**DISEÑO E IMPLEMENTACIÓN DE UNA RED NEURONAL ARTIFICIAL PARA LA ESTABILIZACIÓN DE UN CUADRICÓPTERO**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tutora Revisor

Ing. Evelenir Barreto

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tesista Tesista

Luis Vicens Yoshua Nava

**Planteamiento del problema**

En los últimos tiempos, con el desarrollo de las telecomunicaciones y la microelectrónica, existe una tendencia hacia el desarrollo de vehículos aéreos no tripulados, ya sea manejados a distancia o autónomos. En particular, se ha dado especial atención al desarrollo de multi-rotores, ya que estos brindan una gran maniobrabilidad y precisión durante el vuelo, características muy útiles para tareas de reconocimiento, vigilancia, exploración, etc. Entre los multi-rotores uno de los más conocidos, y sobre el que se tratará en el siguiente trabajo especial de grado, es el cuadricóptero: un multi-rotor propulsado por cuatro motores posicionados de forma simétrica.

Muchas ideas sobre desarrollo de cuadricópteros vienen de épocas anteriores al descubrimiento del transistor, y sólo han podido llevarse a cabo en tiempos recientes, debido al avance de los microprocesadores y las baterías químicas. En principio sólo organizaciones militares y de investigación podían participar en el desarrollo de los cuadricópteros, por el alto costo de la microelectrónica en sus primeros años. Pero con el paso del tiempo se ha llegado a una época en la que el desarrollo de cuadricópteros es un pasatiempo común entre aficionados a la electrónica, los cuales pueden acudir a un mercado muy competitivo de piezas y componentes de construcción de estos vehículos. Incluso, hay grandes proyectos como Arduino o Raspberry Pi que buscan brindar al usuario común la oportunidad de aproximarse al desarrollo de los cuadricópteros (Y de la electrónica en general) con una mayor simplicidad.

Para usuarios novatos y con el prospecto de realizar proyectos de bajo coste, se han realizado varios trabajos sobre el desarrollo de cuadricópteros que han planteado el uso de diversas plataformas de hardware que funcionan de manera relativamente parecida, pero no se ha logrado la estandarización de un sistema de control de estabilidad y vuelo que pueda adaptarse a las características del medio en el que se desenvuelve, sin que el usuario tenga que optimizar por sí mismo los parámetros de los sistemas de control del cuadricóptero. Considerando al usuario novato que desea acercarse por primera vez al desarrollo de un vehículo de este tipo, hay varios casos de desarrollo de plataformas de cuadricópteros de bajo coste que difieren altamente en la implementación del sistema de control, y que pueden presentarle mayor cantidad de dificultades a la hora de intentar construir un vehículo por su propia cuenta.

En universidades y organizaciones militares se han realizado varios proyectos sobre diseño y construcción de cuadricópteros que puedan mantener una alta estabilidad y maniobrabilidad en el aire. Pero, en su mayoría, han necesitado del apoyo de sensores externos, y además, por el gran coste computacional de los algoritmos de control tradicionales, se ha tenido que realizar el procesamiento de datos para mantener el control sobre el cuadricóptero en computadoras separadas del vehículo, haciendo que éstos no puedan tener real independencia de otros sistemas de cómputo.

Se propone desarrollar un algoritmo basado en redes neuronales artificiales capaz de estabilizar y mantener un control eficaz sobre la dinámica en vuelo de un cuadricóptero que funcione sobre la plataforma Arduino sin el apoyo de un sistema de control Proporcional-Integral-Derivativo. En caso de llegar a un resultado satisfactorio el proyecto, se estaría en presencia de un método de estabilización del cuadricóptero de bajo costo computacional (Pues las redes neuronales luego de ser entrenadas se ejecutan rápidamente), y que pudiera servir para complementar aquellos proyectos desarrollados anteriormente en torno a la construcción física de los cuadricópteros.

**Objetivo General:**

Diseñar e implementar una red neuronal artificial que permita estabilizar un cuadricóptero desarrollado sobre la plataforma Arduino.

**Objetivos Específicos:**

* Diseñar e implementar un cuadricoptero con una unidad de control basada en Arduino.
* Diseñar e implementar una interfaz de comunicación inalámbrica entre el cuadricóptero y una computadora para tareas de encendido, apagado, movimientos simples en tres dimensiones y recopilación de información de los sensores del cuadricóptero.
* Evaluar los distintos tipos de redes neuronales artificiales en función de su capacidad de convergencia, estabilidad y memoria asociativa.
* Evaluar los distintos tipos de algoritmos de entrenamiento que se utilizarán en la red neuronal artificial seleccionada.
* Diseñar e implementar la red neuronal artificial que permita la estabilización del cuadricóptero.
* Diseñar e implementar una plataforma de pruebas en tiempo real.

**Alcance**

El siguiente trabajo especial de grado tiene por objeto el desarrollo de un algoritmo de estabilización de un cuadricóptero basado en redes neuronales artificiales. El cuadricóptero en cuestión poseerá una unidad de control basada en la plataforma Arduino, y, para objeto de pruebas, podrá establecer comunicación con un agente externo que permitirá realizar las siguientes acciones:

* Encendido y apagado del cuadricóptero de forma remota.
* Control remoto de los movimientos del cuadricóptero desde la computadora. Se implementarán una serie de comandos de control los cuales permitirán mover el cuadricóptero hacia arriba, abajo, adelante, atrás, izquierda, y derecha; sin poder combinar entre sí estos movimientos.
* Obtención de los datos de los sensores para evaluar la eficiencia del algoritmo y la estabilidad del robot.

**Limitaciones**

- La plataforma a utilizar para el manejo del cuadricóptero será un microcontrolador Arduino Mega2560, que trabaja a una frecuencia máxima de 16MHz.

- La red neuronal artificial se desarrollará sobre el lenguaje de programación Processing.

- El sistema usará una batería de 2350mAh que servirá para alimentar a los motores, y durará entre 15 y 20min de uso continuo.

- El sensor Sharp GP2Y0A02YK0F que se utilizará para medir la distancia respecto al suelo tendrá un rango máximo de un metro y medio, por lo cual, al realizar el despegue de forma autónoma, sólo se podrá alcanzar dicha distancia máxima.

- El cuadricóptero durante el vuelo no podrá detectar objetos a su alrededor ni evadirlos, por lo cual el ambiente de pruebas debe estar totalmente despejado.

- Sólo se programará al cuadricóptero para realizar seis movimientos simples guiados por el usuario: adelante, atrás, izquierda, derecha, ascenso y descenso.

- Las pruebas en tiempo real serán realizadas en un ambiente a puerta cerrada.

**Justificación**

El motivo para realizar este trabajo especial de grado se debe a que en la actualidad existe una gran demanda en el área de desarrollo de vehículos aéreos no tripulados, tanto para aplicaciones civiles como para aplicaciones militares. Debido a que estos dispositivos pueden servir para realizar tareas en lugares no alcanzables fácilmente por el ser humano, el poder disminuir el riesgo humano es un factor que motiva la construcción y avance en este campo. Aun así, no se ha podido desarrollar un método de control y estabilización que no dependa de sensores externos, o que pueda adaptarse al medio sin necesitar una configuración previa exhaustiva por parte del usuario.

Desarrollar un algoritmo basado en redes neuronales artificiales, que pueda aprender a mantener estable el cuadricóptero permitiría que los usuarios novatos en la construcción de cuadricópteros que no posean la instrucción necesaria en materia de sistemas de control puedan dar mayor rienda suelta a su creatividad a la hora de construir un sistema autónomo, o controlado por el usuario. En caso de llegar a buen puerto el desarrollo del proyecto, se tendría un algoritmo que podría adaptarse a las condiciones de diversos ambientes externos y configuraciones físicas de los cuadricópteros, y liberar al usuario del proceso de configuración del sistema de control.

Han habido múltiples trabajos en torno al tema del desarrollo de una plataforma estandarizada para el desarrollo de cuadricópteros, pero no se ha logrado estandarizar el funcionamiento y la configuración básica que debería tener el sistema de control. Se considera que el presente Trabajo Especial de Grado podría representar un aporte, y un nuevo paso adelante en el desarrollo de una plataforma para el control de cuadricópteros mediante Arduino desarrollada en **[6]**, puesto que en dicha tesis se llevó a cabo el desarrollo completo del hardware del cuadricóptero, los algoritmos de lectura de sensores y manejo de motores, y quedó por desarrollar el algoritmo de estabilización y control.

Se han dado varios antecedentes de desarrollo de algoritmos de comportamiento de robots terrestres basados en algoritmos bioinspirados, y se considera que el siguiente Trabajo Especial de Grado puede representar un primer acercamiento al control de sistemas robóticos voladores mediante redes neuronales artificiales, pudiendo representar, a futuro, un antecedente al desarrollo de un vehículo aéreo no tripulado cuyo sistema de estabilización, navegación y comportamiento esté basado en técnicas de programación bioinspirada.

**Marco Referencial**

**Redes neuronales artificiales (RNA)**

Se definen como “La disciplina científica técnica que se ocupa del estudio de las ideas que permiten ser inteligentes a los ordenadores” (definición de H. Winston). También se puede definir como el modelo computacional inspirado en la forma como trabaja el cerebro para resolver problemas. **[10]**

Las RNA son Unidades enlazadas a través de conexiones cargadas por pesos numéricos con las siguientes reglas:

* El aprendizaje se fundamenta en la actualización de esos pesos que se inicializan y ajustan en la fase de entrenamiento de la red
* Está formada por neuronas de entrada, neuronas de salida y eventualmente neuronas de ocultas dentro de la red.
* El nivel de activación de la neurona artificial (equivalente al impulso excitatorio) es un cálculo individual en cada neurona, sin control global.

**[8]**

Las redes neuronales tienen características muy particulares y no se encuentran en el modelo computador propuesto por Von Neumann, algunas de estas son:

* Procesamiento Paralelo
* Cálculo y representación distribuida
* Habilidad de aprender Socráticamente (por medio de ejemplos)
* Habilidad de generalizar Adaptabilidad
* Tolerancia a fallas
* Las conexiones en la RNA almacenan la información del patrón de entrenamiento

**[13]**

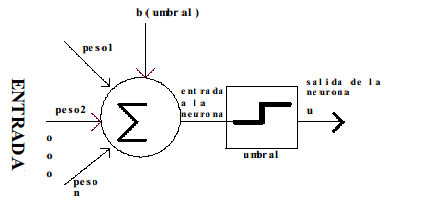
**Neurona artificial**

Es un procesador elemental que posee la capacidad limitada de calcular, en general, una suma ponderada de sus entradas y luego le aplica una función de activación para obtener una señal que será transmitida para la próxima neurona. Estas neuronas artificiales se agrupan en capas o niveles y poseen un grado de conectividad entre ellas, conectividad que es ponderada por los pesos. A través de un algoritmo de aprendizaje supervisado o no supervisado las RNA ajustan su arquitectura y parámetro de manera de poder minimizar alguna función de error que indique el grado de ajuste a los datos y la capacidad de generalización de las RNA **[10]**.

Las neuronas están compuestas por 3 funciones principales:

* **La función de activación:** Es la encargada de relacionar la información de entrada de la neurona con el siguiente estado de activación que tenga esa neurona. Existen dos modelos de función de activación:
  + Modelos acotados: El valor de la activación de la neurona puede ser cualquiera dentro de un rango continuo de valores.
  + Modelos No acotados: No existe ningún límite para los valores de activación.
* **La función de salida:** Esta función convierte el estado de la neurona en la salida hacia la siguiente neurona que se transmite por las sinapsis. Usualmente no se considera y se toma la identidad, esto es, de manera que la salida es el propio estado de activación de la neurona. Existen algunas redes que transforman su estado de activación en una salida binaria y para eso usan la función escalón antes mostrada como salida. Cuando se diseña una red debe establecerse cómo van a ser los valores de activación de cada neurona y se debe decidir la función de activación (FA) con la que cada neurona procesará las entradas **[5]**.
* **La función de ponderación**: Esta función se encarga de transformar las diferentes entradas que provienen de la sinapsis en el potencial de la neurona. Normalmente se usa como función de propagación la suma ponderada de las entradas multiplicadas por los pesos. En esta función se interpreta como un regulador de las señales que se emiten entre neuronas al ponderar las salidas que entran a la neurona **[5]**.

La neurona artificial o PE propuesto por McCulloch y Pitts es una de las más simples, como se puede observar en la figura1:



**Figura1**

Se presentan varias señales de entrada a la neurona y cada una de estas señales es multiplicada por un valor (peso de conexión). En el caso más simple (este caso), estos productos solo son sumados y alimentados a través de una función de transferencia de límites rígidos para finalmente generar un resultado binario. Cuando la entrada a la neurona es menor al umbral entonces la salida será 0, en caso contrario será 1. En este modelo se presentan muchas simplificaciones que no reflejan el verdadero funcionamiento de una neurona biológica **[13 ]**. El modelo matemático de la neurona McCulloch-Pitts está dado por:

Dónde:

* u = Valor de la señal de salida de la neurona, regularme e se considera igual al nivel de activación de la neurona.
* w = Peso de la entrada j.
* y= Valor de señal de entrada j.
* n = Número de entradas a la neurona.

**Arduino**

Arduino es una placa controladora de hardware libre muy accesible y de sencilla programación la cual permite montar proyectos con rapidez. Está basada en una sencilla placa con entradas y salidas analógicas y digitales, en un entorno de desarrollo que implementa el lenguaje de programación Processing. Originalmente estaba basado en el microcontrolador ATMEL ATmega328, un chip sencillo y de bajo coste que permite el desarrollo de múltiplos diseños. Según ha ido avanzando el tiempo, el procesador se ha ido actualizando hasta llegar al modelo Atmega1280 con mejores prestaciones, como por ejemplo, más memoria flash (128 kb) y 16 puertos de entrada y salida**. [1]**

**Cuadricóptero**

Se puede definir como una aeronave que se eleva y se desplaza por el movimiento de cuatro motores colocados en los extremos de una estructura en forma de cruz. Normalmente se utiliza el nombre inglés quadrotor aunque también existe la traducción cuadricóptero. El vehículo dispone de 4 motores con sus palas respectivas, se utiliza la velocidad de los motores para controlar la estabilidad y movimientos del vehículo aéreo.

Una de las características a destacar es la gran maniobrabilidad que pose este tipo de vehículo. Al disponer de cuatro motores el control es bastante exacto, lo que ayuda a utilizarlo en aplicaciones donde la exactitud de vuelo estacionario sea muy importante. Una aplicación donde se aprecia esta característica es en la navegación de interiores y sitios de espacio reducido.

El problema fundamental de los cuadricopteros es su control. El sistema debe de incorporar mecanismos de estabilización para ayudar a la navegación. La capacidad de carga es bastante alta con relación al peso de todo el sistema, pueden encontrarse vehículos que soporten una carga superior al peso que tienen. Esta característica hace posible el incorporar un gran número de sensores.

Una de las características más importantes a tener en cuenta en los sistemas de vuelo es la autonomía. La autonomía de vuelo no suele ser muy buena, esta fue una de las limitaciones por la que los UAV tardaron un cierto tiempo en avanzar**. [6]**

**Teoría de control**

La teoría de control es un estudio matemático acerca de cómo manipular los parámetros que afectan el comportamiento de un sistema, para producir un comportamiento deseado u óptimo. **[14]**

Las principales ramas de la teoría de control que se utilizan habitualmente son la teoría de control clásica (También denominada teoría de control convencional), la teoría de control moderna y la teoría de control robusto. La teoría de control clásica se basa en el modelo de sistemas con una entrada y una salida en el dominio de la frecuencia compleja. La teoría de control moderna se basa en el análisis en el dominio temporal de los sistemas de ecuaciones diferenciales que describen el funcionamiento de un sistema. La teoría de control robusta se basa en definir en primer lugar el rango de posibles errores y después en el diseño del controlador de forma que, si el error del sistema está en dicho rango, el sistema de control diseñado permanezca estable. **[7]**

La teoría de control se ocupa del diseño de sistemas de control. Un sistema de control puede definirse como un arreglo de componentes acoplados de tal manera, que el arreglo pueda comandar, dirigir, o regularse a sí mismo o a otro sistema. **[4]**. Un sistema de control está constituido por entradas, salidas y estados. **[12]**

**Sistema de control Proporcional-Integral-Derivativo (PID)**

El control PID es una de las primeras estrategias de control desarrolladas. Sus implementaciones tempranas fueron en dispositivos neumáticos, seguidos por aspiradoras y electrónica análoga de estado sólido, antes de arribar a la implementación digital de hoy en microprocesadores. Tiene una estructura de control simple que fue en su momento fácilmente entendida y entonada por los operadores de plantas.

La estructura típica de un sistema de control PID viene dada por una función matemática como la que sigue a continuación:

**,**

en la que la señal de error en el tiempo e(t) es usada para generar acciones integrales, derivativas y proporcionales; con las señales resultantes sujetas a un valor peso y sumadas para formar la señal de control u(t) aplicada al modelo de la planta. **[3]**

A continuación se definen los tipos de acción que realiza un controlador PID, y sus componentes de tiempo:

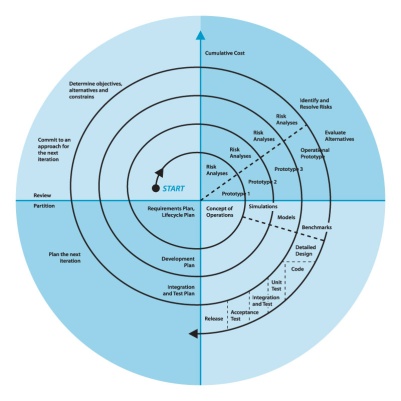
* Acción proporcional (**P**): es la acción que produce una señal proporcional a la desviación de la salida del proceso respecto al punto de consigna.
* Acción integral (**I**): es la acción que produce una señal de control proporcional al tiempo que la señal del proceso ha sido diferente del punto de consigna.
* Acción derivativa (**D**): es la acción que produce una señal de control proporcional a la velocidad con que la salida del proceso está cambiando respecto del punto de consigna.
* Constante de tiempo integral (: es el tiempo que debe transcurrir para que la acción integral alcance, iguale o repita a la acción proporcional.
* Constante de tiempo derivativa (: es el intervalo de tiempo en el que la acción derivativa adelanta a la acción proporcional.

Cada acción de control tiene una respuesta característica:

* La acción proporcional varía instantáneamente con el error y alcanza un valor estacionario cuando lo alcanza éste.
* La acción integral tiene en cuenta la historia pasada del error y se anula cuando se hace cero.
* La acción derivativa predice los cambios en el error y se anula cuando alcanza un valor estacionario. **[15]**

**Metodología**

Por las características del proyecto a desarrollar, se utilizará una adaptación de la metodología de desarrollo de software en espiral. Usando el modelo en espiral, el software es desarrollado en una serie de entregas incrementales. Durante las primeras iteraciones, la entrega incremental puede ser un modelo en papel o un prototipo. A lo largo de las últimas iteraciones, versiones en aumento más completas de los sistemas desarrollados son producidas **[9].**



**Ilustración 1: Ciclo de vida en espiral. Obtenido de:**

[**http://eternalsunshineoftheismind.files.wordpress.com/2013/03/spiral.jpg**](http://eternalsunshineoftheismind.files.wordpress.com/2013/03/spiral.jpg)

Un modelo en espiral está dividido en un número de etapas definidas, también llamadas regiones de tareas. Típicamente, hay entre tres y seis regiones de tareas **[9]**.



Las etapas del modelo en espiral pueden ser resumidas de la siguiente manera:

1. Los requerimientos del nuevo sistema son definidos con tanto detalle como sea posible.
2. Un diseño preliminar es creado para el nuevo sistema.
3. Un primer prototipo del nuevo sistema es construido desde el diseño preliminar. Éste es usualmente una versión reducida del sistema, y representa una aproximación de las características del producto final.
4. Un segundo prototipo es desarrollado a partir de un cuádruple procedimiento:
   1. Evaluar el primer prototipo en términos de sus fortalezas, debilidades y riesgos.
   2. Definir los requerimientos del segundo prototipo.
   3. Planear y diseñar el segundo prototipo.
   4. Construcción y pruebas del segundo prototipo.
5. A la opción del consumidor, el proyecto entero puede ser cancelado si el riesgo es considerado muy alto. Factores de riesgo pueden involucrar sobrepasar el costo de desarrollo, cálculos erróneos en el costo de operación, o cualquier otro factor que podría, bajo el juicio del consumidor, resultar en un producto no satisfactorio.
6. El prototipo existente es evaluado de la misma manera que el segundo prototipo, y, si es necesario, otro prototipo es desarrollado a partir de él, de acuerdo con el cuádruple procedimiento expuesto más arriba.
7. Las etapas anteriores son repetidas hasta que el consumidor se encuentre satisfecho con la manera en que el prototipo refinado representa el producto final deseado.
8. El sistema final es construido, basado en el prototipo refinado.
9. El sistema final es evaluado y probado a fondo. Mantenimiento de rutina es llevado a cabo de manera continua para prevenir fallas de larga escala y prevenir falta de disponibilidad. **[11]**

**En qué consiste la adaptación de la metodología**

Como no se tiene un consumidor definido que retroalimente al grupo de trabajo constantemente, se mantendrá el desarrollo del sistema de estabilización del cuadricóptero procurando minimizar los desbalances y oscilaciones en vuelo de éste. Se utilizará un criterio cuantitativo basado en la comparación de las mediciones en funcionamiento del cuadricóptero respecto a umbrales determinados de oscilaciones y balance.

**Justificación de la metodología a utilizar**

La elección de la metodología en espiral se debe principalmente, a que el desarrollo de componentes de software en robótica suele necesitar de un constante proceso de diseño, implementación, pruebas, y rediseño, con el fin de perfeccionar el producto final con cada iteración realizada. Esto porque, aunque pudiera realizarse un análisis previo de los requerimientos del software, es posible que hayan ciertas condiciones propias del sistema robótico que no se hayan podido observar antes del desarrollo del primer prototipo, y sí se puedan observar al realizar un rediseño basado en las fortalezas y debilidades de los prototipos anteriormente desarrollados.

Otra razón de peso por la que se eligió el modelo de desarrollo en espiral es porque el proyecto a realizar es de carácter experimental, y para alcanzar un desarrollo óptimo es muy probable que antes tengan que desarrollarse una serie de prototipos anteriores que permitan al equipo de desarrollo el análisis de las características que pudieran resultar más beneficiosas en el sistema a implementar.

**Referencias bibliográficas**

1. Banzi, M. y Cuartielles, D., Descripción de la plataforma Arduino. Obtenido de <http://www.arduino.cc/>
2. Burka, A. y Foster, S. (2012). *Neato Quadcopters.* Swarthmore College, Pennsylvania, Estados Unidos de América.
3. Dignyu, X., YangQuan, C y Atherton, D. (2007). *Linear feedback control.* Siam.
4. Dulhost, J. *Teoría de control.* Universidad de los Andes. Mérida, Venezuela.
5. González, Alfonso. “*Tutorial para el Diseño de una Red Neuronal con JRedesNeuronales*”. Universidad de Málaga. Escuela de Ingeniería Informática. Málaga, España.
6. Nadales, C. (2009). *Control de un quadrotor mediante la plataforma Arduino.* Tésis de Ingeniería Técnica de Telecomunicaciones, Universidad Politécnica de Catalunya, Barcelona, España.
7. Ogata, K. (2011). *Ingeniería de control moderna.* Quinta edición.Pearson.
8. Pereira, W. *Introducción a las Redes Neurales Artificiales*. Material de la cátedra de Robótica e Inteligencia Artificial. Universidad Católica Andrés Bello. Caracas, Venezuela.
9. Pressman, R. (2001). *Software Engineering: A practitioner’s approach.* Fifth edition. Mc-Graw Hill.
10. Querales y otros (2011). *INTELIGENCIA ARTIFICIAL (Red Neuronal Artificial).* UNEFA Núcleo Lara, República Bolivariana de Venezuela.
11. Rouse, M. (2007). *Spiral model (spiral lifecycle model).* Obtenido de <http://searchsoftwarequality.techtarget.com/definition/spiral-model>
12. Vidyasagar, M. (2010). *A tutorial overview of Control Theory for Non-Engineers.* The University of Texas at Dallas. Texas, Estados Unidos de América.
13. Villar, Jacob (2010). *Diseño e implementación de un neurocontrolador aplicado a una planta de posición y velocidad, Universidad Nacional Del Callao Facultad De Ingeniería Eléctrica Y Electrónica.*
14. Zabczyk, J. (1993). *Mathematical control theory: An introduction.* Birkhäuser. Boston, Massachusetts, Estados Unidos de América.
15. *El controlador PID básico.* Material de la cátedra de Laboratorio Remoto de Automática. Universidad de León. León, España.