ESCUELA DE INGENIERIA MINERA E INDUSTRIAL DE ALMADÉN

TRABAJO FIN DE GRADO

CONTROL DIFUSO E INTELIGENTE DE LA ILUMINACIÓN DE UNA VIVIENDA

UNIVERSIDAD DE CASTILLA - LA MANCHA

GRADO EN INGENIERÍA ELÉCTRICA

Autor: **Jonathan F. Negrete Lindsay**

Director: **Dr. Javier A. Albusac Jiménez**

Curso 2014 - 2015

AGRADECIMIENTOS Y DEDICATORIAS

Me gustaría comenzar este primer apartado agradeciendo a mi director de proyecto, al *Doctor y Profesor D. Javier Alonso Albusac*, por ofrecerme la posibilidad de hacer este Trabajo de Fin de Grado y manifestarme su confianza, colaboración, consejos, apoyo y sobre todo por su interés mostrado en mi proceso de realización, que incluso cuando al principio del curso, cuando iba a comenzar a realizar dicho TFG, me encontraba incapacitado para desplazarme hasta Almadén donde reunirme con él, y me propuso distintos puntos de encuentro donde poder reunirnos.

Agradecer a mi familia y seres más queridos, por estar ahí en cada momento dándome ánimos y confiando en mí siempre. Cabe destacar la figura de mi novia Lourdes, mis hermanos Darren y Brian, y mis padres Fernando y Pauline, por apoyarme y aguantarme este tiempo en mis momentos de frustraciones cuando no salían las cosas, por lo que este trabajo va dedicado a todos ellos.

Dedicar este trabajo para mí es muy importante ya que simboliza que la presentación de él da por finalizada esta carrera y ha sido posible gracias a ellos. Primero a mi novia por darme su apoyo y no dejar que me relaje durante la realización del proyecto e incluso durante toda la carrera. A mis padres por confiar en mí y hacer posible que estos cuatro años de carrera hayan llegado a su fin. A los padres de mi novia, Beni y José, quién incluso me ayudó a la hora de elaborar el Entorno de Pruebas. A mis compañeros de universidad y ya más que amigos (Virginia, David, Clemente, Ramón, etc...), por estar ahí cuando los he necesitado y por hacer mi estancia en Almadén durante estos 4 años, de una forma más que llevadera. A mis abuelos Robert, Margaret y Ramón que aunque ya no están, siempre van conmigo y me acuerdo mucho de ellos; y como no, a mí única abuela que me queda Maruchi y mis tíos españoles Mila, Juan Carlos y José Ramón, por todo su amor que me demuestran día a día. También dedicárselo a mi familia de Escocia e Inglaterra, que aunque no los vea a menudo, sé que están muy orgullosos de mí, por todo lo logrado hasta el momento. Y para terminar, que no menos importante, dedicárselo a mis amigos y hermanos, ya que en los momentos difíciles durante el desarrollo del proyecto, eran capaces de sacarme una sonrisa.

A todos aquellos, les doy gracias por hacer posible llevar a cabo este proyecto.

Almadén, Julio 2015

RESUMEN

El objetivo de este proyecto está basado principalmente en la domótica, ya que consiste en el diseño y construcción de un entorno de pruebas, donde se realizará un control difuso de la iluminación.

Para el control de la iluminación se ha realizado mediante el control de las luces internas y el control difuso de las persianas, teniendo en cuenta la iluminación y temperatura exterior, y la presencia en el interior de la habitación.

Para ello se ha empleado el microcontrolador Arduino y componentes compatibles con él. Una vez construido el entorno de pruebas, se hizo necesario modelar su comportamiento mediante un lenguaje de programación, como es Arduino (lenguaje empleado en los grados de la propia E.I.M.I.A.).

Para definir el comportamiento del entorno de pruebas se ha hecho uso de la Inteligencia Artificial, que está basado principalmente en el modelo matemático de la lógica difusa. De esta manera el comportamiento del sistema de control es adaptativo, dándole una mayor suavidad de control y una respuesta no tan prefijada a la hora de mandar las órdenes.

Además, la realización de este proyecto abre una puerta interesante a los alumnos de la E.I.M.I.A, ya que el entorno de pruebas desarrollado podrá ser empleado para coger ideas o incluso reutilizarlo como un nuevo TFG realizando algunas incorporaciones, como por ejemplo, añadiendo nuevos elementos de control, como el control de climatización o el empleo de energías renovables para usarlo como fuente de alimentación de la instalación. De esta forma, no necesitarán invertir tiempo en solucionar los aspectos básicos de control difuso de las persianas ni de la iluminación y podrán centrarse en el mejoramiento del entorno de pruebas.

Palabras clave: Domótica, Entorno de Pruebas, Inteligencia Artificial, Lógica Difusa, Órdenes.

ABSTRACT

The main objective of this Project is principally based on domotic (home automation), since it consists of the design and construction of a model, where the lighting will be controlled by fuzzy logic.

In order to get the lighting right, we have to monitor both the indoor lights and the fuzzy control of the blinds and shutters, taking into account the temperature and light outside, as well as any human presence inside the room.

To carry out this project, we have used the microcontroller Arduino and compatibles components. Once the construction of the model was set up, we had to create its performance through programming language*, such as the included in Arduino (one of the E.I.M.I.A languages used).

To define the performance of the model we have used Machine Learning, which is mainly based on the Fuzzy Logic mathematic model. In this way, the performance of the control system is adaptable, allowing a smoother control and a less predetermined answer when it comes to giving out commands.

In fact, the execution of this project opens up new doors for other E.I.M.I.A. students, due to the fact that the model developed will be able to be used to get new ideas or even reused it as a new final project once some modifications have been introduced. For instance, adding new control elements, such as climate control, or the use of renewable energies as the power source. Therefore, they will not have to lose any time neither in solving basic Machine learning intelligence aspects for the shutters nor lighting, and they will be able to focus more on improving the model.

Key words: domotic, model, machine learning, fuzzy logic, commands.

ÍNDICE DE CONTENIDOS

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE TABLAS

CAPITULO 1: INTRODUCCIÓN

1. INTRODUCCIÓN

El sector de las viviendas es sin duda uno de los sectores que más ha tardado en la incorporación de las nuevas tecnologías, ya que su incorporación, suponía un encarecimiento muy alto. Otra causa de ello podría ser, que las personas pensaran que la incorporación de elementos innovadores, lejos de parecer ventajas se ven con cierta incertidumbre. Aun así, los seres humanos siempre terminan adaptándose a las nuevas tecnologías, y más aún cuando éstas facilitan las cosas.

Hoy día, las viviendas y los edificios que se construyen, están en continua evolución. Para darse cuenta de ello, sólo hay que echar la mirada unos años atrás, para así comparar las viviendas de mediados del siglo XX con las de hoy en día.

Todo esto ha sido gracias al progreso en paralelo de tres grandes áreas de la tecnología: *electrónica, telecomunicaciones e informática*, que han permitido la incorporación de nuevas tecnologías en el hogar para facilitar el desarrollo de las actividades diarias, lo que actualmente conocemos como **Domótica,** si hablamos de viviendas e **Inmótica** si hablamos de edificios, como por ejemplo: hoteles, hospitales, etc.

El desarrollo de estas tres ramas ha hecho posible que hoy en día se diseñen y apliquen *Sistemas Técnicos* capaces de integrar todos los servicios disponibles mediante una *Automatización Integral.*

Estos avances han desencadenado un aumento considerable de la demanda de instalaciones eléctricas modernas en los últimos años, desplazando el estándar de seguridad, confort y flexibilidad a una nueva dimensión. Lo que hoy día se busca principalmente es la eficiencia energética, es decir, la optimización del consumo de la energía eléctrica y que sea posible realizar muchas de las funciones del hogar con menores implicaciones por parte de los usuarios.

Algunas características que permiten definir a la domótica o vivienda inteligente son:

- Que integre a todos los sistemas eléctricos o electromecánicos.
- Que pueda actuar con condiciones ilimitadas, ligadas o no entre sí.
- Que tenga memoria y noción temporal.
- Que tenga capacidad matemática avanzada.
- Que sea modificable de manera sencilla.
- Que disponga de capacidad de autocorrección.
- Que se comunique con el usuario de forma agradable.

En un edificio inteligente encontraremos: sensores, actuadores electromecánicos, infraestructura de cableado, procesador u ordenador personal, que permiten la interrelación y el funcionamiento del hardware y el software de control.

Las características básicas que debe ofrecer un *Sistema de la Gestión Técnica* de un edificio son:

- ***Integración:*** debe ser capaz de implicar a diferentes componentes o equipos pertenecientes a diferentes áreas de la gestión del edificio, permitiendo el intercambio de información entre ellos e interaccionando tanto con los demás equipos como con el usuario.
- ***Flexibilidad:*** debe posibilitar la adaptación de la instalación a necesidades futuras.
- ***Estructura Modular:*** para evitar que el mal funcionamiento de una parte de la instalación afecte a las demás.
- ***Reprogramación:*** debe permitir al usuario la modificación de unos determinados parámetros como, por ejemplo, los horarios de actuación, temperatura de referencia, grados de iluminación, etc.
- ***Facilidad de utilización:*** es de gran importancia que la interacción o interface del usuario sea fácil e intuitiva de utilizar, por ejemplo, mediante interruptores, pulsadores, mandos a distancia, etc.

También cabe decir que las instalaciones automatizadas se pretenden cubrir mediante las siguientes cuatro áreas de servicios:

- **Área de Gestión del Confort:** Accionamiento automático de persianas, riego automático o telemando vía infrarrojos.
- **Área de Gestión de la Seguridad:** Detección de riesgos técnicos (incendios, fugas de agua y gas, etc.), simulación de presencia o visualización de accesos con cámaras de vigilancias.
- **Área de Gestión de la Energía:** Programación de la calefacción y el aire acondicionado o información del consumo eléctrico.
- **Área de Gestión de las Comunicaciones:** Red interior de distribución de las señales de audio, vídeo e informática o videotexto.

Llegados a este punto, el proyecto, motivado por la reciente introducción de estas tecnologías en el hogar, me ha llevado a escoger un proyecto relacionado con éste ámbito. Por lo que mi *Trabajo de fin de Grado* ha consistido en el **Diseño de dispositivos empotrados multisensoriales con Arduino para edificios inteligentes** en mi caso en el ***Control difuso e Inteligente de la iluminación de una Vivienda***, controlados principalmente por sensores de temperatura luminosidad y presencia.

Lo que se busca con este proyecto, es que, la iluminación interior se controle de forma gradual y automática, y que las persianas estén motorizadas y dispongan de un autocontrol, gracias a los sensores incorporados (LDR y PIR), esto último se llevará a cabo empleando la **lógica difusa**, que no es otra cosa más que, a la hora de controlar algo sea más sencilla para el usuario, ya que no necesitará de conocimientos técnicos para su manejo, debido a que no usan datos numéricos, sino que asocia un rango de números a una etiqueta lingüística, como por ejemplo, en este caso, *frío, calor, muy oscuro, muy claro, etc.*

En la actualidad se está utilizando mucho los microcontroladores integrados, ya que les permiten conectar entradas y salidas sin necesidad de cablear el microcontrolador ni soldar los terminales. Existes varios tipos de estas placas, una de las que más se están utilizando son las placas *Arduino*, ya que su bajo coste lo hace accesible a estudiantes de ingeniería y hace posible la realización de trabajos con microcontroladores.

Para poder demostrar prácticamente dicho funcionamiento, se va a llevar a cabo la construcción de un entorno de pruebas, manejado con el microcontrolador **Arduino**, sobre los que se podrán simular las múltiples hipótesis.

La inteligencia del entorno de pruebas, no es más que la forma de programar que se le da al sistema para que actúe de una forma determinada ante unas situaciones específicas. Actualmente en las viviendas, se están llevando a cabo la *inteligencia artificial (IA)*, dotándolas de la capacidad de aprender los hábitos de consumo energético de los habitantes, así como la regulación óptima o para utilizar electrodomésticos o regular la climatización e iluminación de la vivienda.

Por último, decir que para la realización de dicho proyecto, los conocimientos se han ido adquiriendo a lo largo de estos 4 años en la obtención del título de Grado en Ingeniería Eléctrica, en la Escuela de Ingeniería e Industrial de Almadén, mediante las siguientes asignaturas:

- **Informática** (elaboración de los códigos o programas de las persianas y luces)
- **Sistemas de Comunicación en Edificios** (Para la interacción entre persianas y luces)
- **Máquinas Eléctricas** (Para saber el funcionamiento de los motores de las persianas)
- **Ciencia de los Materiales** (Elección de que materiales usar)
- **Control de Máquinas Eléctricas** (Saber cómo controlar el motor)
- **Domótica** (comprender en que consiste la Domótica, conocimiento de los distintos sensores y actuadores, y una introducción a cómo manejar el Arduino)
- **Luminotecnia** (Tener un conocimiento de las distintas luminarias, para así poder escoger la que mejor se adapte al Sistema utilizado)
- **Instalaciones Generales de Baja Tensión** (Saber cuál es el cableado reglamentario en una vivienda)
- **Proyectos en la Ingeniería** (aprender a cómo estructurar proyectos)
- **Instrumentación Industrial y Medida** (conocimiento del funcionamiento de los sensores)
- **Ofimática aplicada a la Ingeniería** (Para tener mejor manejo con el Microsoft Word y así poder redactar mejor el proyecto)

CAPITULO 2: OBJETIVOS

2. OBJETIVOS

La realización de este proyecto está enfocada a la aplicación de las competencias generales de la titulación de Ingeniería Eléctrica en la construcción de un *entorno de pruebas* y en la aplicación de lenguajes de programación a casos de ingeniería.

Se trata de la **construcción de un entorno de pruebas** sobre el que poder simular el funcionamiento automatizado de las persianas y control de la intensidad de las luces en función de la iluminación y temperatura exterior, y solamente se accionará cuando haya presencia. Todo esto se conseguirá mediante el empleo de sensores y actuadores, que serán controlados por el microcontrolador **Arduino.**

Los sensores serán los encargados de, captar la cantidad de luminosidad (LDR) y temperatura (termopar) que hay en el exterior y detectar cuando la habitación está o no ocupada (mediante un *detector de presencia o movimiento* o también conocido como *PIR*). Para poder determinar la temperatura en el entorno de pruebas se ha optado por el empleo de un potenciómetro, mediante el cual se simulará la temperatura exterior, ya que iba a resultar muy complicado poder simular distintas hipótesis con un sensor de temperatura, termopar. Y para poder subir o bajar las persianas se va a utilizar microservomotores.

Para cumplir los objetivos generales expuestos anteriormente es necesario satisfacer los siguientes puntos:

1) ***Diseño y construcción desde cero de una maqueta o entorno de pruebas, lo más económico posible:*** Se deberá elegir los microcontroladores y componentes que sean más económicos, pero que a su vez tengan buenas características y se consigan los objetivos marcados, es decir, que haya un equilibrio entre calidad precio.
2) ***Modelación de un sistema difuso para definir el comportamiento de las persianas:*** El sistema difuso o razonamiento de las persianas, estarán basados principalmente en el modelo matemático de la *lógica difusa*. Con esta forma de control, se busca que éste se comporte de forma distinta ajustada a la situación actual en la que se encuentre. Es decir, **le da una mayor suavidad de control y una respuesta no tan prefijada** a la hora de mandar las órdenes.

Para poder controlar tanto las persianas, como las luces interiores, habrá que emplear lenguajes de programación, en este caso, sobre el Arduino (microcontrolador), que se llevará a cabo mediante un ordenador. Una vez introducido el código al microcontrolador Arduino, se va a convertir en un sistema autosuficiente, es decir, no se va a necesitar tener conectado el Arduino al PC para su funcionamiento. Sólo se va a necesitar de un PC, por si surge algún problema o se quiera modificar el funcionamiento del sistema.

El **uso de la lógica difusa** es principalmente por tener la posibilidad de procesar situaciones que implican vaguedad como "subir mucho", "muy oscuro", "frio" y **no tener un comportamiento lineal.** Por lo que se podrá conseguir un comportamiento más cercano para el usuario, ya que no trabajará con números, sino que trabajará con términos lingüísticos, que son entendibles por la mayoría de los usuarios. Gracias al control difuso, el entorno de pruebas será capaz de reconocer órdenes que forman parte del comportamiento en base a las condiciones ambientales. Por ejemplo, *si tenemos una situación donde está haciendo mucho calor y además hace un día muy nublado (oscuro), las persianas se encontrarán un poco subidas y las luces encendidas. Y si de repente, se comienzan a despejar el día, es decir, pasa a ser un día claro, las persianas se subirán mucho y las luces se apagan.*

Con el empleo de la lógica difusa, aparte de conseguir un sistema sencillo para el uso del usuario, se consigue crear un sistema de acuerdo a sus gustos o preferencias, mediante el uso de términos lingüísticos.

Para el ***control de persianas*** como ya se ha mencionado anteriormente, se va a emplear la *lógica difusa* para así obtener un mejor control. Con el empleo de un sistema difuso, aparte de todas las ventajas mencionadas anteriormente, en cualquier momento se podrán cambiar el comportamiento del sistema, cambiando los rangos de valores sobre los que se trabajarían de una forma muy sencilla sin tener que entrar en la parte compleja de la programación, es decir, si se decide modificar la regla "*Mucho calor*", que hace referencia a 30-35ºC, ampliándolo hasta los 40º, dicha regla pasaría a cubrir los casos comprendidos entre 30 y 40ºC. Por lo que es una gran ventaja, ya que se puede adaptar en cualquier momento de una manera muy sencilla a las preferencias de cada persona.

Para el ***control de las luces*** como ya se ha comentado anteriormente, se encenderán solas mediante el empleo de *detectores de presencia,* y solamente cuando detecten a un ser vivo y sea necesario, y por el contrario, se apagarán cuando deje de detectar presencia, es decir, cuando abandonen la habitación. Para el encendido de las luces, también se tendrá en cuenta la iluminación que tengamos en el momento de presencia, ya que las luces iluminarán de forma gradual, es decir, iluminarán en función de la cantidad de luz que se requiera en dicho instante, por lo que se estará consiguiendo un ahorro energético.

Una vez que se ha explicado cómo se va a realizar el control de las persianas y de las luces, se va a explicar cómo se va a distribuir el entorno de pruebas. Se van a crear dos habitaciones en forma de L, y separadas entre ellas por un muro, para así poder simular a la vez 2 sistemas simultáneamente, ya que, se puede dar el caso de que una zona de la maqueta esta iluminada y la otra no, por lo que funcionarían indistintamente.

Con este proyecto se busca una mejora de eficiencia energética, ya que si se elabora el sistema correctamente, siempre se empleará la iluminación necesaria en todo momento, utilizando siempre que se pueda la luz natural.

CAPITULO 3: ESTADO DEL CONOCIMIENTO

3. ESTADO DE CONOCIMIENTO

3.1. Historia

La antesala a la Edificación inteligente fue la **Automatización.** Ascensores, lavadoras, televisión, etc. fueron algunos de los ejemplos de progreso y comodidad. En la década de 1970, la mayoría de los automatismos estaban basados en una electromecánica compleja, basada en circuitos electrónicos exclusivos de cada sistema, que posteriormente serían mejorados y se buscaría hacerlos siempre más pequeños, aunque su capacidad de trabajo era limitada, ya que solo obedecía a una única condición de funcionamiento.

A principios de 1980, se produjo la aparición del **Microprocesador**, que permitió la interrelación de sistemas automáticos para conseguir así Edificios Inteligentes. Los primeros casos que se buscaron mejorar en las viviendas, fueron los relacionados con el ahorro y el confort, ya que son los factores prioritarios. Los primeros pasos se dieron para instalar sistemas de climatización gobernados por electrónica microprocesada por autómatas y posteriormente por ordenadores. Surge así el *Control Inteligente de la Climatización*, que en su momento se le llamó: *Gestión Centralizada de la Climatización.*

El principio de la década de 1990, puede considerarse como la fecha de entrada de los *Sistemas Inteligentes*, aunque por separado. Empiezan a ser frecuentes los Sistemas Inteligentes de Seguridad Anti-intrusión, los Sistemas Inteligentes contra Incendio, de Iluminación, de Climatización, de Aparcamiento, etc.

Aunque los edificios que incorporen alguno de estos Sistemas Inteligentes no se les debería llamar Edificios Inteligentes, ya que para poder ser llamados como Edificios Inteligentes, los sistemas deben de estar relacionados entre sí y deben poder intercambiarse datos y parámetros de actuación. Es decir, deben funcionar de *forma integrada.*

Un ejemplo de **funcionamiento integrado** lo tenemos, en el simple caso de cuando un sensor detecta que hay presencia en una habitación, no solo activarán las luces de dicha habitación, sino que también se elevarán las persianas, y tanto la activación de las luces y de la persiana, dependerán de más sensores (LDR y termopar).

Una característica fundamental del Edificio Inteligente es su **Preinfraestructura Inteligente,** o también conocido como **Aprendizaje automático,** que significa que es

fácilmente reprogramable e incluso puede estar programado para autoprogramarse, y que, estará diseñado poder crecer funcionalmente, ampliar su ámbito de actuación con facilidad y prevenir agresiones exteriores o interiores que puedan alterar su ininterrumpible buen funcionamiento.

Los requisitos para la catalogación de Edificio Inteligente son difíciles de precisar. Pueden, sin embargo, agruparse en tres grandes niveles.

- Sistema Técnico Utilizado
- Usuarios del edificio
- Aspectos Constructivos

3.2. Edificio Inteligente

El nuevo *RD 346/2011, del 11 de marzo*, aprueba el Reglamento regulador de las infraestructuras comunes de telecomunicaciones para el acceso a los servicios de telecomunicación en el interior de las viviendas, define el Edificio Inteligente como el lugar donde, mediante convergencia de infraestructuras, equipamientos y servicios, son atendidas las necesidades de sus habitantes en materia de confort, seguridad, ahorro energético e integración medioambiental, comunicación y acceso a contenidos multimedia, teletrabajo, formación y ocio, que requiere de un conjunto de infraestructuras y equipamientos que faciliten el acceso a muchos servicios existentes y la incorporación de otros que llegarán en el futuro.

3.2.1. Grupos y niveles de servicios del Edificio Inteligente

Un Edificio Inteligente se puede clasificar en varios niveles o grupos:

- ***Comunicaciones***

Proporciona el medio de transporte de la información (voz, datos o imagen) entre el usuario y los distintos dispositivos y servicios, o entre distintos dispositivos del hogar.

- ***Eficiencia energética***

Estará diseñado para una gestión inteligente de la climatización y la iluminación, así como el resto de los posibles consumos eléctricos de la vivienda.

- ***Seguridad***

Permite controlar de forma local o remota, cualquier zona de la vivienda y cualquier incidencia relativa a la seguridad de la vivienda y bienes de las personas.

- ***Control del entorno***

Permite un control integrado de los diferentes sistemas que utilizan los servicios generales de una vivienda, y satisface las necesidades de seguridad, eficacia energética y confort al usuario. Favoreciendo así que la vivienda alcance el grado máximo de flexibilidad, economía, confort, seguridad para sus ocupantes y una comunicación eficaz.

- ***Acceso interactivo a contenidos multimedia***

Proporciona la opción de poder acceder de una forma interactiva a contenidos como archivos de texto, documentos, imágenes, páginas web, gráficos, etc. utilizados para proporcionar y comunicar información vía web.

- ***Ocio y entretenimiento***

Permite a las personas disfrutar de sus ratos libres de forma pasiva o interactiva, mediante contenido multimedia al que se puede acceder desde un equipo reproductor/visualizador. Dicho contenido puede encontrarse en el hogar o bien recibido de fuentes externas. El objetivo es que sean capaces de tomar decisiones y adelantarse a las necesidades de los usuarios asistiéndoles en las tareas cotidianas.

3.2.2. Instalaciones del Edificio Inteligente

Las infraestructuras comunes de telecomunicaciones (ICT) consiguen que las tecnologías entren en el hogar y proporcionen un soporte físico y lógico de los nuevos servicios mencionados anteriormente.

En Edificios Inteligentes, no basta con dotar a las viviendas de una serie de equipamientos que proporcionen el confort, seguridad, ahorro energético, accesibilidad, etc., sino que también han de estar interconectados entre todos ellos, para facilitar así su gestión y control, con el fin de poder hacerlo desde fuera del hogar, bien sea de una forma personal o a través de servicios ofrecidos por empresas especializadas.

Hay que destacar que las **comunicaciones** son el elemento que posibilita los nuevos servicios de control, tanto dentro como fuera de la vivienda.

El acceso de las redes de los distintos operadores a la vivienda, posibilita la existencia de líneas de banda ancha y, en consecuencia, la posibilidad de que estén operativos. Además, la existencia en la edificación de instalaciones internas propias, permite el desarrollo de servicios como la televisión digital terrestre (TDT).

Esto supone que, para ser calificada como ¨Hogar digital¨ o ¨Vivienda Inteligente¨ debe disponer, además de una **red interna de comunicaciones con cableado estructurado (RAD)**, de una **red de gestión, control y seguridad (RGCS).**

Definimos la ***RGCS*** como una red de datos adicional que representa soporte a un conjunto de servicios específicos de la vivienda y puede ser parcialmente soportada por otros medios de transmisión además del cableado.

La interconexión entre ambos tipos de redes se consigue gracias a la pasarela residencial que actúa como elemento integrador, habilitando la mayoría de los servicios en la vivienda. Por tanto, se deberá dotar a la vivienda de las infraestructuras necesarias, para poder considerarlo como Vivienda Inteligente.

3.2.3. Servicios del Edificio Inteligente

En este apartado se recogen, dentro de los grupos anteriormente definidos, los servicios de una forma individualizada.

- ***Seguridad.***

a) Alarmas técnicas de incendio y/o humo.
b) Alarmas técnicas de gas.
c) Alarmas técnicas de inundación.
d) Alarmas de Intrusión.
e) Alarma de Pánico SOS.
f) Control de accesos: Vídeo – portero.
g) Control de accesos: Tarjetas de proximidad.
h) Videovigilancia.
i) Teleseguridad: Central Receptora de Alarmas.

- ***Control del Entorno.***

a) Simulación de presencia.
b) Telemonitorización.
c) Telecontrol.
d) Automatización y control de toldos y persianas.
e) Creación de ambientes.
f) Control de Temperatura y climatización.
g) Diagnóstico y mantenimiento remoto.

- ***Eficiencia Energética.***

a) Gestión de dispositivos eléctricos.
b) Gestión de electrodomésticos.
c) Gestión de circuitos eléctricos prioritarios.
d) Monitorización de consumos.
e) Control de consumos.
f) Control de iluminación.

- ***Ocio y entretenimiento.***

a) Radio difusión Sonora (AM, FM, DAB).
b) TDT.
c) Televisión por satélite/cable.
d) Video bajo demanda (VOD).
e) Distribución multimedia/multiroom.
f) Televisión IP.
g) Música on-line.
h) Juegos on-line.

- ***Comunicaciones.***

a) Telefonía básica.
b) Acceso a Internet con banda ancha.
c) Red de Área Doméstica (Cableado UTP).
d) Telefonía IP.
e) Videotelefonía.

- ***Acceso Interactivo a Contenidos Multimedia.***

a) Tele-asistencia básica.
b) Videoconferencia.
c) Tele-trabajo/Tele-educación.

3.3. Terminología que comprenden una Instalación Domótica

En la ITC-BT-51 del Ministerio de Industria, llamada como ¨*Instalaciones de sistemas de automatización, gestión técnica de la energía y seguridad para viviendas y edificios*¨, se definen los diferentes términos que forman parte de una instalación domótica.

3.3.1. Nodo

Cada una de las unidades del sistema capaces de recibir y procesar información comunicando, cuando proceda con otras unidades o nodos, dentro del mismo sistema. Por ejemplo sería un ordenador o un autómata que reciba señales de sensores de temperatura, humedad, luz, etc. y procese dichas señales para dar órdenes a los sistemas que actúan sobre la climatización, la iluminación, las persianas o el sistema de riego. Asimismo, podría estar conectada a la red de tecnologías de la información (RTI) para recibir información de las previsiones meteorológicas.

3.3.2. Actuador

Es el dispositivo encargado de realizar el control de algún elemento del Sistema, como por ejemplo, electroválvulas (suministro de agua, gas, etc.), motores (persianas, puertas, etc.), sirenas de alarma, reguladores de luz, etc.

3.3.3. Dispositivo de Entrada

Sensor, mando a distancia, teclado u otro dispositivo que envía información al nodo. Serían los medidores de temperatura o humedad, las células fotoeléctricas, etc. que envían información al nodo.

3.3.4. BUS (Binary Unit System)

Línea de intercambio de datos a la que se pueden conectar gran cantidad de componentes, permitiendo la comunicación entre éstos. Los componentes que se pueden conectar, pueden ser: nodos, actuadores o dispositivos de entrada.

3.3.5. Pasarela Residencial (Residential Gateway)

Elemento de conexión entre diferentes redes de una vivienda o edificio (control domótico, telefonía, televisión y tecnologías de la información) a una red pública de datos, como por ejemplo Internet, efectuando en su caso, la adaptación y traducción entre diferentes protocolos. La red de control del sistema domótico puede estar o no conectada a la pasarela residencial; en el caso de que esté conectada, el nodo puede desempeñar también las funciones de pasarela residencial.

3.3.6. Punto de Acceso al Usuario (PAU)

Es el elemento en el que comienza la red interior de telecomunicación del domicilio del usuario, que permite la delimitación de responsabilidades en cuanto al origen, localización y reparación de averías. Se ubica en el interior del domicilio del usuario.

3.3.7. Protocolo

Lenguaje de comunicación entre periféricos con objeto de establecer la transmisión de datos con un sistema central o entre sí, de forma ordenada.

3.3.8. Radiofrecuencia

Transmisión de señal sin requerir de un medio físico, ni de alineación libre de obstáculos entre el emisor y el receptor, generalmente de frecuencia comprendida entre 3 kHz y 3 GHz.

3.3.9. Topología

Término utilizado para definir la estructura de la red y la configuración del sistema.

3.4.Sistemas de control

Los **sistemas de control** son aquellos sistemas capaces de recoger información de unas entradas o **sensores**, procesarlas y emitir órdenes o comandos a las salidas o **actuadores**, con el fin de conseguir la automatización programada.

La topología de los sistemas de control, nos muestra como estarán intercomunicados entre ellos, los distintos elementos del sistema. En la actualidad se pueden usar tres tipos de topologías:

3.4.1. Sistema de control centralizado

El sistema de control centralizado es el elemento encargado de recoger toda la información proporcionada por los sensores distribuidos en los distintos puntos de control de la vivienda o edificio inteligente, procesarla y generar órdenes que ejecutarán los actuadores.

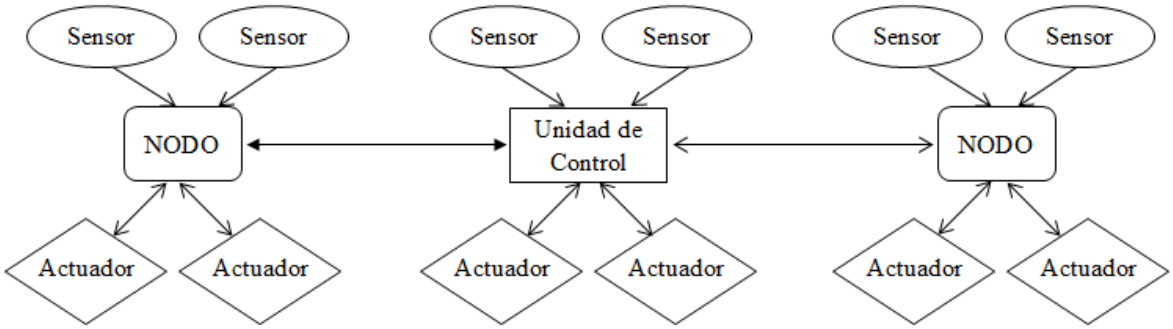


Ilustración 1: Sistema de control centralizado

Uno de los inconvenientes que se ha de resaltar en este tipo de comunicaciones, es que toda la instalación está dirigida o gobernada por una única unidad de control, y en el peor de los casos, si ésta dejase de funcionar, los nodos que están conectados a la unidad de control dejarían de recibir las instrucciones. Lo que conlleva a que la instalación dejaría de funcionar.

3.4.2. Sistema de control descentralizado

Son aquellos sistemas donde existe más de una unidad de control, en el que cada uno de ellos dispone de la suficiencia o capacidad de tratar cualquier información. Es decir, cada unidad de control actúa de forma independiente.

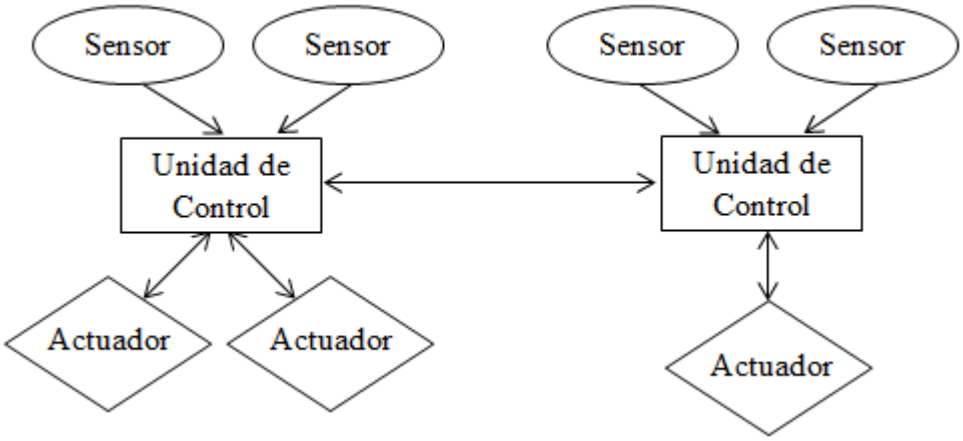


Ilustración 2: Sistemas de control descentralizado

Al contrario que ocurre en los Sistemas centralizados, ya no tendremos ninguna Unidad de Control Principal, es decir, que en el caso de que se produjese algún fallo en alguno de los nodos o unidades de control, este fallo sería local y no afectaría al resto de la instalación. Y también mencionar, que las unidades de control se colocan lo más próximo posible a los actuadores que quieran ser controlados.

Otro aspecto a destacar es la comunicación entre los nodos o unidades de control, debido a que están unidos mediante buses de comunicación y otro medio físico, por lo que se debe de establecer un *protocolo de comunicación entre las diferentes unidades de control*, con la finalidad de poder intercambiar datos, sin que haya errores.

1.4.3. Sistema de control distribuido.

Este sistema es una mezcla de los dos anteriores, en el que los distintos unidades de control están unido mediante buses de comunicación o cualquier otro medio, como wifi, bluetooth o radiofrecuencia, con el fin de intercambiar datos.

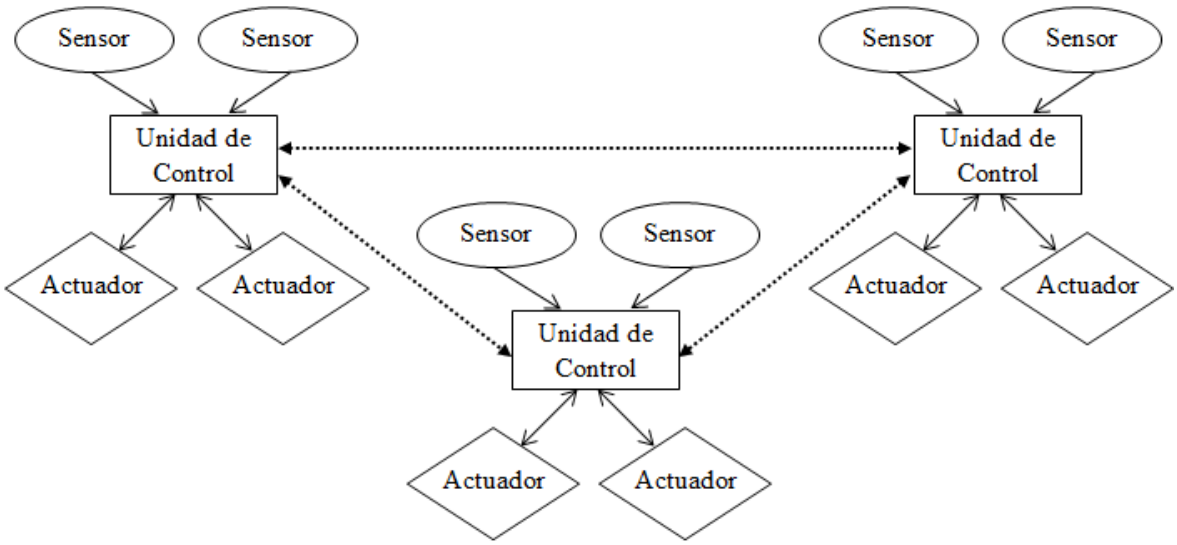


Ilustración 3: Sistema de control distribuido

Al igual que pasa en los Sistemas Descentralizados, las unidades de control, se colocan lo más próximo a los actuadores que se quieran controlar.

3.5.Sistemas de comunicación

Dependiendo del tipo de enlaces que se utilicen en el sistema domótico se pueden clasificar en dos categorías:

- ***Cableados:*** Sistemas en los que todos los sensores, actuadores y controladores están conectados entre sí con cables. Se pueden catalogar los sistemas cableados en función de si el cableado que se utiliza es exclusivo o compartido con otro sistema:
    - **Cableado exclusivo:** en los hogares normalmente no existe ningún otro cableado que no sea el de la tensión eléctrica a 220V o el telefónico, excepto en obras nuevas que pueden disponer de cableado para transmisión de música. Cuando hablo de cableado exclusivo me refiero a tener un sistema de cableado solo para el sistema domótico, sin compartirlo con ningún otro sistema. Esto implica cablear toda la vivienda para añadir un nuevo bus de comunicaciones que permita el envío de datos con los dispositivos domóticos. Esto resulta práctico cuando se aplican a viviendas de obra nueva mientras se construyen, ya que no supone un gran coste adicional añadir tres cableados en vez de dos.
    - **Cableado compartido:** Cuando no hay posibilidad de cablear la vivienda de nuevo se puede optar por esta solución, que consiste en utilizar un cableado ya existente y compartirlo. Concretamente se suele utilizar la línea eléctrica, la tecnología se llama Power Line Carrier (PLC), funciona mediante la modulación de una onda portadora cuya frecuencia oscila entre los 20 y 200 kHz inyectada directamente en el cableado eléctrico.
- ***Inalámbricos:*** Cuando no es posible cablear la vivienda y tampoco utilizar la tecnología PLC se puede optar por utilizar tecnologías inalámbricas como WiFi, Bluetooth, infrarrojos o radiofrecuencia. Estas tecnologías permiten que el dispositivo domótico no necesite estar en un lugar fijo, ya que puede comunicarse con el sistema desde cualquier lugar dentro del alcance del receptor.

3.6.Topologías de Red

Los criterios de diseño tenidos en cuenta en el momento de definir y dimensionar una Red o Sistema están muy relacionados con factores tales como: el coste, la modularidad, la fiabilidad, la flexibilidad, la rapidez etc... y también con los criterios de diseño del propio edificio y con el Sistema Técnico que se implante.

Las Topologías básicas son: ***Redes en Línea o Bus***, **Redes en Estrella** y **Redes en Anillo**, aunque existen otras variaciones realizadas sobre la combinación de estos tres tipos básicos: ***Redes en Árbol, Redes en Malla, Redes Mixtas*** etc...

3.6.1. Red en Estrella

Todos los elementos que configuran el Sistema están unidos entre sí a un Controlador Principal, que es el que realiza las funciones de control y supervisión. Se denomina también, por este motivo, a la Topología en Estrella: **Topología Centralizada.**

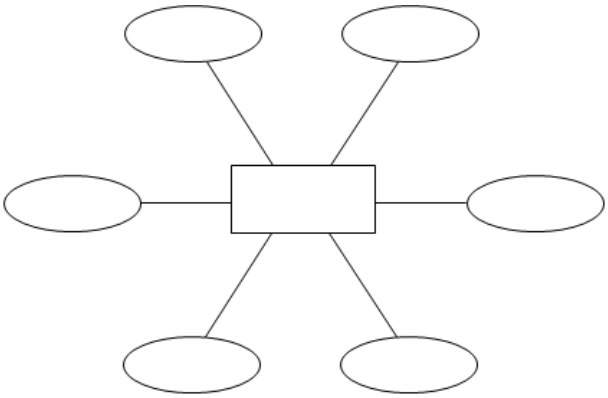


Ilustración 4: Red en estrella

Las principales **ventajas** al utilizar una Topología en Estrella son:

- Es muy fácil añadir nuevos elementos a la Red, ya que solo hay que conectarlos al Controlador Principal.
- Permite independizar e identificar rápidamente las averías, multiplica la velocidad de transmisión de la información, incrementa la seguridad del edificio y posibilita su expansión mediante la creación de subnúcleos.
- Un fallo en un elemento de la instalación no afecta al resto, excepto si falla el Controlador Principal.

Los **inconvenientes** son:

- Se necesita mucho más cable para la instalación y, debido a ello, ésta se hace más compleja ante la necesidad de tener identificado cada cable.
- Un fallo en el Controlador Principal provoca un fallo general en toda la instalación.
- Es propensa a congestionarse fácilmente.

3.6.2. Red en Anillo

Todos los componentes del Sistema se interconectan formando un anillo, es decir, los componentes de la Red se instalan en serie formando un camino cerrado en el que los extremos del cable se cierran.

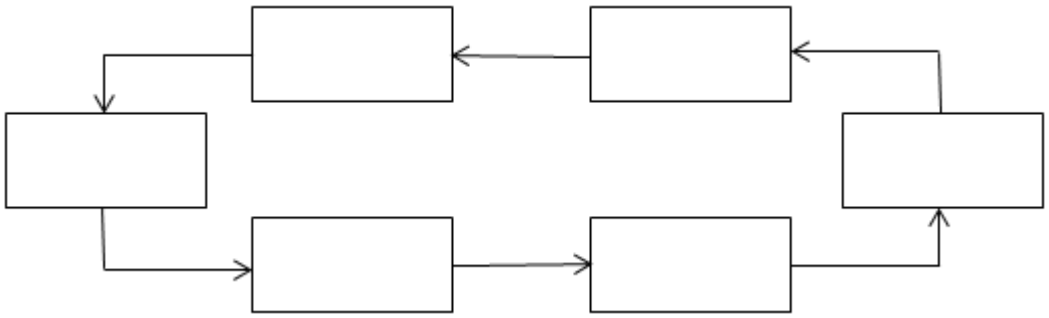


Ilustración 5: Red de anillo

La información que se transmite pasa por todos los componentes existentes entre el emisor y el receptor. Este Sistema es poco utilizado en los Sistemas Técnicos de automatización de edificios ya que su principal inconveniente es el mal funcionamiento de toda la instalación en el momento en el que un componente deja de funcionar.

La línea de transmisión es unidireccional, ya que los datos que se transmiten a la Red viajan de nodo a nodo en un único sentido. Esta Topología presenta la **ventaja** de que el control del Sistema es sencillo, pero presenta los siguientes **inconvenientes:**

- Es una Topología muy vulnerable a fallos. El retardo es variable, en función de la situación de los nodos origen y destino.
- Es necesario utilizar un cable de mayor longitud que en la Topología en Bus.
- Añadir nodos es más complicado y hay que interrumpir el funcionamiento de la Red

3.6.3. Red en Línea o Bus

La Red en Bus consiste en una línea de comunicación que comparten todos y cada uno de los elementos del Sistema. Todos los equipos mandan o reciben sus mensajes a través del Bus, existiendo un sistema de direccionamiento que permite la emisión o recepción del mensaje mediante dos direcciones: dirección del elemento que envía y dirección del elemento que recibe.

El concepto de Bus surge cuando a través de esa línea se transporta información. La aplicación correcta del concepto Bus no implica topología en línea, ya que Bus es todo cable o circuito destinado a transportar información. Sin embargo, generalmente al hablar de Buses, se hace referencia a topologías en línea.

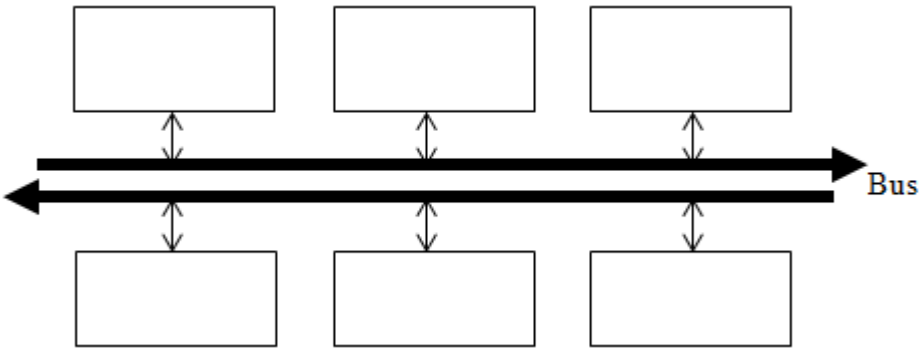


Ilustración 6: Red de Bus

Las principales **ventajas** de la topología en Línea son:

- Es muy fácil la instalación y reconfiguración del Sistema en Bus, es decir, es muy fácil insertar y eliminar componentes.
- No es necesario, (aunque algunos Sistemas lo tienen), un elemento que controle el Bus, ya que, perfectamente, pueden realizar el control cada uno de los componentes que forman el Sistema.
- El funcionamiento defectuoso de un componente no afecta al resto del Sistema.
- La longitud del cable a instalar se minimiza.
- La velocidad de transmisión de información es, en general, suficiente y es posible compartir distintos tipos de información, (voz, datos, vídeo, control, ect...).

Los **inconvenientes** son importantes:

- Si se produce un fallo o avería en alguno de las dos líneas del bus, ya que la transmisión dejaría de ser factible y todos los sistemas conectados al bus quedarían incomunicados. Para detectar las averías o interrupción de un hilo en una parte de la línea, los Buses incorporan en ambos extremos unos **Terminadores de Línea.**
- Es Fácilmente saboteable e inutilizable.
- Si hay exceso tráfico de información, como el camino es único, el flujo de información se ralentizaría, surgiendo atascos y pudiendo llegar a colapsarse.
- Es necesario que los nodos tengan cierto grado de inteligencia y es necesario, también, asegurar mecanismos de control para evitar que dos elementos intenten acceder en el mismo instante a la Red, lo que hace que los Protocolos de Comunicación tengan que ser más sofisticados que en otras Topologías de Red.

Por tanto, este tipo de Sistema es adecuada cuando el número de puntos a gobernar es reducido o cuando el nivel de seguridad que se pretende establecer no es muy importante o cuando la velocidad de transmisión de información no necesita ser muy alta (ej. Sistemas de Climatización normales de un edificio).

3.6.4. Red en Árbol

Puede considerarse una Topología híbrida de las anteriores, sobre todo de las Topologías en Bus y en Estrella. Las ventajas y desventajas de penderán de la configuración final establecida

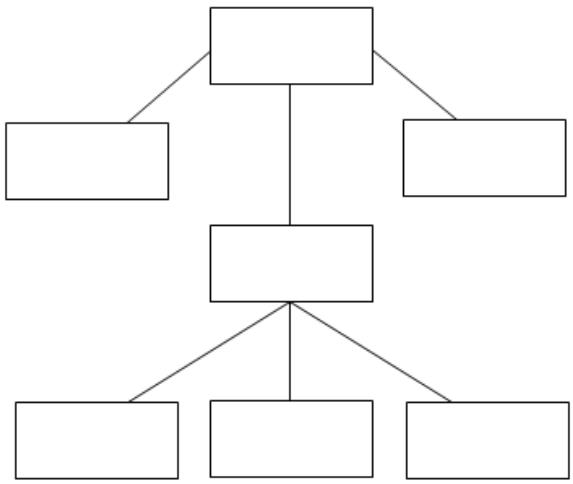


Ilustración 7: Red en Árbol

3.6.5. Red en Malla

Este tipo de topología que se basa en la unión de todos los nodos entre ellos a través de diferentes medios físicos, pudiendo hacer llegar la información a un nodo por más de un camino, como podemos observar en la *Figura 8*.

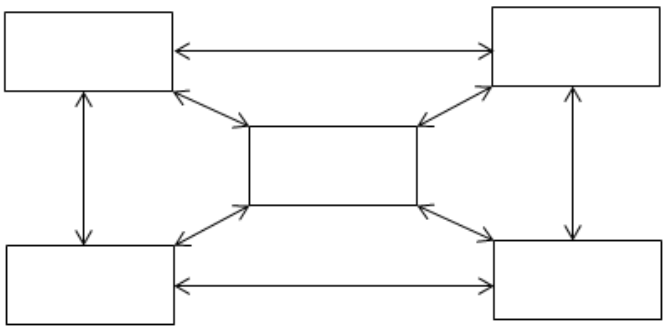


Ilustración 8: Red en malla

Por lo que, en el caso que se produjese un fallo en alguno de los distintos caminos, nuestro sistema no dejaría de funcionar, ya que existiría otro camino por el que hacer llegar la información.

Alguno de los **inconvenientes** sería:

- Este tipo de topología conlleva un elevado coste, debido a tanto cableado.
- En caso de avería, la dificultad que supone encontrar dicha avería.

3.7.Elementos de un Sistema Domótico

Las principales partes de un sistema domótico son: *la unidad de control, los sensores y los actuadores.*

3.7.1. Unidad de Control.

Es la parte del sistema domótico encargada de **almacenar y transmitir toda la información** de los distintos elementos de control por los ramales de comunicaciones, para así hacer llegar dicha información a cada uno de los dispositivos destinatarios. También es la encargada de **gestionar todos los diferentes estados** de los dispositivos.

A las unidades de control también se les conoce como ***nodos***, y tienen una comunicación directa con todos los dispositivos que están conectados en el mismo ramal de comunicación.

Los *nodos* llevan incorporado en su hardware unos microcontroladores que se encargan de compilar y ejecutar el programa efectuado por el usuario

Nos podemos encontrar distintos tipos de Unidades de control, que dependerá del tipo de sistema domótico que se elija para la instalación de la vivienda (sist. basados en corrientes portadoras, basados en autómatas programables, sist. domóticos inalámbricos, etc.).

- ***Tipos de señales***

Las señales de comunicación entre los dispositivos que forman parte de un Sistema de control pueden ser **Analógicas** o **Digitales**. En ocasiones, las señales de entrada y salida pueden no ser iguales, es decir, la entrada puede ser analógica y la salida digital o viceversa.

Las **señales analógicas** son aquellas en las que el nivel o la amplitud de la señal varían de forma continua con el tiempo. La señal puede tomar infinitos valores comprendidos entre dos niveles, denominados máximo y mínimo y pueden ser Periódicas o No Periódicas.

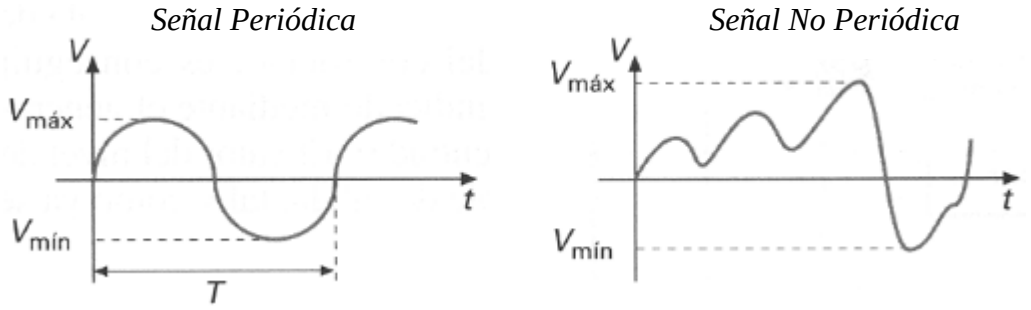


Ilustración 9: Señales Analógicas

Las **señales digitales** son aquellas que varían a lo largo del tiempo de manera discreta. Estas señales solo adquieren un número finito de valores. Un tipo característico de señal digital es la *Señal Binaria*, en la que se producen dos estados asociados a dos valores: Cero y Uno. Igualmente, las señales digitales pueden ser Periódicas o No Periódicas.

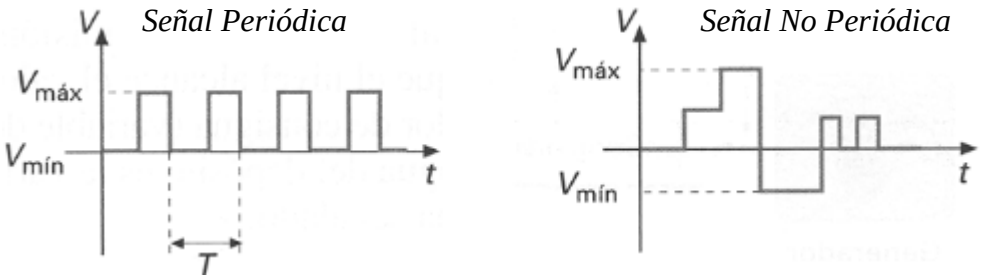


Ilustración 10: Señales Digitales

3.8. Elementos Sensores

Cualquier Sistema de Control tendrá, al menos, una señal de entrada, que procederá de la medida o captación de una variable física, que puede ser, en algunos casos, la propia variable de salida del Sistema, que es realimentada a la entrada del Controlador.

Los elementos encargados de realizar la captura de una variable física se denominan: **elementos sensores** o **transductores**. Un *elemento sensor* está compuesto de estas tres etapas:

- ***Etapa transductor:*** Su misión es convertir las variaciones físicas en magnitudes de señal.
- ***Etapa de acondicionamiento de señal:*** Su misión es regular y adecuar la señal captada por el transductor y adecuarla para la última etapa.
- ***Etapa de salida:*** Su misión es adecuar la señal recibida y enviársela a la unidad de control, nodo o actuador.

En función de la señal de salida podemos distinguir 3 tipos de señores:

- ***Analógicos o Continuos:*** proporcionan una señal de salida que varía de forma continua en función del valor de la magnitud medida. En este grupo podemos mencionar los **señores de temperatura, luminosidad y viento**.
- ***Digitales o Discretos:*** proporcionan una señal de salida que puede presentar un número finito de valores concretos, en función de la magnitud medida (0 o 1). Un ejemplo de sensor digital elemental lo constituye el **pulsador** o el **interruptor**, que pueden adoptar las posiciones de Abierto y Cerrado.
- ***Todo o nada:*** Son aquellos cuyas magnitudes en la etapa de salida pueden ser dos únicos estados (0 o 1). Actúan en un relé interno (*contacto libre de potencial*) que se cierra o se abre según la señal enviada por la etapa de salida. En este grupo podemos mencionar los **sensores de detección de presencia¸ detectores de gas y de humos** en una vivienda.

Es importante mencionar que, el encapsulado del elemento sensor es fundamental para lograr un buen funcionamiento, dado que siempre va a estar sometido a condiciones ambientales que pueden falsear las medidas, introduciendo errores o perturbaciones. Existe una normativa de obligado cumplimiento por parte del fabricante y del instalador, con el fin de garantizar el buen funcionamiento del sensor.

Intentar nombrar todos los **tipos de sensores** existentes sería elaborar una lista muy implica, por lo que solo se van a nombrar los más empleados:

- **Sensores de interrupción:** detectan si pasa corriente o no. Se utilizan como sensores de choque o para contar el número de veces que sucede un hecho, como puede ser contar el número de veces que el robot choca contra un objeto.
- **Sensores de posición:** este sensor permite saber la posición de los objetos del robot, como puede ser a través de un potenciómetro, que al variar su valor de resistencia se puede determinar en qué posición se encuentra.
- **Sensores efecto hall:** estos sensores se utilizan para la detección de metales. Se basan en la variación de la conductividad al acercarse a un cuerpo metálico.
- **Sensores de luz o brillo:** son sensores que detectan la luz. Algunos sensores pueden detectar colores y si son brillantes o no. Una aplicación de estos sensores seria detectar si hay luz en el exterior de una vivienda para subir o bajar las persianas.
- **Sensores de temperatura:** son dispositivos que transforman los cambios de temperatura en cambios en señales eléctricas que son procesados por equipo eléctrico o electrónico. Hay tres tipos de sensores de temperatura, *los termistores, los RTD y los termopares.*
- **Sensores infrarrojos:** estos sensores envían una señal luminosa infrarroja, y según sea el sensor se puede utilizar para medir distancias o detectar intrusiones.
- **Utilización de captación de video como sensor:** últimamente se tiende mucho a incorporar cámaras a los robots, ya que es una de las opciones que más información capta. Este proyecto lleva incorporado una cámara como ya se ha dicho en capítulos anteriores.

3.9. Actuadores

En todo Proceso de Control siempre existe un elemento sobre el que es necesario actuar para modificar el estado del Sistema. Este elemento puede ser: lámpara, radiador, equipo de aire acondicionado, electroválvula, sirena, motor etc...

Por medio del dispositivo actuador, la Unidad de Control puede llevar a cabo las acciones físicas para las que se haya diseñado el sistema. Los **actuadores** son transductores que transforman una señal, generalmente eléctrica, en calor, luz o movimiento y pueden mantener niveles de salida continuos o discretos.

Los tipos de Actuadores que pueden encontrarse en una instalación automatizada pueden clasificarse en:

- ***Actuadores Eléctricos:*** Generalmente son resistencias por las que, al pasar una corriente eléctrica, se produce una generación de calor que eleva la temperatura del medio en el que se encuentran.
- A***ctuadores Electromecánicos:*** Transforman energía eléctrica en mecánica. Algunos ejemples serían: las *electroválvulas,* los *electroimanes,* los *relés* o los *contactores.*
- ***Actuadores Electrohidráulicos* o *Electroneumáticos:*** Pueden ejercerse fuerzas levadas utilizando fluidos sometidos a alta presión. En los *Sistemas Hidráulicos* el fluido es, generalmente, aceite (líquido), y en los *Sistemas Neumáticos* el fluido es aire (gas).

Dentro de los actuadores se van a explicar con más profundidad los motores, y los distintos tipos de motores que existen.

3.9.1. Motores

Un motor eléctrico es una máquina que convierte la energía eléctrica en energía magnética y después en mecánica. Los motores disponen de un eje, para acoplarle engranajes, ruedas, o cualquier mecanismo para trasmitir movimientos creados por el motor. Para darle desplazamiento a los robots necesitamos motores.

La elección de los motores es una tarea complicada, antes de nada, lo que hay que ver, es el empleo que vamos a darle al motor, es decir, para que se vaya a utilizar dicho motor. Si se tiene en cuenta todas las características que definen al motor, como puede ser tamaño, peso, velocidad, rendimiento, par, tensión, etc. Son muchos parámetros a tener en cuenta y es una labor complicada. A continuación se verán los diferentes tipos de motores que se pueden utilizar, para así poder seleccionar el motor que mejor se adapte a lo que se busca.

➢ *Motores de corriente continua (CC):*

Como todo convertidor electromecánico de energía, el **motor de cc** *(Ilustración 11)* es reversible. Si se conecta convenientemente a la red de tensión continua, gracias a la acción de campos magnéticos opuestos hacen girar al *rotor* (eje interno) en dirección opuesta al *estator* (bobina). Para invertir el giro de estos motores, basta con invertir la polaridad de la corriente eléctrica. La mayoría de vece lleva incorporado un juego de engranajes que hacen reducir o aumentar la velocidad, o dar más par o menos, ya que tienen un número de revoluciones muy alto.

Ilustración 11: Motor de Corriente Continua

El motor de Corriente Continua es particularmente adecuado para *tracción*, como por ejemplo en el empleo para *trenes y ascensores*. Debido a su ***alto torque de arranque,*** es decir, la fuerza de giro que posee el motor al arrancar. Dado que el torque es muy elevado puede romper la inercia que posee la masa del tren o del ascensor cuando está detenida y debe iniciar la marcha, repleto de gente.

➢ *Motores de paso a paso (PA):*

La principal característica de este motor es que podemos hacer que se posicione su eje en una determinada posición de giro; además, es posible tener un control muy preciso de su velocidad de giro.

Ilustración 12: Motor paso a paso

El estator de estos motores está constituido por varios electroimanes y el rotor por uno o varios imanes permanentes (*Ilustración 12)*. A las diferentes bobinas del estator se alimenta mediante impulsos, proporcionados normalmente por un circuito electrónico, y se consigue que el rotor se posicione *paso a paso* según sea el avance de dichos impulsos.

La velocidad de giro del rotor depende de la frecuencia de los impulsos y del número de polos. Se fabrican motores con diferentes ángulos de paso, como, por ejemplo, 24 pasos por revolución, 28, 96, 200, etc. De tal forma que si un motor paso a paso posee 24 pasos, su ángulo de paso será igual a: *360º/24 = 15º*

Entre otras, las aplicaciones de estos motores son: impresoras, plotters, cintas magnéticas, equipos médicos, contadores, lectores de tarjetas magnéticas, etc.

- ***Servomotores:***

El **servomotor** (I*lustración 13*) lleva en su interior un pequeño motor con un reductor de velocidad y un multiplicador de fuerza. Es un dispositivo pequeño pero muy potente. El recorrido del eje de salida es de 180º en la mayoría de ellos, aunque los hay que giran 360º que actuaría como un motor común. Su principal ventaja es que con ellos se consiguen movimientos muy precisos gracias a la regulación y control electrónico que se ejerce sobre ellos. El inconveniente es que son lentos.

Ilustración 13: Servomotores

Estos tipos de motores son muy utilizados en las máquinas herramientas modernas, para mover brazos robóticos o controlar la dirección.

3.10. Clasificación de los sistemas domóticos

En la actualidad existe una gran variedad de sistemas domóticos para una vivienda, con características y protocolos de comunicación diferentes. A la hora de decantarse entre un sistema u otro, nos cuestionaremos lo siguiente:

- Desde el **punto de vista de la vivienda:** de si la vivienda es nueva o ya ha sido habitada.
- Desde el **punto de vista técnico:** nos hemos de decantar entre:
  - *El protocolo de comunicación.*
  - *Topología de la red.*
  - *Medio de transmisión.*
  - *Sistema de control.* En el sistema de control hemos de escoger entre los sistemas de control ya mencionados anteriormente, el centralizado, descentralizado o el distribuido.

Podemos realizar la siguiente clasificación:

- *Sistemas por corrientes portadoras.*
- *Sistemas mediante controladores programables.*
- *Sistemas mediante BUS.*

En la *tabla 1* se observan alguno de los factores que diferencian a los tres tipos.

Sistema Técnico	*Medio de Transmisión*	*Tipo de Control*	*Aplicación*	*Capacidad*
Corrientes Portadoras	La propia Red Eléctrica	Descentralizado	Viviendas	Baja
Controlador Programable	Línea de Datos específicas	Centralizado	Viviendas y Pequeños Edificios	Media
BUS	Bus de Datos	Centralizado o Descentralizado	Viviendas y Edificios	Alta

Tabla 1: Clasificación de Sistemas Domóticos

La centralización del control hace referencia a la ubicación de los elementos que ejercen el control real de la instalación.

La capacidad sirve para indicar el volumen máximo de señales o puntos de control que permite el Sistema.

3.10.1. Sistemas por Corrientes Portadoras

Son los más sencillos y económicos. Su implantación se puede realizar sobre instalaciones convencionales no domotizadas, ya que el medio físico de transmisión de la información, (órdenes, datos etc...), se lleva a cabo a través de la *propia Red de Potencia de la instalación.*

Son de uso casi *exclusivo para viviendas unifamiliares*, pues su sistema de codificación solo alcanza, en principio, a 256 elementos emisores o receptores.

El principio de funcionamiento de los Sistemas por Corrientes Portadoras está basado en la utilización de dos tipos de elementos: emisores y receptores. Estos elementos, a pesar de poder integrarse en una misma pieza, tienen la misión de enviar órdenes de actuación codificadas, modulando una portadora de alta frecuencia, (decenas de kilociclos), lanzada a la misma Red.

Las señales del Sistema conviven en la misma Red y basta con el empleo de una serie de filtros para recoger en los distintos receptores la información codificada y, con ella, actuar sobre las cargas que correspondan.

La *capacidad de control* de estos Sistemas es muy limitada y, en todo caso, se hallará en el elemento emisor o en el receptor. Salvo algunos casos de reciente aparición, en casi todos los Sistemas por Corrientes Portadoras existen pocas posibilidades de trabajar con señales analógicas.

Algunos ejemplos de Sistemas por Corrientes Portadoras, (Power Line Carrier Communication):

- **Delta Dore** (Francia).
- **X-10** (Home Systems, EEUU).
- **CEBUS EIA / IS- 60.**

➢ ***Sistema X-10***

El sistema X-10 consta de una gama de transmisores que permiten realizar tareas en diversos campos (seguridad, control de iluminación, automatización del hogar o controladores de uso general), y una gama de receptores que reaccionan a los comandos enviados por los transmisores.

La filosofía fundamental de diseño de X-10 es que los productos puedan interoperar entre ellos y sean compatibles con los productos anteriores de la misma gama, es decir, equipos que habiendo sido instalados hace 20 años sigan funcionando con la gama actual.

El sistema X-10 ha sido desarrollado para ser flexible. Se puede empezar con un producto en particular, por ejemplo un mando a distancia, y expandir luego el sistema para incluir la seguridad o el control con el ordenador, siempre que se desee, con componentes fáciles de instalar y que no requieran cableados adicionales.

Los fundadores de X-10 establecieron ciertos principios estratégicos que permanecen a pesar del paso de los años:

- Diseñar productos que incluyan circuitos integrados propios cumpliendo objetivos de rendimiento.
- Diseñar productos para un amplio sector del mercado, con un bajo coste de manufacturación.
- Introducir los productos a precios competitivos.

La tecnología X-10 de corrientes portadoras fue desarrollada entre 1976 y 1978 por Ingenieros de Pico Electronics Ltd, en Glenrothes, Escocia. Los ingenieros de Pico habían estado diseñando componentes microelectrónicos desde que se introdujeron los circuitos integrados en 1969.

Aunque el mercado principal de X-10 continúa siendo el americano, X-10 distribuye productos en Europa, Asia, África, Latinoamérica y Oceanía.

3.10.2. Sistemas mediante Controladores Programables.

Tienen una *capacidad media* y son aplicables a instalaciones que no sobrepasen un número demasiado grande de señales o de elementos.

El Sistema dispondrá de una **Unidad Central de Proceso y de Cálculo** y de una serie de Unidades Periféricas, concentradas junto a la Unidad Central o separadas de ésta, con una Línea de Datos que las una.

La capacidad de control y el número de funciones en los Controladores Programables son mucho mayores que en los Sistemas por Corrientes Portadoras ya que, en la práctica, se trata de un Autómata Programable de aplicación generalizada.

La posibilidad de manipular señales analógicas y de disponer de distintas Unidades independientes de Entrada y Salida hace que los Sistemas mediante Controladores Programables sean muy versátiles y adaptables a numerosas instalaciones.

El montaje de estos Sistemas resulta algo complejo debido a la necesidad de unir las Unidades de Entrada o Salida de Datos con la Unidad de Control, mediante Líneas de Datos específicas.

Algunos ejemplos de Sistemas mediante Controladores Programables:

- Logo!.

  - ***Autómata programable LOGO!***

Logo! es un autómata programable utilizado en instalaciones domóticas de viviendas y en aplicaciones industriales

El Logo! lleva incorporado una pantalla LCD, que facilitara la interacción entre el usuario y el autómata programable, mediante unos botones que lleva incorporado. También existe la posibilidad de la comunicación entre el autómata y el PC.

El Logo! consta de 24 entradas digitales, 8 entradas analógicas, 16 salidas digitales y 2 analógicas.

3.10.3. Sistemas mediante BUS.

La utilización de un BUS único para soportar el tráfico de información entre los diferentes elementos emisores y receptores de una instalaciónes la mejor solución, tecnológicamente hablando, ya que elimina prácticamente todas las barreras de los Sistemas anteriores, con el fin de poder manejar grandes volúmenes de datos y velocidades de transmisión muy elevadas.

En los Sistemas mediante BUS, generalmente, el control está descentralizado, es decir no existe ningún órgano en el Sistema del que dependa el control de los demás.

Podría decirse que el Sistema está formado por unidades autónomas e inteligentes, capaces de comunicarse entre ellas a través de una "autopista común", haciendo uso de unos Protocolos de comunicación, de señalización y de identificación de elementos.

Referirse a un Sistema mediante BUS es como hablar de una Red de Comunicaciones de Área Local, (LAN), con la diferencia de que el BUS permite un mayor número de elementos a conectar y de que las velocidades de transmisión son mucho más bajas que en las Redes de Área Local.

Algunos de las principales compañías que usas este tipo de sistema son:

- **Konnex Asociaton o KNX / EIB**
- **LonWORKS**
- **Simon Vit@**

➢ *LonWORKS*

Es una tecnología de control desarrollado por la compañía americana Echelon Corp. Puede utilizar una gran variedad de medios de transmisión. Aire, par trenzado coaxial, fibra o red eléctrica. Necesita la instalación de una serie d nodos a lo largo de la red, estos gestionan los diferentes sensores y actuadores. La configuración de estos nodos se tiene que realizar utilizando la herramienta que ofrece la propia empresa, Lonmaker.

Se trata de una tecnología muy robusta y fiable, especialmente indicada para la automatización industrial, que es el ámbito de donde proviene. Aunque su uso se ha extendido a la domótica.

➢ *Sistema KNX*

Inicialmente, el **Sistema KNX**, hace 20 años se llamaba *Instabus*, hasta que se formó *KNX Association* en 1999.

Es la iniciativa de tres asociaciones europeas:

1. *EIBA* (European Installation Bus Association).
2. *Batibus Club International.*
3. *EHSA* (European Home Systems Association).

Para introducir y gestionar un estándar global para todos los productos de control de las viviendas y oficinas, mediante el software específico *ETS* (European Tool Software).

Una de las grandes ventajas de este sistema es que no se trata de un sistema propietario, es decir, no existe una marca comercial detrás de EIB. Son los fabricantes agrupados en la asociación EIBA, quienes desarrollan productos para EIB.

Además, EIB presenta las ventajas inherentes a este tipo de sistemas frente a las instalaciones tradicionales:

- Reducción del cableado y por lo tanto de las posibilidades de incendios.
- Reducción de los costes asociados a la instalación.
- Integración de diferentes funciones en un solo sistema.
- Flexibilidad para ampliaciones y modificaciones futuras. Es posible reprogramar el funcionamiento de la instalación conectando un ordenador al sistema o incluso a distancia mediante un enlace telefónico o a través de Internet.

3.11. Arduino

El principal problema de que no se termine de fomentar la domótica en las viviendas, es porque todos los sistemas domóticos mencionados en el apartado anterior, tienen un precio muy alto de instalación, con lo que muy pocos son los capaces de implantarlo en sus viviendas. Una alternativa más barata y casera es el empleo de las ***placas Arduino.***

El **Arduino** es una plataforma de hardware libre creada en 2005, basada en una placa con un microcontrolador y un entorno de desarrollo, diseñada para facilitar el uso de la electrónica en distintos proyectos, como en este caso para el control de las luces y persianas de una vivienda.

Arduino capta información del entorno a través de sus pines de entrada, mediante una gama amplia de sensores disponibles, pudiendo controlar el entorno que nos rodea, como puede ser motores, servos, luces, etc. Para trabajar con Arduino se necesita tener conocimientos de electrónica y programación.

El microcontrolador situado en la placa se controla con el lenguaje de programación Arduino (basado en *Wiring*) y el entorno de desarrollo Arduino (basado en *Processing*). Los proyectos hechos con Arduino se pueden ejecutar sin necesidad de conectarlos a un PC.

Las placas pueden ser hechas a mano o compradas montadas de fábrica. La principal característica que diferencia las distintas placas, es el **hardware**.

El *hardware* consiste en una placa con un microcontrolador Atmel AVR y puertos de entrada/salida. Los microcontroladores más utilizados son Atmega168, Atmega328, Atmega1280, Atmega8 por su bajo coste y sencillez.

3.11.1. Placas Arduino

Actualmente existen una gran variedad de modelos en el mercado, por lo que vamos a explicar las características principales de cada uno.

- ***Arduino NANO***

Es una placa compacta, se conecta al ordenador con un cable Mini-B USB y además no posee conector para la alimentación externa. Basado en el *ATmega328* (Arduino Nano 3.0) o *ATmega168* (Arduino Nano 2.x) que se usa conectándola a una protoboard.

Las características de la placa son:

- **Pines E/S Digitales:** 14 (de los cuales 6 se pueden utilizar de salida PWM)
- **Entradas Analógicas:** 8
- **Voltaje de funcionamiento:** 5V
- **Corriente máx., PIN de E/S:** 40 mA
- **Memoria Flash:** 16 KB (ATmega168) o 32 KB (ATmega328)
- **SRAM:** 1 KB (ATmega168) o 2 KB (ATmega328)
- **EEPROM:** 512 bytes (ATmega168) o 1 KB (ATmega328)
- **Frecuencia de reloj:** 16 MHz
- **Dimensiones:** 18,5mm x 43.2mm

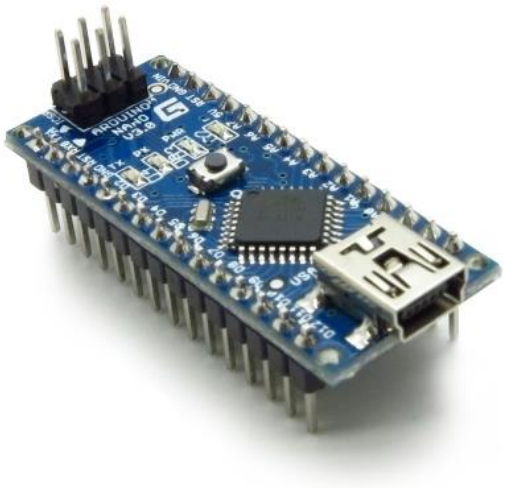


Ilustración 14: Placa Arduino NANO

- ***Arduino UNO***

Es la placa estándar y posiblemente la más conocida y documentada. Contiene una conexión USB HID, un conector de alimentación y conector ICSP. Viene con un *Atmega328* con 32Kbytes de ROM para el programa y la placa funciona con una alimentación de 5V.

Las características de la placa son:

- **Pines E/S Digitales:** 14 (de los cuales 6 se pueden utilizar de salida PWM)
- **Entradas Analógicas:** 6
- **Voltaje de funcionamiento:** 5V
- **Corriente máx, PIN de E/S:** 40 mA
- **Memoria Flash:** 32 KB
- **SRAM:** 2 KB
- **EEPROM:** 1 KB
- **Frecuencia de reloj:** 16 MHz
- **Dimensiones:** 50.5mm x 68mm

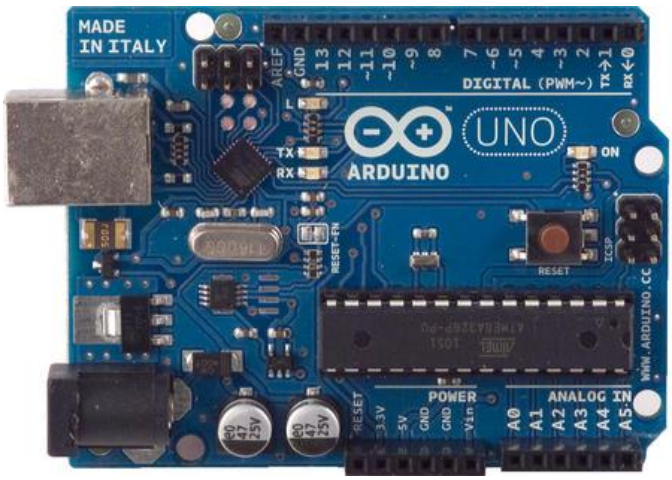


Ilustración 15: Placa Arduino UNO

- ***Arduino LEONARDO***

Esta tarjeta tiene una forma muy parecida a la *placa de Arduino UNO.* La gran ventaja de esta tarjeta es que está basada en el procesador *ATmega32u4*, el cual tiene comunicación USB integrada, por lo que no requiere de un convertidor Serial-USB ni de un cable FTDI para programarse, además de ser un poco más económico que el Arduino Uno por requerir menos componentes.

Gracias a sus capacidades USB puede emular las funciones de un teclado o ratón, permitiéndote dar clic, doble clic, scroll o escribir texto de una manera muy sencilla.

Las características de la placa son:

- **Pines E/S Digitales:** 20 (de los cuales 7 se pueden utilizar de salida PWM)
- **Entradas Analógicas:** 12
- **Voltaje de funcionamiento:** 5V
- **Corriente máx., PIN de E/S:** 40 mA
- **Memoria Flash:** 32 KB
- **SRAM:** 2.5 KB
- **EEPROM:** 1 KB
- **Frecuencia de reloj:** 16 MHz
- **Dimensiones:** 53mm x 75mm

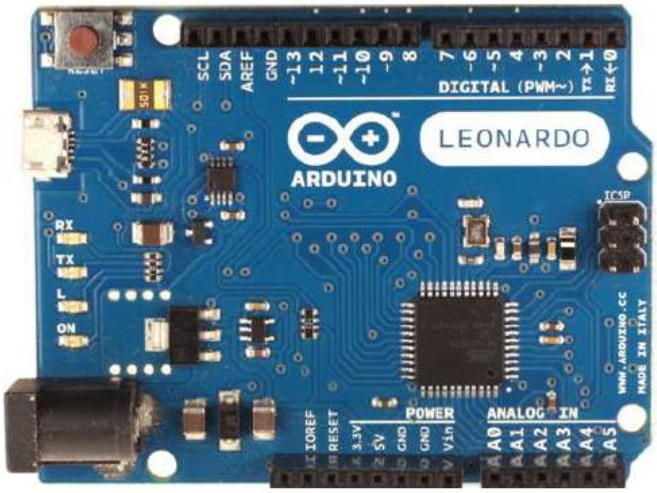


Ilustración 16: Placa Arduino LEONARDO

➢ ***Arduino MICRO***

Esta tarjeta es la versión en miniatura del Arduino Leonardo, por lo que cuenta con sus mismas capacidades, tales como comunicación USB nativa, emulación de Ratón/Teclado y basado en ATmega32u4, además de permitirte utilizarlo junto a un protoboard y reducir en gran medida el tamaño de tu circuito por su diminuto tamaño.

Las características de la placa son:

- **Pines E/S Digitales:** 20 (de los cuales 7 se pueden utilizar de salida PWM)
- **Entradas Analógicas:** 12
- **Voltaje de funcionamiento:** 5V
- **Corriente máx., PIN de E/S:** 40 mA
- **Memoria Flash:** 32 KB
- **SRAM:** 2,5 KB
- **EEPROM:** 1 KB
- **Frecuencia de reloj:** 16 MHz
- **Dimensiones:** 18mm x 48mm

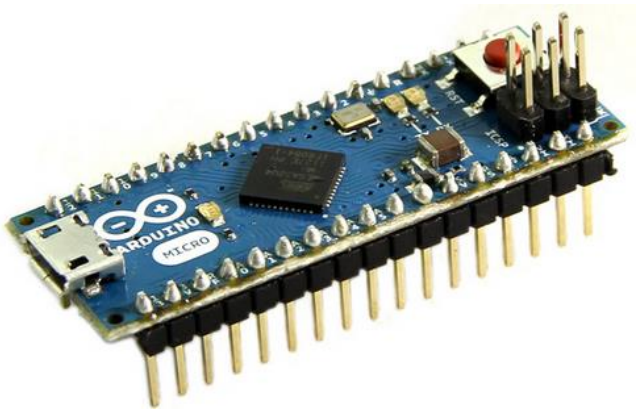


Ilustración 17: Placa Arduino MICRO

➢ ***Arduino YÚN***

Está basado en ATmega32u4 y es la combinación de un Arduino Leonardo, con un chip Wifi que utiliza Linino (un MIPS GNU/Linux basada en OpenWRT). Incorpora Linux en la PCB de la placa Arduino Leonardo y están conectados los dos para ejecutar comandos en Linux, y se utiliza como una interfaz Ethernet y Wifi.

Las características de la placa son:

- **Pines E/S Digitales:** 20 (de los cuales 7 se pueden utilizar de salida PWM)
- **Entradas Analógicas:** 12
- **Voltaje de funcionamiento:** 5V
- **Corriente máx, PIN de E/S:** 40 mA
- **Memoria Flash:** 32 KB
- **SRAM:** 2.5 KB
- **EEPROM:** 1 KB
- **Frecuencia de reloj:** 16 MHz
- **Dimensiones:** 53mm x 73mm

Ilustración 18: Placa Arduino YÚN

- ***Arduino MEGA***

Es con mucha diferencia el más potente y el que más pines i/o tiene, apto para trabajos ya algo más complejos aunque tengamos que sacrificar un poco el espacio, cuenta con el microcontrolador Atmega2560 con más memoria para el programa, más RAM y más pines que el resto de los modelos. Contiene todo lo necesario para hacer funcionar el microcontrolador; simplemente conectándolo al ordenador, con el cable USB o aliméntalo con un trasformador o batería para empezar.

Las características de la placa son:

- **Pines E/S Digitales:** 54 (de los cuales 14 se pueden utilizar de salida PWM)
- **Entradas Analógicas:** 16
- **Puertos serie por Hardware (UARTS):** 4
- **Voltaje de funcionamiento:** 5V
- **Corriente máx., PIN de E/S:** 40 mA
- **Memoria Flash:** 256 KB
- **SRAM:** 8 KB
- **EEPROM:** 4 KB
- **Frecuencia de reloj:** 16 MHz
- **Dimensiones:** 54mm x 102mm

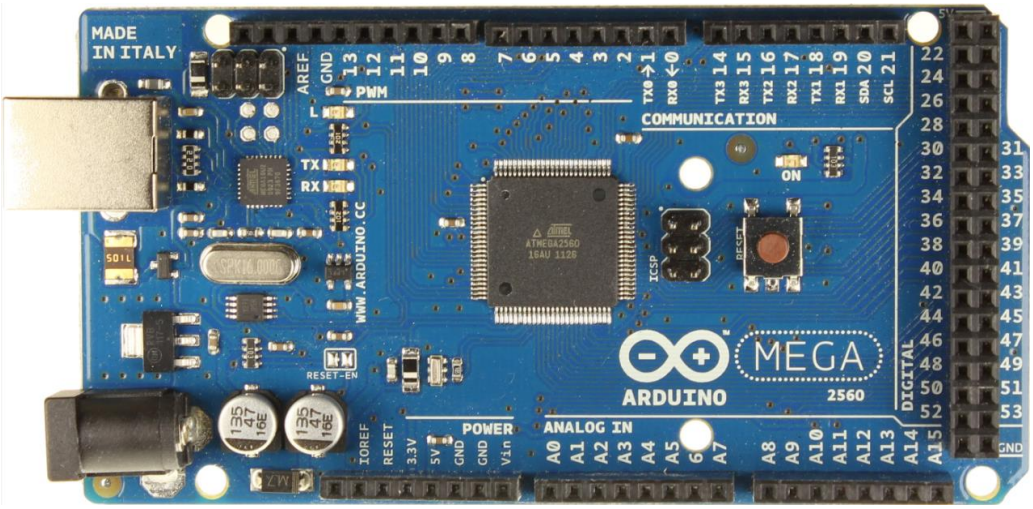


Ilustración 19: Placa Arduino MEGA

- ***Arduino MEGA ADK***

Esta tarjeta es compatible pin con pin con el Arduino Mega, con la gran ventaja de incluir una interfaz Host USB, que te permite conectar tu Arduino a un teléfono con sistema operativo de Android y comunicarte con él para acceder a la información de sus sensores o recibir órdenes de tu celular para controlar motores, LEDs, etc. También está basado en el microcontrolador ATmega2560.

Con ésta tarjeta puedes incluso recargar tu teléfono celular si requiere menos de 750mA y tienes una fuente de suficiente capacidad de corriente.

Las características de la placa son:

- **Pines E/S Digitales:** 54 (de los cuales 14 se pueden utilizar de salida PWM)
- **Entradas Analógicas:** 16
- **Puertos serie por Hardware (UARTS):** 4
- **Voltaje de funcionamiento:** 5V
- **Corriente máx, PIN de E/S:** 40 mA
- **Memoria Flash:** 256 KB
- **SRAM:** 8 KB
- **EEPROM:** 4 KB
- **Frecuencia de reloj:** 16 MHz
- **Dimensiones:** 54mm x 102mm

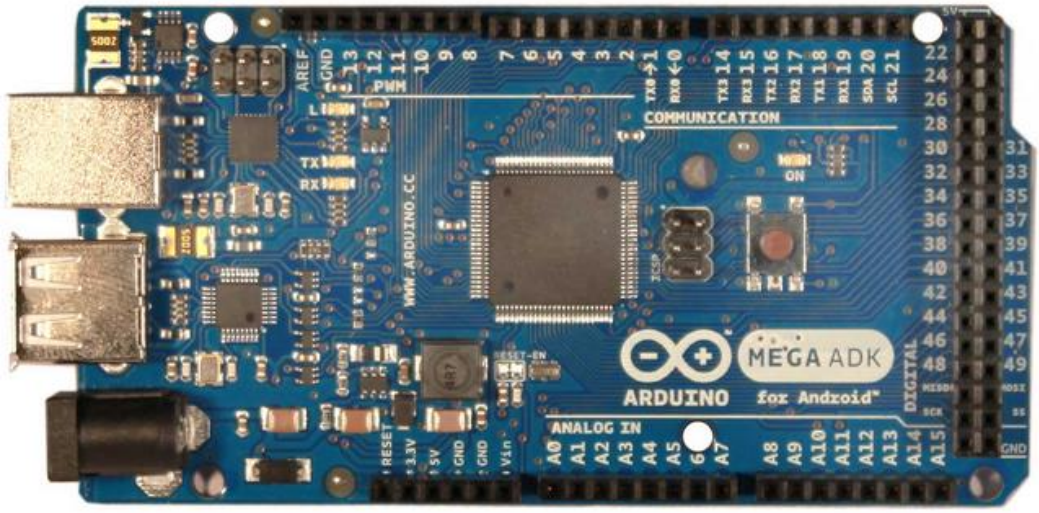


Ilustración 20:Placa Arduino MEGA ADK

- ***Arduino DUE***

Esta tarjeta está basada en un procesador ARM-Cortex de 32-bits a 84MHz, es la tarjeta Arduino con mayor capacidad y velocidad de procesamiento de ésta lista.

Tiene un footprint similar al del Arduino Mega, pero tiene periféricos adicionales, como 2 convertidores DAC (convertidores de analógico-digital), 2 conectores USB un conector de alimentación, un conector ICSP, un conector JTAG.

En uno de los USB se puede conectar otros dispositivos USB a la tarjeta (ratones, teclados, etc...).

Las características de la placa son:

- **Pines E/S Digitales:** 54 (de los cuales 12 se pueden utilizar de salida PWM)
- **Entradas Analógicas:** 12
- **Puertos serie por Hardware (UARTS):** 4
- **Voltaje de funcionamiento:** 3.3V
- **Corriente máx., PIN de E/S:** 130 mA
- **Memoria Flash:** 512 KB
- **SRAM:** 96 KB (dos bancos: 64kB y 32kB)
- **Frecuencia de reloj:** 84 MHz
- **Dimensiones:** 53mm x 104mm

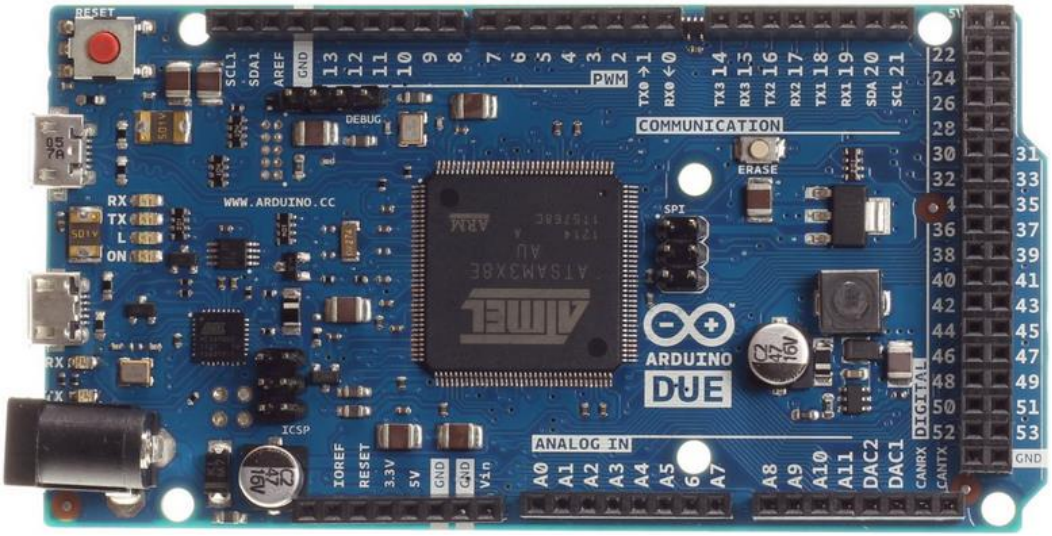


Ilustración 21: Placa Arduino Due

- ***Arduino ETHERNET***

Ésta tarjeta está basada en el microcontrolador ATmega328, al igual que el Arduino Uno, pero cuenta además con capacidad de conectarse a una red vía su puerto Ethernet. Cuenta además con un slot para tarjetas SD por lo que podrás guardar una gran cantidad de información y utilizarla cuando lo requieras, aún después de haber reseteado la tarjeta.

Se requieren de los pines 10 – 13 para la comunicación Ethernet, por lo que no deberán usarse para otros propósitos.

Las características de la placa son:

- **Pines E/S Digitales:** 14 (de los cuales 4 se pueden utilizar de salida PWM)
- **Entradas Analógicas:** 6
- **Voltaje de funcionamiento:** 5V
- **Corriente máx., PIN de E/S:** 40 mA
- **Memoria Flash:** 32 KB
- **SRAM:** 2 KB
- **EEPROM:** 1 KB
- **Frecuencia de reloj:** 16 MHz
- **Dimensiones:** 53mm x 68mm

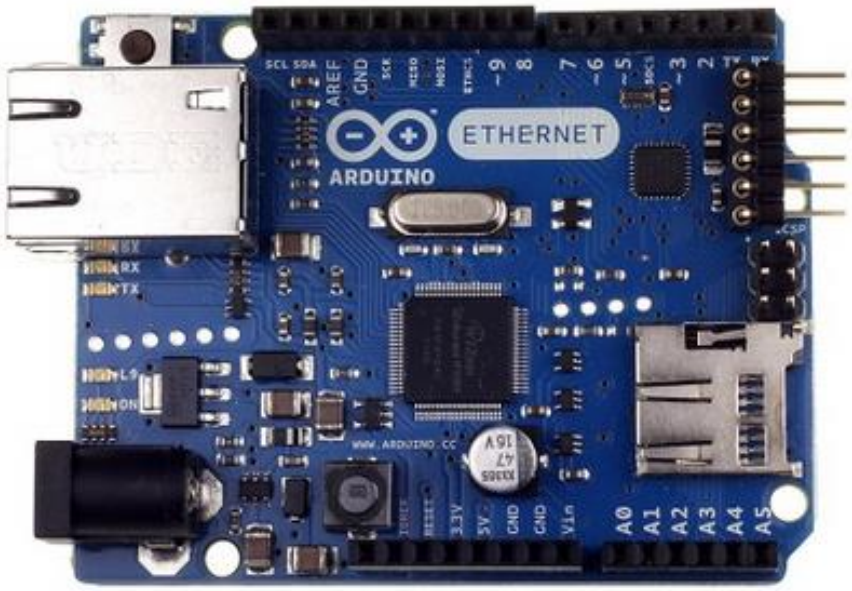


Ilustración 22: Placa Arduino ETHERNET

3.11.2. Shield Arduino.

Los ***"Shields" o escudos***, son placas que pueden ser conectadas directamente sobre las Placas de Arduino, para así mejorar o aumentar sus capacidades. Las diferentes *"shields"* siguen la misma filosofía que el conjunto original: son fáciles de montar, y baratas de producir.

En la siguiente tabla (*Tabla 2)* podremos observar las *"shields"* más importantes, con sus principales funcionalidades.

Nombre	*Compatibilidad*	*Funcionalidades*
Arduino GSM	Uno, mega y ADK	Conectar a internet, puede enviar SMS
Arduino Ethernet	Cualquier Arduino	Contar a una red Wifi
Arduino Wifi	Cualquier Arduino	Controlador de puente completo

Tabla 2: Principales Shields Arduino

3.11.3. Sensores de Arduino

Los sensores manejados con las placas Arduino, ya vienen preparados para ser conectados directamente a nuestra placa. Existe una enorme variedad de sensores preparados para Arduino, como por ejemplo, sensores de luminosidad, temperatura, de distancia, de presencia, humedad, etc.

En este apartado solo explicaremos los sensores que se van a utilizar en este proyecto, y son: *sensor de luminosidad (LDR), detector de presencia (PIR) y sensor de temperatura.*

- ***Sensor de Luminosidad (LDR)***

Los LDR (Light Dependent Resistors) como su nombre indica son resistencias dependientes de la luz, también son conocidas como *Fotorresistencias,* Son sensores resistivos basados en semiconductores empleados para la medida y detección de radiación electromagnética.

La LDR típica consiste en una fina capa semiconductora dispuesta sobre un sustrato cerámico o de plástico. La forma de la película sensitiva tiene por objeto maximizar la superficie de exposición y al mismo tiempo mantener un espacio reducido entre los electrodos para aumentar la sensibilidad.

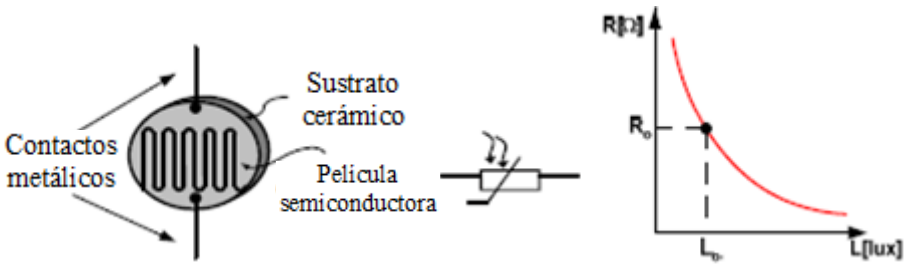


Ilustración 23: LDR, símbolo y gráfica de relación resistencia - iluminación

La tensión máxima que pueden soportar puede llegar hasta 600V y modelos capaces de disipar hasta 1W.

- ***Sensor de Presencia (PIR)***

Los *detectores PIR (Passive Infrared) o Pasivo Infrarrojo*, reaccionan sólo ante determinadas fuentes de energía tales como el calor del cuerpo humano o animales. Básicamente reciben la variación de las radiaciones infrarrojas del medio ambiente que cubre. Es llamado pasivo debido a que no emite radiaciones, sino que las recibe.

Su componente principal es el **sensor piroeléctrico**. Se trata de un componente electrónico diseñado para detectar cambios en la radiación infrarroja recibida. Generalmente dentro de su encapsulado incorporan un transistor de efecto de campo que amplifica la señal eléctrica que genera cuando se produce dicha variación de radiación recibida

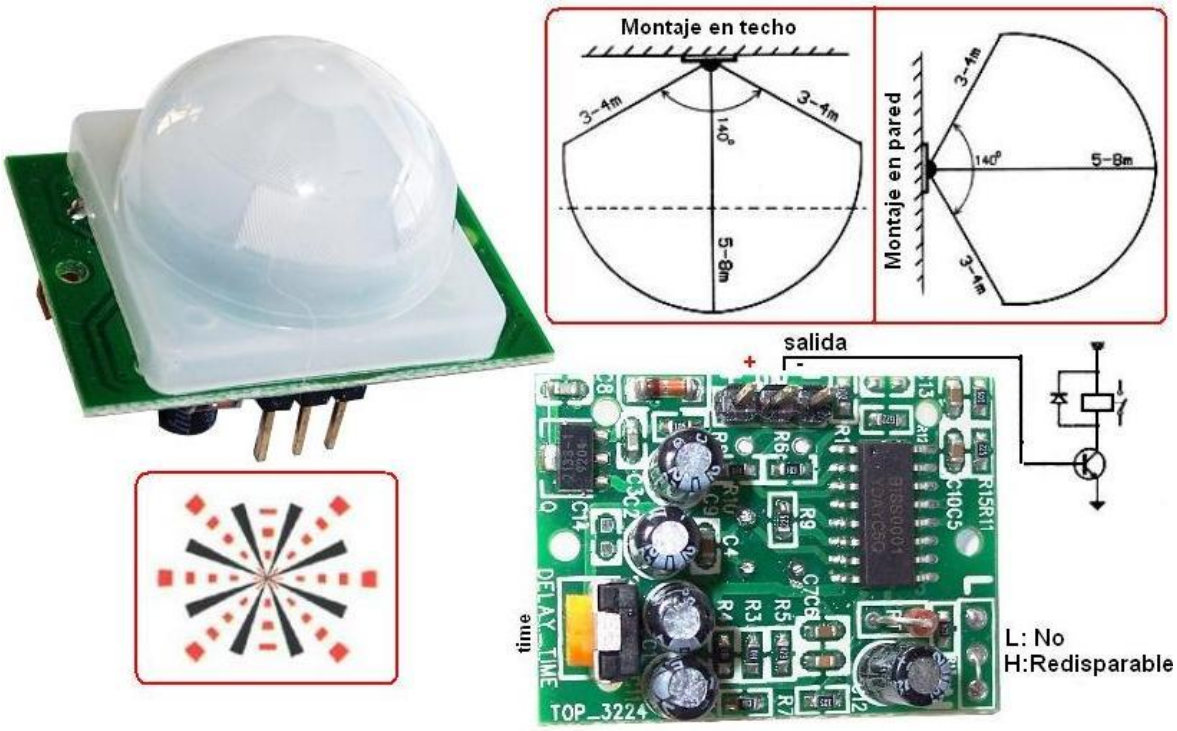


Ilustración 24: Aspecto del Sensor PIR y rango de trabajo

- ***Sensor de temperatura***

Los *sensores de temperatura* son dispositivos que transforman los cambios de temperatura en señales eléctricas que serán procesados por equipos eléctricos o electrónicos, como por ejemplo, microcontroladores.

Existen tres tipos de sensores de temperatura: **termistores, RTD y termopares.** Todos ellos estas constituidos por un elemento sensor, envuelto por una vaina rellena de un material muy conductor de la temperatura, para que los cambios de temperatura se trasmitan lo más rápido posible del elemento sensor al equipo controlador de dichos sensores.

Los **termistores** (*Ilustración 25*) están basados en una resistencia de semiconductores que varían en función de la temperatura. Existen dos tipos de termistores, *NTC y PTC.* En los *NTC* se da que al aumentar la temperatura, disminuirá la resistencia, y los *PTC,* ocurrirá justamente lo contrario, al aumentar la resistencia, aumentará la resistencia.

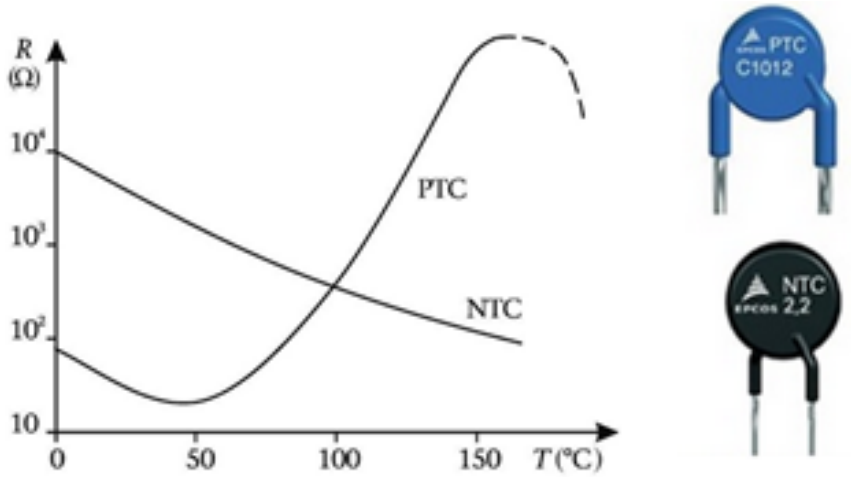


Ilustración 25: Curva característica y aspecto de los sensores PTC y NTC

El **RTD** *(Ilustración 26)***,** es un sensor de temperatura resistivo metálico de coeficiente térmico positivo muy utilizado en la práctica para la medición de la temperatura de medios y superficies, gracias a su gran variedad de formas constructivas. Los metales empleados para un RTD son: cobre, níquel, platino y wolframio.

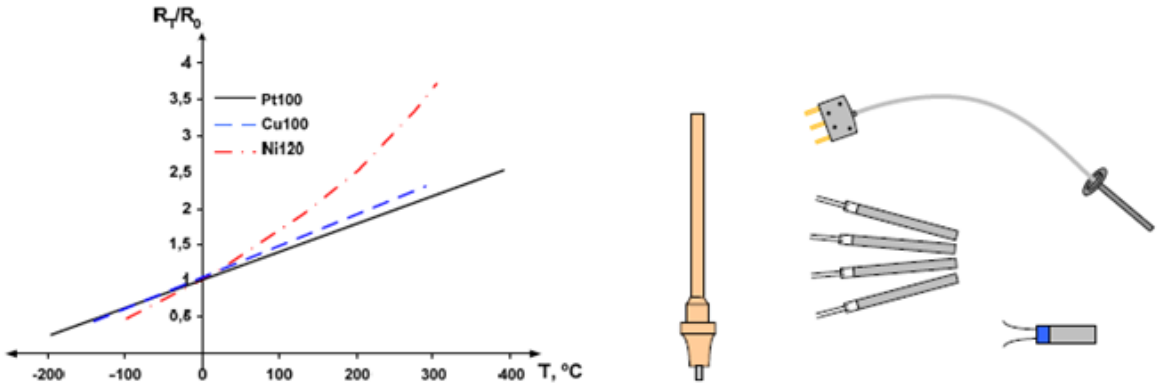


Ilustración 26: Curva característica y aspecto del sensor RTD

Y por último, el **termopar** (*Ilustración 27*)**,** es un sensor de temperatura constituido por dos metales diferentes cuya característica principal es que produce una tensión proporcional a la diferencia de temperaturas entre los puntos de unión entre ambos metales. La principal ventaja de los termopares es que tienen un amplio rango de medida y son muy económicos; pero por el contrario su precisión es menor si se compara con los termistores o RTD.

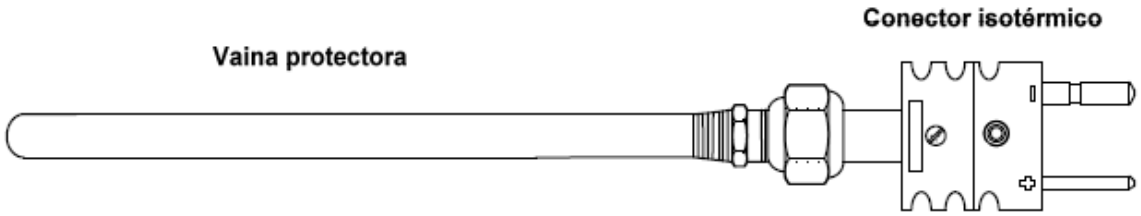


Ilustración 27: Aspecto de un Termopar con vaina protectora

Existe una gran variedad de termopares que irán en función de los materiales empleados, dando lugar a los distintos tipos: **J, K, N, T, R, S y B**. Los 4 primeros son conocidos como Termopares de Metales Base (cobre, níquel, aluminio, etc...), mientras que los 3 últimos son conocidos por Termopares de Metales Nobles (platino y rodio). En la *Tabla 3***,** se puede observar las aplicaciones de los distintos tipos de termopares.

TIPO	APLICACIONES
J	Apropiado para atmósferas inertes o reductoras. Las atmósferas oxidantes disminuyen la vida útil, se oxidan muy rápidamente por encima de 538ºC. No es adecuado para temperaturas por debajo de 0ºC
K	Apropiado para temperaturas mayores a 538ºC. Tiende a oxidarse ante la presencia de oxígeno.
N	Tiene mejor aguante a la oxidación que el tipo **K**
T	Adecuado para atmósferas oxidantes, inertes y reductoras
R, S	Recomendado para altas temperaturas. El tipo **R** se utiliza industrialmente y el **S** en laboratorios principalmente
B	Semejante a los tipos **R** y **S,** aunque su límite de temperatura es mayor

Tabla 3: Aplicaciones de los distintos tipos de termopares

3.12. Lógica difusa

Hoy día la ***lógica difusa*** o también conocido como ***lógica borrosa*** es una de las disciplinas que mayor número de personas emplean en sus proyectos, se basa en una extensión de la lógica tradicional y además de permitir los valores totalmente ciertos o falsos también admiten valores de creencia intermedios, es decir, es la lógica aplicada a conceptos que pueden tomar un valor cualquiera de veracidad dentro de un conjunto de valores que oscilan entre dos extremos, *la verdad absoluta* y la *falsedad total.* La lógica difusa permite tratar información imprecisa, como *persiana subida* o *iluminación media*, en términos de conjuntos borrosos que se combinan en reglas para definir acciones: *si la persiana está bajada iluminar mucho*. Así se consigue que los sistemas de control basados en lógica difusa combinen variables de entrada por medio de reglas que producen uno o varios valores de salida.

La lógica difusa fue investigada por primera vez en 1965, por *Lotfi Zadeh,* ingeniero eléctrico iraní y profesor de la Universidad de California, en el artículo de lógica difusa llamado "*Fuzzy Sets*" (Martín del Brio & Sanz Molina, 2006), donde se dan a conocer por primera vez los conceptos de esta técnica, como este: '*Conforme la complejidad de un sistema aumenta, nuestra capacidad para ser precisos y construir instrucciones sobre su comportamiento disminuye hasta el umbral más allá del cual, la precisión y el significado son características excluyentes*'.

Con esto quiso decir que el pensamiento humano se basa en etiquetas lingüísticas y no en números. Es decir, para el ser humano le es más cómodo trabajar con términos lingüísticos, aun siendo más imprecisos que los números.

Para un mejor entendimiento sobre la lógica difusa vamos a emplear el ejemplo que hizo *Zadeh* "*Los hombres altos*". Según la teoría de lógica clásica, al conjunto de hombres altos solo pertenecen los que miden más de una determinada altura, 1.80 metros, por lo que un hombre que mide 1.81 metros sería considerado alto, mientras que un hombre de 1.79 metros sería considerado bajo. Esto no sería muy lógico ya que tan sólo se está hablando de 2 centímetros de diferencia entre ambos, para decidir si se es bajo o alto. Hay casos en los que no es fácil catalogar algo y se introduzca la lógica difusa. Según esta lógica, no tiene un límite claro que indique que pertenece a un grupo u otro. El evaluar si un hombre es alto o bajo, se hace mediante una función que define la transición entre alto a bajo y para ellos asigna a las distintas alturas un valor entre 0 y 1. Según sea este valor se considera que se pertenece al conjunto o no.

Aplicando esto al caso anterior un hombre que mida 1.79 metros se puede decir que pertenece al conjunto de hombres altos con un grado de 0.75 y el hombre que medía 1.81 metros pertenece al conjunto de hombres altos con un grado de 0.8. Si representamos esto en una gráfica se obtendrá que la transición entre alto o bajo con la lógica borrosa es una curva con cambios no abruptos (*Ilustración 27*) mientras que con la lógica clásica, el paso de alto a bajo o viceversa es brusco (*Ilustración 28*):

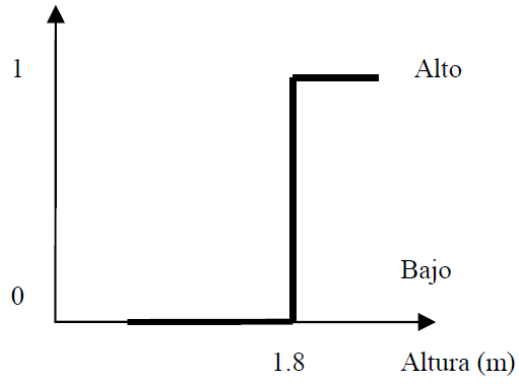


Ilustración 288: Gráfica de pertenencia de L. Clásica

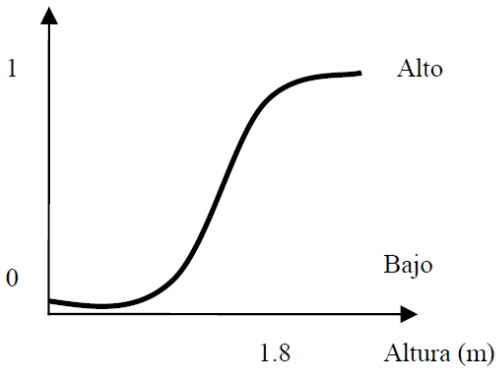


Ilustración 27: Gráfica de pertenencia de L. Difusa

Así pues, los conjuntos difusos pueden ser considerados como una generalización de los conjuntos clásicos, es decir, la teoría clásica sólo contempla la pertenencia o no de un elemento a un conjunto, sin embargo la teoría de conjuntos difusos contempla la

pertenencia parcial de un elemento a un conjunto, es decir, cada elemento presenta un grado de pertenencia a un conjunto difuso que puede tomar cualquier valor entre 0 y 1. Este grado de pertenencia se define mediante la función característica asociada al conjunto difuso, que quiere decir, cada valor pueda tomar un elemento o variable de entrada 'x' la función característica $\mu_A(x)$ proporciona el grado de pertenencia de este valor de 'x' al conjunto difuso 'A'.

Un conjunto clásico A, en un universo$U$, se puede definir de varias formas:

- Especificando las propiedades que deben cumplir los elementos a ese conjunto
- En términos de la función de pertenencia $\mu_A(x)$:

$$\mu_A(x) = \begin{cases} 1 & si \quad x \in A \\ 0 & si \quad x \notin A \end{cases}$$

Podemos además decir que el conjunto A es matemáticamente equivalente a su función de pertenencia o característica $\mu_A(x)$, ya que conocer $\mu_A(x)$ es lo mismo que conocer A.

Un conjunto difuso en el universo U se caracteriza por una función de pertenencia $\mu_A(x)$ que toma valores entre [0,1], y puede representarse como un conjunto de pares ordenados de un elemento x y su valor de pertenencia al conjunto:

$$A = \{(x, \mu_A(x)) \,|\, x \in U\}$$

La **función característica** proporciona una medida del grado de similaridad de un elemento U con el conjunto difuso. La forma de la función característica utilizada, depende del criterio aplicado en la resolución de cada problema y variará en función del usuario. La única condición que debe cumplir una función característica es que tome valores entre 0 y 1, con continuidad. Las funciones características más comúnmente utilizadas por su simplicidad matemática y su manejabilidad son: ***triangular, gaussiana, sigmoidal, gamma, pi, campana, etc…*** (*Ilustración 29*).

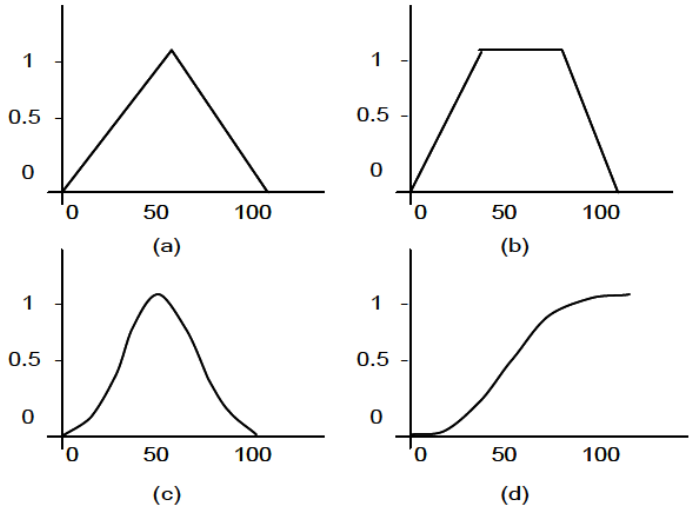


Ilustración 29: Alguna de las funciones características más importantes: (a) triangular, (b) trapezoidal, (c) gaussiana y (d) sigmoidal.

El número de funciones características asociadas a una misma variable es elegido por el creador, a mayor número de funciones características tendremos mayor resolución, pero supondrá una mayor complejidad.

Además estas funciones pueden estar solapadas o no, el hecho de estar solapadas pone de manifiesto un aspecto clave de la lógica difusa: una variable puede pertenecer con diferentes grados a varios conjuntos difusos a la vez.

3.12.1. Aplicaciones generales

Actualmente la lógica difusa tiene un sin número de aplicaciones que afectan a nuestra vida cotidiana de alguna forma, pero no nos damos cuenta de ello. La lógica difusa se ha desarrollado en diferentes áreas y a continuación se mencionan algunas:

- Control inteligente de **electrodomésticos.**
- **Enfoque automático** en cámaras digitales.
- Control en **procesos industriales.**
- **Comercio electrónico.**
- **Sistemas de escritura.**
- Mejora en la **eficiencia del consumo de combustible** en vehículos.
- **Sistemas expertos** que simulan el comportamiento humano.

Se va a explicar la aplicación de la lógica difusa en uno de sus principales ámbitos, y que es utilizado por la mayoría de las personas, como son los *electrodomésticos*, para una

mejor comprensión. La lógica difusa se aplica en los electrodomésticos, en el momento en el que, por ejemplo una lavadora, decide que programa ha de tomar, que cantidad de agua deja entrar, cómo controlar la humedad en el secado de la ropa, etc.

3.12.2. Esquema general de un Sistema Difuso.

Un sistema de control difuso trabaja de manera muy diferente a los sistemas de control convencionales. Estos usan el conocimiento experto para generar una base de conocimientos que dará al sistema la capacidad de tomar decisiones sobre ciertas acciones que se presentan en su funcionamiento. Los sistemas de control difuso permiten describir un conjunto de reglas que utilizaría una persona para controlar un proceso y a partir de estas reglas generar acciones de control. El control difuso puede aplicarse tanto en sistemas muy sencillos como en sistemas cuyos modelos matemáticos sean muy complejos. La estructura de un controlador difuso, es como se muestra en la *Ilustración 30*.

Ilustración 30: Estructura de un módulo difuso

- ***Modelado***

Lo primero que se tiene que hacer es definir las **variables de entrada** y las **variables de salida.** Posteriormente definiremos el tipo de dominio de definicion que se va a utilizar en cada variable.

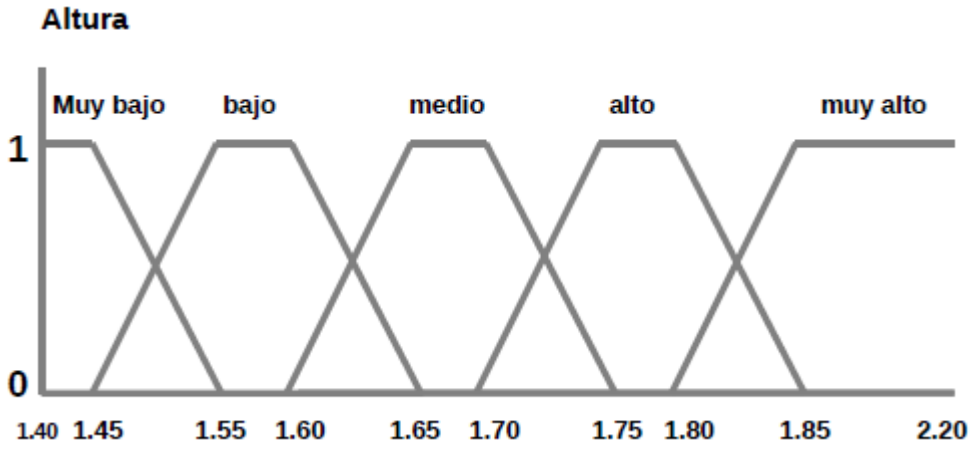


Ilustración 31: Ejemplo de sistema difuso

➢ *Fusificación*

La **fusificación** tiene como objetivo convertir *valores numéricos* en *valores lingüísticos*. En la fusificación se asignan *grados de pertenencia* a cada una de las variables de entrada con relación a los conjuntos difusos previamente definidos utilizando las funciones de pertenencia asociadas a los conjuntos difusos.

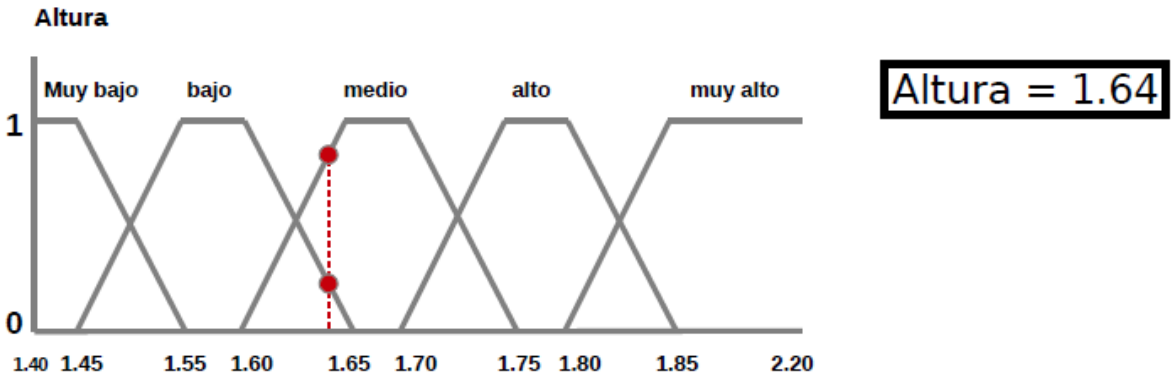


Ilustración 32: Resultado al escoger una altura cualquiera en la gráfica

Para obtener la **función de pertenencia** de uno conjunto trapezoidal:

$$\prod(u;a,b,c,d) = \begin{Bmatrix} 0 & u < a \\ \frac{(u-a)}{(b-a)} & a \le u < b \\ 1 & b \le u \le c \\ \frac{(d-u)}{(d-c)} & c < u \le d \\ 0 & u > d \end{Bmatrix}$$

Si se diera el caso de hacerlo sobre una función de pertenencia de un conjunto triangular:

$$\Delta(u;a,b,c,) = \begin{Bmatrix} 0 & u < a \\ \frac{(u-a)}{(b-a)} & a \le u < b \\ \frac{(c-u)}{(c-b)} & b < u \le c \\ 0 & u > c \end{Bmatrix}$$

➢ *Inferencia*

La **inferencia** relaciona los conjuntos difusos de entrada y salida para representar las reglas que definirán el sistema. En la inferencia se utiliza la información de la base de conocimiento para generar reglas mediante el uso de condiciones, por ejemplo:

*"**Si** antecedentes **entonces** consecuente"*

Existe variedad de tipos de reglas, una de ellas sería, las **reglas difusas de Mamdani.**

La ***Reglas difusas de Mamdani*** está basado en la superposición de las salidas de cada una de las reglas que componen el modelo difuso para una determinada entrada. Como por ejemplo:

Regla 1: *IF* x es A_1 e y es B_1 *THEN* z es C_1

Regla 2: *IF* x es A_2 e y es B_2 *THEN* z es C_2

La relación difusa es interpretada como una intersección de conjuntos difusos. Para ello sigue los siguientes pasos:

1) Obtención del grado de satisfacción de los antecedentes.

Regla 1: $W_1 = mín\left[\mu_{A_1}(x_0), \mu_{B_1}(y_0)\right]$

Regla 2: $W_2 = mín\left[\mu_{A_2}(x_0), \mu_{B_2}(y_0)\right]$

2) Obtención de la conclusión de cada regla.

Conclusión de la Regla 1: $\mu_{C_1}(x_0) = mín\left[W_1, \mu_{C_1}(z)\right] \quad \forall_z \in Z$

Conclusión de la Regla 2: $\mu_{C_2}(x_0) = mín\left[W_2, \mu_{C_2}(z)\right] \quad \forall_z \in Z$

3) Obtener una conclusión mediante la agregación de las conclusiones anteriores.

Conclusión final: $\mu_C(z) = máx\left[\mu_{C_1}(z), \mu_{C_1}(z)\right]$

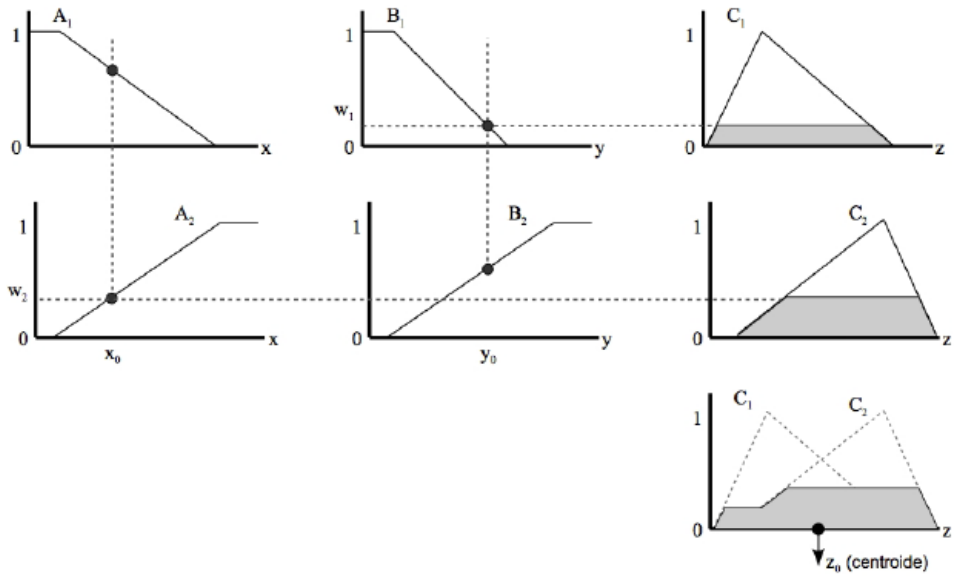


Ilustración 33: Método Mamdani

- ***Defusificación***

El objetivo principal del proceso de **defusificación** es volver a obtener un valor numérico preciso. Para poder realizar dicho proceso, se realiza el *centro de masas* del conjunto difuso:

$$z_0 = \frac{\int u_c(z) \cdot z \, dz}{\int u_c(z) \, dz}$$

En la defusificación se utilizan métodos matemáticas simples como el método del *Centroide, Método del Promedio Ponderado* y *Método de Membresía del Máximo*.

Para el diseño de un controlador basado en lógica difusa debe tener un compromiso entre diversos criterios de diseño: velocidad, precisión y flexibilidad.

Para que el sistema pueda conseguir los resultados deseados debe tener una velocidad adecuada para que no haya mucho tiempo de respuesta. Si deseamos una alta precisión en el control, se necesitará una gran cantidad de conjuntos para cada variable y un alto número de reglas, lo que aumentará el tiempo de respuesta al tener un mayor número de operaciones. Por lo tanto habrá que alcanzar un equilibrio entre diversos criterios.

Para el campo de los sistemas basados en lógica borrosa se han introducido diversos sistemas de desarrollo de propósito específico. Uno de los más utilizados es Matlab y el que se utilizará en este proyecto.

3.12.3. Toolbox Fuzzy de Matlab

Matlab es un software matemático que ofrece un entorno de desarrollo integrado (IDE) con un lenguaje de programación propio (*Lenguaje M*). Este integra computación, visualización y programación, en un entorno fácil de usar donde los problemas y las soluciones son expresados en la más familiar notación matemática.

El ***Fuzzy Logic Toolbox,*** como su nombre indica, desarrolla programas de la lógica difusa, basándose en grados de pertenencia y obtener respuesta de varios problemas.

A continuación se va a explicar, lo fundamental para tener un conocimiento básico de esta herramienta

Lo primero que se ha de hacer es abrir el programa, y en la ventana de comandos, escribir la palabra '*fuzzy'* y dar a '*INTRO' (Ilustración 34).*

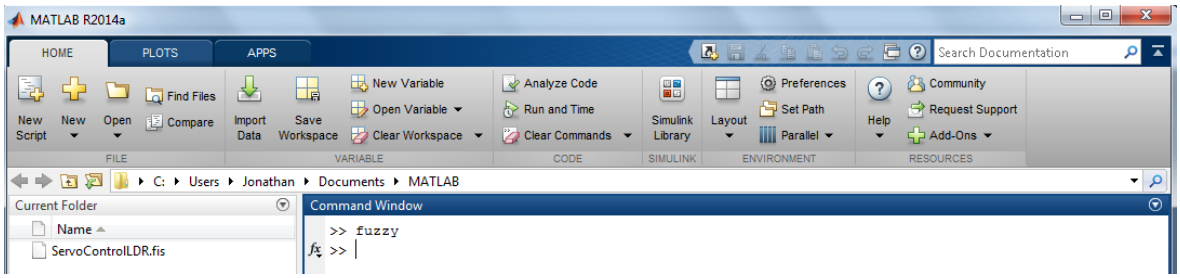


Ilustración 34: Workspace de Matlab con el comando fuzzy.

Y a continuación aparecerá la siguiente ventana. (*Ilustración 35*)

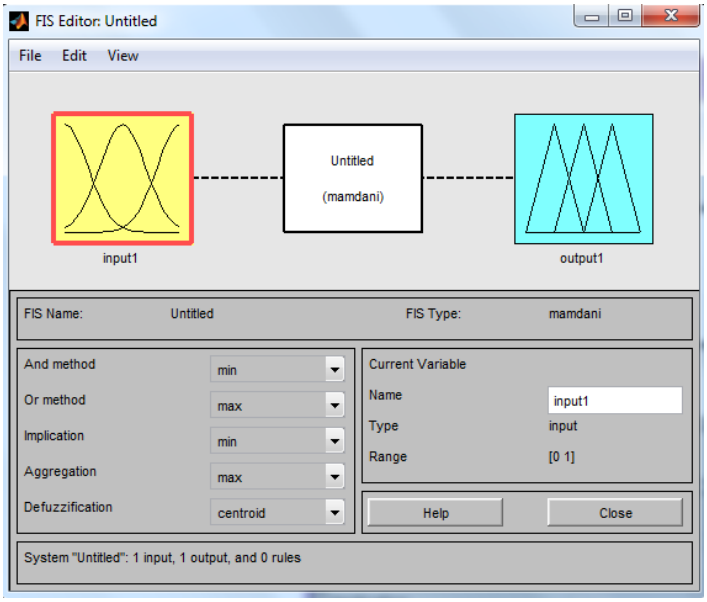


Ilustración 35: Toolbox Fuzzy de Matlab

A través de este editor, se pueden modificar los métodos de los operadores lógicos '*and*' y '*or*', los métodos de implicación, de agregación y de defuzzificación. Permite elegir el modelo a usar Sugeno o Mamdani, a través de las distintas opciones de la barra de menús.

Se pueden agregar variables de entrada y de salida, así como definir sus conjuntos borrosos, eligiendo rango, forma y parámetros.

Las reglas de control del método, se editan a través de un editor específico, en el que se van seleccionando los distintos valores de las variables de entrada y de salida. Para poder implementar el controlador es necesario guardarlo, para poder importar el archivo al workspace, para que luego Matlab lo pueda reconocer y pueda ser usado por cualquier aplicación que lo necesite, o por Simulink. Si se desea trabajar con un modelo ya guardado se debe importar desde el menú fuzzy primero y luego exportarlo al workspace.

Es este proyecto se ha utilizado para la realización de un prediseño del sistema difuso empleado y para poder hacer distintas simulaciones, pudiendo obtener gráficamente el valor de las distintas salidas.

3.13. Inteligencia Artificial o Machine Learning

La *Inteligencia Artificial (IA),* se podría decir que tiene como principal objetivo el estudio del comportamiento de las máquinas. A su vez, el comportamiento inteligente trata de percibir, razonar, aprender, comunicarse y actuar por sí solo, sin necesidad de ser manipulado por el ser humano, para así conseguir una mayor libertad. Una de las metas de la IA es que las máquinas consigan llegar a comprender todos estos comportamientos, igual o mejor que el ser humano.

Hoy en día la IA la podríamos encontrar en las siguientes aplicaciones:

- ***Planificación autónoma.***

El programa de la *NASA Agente Remoto* se convirtió en el primer programa de planificación autónoma a bordo que controlaba la planificación de las operaciones de una nave espacial desde abordo. El Agente Remoto generaba places a partir de objetivos generales especificados desde tierra, y monitorizaba las operaciones de la nave sobre la marcha.

- ***Juegos.***

Deep Blue de *IBM* fue el primer sistema que derrotó a un campeón mundial en una partida de ajedrez, superando a Garry Kasparov en 1997. La revista *Newsweek* describió la partida como "La partida final".

- ***Control autónomo.***

El sistema de visión *ALVIN* fue diseñado para dirigir un coche de forma que fuese siguiendo una línea. Se instaló en una furgoneta controlado por un ordenador en el *NavLab* de UCM y se utilizó para dirigir al vehículo por Estados Unidos. El *NavLab* posee videocámaras que transmiten imágenes de la carretera a ALVIN, que posteriormente calcula la mejor dirección para tomar, basándose en las experiencias almacenadas en los viajes de entrenamiento.

- ***Diagnosis.***

Programas de diagnósticos médicos basados en el análisis de la probabilidad, han llegado a alcanzar niveles semejantes al de médicos expertos en algunas áreas de la medicina.

- ***Planificación logística.***

En 1991 durante la crisis del *Golfo Pérsico*, las fuerzas de Estados Unidos desarrollaron la herramienta *Dynamic Analysis and Replanning Tool,* para automatizar la planificación y organización logística del transporte. Las técnicas de planificación de IA permitieron que se generase un plan en cuestión de horas que podría haber llevado semanas con otros métodos. La agencia DARPA (*Defense Advanced Research Project Agency*) afirmó que esta aplicación por sí sola había más que amortizado los 30 años de inversión.

- ***Robótica.***

Un ejemplo de la IA en este ámbito sería en el empleo de robot en operaciones de microcirugía. El *HipNav* es un sistema que utiliza técnicas de visión por ordenador para crear un modelo tridimensional de la anatomía interna del paciente y después utiliza un control robotizado para guiar el implante de prótesis de cadera.

- ***Procesamiento de lenguaje y resolución de problemas.***

El programa informático *Prover B*, es capaz de resolver crucigramas mejor que la mayoría de los humanos, utilizando restricciones en programas de relleno de palabras.

3.13.1. Omnisciencia, aprendizaje y autonomía

Es necesario tener cuidado al distinguir entre racionalidad y **omnisciencia.** Un agente omnisciente conoce el resultado de su acción y actúa de acuerdo a él; sin embargo, en realidad la omnisciencia no es posible. No siempre que se actúa racionalmente, significa que es la perfección, porque siempre puede ocurrir un imprevisto que nadie pudiera esperar, como por ejemplo, tumbarte en el sofá, sin ningún tipo de peligro posible, y que de pronto se produzca un terremoto y se venga la casa abajo.

El asunto es que resulta imposible diseñar un agente que siempre lleve a cabo, de forma sucesiva, las mejores acciones después de un acontecimiento. Por lo que para la racionalidad no se requiere la omnisciencia, ya que la elección racional depende sólo de la secuencia de percepción hasta la fecha.

Llevar a cabo acciones *con la intención de modificar percepciones futuras*, lo que se conoce como **recopilación de información**, es una parte importante de la racionalidad. Un segundo ejemplo de recopilación de información lo proporciona la **exploración** sobre algún medio inicialmente desconocido.

Una vez mencionado todo esto, llegamos a la conclusión de que el agente racional no sólo tiene que recopilar información, sino que ha de **aprender** lo máximo posible de lo que está percibiendo. Se pueden dar dos casos, una que la configuración inicial del agente puede reflejar un conocimiento preliminar del entorno, pero a medida que el agente adquiere experiencia éste puede modificarse y aumentar. Y por otro lado, se puede dar el caso en los que se conoce totalmente el entorno a priori. En estos casos, el agente no necesita percibir ni aprender; simplemente actúa de forma correcta. El gran inconveniente de estos agentes es que son muy frágiles.

Se dice que un agente carece de **autonomía** cuando se apoya más en el conocimiento inicial que le proporciona su diseñador que en sus propias percepciones. Un agente racional ha de ser autónomo, es decir, debe aprender cómo tiene que compensar el conocimiento incompleto inicial. Por ejemplo, un agente persiana que aprenda cuando se hace de día o de noche, será más útil y cómodo que otro que no aprenda. En la práctica, pocas veces se necesita autonomía completa desde el comienzo, es decir, cuando el agente haya tenido poca o ninguna experiencia, tendrá que actuar de forma aleatoria a menos que el diseñador le haya proporcionado ayuda. Así, sería razonable proporcionar a los agentes

que disponen de inteligencia artificial un conocimiento inicial, así como la capacidad de *aprendizaje*. Después de las suficientes experiencias interaccionando con el entorno, el comportamiento del agente racional, ya sí se le podrá llamar como *independiente*. De ahí, que la incorporación del aprendizaje facilite el diseño de agentes racionales individuales que tendrán éxito en una gran cantidad de medios.

3.13.2. Aprendizaje Automático

La idea de aprendizaje consiste en utilizar las percepciones no sólo para actuar, sino también para mejorar la habilidad del agente para actuar en el futuro. El aprendizaje entra en juego cuando el agente observa sus interacciones con el mundo y sus procesos de toma de decisiones. Puede ir desde el estudio del almacenamiento de datos debido al cambio de estados, hasta la creación de teorías científicas, como mostró Albert Einstein. En este proyecto se va a tratar el **aprendizaje automático** a partir de observaciones, en el que se pueden distinguir varios campos.

Los ***sistemas recomendadores***, basados en las preferencias y similitudes de los usuarios están cada vez más presentes en los entornos y aplicaciones de uso diario. Un ejemplo es el que se utiliza en la página web de *AMAZON*, donde los usuarios reciben recomendaciones basadas en las compras realizadas por usuarios considerados como semejantes o de gustos similares.

También están los ***sistemas de predicción,*** que se utiliza sobre todo en el ámbito de la economía, para tratar de conocer la futura evolución de los mercados. También es posible realizar tareas de clasificación e identificación, como demuestran muchos programas de ordenador para la hora de detección de elementos maliciosos en la web.

Como en este proyecto se va a trabajar sobre una vivienda, hay que tener en cuenta los diferentes elementos que lo componen, como los diferentes elementos de consumo (en mi caso la energía consumida por los motores de las persianas y el consumo de la iluminación), fuentes de incertidumbre (temperatura exterior y el comportamiento de sus ocupantes) y fuentes de producción de energía (equipos de climatización, placas solar o fotovoltaicas si se tienen, etc).

Estudios previos demuestran que modelos predictivos basados en redes neuronales (González Lanza & Zamarreño Cosme, 2002), (Zhao & Magoulès, 2012), son capaces de predecir tanto la evolución de la temperatura en un edificio como el consumo de sus equipos, lo que permite elaborar medidas correctivas que ayuden a mejorar la *eficiencia energética de los hogares*.

3.14. Computación Ubicua o Inteligencia ambiental.

La informática en los últimos años ha ido evolucionando (*Ilustración 36*) desde sus iniciales aplicaciones orientadas a realizar tareas repetitivas y de almacenamiento de datos sobre las enormes máquinas basadas en electroimanes, hasta la era de los ordenadores personales usados en la mayoría de los hogares. A partir de aquí, la tercera era ha empezado, siendo catalogada como *Computación Ubicua*. Según *Mark Weiser* la **computación ubícua** es un mecanismo por el cual se facilita el uso de compuitadores, haciendo que en nuestro entorno existan múltiples sistemas de computación, pero siendo éstos "imprescindibles" para el usuario (Barro Ameneiro & Bugarín Diz, 2002).

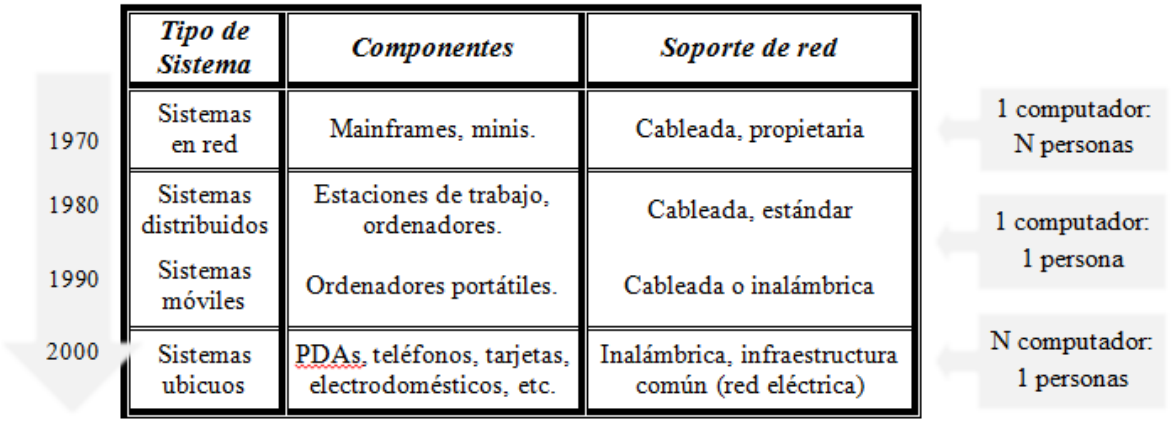


Ilustración 36: Evolución de la Informática hasta los Sistemas ubicuos

La *Computación Ubicua o Internet de las Cosas* pretende la integración de las nuevas tecnologías en el entorno personal, insertando dispositivos inteligentes en las tareas diarias, haciendo que interactúen de forma natural y desinhibida en todo tipo de situaciones y circunstancias. De esta forma se pretende unir el mundo real con una representación virtual, apoyándose sobre la inteligencia ambiental y logrando el entorno inteligente.

La mejora tecnológicas tanto hardware como software, han permitido disponer de dispositivos cada vez más potentes y reducidos, posibilitando su integración en lugares

más estratégicos o simplemente, su uso para nuevas aplicaciones. Esta evolución, permite cubrir nuevos espacios, considerar nuevas aplicaciones, y/o ofrecer nuevas características a los usuarios en pos de la red inteligente global.

Uno de los objetivos más importantes de la Computación Ubicua es integrar los dispositivos computacionales lo más posible, para conseguir que se mezclen en la vida cotidiana, y permitir a los usuarios que se centren en las tareas que deben hacer, en vez de las herramientas que deben usar, por lo que agilizará dichos sistemas. El hecho de enviar la computación a un "segundo plano" tiene dos significados:

- El primero es el significado literal, detallando que la tecnología de la computación se debe integrar en los objetos, cosas, tareas y entornos cotidianos.
- El segundo se refiere a que esta integración se debe realizar de forma que estos elementos no deben interferir en las actividades para las que son usadas, proporcionando un uso más cómodo, sencillo y útil de los mismos.

Los objetos cotidianos en los que se integra la tecnología de computación, tienen una serie de características que permiten y delimitan la creación del entorno ubicuo buscado: Comunicación entre dispositivos, ya que los elementos del sistema disponen no sólo de capacidad de computación, sino también de comunicación, tanto con el usuario como con los demás elementos a su alrededor mediante WiFi, Bluetooth, GPRS/UMTS, UWB, RFID, etc. Disponibilidad de memoria, que puede ser usada para almacenar información y así mejorar la interacción con el resto de dispositivos. Sensibilidad al contexto, siendo capaces de adaptarse a diferentes situaciones, como su situación geográfica, las preferencias de los usuarios, los dispositivos que se encuentran en el entorno, etc., actuando en base a este contexto. Por último, son capaces de reaccionar ante ciertos estímulos o eventos, que pueden percibir en su entorno a través de sensores o mediante interacción con otros dispositivos.

De este modo, la Computación Ubicua, incorpora cuatro nuevos conceptos:

- ***Uso eficaz de espacios "perspicaces":*** Se basa, en la detección del estado de un individuo y de sus necesidades, deducidas de dicho estado, ya sea en la oficina, sala de reuniones, clase, domicilio, coche, etc. El espacio perspicaz surge cuando varios dispositivos inteligentes coinciden en el mismo espacio físico e interactúan

colaborativamente para dar soporte a los individuos que se encuentren en él. La domótica, computación ubicua en el domicilio, es la aplicación más popular.

- ***Invisibilidad:*** Actualmente, se está lejos de la propiedad expuesta por Weiser para los sistemas ubicuos, la completa desaparición de la tecnología de la consciencia del usuario. Una buena aproximación es tener presente, en el diseño de estos sistemas, la idea de mínima distracción del usuario. La invisibilidad va a requerir del cambio drástico en el tipo de interfaces que nos comunican con los computadores. Reconocimiento de voz y de gestos, comprensión del lenguaje natural y del texto manuscrito, en la dirección hombre-máquina y en el sentido contrario, síntesis de lenguaje hablado y escrito y de representaciones gráficas.
- ***Escalabilidad local.*** El concepto de localidad de servicios en computación ubicua es fundamental frente a la universalidad de servicios de Internet. Los usuarios disponen de capacidades asociadas al contexto en el que se encuentran, careciendo de sentido, por ejemplo, que las aplicaciones domóticas situadas en el domicilio particular tengan que estar escrutando las necesidades del usuario que se encuentra trabajando en ese momento en la oficina. Al igual que la mayoría de las interacciones en la naturaleza, la proporcionada por estos sistemas, decrece con la distancia al usuario.
- ***Ocultación de los desniveles de acondicionamiento:*** Dependiendo de la infraestructura y del desarrollo tecnológico disponible, la distribución de los servicios ofrecidos puede ser muy poco uniforme, en esta situación el principio de invisibilidad puede no cumplirse ya que el usuario detectaría desagradables transiciones. Este requisito es hoy día el más alejado respecto de la situación ideal, los sistemas que incorporan computación ubicua están aislados, sin continuidad entre unos y otros.

En la *Ilustración 37* se describe una arquitectura genérica de un dispositivo de computación ubicua, que permite interactuar con el usuario a través de su interfaz (tanto de entrada como de salida), obtener el contexto e información relevante del mundo real para dar el soporte adecuado a sus necesidades y modificar el entorno en base a la información capturada por los sensores y las acciones realizadas a través de los actuadores, y mediante su interfaz de red puede coordinarse con otros elementos del sistema.

Ilustración 37: Modelo de dispositivo de computación ubicua

Hemos podido ir viendo cómo el usuario toma un rol principal en el sistema, por lo que tiene sentido adoptar una perspectiva de diseño centrada en el usuario, a la hora de idear e implementar los posibles servicios y aplicaciones.

3.15. Eficiencia Energética.

La **Eficiencia Energética (EE)** es el conjunto de acciones que permiten optimizar la relación entre la cantidad de energía consumida y los productos, y servicios finales obtenidos. Por eso, ser eficientes con el uso de la energía significa *"hacer más con menos"*.

Usar la energía de manera eficiente nos permite realizar todas nuestras actividades y ahorrar dinero. Usualmente dicha reducción en el consumo de energía se asocia a un cambio tecnológico, ya sea por la creación de nuevas tecnologías que incrementen el rendimiento de los artefactos o por nuevos diseños de máquinas y espacios habitables, los que pueden disminuir la pérdida de energía por calor. No obstante, no siempre es así, ya que la reducción en el consumo de energía puede estar vinculada a una mejor gestión o cambios en los hábitos y actitudes.

Muchas veces cuando se habla de *Eficiencia Energética* se suele confundir con *Ahorro Energético*, que la principal diferencia es que cuando hablamos de Ahorro Energético, nos referimos a dejar de utilizar o consumir menos energía, sin fijarse en el confort humano ni en nada. Esto puede significar reducir o dejar de realizar determinadas actividades, para evitar el consumo de energía. Sin embargo cuando hablamos de Eficiencia Energética nos

referimos a la reducción de consumo de energía, manteniendo los mismos servicios energéticos, sin disminuir el confort ni la calidad de vida, asegurando el abastecimiento, protegiendo el medio ambiente y fomentando la sostenibilidad.

Por ejemplo, el ahorro energético se genera cuando apagamos la luz para reducir el consumo de energía. Si, en cambio, reemplazamos la bombilla incandescente por una eficiente, estamos tomando una medida de Eficiencia Energética, que nos proporcionará una disminución en el consumo de energía, sin perjuicio del desarrollo de nuestras actividades.

Tampoco se debe confundir la EE con la Energía Renovable (ER), esta última corresponde a la energía que se obtiene de fuentes naturales virtualmente inagotables, tales como el sol o el viento. En resumen, la ER es un tipo de fuente de energía, mientras que la EE es un análisis de todo el sistema, que podrá presentar como medidas de reducción de consumo de energía, el uso de ER.

El ***IDEA (Instituto para la Diversificación y Ahorro de la Energía)*** crea un *Plan de Acción de Ahorro y Eficiencia Energética 2011-2020:*

El artículo 4 de la Directiva 2006/32/CE sobre la eficiencia del uso final de la energía y los servicios energéticos fija un objetivo mínimo orientativo de ahorro energético del 9% en 2016.

Por otra parte, el Consejo Europeo de 17 de junio de 2010 ha fijado como objetivo para 2020 ahorrar un 20% de su consumo de energía primaria.

Como consecuencia de estas obligaciones, el Ministerio de Industria, Turismo y Comercio, en colaboración con el IDAE, ha elaborado el Plan de Acción de Ahorro y Eficiencia Energética 2011-2020, que incluye un anexo con la cuantificación de los ahorros energéticos obtenidos en el año 2010 respecto a los años 2004 y 2007, de acuerdo con las recomendaciones metodológicas sobre medida y verificación de los ahorros de la Comisión Europea. Ambos documentos fueron aprobados por el Consejo de Ministros del 29 de julio de 2011.

Por todo esto, es fundamental fomentar la Eficiencia Energética debido a que es la forma más económica, segura y limpia de utilizar la energía.

Para poder conseguir un buen ahorro energético en una vivienda, se ha de buscar una *construcción bioclimática*, *la elección de equipos de calefacción, electrodomésticos e iluminación, más eficientes* y la *modificación de los hábitos de consumo de sus habitantes*.

- ***Calefacción y agua caliente.***

La **calefacción** puede ser objeto de ahorro de energía principalmente con *hábitos de consumo* tales como un uso racional del mismo, el consumo total de una vivienda suele ser del 46 % del total del consumo (pudiendo alcanzar el 60 % si se incluye el agua caliente). El ahorro de energía puede producirse bien por la correcta elección de una caldera eficiente, o por el correcto aislamiento térmico de las habitaciones.

Se debe tener presente que una temperatura para un hogar está entre los 19 y los 21 °C por el día, y 15 a 17 °C por la noche, cada grado aumenta el consumo en un 7 %. Con estas consideraciones ***se aconseja***:

- Adecuar el vestido en el domicilio con las condiciones de temperatura, se pueden emplear edredones, mantas y prendas similares.
- No tapar u obstruir los radiadores ya que su función es la de emitir calor, y esta se ve entorpecida con la colocación de muebles.
- Vigilar el aislamiento de las habitaciones, impidiendo fugas de calor o entradas de aire frío procedente de ventanas abiertas.

Respecto del **agua caliente** puede emplearse también como ayuda la *energía solar térmica*, mediante uso de sistemas de almacenamiento de energía que retengan el calor para que el agua caliente esté disponible la mayor parte de tiempo posible.

El empleo del agua caliente se realiza en la vivienda bajo ciertas ocasiones muy específicas como puede ser la ducha, o el baño, limpiando los platos y la cubertería, etc. En todos ellos ***se aconseja***: Emplear agua caliente sólo cuando se necesite, al lavar no siempre se necesita.

- ***Electrodomésticos.***

La normativa europea expresa la eficiencia energética de los **electrodomésticos** en una escala de 7 clases de eficiencia, y se identifican mediante un código de color y letras que van desde el verde y la letra A, para los equipos con mayor eficiencia, hasta el color rojo y la letra G para los equipos de menor eficiencia. Un electrodoméstico de clase A puede

llegar a consumir un 55 % menos que el mismo en una clase media, la elección de un electrodoméstico con esta información puede suponer un ahorro económico.

Respecto a los hábitos, por regla general inciden sobre un uso racional y en un correcto mantenimiento de los mismos:

- ***Refrigerador:*** Mantener bien cerrada la puerta en todo momento y preferir abrir el portalón una vez que innumerables veces.
- ***Lavadora y lavavajillas:*** Planificar los lavados, de tal forma que cada lavado tenga su máxima carga. La lavadora consume casi igual a plena carga que a media.

➢ ***Iluminación.***

La **iluminación eléctrica** en las viviendas suele suponer entre el 18 % y el 20 % del consumo doméstico, en algunos casos basta con una actitud preventiva adquiriendo por ejemplo lámpara de bajo consumo, poniendo múltiples fuentes de luz de bajo consumo en lugar de uno, aumentando la superficie de las ventanas.

Para ahorrar basta con adquirir hábitos, como por ejemplo:

- Apagar luces en estancias donde no se habite
- Emplear una fuente de luz eliminando las fuentes luminosas redundantes
- Si se dispone de la opción abrir ventanas y emplear la luz natural en lugar de la artificial

CAPITULO 4: CONTROL DIFUSO E INTELIGENTE DE LA ILUMINACIÓN DE UNA VIVIENDA

4. CONTROL DIFUSO E INTELIGENTE DE LA ILUMINACIÓN DE UNA VIVIENDA.

4.1. Introducción

Como ya se ha comentado con anterioridad, lo que se busca con este proyecto es la construcción de un entorno de pruebas para poder simular la automatización de las persianas e iluminación de una vivienda real. En el que se ha empleado la lógica difusa para lograr un control mucho más personal y cercano sobre el usuario, ya que en cualquier momento se podrá adaptar el sistema a gusto del usuario, es decir, él podrá elegir cuando hace mucha o poca luz, o mucho o poco calor, y decidir cómo quiere que actúe la persiana.

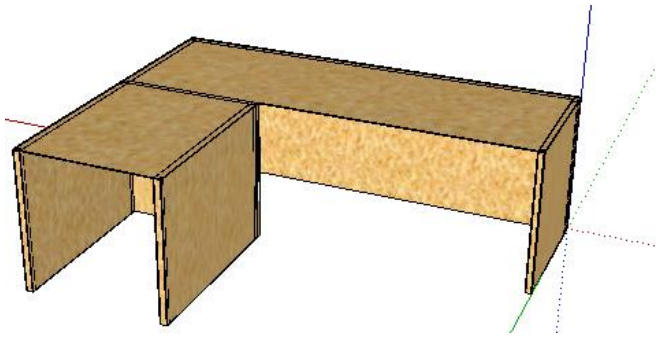


Ilustración 38: Prediseño del Entono de Pruebas

Para la construcción de nuestro entorno de pruebas lo primero que se ha hecho, ha sido un previo diseño de la maqueta a ordenador (*Ilustración 38)*, mediante el programa informático de *SketchUp*, para así poder estimar la dimensión de dicha maqueta, y de la cantidad de los distintos componentes que se van a necesitar. Para ello, se ha ido diseñando con unas medidas aproximadas (*Ilustración 39)*, para facilitar aún más su posterior construcción.

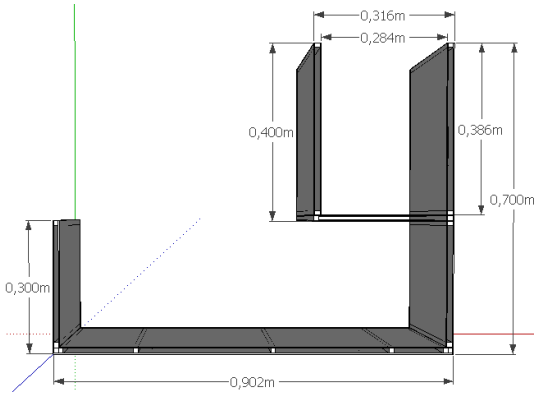


Ilustración 39: Medidas del Entorno de Pruebas

4.2.Componentes del entorno de pruebas

Tras un previo estudio sobre qué es lo que se busca sobre este proyecto, se va a optar por el control de las persianas y luces, teniendo en cuenta la iluminación y temperatura exterior, y la presencia, para lo que se va a necesitar de un sensor de luminosidad (LDR), otro de temperatura (aunque en nuestro caso se utilizará un potenciómetro, para una mejor simulación), de un sensor PIR, motores para la automatización de las persianas (servomotores), leds para simular la iluminación de la vivienda y un microcontrolador para poder controlar todos los distintos elementos, que en nuestro caso usaremos el *Arduino*. A continuación se detallarán cada una de los distintos componentes.

4.2.1. Arduino ATMega2560

Anteriormente, en el *apartado 3.11,* ya se explicó que es el Arduino, y cuáles eran las características de las distintas placas de Arduino.

Para la elaboración de este proyecto, hubiera servido el *Arduino UNO*, que es algo más barato (unos 6€ más barato), pero se ha decidido utilizar el ***Arduino ATMega2560 (****Ilustración 40****)***, ya que dispone de casi 4 veces más pines digitales, por lo que si en un futuro se quisiera añadir algún detalle más al entorno de pruebas, no supondría ningún inconveniente.

También decir, que el *Arduino ATMega2560*, será el encargado de controlar y gestionar las persianas e iluminación de una vivienda cualquiera, en este caso, sobre el entorno de pruebas, como ya se explicará más adelante.

A continuación se describirán las distintas partes del *Arduino ATMega2560:*

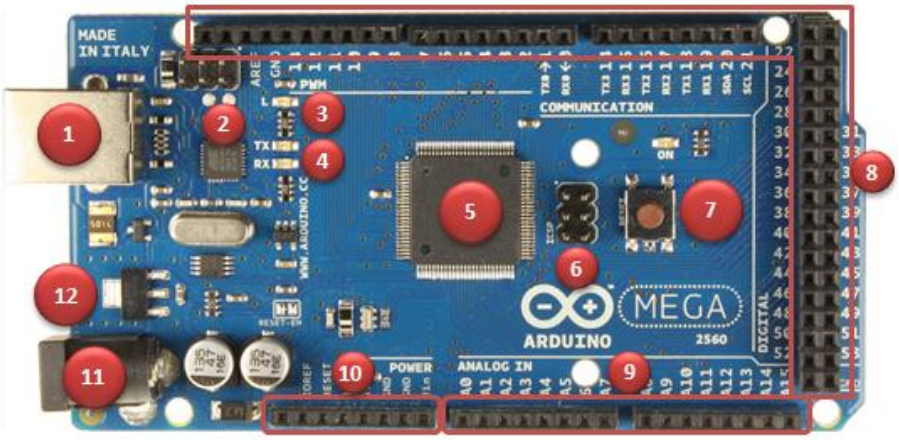


Ilustración 40: Partes del Arduino ATMega2560

1) Conector USB para el cable tipo AB
2) ATMega 2560 encargado de la comunicación
3) Led naranja conectado al pin13
4) Led TX (Transmisor) y RX (Receptor) de la comunicación serial
5) Microcontrolador ATMega 2560
6) Puerto ICSP para programación serial
7) Botón para resetear el microcontrolador
8) Pines E/S digitales, PWM y de comunicación
9) Entradas analógicas
10) Pines de voltaje y tierra
11) Conector hembra 2.1 mm con centro positivo
12) Regulador de voltaje

Los distintos componentes y pines que se utilizan en el proyecto viene explicado en el apartado *4.4 Aspecto y esquema de conexiones del entorno de pruebas.*

4.2.2. Servomotor Tower Pro SG90

El motor que se va a utilizar para la automatización de las persianas será, el *Servomotor Tower Pro SG90 (Ilustración 41).*

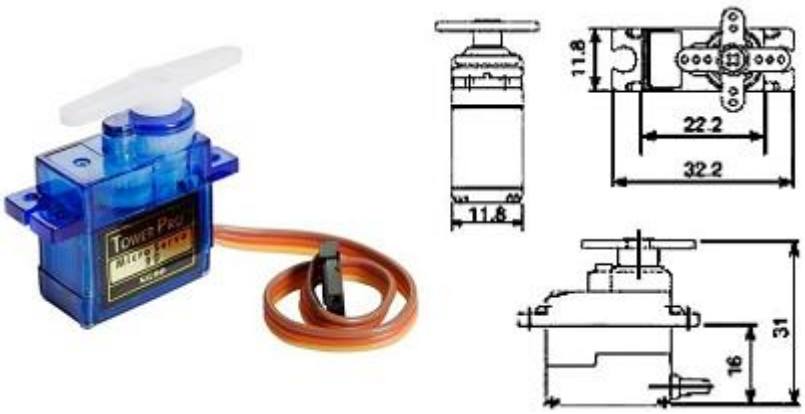


Ilustración 41: Apariencia y medidas del Servomotor Tower Pro SG90

Se ha elegido este motor en concreto, por ser bastante económico y principalmente por ser el que mejor se adaptaba a nuestra maqueta, gracias a su reducido tamaño, pudiendo realizar sin ningún inconveniente la tarea de hacer subir o bajar la persiana. Sus principales características son:

- **Medio de control:** PWM
- **Torque:** 1.6Kg/mm (4.8V) a 1.8Kg/mm (6V)
- **Voltaje de alimentación:** 4.8V a 6V
- **Compatible con Arduino**

- **Velocidad HI-SPEED:** 0.10 seg/60º (4.8V) a 0.08 seg/60º (6V)
- **Peso:** 9 g
- **Mecanismo:** Nylon Reforzado
- **Tamaño:** 2,25 x 1,22 x 2,8 cm
- **Cables: *Rojo* (Alimentación +), *Marrón* (tierra), *Naranja* (Señal PWM).**

Comercialmente están disponibles en tres versiones: *normales* (con una rotación de 180º), *rotativos* (limitados a unas pocas vueltas) y de *rotación continua*. Estos últimos son interesantes porque añaden a la característica de poder mover cargas pesadas la de poder girar de forma continua en un sentido u otro, lo que los hace ideales para utilizarlos como elementos de tracción.

En la maqueta nos encontraremos son cuatros servomotores *normales* iguales que el de la *Ilustración 41*, uno por cada ventana. Estos servomotores tienen el inconveniente que solo tienen un giro de 180º como máximo, es decir, que no tienen un giro completo y continuo. Pero, se puede llegar a conseguir que cualquier servo trabaje como un *servo de rotación continua* realizando unas modificaciones.

Básicamente se pueden modificar de 2 formas muy parecidas, sólo se va a explicar a fondo una de ellas, que es la que se ha realizado con los servomotores de esta maqueta; mientras que de la otra forma solo se va a mencionar como se haría.

Lo primero que habría que hacer, será quitar los cuatro tornillos que hay en la parte inferior del servomotor, para así poder extraer las tapas superior e inferior.

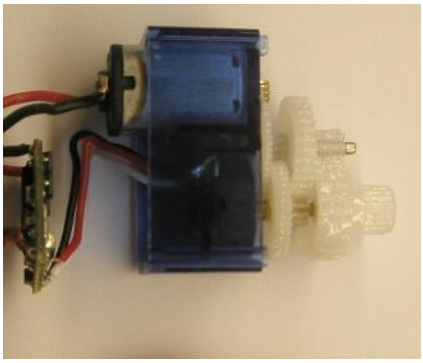


Ilustración 42: Servomotor sin las tapas superior e inferior

Una vez que se tiene acceso a los engranajes, se extraer todos los engranajes con cuidado de no perderlos, y si se puede, a la hora de guardarlos, hacerlo en función al orden que les

corresponde en el montaje, para que cuando se vuelva a montar que no haya ningún problema.

Cuando ya se hayan extraído todos los engranajes, se coge el de salida (el mayor), y se podrá observar que tiene un tope mecánico, que habrá que cortárselo con unos alicates o cúter, y después habrá que limarlo para que pueda girar libremente. También, mediante una broca de 1'5mm, habrá que eliminar el rebaje del eje que tiene en el interior del piñón.

Por último lo se va a buscar, mediante un multímetro dejar el potenciómetro, en su valor medio (el potenciómetro girado a 90º), ya que de no hacer esto, el motor estará continuamente girando, y nunca se conseguiría detenerlo. Una vez que se tiene el potenciómetro en su valor medio, se bloque el potenciómetro, por la parte superior, echándole cualquier pegamento instantáneo, estaño, etc.

Una vez bloqueado el potenciómetro, se vuelve a montar todo como estaba, y ya se habrá conseguido que estos servomotores tengan un giro completo.

El otro método habría sido todo igual, pero con la diferencia, que para anular el potenciómetro habría que desconectarlo de la placa base y conectar dos resistencias del mismo valor (con tolerancia 1%). Éste método es más fiable, pero más laborioso. Al no disponer de estaño ni soldador de estaño, es por eso, por lo que se eligió el primer método.

4.2.3. Sensor piroeléctrico infrarrojo 'PIR' *HC-SR 501*

Este sensor es el encargado de detectar la presencia a través de infrarrojos, produciendo una salida digital de 3,3V cuando detecta algún tipo de presencia. Se puede ajustar tanto el tiempo de activación de la salida, como la sensibilidad del sensor.

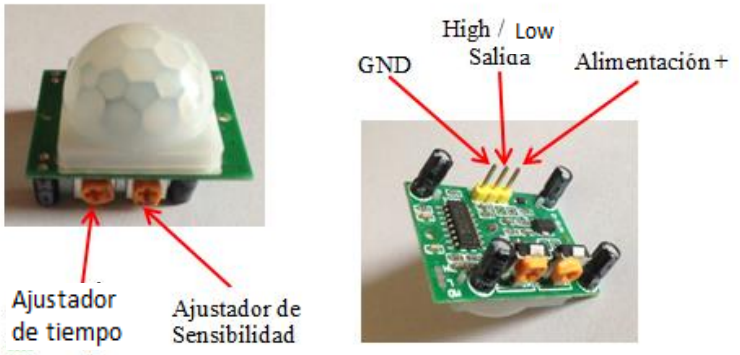


Ilustración 43: Aspecto del sensor PIR HC-SR 501

Sus datos técnicos son:

- **Voltaje de alimentación:** 4.8V a 6V
- **Niveles de salida:** *High* 3,3V / *Low* 0V
- **Modos de disparo:** en *Low* no se repite el disparo y en *High* si se repite
- **Tiempo de disparo:** desde 3s hasta 300s (ajustable)
- **Lente Fresnel de 19 zonas:** ángulo <110º
- **Temperatura de trabajo min y máx.:** -15ºC y 70ºC
- **Tamaño de la lente de sensor:** 23mm de diámetro
- **Tamaño de tarjeta:** 3,2 x 2,4 x 2,6 cm

Se suele utilizar para el control inteligente de iluminación, domótica, sistemas de alarmas, ahorro de energía en iluminación y ventilación, etc. Por lo que, tiene las cualidades que se requieren para este proyecto. Su función, como ya se ha mencionado anteriormente, van a ser los encargados de detectar presencia, y dar la señal para encender las luces internas. Se podría llegar a utilizar sobre un entorno real, como detector de presencia.

4.2.4. Sensor LDR (CE-C2795).

Las ***fotorresistencias o LDR*** dependen de la luz y está formado por una célula de Sulfuro de Cadmio, altamente estable, encapsulada con una resina epoxi transparente y resistente a la humedad. La respuesta espectral es similar a la del ojo humano.

Ilustración 44: Aspecto del sensor LDR CE-C2795

Como podemos observar en la siguiente gráfica (*Ilustración 45*), Su nivel de resistencia aumenta cuando el nivel de luz disminuye, y viceversa.

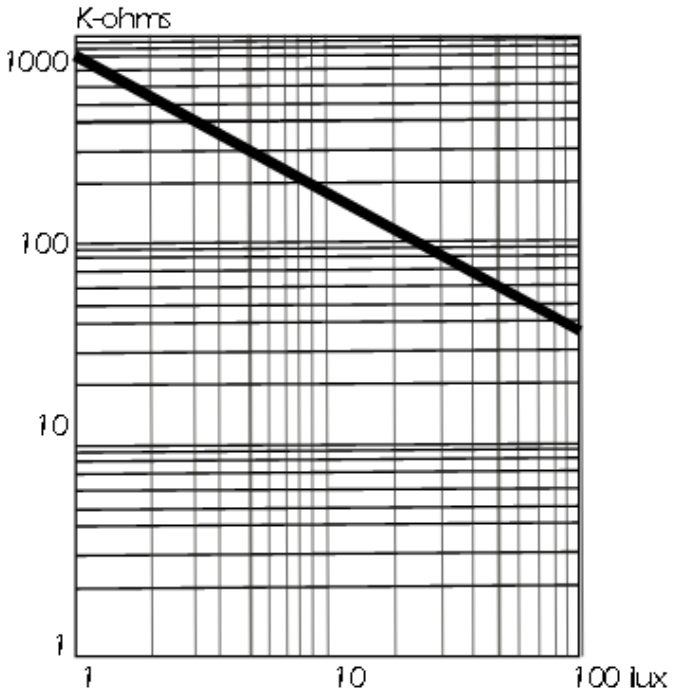


Ilustración 45: Comportamiento del sensor LDR CE-C2795 en de la iluminaria

Con este sensor lo que se busca, es que capte la iluminación que tenemos en el exterior, para así poder mandar señales de cuando está oscuro el día, si es de noche o si es de día; para así, poder mantener un control de las persianas, en función de la cantidad de luz que tengamos en el exterior, para subir o bajar las persianas. Se podría llegar a utilizar en un entorno real.

4.2.5. Pantalla LCD

Una ***LCD*** (*pantalla de cristal líquido* o liquid crystal display) es una pantalla delgada y plana, que disponen de un gel líquido cristalino que se intercala entre dos paneles de vidrio o plástico de visualización. El gel se divide en pixeles por una rejilla de transistores de metal, que suministran a cada píxel con una corriente eléctrica diferente, lo que les permite iluminar en diferentes colores.

Las pantallas o display LCD, tienen la capacidad de mostrar cualquier carácter alfanumérico, permitiendo representar la información que genera cualquier equipo electrónico de una forma fácil y económica. El proceso de visualización es gobernado por un microcontrolador incorporado a la pantalla, siendo el Hitachi 44780 el modelo de controlador más utilizado.

Existen muchos tipos de LCD (distintos tamaños, colores, iluminados, sin iluminar...) pero el funcionamiento general de todos es el mismo.

En este proyecto se va a utilizar una pantalla LCD (*Ilustración 46)* de 16 píxeles de largo por 2 de alto, es decir, vamos a disponer de 2 filas, en las que cada fila vamos a disponer de 16 caracteres.

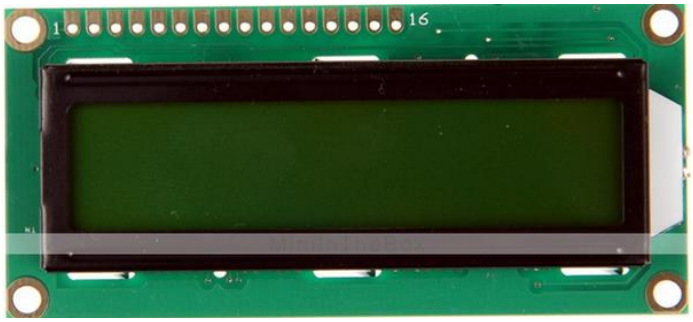


Ilustración 46: Aspecto de la parte superior de un Pantalla LCD 1602

En la *Ilustración 46* podemos observar en la parte superior izquierda que disponemos de 16 pines, que están enumerados del 1 al 16 de izquierda a derecha, tal y como aparece en la foto.

En la *Tabla 4* aparece la función de cada uno de los pines.

PIN	NOMBRE	FUNCIÓN
1	GND	Tierra
2	Vcc	5v
3	Contraste	Regula el contraste de la pantalla.
4	RS	Selección del registro a leer o escribir
5	R/W	selecciona si vamos a leer o escribir
6	E	Enable
7	DB0	Datos
8	DB1	Datos
9	DB2	Datos
10	DB3	Datos
11	DB4	Datos
12	DB5	Datos
13	DB6	Datos
14	DB7	Datos
15	LEDA	Fuente de alimentación para la retroalimentación del led
16	LEDK	

Tabla 4: Funciones de los pines de la Pantalla LCD

Por lo general las pantallas LCD para Arduino suelen necesitar bastantes pines digitales para funcionar, de 6 a 13 según la pantalla, y eso hace que nuestro Arduino se quede sin pines para conectar otras cosas.

Pero eso tiene fácil solución, simplemente hace falta usar un módulo que convierta la conexión en paralelo de la pantalla a conexión en serie mediante alguno de los protocolos de comunicación que soporta Arduino.

En cuanto a los módulos podemos encontrarlos con conexión ISP, Serial e I2C, siendo este último el más común de todos y del que trata esta entrada.

- ***Módulo bus I2C***

El ***I2C*** *(Ilustración 47)* es un bus de comunicaciones en serie que utiliza dos líneas para transmitir la información a la LCD, una para los datos y por otra la señal de reloj.

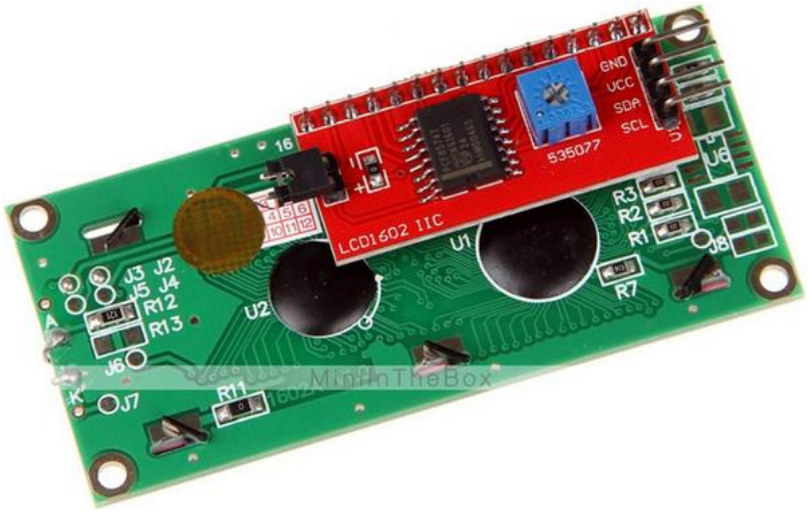


Ilustración 47: Aspecto del Módulo I2C incorporado a la pantalla LCD

4.2.6. Otros Componentes

Para la elaboración de la maqueta, también se van a usar los siguientes componentes:

- ***Diodos LEDs*** 5 mm, para la iluminación interior.
- ***Potenciómetro*** para simular un sensor de temperatura.
- ***Contrachapado*** para las paredes y el techo.
- ***Cable Jumper*** de 2,54 mm, para toda la instalación.

4.3.Montaje del entorno de pruebas

Para el montaje de dicha maqueta se ha optado por utilizar contrachapado de 3mm de grosor, para la construcción de las paredes. Para la construcción de cada pared, como se puede observar en la *Ilustración 48*, va a constar de dos láminas de contrachapado, para así poder introducir el Servomotor y todo el cableado necesario, y conseguir que no esté a la vista. Para ello se han utilizado unas columnas de madera de 1.3 cm de grosor, para que el Servomotor (1.22 cm) pueda entrar sin ningún problema.

Ilustración 48: Contrachapado usado en el entorno de pruebas antes de ser cortada

Antes de comenzar la construcción, surgieron algunas dudas, respecto a cómo unir las distintas paredes entre sí. Al final, se optó por que las láminas de contrachapado exteriores de cada pared, fuesen encoladas y clavadas, mientras que las láminas de contrachapado interiores fuesen atornilladas, para así, si surgiese algún tipo de inconveniente con algún motor, cualquier problema de conexionado, problemas con los sensores, etc…, poder tener una vía de acceso a ellos, ya que en cualquier momento podríamos desatornillar la pared interna y manipularlos.

A continuación se irá viendo paso a paso el montaje de la maqueta:

Lo primero que se hace es cortar todas las distintas maderas en su justa medida, que se sabe ya que se hizo un prediseño (*Ilustración 49*) anteriormente con el programa informático *SketchUp*.

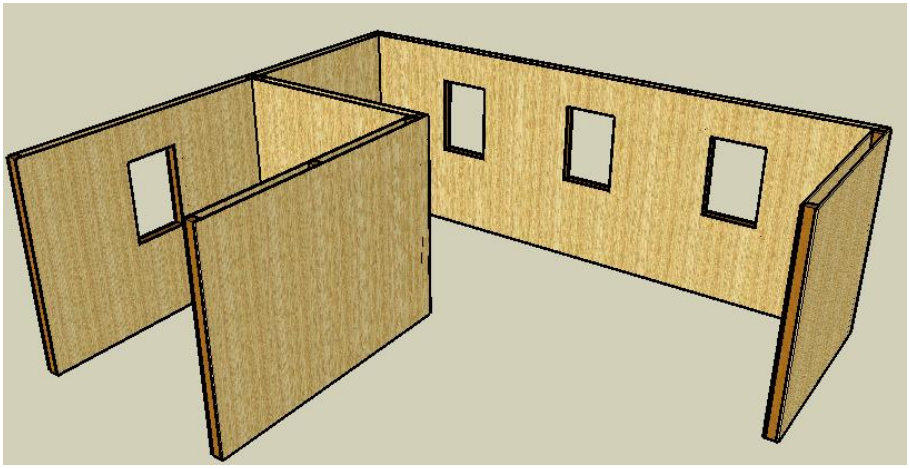


Ilustración 49: Prediseño del entorno de pruebas, mediante SketchUp

Una vez que se tienen todas las paredes cortadas. El siguiente paso es hacer los huecos de las ventanas, con mucho cuidado para que coincidan las dos láminas de contrachapado que tienen cada pared. Para cortar los distintos tableros de contrachapado se ha utilizado una sierra de calar.

Cuando se tienen cortadas todos los tablones en su medida, y teniendo el hueco de las ventanas hechas, se empieza a montar los paneles exteriores de la maqueta, encolando y clavándolas a las columnas de madera (*Ilustración 50).*

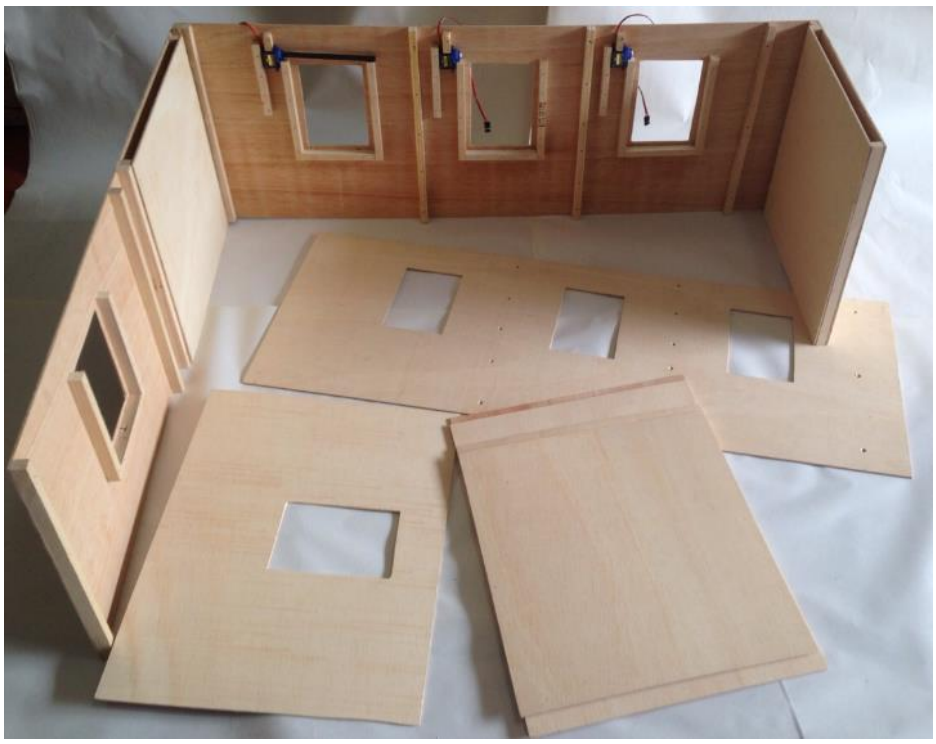


Ilustración 50: Montaje de las paredes del entorno de pruebas

A continuación se comenzará a colocar los distintos servomotores y sensores. Para la sujeción de los servomotores se ha optado, por aprovechar las columnas de madera sobrante, para conseguir encajar el servomotor, y así, cuando se coloque la pared interna quede totalmente sujeta. En el otro extremo hemos colocado otra madera con un agujero, en el cuál se alojará el eje rotatorio de la persiana.

Para la elaboración de las persianas, se ha utilizado un tubo de regadío de 4.55mm de diámetro interior como eje de las persianas, y se han cortado a 12 cm de longitud. Para la simulación de las lamas de la persiana, se ha utilizado tela de paraguas, ya que se pueden enrollar y desenrollar, sin llegar a coger forma. Las medidas de las telas del paraguas son 18 cm de largo por 10 cm de ancho. En la *Ilustración 51* se puede observar cómo quedaría la persiana terminada.

Ilustración 51: Colocación de las persianas en el interior del Entorno de Pruebas

Los sensores LDR se han colocado en la parte superior derecha de cada ventana (*Ilustración 52*), para así conseguir un control óptimo de cada una de los servomotores de forma individual.

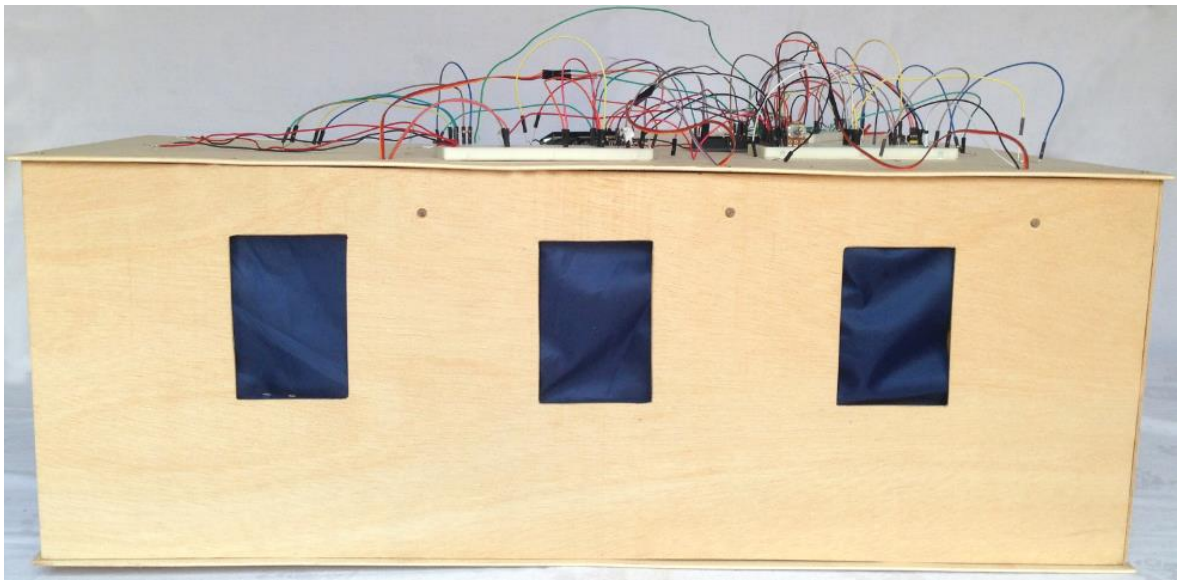


Ilustración 52: Colocación de los sensores LDRs sobre el entorno de pruebas

Una vez que tenemos todos los componentes de las paredes colocados (persianas y LDRs), pasamos a cortar el listón de madera en dos trozos, que servirán para formar la base y el techo del entorno, con lo que obtendremos una mayor sujeción.

Sobre el tejado se harán una serie de huecos, sobre los que se alojarán los distintos puntos de iluminación (leds) y los sensores de presencia (PIR). La disposición de las mismas se puede observar en la *Ilustración 53*, donde se han colocado 16 leds para la iluminación en la zona más grande junto a un sensor PIR, mientras que en la zona de menor tamaño, se han colocado 5 leds y un sensor PIR.

Ilustración 53: Distribución de sensores PIR y leds, de la habitación pequeña (izquierda) y habitación grande (derecha) sobre el tejado del entorno de pruebas

Una vez que se han colocados los distintos componentes del entorno de pruebas, se procederán a la conexión con Arduino, que estará alojado en el tejado del entorno de pruebas. Los tipos de conexionados se explicaran en el siguiente apartado.

4.4. Aspecto y esquema conexiones del entorno de pruebas

En este apartado explicaremos como se han de conectarse todos los componentes individualmente y colectivamente, que se emplean en la creación del entorno de pruebas, con el Arduino (Para este proyecto se va a usar cables Jumper de 2.54mm)

➢ *Conexión de los servomotores*

Como podemos observar en la *Ilustración 41: Apariencia y medidas del Servomotor Tower Pro SG90 (apartado 4.2.2)*, estos servomotores no requieren de componentes externos para conectarse y disponen de una interfaz de conexión de tres cables, de colores *marrón* (puede darse el caso que sea negro en algunas ocasiones), *naranja* (puede darse el caso que sea azul, amarillo u otros) y *rojo.*

Por forma general el cable *rojo* es el positivo o fase, el *marrón o negro* es la tierra o masa, y el cable restante, puede no tiene un color especificado, pero en este caso será el de color *naranja*, se usará para enviar/recibir señales de PWM, que nos permitirá controlar el servomotor con precisión.

Para poder controlar el servomotor, se deberá conectar en uno de los pines que estén marcados como PWM en Arduino (en este caso en 1 de los pines entre el 2 y el 13, ya que se trabaja con un Arduino ATMega2560).

Una posible forma de conexión es la que se observa en la siguiente *Ilustración 54.*

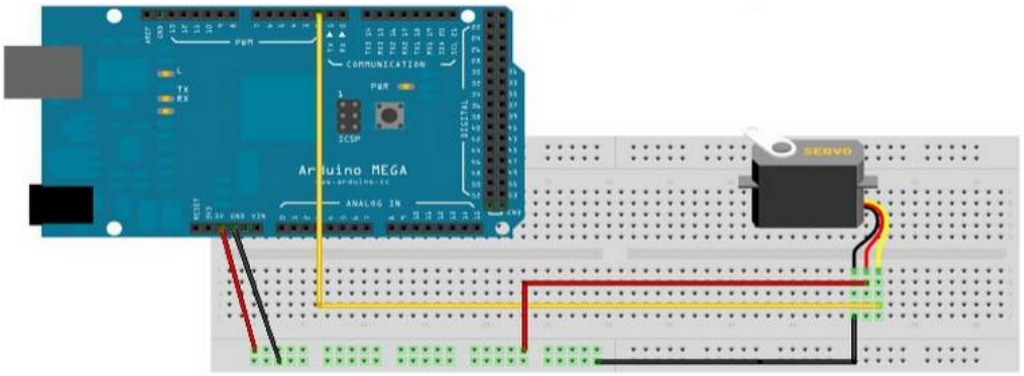


Ilustración 54: Ejemplo de conexión de un servomotor con Arduino ATMega2560

Tal y como se puede observar, se conecta el servo al positivo que nos genera Arduino, y al GND indistintamente, y por último se conecta el cable amarillo (en nuestro caso naranja) al pin número 2, marcado como PWM.

Por último decir, que estos servomotores se alimentan con 6v, pero no hay problema en conectarlo a los 5v sobre los que trabaja Arduino. Pero siempre se tendrá que alimentar al Arduino con una fuente de alimentación externa, ya que los motores consumen bastante corriente, por lo que si se hace esto se podría dar el Arduino. En este proyecto como se va a usar 4 servomotores, se va a usar una fuente de alimentación de 9 voltios.

➢ *Conexión del Sensores PIR*

El sensor de movimiento PIR HC-SR501, como se observa en la *Ilustración 43: Aspecto del sensor PIR HC-SR 501*, dispone de 3 pines de conexión de alimentación (+5v), señal (3.3v) y GND (tierra). Para conectarlo a Arduino se necesitará de 3 cables, siempre se suele usar los mismos colores que se mencionaron anteriormente, rojo para alimentación, negro o marrón para tierra y el resto de colores para la señal, en este caso se conectará a uno de los pines digitales de Arduino, ya que dicho sensor genera salidas digitales (0 cuando no hay presencia y 1 cuando sí lo hay). La conexión con Arduino quedaría de la siguiente forma:

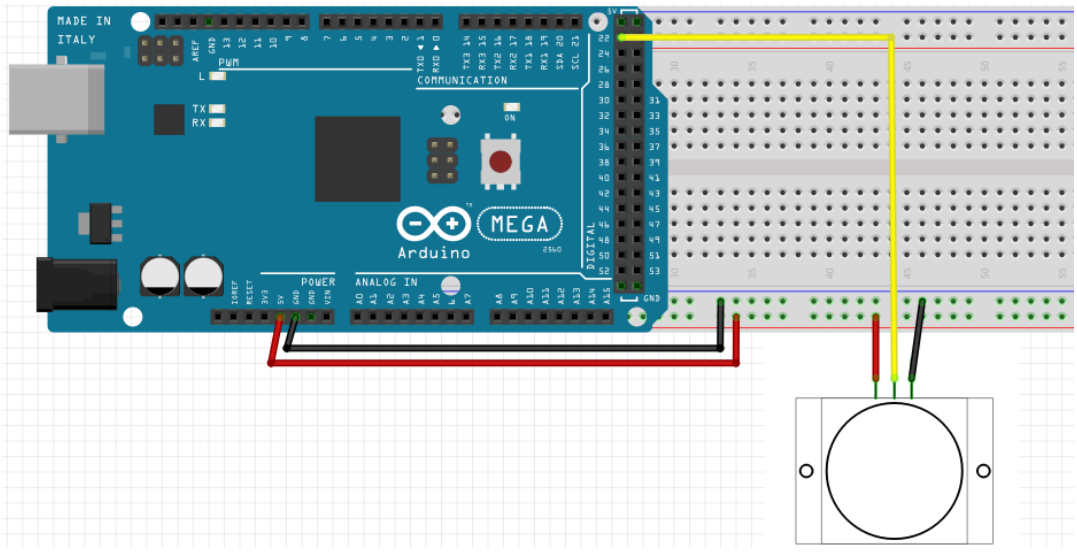


Ilustración 55: Ejemplo de conexión de un Sensor PIR con Arduino ATMega2560

También dispone de dos resistencias variables de calibración, CH1 y RL2. Con *CH1* se puede controlar el tiempo que está activa la salida del sensor, y con RL2, permite ajustar la distancia de detección.

➢ *Conexión del sensor LDR*

Para poder medir valores con un sensor LDR, tiene dos formas de conexión, tal y como se observa en la siguiente *Ilustración 56,* ya sea con una configuración *Pull-up* (como la conexión de la izquierda) o *Pull-down* (derecha).

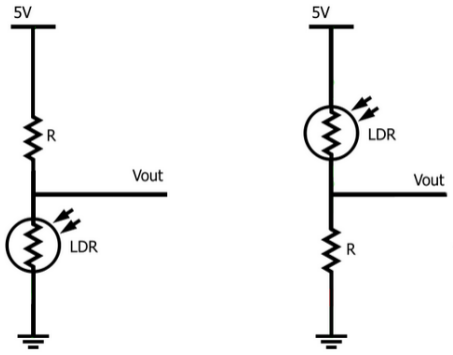


Ilustración 56: Las dos posibilidades de conexión de un sensor LDR (Pull-up y Pull-down)

Con la configuración ***Pull-up***, la tensión de salida V_{out}, irá en aumento a medida que el sensor capte más luz, y por el contrario, la V_{out} disminuirá cuando haya falta de luz. Y con la configuración ***Pull-down***, va a ocurrir justamente lo contrario, es decir, V_{out} aumentará cuando menos luz capte, y disminuirá cuanta más luz haya.

En la siguiente *Ilustración 57* se puede observar cómo se conecta a Arduino, se usará la forma de conexión *Pull-up*, que es la que mejor nos conviene. Ya sea en un tipo de conexión u otra, la V_{out} siempre se conecta a una de las *entradas analógicas*, ya que el sensor LDR nos va a producir señales analógicas.

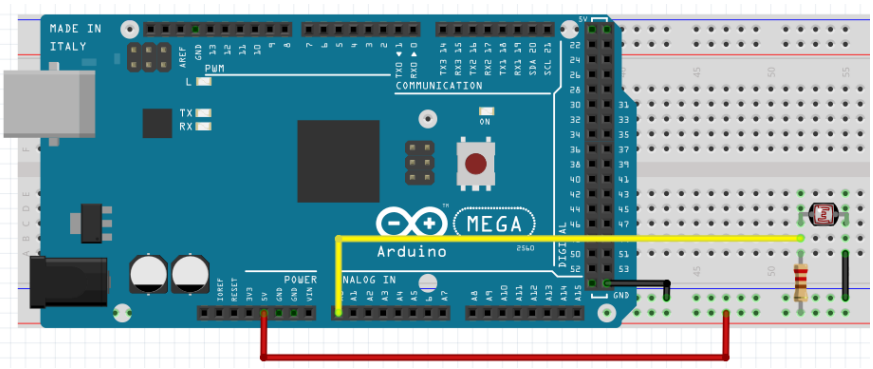


Ilustración 57: Conexión Pull-up del sensor LDR con Arduino ATMega2560

A la hora de conectar el LDR con Arduino no importa si se hace de una forma u otra, ya que siempre se podrá invertir los valores del LDR, metiendo el código '*map*', que se /explicará en el siguiente apartado como hacerlo

➢ *Conexión de la Pantalla LCD con módulo I2C*

Gracias al uso del **módulo I2C** (el cual se puede comprar por separado o directamente soldado a la pantalla LCD), se consigue poder disponer de muchos más pines en el Arduino, ya que sólo tendremos que usar 2 cables para su comunicación (*Ilustración 58*).

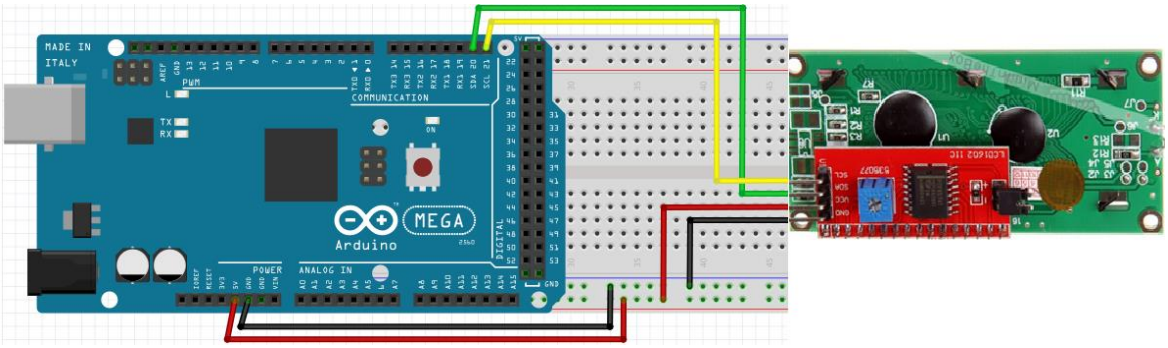


Ilustración 58: Conexión de pantalla LCD I2C con el Arduino ATMega2560

En la mayoría de las placas Arduino, SDA (línea de datos) está en el pin analógico 4, y SCL (línea de reloj) está en el pin analógico 5. En Arduino Mega, SDA está en el pin digital 20 y SCL en el 21.

Alimentaremos el módulo con los pines de alimentación y tierra provenientes de la tarjeta Arduino, para los que se utilizaran un cable rojo y otro negro. Para la comunicación necesitaremos otros 2 cables, uno verde para SDA (datos) y otro amarillo para SCL (reloj). El color de los últimos dos colores no obedece ninguna normativa, por lo que podría servir cualquier color.

➢ *Conexión del potenciómetro*

Como ya se ha mencionado anteriormente se va a utilizar un potenciómetro, para la simulación de un sensor de temperatura.

Un potenciómetro dispone de tres pines de conexión, entre los dos pines de sus extremos siempre existe un valor fijo de resistencia, por lo que indistintamente una se conectará a tierra y la otra a la alimentación (cables negro y rojo), y el pin central a una entrada analógica, ya que se busca obtener entradas analógicas.

La conexión del potenciómetro con Arduino queda de la siguiente forma (*Ilustración 59*):

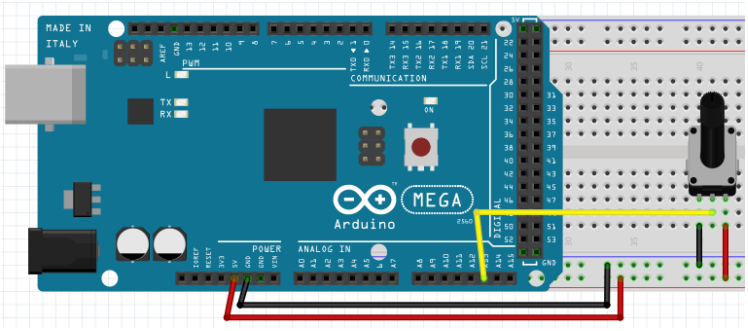


Ilustración 59: Conexión de un potenciómetro a Arduino ATMega2560

➢ *Conexión de un Diodo LED*

Para conectar un diodo led hay saber distinguir cuál de las pines del led es el ánodo (positivo) y cuál es el cátodo (negativo). El pin del led más largo es el ánodo y el más corto el cátodo.

Una vez que se ha distinguido entre ánodo y cátodo, se conecta el ánodo a un pin PWM de Arduino y el cátodo a tierra. Entre el pin PWM de Arduino y el ánodo, se ha de conectar una resistencia de 330Ω o 320Ω, para proteger al diodo led, ya que funcionan con bajas tensiones y se podría llegar a romper.

La razón de conectar el diodo led a una salida PWM, en vez de una salida digital, es porque se busca tener un control gradual de las leds, en vez de un control simple de apagado y encendido de los leds, es decir, 0 y 1.

Para este proyecto, se ha decidido dividir los **21 leds** que se van a utilizar para la **iluminación del entorno de pruebas**, tal y como se mencionó anteriormente.

Por un lado se van a utilizar **16 leds** para la **habitación de mayor tamaño**, donde para obtener un control más eficiente se han los 16 leds dividido en tres grupos (2 grupos de 5 leds cada uno y otro de 6), donde cada grupo se conectarán a una salida PWM de Arduino y serán controladas con un sensor LDR; lo único que van a tener en común los 16 leds de la habitación grande, es que se van a activar con el mismo sensor PIR.

Y por otro lado para la **habitación de menor tamaño** se van a utilizar los **5 leds** restantes, en los que también se agruparán formando un único grupo, conectado a un pin de PWM de Arduino, y controlados a través de un sensor LDR y PIR distintos a los usados en la habitación de mayor tamaño. Con esta forma de conexión conseguimos que si a una zona

de la habitación le llega más intensidad de luz exterior que a la otra, se encenderán solamente las luces necesarias, por lo que estaremos ahorrando energía eléctrica.

La conexión de los distintos grupos de leds, sería igual que la conexión que se puede apreciar en la *Ilustración 60*.

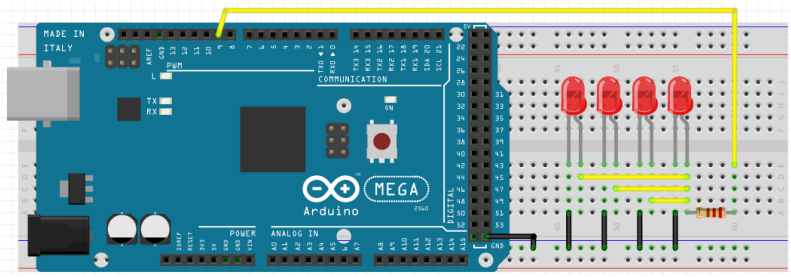


Ilustración 60: Conexión de un diodo LED a Arduino ATMega2560

➢ *Conexión de todos los componentes a Arduino*

Una vez que ya se han explicado la conexión por separado, de cada uno de los componentes del entorno de pruebas, mediante la *Tabla 5*, se va a especificar cuáles son los pines que se van a utilizar del Arduino Mega.

Componente	*Nº de pin*	*Tipo de señal*
Pantalla LCD	20 (SDA)	**COMUNICACIÓN**
	21 (SCL)	
Potenciómetro	A15	**ANALÓGICAS**
Sensor LDR 1	A0	
Sensor LDR 2	A2	
Sensor LDR 3	A4	
Sensor LDR 4	A6	
Servomotor 1	2	**PWM**
Servomotor 2	3	
Servomotor 3	5	
Servomotor 4	4	
Pack LEDs 1	10	
Pack LEDs 2	11	
Pack LEDs 3	13	
Pack LEDs 4	12	
Sensor PIR 1	22	**DIGITALES**
Sensor PIR 2	52	
Alimentación	5v	**POWER**
Tierra	GND	

Tabla 5: Conexión de los distintos componentes a Arduino

La elección de unos pines u otros, da igual cuál escoger siempre y cuando se respete el tipo de señal que utiliza cada componente. En este caso se han ido ocupando los distintos pines del Arduino, en función a la comodidad de conexión.

A continuación se podrá observar un ejemplo de conexiones conjunta con una muestra de cada componente en la *Ilustración 61*.

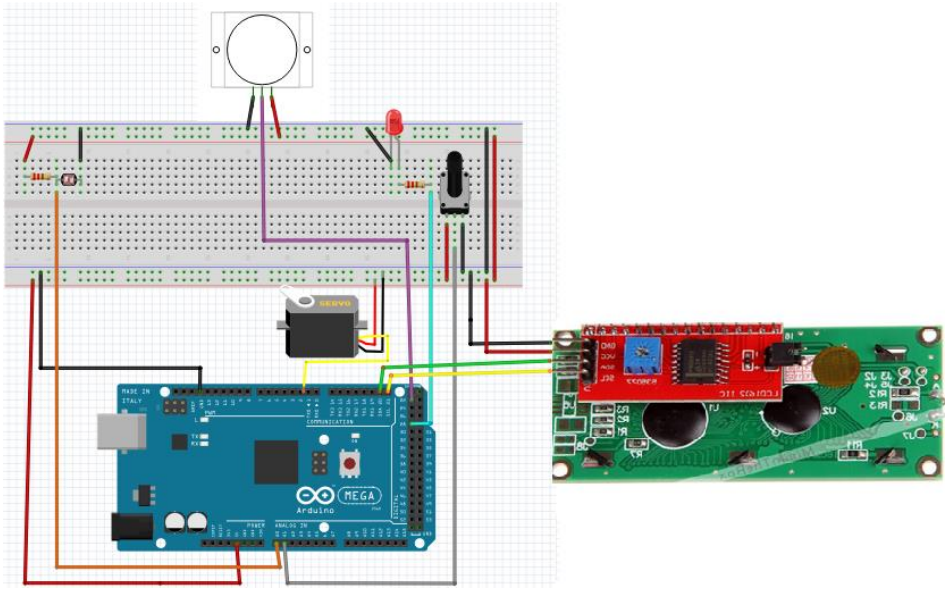


Ilustración 61: Conexión de los componentes en Arduino

Por último en la *Ilustración 62*, se puede observar la conexión final del entorno de pruebas.

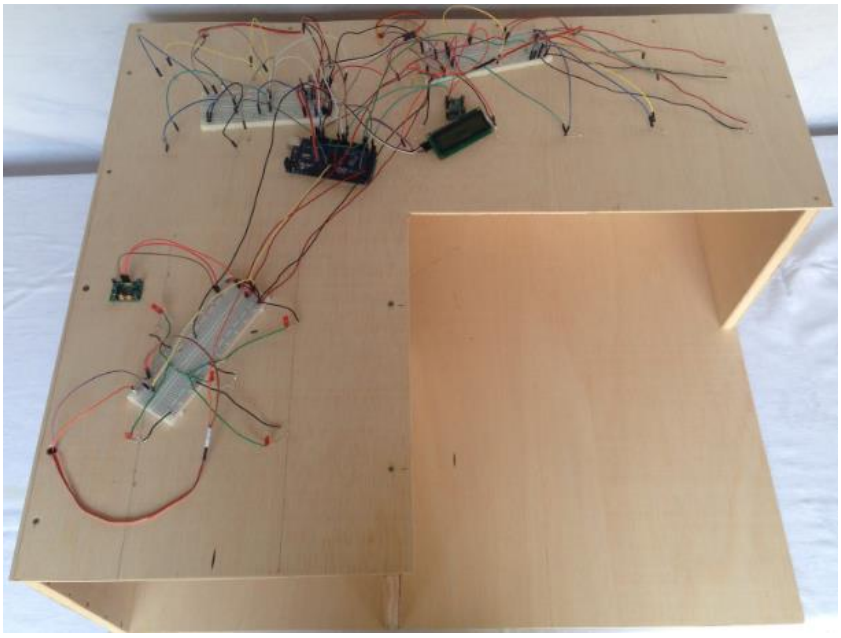


Ilustración 62: Conexión final del entorno de pruebas

4.5. Elaboración del Sistema difuso

Antes de comenzar con la explicación de la elaboración del sistema difuso, es importante mencionar que para el control gradual de las luces, no ha hecho falta de la elaboración de un sistema difuso para su control, ya que, con las características del sensor LDR, se consigue un correcto funcionamiento de las luces. Por lo que cuando se hable del sistema difuso solo se estará haciendo referencia al control de las persianas. Una vez se ha comentado esto, se comenzará a la explicación del sistema difuso.

Primeramente se deberá diseñar el sistema difuso que cumpla con las especificaciones para el que es diseñado. El diseño de un sistema difuso para este entorno de pruebas, tiene como principal objetivo el control del movimiento de cuatro persianas, en función de unos factores pautados.

Un sistema difuso necesita una o varias variables de entrada y una o varias salidas. Cada variable se define mediante un dominio de definición: conjuntos difusos que abarca el rango de valores que podría tomar esa variable.

En nuestro caso se va a disponer de *cuatro variables, tres de entrada y una de salida*, en las que cada dominio estará comprendido por diferentes conjuntos difusos. A cada conjunto difuso se le asignará una etiqueta lingüística para una mejor interpretación.

Para la elaboración del sistema difuso, he utilizado el programa Matlab, ya que es muy fácil de usar, y además te da la posibilidad de simular los resultados, y en todo momento tienes a mano las distintas gráficas de cada una de las entradas. El procedimiento de cómo utilizar dicho programa, esta mencionado en el *apartado 3.13.3 Toolbox Fuzzy de Matlab.*

El primer dominio de definición será a lo que se ha llamado ***Sensor LDR***, en el que vamos a reflejar la luz que hace en todo momento en el exterior. El rango de valores que vamos a utilizar, dependerá de donde vayamos a tener la maqueta, en nuestro caso he realizado un estudio previo en varios lugares y voy a poner una media de los valores máximos y mínimos obtenidos. El rango de valores que utilizaremos en principio será de 250 a 1000, que estará comprendido en 5 conjuntos difusos con las siguientes etiquetas lingüísticas: **muy claro, claro, nublado, oscuro** y **muy oscuro.**

El segundo dominio de definición es el ***Simulador de Temperatura***, con un rango de valores de 0 a 44, que será las temperaturas que vamos a simular, a través de un potenciómetro. Va a estar comprendido en 3 conjuntos difusos con las siguientes etiquetas lingüísticas: **frío, normal** y **calor.**

El tercer dominio de definición es el ***PIR***, con un rango de valores de -1 a 2, que estará comprendido en 2 conjuntos difusos con las siguientes etiquetas lingüísticas: **sí presencia** y **no presencia.**

Y por último **el cuarto y último dominio de definición**, que corresponde a la salida, se le ha llamado ***Persiana***, con un rango de definición de 0 a 8050, que estará comprendido en 6 conjuntos difusos con las siguientes etiquetas lingüísticas: **bajada, subir muy poco, subir poco, subir medio, subir mucho** y **subida total.**

El proceso que seguirá el sistema de lógica difusa será el que se mencionó en el apartado *3.13 Lógica difusa*, en la siguiente *Ilustración 63* veremos un esquema donde se marcarán las distintas etapas.

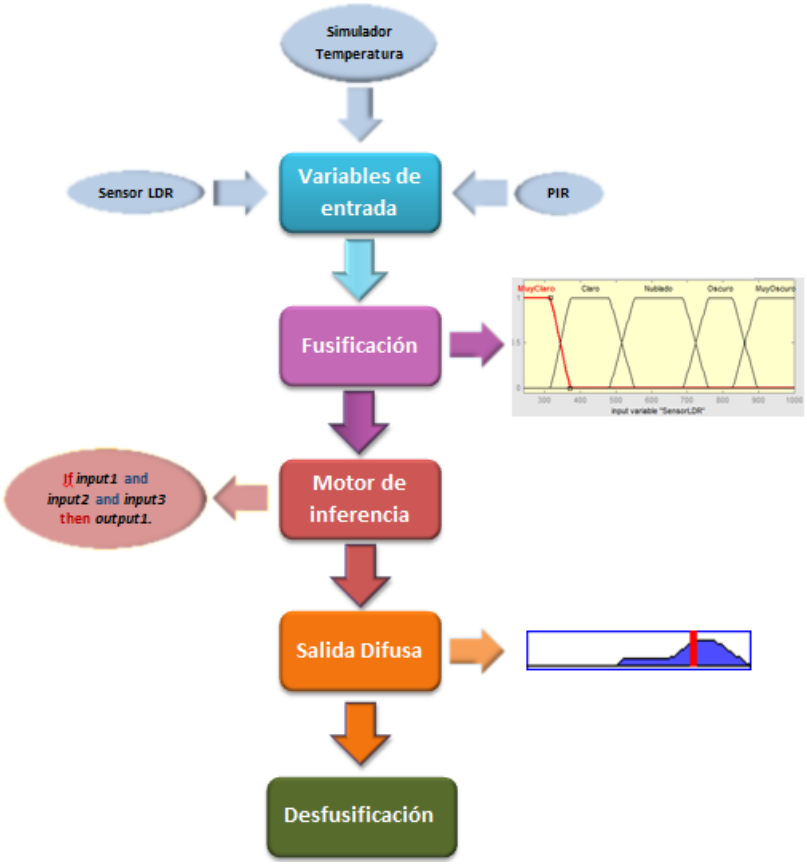


Ilustración 63: Proceso de nuestro sistema difuso

El primer paso de nuestro sistema de lógica difusa es convertir las señales de entrada (Sensor LDR, Simulador de temperatura y PIR) en un conjunto de variables difusas (***Fusificación).*** Se asignan valores (valores difusos) a través de un conjunto de funciones de pertenencia. El *fusificador* se utiliza para saber el nivel de pertenencia al que pertenece las señales de entrada.

El siguiente punto, en el ***Motor de interferencia***, que es el encargado de la activación de las reglas en función del nivel de pertenencia, determinando el valor de la salida del sistema.

El último paso es la ***Defusificación***, que es el encargado de pasar los valores difusos a valores reales, para que así nuestros distintos motores de las persianas, puedan moverse.

Una vez que se ha hecho nuestro prediseño de nuestro sistema difuso a través de la herramienta *Tollbox fuzzy* de *Matlab*, como se ha mencionado anteriormente. A continuación se va a explicar los valores de las distintas entradas y salidas del sistema.

La variable entrada 1 será la que nos genere nuestro sensor LDR, por lo que le hemos puesto de nombre ***Sensor LDR***, tal y como observamos en la siguiente *Ilustración 64.*

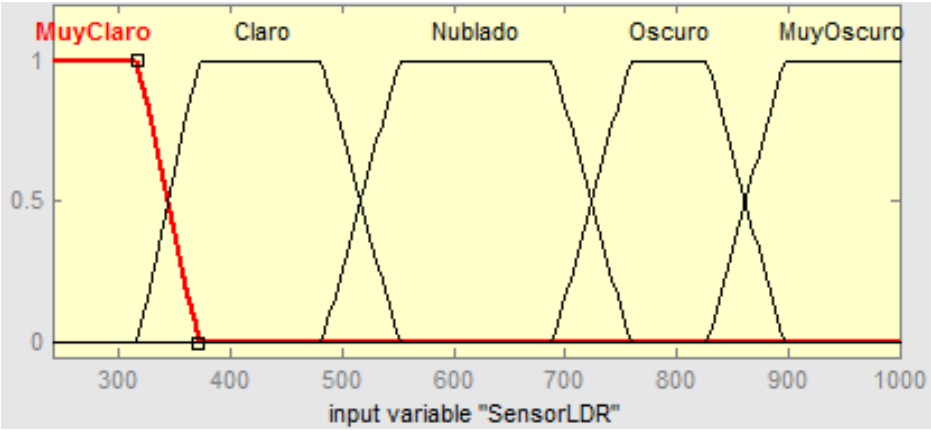


Ilustración 64: Variable de entrada *Sensor LDR*

El rango de valores asignado, no son los valores reales de luminosidad, sino los valores analógicos que nos genera el LDR. Estos valores se cambiarán en función de donde se vaya a instalar las persianas, se tomarán distintas medidas de dicha zona, para poder tener un valor mínimo y un valor máximos sobre los que poder trabajar.

Como se ha mencionado anteriormente y cómo podemos observar en la *Ilustración 65* la variable de entrada está dividida en 5 casos distintos:

- ***Muy Claro:*** es un trapecio y se ha definido en (240, 240, 315, 370) ya que el máximo de luz que he obtenido ha sido 240, pero se ha preferido dar un poco de margen por si se obtiene un resultado menor al obtenido hasta el momento. Este valor solo se obtiene con el cielo despejado, y cuando el sol incide directamente sobre el sensor. En nuestra maqueta se simulara mediante una linterna, una lámpara.
- ***Claro:*** también es un trapecio y se ha definido entre (315, 370, 485, 555), estos datos se obtendrán cuando el día también este aclaro pero no haya tanta iluminación como en el anterior caso, ya porque haya algo de nubes, esté amaneciendo o atardeciendo.
- ***Nublado:*** vuelve a ser un trapecio y se ha definido entre (485, 555, 690, 760), y como su nombre indica esto se obtendrá cuando se tenga un día nublado o lluvioso, en el que no haya mucha iluminación.
- ***Oscuro:*** es un trapecio y se ha definido entre (690, 760, 830, 895) esté caso será cuando este anocheciendo o haga niebla, es decir, en los casos en los que se empieza a necesitar encender las luces, por falta de visión.
- ***Muy Oscuro:*** es un trapecio y se ha definido en (830, 895, 1020, 1020), sólo se dará en los casos que se haya hecho de noche por completo.

Una vez que tenemos definida la variable de entrada 1, se pasa a definir la siguiente variable de entrada 2. Esta segunda variable se llama ***Simulador de Temperatura***, ya que como se va a realizar un entorno de pruebas, para poder demostrar distintos casos de temperatura, resulta más fácil realizarlo mediante un potenciómetro y según la variación de la resistencia, obtendremos una simulación de la temperatura.

El rango de valores pone de 0 a 44, que serían los grados sobre los que he decidido trabajar, como podemos observar en la *Ilustración 67*. Aunque mi potenciómetro tiene valores de 0 a 1024, gracias al comando de Arduino '*map (value, fromLow, fromHigh, toLow, toHigh)*'. Que re-mapea un número desde un rango hacia otro. Esto significa que, un valor '*value*' con respecto al rango '*fromLow-fromHight'* será mapeado al rango '*toLow-toHigh'*. En nuestro caso se ha conseguido de la siguiente manera: ***"temperatura = map (valorPot, 0, 1024, 0, 44)".***

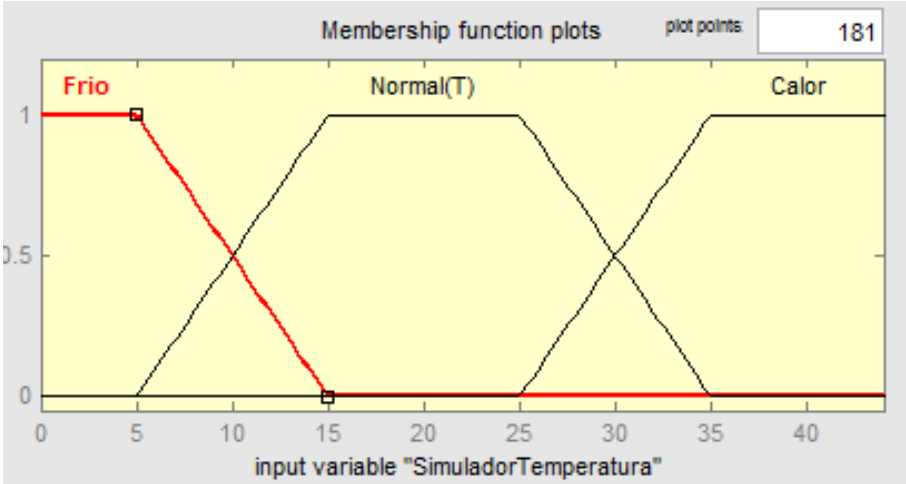


Ilustración 65: Variable de entrada *Simulador de Temperatura*

La variable de ***Simulador de temperatura*** está dividida por 3 casos, que se explican a continuación:

- ***Frio***, definido por un trapecio (0, 0, 5, 15); para la etiqueta lingüística de frio, se ha decidido definir frio al rango de temperaturas entro o y 15ºC, este y los demás rangos se podrían modificar en cualquier momento como ya se ha comentado anteriormente.
- ***Normal (T)***, definido por un trapecio (5, 15, 25, 35).
- ***Calor***, definido por un trapecio (25, 35, 44, 44).

La última variable de entrada es a la que hemos llamado ***PIR***, y va a ser el encargado de hacer funcionar al sistema únicamente cuando haya presencia. Debido a que un detector de presencia solo envía 0 (si no hay presencia) y 1 (si hay presencia), se ha decidido crear de la siguiente manera.

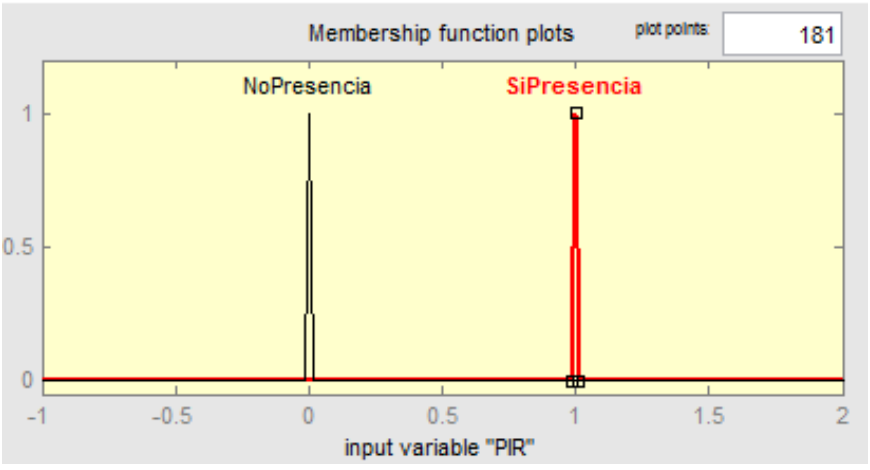


Ilustración 66: Variable de entrada *PIR*

Si se quisiera modificar el comportamiento de las persianas, sería tan fácil como cambiar los conjuntos difusos de las variables de entrada sin la necesidad de cambiar una sola línea de la programación.

Una vez definida todas las variables de entrada, comenzamos a definir la *variable de salida.* En este caso nuestra variable de salida se ha llamado ***Persiana*** (*Ilustración 67*)**.** Esta variable de salida va a ser el resultado de las reglas definidas del programa que se verán más adelante, que hará girar más o menos, en un sentido u en otro.

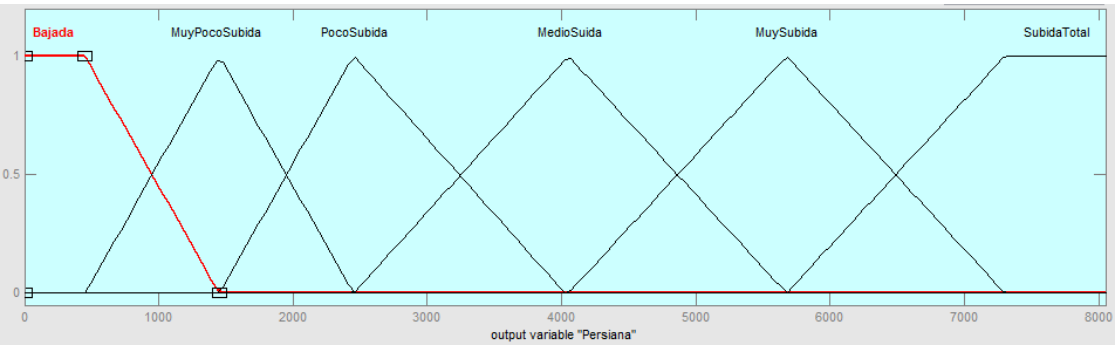


Ilustración 67: Variable de salida *Persiana*

La variable de salida ***Persiana*** está definida entre los valores 0 y 8050, que este valor será el tiempo (en milisegundos) de giro que se va a mantener a la persiana girando en un sentido u en otro, y también servirá para determinar en la posición que se encuentre en todo momento, ya que si con anterioridad había subido durante de 3000ms y sale una nueva orden de girar 2500ms solo gire la diferencia entre ambos valores, consiguiendo así un control total de la persiana. Se explicará más detalladamente cuando se realice más adelante el programa con el que se controlará todo el entorno de pruebas

La variable de salida estará cubierto por 6 conjuntos difusos, tal y como se pueden observar en la *Ilustración 68.*

- ***Bajada,*** está definido con un trapecio (-1070, -1070, 450, 1450), siempre que se produzca la esta salida, la persiana se va a situar entre al 0% y el 12%, dependerá del grado de pertenencia de las entradas y de las reglas que se han creado.
- ***Muy Poco Subida,*** está definido con un triángulo (450, 1450, 2450), en este caso la persiana, siempre que se produzca dicha salida, se situará dependiendo del grado de pertenencia entre el 12% y el 25%.
- ***Poco Subida,*** está definido con un triángulo (1450, 2450, 4025), dependiendo el grado de pertenencia que se dé, se situará entre el 25% y el 40%.

- ***Medio Subida,*** está definido con un triángulo (2450, 4025, 5675), se situará entre el 40% y el 60%.
- ***Muy Subida,*** está definido con un triángulo en (4025, 5675, 7300), se situará entre el 60% y 80%.
- ***Subida Total,*** está definido con un trapecio (5675, 7300, 12045), entre 80-100%.

Como se puede observar en las salidas *Subida Total* y *Bajada*, están definidos por trapecios en vez de triángulos, y es así porque, como para la defusificación se utiliza el *centro de masas*, para su cálculo, para conseguir una mayor pertenencia se hace empleo de un trapecio para conseguir una pertenencia total. Y el empleo del *centro de masas*, también ha provocado tener que coger un rango más amplio tanto en *Bajada* como *Subida Total*, para que cuando se de una pertenencia total, dé como resultado 0 y 8050, respectivamente

Una vez que se tienen todas las variables de entrada y las de salida, se puede comenzar a definir las distintas reglas del sistema, **estas reglas van a suponer el motor de nuestro sistema artificial.** Antes de empezar a definir las distintas reglas se va a explicar cuál es el comportamiento, que es lo que se buscara obtener de las persianas.

A partir de los distintos valores de entrada que nos van a proporcionar los distintos sensores (PIR, LDR y Temperatura), se usan las funciones de pertenencia a los conjuntos difusos, que hemos elaborado anteriormente, y se sustituye ese valor numérico por etiquetas lingüísticas. En segundo lugar, se mira la activación de las reglas. Una regla se activa cuando se satisfacen sus antecedentes de forma parcial o absoluta. El consecuente de las reglas activadas, determina el valor de salida del sistema (*Persiana).*

Se pretende que la persiana no tenga una respuesta prefijada, sino que dependa de una serie de variables (Sensor LDR, Simulador de temperatura y PIR). En función de los distintos rangos que se obtengan, las persianas harán una cosa u otra, no siempre siendo la misma, sino que para cada situación tendrá una respuesta distinta.

Una vez explicado el comportamiento que se quiere dar a las persianas, hay que definir ese comportamiento en reglas teniendo en cuenta todas las variables declaradas anteriormente, sin que se quede ningún caso sin definir, ya que habría situaciones en el que el sistema no sabría qué hacer.

1. If (SensorLDR is MuyClaro) and (SimuladorTemperatura is Frio) and (PIR is SiPresencia) then (Persiana is MuySubida) (1)
2. If (SensorLDR is MuyClaro) and (SimuladorTemperatura is Normal(T)) and (PIR is SiPresencia) then (Persiana is SubidaTotal) (1)
3. If (SensorLDR is MuyClaro) and (SimuladorTemperatura is Calor) and (PIR is SiPresencia) then (Persiana is MedioSuida) (1)
4. If (SensorLDR is Claro) and (SimuladorTemperatura is Frio) and (PIR is SiPresencia) then (Persiana is MuySubida) (1)
5. If (SensorLDR is Claro) and (SimuladorTemperatura is Normal(T)) and (PIR is SiPresencia) then (Persiana is MuySubida) (1)
6. If (SensorLDR is Claro) and (SimuladorTemperatura is Calor) and (PIR is SiPresencia) then (Persiana is MedioSuida) (1)
7. If (SensorLDR is Nublado) and (SimuladorTemperatura is Frio) and (PIR is SiPresencia) then (Persiana is PocoSubida) (1)
8. If (SensorLDR is Nublado) and (SimuladorTemperatura is Normal(T)) and (PIR is SiPresencia) then (Persiana is MedioSuida) (1)
9. If (SensorLDR is Nublado) and (SimuladorTemperatura is Calor) and (PIR is SiPresencia) then (Persiana is PocoSubida) (1)
10. If (SensorLDR is Oscuro) and (SimuladorTemperatura is Frio) and (PIR is SiPresencia) then (Persiana is MuyPocoSubida) (1)
11. If (SensorLDR is Oscuro) and (SimuladorTemperatura is Normal(T)) and (PIR is SiPresencia) then (Persiana is PocoSubida) (1)
12. If (SensorLDR is Oscuro) and (SimuladorTemperatura is Calor) and (PIR is SiPresencia) then (Persiana is MuyPocoSubida) (1)
13. If (SensorLDR is MuyOscuro) and (SimuladorTemperatura is Frio) and (PIR is SiPresencia) then (Persiana is Bajada) (1)
14. If (SensorLDR is MuyOscuro) and (SimuladorTemperatura is Normal(T)) and (PIR is SiPresencia) then (Persiana is Bajada) (1)
15. If (SensorLDR is MuyOscuro) and (SimuladorTemperatura is Calor) and (PIR is SiPresencia) then (Persiana is MuyPocoSubida) (1)

Ilustración 68: Reglas del comportamiento de las persianas

Aunque no aparece en la *Ilustración 68*, siempre que se dé el caso *'No Presencia'*, el Arduino mandará una señal a la persiana afectada, para que se esté quieta y no se mueva.

Para la modificación de los valores de las variables no hace falta cambiar todo el programa, solo basta con modificar la definición de las variables, con esto es suficiente y sin necesidad de modificar las reglas. Si se cambian los valores el sistema tendrá un comportamiento totalmente distinto aunque atienda a las mismas reglas definidas.

Una vez diseñado el sistema se hará una simulación mediante Matlab. Se cogerá de "*SensorLDR*" una entrada de 538, del "*SimuladorTemperatura*" una entrada de 24.1 y del "*PIR*" con un 1.01, es decir, con presencia. Como resultado se obtiene que *Persiana* es igual a 4150, lo que supone que se situará al 50,3%.

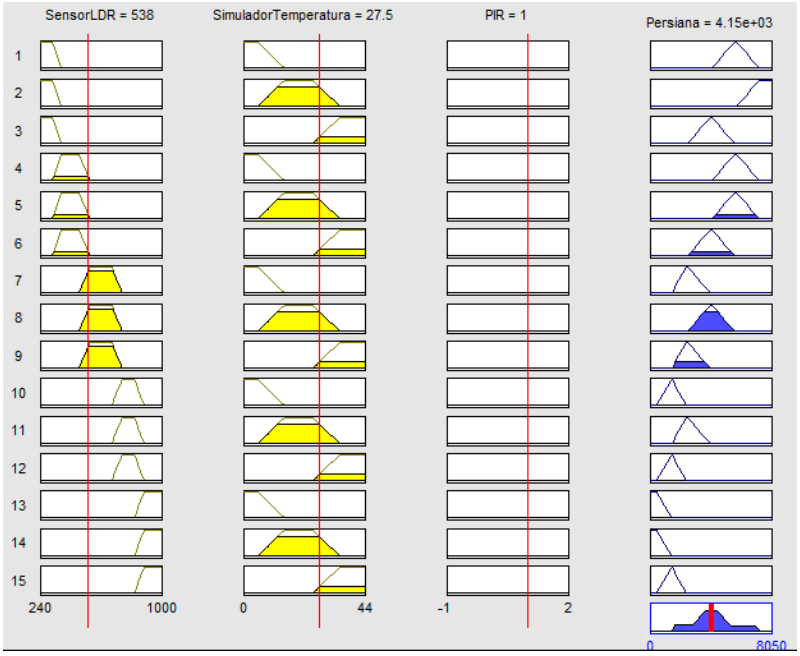


Ilustración 69: Ejemplo de funcionamiento

Como se puede observar, la salida ha sido una suma de las **reglas 5, 6, 8 y 9**, en donde la número 8, es la que tiene mayor grado de pertenencia, lo que hace que la salida sea lo más parecida a su salida pautada anteriormente, que en este caso es *Subir Media.*

El valor de la salida será el resultado de la ***defusificación***, ya que es el encargado de pasar el valor difuso a valores numéricos. El método de *defusificación* que se lleva a cabo en este sistema es por el *método del centroide*. El valor de incremento que se da es 4050.

El ***método del centroide*** se transforma la salida difusa en un número real el cual es la coordenada del centro de gravedad de tal conjunto difuso de salida.

$$y_d = \frac{\int_S y u_\gamma(y) dy}{\int_S u_\gamma(y) dy}$$

Donde u_γ es la función de pertenencia del conjunto de salida y_d. 'S' es el dominio o rango de integración. El método es costoso de hacer pero el más preciso, por lo tanto con un ordenador este método se puede haces como se ha hecho en este caso, o también mediante Arduino, que te saca también directamente el valor final.

4.6. Programación con Arduino.

Para poder controlar nuestra maqueta tendremos que usar el Software de Arduino, para ello vamos a tener que crear el programa, que habrá que hacerlo con el lenguaje de programación C/C++.

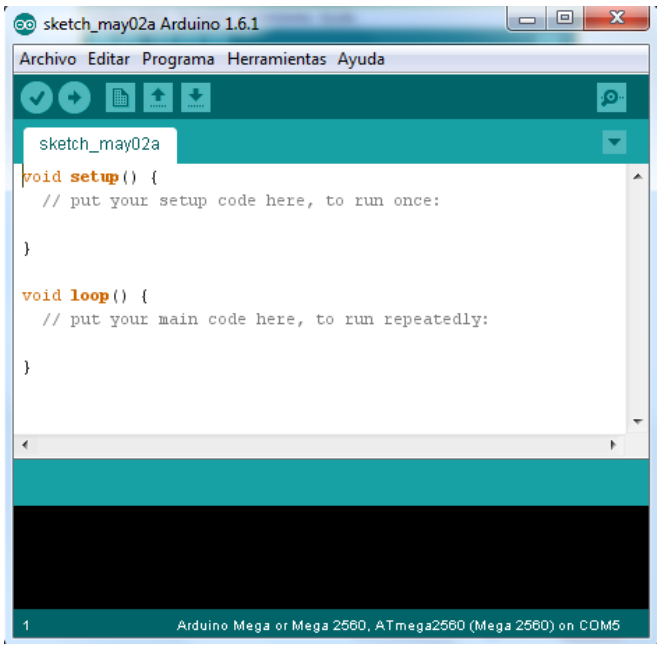


Ilustración 70: Programa de Arduino

Llegados a este punto, se tenían dos posibilidades, o bien, como ya teníamos terminado el sistema difuso de nuestro sistema en el programa de Matlab, hay unos comandos con los que se permite la comunicación entre Arduino y Matlab, pero esto supondría tener que tener enganchado en todo momento nuestro Arduino Mega al ordenador. Por lo que, aunque haya sido más laborioso, he decidido realizar todo el programa en su totalidad a través del programa Arduino, para así conseguir una mayor independencia, ya que si no queremos, no tenemos por qué tenerlo enchufado al ordenador, para su correcto funcionamiento.

Para poder realizar el programa anteriormente, se ha tenido que descargar las siguientes librerías, a través de la siguiente página: https://github.com/zerokol/eFLL

- FuzzyRule.h
- FuzzyComposition.h
- Fuzzy.h
- FuzzyRuleConsequent.h
- FuzzyOutput.h
- FuzzyInput.h
- FuzzyIO.h
- FuzzySet.h
- FuzzyRuleAntecedent.h

El programa que se va a desarrollar a continuación, va a servir para el control de 4 servomotores y 21 leds, a través de 4 sensores LDR, 2 sensor PIR, un potenciómetro (para simular el sensor de temperatura) y una pantalla LCD donde se reflejará en todo momento la temperatura que se está simulando y el porcentaje de apertura de las persianas.

La estructura de control de los servomotores y de las luces (leds), es el siguiente:

- Como se pudo ver en el *Apartado 4.3 Montaje del entorno de pruebas,* nuestro entorno de pruebas está dividido en dos partes, una con tres ventanas y otra con una única ventana, separadas por una pared. Para el control del área con tres ventanas (tres servomotores) y de 16 leds, se van a utilizar tres sensores de luminosidad (uno por cada ventana) y un sensor de presencia. Con la introducción de un sensor de luminosidad para cada servomotor, se va a conseguir un control independiente, para cada servomotor, pero se activarán con el mismo PIR.

- Y por otro lado se controlará de forma independiente, la zona en la que sólo dispone de una ventana y 5 leds. Para dicho control se un sensor de luminosidad y de presencia, distintos a los mencionados anteriormente.

Para controlar los servomotores y la pantalla LCD I2C, se van a tener que introducir las siguientes librerías:

- Servo.h
- LiquidCrystal_I2C.h
- LCD.h
- Wire.h

Y por último se ha añadido la librería de *EEPROM*, para así conseguir que no se pierdan los datos almacenados, cuando se reinicie el Arduino, ya que de no ser así, si se apaga el Arduino, en este caso la posición a la que estuvieran las persianas se perderían, y el sistema empezaría dando por supuesto que todas las persianas estaría bajadas totalmente, cuando a lo mejor algunas de ellas estaban parcialmente subidas. Gracias a que el microcontrolador trae una memoria no volátil (no se reinicia cuando se apague), llamada **EEPROM,** se conseguirá guardar los datos que sean útiles (última posición de cada una de las persianas), por lo que al reiniciar el Arduino las persianas conservarían su última posición. En el caso del Arduino ATMega2560, como se comentó en el *Apartado 3.11.1 Placas Arduino*, dispone de una capacidad de 4 KB, que es suficiente para guardar en todo momento la posición de las 4 ventanas.

A continuación aparecerá el programa que realizará todas las funciones mencionadas anteriormente, y con el que podremos controlar todo el entorno de pruebas al completo:

```
#include <FuzzyRule.h>
#include <FuzzyComposition.h>
#include <Fuzzy.h>
#include <FuzzyRuleConsequent.h>
#include <FuzzyOutput.h>
#include <FuzzyInput.h>
#include <FuzzyIO.h>
#include <FuzzySet.h>
#include <FuzzyRuleAntecedent.h>
#include <Servo.h>
#include <Wire.h>
#include <LCD.h>
#include <LiquidCrystal_I2C.h>
#include <I2CIO.h>
#include <EEPROM.h>
```

```
Fuzzy* fuzzy = new Fuzzy();

FuzzySet* MuyClaro= new FuzzySet(400, 450, 555, 595);
FuzzySet* Claro = new FuzzySet(555, 595, 675, 725);
FuzzySet* Nublado = new FuzzySet(675, 725, 825, 875);
FuzzySet* Oscuro = new FuzzySet(825, 875, 925, 975);
FuzzySet* MuyOscuro = new FuzzySet(925, 975, 1024, 1024);

FuzzySet* Frio = new FuzzySet(0, 0, 5, 15);
FuzzySet* Normal = new FuzzySet(5, 15, 25, 35);
FuzzySet* Calor = new FuzzySet(25, 35, 44, 44);

FuzzySet* SiPresencia = new FuzzySet(0.99, 1, 1, 1.01);
FuzzySet* NoPresencia = new FuzzySet(-0.01, 0, 0, 0.01);

FuzzySet* Bajada= new FuzzySet(-1070, -1070, 450, 1450);
FuzzySet* MuyPocoSubida = new FuzzySet(450, 1450, 1450, 2450);
FuzzySet* PocoSubida = new FuzzySet(1450, 2450, 2450, 4025);
FuzzySet* MedioSubida = new FuzzySet(2450, 4025, 4025, 5675);
FuzzySet* MuySubida = new FuzzySet(4025, 5675, 5675, 7300);
FuzzySet* SubidaTotal = new FuzzySet(5675, 7300, 12045, 12045);

//                    Addr, En, Rw, Rs, d4, d5, d6, d7, backlighpin, pol
arity
LiquidCrystal_I2C lcd( 0x27, 2,  1,  0,  4,  5,  6,  7,           3, POS
ITIVE );

int potenciometro = 15;
int valorPot = 0;
int temperatura = 0;

// HABITACIÓN GRANDE

Servo miServo1;
Servo miServo2;
Servo miServo3;
int ldr1 = A0;
int ldr2 = A2;
int ldr3 = A4;
int pir1 = 22;
int led1 = 8;
int led2 = 9;
int led3 = 10;

int presencia1 = 0;
int valorLdr1 = 0;
int valorLdr2 = 0;
int valorLdr3 = 0;
float salida1 = 0;
float salida2 = 0;
float salida3 = 0;
int posicion1 = 0;
int posicion2 = 0;
int posicion3 = 0;
int subida1 = 0;
int eeprom1 = 0; // se guardaran en la posicion 0 y 1
int cociente1 = 0; // resultado de la dividision que se guardará en la
posicion 0
int resto1 = 0; // el resto de la division y se guardará en la posicion 1
int subida2 = 0;
int eeprom2 = 2; // se guardaran en la posicion 2 y 3
```

```
int cociente2 = 0; // resultado de la dividision que se guardará en la
posicion 2
int resto2 = 0; // el resto de la division y se guardará en la posicion 3
int subida3 = 0;
int eeprom3 = 4; // se guardaran en la posicion 4 y 5
int cociente3 = 0; // resultado de la dividision que se guardará en la
posicion 4
int resto3 = 0; // el resto de la division y se guardará en la posicion 5

int quieto1 = 76;
int giroSubida1 = quieto1 + 9;
int giroBajada1 = quieto1 - 9;

int quieto2 = 83;
int giroSubida2 = quieto2 + 11;
int giroBajada2 = quieto2 - 9;

float quieto3 = 78;
int giroSubida3 = quieto3 + 11;
int giroBajada3 = quieto3 - 9;

// HABITACIÓN PEQUEÑA

Servo miServo4;
int ldr4 = A6;
int pir2 = 53;
int led4 = 11;

int valorLdr4 = 0;
int presencia2 = 0;
float salida4 = 0;
int posicion4 = 0;
int subida4 = 0;
int eeprom4 = 6; // se guardaran en la posicion 6 y 7
int cociente4 = 0; // resultado de la dividision que se guardará en la
posicion 6
int resto4 = 0; // el resto de la division y se guardará en la posicion 7
int quieto4 = 75;
int giroSubida4 = quieto4 + 10;
int giroBajada4 = quieto4 - 10;

void setup() {
  Serial.begin(9600);

  miServo1.attach(2,-360,360);
  pinMode(led1, OUTPUT);

  miServo2.attach(3,-360,360);
  pinMode(led2, OUTPUT);

  miServo3.attach(4,-360,360);
  pinMode(led3, OUTPUT);

  miServo4.attach(5,-360,360);
  pinMode(led4, OUTPUT);

  lcd.begin(16,2);
  lcd.backlight();
  lcd.setCursor(0,0);
  lcd.print("Temperatura =");
```

```
  // FuzzyInput

  FuzzyInput* SensorLDR = new FuzzyInput(1);
    SensorLDR->addFuzzySet(MuyClaro);
    SensorLDR->addFuzzySet(Claro);
    SensorLDR->addFuzzySet(Nublado);
    SensorLDR->addFuzzySet(Oscuro);
    SensorLDR->addFuzzySet(MuyOscuro);
  fuzzy->addFuzzyInput(SensorLDR);

    // FuzzyInput

  FuzzyInput* SimuladorTemperatura = new FuzzyInput(2);
    SimuladorTemperatura->addFuzzySet(Frio);
    SimuladorTemperatura->addFuzzySet(Normal);
    SimuladorTemperatura->addFuzzySet(Calor);
  fuzzy->addFuzzyInput(SimuladorTemperatura);

     // FuzzyInput

  FuzzyInput* PIR = new FuzzyInput(3);
    PIR->addFuzzySet(SiPresencia);
    PIR->addFuzzySet(NoPresencia);
  fuzzy->addFuzzyInput(PIR);

    // FuzzyOutput

  FuzzyOutput* Persiana = new FuzzyOutput(1);
    Persiana->addFuzzySet(Bajada);
    Persiana->addFuzzySet(MuyPocoSubida);
    Persiana->addFuzzySet(PocoSubida);
    Persiana->addFuzzySet(MedioSubida);
    Persiana->addFuzzySet(MuySubida);
    Persiana->addFuzzySet(SubidaTotal);
  fuzzy->addFuzzyOutput(Persiana);

  // Building FuzzyRule1

  FuzzyRuleAntecedent* SensorLDRMuyClaroAndSimuladorTemperaturaFrio = new
FuzzyRuleAntecedent();
  SensorLDRMuyClaroAndSimuladorTemperaturaFrio -
> joinWithAND(MuyClaro, Frio);
  FuzzyRuleAntecedent* PIRSiPresencia = new FuzzyRuleAntecedent();
  PIRSiPresencia->joinSingle(SiPresencia);
  FuzzyRuleAntecedent* ifSensorLDRMuyClaroAndSimuladorTemperaturaFrioAndP
IRSiPresencia = new FuzzyRuleAntecedent();
  ifSensorLDRMuyClaroAndSimuladorTemperaturaFrioAndPIRSiPresencia-
>joinWithAND(SensorLDRMuyClaroAndSimuladorTemperaturaFrio, PIRSiPresencia
);

  FuzzyRuleConsequent* thenPersianaMuySubida = new FuzzyRuleConsequent();
  thenPersianaMuySubida->addOutput(MuySubida);

  FuzzyRule* fuzzyRule1 = new FuzzyRule(1,
ifSensorLDRMuyClaroAndSimuladorTemperaturaFrioAndPIRSiPresencia,
thenPersianaMuySubida);
  fuzzy->addFuzzyRule(fuzzyRule1);

  // Building FuzzyRule2
```

```
 FuzzyRuleAntecedent* SensorLDRMuyClaroAndSimuladorTemperaturaNormal = ne
w FuzzyRuleAntecedent();
  SensorLDRMuyClaroAndSimuladorTemperaturaNormal -
> joinWithAND(MuyClaro, Normal);
  FuzzyRuleAntecedent* ifSensorLDRMuyClaroAndSimuladorTemperaturaNormalAn
dPIRSiPresencia = new FuzzyRuleAntecedent();
  ifSensorLDRMuyClaroAndSimuladorTemperaturaNormalAndPIRSiPresencia-
>joinWithAND(SensorLDRMuyClaroAndSimuladorTemperaturaNormal, PIRSiPresenc
ia);

  FuzzyRuleConsequent* thenPersianaSubidaTotal = new
FuzzyRuleConsequent();
  thenPersianaSubidaTotal->addOutput(SubidaTotal);

  FuzzyRule* fuzzyRule2 = new FuzzyRule(2,
ifSensorLDRMuyClaroAndSimuladorTemperaturaNormalAndPIRSiPresencia,
thenPersianaSubidaTotal);
  fuzzy->addFuzzyRule(fuzzyRule2);

    // Building FuzzyRule3

  FuzzyRuleAntecedent* SensorLDRMuyClaroAndSimuladorTemperaturaCalor = ne
w FuzzyRuleAntecedent();
  SensorLDRMuyClaroAndSimuladorTemperaturaCalor -
> joinWithAND(MuyClaro, Calor);
  FuzzyRuleAntecedent* ifSensorLDRMuyClaroAndSimuladorTemperaturaCalorAnd
PIRSiPresencia = new FuzzyRuleAntecedent();
  ifSensorLDRMuyClaroAndSimuladorTemperaturaCalorAndPIRSiPresencia-
>joinWithAND(SensorLDRMuyClaroAndSimuladorTemperaturaCalor, PIRSiPresenci
a);

  FuzzyRuleConsequent* thenPersianaMedioSubida = new
FuzzyRuleConsequent();
  thenPersianaMedioSubida->addOutput(MedioSubida);

  FuzzyRule* fuzzyRule3 = new FuzzyRule(3,
ifSensorLDRMuyClaroAndSimuladorTemperaturaNormalAndPIRSiPresencia,
thenPersianaMedioSubida);
  fuzzy->addFuzzyRule(fuzzyRule3);

  // Building FuzzyRule4

  FuzzyRuleAntecedent* SensorLDRClaroAndSimuladorTemperaturaFrio = new
FuzzyRuleAntecedent();
  SensorLDRClaroAndSimuladorTemperaturaFrio -> joinWithAND(Claro, Frio);
  FuzzyRuleAntecedent* ifSensorLDRClaroAndSimuladorTemperaturaFrioAndPIRS
iPresencia = new FuzzyRuleAntecedent();
  ifSensorLDRClaroAndSimuladorTemperaturaFrioAndPIRSiPresencia-
>joinWithAND(SensorLDRClaroAndSimuladorTemperaturaFrio, PIRSiPresencia);

  FuzzyRule* fuzzyRule4 = new FuzzyRule(4,
ifSensorLDRClaroAndSimuladorTemperaturaFrioAndPIRSiPresencia,
thenPersianaMuySubida);
  fuzzy->addFuzzyRule(fuzzyRule4);

   // Building FuzzyRule5

  FuzzyRuleAntecedent* SensorLDRClaroAndSimuladorTemperaturaNormal = new
FuzzyRuleAntecedent();
  SensorLDRClaroAndSimuladorTemperaturaNormal -
> joinWithAND(Claro, Normal);
```

```
  FuzzyRuleAntecedent* ifSensorLDRClaroAndSimuladorTemperaturaNormalAndPI
RSiPresencia = new FuzzyRuleAntecedent();
  ifSensorLDRClaroAndSimuladorTemperaturaNormalAndPIRSiPresencia-
>joinWithAND(SensorLDRClaroAndSimuladorTemperaturaNormal, PIRSiPresencia)
;

  FuzzyRule* fuzzyRule5 = new FuzzyRule(5,
ifSensorLDRClaroAndSimuladorTemperaturaNormalAndPIRSiPresencia,
thenPersianaMuySubida);
  fuzzy->addFuzzyRule(fuzzyRule5);

  // Building FuzzyRule6

  FuzzyRuleAntecedent* SensorLDRClaroAndSimuladorTemperaturaCalor = new
FuzzyRuleAntecedent();
  SensorLDRClaroAndSimuladorTemperaturaCalor -
> joinWithAND(Claro, Calor);
  FuzzyRuleAntecedent* ifSensorLDRClaroAndSimuladorTemperaturaCalorAndPIR
SiPresencia = new FuzzyRuleAntecedent();
  ifSensorLDRClaroAndSimuladorTemperaturaCalorAndPIRSiPresencia-
>joinWithAND(SensorLDRClaroAndSimuladorTemperaturaCalor, PIRSiPresencia);

  FuzzyRule* fuzzyRule6 = new FuzzyRule(6,
ifSensorLDRClaroAndSimuladorTemperaturaCalorAndPIRSiPresencia,
thenPersianaMedioSubida);
  fuzzy->addFuzzyRule(fuzzyRule6);

  // Building FuzzyRule7

  FuzzyRuleAntecedent* SensorLDRCNubladoAndSimuladorTemperaturaFrio = new
FuzzyRuleAntecedent();
  SensorLDRCNubladoAndSimuladorTemperaturaFrio -
> joinWithAND(Nublado, Frio);
  FuzzyRuleAntecedent* ifSensorLDRCNubladoAndSimuladorTemperaturaFrioAndP
IRSiPresencia = new FuzzyRuleAntecedent();
  ifSensorLDRCNubladoAndSimuladorTemperaturaFrioAndPIRSiPresencia-
>joinWithAND(SensorLDRCNubladoAndSimuladorTemperaturaFrio, PIRSiPresencia
);

  FuzzyRuleConsequent* thenPersianaPocoSubida = new
FuzzyRuleConsequent();
  thenPersianaPocoSubida->addOutput(PocoSubida);

  FuzzyRule* fuzzyRule7 = new FuzzyRule(7,
ifSensorLDRCNubladoAndSimuladorTemperaturaFrioAndPIRSiPresencia,
thenPersianaPocoSubida);
  fuzzy->addFuzzyRule(fuzzyRule7);

  // Building FuzzyRule8

  FuzzyRuleAntecedent* SensorLDRCNubladoAndSimuladorTemperaturaNormal = n
ew FuzzyRuleAntecedent();
  SensorLDRCNubladoAndSimuladorTemperaturaNormal -
> joinWithAND(Nublado, Normal);
  FuzzyRuleAntecedent* ifSensorLDRCNubladoAndSimuladorTemperaturaNormalAn
dPIRSiPresencia = new FuzzyRuleAntecedent();
  ifSensorLDRCNubladoAndSimuladorTemperaturaNormalAndPIRSiPresencia-
>joinWithAND(SensorLDRCNubladoAndSimuladorTemperaturaNormal, PIRSiPresenc
ia);
```

```
  FuzzyRule* fuzzyRule8 = new FuzzyRule(8,
ifSensorLDRCNubladoAndSimuladorTemperaturaNormalAndPIRSiPresencia,
thenPersianaMedioSubida);
  fuzzy->addFuzzyRule(fuzzyRule8);

  // Building FuzzyRule9

  FuzzyRuleAntecedent* SensorLDRCNubladoAndSimuladorTemperaturaCalor = ne
w FuzzyRuleAntecedent();
  SensorLDRCNubladoAndSimuladorTemperaturaCalor -
> joinWithAND(Nublado, Calor);
  FuzzyRuleAntecedent* ifSensorLDRCNubladoAndSimuladorTemperaturaCalorAnd
PIRSiPresencia = new FuzzyRuleAntecedent();
  ifSensorLDRCNubladoAndSimuladorTemperaturaCalorAndPIRSiPresencia-
>joinWithAND(SensorLDRCNubladoAndSimuladorTemperaturaCalor, PIRSiPresenci
a);

  FuzzyRule* fuzzyRule9 = new FuzzyRule(9,
ifSensorLDRCNubladoAndSimuladorTemperaturaCalorAndPIRSiPresencia,
thenPersianaPocoSubida);
  fuzzy->addFuzzyRule(fuzzyRule9);

  // Building FuzzyRule10

  FuzzyRuleAntecedent* SensorLDRCOscuroAndSimuladorTemperaturaFrio = new
FuzzyRuleAntecedent();
  SensorLDRCOscuroAndSimuladorTemperaturaFrio -
> joinWithAND(Oscuro, Frio);
  FuzzyRuleAntecedent* ifSensorLDRCOscuroAndSimuladorTemperaturaFrioAndPI
RSiPresencia = new FuzzyRuleAntecedent();
  ifSensorLDRCOscuroAndSimuladorTemperaturaFrioAndPIRSiPresencia-
>joinWithAND(SensorLDRCOscuroAndSimuladorTemperaturaFrio, PIRSiPresencia)
;

  FuzzyRuleConsequent* thenPersianaMuyPocoSubida = new
FuzzyRuleConsequent();
  thenPersianaMuyPocoSubida->addOutput(MuyPocoSubida);

  FuzzyRule* fuzzyRule10 = new FuzzyRule(10,
ifSensorLDRCOscuroAndSimuladorTemperaturaFrioAndPIRSiPresencia,
thenPersianaMuyPocoSubida);
  fuzzy->addFuzzyRule(fuzzyRule10);

  // Building FuzzyRule11

  FuzzyRuleAntecedent* SensorLDRCOscuroAndSimuladorTemperaturaNormal = ne
w FuzzyRuleAntecedent();
  SensorLDRCOscuroAndSimuladorTemperaturaNormal -
> joinWithAND(Oscuro, Normal);
  FuzzyRuleAntecedent* ifSensorLDRCOscuroAndSimuladorTemperaturaNormalAnd
PIRSiPresencia = new FuzzyRuleAntecedent();
  ifSensorLDRCOscuroAndSimuladorTemperaturaNormalAndPIRSiPresencia-
>joinWithAND(SensorLDRCOscuroAndSimuladorTemperaturaNormal, PIRSiPresenci
a);

  FuzzyRule* fuzzyRule11 = new FuzzyRule(11,
ifSensorLDRCOscuroAndSimuladorTemperaturaNormalAndPIRSiPresencia,
thenPersianaPocoSubida);
  fuzzy->addFuzzyRule(fuzzyRule11);
```

```
  // Building FuzzyRule12

  FuzzyRuleAntecedent* SensorLDROscuroAndSimuladorTemperaturaCalor = new
FuzzyRuleAntecedent();
  SensorLDROscuroAndSimuladorTemperaturaCalor -
> joinWithAND(Oscuro, Calor);
  FuzzyRuleAntecedent* ifSensorLDROscuroAndSimuladorTemperaturaCalorAndPI
RSiPresencia = new FuzzyRuleAntecedent();
  ifSensorLDROscuroAndSimuladorTemperaturaCalorAndPIRSiPresencia-
>joinWithAND(SensorLDROscuroAndSimuladorTemperaturaCalor, PIRSiPresencia)
;

  FuzzyRuleConsequent* thenPersianaBajada = new FuzzyRuleConsequent();
  thenPersianaBajada->addOutput(Bajada);

  FuzzyRule* fuzzyRule12 = new FuzzyRule(12,
ifSensorLDROscuroAndSimuladorTemperaturaCalorAndPIRSiPresencia,
thenPersianaBajada);
  fuzzy->addFuzzyRule(fuzzyRule12);

  // Building FuzzyRule13

  FuzzyRuleAntecedent* SensorLDRCMuyOscuroAndSimuladorTemperaturaFrio = n
ew FuzzyRuleAntecedent();
  SensorLDRCMuyOscuroAndSimuladorTemperaturaFrio -
> joinWithAND(MuyOscuro, Frio);
  FuzzyRuleAntecedent* ifSensorLDRCMuyOscuroAndSimuladorTemperaturaFrioAn
dPIRSiPresencia = new FuzzyRuleAntecedent();
  ifSensorLDRCMuyOscuroAndSimuladorTemperaturaFrioAndPIRSiPresencia-
>joinWithAND(SensorLDRCMuyOscuroAndSimuladorTemperaturaFrio, PIRSiPresenc
ia);

  FuzzyRule* fuzzyRule13 = new FuzzyRule(13,
ifSensorLDRCMuyOscuroAndSimuladorTemperaturaFrioAndPIRSiPresencia,
thenPersianaBajada);
  fuzzy->addFuzzyRule(fuzzyRule13);

  // Building FuzzyRule14

  FuzzyRuleAntecedent* SensorLDRCMuyOscuroAndSimuladorTemperaturaNormal =
 new FuzzyRuleAntecedent();
  SensorLDRCMuyOscuroAndSimuladorTemperaturaNormal -
> joinWithAND(MuyOscuro, Normal);
  FuzzyRuleAntecedent* ifSensorLDRCMuyOscuroAndSimuladorTemperaturaNormal
AndPIRSiPresencia = new FuzzyRuleAntecedent();
  ifSensorLDRCMuyOscuroAndSimuladorTemperaturaNormalAndPIRSiPresencia-
>joinWithAND(SensorLDRCMuyOscuroAndSimuladorTemperaturaNormal, PIRSiPrese
ncia);

  FuzzyRule* fuzzyRule14 = new FuzzyRule(14,
ifSensorLDRCMuyOscuroAndSimuladorTemperaturaNormalAndPIRSiPresencia,
thenPersianaBajada);
  fuzzy->addFuzzyRule(fuzzyRule14);

  // Building FuzzyRule15

  FuzzyRuleAntecedent* SensorLDRCMuyOscuroAndSimuladorTemperaturaCalor =
new FuzzyRuleAntecedent();
  SensorLDRCMuyOscuroAndSimuladorTemperaturaCalor -
> joinWithAND(MuyOscuro, Calor);
```

```
  FuzzyRuleAntecedent* ifSensorLDRCMuyOscuroAndSimuladorTemperaturaCalorA
ndPIRSiPresencia = new FuzzyRuleAntecedent();
  ifSensorLDRCMuyOscuroAndSimuladorTemperaturaCalorAndPIRSiPresencia-
>joinWithAND(SensorLDRCMuyOscuroAndSimuladorTemperaturaCalor, PIRSiPresen
cia);

  FuzzyRule* fuzzyRule15 = new FuzzyRule(15,
ifSensorLDRCMuyOscuroAndSimuladorTemperaturaCalorAndPIRSiPresencia,
thenPersianaMuyPocoSubida);
  fuzzy->addFuzzyRule(fuzzyRule15);
}

// FUNCION PARA COMPROBAR LOS SENSORES LDR

int comprobacion_luz(int ldr){
  int valorL = analogRead(ldr);
  return (valorL);
}

// FUNCION PARA COMPROBAR LOS SENSORES PIR

int comprobacion_presencia(int pir){
  int valorP = digitalRead(pir);
  return (valorP);
}

// FUNCION PARA OBTENER LAS SALIDAS DIFUSAS

float salida_fuzzy (int valorLDR, int Temp, int valorPIR){
  fuzzy->setInput(1, valorLDR);
  fuzzy->setInput(2, Temp);
  fuzzy->setInput(3, valorPIR);
  fuzzy->fuzzify();
  float output = fuzzy->defuzzify(1);
  return (output);
}

// FUNCIÓN PARA SUBIR MOTOR

void subir_motor (int valorpir, float salida, float posicion, Servo
miServo, int giroSubida, int quietos){
  if (valorpir == 1 && salida > posicion){
    miServo.write(giroSubida);
    delay(salida - posicion);
    miServo.write(quietos);
  }
}

// FUNCIÓN PARA BAJAR MOTOR

void bajar_motor (int valorpir, float salida, float posicion, Servo
miServo, int giroBajada, int quietob){
  if (valorpir == 1 && salida < posicion){
    miServo.write(giroBajada);
    delay(posicion - salida);
    miServo.write(quietob);
  }
}
```

```
// FUNCIÓN PARA PARAR EL MOTOR

void stop_motor (int valorpir, float salida, float posicion, Servo
miServo, int quieto){
  if (valorpir == 0 || salida == posicion){
    miServo.write(quieto);
  }
}

// FUNCION PARA EL ENCENCIDO DE LAS LUCES

void encendido (int led, int valorldr, int presencia){
  int luz = constrain (valorldr, 350, 1050);
  luz = map (luz, 350, 1050, 0, 255);
  if (presencia == 1){
    analogWrite(led,luz);
  }else{
    digitalWrite(led,LOW);}
}

// FUNCION PARA GUARDAR LOS DISTINTOS VALORES EN LA MEMORIA EEPROM

void Guardar_dato_EEPROM(int valor, int cociente, int resto, int eeprom){
  cociente = valor/255;
  EEPROM.write(eeprom,cociente);
  resto = valor%255;
  EEPROM.write(eeprom+1, resto);
}

// FUNCION PARA EXTRAER LOS DATOS DE LA MEMORIA EEPROM

int Sacar_dato_EEPROM(int x){
  int C = EEPROM.read(x); // extrae el dato de la posidion x
  int R = EEPROM.read(x +1); // extrae el dato de la posicion x+1
  int D = 255* C + R;
  return(D);
}

void loop() {

// COMTROL DE LA HABITACIÓN GRANDE

presencia1 = comprobacion_presencia(pir1);

  // ****** luces 1 *******

  encendido(led1, valorLdr1, presencia1);

 // Control del motor 1 y sus correspondientes luces (led1)

valorLdr1 = comprobacion_luz(ldr1);
subida1 = Sacar_dato_EEPROM(eeprom1); //Extraemos el dato guardado en la
posición 1 de la memoria EEPROM

  // ****** fuzzy *******

  if (presencia1 == 1){
    salida1 = salida_fuzzy(valorLdr1, temperatura, presencia1);
  }
```

```
  // ****** MOTOT 1 *****

      // --- Subir ---
      subir_motor(presencia1, salida1, subida1, miServo1, giroSubida1, qu
ieto1);
      // --- Bajar ---
      bajar_motor(presencia1, salida1, subida1, miServo1, giroBajada1, qu
ieto1);
      // --- Quieto ---
      stop_motor(presencia1, salida1, subida1, miServo1, quieto1);

  if (presencia1 == 1){
    subida1 = salida1;
    Guardar_dato_EEPROM(subida1, cociente1, resto1, eeprom1);
  }
  posicion1 = map (subida1, 0, 8050, 0, 100);

// Control del motor 2 y sus correspondientes luces (led2)

valorLdr2 = comprobacion_luz(ldr2);
subida2 = Sacar_dato_EEPROM(eeprom2); //Extraemos el dato guardado en la
posición 2 de la memoria EEPROM

  // ****** luces 2 *******

  encendido(led2, valorLdr2, presencia1);

  // ****** fuzzy *******

  if (presencia1 == 1){
    salida2 = salida_fuzzy(valorLdr2, temperatura, presencia1);
  }

  // ****** MOTOT 2 *****

      // --- Subir ---
      subir_motor(presencia1, salida2, subida2, miServo2, giroSubida2, qu
ieto2);
      // --- Bajar ---
      bajar_motor(presencia1, salida2, subida2, miServo2, giroBajada2, qu
ieto2);
      // --- Quieto ---
      stop_motor(presencia1, salida2, subida2, miServo2, quieto2);

  if (presencia1 == 1){
    subida2 = salida2;
    Guardar_dato_EEPROM(subida2, cociente2, resto2, eeprom2);
  }
  posicion2 = map (subida2, 0, 8050, 0, 100);

// Control del motor 3 y sus correspondientes luces (led3)

valorLdr3 = comprobacion_luz(ldr3);
subida3 = Sacar_dato_EEPROM(eeprom3); //Extraemos el dato guardado en la
posición 3 de la memoria EEPROM

  // ****** luces 3 *******

  encendido(led3, valorLdr3, presencia1);
```

```
  // ****** fuzzy *******

  if (presencia1 == 1){
    salida3 = salida_fuzzy(valorLdr3, temperatura, presencia1);
  }

  // ****** MOTOT 3 *****

      // --- Subir ---
      subir_motor(presencia1, salida3, subida3, miServo3, giroSubida3, qu
ieto3);
      // --- Bajar ---
      bajar_motor(presencia1, salida3, subida3, miServo3, giroBajada3, qu
ieto3);
      // --- Quieto ---
      stop_motor(presencia1, salida3, subida3, miServo3, quieto3);
  if (presencia1 == 1){
    subida3 = salida3;
    Guardar_dato_EEPROM(subida3, cociente3, resto3, eeprom3);
  }
  posicion3 = map (subida3, 0, 8050, 0, 100);

// CONTROL DE LA HABITACIÓN PEQUEÑA

presencia2 = comprobacion_presencia(pir2);

// Control del motor 4 y sus correspondientes luces (led4)

valorLdr4 = comprobacion_luz(ldr4);
subida4 = Sacar_dato_EEPROM(eeprom4); //Extraemos el dato guardado en la
posición 4 de la memoria EEPROM

    // ****** luces 4 *******

  encendido(led4, valorLdr4, presencia2);

    // ****** fuzzy *******

  if (presencia2 == 1){
    salida4 = salida_fuzzy(valorLdr4, temperatura, presencia2);
  }

  // ****** MOTOT 4 *****

      // --- Subir ---
      subir_motor(presencia2, salida4, subida4, miServo4, giroSubida4, qu
ieto4);
      // --- Bajar ---
      bajar_motor(presencia2, salida4, subida4, miServo4, giroBajada4, qu
ieto4);
      // --- Quieto ---
      stop_motor(presencia2, salida4, subida4, miServo4, quieto4);

  if (presencia2 == 1){
    subida4 = salida4;
    Guardar_dato_EEPROM(subida4, cociente4, resto4, eeprom4);
  }
  posicion4 = map (subida4, 0, 8050, 0, 100);
```

```
// IMPRESIÓN DE DATOS EN LA PANTALLA LCD

  //Impresión de la simulación de la temperatura
  valorPot = analogRead (potenciometro);
  temperatura = map (valorPot, 0, 1023, 0, 44);

  if (temperatura<10){
    lcd.setCursor(14,0);
    lcd.print("0");
    lcd.setCursor(15,0);
    lcd.print(temperatura);
  }
  else if (temperatura>=10){
    lcd.setCursor(14,0);
    lcd.print(temperatura);
  }

  // Impresion de las posiciones de los distintos servomotores

      // Posicion servomotor 1
  lcd.setCursor(0,1);
  lcd.print(posicion1);
  if (posicion1<10){
    lcd.setCursor(1,1);
    lcd.print("%");
    lcd.setCursor(2,1);
    lcd.print("  ");
  }
  else if (posicion1>10 && posicion1<100){
    lcd.setCursor(2,1);
    lcd.print("%");
    lcd.setCursor(3,1);
    lcd.print(" ");
  }
  else if (posicion1==100){
    lcd.setCursor(3,1);
    lcd.print("%");
  }

  // Posicion servomotor 2
  lcd.setCursor(4,1);
  lcd.print(posicion2);
  if (posicion2<10){
    lcd.setCursor(5,1);
    lcd.print("%");
    lcd.setCursor(6,1);
    lcd.print("  ");
  }
  else if (posicion2>10 && posicion2<100){
    lcd.setCursor(6,1);
    lcd.print("%");
    lcd.setCursor(7,1);
    lcd.print(" ");
  }
  else if (posicion2==100){
    lcd.setCursor(7,1);
    lcd.print("%");
  }
  // Posicion servomotor 3
  lcd.setCursor(8,1);
  lcd.print(posicion3);
```

```
  if (posicion3<10){
    lcd.setCursor(9,1);
    lcd.print("%");
    lcd.setCursor(10,1);
    lcd.print("  ");
  }
  else if (posicion3>10 && posicion3<100){
    lcd.setCursor(10,1);
    lcd.print("%");
    lcd.setCursor(11,1);
    lcd.print(" ");
  }
  else if (posicion3==100){
    lcd.setCursor(11,1);
    lcd.print("%");
  }

  // Posición servomotor 4
  lcd.setCursor(12,1);
  lcd.print(posicion4);
  if (posicion4<10){
    lcd.setCursor(13,1);
    lcd.print("%");
    lcd.setCursor(14,1);
    lcd.print("  ");
  }
  else if (posicion4>10 && posicion4<100){
    lcd.setCursor(14,1);
    lcd.print("%");
    lcd.setCursor(15,1);
    lcd.print(" ");
  }
  else if (posicion4==100){
    lcd.setCursor(15,1);
    lcd.print("%");
  }
}
```

Es importante mencionar, que para que funcione correctamente, la primera vez que se ponga en marcha el entorno de pruebas, se va a suprimir donde pone en los 4 servomotores "`subida1 = EEPROM.read(1, subida1);`", ya que si no se hace esto, aparecerá un error diciendo que no hay ningún dato almacenado en la memoria EEPROM. Una vez ejecutado el programa (se habrán ido guardando la última posición de cada motor), podremos volver a cargar el programa con "`subida1 = EEPROM.read(1, subida1);`" de los 4 servomotores, ya que entonces si habrán datos almacenados en la memoria EEPROM.

Una vez que se ha realizado el programa, se puede comenzar a realizar distintas pruebas para asegurar su correcto funcionamiento, y de haber algún fallo poder modificarlo antes de llevarse a cabo.

4.7. Suposición sobre un Entorno Real.

En este apartado se va a explicar que sería necesario para poder llevarlo a la práctica sobre un entorno real. Para ello se va a hacer una suposición para lograr el control de única ventana y la iluminación (una bombilla), empleando el microcontrolador Arduino. Por último se dará un presupuesto del coste total que supondría llevarlo a cabo.

El programa utilizado en el Entorno de Pruebas podría llegar a servir casi en su totalidad, pero habría que realizar algunas modificaciones, ya que por ejemplo se dejaría de controlar un micro servomotor, y se pasaría a controlar un motor de corriente continua. Luego también habría que añadir el control de un sensor de temperatura, ya que en el Entorno de Pruebas, se hacía una simulación de ello a través de un potenciómetro, tal y como se ha comentado con anterioridad.

Antes de nada, es importante explicar que es un ***relé*** (*Ilustración 71*) y como funciona. El relé es un dispositivo eléctrico en el que, por medio de un electroimán; se acciona un juego de uno o varios contactos que permiten abrir o cerrar otros circuitos eléctricos independientes, es decir, se puede separar dos circuitos que trabajen a distintas tensiones.

Ilustración 71: Aspecto de placa de 2 relés 5V para Arduino

Una vez explicado que es un relé, se va a comenzar a explicar que sería necesario para **controlar una persiana** y de la **iluminación.**

Los motores de las persianas, suelen ser motores de corriente continua, y trabajan a 220v, mientras que el Arduino trabaja a tan solo 5v. Para poder realizar el control del motor de la persiana con el Arduino, se van a tener utilizar dos relés, ya que cada uno de los relés se encargará de hacer girar el motor en un sentido u otro (*Ilustración 72).*

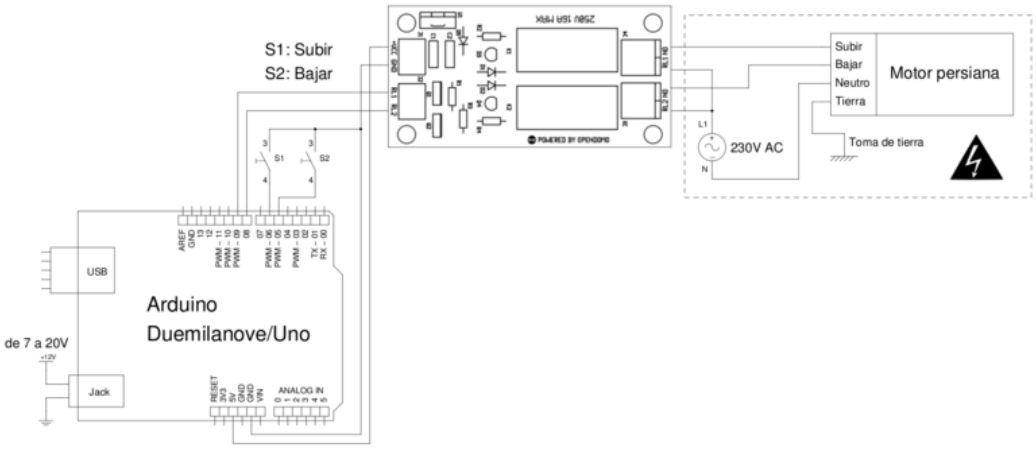


Ilustración 72: Conexión de motor de 220v con Arduino

Una vez explicado lo necesario para controlar una persiana, se va a explicar que se necesita para lograr el ***control de la iluminación*** de una vivienda, en este caso, el de una bombilla. Para ello también se va a necesitar el uso de relés, ya que las bombillas de las viviendas funcionan a 220v (corriente alterna). Para un correcto control de las bombillas, se suele utilizar un *transistor BJT*, por si la potencia que nos proporciona Arduino no fuera suficiente, y también se ha de utilizar un *diodo rectificador* que se encargue de que las corrientes solo se muevan en un sentido y proteger el Arduino (*Ilustración 73*).

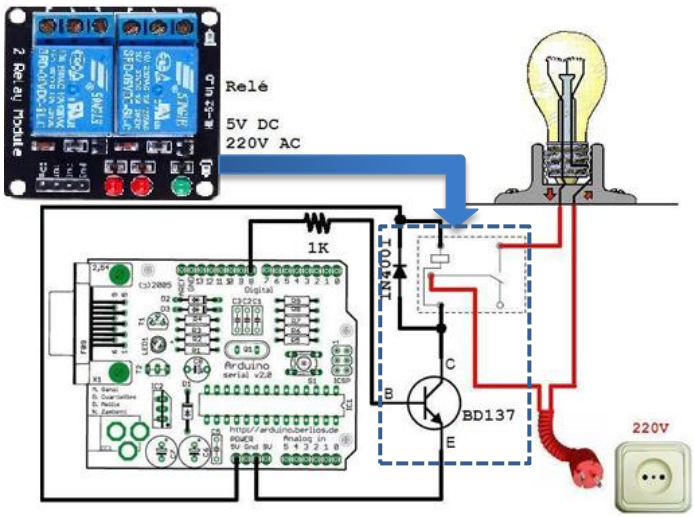


Ilustración 73: Conexión de un punto de luz (220v c.a.) con Arduino

Con un único relé se va a poder controlar una o un conjunto de bombillas, y se encargara de controlar el encendido o apagado de la o las bombillas.

CAPITULO 5: RESULTADOS

5. RESULTADOS.

Todos estos resultados están detallados en el apartado de resultados del documento principal. A continuación se irán viendo algunos resultados obtenidos y todos ellos han sido satisfactorios.

Lo primero que se ha realizado, ha sido una comprobación de cuantas vueltas da la persiana, desde la posición más baja de la ventana, a su posición más alta, para recoger en su totalidad, el movimiento de la persiana.

Una vez que se ha comprobado del número de vueltas que va a poder dar las persianas, que en este caso va a poder **girar 5 vueltas** como máximo, se comienza a realizar los distintos ensayos con cada uno de los servomotores para comprobar cuantos milisegundos se necesitan activar cada servomotor para que realice los 5 giros completos.

En la *Tabla 6* se muestran los distintos valores se muestran los milisegundo que necesitan cada uno de los servomotores para girar 5 vueltas, girando todos con la misma salida PWM, que en este caso, ha sido 10PWM por encima (subida) o por debajo(bajada) de su valor de reposo.

Servomotores	5 vueltas		Valor de reposo (PWM)	Velocidad de giro	
	Subida (ms)	Bajada (ms)		Subida (PWM)	Bajada (PWM)
1	8450	8000	77	87	67
2	8050	7900	80	90	70
3	8050	8300	84	94	74
4	7575	7500	78.5	88.5	68.5

Tabla 6: Prueba 1, tiempo en girar 5 vueltas

Como se puede observar en los valores de cada uno de los 4 servomotores, tienen un valor distinto. Esto es debido al bajo coste de los servomotores, y a que han sido modificados manualmente para ahorrar en costes. Este problema no se tendría con uno servomotores reales, ya que son más fiables.

Para solucionar este problema se ha intentado ajustar la velocidad de giro de cada uno de los servomotores por separado, para conseguir que todos tarden lo mismo en subir y bajar. Por lo que la nueva *Tabla 7* quedaría de la siguiente manera.

Servomotores	5 vueltas		Valor de reposo (PWM)	Velocidad de giro	
	Subida (ms)	Bajada (ms)		Subida (PWM)	Bajada (PWM)
1	8050	8050	77	88	67
2	8050	8050	80	90.1	70.1
3	8050	8300	84	95.2	75
4	8050	8050	78.5	88	69.5

Tabla 7: Prueba 2, ajustar las velocidades

Ahora ya se ha conseguido solucionar el error, en cierto modo, pero se puede observar que el servomotor número 3 ha sido el único que no se ha conseguido modificar su tiempo. Por lo tanto, como en el entorno de pruebas hay 2 habitaciones, 1 de ellas con tres ventanas y otra con una sola venta, se ha tomado la decisión de que el único servomotor número 3 sea el empleado para controlar la ventana de la habitación pequeña, y los otros tres servomotores (1, 2 y 4), se utilicen para la habitación grande (la que tiene 3 ventanas).

El inconveniente de no poder hacer girar a todos los servomotores con la misma salida de PWM y que tarden lo mismo en girar 360º, es debido a que se han tenido que modificar manualmente, para conseguir un giro continuo de los servomotores.

5.1. Resultados de las distintas hipótesis del Sistema Difuso.

En este apartado, se va a tratar de realizar un pequeño estudio sobre las distintas hipótesis que se podrían generar en un futuro, y comprobar cómo reaccionaría el entorno de pruebas.

Las hipótesis que se van a llevar a cabo son las siguientes:

1) Que no haya presencia (lo que supone que el sistema va a estar parado).
2) Que haya presencia, que haga un día muy claro y calor.
3) Que haya presencia, que haga un día muy claro y normal.
4) Que haya presencia, que haga un día muy claro y frío.
5) Que haya presencia, que haga un día claro y calor.
6) Que haya presencia, que haga un día claro y normal.
7) Que haya presencia, que haga un día claro y frío.
8) Que haya presencia, que haga un día nublado y calor.
9) Que haya presencia, que haga un día nublado y normal.
10) Que haya presencia, que haga un día nublado y frío.
11) Que haya presencia, que haga un día oscuro y calor.

12) Que haya presencia, que haga un día oscuro y normal.
13) Que haya presencia, que haga un día oscuro y frío.
14) Que haya presencia, que haga un día muy oscuro y calor.
15) Que haya presencia, que haga un día muy oscuro y normal.
16) Que haya presencia, que haga un día muy oscuro y frío.

Para la realización de dicho estudio, se van a ir simulando las distintas suposiciones sobre el entorno de pruebas, y se anotarán los resultados finales en la *Tabla 8*. A su vez se irán copiando una imagen del centroide que genera el sistema difuso y se adjuntará a la misma tabla.

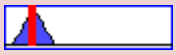
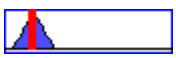
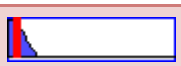
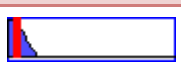
Presencia	Luminosidad	Temperatura	Salida	Salida Difusa	Salida (ms)	Porcentaje de subida
No	-	-		Quieto	0	-
Sí	Muy Claro	Calor		Medio Subida	4062	50
Sí	Muy Claro	Normal		Subida Total	8050	100
Sí	Muy Claro	Frío		Muy Subida	5570	70
Sí	Claro	Calor		Medio Subida	4062	50
Sí	Claro	Normal		Muy Subida	5567	69
Sí	Claro	Frío		Muy Subida	5570	70
Sí	Nublado	Calor		Poco Subida	2737	34
Sí	Nublado	Normal		Medio Subida	4062	50
Sí	Nublado	Frío		Poco Subida	2737	34
Sí	Oscuro	Calor		Bajada	1	0
Sí	Oscuro	Normal		Poco Subida	2737	34
Sí	Oscuro	Frío		Muy Poco Subida	1450	18
Sí	Muy Oscuro	Calor		Muy Poco Subida	1450	18
Sí	Muy Oscuro	Normal		Bajada	46	0
Sí	Muy Oscuro	Frío		Bajada	46	0

Tabla 8: Resultados de las distintas reglas sobre el Entorno de Pruebas

Como se puede observar en la *Tabla 8*, los valores obtenidos en las distintas salidas, son las mismas que las marcadas en las Reglas del sistema difuso (*Ilustración 68: Reglas del comportamiento de las persianas*), por lo que, se podría afirmar que el sistema difuso funciona correctamente.

Es importante mencionar, que no siempre se van a dar estos valores mostrados, ya que van a depender del grado de pertenencia de cada uno de los distintos campos de entrada (luminosidad y temperatura). Los resultados mostrados en la *Tabla 8*, muestran los valores más críticos de las distintas reglas mencionadas anteriormente.

En la *Ilustración 74* se podrá observar un gráfico en 3D, donde se muestran las distintas salidas del Sistema Difuso elaborado para el control de las distintas persianas para cuando hay presencia, en función de las entradas difusas del sistema, *Simulador de Temperatura* y *Sensor LDR*. Estos valores de salida que se reflejan en el gráfico 3D, van a depender principalmente de las reglas creadas para dicho Sistema Difuso, y en función de los distintos valores de entrada que se pueden llegar a dar, van a suponer una salida u otra.

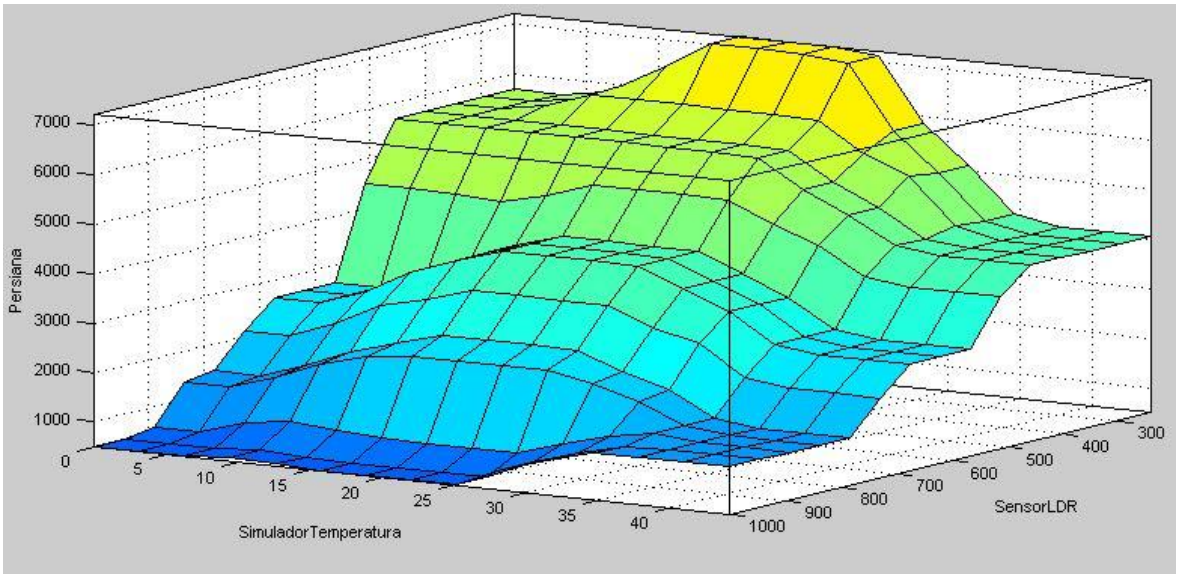


Ilustración 74: Surface del Control Difuso de las persianas

CAPITULO 6: ESTUDIO ECONÓMICO

6. ESTUDIO ECONÓMICO.

Para realizar un estudio económico del proyecto, se va a dividir en dos partes, una donde se reflejará el coste que ha supuesto la elaboración del *Entorno de Pruebas*, incluyendo el tiempo y todos los materiales empleados para ello. Y una segunda parte, en la que se reflejará cuánto costaría la implantación de este proyecto sobre un caso real. Para ello se va a hacer el coste de la implantación y control, de una persiana (con motor incorporado) y 8 bombillas.

6.1. Presupuesto del entorno de pruebas

Para la elaboración de este proyecto se pretendió desde un principio hacerlo lo más económico posible. Se hizo un estudio de todos los componentes que se van a necesitar para la elaboración del Entorno de Pruebas y darle las características deseadas. Una vez que se hizo la lista de los distintos componentes necesarios, se buscaron en varias tiendas online los componentes para poder comparar los precios y conseguirlos al menor coste posible. Aunque a en alguna ocasión se ha escogido algún componente algo más caro, ya que el envío era más rápido.

Hay una excepción, y es, que los **listones de contrachapado**, se compraron en una tienda física de Ciudad Real (Bricorama). Se compraron 2 de 1.22x2.44 y 3mm de grosor por 22.40€ cada listón.

Para la **persiana** no voy a poner ningún valor económico, ya que se ha utilizado como eje rotatorio, un tubo de regadío de 4.55 mm de diámetro interno, que se ha reciclado. Dicho tubo, tiene el diámetro requerido para poder adaptarse correctamente a los servomotores, aunque se pegarán con pegamento instantáneo, para asegurarnos de que no deslicen al girar.

Para hacer una simulación de las lamas de las persianas, se ha reciclado retales de nailon de un paraguas, ya que es opaco y se puede enrollar y desenrollar sin coger forma.

A continuación se detalla una lista donde aparecen los componentes buscados en diferentes tiendas online.

➢ *Placa Arduino Mega*

- Arduino ATMega2560 + USB, 10.98€ con envío gratis (15 – 20 días laborales).
  - http://www.ebay.es/itm/16AU-Microcontroladores-Bordo-USB-Cable-a-Arduino-MEGA2560-Modulo-R3-ATmega2560-/141430160413?pt=LH_DefaultDomain_186&hash=item20ede4fc1d
- Arduino ATMega2560 + USB, 17.99€ con envío gratis (10-20 días laborales).
  - http://www.miniinthebox.com/es/tablero-de-fabricante-de-micro-controlador-arduino-mega-2560-atmega-2560-16au_p340810.html

➢ *Servomotores*

- 4 Servomotores SG90 de 9g cada uno, 11.52€ + 2.69€ de gastos de envío (5-12 días laborales)
  - http://www.ebay.es/itm/4x-pcs-SG90-9g-Mini-Micro-Servo-RC-helicopter-airplane-car-boat-Arduino-Pi-/171604369077?pt=LH_DefaultDomain_3&hash=item27f46adeb5
- 1 Servomotor SG90 de 9g, 4.18€ con envío gratis (10-20 días laborales). Se pueden llegar a encontrar los mismos servos en oferta, hasta por 2.99€ cada uno.
  - http://www.miniinthebox.com/es/towerpro-sg90-9g-mini-servo-con-accesorios-para-arduino-funciona-con-las-juntas-de-oficiales-arduino_p1141452.html

➢ *Sensores LDR*

- Sensor LDR GL5528, 2 x 1.49€ envío gratis, 5 x (1€+0.95€ de envío) ó 10 x 2.49€ con envío gratis.
  - http://www.ebay.es/itm/2x-fotorresistencia-LDR-gl5528-10k-luz-luminosidad-Arduino-PIC-Prototipos-/251935406656?pt=LH_DefaultDomain_186&hash=item3aa8853a40
  - http://www.ebay.es/itm/5-x-FOTORESISTENCIA-5528-Tipo-LDR-10k-1MOhm-ARDUINO-fotoresistencias-gl5528-/221217252689?pt=LH_DefaultDomain_186&hash=item3381936551
  - http://www.ebay.es/itm/10-x-FOTORESISTENCIA-GL5528-Tipo-LDR-10k-1M-arduino-pic-pcb-5mm-fotorresistencia-/181573061559?pt=LH_DefaultDomain_186&hash=item2a46990bb7

- 10 x Alta sensibilidad 10K Ohm Fotorresistores, 1.99€ con envío gratis.
  - http://www.miniinthebox.com/es/alta-sensibilidad-10k-ohm-fotoresitores-rojo-blanco-10-piezas_p1092712.html

➢ ***Sensores PIR***

- Modulo Detector De Movimiento PIR HC-SR501, 1.80€ + 2€ de envío (3-5 días laborales).
  - http://www.ebay.es/itm/Modulo-Detector-De-Movimiento-PIR-Sensor-Infrarrojo-Pasivo-HC-SR501-Arduino-/111624652148?pt=LH_DefaultDomain_186&hash=item19fd590574
- Módulo del sensor de infrarrojos HC-SR501, 1.89€ con envío gratis (10-20 días laborales).
  - http://www.miniinthebox.com/es/del-cuerpo-humano-modulo-del-sensor-de-infrarrojos_p406976.html

➢ ***Fuente de alimentación de 9v.***

Esta fuente es necesaria, ya que para el empleo de los servomotores y de la pantalla LCD, es necesario el empleo de una fuente de alimentación externa de al menos de 9v, para no dañar el Arduino.

- Fuente de alimentación 9V 1A, 5.75€ + 1€ de gastos de envío (3-5 días laborales)
  - http://www.ebay.es/itm/291279280247?ssPageName=STRK:MEWNX:IT&_trksid=p3984.m1439.l2649
- Fuente de alimentación 9V 1A, 4.99€ gastos de envío gratis (10-20 días laborales)
  - http://www.miniinthebox.com/es/9v-1a-del-adaptador-del-cargador-de-alimentacion-para-arduino-120cm-de-cable_p905379.html

➢ ***Pantalla LCD 1602 I2C***

- Pantalla IIC/I2C/TW Interfaz 1602 16X2, 4.19€ gastos de envío gratis (15 -25 días laborales)
  - http://www.ebay.es/itm/Monitor-Pantalla-IIC-I2C-TW-Interfaz-1602-16X2-Serial-Modulo-LCD-para-Impresora-/251845400611?pt=LH_DefaultDomain_186&hash=item3aa327d823

- Geeetech IIC / I2C / TWI 1602 pantalla del módulo LCD serie para Arduino, 5.53€ gastos de envío gratis (10-20 días laborales)
  - http://www.miniinthebox.com/es/geeetech-iic-i2c-twi-1602-pantalla-del-modulo-lcd-serie-para-arduino_p2867274.html

➢ *Pack de componentes de Arduino*

Este es un pack, donde tenemos una gran variedad de componentes, que van a ser útiles para la construcción de la maqueta. Cierto es que hay muchos de los componentes que se van a quedar sin utilizar, pero debido al bajo coste de dicho pack, se ha optado a comprarlo así, en vez de, pillarlos por separado cada uno de los componentes necesitados (resistencias, leds o potenciómetros).

Los componentes de lo que consta son:

- Resistencias (30x100Ω, 30x1kΩ, 30x4.7kΩ, 30x10kΩ, 30x47kΩ, 30x100kΩ, 30x1MΩ)
- Potenciómetros rotatorios (2x10 kΩ y 2x100kΩ)
- Leds (3 verdes, 3 rojos, 3 amarillos y 3 blancos)
- 6x12x12 Pulsadores

El siguiente enlace te dirige a la tienda online, donde se podrá comprar dicho pack, por tan solo 5.70€ con los gastos de envío gratis (10-20 días laborales)

- http://www.miniinthebox.com/es/resistentes-carbono-universales-partes-rotativas-potenciometros-fijados-para-arduino_p2580197.html

➢ *Cables Jumper para Arduino*

- 40xCables Jumper macho/macho de 20 cm, 1.83€ envío gratuito (12-17 días laborales)
  - http://www.ebay.es/itm/40Pin-Dupont-wire-jumper-cable-20cm-2-54MM-male-to-male-1P-1P-For-Arduino-HLRG-/351256995018?pt=LH_DefaultDomain_186&hash=item51c88c64ca
- 40xCables Jumper Macho a Macho para Arduino 22cms, 1.99€ envío gratuito (10-20 días laborales)
  - http://www.miniinthebox.com/es/cables-macho-a-macho-para-tableros-de-circuitos_p364598.html

La lista anterior corresponde a las tiendas donde se ha mirado a la hora de comprar los distintos componentes. Los precios corresponden a los precios que hay actualmente, y muchos de ellos los compré hace ya un año, como el Arduino ATMega2560, y entonces costó algo más. Aun así, se va a realizar un presupuesto de cuanto supondría el coste material del entorno de pruebas (*Tabla 9*).

MATERIAL	Precio (€)
Listones de madera	44,8
Arduino Mega + USB	10,98
Servomotores	14,21
Sensores LDR	1,95
Sensores PIR	3,78
Fuente de alimentación 9v	4,99
Pantalla LCD1602 I2C	4,19
Pack de componentes Arduino	5,7
Cables Jumper para Arduino	1,83
TOTAL	**92,43€**

Tabla 9: Presupuesto de los materiales empleados en el Entorno de Pruebas

Las horas invertidas en la realización del proyecto son:

- En el mes de **octubre** y **noviembre** se investigó y estudió sobre todo lo que conlleva un Edificio Inteligente, dedicando cuatro días a la semana y 2 horas al día, haciendo un total de aproximadamente ***72 horas*** en los dos meses.
- En **diciembre** se siguió con el estudio e investigación sobre los Edificios Inteligentes mientras a la vez se empezó a investigar sobre lógica difusa y aprendizaje automático. En este mes se dedicaron una media de 4 días a la semana con un total de 4 horas diarias aproximadamente. En total en este mes se dedicaron alrededor de ***80 horas***.

- En el mes de **enero** y **febrero** se empezó a estudiar sobre lógica difusa, se empezó a construir la maqueta y a empezar a programar con Arduino, dedicando cinco días a la semana con tres horas cada día. En estos dos meses se han hecho aproximadamente **1*20 horas***.
- **Marzo y Abril** fueron cruciales, ya que se terminó de montar la maqueta, se acabó la lógica difusa y también se terminó de programar en Arduino, dedicando cinco días a la semana y cada día seis horas. En este mes se dedicó aproximadamente ***195 horas***.
- La **Mayo** y la **primeras dos semanas de Junio** se han dedicado muchas horas de trabajo para completar el documento e ir dando los últimos retoques al proyecto. En estas últimas semanas se dedicó tiempo todos los días, llegando días en los que se echaron 11 horas. En estas dos semanas se han dedico aproximadamente ***305 horas***.

Sumando todas las horas dedicas al proyecto hacen un total de **772 horas**. Estas horas si las tiene que dedicar un ingeniero en su puesto de trabajo equivaldría aproximadamente 5 meses de trabajo. Un ingeniero recién graduado cobra en España alrededor de 1400 euros mensuales, por lo tanto, este proyecto hubiera costado desarrollarlo aproximadamente *6755 euros*, y si a esto se le suman los componentes empleados, hacen un total aproximadamente de ***6847.43 euros***. La suma de todos estos valores se podrán observar en la *Tabla 10*:

MES	*Nº de días*	*Precio*hora (8,75€/h)*
Octubre	72	630
Noviembre		
Diciembre	80	700
Enero	120	1050
Febrero		
Marzo	195	1706,25
Abril		
Mayo	305	2668,75
Junio		
Total	*772*	*6755 €*
	Componentes (92,43€)	*6847,43 €*

Tabla 10: Suma total del presupuesto del Entorno de Pruebas

6.2. Presupuesto del entorno de real

Para el presupuesto sobre un entorno real, se pueden usar los sensores utilizados en el entorno de pruebas, junto con el mismo Arduino, por lo que se cogerán el coste de estos materiales del apartado anterior.

Comenzaremos calculando el coste que supone el montaje de una persiana. El motor de persiana sobre el que se ha pedido presupuesto ha sido de la marca Gaviota, el modelo GM-12 (*Ilustración 75*).

Ilustración 75: Motores de persianas *Gaviota*

Estos motores son los indicados para persianas de 0 a 15 kg. Traen incorporado todo lo necesario para monitorizar la persiana (motor, eje, soportes, 2m de cable, interruptor y manual de instrucciones).

Para saber que motor es el adecuado para una persiana cualquiera, lo primero que hay que conocer es la superficie total de la ventana y el peso por m^2 de la lama de la persiana que se va a querer utilizar.

- Peso por m^2 lama de aluminio = $4kg/m^2$
- Peso por m^2 lama de PVC = $6kg/m^2$
- Peso por m^2 lama de madera = $10kg/m^2$
- Peso por m^2 lama de aluminio extrusionado (seguridad) = $12kg/m^2$

En este caso, a la hora de pedir presupuesto se pidieron lamas de PVC, para una ventana de 1.5 x 1.5 m, por lo que supondría una superficie de 2.25 m^2, y al ser lamas de PVC, supondría un peso total de **13.5 kg**.

El coste material que supondría **una ventana** de 1.5 x 1.5 m, sería de **144 €**:

- Motor GM-20 => 84 € (IVA incluido)
- Lamas de PVC (0.05 x 1.5 m) => 30 lamas x 2 €/unidad = 60 € (IVA incluido)

El presupuesto de la persiana (motor y lamas de PVC), corresponde a una tienda local de Miguelturra llamada *Metálicas Emar*.

A parte de este coste habría que añadir el coste del **módulo de relé de 2 canales**, sobre los que se ha mirado precios en dos sitios distintos:

- Módulo relé 5V 2 canales para Arduino, 3€ con envío gratis.
  - http://www.ebay.es/itm/MODULO-RELE-5V-2-CANALES-Arduino-placa-PIC-aislado-channels-2-reles-/221669941880?pt=LH_DefaultDomain_186&hash=item339c8ee278
- Tarjeta de relé 5v módulo de ampliación de 2 canales para Arduino, 2.89 € con envío gratis.
  - http://www.miniinthebox.com/es/arduino-2-canal-de-rele-de-5v-modulo-de-tarjeta-de-expansion_p364575.html

Para el *control de la iluminación* se va a comenzar mirando por el coste de un **módulo de relé de 8 canales,** ya que para una vivienda se controlará más de una bombilla.

- Módulo relé de 8 canales de 5V para Arduino, 8.88€ con envío gratis.
  - http://www.ebay.es/itm/Modulo-8-canales-de-reles-de-5V-para-Arduino-y-microcontroladores-relay-TE181-/311355848287?pt=LH_DefaultDomain_186&hash=item487e41365f
- Módulo relé de 8 canales de 5V para Arduino, 13.03€ con envío gratis.
  - http://www.miniinthebox.com/es/modulo-de-rele-de-12v-de-8-canales-nueva-marca-azul-y-rojo_p1236001.html

Como ya se mencionó en el apartado anterior, para un correcto control de bombillas se necesita de un *transistor BJT* y un *diodo rectificador*, para lo cual se ha mirado el coste en tiendas online y físicas.

- 5 x BD137 Transistores de tipo NPN, 1.40€ + 0.60€ de gastos de envío.
  - http://www.ebay.es/itm/5x-BD137-Transistor-NPN-TO-126-60V-1-5A-12-5W-BJT-/321405467841?pt=LH_DefaultDomain_186&hash=item4ad5423cc1

  También se puede encontrar en cualquier tienda física de electrónica y tienen un precio rondando los 30 céntimos la unidad.
- 50 x Diodo rectificador 1N4001 para Arduino, 1.95€ con envío gratuito.
  - http://www.ebay.es/itm/50-Diodos-Rectificador-1N4001-1A-50V-Arduino-Rectifier-Diodes-Dioden-diode-/151688397606?pt=LH_DefaultDomain_186&hash=item2351554326

  Estos de componentes también se pueden llegar a encontrar en tiendas físicas de electrónica, su precio suele oscilar los 15 céntimos la unidad.

También se ha buscado el coste de *sensores de temperatura analógicos*, en este caso del modelo *TMP36*, ya que permite realizar unas medidas de temperatura bastante precisas y de un coste bajo.

- Sensor de temperatura analógico TMP36 compatible con Arduino, 4€ con gastos de envío incluidos.
  - http://www.ebay.es/itm/TMP36-GZ-Temperatur-Sensor-Digital-Thermometer-Temperature-ARDUINO-A768-/261765463213?pt=LH_DefaultDomain_77&hash=item3cf26ffcad

El coste de las bombillas, dependiendo del tipo de bombillas que se quiera utilizar va a poder cambiar considerablemente el precio de la unidad. En este caso se van a dar presupuesto para ***bombillas de leds***, ya que si se comparan con bombillas de bajo consumo o con una bombilla tradicional incandescente son muchas las ventajas. Por ejemplo una bombilla led consume 2.5 veces menos que una de bajo consumo y 8.9 veces menos que una bombilla de incandescencia, generando los mismos lúmenes, por lo que se estará ahorrando hasta un 80% de energía eléctrica, es decir, una mayor eficiencia energética.

Otras de las ventajas de las bombillas de led serían: son regulables, larga vida útil (50.000 horas frente a las 2.000 horas de una tradicional), encendido inmediato, no genera luz ultravioleta ni infrarroja, etc... El único inconveniente sería que el precio por unidad en comparación con una bombilla de bajo consumo, ya que es prácticamente el doble.

Una vez que se ha explicado todo lo anterior, se va a dar el presupuesto final, incluyendo la mano de obra que supondría el montaje de las persianas y bombillas del hipotético caso planteado.

MATERIALES	PRECIO
Arduino Mega + USB	10,98 €
Sensores LDR	1,95 €
Sensores PIR	3,78 €
Fuente de alimentación 9v	4,99 €
Pantalla LCD1602 I2C	4,19 €
Cables Jumper para Arduino	1,83 €
Motor tubular GM-20	10,00 €
Lamas de PVC	60,00 €
Módulo relé 5v 2 canales	2,89 €
8 x Bombillas LED (9w y 700lm)	47,60 €
Módulo relé 5v 8 canales	8,88 €
8 x Transistor BJT NPN BD137	2 €
8 x Diodo rectificador 1N4001	1 €
1 x Sensor de temperatura TMP36	4 €
Mano de obra de instalación	60 €
Programa	50 €
TOTAL	***274,69 €***

Tabla 11: Presupuesto del coste que supondría implantarlo sobre un Entorno Real

En el precio total indicado en la *Tabla 11*, solamente se hace referencia al coste material, y a esto habría que añadir el coste de la mano de obra que supondría la instalación de la misma, y dependiendo de la compañía que se contrate va a tener un precio u otro.

En este caso se ha solicitado un presupuesto como ya se mencionó anteriormente a una empresa local (*Metálicas Emar*), y el *coste de la mano de obra* para la instalación de la persiana es de **60€.** A la hora de dar el presupuesto se ha dado por hecho, que la vivienda donde se fuera a implantar la persiana monitorizada y los distintos puntos de luz, ya tenía la instalación eléctrica implantada.

También se le ha añadido un coste extra, del *programa* que se ha diseñado a lo largo del proyecto para el control de la instalación, y se le ha puesto un precio de **50€**. Este coste sería el mismo si se instala una o más persianas.

Para hacer una comparativa del coste que supondría la implantación de a unas persianas motorizadas e iluminación (*167.60€*) y el coste de la implantación de este proyecto a un caso real (*274.69€*), supone una diferencia de **107€**.

CAPITULO 7: CONCLUSIONES

7. CONCLUSIONES.

Para el desarrollo del presente trabajo de fin de grado se han invertido muchas horas de investigación en búsqueda de información y desarrollo sobre los distintos temas tratados, como por ejemplo, la lógica difusa, domótica y todo lo que conlleva el control de sistemas automatizados con Arduino. Todas estas horas de trabajo me han servido para ampliar mis conocimientos sobre la lógica difusa y conocer las competencias sobre su uso. También he visto la importancia de la programación en la creación de sistemas autosuficientes, así como la aplicación de las matemáticas y leyes de electricidad/electrónica para la resolución de problemas de ingeniería. Todos estos conocimientos han sido un complemento muy satisfactorio para mi formación en el grado de Ingeniería Eléctrica.

En este trabajo fin de grado se propuso como objetivo fundamental construir un entorno de pruebas desde cero, de la forma más económica posible, para conseguir así una demostración práctica sobre la automatización de persianas e iluminación de viviendas. Una vez se fue desarrollando el proyecto se pensó en aplicar técnicas de lógica difusa, lo que supuso un plus de mejora para el proyecto, ya que gracias a ello se lograron conseguir muchas mejoras, entre ellas, que la persiana se moviese de forma gradual, en vez de tener un comportamiento tan prefijado. Al principio costó llegar a entender lo que era la lógica difusa y su aplicación a dicho entorno, pero luego facilitó mucho la elaboración del programa para controlar las persianas.

Para poder realizar todo lo anterior con garantías, en la primera fase del proyecto, se realizó un estudio del estado del conocimiento de la domótica, para poder llegar a comprender mejor el funcionamiento de los distintos actuadores y sensores. Durante esta fase pudimos ver los posibles problemas que más adelante tendríamos que afrontar con algunas soluciones propuestas. Tras estas soluciones propuestas y el estudio propio realizado sirvió para plantear nuestras propias soluciones y cumplir con los objetivos marcados inicialmente.

Al final del proyecto se dividió en dos partes fundamentales:

- Diseño y construcción desde cero de un entorno de pruebas, para la simulación del control de persianas e iluminación.
- Modelación de un sistema difuso para definir el comportamiento del sistema.

Todos los objetivos citados se han cumplido a lo largo del proyecto. *En primer lugar* como se indica en el *Capítulo 4*, **se diseñó un entorno de pruebas** haciendo uso, primero del programa de diseño SketchUp, y posterior mente su construcción mediante contrachapado y controlado con el microcontrolador Arduino, y componentes compatibles con él (servomotores y sensores). Dicho diseño es limitado debido a la disponibilidad económica.

En segundo lugar, una vez que se había construido el entorno de pruebas, se procedió al diseño de los movimientos de los distintos servomotores y creación de librerías con Arduino. Como en todo proyecto, las soluciones que se plantearon inicialmente presentaban problemas y antes estos problemas se proponían nuevas soluciones, llegando a una versión de movimientos satisfactorios. La creación del **sistema difuso** fue el mayor reto al que me he enfrentado, ya que era algo totalmente novedoso para mí. Tras ensayos de comportamiento del control difuso se dieron ciertos problemas, que se fueron resolviendo realizando nuevos ensayos y estudios con diversos factores y variables, llegando a una solución final donde el entorno de pruebas tiene un control difuso muy bueno cumpliéndose este objetivo.

La consecución de estos objetivos, ha permitido obtener unos resultados positivos, al conseguir que el entorno de pruebas tuviera un autocontrol, por lo que no es necesario el esfuerzo humano para la puesta en marcha de las luces o persianas. Dicho control del sistema, será especificado por cada usuario, es decir, que el usuario elegirá en todo momento como quiere que actúen las persianas y la iluminación de la vivienda. Incluso puede ser el mismo usuario quien pueda modificar el comportamiento del sistema, sin tener por que saber de lenguaje de programación, ni de los datos técnicos de los distintos sensores usados, ya que se van a trabajar con términos lingüísticos, que sí se comprende por cualquier usuario, como por ejemplo, mientras que haya presencia, haga frío haga un día muy claro, quiero que la persiana se suba del todo.

También se hizo un estudio, para saber que se necesitaría para llevarlo a cabo sobre un **entorno real**. El programa creado podría valer casi en su totalidad, pero habría que realizar algunas modificaciones, ya que se dejaría de controlar un servomotor, y se pasaría a controlar motores de corriente continua. Lo que supone una gran ventaja, ya que llevan incorporados en su sistema, finales de carrera, y otra ventaja sería que se pueden controlar los grados de giro, mientras que con los servomotores se controlaban, mediante el tiempo que estaban girando. Pero habría un inconveniente, y es que para poder llegar a controlar

los motores de las persianas reales, que trabajan a 220v, habría que disponer de al menos 2 relés de potencia, ya que Arduino trabaja con 5v. Y lo mismo ocurriría para controlar la iluminación de una vivienda, que habría que utilizar al menos 1 relé. Por lo demás, los sensores usados en el entorno de pruebas, podrían llegar a valer en un entorno real.

Me gustaría resaltar que la realización de este proyecto ha sido una de las mejores experiencias, con un final muy satisfactorio durante el desarrollo formativo en el grado de ingeniería eléctrica, puesto que se puede ver en él, gran parte de los conocimientos adquiridos durante estos 4 años de carrera. Para cualquier estudiante, es de gran satisfacción que un proyecto donde se inicia con ideas, suposiciones, colaboración del profesorado dándome sugerencias, y afrontar retos de temas a tratar sobre los que no dispones de conocimientos iniciales; se logre el funcionamiento correcto, cumpliendo los objetivos marcados.

Para terminar, mencionar mi total disposición para ceder dicho Entorno de Pruebas, para que sirva tanto como muestrario para la atracción de nuevos alumnos o para el uso de futuros alumnos, donde poder coger ideas o incluso para reutilizarlo como un nuevo TFG, donde puedan añadir nuevos elementos de control, como por ejemplo, el control de climatización o el empleo de energías renovables como fuente de alimentación de la instalación.

CAPITULO 8: BIBLIOGRAFÍA

8. BIBLIOGRAFÍA.

Barro Ameneiro, S., & Bugarín Diz, A. (2002). *Fronteras de la Computación.* Diaz de Santos, S.A.

Gallardo Vázquez, S. (2013). *Configuración de Instalaciones Domóticas y Automáticas.* Paraninfo SA.

Gómez, E. V., & Rey Martínez, F. J. (2006). *Eficiencia Energática en Edificios, Certificación y Auditoríias Energéticas.* Paraninfo.

Harke, W. (2007). *Smart (Home) Control.* Marcombo Ediciones Técnicas.

Henríquez, M. R., & Palma, P. A. (2011). *Control Automático de Condiciones Ambientales en Domótica usando Redes Neuronales Artificiales.* Obtenido de Información tecnológica: http://www.scielo.cl/scielo.php?pid=S0718-07642011000300014&script=sci_arttext

Martín del Brio, B., & Sanz Molina, A. (2006). *Redes neuronales y sistemas corrosos.* RA-MA.

Miguel, P. A. (2011). *Electrotecnia.* Paraninfo.

Ministerio de Industria, T. y. (2011). *Reglamento regulador de las Infraestructuras Comunes de Telecomunicaciones para el acceso a los servicios de telecomunicaciones en el interior de edificios.* Paraninfo.

Nilsson, N. J. (2001). *Inteligencia Artificial, Una nueva síntesis.* McGRAW - HILL.

Norving, S. R. (2008). *Inteligencia Artificial, Un enfoque moderno.* PEARSON EDUCATION, S.A.

Rich, E., & Knight, K. (1994). *Inteligencia Artificial.* McGRAW-HILL.

Rodríguez Fernández, J. (2012). *Instalaciones Domóticas.* Paraninfo SA.

Russell, S., & Norving, P. (2004). *Inteligencia Artificial, Un enfoque moderno.* PEARSON EDUCACIÓN, S.A.

Sole, A. C. (2005). *Instrumentación Industrial.* MARCOMBO S.A.

Tobajas Garcia, C. (2011). *Instalaciones Domóticas.* Cano Pina S.L.

Trillas, E., Terricabras, J., & Alsina , C. (1995). *Introducción a la lógica difusa.* Ariel.