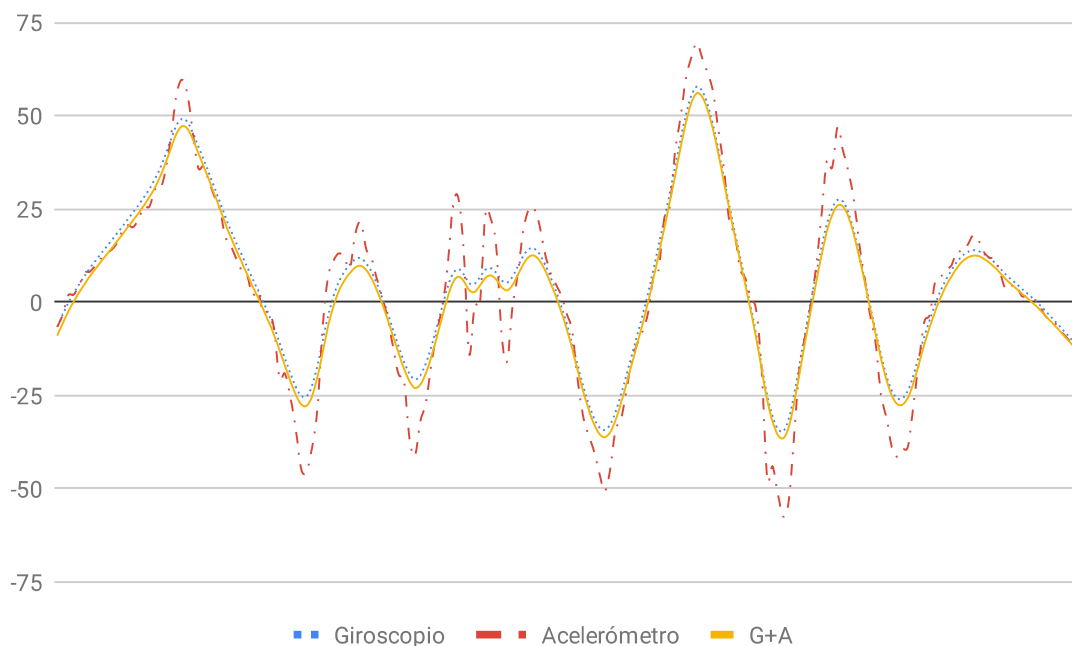


Figura 13: Demostración de la fusión de sensores

### 3.2.2.2 PWM y Motores

El PWM es el método más empleado para controlar motores, leds, servos, etc. El Pic24F cuenta con cinco módulos de *Output Compare*[14] (OC). Estos módulos pueden comparar el valor de un temporizador con el de un registro para mandar un pulso o un tren de pulsos en el momento indicado y puede configurarse para operar como PWM. Envía una señal de alrededor de +5V a 5mA, insuficiente para los motores que funcionan a 12V y pueden llegar a consumir 7A. Por ello el OC se utiliza para generar una señal de control que se envía a los puentes en H para su amplificación.

Los puentes en H están directamente conectados a la batería para aprovechar toda su potencia. Estos repiten las señales generadas por los OC, pero a 12V y con una corriente máxima de 20A, suficiente para cada motor. Cada puente en H posee dos generadores de PWM y dos salidas para la señal y se han configurado para que cada generador envíe una corriente opuesta a la del otro. De esta forma los motores pueden ser movidos en un sentido u otro selectivamente. Para poder controlar cada uno por separado se han empleado dos puentes en H. Así se necesitan 4 señales de PWM diferentes, enviadas de dos en dos a cada puente H para controlar independientemente ambos motores.

La configuración del PWM es crucial para que los motores tengan un rendimiento adecuado. Concretamente, la frecuencia a la que opere es muy importante, puesto que si no es lo suficientemente alta, el rendimiento de los motores es malo y no responden con toda la rapidez que podrían a los niveles de energía que se les envía. Además, se puede producir una fuerte vibración en los motores que contribuye en gran parte a la inestabilidad. Por lo comprobado empíricamente, cuanto mayor sea la frecuencia del PWM mejor es el rendimiento de los

motores, por lo que se ha configurado a lo más alto permitido por los puentes H y los compromisos del uso compartido de algunos periféricos. La máxima frecuencia soportada por los puentes H es de 20kHz y la frecuencia a la que opera el oscilador del microcontrolador es de 16MHz (frecuencia de instrucción). El OC se utiliza en conjunto con uno de los cinco temporizadores que incluye el microcontrolador, aunque solo es posible escoger dos de ellos. Los temporizadores disponibles son el 2 y el 3. El n°2 se emplea en el protocolo de comunicación por I2C con el módulo de sensores para control de errores, por lo que no está disponible para funcionar continuamente ni es posible configurarlo con libertad. Esto deja al n°3 como la única opción para controlar los OCs. La máxima frecuencia a la que podría operar un temporizador cualquiera del robot es cercana a la de instrucción, ochocientas veces más rápido que el máximo de los puentes en H. El T3 ha sido empleado también para el módulo *Input Capture* (IC). Debido a características del funcionamiento de este módulo, se ha necesitado escoger un período que fuera potencia de dos. La potencia de dos más cercana al límite impuesto por el puente en H es 10, un período de 1024 cuentas. Dado que el período de instrucción es de  $\frac{1}{1,6 e^7}$  s, la frecuencia del temporizador queda en:

$$1024 \cdot \frac{1}{1,6 e^7} = 0,000064 \text{ s} = 15.625 \text{ Hz}$$

Esta es una frecuencia alta y bastante ajustada al límite de 20kHz. Con esta configuración los motores funcionan con gran eficiencia. La vibración se transforma en un zumbido agudo y ligero que no afecta a la estabilidad y los motores responden suave y rápidamente a la potencia transmitida. Conseguir que los motores sean eficientes es de gran importancia, puesto que, para mantener el equilibrio, el robot ha de reaccionar con mucha precisión y rapidez.

### 3.2.2.3 *Input Capture*[15] y Codificadores

Los codificadores son el elemento clave de las funciones avanzadas del robot. Con los codificadores y un control de equilibrio robusto es posible controlar el movimiento del robot y conseguir estabilidad en planos inclinados. La forma en que se han empleado los codificadores es programando un odómetro/velocímetro, a través del módulo *Input Capture* (IC).

El codificador que se encuentra en cada motor es incremental con resolución de dos bits en código Gray. A pesar de la resolución de dos bits, solo se ha utilizado un bit de cada codificador, puesto que implementar el velocímetro con los dos bits es demasiado complejo. Además, para ahorrar en tiempo de cómputo, solo se capturan los flancos de subida del bit. Con solo el flanco de subida de un bit la resolución del codificador desciende de 1920 a 480 y se pierde la capacidad de distinguir el sentido de la rotación. Esto no es un problema, ya que 480 es suficiente resolución y el sentido de la rotación es conocido en todo momento puesto que lo impone el programa.

El IC es un módulo con una entrada digital que captura el momento en el que detecta un flanco de subida o de bajada en dicha entrada. Para ello necesita trabajar en conjunto con un temporizador. Igual que en el caso del OC, solo puede emplear los temporizadores n° 2 y 3, y puesto que el n°2 está ocupado, solo queda el n°3. Afortunadamente el T3 funciona en un modo compatible tanto con el IC como con el OC.

El IC ha sido configurado para capturar cada flanco de subida del bit1 del codificador. Cada vez que detecta el flanco guarda el tiempo en ese momento y el del inmediatamente anterior detectado. Con la diferencia de tiempo entre ambos, es posible calcular la velocidad del giro. Como se ha mencionado anteriormente, el período del temporizador utilizado en el IC es de 1024 cuentas (a 16MHz, 1024 cuentas equivalen a 0.064 ms). Esto deja muy poco margen para medir antes de que cada iteración del temporizador termine, abriendo la posibilidad de que un tiempo medido sea menor que el anterior, lo cual daría un valor de tiempo equivocado. Ya que incrementar el periodo del temporizador no es una opción, la solución para este problema ha sido construir un contador por software. Cada vez que el ciclo del temporizador se completa, una variable de tipo entero sin signo de 32 bits es incrementada por valor de 1024. Esta variable puede crecer hasta un tamaño de más de  $4,2e^9$  lo que significa un período de 4,47 minutos. Cada vez que se hace una captura, se almacena el valor del temporizador sumado al del contador por software, dejando un margen amplio para calcular la diferencia de tiempo entre una captura y la anterior. De todas formas, este sistema no es perfecto ya que la resolución extra se añade en la rutina de atención a la interrupción que tiene lugar en cada captura. La rutina de atención es lo más ligera posible y la suma se hace en pocos ciclos, pero existe un pequeño margen en el que el contador software puede actualizarse antes de guardar su valor adecuado, en cuyo caso el valor almacenado podría exceder en 1024 al valor correcto.

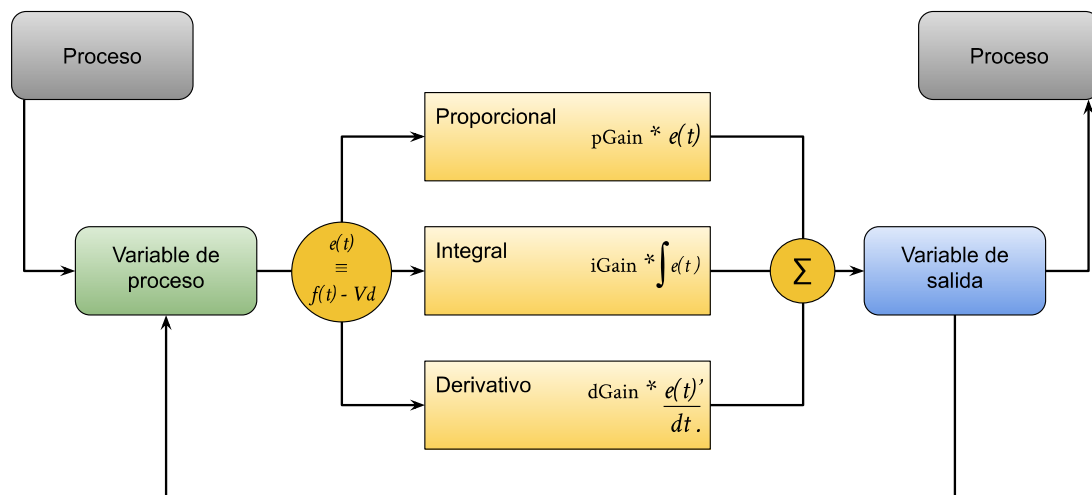
## 3.2.3 Sistema de control: PID

### 3.2.3.1 Introducción

Los principios en los que se basa el control PID son conocidos y la literatura es extensa. Por ello, se hace una introducción con la información básica acerca de su funcionamiento. Para más detalles consultar las referencias [16] y [17].

El PID es un tipo de sistema de control que mantiene o aproxima el valor de una variable a un valor deseado en un proceso como, por ejemplo, la temperatura en una estancia. Se basa en un bucle que reacciona al estado de la variable escogida produciendo una respuesta para compensarlo. Más concretamente, funciona teniendo en cuenta la variable que se desea controlar, llamada variable de proceso (PV), y aplicando un cambio a la salida del proceso en proporción a la diferencia (o error) entre la PV y el valor deseado (o *set point* (SP)). En el ejemplo del robot balancín, para un PID que controle el equilibrio, PV es la inclinación del robot, SP corresponde a la inclinación en la que el robot se encuentre en posición de equilibrio y la salida del proceso es la potencia que se envía a las ruedas

La magnitud de la salida del proceso se calcula concretamente en proporción directa, integral y derivativa del error. De ahí el nombre PID. Estos son los ‘términos’ del PID y su influencia se determina por una constante de ganancia correspondiente a cada uno de ellos. Si el comportamiento de un sistema o proceso se corresponde con una función  $f(x)$ , el término proporcional se corresponde con  $f(x)$ , el término integral se corresponde con  $\int_{-\infty}^{\infty} f(x)$  y el término derivativo se corresponde con  $f'(x)$ .



\*Vd = valor deseado

Figura 14: Esquema del proceso del PID

Este es el PID descrito según la teoría y que se define por la siguiente fórmula:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt}$$

Esta es conocida como la forma paralela, puesto que cada termino se ve afectado individualmente por su propio valor de ganancia. Calibrar este tipo de PID es poco intuitivo y a menudo complicado y por ello se reserva para desarrollos teóricos. En aplicaciones industriales y comerciales se emplean otras variantes del PID, tales como la forma *estándar*. Esta es, de hecho, la más utilizada en algoritmos de calibración, pues la mayoría se basan o son compatibles con ella. Se define por la siguiente fórmula:

$$u(t) = K_p \left( e(t) + \frac{1}{T_i} \int_0^t e(\tau) d\tau + T_d \frac{de(t)}{dt} \right)$$

Donde  $T_i$  es el tiempo integral y  $T_d$  es el tiempo derivativo.

A pesar de lo comentado, la forma paralela resulta más apropiada para la aplicación de este proyecto. Basandose en la implementación de PID utilizada en otros proyectos estudiados, se ha considerado que la más adecuada para este proyecto es la paralela. No solo porque es el ejemplo más abundante y funciona correctamente, sino porque para el caso del robot balancín es fácil de calibrar manualmente y porque estudiar la forma estándar suponía demasiado consumo de tiempo en el desarrollo del proyecto.

### 3.2.3.2 Implementación[18]

La realización del control por PID en la práctica difiere en varios aspectos de la teoría debido a las limitaciones inherentes de un sistema físico real. Los aspectos que limitan la implementación del PID son varios y tienen su origen en diferentes ámbitos del sistema.

La implementación del PID ha sido inspirada por varios algoritmos y trabajos disponibles en el dominio público. Las referencias están incluidas en la bibliografía.

♦ **f(x)**

El primer problema que surge es la definición de  $f(x)$ . En el esquema teórico del PID  $f(x)$  representa el valor de la variable de proceso a lo largo del tiempo. En el modelo físico esta variable de proceso se corresponde con la inclinación del péndulo invertido. Puesto que el proceso es un sistema real modelado por un sistema físico inestable, el valor que puede adoptar en un momento dado es impredecible o, en el mejor de los casos, aproximable pero no definible por una función matemática simple ni conocida. Por ello, el cálculo de los términos Integral y Derivativo no es tan simple como la obtención de la integral o la derivada de esta función matemática.

En lugar de ello, para obtener el término Integral se emplea integración rectangular[19]. A medida que avanza el tiempo y se obtienen lecturas del módulo de sensores, se calcula el error por cada una y se suma en una variable (*errOT*, *error over time* o error en el tiempo). En principio, cada uno de estos valores de error debe ser expresado en función del momento en el que ha sido obtenido  $\sum \frac{err}{\Delta t}$ , pero ya que es posible ejecutar el

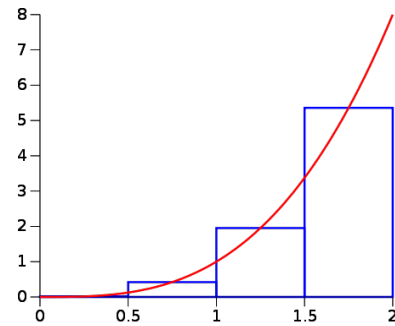


Figura 15: Integración rectangular

algoritmo en un período regular de tiempo, haciendo que  $\Delta t$  sea constante, esta operación puede ser omitida. La suma del error en el tiempo es, pues, la integral del error.

Para obtener el término derivativo se emplea la aproximación a través del cociente diferencial de Newton. Este dice que dados dos valores “ $y$ ” e “ $y'$ ” de una función  $f(x)$  que se correspondan con una “ $x$ ” y “ $x+h$ ” tal que “ $h$ ” que sea lo bastante pequeña, entonces la fórmula:

$$\frac{f(x+h)-f(x)}{(x+h)-x}$$

devolverá un valor aproximado a la tendencia de cambio de la “ $y$ ” producida por  $f(x)$ . Cuanto menor sea el valor de  $h$ , más precisa es la aproximación. Afortunadamente el proceso del PID se puede repetir a un ritmo de aproximadamente 200 Hz. Esto hace posible la implementación del término integral según la siguiente fórmula:

$$de(err) = \frac{err_t + err_{t-1}}{\Delta t}$$

Donde “ $err_t$ ” y “ $err_{t-1}$ ” son el error medido en una iteración (o momento de tiempo concreto) y el error calculado en la iteración anterior y “ $\Delta t$ ” es el período de tiempo transcurrido entre estas dos.

De lo anteriormente dicho también se puede deducir que cuanto mayor sea la frecuencia a la que se ejecuta el algoritmo, mayor precisión tendrá el término derivativo. Si bien a partir de un determinado umbral, el aumento de la frecuencia no mejora los resultados.

#### ♦ **Windup integral**

Es un fenómeno que se da cuando se produce una cantidad de error muy grande en un período de tiempo corto. En un sistema ideal, no se contempla ninguna limitación en la magnitud de la salida del proceso. En un sistema físico real, en cambio, la respuesta que se puede esperar de éste es evidentemente limitada. Por ejemplo, en el robot balancín la respuesta al desequilibrio la producen los motores al acelerar las ruedas y también a través del par de torsión que produce esta aceleración. Puede suceder que se produzca un gran error y los motores no puedan producir tanta potencia como el algoritmo calcula que necesita. A consecuencia de esto el error integral no se reduce tanto como debería, lo cual tiene como resultado un *overshoot* (exceso) en la salida del proceso. Esto causa oscilaciones en la variable PV y puede suponer una pérdida total de la estabilidad.

La solución a este problema es el llamado Anti-*Windup*, que es una parte del algoritmo que limita el valor máximo que puede alcanzar el valor del término integral, para que no crezca hasta proporciones descontroladas.

#### ♦ **Muestreo periódico**

En la teoría, los términos integral y derivativo se calculan teniendo en cuenta la diferencia de tiempo entre una muestra y la anterior. Esto añade una operación más para el cálculo de estos términos por cada iteración, que en coma flotante tiene un costo considerable y merece la pena ser eliminado. Si se mantiene un período de muestreo constante se pueden eliminar estas dos operaciones, pues se vuelven un valor constante que se puede calcular de antemano multiplicándolo por la correspondiente constante de ganancia.

#### ♦ **Derivada de la variable de proceso**

En el caso de un cambio brusco en la variable SP, el término derivativo puede generar un pico en la salida debido al repentino cambio en el error. Para mitigar este efecto se pasa de calcular la derivada del error a calcular la derivada de PV, puesto que ambas derivadas son iguales, pero la segunda se mantiene constante, a pesar de los cambios en el SP.

#### ♦ **Cambios de calibración**

Debido a un cambio brusco en la calibración, el término integral puede generar un gran cambio en la salida. Para solucionarlo, el término integral debe ser reiniciado en cada cambio. Para no perder el valor del término integral cada vez que su ganancia cambia, se modifica la forma en que este término es calculado. En lugar de multiplicar su valor por  $K_i$  en la fórmula del PID, se multiplica por el valor del error al calcular el valor del término integral.

Para el control del robot, se han implementado dos PID's diferentes, en una configuración denominada PID en cascada. El primer PID es el de equilibrio, basado en la inclinación y el control de las ruedas. El segundo es el PID de velocidad, basado en la velocidad de las ruedas y en la manipulación de la inclinación (a través de la modificación del SP).

### 3.2.3.3 PID de equilibrio

El PID de equilibrio es el sistema de control que mantiene el robot erguido. En este apartado se muestran las particularidades que tiene con respecto a la implementación genérica previamente explicada.

En este PID la PV es la inclinación del robot en grados, el SP es la inclinación en la que el robot se encuentra en posición de equilibrio y la salida del proceso es la potencia enviada en tanto por mil. El sistema de control busca por tanto aproximar el error entre la posición actual del robot y su posición ideal a través de la aceleración de la base y el par de torsión generado por los motores.

Puesto que el péndulo invertido es un modelo muy inestable, el PID debe trabajar a una frecuencia de muestreo alta, realizando pequeñas correcciones en con rapidez, antes de que se produzcan pérdidas mayores en el equilibrio. Para ello, la estructura del programa interno del robot se ha diseñado entorno a la idea de alcanzar la mayor frecuencia de operación posible. El sistema de coordinación del PID y el diseño del bucle principal se estudiarán a fondo en el apartado 3.2.5, "Aspectos misceláneos de la programación". Debido a los compromisos con el sistema de sincronización del PID, ha sido imposible mantener un período regular entre muestreo y muestreo, por lo que para el cómputo de las componentes integral y derivativa se ha tenido que medir la diferencia de tiempo. La fórmulas resultantes para cada término, incluyendo las modificaciones mencionadas en el apartado anterior, son las siguientes:

$$error = \theta_n - \alpha$$

$$integral = K_i \cdot error \cdot \Delta t$$

$$derivada = (\theta_n - \theta_{n-1}) / \Delta t$$

Donde  $\theta_n$  es el ángulo de inclinación en cada momento en grados,  $\alpha$  es la inclinación deseada y  $\Delta t$  es la diferencia de tiempo entre la muestra actual y la anterior.

Elevar la frecuencia de muestreo no solo permite realizar correcciones antes de que el error sea demasiado grande, también permite incrementar la magnitud de las constantes de ganancia. Esto hace que el control sea más robusto. Si se usa una configuración de alta ganancia a una frecuencia de muestreo baja puede producirse inestabilidad en forma de pequeñas oscilaciones de alta frecuencia. Asimismo, un elemento importante para operar a mayores frecuencias es la precisión de los sensores. Si el sensor no proporciona una lectura lo bastante precisa, el sistema reacciona ante perturbaciones que no son reales, causando también inestabilidad.

El otro elemento particular del PID de equilibrio es lo que se ha bautizado como el diferencial. Este mecanismo reparte la potencia calculada por el PID entre ambos motores, dando la capacidad de realizar giros graduales o hasta rotar sobre una rueda sin dejar de mantener el equilibrio. En principio, cuando el PID calcula una salida para los motores, el valor obtenido se asigna a ambas ruedas por igual. Este sistema consiste en el reparto de un porcentaje



complementario entre los dos motores. De esta manera envía la misma potencia total a ambos motores con un reparto equitativo. Esto implica que cuanto más rápido se esté desplazando éste, el giro que podrá realizar será menor, ya que una de las ruedas tendrá que compensar el impulso que le falte a la otra.

Funciona de la siguiente manera. Con un valor entre -100 y 100, asigna la salida del proceso en proporción a éste a cada rueda según las fórmulas:

$$MotorDerecho = B_{Pout} (differential) / 100$$

para el motor derecho y

$$MotorIzquierdo = B_{Pout} - (MotorDerecho - B_{Pout})$$

para el izquierdo.

Donde  $B_{Pout}$  es la señal de salida del PID.

Por ejemplo, con el valor 0 asigna la salida en proporción 50%-50%, con el valor 50 asigna la salida en proporción 25%-75% y con el valor 100 asigna la salida en proporción 0%-100%. Así, el PID sigue regulando el equilibrio mientras se realizan giros.

### 3.2.3.4 PID de velocidad

Una vez calculado el PID de equilibrio, se calcula el PID de velocidad, que regula la velocidad de las ruedas. Éste es más parecido al estándar y no incluye mecanismos internos especiales.

Este PID es lo que se denomina un PID en cascada, es decir, un PID que como variable de salida tiene el *set point* (valor deseado) de otro PID. Se les denomina PID externo e interno. El externo controla directamente una variable física como la velocidad, mientras que el interno toma la salida del externo como valor deseado. En el caso del prototipo, el controlador externo es el control de velocidad, y marca la inclinación deseada para el PID de equilibrio. Mientras el controlador interno se ajusta rápidamente al valor deseado indicado en cada momento, el controlador externo ajusta su valor deseado más lentamente hasta que los codificadores miden la velocidad deseada.

En este PID la variable del proceso (PV) es la velocidad de las ruedas en cm/s, el valor deseado es la velocidad indicada (habitualmente 0 cm/s) y la salida del proceso es el desfase en grados sobre la posición de equilibrio del robot. El sistema de control busca por tanto aproximar el error entre la velocidad actual de las ruedas y la velocidad deseada a través de la inclinación del robot.

La salida producida por este sistema de control no se aplica directamente al *SP* del PID de equilibrio, sino que se suma como un *offset* de este. La fórmula del término proporcional del control de equilibrio queda por tanto:

$$error = \theta_n - (\alpha + \alpha')$$

Donde  $\alpha$  sigue siendo el *set point* del PID de equilibrio y  $\alpha'$  es la variable de salida del PID de velocidad.

Las fórmulas de los términos del PID de velocidad son las siguientes:

$$error = v_n - V$$

$$integral = K_i \cdot error \cdot \Delta t$$

$$derivada = (v_n - v_{n-1}) / \Delta t$$

Donde  $v_n$  es la velocidad de las ruedas en la muestra  $n$  y  $V$  es la velocidad deseada.

#### ♦ El término integral en el PID de velocidad

Dado que este PID mide la velocidad para controlarla, su término integral representa la posición del robot, es decir, mide la distancia recorrida. Esto significa que el PID recuerda automáticamente la distancia que recorre desde su posición inicial, lo que tiene como resultado que el robot se desplace en dirección opuesta al movimiento inicial, hasta reducir su término integral 0. Esta característica es clave para la función de recuperación de la posición y para el equilibrio en planos inclinados y cambios del centro de masa.

## 3.2.4 Calibración del PID

### 3.2.4.1 Distintos métodos

La calibración del PID es el proceso que consiste en escoger el valor de las constantes de ganancia de cada término. Según la proporción de cada una de ellas y sus magnitudes, el PID adopta diversos comportamientos. Es decir, puede tener una respuesta más rápida o más lenta, un control más robusto que absorbe perturbaciones o alcanzar el valor deseado sin nada de *overshoot*.

La calibración del PID puede resultar complicada puesto que no hay una única configuración válida para cada sistema y cada uno tiene sus propias características. A pesar de ello, existen varios métodos de calibración que ayudan a obtener dichos resultados. Métodos como el Ziegler–Nichols, el Cohen-Coon o el Lambda pueden conseguir este tipo de resultados, pero están diseñados para variantes de PID diferentes a la escogida para el robot.

El método Ziegler–Nichols[20] fue introducido en los años 40 para PIDs en serie interactivos y puede obtener una configuración para controles de tipo P, PI, PD y PID con diferentes niveles de *overshoot*. Consiste en bajar a cero  $T_i$  y  $T_d$  y buscar un valor de  $K_p$  en el que la salida del PID oscila con estabilidad al que se llama  $K_u$ . Sobre el valor de este parámetro se aplica un conjunto de reglas matemáticas para hallar los valores de  $T_i$ ,  $T_d$ ,  $K_i$  y  $K_d$ .

Los métodos Cohen-Coon[21] y Lambda fueron introducidos en los años 50 para PIDs en serie no interactivos o estándar. Se basan en la  $K_p$ , el ‘*dead time*’ y una ‘constante de tiempo’. Estas dos últimas son el retardo con el que la señal de salida enviada tarda en afectar al PV y la velocidad a la que el PV responde a cambios en la salida del proceso. En base a estos parámetros se calculan los valores para el resto de constantes utilizando su propio conjunto de reglas matemáticas.

### 3.2.4.2 Método manual

Los parámetros del PID son complicados de calcular con la precisión necesaria y las reglas se aplican a variantes del PID diferentes a la escogida. Por ello se ha llevado a cabo la calibración inicial a través de prueba y error y de la observación del comportamiento del error del proceso,

es decir, en cómo el robot se balancea y no en como oscila la señal de salida, que sería el valor que se le pasa al PWM.

El método de calibrado empleado se basa en el método manual de calibración, apropiado para el PID paralelo. Aunque no es lo más utilizado en sistemas reales y se reserva para el estudio teórico, se ha considerado apropiada la variante paralela para el robot. De hecho, analizando otros trabajos disponibles en la red, es la variante que siempre se emplea en robots equilibristas.

El método manual consiste en dejar a cero todas las constantes de ganancia e incrementarlas una a una en el orden  $K_p, K_i$  y  $K_d$  hasta obtener el resultado deseado. El método dice que se ha de incrementar la primera variable hasta que la salida del proceso oscile. Tras esto se recomienda reducir el valor de ésta aproximadamente a la mitad. A continuación se debe incrementar la  $K_i$  hasta que el error en régimen estacionario se reduzca lo suficientemente rápido, pero cuidando que no genere demasiado *overshoot*, pues desestabiliza rápidamente el proceso. Finalmente se ha de incrementar la  $K_d$  hasta que reduzca suficientemente el tiempo en que la PV se estabiliza sobre el valor deseado.

Este es el método que se ha empleado para calibrar el robot manualmente con la única diferencia de que se ha descubierto que es mejor basarse en las oscilaciones del error del PV antes que en las de la salida del proceso. De esta manera se ha podido estimar visualmente los valores en los que se producen las oscilaciones.

#### **3.2.4.3 Calibrador automático**

El método automático es una reproducción del proceso manual. Itera a través de las constantes de ganancia incrementándolas gradualmente hasta alcanzar los valores óptimos. Cabe notar que el criterio principal para determinar cuando se alcanza un valor óptimo en el proceso teórico es la oscilación de la salida del proceso. En lugar de ello se ha comprobado que es más fácil observar la oscilación del error de PV al igual que en el procedimiento manual. La proporción entre la frecuencia y la amplitud de la oscilación del error está estrechamente relacionada con los valores óptimos de la ganancia. Para realizar el análisis de las oscilaciones se ha tenido que desarrollar un programa que realiza un análisis básico de la señal midiendo la frecuencia, amplitud y valor medio de la señal. Así se ha podido recabar la información necesaria para buscar la relación entre la frecuencia y la amplitud.

Al comienzo se ha escogido un umbral para la amplitud y la frecuencia arbitrarios en los que aparentemente las oscilaciones podrían ser lo bastante estables y amplias. Después se han ajustando estos dos umbrales hasta que el calibrador automático escoge un valor aproximadamente similar al obtenido en la prueba manual. Al hacer esto, se ha comprobado que aproximadamente en el momento en el que la amplitud se vuelve mayor que la frecuencia la ganancia alcanza un valor muy similar a  $2 \cdot K_p$ , precisamente la meta del primer paso del método manual.

Esto se debe a que la velocidad de los motores es limitada y las oscilaciones no pueden pasar de cierta frecuencia mientras se mantiene una gran amplitud. Se establece como condición para estar oscilando, que la frecuencia de oscilación supere la amplitud del movimiento como se puede apreciar en la Figura 16.

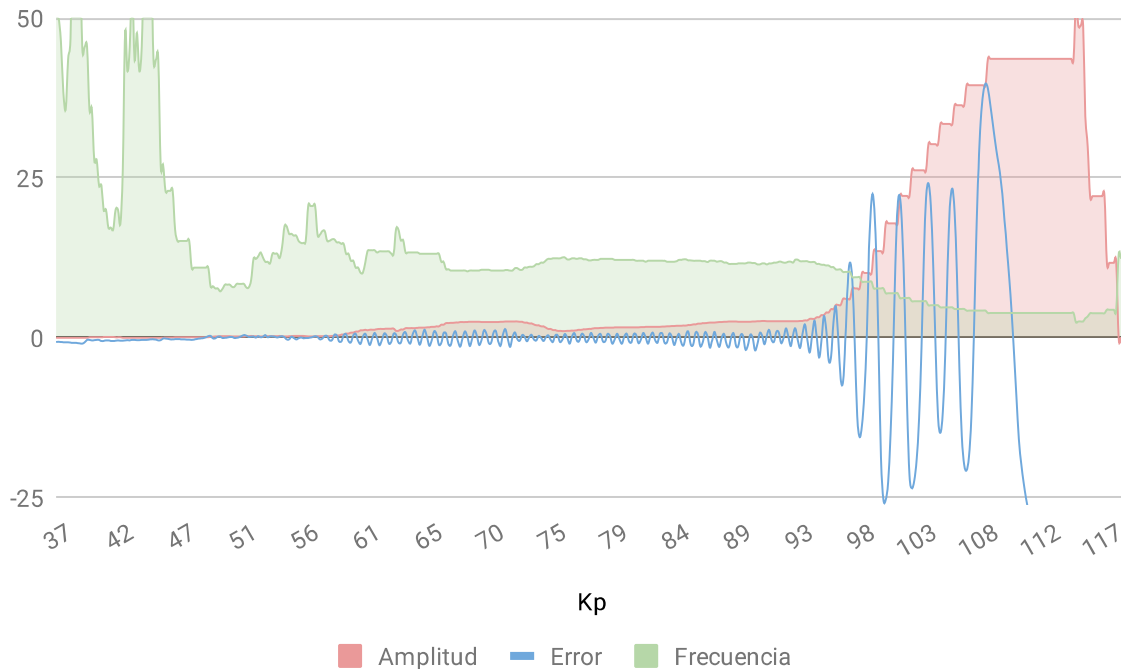


Figura 16: Oscilaciones del robot superando la frecuencia en amplitud

Este criterio se puede volver a emplear para ajustar el término integral puesto que el criterio para el valor óptimo es similar. Se trata de incrementar  $K_i$  hasta que el bucle de control pierde la estabilidad.

El término derivativo, en cambio, no es compatible con este criterio ya que no se trata de oscilaciones de gran amplitud. En su lugar hay que buscar oscilaciones rápidas y de baja amplitud. Cuando el término derivativo es demasiado grande comienza a causar nuevamente inestabilidad en el sistema. Para detectar este tipo de oscilación se analiza la proporción entre la amplitud y la frecuencia. El oscilómetro no es muy preciso y solamente da información de suficiente fiabilidad cuando las oscilaciones son consistentes. A causa de esto, no hay que fiarse de valores altos de frecuencia si no van acompañados de cierta consistencia en la amplitud. Sin embargo como se busca detectar oscilaciones de alta frecuencia y baja amplitud no se puede esperar que la amplitud supere a la frecuencia como en los dos casos anteriores. No obstante, hay otra forma de detectarlas.

Cuando el error no oscila consistentemente, se miden frecuencias muy altas que varían drásticamente en poco tiempo. Igualmente la amplitud carece de toda consistencia, pero se mantiene en valores menores a uno. En cambio, cuando sí oscila de manera consistente, la frecuencia se estabiliza en un valor alto pero más moderado y la amplitud es mayor que uno. Esta relación característica de una oscilación rápida y corta se ve reflejada en el cociente entre la frecuencia y la amplitud. Cuando el cociente es menor que la frecuencia se considera que la oscilación es consistente. Escogiendo una frecuencia mínima y un umbral mínimo para este cociente podemos ajustar las condiciones en las que la ganancia derivativa es óptima.

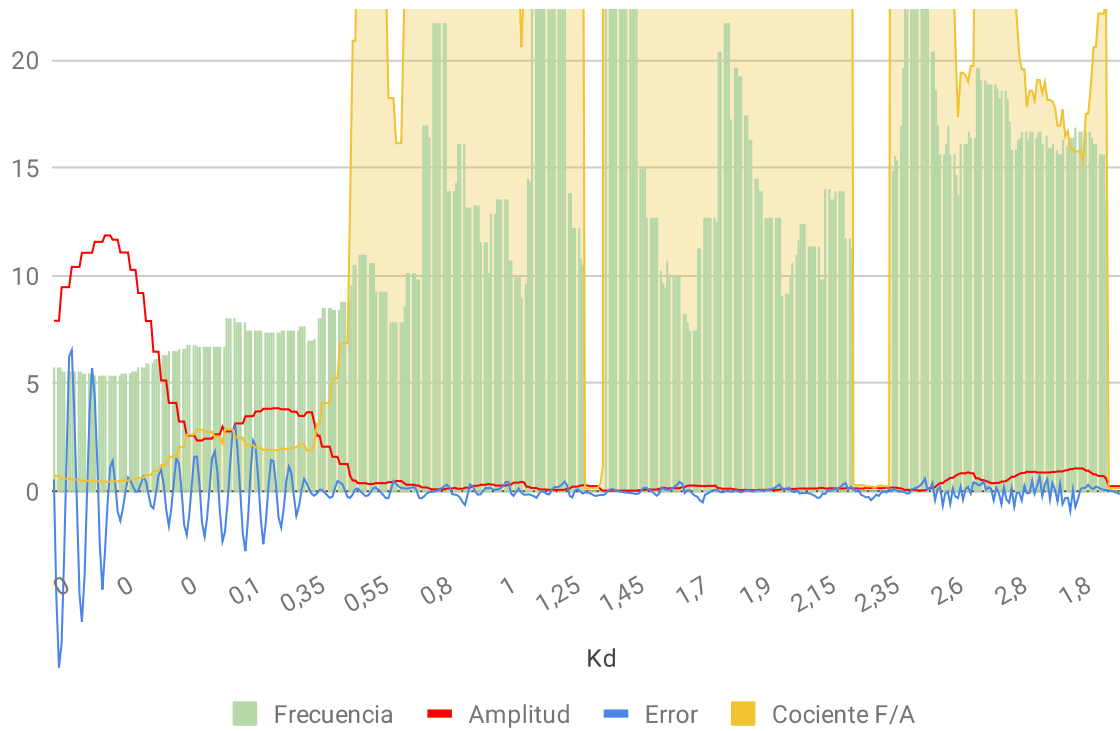


Figura 17: Estabilización de las oscilaciones en altas frecuencias debido al efecto de una alta ganancia en el término derivativo

En la Figura 12 se puede apreciar cómo en la parte final de la gráfica, entre la reducción de  $K_d$  de 2,8 a 1,8, el error oscila a gran velocidad, la amplitud supera el 1, ganando estabilidad, y la frecuencia se estabiliza superando por primera vez al cociente F/A. En este momento se guarda la tercera parte del valor actual de  $K_d$ .

## 3.2.5 Aspectos misceláneos de la programación

### ♦ Bucle principal

El bucle principal es el núcleo del programa. Desde aquí se controlan todas las acciones que realiza el programa. Tras ejecutar la secuencia de inicialización, el programa entra en el bucle hasta que se reinicia el robot. A través del temporizador nº4 (T4) se controla la frecuencia a la que se repite este bucle. Cuando se termina una iteración se espera a que este temporizador cumpla su ciclo para realizar la siguiente. Al mismo tiempo hay que tener cuidado de no sobrecargar la CPU puesto que el ciclo podría durar más de lo esperado y causar retrasos en sistemas que tienen que funcionar en tiempo real, como los PIDs. Por este motivo, la frecuencia máxima es limitada. La frecuencia escogida finalmente ha sido de 200Hz. A esta frecuencia se dispone de tiempo suficiente para ejecutar una lectura de los sensores, una muestra del PID de equilibrio y una muestra del PID de velocidad, además del resto de funciones sin sobrecargar la CPU.

Dentro del bucle, no todas funciones a las que llama se ejecutan en cada iteración. Solamente la lectura de los sensores y la ejecución de los PIDs sucede en cada iteración, en el caso de que estén activados. El resto de funciones se ejecutan a una frecuencia inferior a la frecuencia del bucle principal. Por ejemplo, el display de menú LCD en tiempo real sucede cada veinte iteraciones. La Figura 18 muestra el diagrama de flujo del bucle principal.

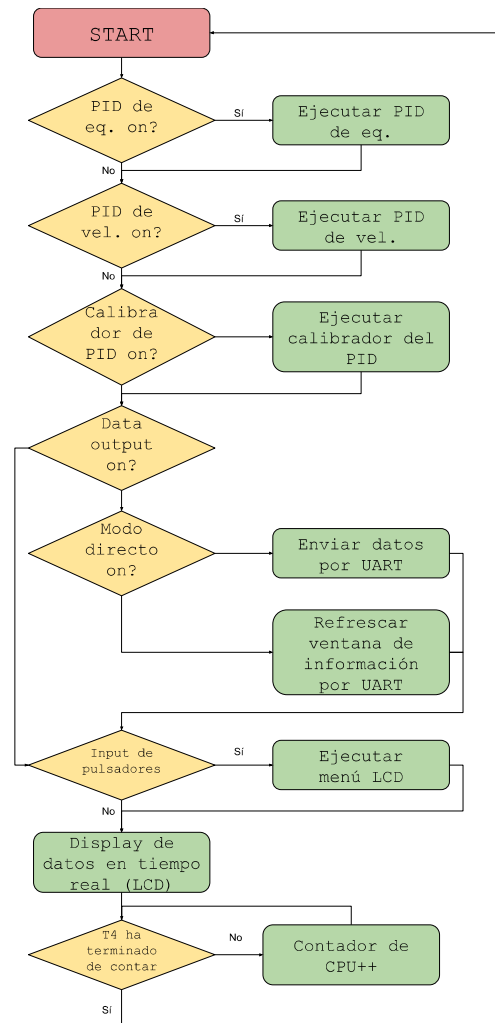


Figura 18: Diagrama de flujo del bucle principal

### ♦ Funciones de otros periféricos

#### ◦ Envío de datos por UART

A través de la UART se han implementado dos funciones diferentes para enviar información en tiempo real al PC. Una de ellas consiste en una ventana de información que muestra selectivamente datos de varios dispositivos del robot en tiempo real, como cada módulo *Output Compare* o la lectura del sensor. Además, permite enviar comandos al robot desde el PC para controlar diferentes funciones. La segunda función es el modo directo. A través de esta función se envía en períodos regulares el valor de un conjunto de variables seleccionadas, tabuladas para la formación de gráficas.

### ♦ Menú LCD

El menú LCD es la herramienta principal para controlar el robot. A través de este menú se pueden activar el PID de equilibrio, el PID de velocidad, la salida de datos o calibrar ambos PIDs, tanto manualmente como automáticamente, entre otras funciones. En la Figura 10 se muestra la estructura del menú principal y sus submenús.

#### ◦ **Controles**

El menú de controles permite activar y desactivar las siguientes funciones: el PID de equilibrio, el PID de velocidad, recuperación de posición, el modo directo, la salida de datos y la grabación de la UART. Al seleccionar una de estas opciones se alterna su estado entre activado y desactivado. Las dos primeras permiten activar los sistemas de control de equilibrio y velocidad. La siguiente activa la recuperación del control de velocidad. Alterna el modo de representación de datos de la UART entre ventana y salida directa. Por último, grabar es una función especial que fuerza el modo directo, activa la UART y comienza a enviar el conjunto de variables seleccionadas. Para seleccionar las variables a enviar hace falta modificar el programa.

#### ◦ **Calibrar PID**

Este menú contiene las opciones para calibrar los PIDs. Se puede seleccionar calibración manual o automática.

#### ◦ **Display Var**

*Display Var* permite mostrar varias variables en tiempo real a través de la pantalla LCD. *PID comps* muestra los términos del PID de equilibrio, *CPU per* muestra el porcentaje de tiempo de CPU libre al final de cada iteración, *Main SP* muestra la variable de salida de ambos PIDs, *velocímetro* muestra la velocidad de ambas ruedas y *distancia* muestra la distancia recorrida desde la inicialización. La mayoría de estas opciones han resultado útiles para el desarrollo del programa y no cumplen una función práctica en el uso normal del robot

#### ◦ **Overload LED**

Permite apagar el led (D5) que indica la sobrecarga de la CPU manualmente. Éste también se apaga al cabo de 5s independientemente.

#### ◦ **Lista de acciones**

Permite ejecutar un *script* con una lista de acciones que se suceden al cumplirse una determinada cantidad de tiempo o recorrerse una distancia. Se programan modificando el archivo *toDoList.c*.

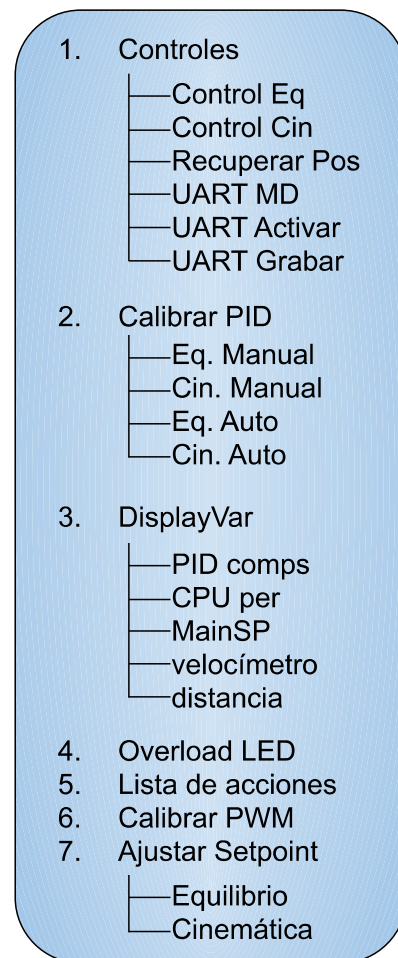


Figura 19: Estructura del menú LCD

- **Calibrar PWM**

Permite ajustar el mínimo/máximo del ciclo de trabajo del PWM para ayudar a compensar la diferencia entre el motor izquierdo y el derecho.

- **Ajustar Setpoint**

Las opciones de este menú permiten cambiar el punto de equilibrio (*set point* de equilibrio) y la velocidad deseada (*set point* de velocidad), en grados y cm/s respectivamente.

## 3.3 Puesta en marcha

---

En este apartado se analiza el proceso de arranque del robot desde que se enciende, los pasos a dar para ponerlo en marcha y las distintas funcionalidades que este ofrece.

- ◆ **Encendido y secuencia de arranque**

Para encender el robot hay que asegurarse de que la batería esta conectada a la placa y simplemente activar el interruptor soldado a la parte superior derecha de la Explorer16. Una vez se activa este interruptor la placa comienza a recibir corriente y el programa cargado en el microcontrolador comienza a ejecutarse. El robot necesita aproximadamente dos segundos para realizar el proceso de inicialización en el que todos sus módulos internos son configurados y puestos en marcha mientras el LCD muestra un mensaje que indica que este proceso está teniendo lugar. Al cabo de un segundo aproximadamente este proceso termina y un temporizador de un segundo se muestra en pantalla.

- ◆ **Estado tras la secuencia de arranque**

Cuando la secuencia de inicialización ha terminado, el robot ya se encuentra calculando la inclinación en todo momento, mostrando si se encuentra aproximadamente en posición de equilibrio con los leds D9, D8 y D7 de la Explorer16. Cuando el robot se encuentra en posición erguida solo el led central (D8) se enciende. Cuando sale de esta posición, el led D9 o el D7 se enciende, indicando la dirección de la inclinación. Esto significa que el error del PID de equilibrio se está calculando en todo momento desde que el robot se enciende, aunque el sistema de control en sí esté desactivado.

Lo mismo sucede con el velocímetro/odómetro. Este funciona en todo momento, calculando la velocidad actual del robot y midiendo su posición. A través de la opción *DisplayVar>Velocímetro* del menú LCD se puede comprobar su funcionamiento en tiempo real. Igualmente, con la opción *DisplayVar>Distancia* se puede comprobar la distancia recorrida. Nótese que, tal y como se ha programado este dispositivo, sin el PID de equilibrio funcionando no se distingue el sentido del giro de las ruedas, por lo que ambas magnitudes se miden en términos absolutos.

Al mismo tiempo, se indica la frecuencia de trabajo del Bucle principal (3.2.5, “Aspectos misceláneos de la programación”) a través del parpadeo de los leds D4 y D3. El D3 parpadea en cada iteración, mientras que el D4 parpadea cada diez iteraciones. Además de estos dos, hay un tercer led (D5) que permanece apagado a no ser que se produzca una sobrecarga en la CPU y el bucle principal no complete su ciclo a tiempo. En este caso el led permanece



encendido durante 5 segundos, a no ser que sea apagado con la opción *Overload LED* del menú LCD.

Este es el estado en que el robot se encuentra tras la secuencia de arranque. A partir de este momento es posible navegar por el menú LCD para activar las distintas funciones.

#### ♦ Navegación por el menú

Una vez completada la cuenta atrás mostrada por pantalla se muestra la primera opción del menú LCD. A partir de este momento se pueden emplear los pulsadores de la Explorer16 para navegar. Los pulsadores S3 y S6 sirven para navegar a través de las opciones en el menú principal y cada submenú, recorriéndolos como listas de arriba a abajo (o izquierda y derecha si se prefiere). En todos los casos, el pulsador S4 sirve para seleccionar en cada menú y el S5 vuelve a la opción *controles* del menú principal. Cuando se muestra la opción *controles* es posible volver a activar el pulsador S5 para acceder rápidamente a la opción *Overload LED*, la cual se encuentra aproximadamente en la mitad del menú principal.

Para arrancar los sistemas de control hay que seleccionar *controles*. En esta opción lleva al submenú de controles en la que se encuentran las opciones *Control\_Eq* y *Control\_Vel*, que permiten activar los PIDs de equilibrio y velocidad respectivamente.

Al activar el PID de equilibrio se recomienda poner el robot en posición erguida, pues este comienza a corregir la posición instantáneamente y puede reaccionar de manera brusca. Si en algún momento el robot supera los 30° de inclinación, el control de equilibrio se desactiva automáticamente. Por ello el robot debe encontrarse, al menos, dentro de un margen de  $\pm 30^\circ$  con respecto a la posición vertical para activar el control de equilibrio.

Activar el PID de velocidad no tiene ningún efecto si el PID de equilibrio no se encuentra activo al mismo tiempo. No obstante, es posible activarlo antes y tenerlo funcionando sin efecto aparente hasta que se active el PID de equilibrio.

El resto de opciones junto con la estructura del menú LCD se muestran en el subapartado Menú LCD (3.2.5 “Aspectos misceláneos de la programación”).



# 4

---

## Resultados

---

A lo largo del proyecto se ha conseguido una solución para todos los objetivos propuestos al comienzo del mismo.

## 4.1 Sistemas de control

---

### 4.1.1 Equilibrio

El sistema de control escogido para el equilibrio ha sido un PID según la forma paralela, aunque ésta no suele ser la mejor solución para sistemas reales. En la mayoría de aplicaciones comerciales e industriales se emplea alguna variante como un sistema de control de tipo PI o PD, o una variante diferente del PID, como la forma estándar. Esta forma no suele ser escogida porque es más difícil de calibrar.

En sistemas auto-balanceados, en cambio, la forma paralela es la más común y se encuentra en casi todos los ejemplos disponibles. Con este tipo de control y una calibración correcta se ha conseguido un equilibrio robusto capaz de aguantar indefinidamente y alcanzar la inclinación deseada de forma rápida y precisa, sin apenas *overshoot*.

Este sistema de control sólo aporta la solución para la tarea básica de conservación del equilibrio. No obstante es una parte fundamental para el resto de objetivos, puesto que sirve como base para el control de velocidad. Sin un control de equilibrio fiable, de respuesta rápida y precisa, habría sido imposible implementar un control de velocidad efectivo y la mayoría de los objetivos no podrían haber sido alcanzados.

### 4.1.2 Velocidad

Para el control de velocidad se ha utilizado un PID conocido como PID en cascada. Este tipo de control consiste en dos PID's anidados, donde uno de ellos controla el valor deseado del otro. Se les denomina PID externo e interno. El externo controla directamente una variable física como la velocidad, mientras que el interno toma la salida del externo como valor deseado. En el caso del prototipo, el controlador externo es el control de velocidad, marcando una inclinación deseada para el PID de equilibrio. Mientras el controlador interno se ajusta rápidamente al valor deseado indicado en cada momento, el controlador externo ajusta su valor deseado más lentamente hasta que los codificadores miden la velocidad indicada.

Este sistema, con la configuración adecuada, ha servido como solución para cumplir los objetivos de estabilidad en planos inclinados, contrarrestar cambios repentinos en el centro de masa, desplazamiento, recuperación de caídas y recuperación de la posición inicial.

## 4.2 Tareas resueltas

---

### 4.2.1 Objetivos de estabilidad

- ♦ **Estabilidad en planos inclinados y cambios del centro de masa**

Gracias al control de velocidad, es fácil implementar una solución para ambas situaciones. Basta con fijar la velocidad deseada a cero. De esta manera el sistema compensa el movimiento causado en ambos casos y, gracias al término integral del PID, recuerda la inclinación necesaria para mantenerse estático en un plano inclinado o en el caso de cambiar su punto de equilibrio. También funciona con cualquier otra velocidad. Si se le indica, puede alcanzarla o al menos aproximarla en el caso de que se requiera una inclinación demasiado grande para compensar la pendiente y salga de los límites del anti-*windup*.

Aun así, esto tiene una solución simple consistente en ajustar este límite para cada situación. También se puede conseguir aplicando el valor de inclinación puesto por el control de velocidad al SP base del control de equilibrio, ya que la salida del controlador externo se aplica como *offset* sumado al SP del control interno.

- ♦ **Recuperación de caídas**

A diferencia de como se detallaba en el apartado de tareas a resolver, la consecución de un control de equilibrio robusto no es suficiente para completar esta tarea. En lugar de ello hace falta la acción combinada del control de equilibrio con la del control de velocidad. Sin la acción del PID de velocidad, el robot simplemente se mantendría avanzando en pie en la dirección del impulso hasta que parar por fricción. También sería más vulnerable a perturbaciones externas, pues sin el efecto del control de velocidad hace falta menos fuerza para llevarlo a una inclinación de la que no sea posible la recuperación.

Con el control de velocidad configurado para mantener la velocidad a cero se puede cumplir este objetivo. En este caso, aplicar una fuerza al robot que lo haga salir de la velocidad deseada causa un efecto de compensación, inclinándose drásticamente en sentido opuesto al del impulso, frenando rápidamente y recuperando, por completo o en parte, su posición original.

### 4.2.2 Objetivos de cinemática

- ♦ **Recuperación de la posición original**

Como se ha mencionado previamente en el apartado de recuperación de caídas, el control de velocidad produce una drástica inclinación en sentido opuesto al movimiento cuando se mide una velocidad diferente a la deseada. Cuando esto sucede, se produce un efecto de recuperación de la posición muy preciso, puesto que no se puede alcanzar una posición de reposo hasta que el término integral se reduzca a cero. No importa que la velocidad vuelva a ser cero antes de que esto suceda. Es gracias a este efecto de *windup* que funciona también la estabilidad en planos inclinados y el cambio de centro de masa.

Sin embargo, la recuperación de la posición no funciona completamente. Debido al código anti-*windup* que incluye el PID de velocidad, la posición no se puede recuperar desde

cualquier distancia. Si el impulso hace que se supere una distancia en la que el código anti-*windup* entra en funcionamiento, cualquier distancia recorrida a partir de este punto no será recuperada. La solución es simplemente evitar esta función del PID. De esta manera se recuperará cualquier distancia recorrida.

♦ **Desplazamiento**

Este es el objetivo por el que fue principalmente concebido el control de velocidad. Con él se puede indicar la velocidad deseada y el movimiento comienza automáticamente, regulado por el PID de velocidad.

♦ **Maniobrabilidad**

La habilidad de girar mientras se mantiene el equilibrio se ha conseguido a través del PID de equilibrio. En el interior del algoritmo se ha incluido lo que se ha bautizado como ‘el diferencial’. Esta es una parte del código que, tras calcular el valor de la señal de salida, la reparte entre los dos motores. De esta manera, si se calcula un valor para la salida, en lugar de asignar este valor a ambos motores por igual, se reparte un porcentaje de esta a un motor y el resto al otro, por ejemplo en una proporción 25%-75%. Esto permite seguir regulando el equilibrio mientras se realizan giros. Para más detalles sobre este mecanismo, consultar el apartado 3.2.3.3 PID de equilibrio.

La sensibilidad del control de velocidad se ve afectada cuanto mayor es la diferencia en el reparto entre un lado y el otro, por lo que se requiere una nueva configuración de calibración del PID para giros cerrados.

# 5

---

## Conclusiones y trabajo futuro

---

El sistema de control PID implementado es altamente efectivo para el control de equilibrio del robot balancín, pero su uso aislado es muy limitado para la consecución de las metas que podrían considerarse imprescindibles en una aplicación seria para un robot autobalanceado.

Aunque se puede conseguir un control de equilibrio altamente fiable, la adición del resto de las características que hacen un buen robot autobalanceado, como la estabilidad en planos inclinados, la locomoción o la conservación de la posición puede ser muy complicada y con grandes limitaciones.

El control del movimiento, por ejemplo, puede ser conseguido fácilmente a través de la manipulación de la inclinación, pero se carece del control de velocidad. Una función como mantener la posición ante impulsos externos o simplemente durante el reposo requiere de la programación de un protocolo para reaccionar ante estos cambios e incluso del uso de calibraciones alternativas a medida que éstos se producen. La estabilidad en planos inclinados es prácticamente imposible de implementar sin añadir modificaciones al hardware del robot y aun así no resulta trivial de implementar.

El control de velocidad en combinación con el de equilibrio a través del PID en cascada completa las características de equilibrio y control que necesita un robot de esta clase, aportando una simple implementación para estabilidad en planos horizontales e inclinados, compensación de cambios en el centro de masa y ajuste del punto de equilibrio, recuperación de la posición y desplazamiento con control de velocidad.

Como se ha expuesto anteriormente, el objetivo de este proyecto ha sido proporcionar un sistema de control de equilibrio y cinemática que sirva para facilitar la implementación de cualquier aplicación que implique la movilidad o estabilidad del robot.

El resultado obtenido ha sido una estructura de robot balancín estable a la que se le pueden añadir aplicaciones específicas como el control remoto o la incorporación de sensores con los que implementar un sistema de control autónomo. Una aplicación interesante a partir de este proyecto podría ser aumentar la escala del robot y construir una versión simple del *Segway*.

Además queda experimentar con sistemas de control alternativos como las otras formas del PID (que ofrecen nuevas posibilidades en materia de calibración) y el empleo de lógica difusa como alternativa al PID.

En materia de sensores, habría que mejorar la programación de varios de los presentes en el robot e, incluso, la adición de nuevos para mejorar la precisión de la percepción de la orientación y situación general del robot. En la lectura de los codificadores de las ruedas, queda por confirmar el sentido de giro. Actualmente, el sentido de giro es conocido en todo momento por el control de equilibrio, pues es impuesto por este y si las ruedas giran por sí mismas no hay forma de determinar el sentido del giro. El módulo de sensores ha sido programado a partir de un código de demostración del fabricante[22]. La programación desde cero de este habría supuesto demasiado tiempo, pues no se dispone de una buena documentación. Investigar el sensor más a fondo puede traer mejoras de precisión y, en especial, de eficiencia. Añadir sensores de proximidad puede ser útil para extender los sentidos del robot, mejorando potencialmente la lectura de la inclinación y permitiendo detectar la inclinación de una superficie, comparando su lectura con la del módulo de sensores. Por último, queda implementar una fusión de sensores a través de un filtro kalman. Esto permitiría combinar las lecturas de todos los sensores presentes en el robot junto con un modelo físico de este y mejorando la percepción de la posición.



Para lo mencionado en el párrafo anterior, la utilización de un microcontrolador con procesador matemático también es interesante, pues aplicaciones como el filtro kalman pueden consumir muchos recursos de CPU debido a altos volúmenes de operaciones en coma flotante.

Finalmente, la implementación de un sistema dinámico que equilibre de la velocidad de los motores en todo momento es necesaria para utilizar el robot en una aplicación real. Ésto es así ya a que, a la larga y tras numerosas correcciones de la inclinación en una posición estática o al realizar desplazamientos, se acumula un error que deriva en la progresiva pérdida de la posición original o del rumbo esperado. Ésto sucede debido a que cada rueda es impulsada por un motor eléctrico independiente. Cada uno de ellos es diferente y cada uno sufre un desgaste diferente, por lo que sin un mecanismo interno que equilibre sus velocidades es imposible mover el robot en línea recta.

# Bibliografía

---

- 1: Markel Picado, Estudio de la cinemática de un móvil de dos ruedas y su control para mantener el equilibrio teledirigido por Bluetooth, 2017, [https://addi.ehu.es/bitstream/handle/10810/15886/PFG\\_MarkelPicadoOrtiz.pdf?sequence=1&isAllowed=y](https://addi.ehu.es/bitstream/handle/10810/15886/PFG_MarkelPicadoOrtiz.pdf?sequence=1&isAllowed=y)
- 2: Anartz Recalde, Estudio de la cinemática de un móvil de dos ruedas y su control para mantener el equilibrio disponiendo de encoder, <https://addi.ehu.es/bitstream/handle/10810/15887/TFGANartzFinal.pdf?sequence=1&isAllowed=y>
- 3: , Self-balancing Table, 2007, <https://www.youtube.com/watch?v=9ViD04b5q5E>
- 4: , Motor GB37Y3530, [https://wiki.dfrobot.com/12V\\_DC\\_Motor\\_251rpm\\_w\\_Encoder\\_\\_SKU\\_\\_FIT0186\\_](https://wiki.dfrobot.com/12V_DC_Motor_251rpm_w_Encoder__SKU__FIT0186_)
- 5: Microchip technology Inc., PIC24FJ128GA Family Data Sheet, 2006, <https://drive.google.com/file/d/1zdDv4Zkg1X6iarTxNcOho2Malzd40Skm/view>
- 6: Thomas Whitton, Explorer 16 LCD - (Código controlador PMP), <https://www.thomaswhitton.com/blog/2011/02/13/explorer-16-lcd/>
- 7: Microchip Technology Inc., Explorer 16 Development Board User's Guide, 2005 - 2014, <http://ww1.microchip.com/downloads/en/devicedoc/50001589b.pdf>
- 8: Novatek, NT7603 LCD Controller Datasheet, <https://www.crystalfontz.com/controllers/Novatek/NT7603/>
- 9: Truly semiconductors LTD, LCD MODULE TSB1G7000-E, 2008, <http://ww1.microchip.com/downloads/en/devicedoc/tsb1g7000-ev1.0.pdf>
- 10: Microchip Technology Inc., MM7150 Motion Module, 2015 - 2016
- 11: Microchip Technology Inc., SSC7150 Motion Coprocessor, 2015, <http://ww1.microchip.com/downloads/en/DeviceDoc/00001885A.pdf>
- 12: Microchip Technology Inc., AN1873 Host API Design for MM7150 Motion Module, 2014, <http://ww1.microchip.com/downloads/en/Appnotes/00001873A.pdf>
- 13: Microchip Technology Inc., MM7150 Motion Module PICtail Plus Evaluation Board (#AC243007) with Explorer 16 Development Board User's Guide, 2014 - 2016, <http://ww1.microchip.com/downloads/en/DeviceDoc/50002322E.pdf>
- 14: Microchip Technology Inc., Section 16. Output Compare, 2006, <http://ww1.microchip.com/downloads/en/DeviceDoc/39706a.pdf>
- 15: Microchip Technology Inc., Section 15. Input Capture, 2006, <http://ww1.microchip.com/downloads/en/DeviceDoc/39701a.pdf>

- 16: Wikipedia, PID controller, , [https://en.wikipedia.org/wiki/PID\\_controller](https://en.wikipedia.org/wiki/PID_controller)
- 17: Jacques Smuts , PID controller algorithms, 2010,  
<https://blog.opticontrols.com/archives/124>
- 18: Brett Beauregard, Improving the Beginner's PID, 2011,  
<http://brettbeauregard.com/blog/2011/04/improving-the-beginners-pid-introduction/>
- 19: Eric Cai, Rectangular Integration (a.k.a. The Midpoint Rule) – Conceptual Foundations and a Statistical Application in R, 2014,  
<https://chemicalstatistician.wordpress.com/2014/01/20/rectangular-integration-a-k-a-the-midpoint-rule/>
- 20: Jacques Smuts , Cohen-Coon tuning rules, 2011,  
<https://blog.opticontrols.com/archives/383>
- 21: Jaques Smuts, Lambda tuning rules, 2010,  
<https://blog.opticontrols.com/archives/260>
- 22: Microchip Technology Inc., Código de muestra para MM7150,  
[https://www.microchip.com/SWLibraryWeb/product.aspx?product=MM7150\\_Host\\_Example](https://www.microchip.com/SWLibraryWeb/product.aspx?product=MM7150_Host_Example)
- 23: Microchip Technology Inc., Section 6. Oscillator, 2009,  
<https://drive.google.com/file/d/1Apl-RBDRPGUTYkIycKUWsAqRWIdS8Uwg/view>
- 24: Microchip Technology Inc., Section 8. Interrupts, 2006,  
<http://ww1.microchip.com/downloads/en/devicedoc/39707a.pdf>
- 25: Microchip Technology Inc., Section 29. Instruction Set, 1997,  
<http://ww1.microchip.com/downloads/en/devicedoc/31029a.pdf>