



ESCUELA TÉCNICA SUPERIOR
DE INGENIEROS DE TELECOMUNICACIÓN

ESCUELA TÉCNICA SUPERIOR DE INGENIEROS DE TELECOMUNICACIÓN

INGENIERÍA DE TELECOMUNICACIONES

PROYECTO DOMÓTICA CON KINECT

Director del proyecto: Antonio Albiol Colomer

Autor del proyecto: Javier Valls Catalá

15 de octubre de 2013

Índice general

1. Introducción.	1
1.1. Introducción.	1
1.2. Presentación del problema	1
1.3. Objetivos del proyecto	2
1.4. Solución propuesta	2
2. Antecedentes	3
2.1. Sistemas existentes	3
2.2. Proyectos de domótica.	4
2.3. Elementos empleados.	7
2.3.1. Kinect.	7
2.3.2. Arduino.	7
3. Análisis	9
3.1. Descripción de la solución	9
3.2. Flujo de trabajo	9
3.3. Requisitos de funcionamiento del sistema domótico.	10
4. Diseño	13
4.1. Diseño del programa del ordenador.	13
4.1.1. Conexión Kinect	13
4.1.2. Reconocimiento de voz	13
4.1.3. Reconocimiento de gestos	14
4.1.4. Puerto Serie	17
4.1.5. Temporizador	17
4.1.6. Sonidos	17
4.2. Programa Arduino	18
4.3. Hardware Robot	19
4.3.1. Componentes Robot.	19
4.3.2. Circuito eléctrico.	21
4.3.3. Estructura robot.	22
4.4. Interfaz Usuario	24
5. Resultados	27
5.1. Introducción.	27
5.2. Ejemplo de aplicación.	27
5.3. Presupuesto.	29
5.4. Cumplimiento de objetivos.	30

6. Conclusiones	31
6.1. Desarrollo del proyecto.	31
6.2. Conclusiones trabajo realizado.	32
6.3. Futuras ampliaciones.	32
A. Manual de usuario	33
6.4. Manual usuario.	33
6.5. Preguntas frecuentes.	35
Bibliografía y referencias	37

Capítulo 1

Introducción.

Este capítulo lo hemos dividido en cuatro secciones. En la primera introduciremos el proyecto y comentaremos su origen. En la segunda presentamos la problemática a desarrollar en este proyecto. En la tercera mostraremos los objetivos del proyecto y, por último, expondremos la solución propuesta al problema planteado.

1.1. Introducción.

En este proyecto vamos a desarrollar una solución domótica centrada en la cámara Kinect, con la que vamos a poder contar con reconocimiento de voz en castellano y reconocimiento de gestos. Se va a construir un robot con el que haremos una demostración práctica del funcionamiento del sistema.

La idea de crear este sistema domótico surgió a raíz de un concurso de programación de aplicaciones para Kinect que se celebró en la Escuela Técnica Superior de Ingenieros de Telecomunicación de la Universidad Politécnica de Valencia. Al concurso se presentaron tres proyectos y éste obtuvo el primer premio. Como resultado de dicho concurso se presenta este proyecto final de carrera.

1.2. Presentación del problema

Actualmente nos encontramos en la era de Internet. Vivimos constantemente rodeados de aparatos electrónicos capaces de conectarse a la red y de proporcionarnos información. La sociedad actual, cada vez más, busca tener todos los elementos cotidianos automatizados e interconectados. Se está produciendo una auténtica revolución y ha aparecido el término ‘Internet de las cosas’. Queremos que todos los electrodomésticos, dispositivos, ropa... se conecten entre si para que resulten más cómodos a los usuarios. Esta automatización, puede suponer una mejora en la calidad de vida de las personas y más aun si hablamos de personas con alguna discapacidad.

La automatización ha originado que el número de dispositivos conectados a internet haya aumentado exponencialmente, lo que ha supuesto un reto para los ingenieros de redes. Hasta ahora se trabajaba con direcciones IPv4 pero el aumento de terminales ha supuesto un agotamiento de las mismas, de ahí, que se haya tenido que migrar a redes IPv6 para satisfacer esta necesidad de automatización.

Además, en los últimos años hemos presenciado un notable aumento del interés por la domótica. Esta tecnología permite mejorar la gestión energética de nuestros hogares, aumentar la seguridad o el confort, o poder controlar todos los servicios domésticos entre otros desde cualquier parte del planeta. El proyecto

que aquí presentamos ha sido creado con la intención de ser un sistema domótico al alcance de todas las economías.

1.3. Objetivos del proyecto

En este proyecto nos hemos planteado los siguientes objetivos:

- Realizar un sistema domótico de bajo coste que sirva de control de una vivienda a través de gestos corporales, de reconocimiento de voz del usuario, pulsadores, o temporizador.
- Dotar de sensores como por ejemplo luz o temperatura al sistema domótico.
- Demostrar el funcionamiento de los actuadores con un robot con motores y luces de colores.

1.4. Solución propuesta

Como respuesta a nuestro proyecto 'Domótica con Kinect', hemos optado, a fin de alcanzar el primer objetivo, por utilizar un sensor Kinect que nos permite tener reconocimiento de voz mediante su array de micrófonos y reconocimiento de gestos gracias a su cámara infrarroja. Por último para poder controlar los diferentes tipos actuadores y sensores que tenga instalados el robot, segundo y tercer objetivo, utilizaremos una placa microcontroladora Arduino.

Como control central del sistema domótico, utilizaremos un ordenador que tenga instalado el sistema operativo Windows 7. A este ordenador le serán conectados la cámara Kinect y el robot.

Capítulo 2

Antecedentes

Dado que el interés por la domótica ha ido creciendo a medida que Internet se ha ido implantando en nuestros hogares, y la necesidad de automatizar las viviendas ha hecho que el número de soluciones para estos servicios aumente, en este capítulo haremos referencia a las características de los sistemas comerciales. Presentaremos una revisión de los proyectos sobre domótica existentes. Por último comentaremos brevemente los dispositivos que se han utilizado en el proyecto.

2.1. Sistemas existentes

En general los sistemas comerciales usan como criterio de clasificación el lugar donde reside la inteligencia del sistema. Esta clasificación da lugar a tres tipos de arquitectura: distribuida, centralizada y mixta [13]. En la arquitectura distribuida el control y la inteligencia del sistema está distribuido entre los diferentes sensores o actuadores. Cuando la información de los diferentes sensores la recibe un controlador central que procesa y envía órdenes a los diferentes actuadores tenemos una arquitectura centralizada. Una arquitectura mixta es aquella que está formada por varios dispositivos capaces de tomar decisiones y transmitirlas a otros dispositivos.

En nuestro proyecto vamos a optar por una solución centralizada, donde un ordenador se ocupará de la inteligencia del sistema, y varios dispositivos se encargarán de darle información a éste para posteriormente actuar según sus órdenes.

Por su parte, las diferentes tecnologías que usan las redes domésticas se clasifican en función del tipo de redes que utilizan. Encontramos las redes que se utilizan para la interconexión de dispositivos, mediante IEEE 1394, Bluetooth, USB, IrDA... redes de control y automatización, KNX, X10, EUB, ZigBee... o redes de datos como Ethernet, Homeplug, HomePNA o Wifi.

También podemos encontrar en el mercado varios tipo de redes estándar que nos permiten conectar los dispositivos sin importar de que marca o fabricante es cada uno:

- X10: Se trata de un protocolo que hace uso de los enchufes eléctricos para el control remoto de dispositivos. Es un estándar de código abierto y muy difundido en el mundo, pero tiene poca fiabilidad frente a ruidos eléctricos, por lo que resulta inadecuado para aplicaciones profesionales.
- KNX/EIB: Bus de instalación Europeo. Es un sistema que utiliza su propio cableado. Es posible utilizarlo a través de pasarelas en sistemas inalámbricos. Actualmente está muy extendido entre los fabricantes.

- ZigBee: Hablamos de un protocolo inalámbrico, basado en el estándar IEEE 802.15.4. Se emplea por su bajo consumo energético, su topología de red en malla y su fácil integración en sistemas dada su simplicidad.
- OSGi: Open Services Gateway Initiative es un protocolo que define las especificaciones necesarias para poder proporcionar diferentes servicios. Se conecta a un proveedor que nos proporciona los múltiples servicios de domótica.
- LonWorks: Dicho estándar se usa para controlar viviendas, edificios, industria o transporte.
- Universal Plug and Play: UPnP es un conjunto de protocolos de software abierto y distribuido para poder intercambiar información entre los dispositivos conectados a una red, no es necesaria una configuración previa.
- Modbus: Se trata del protocolo libre más antiguo. Permite la comunicación a través del RS485 o a través de ethernet. Hoy en día se siguen fabricando dispositivos.

En nuestro proyecto, dado que no existe ninguna red de las presentadas que sea compatible con la cámara Kinect, vamos a optar por emplear un protocolo de comunicación propietario, no basado en ningún estándar.

2.2. Proyectos de domótica.

En esta sección procederemos a presentar algunos de los proyectos de domótica que podemos encontrar a día de hoy.

- Casa domótica controlada por voz gracias a Android ([5]): Se trata de un proyecto de domótica que hace uso de un smartphone con sistema operativo Android para el reconocimiento de voz, un PC y un televisor inteligente compatible con la plataforma 'Google TV', además de equipamiento domótico para controlar los elementos del hogar. El presupuesto de este proyecto en equipamiento para automatización de la casa es de unos 300 dólares.



Figura 2.1: Casa domótica.

- Control domótico con Android ([9], [8]): Este proyecto es similar al precedente, pero en este caso se utiliza una Raspberry Pi para el control de la casa, además de unos módulos para la domótica y una interfaz táctil a través de un teléfono con Android. En este el equipamiento domótico, que utiliza el protocolo X10, tiene un presupuesto parecido al anterior, de aproximadamente 250 euros.

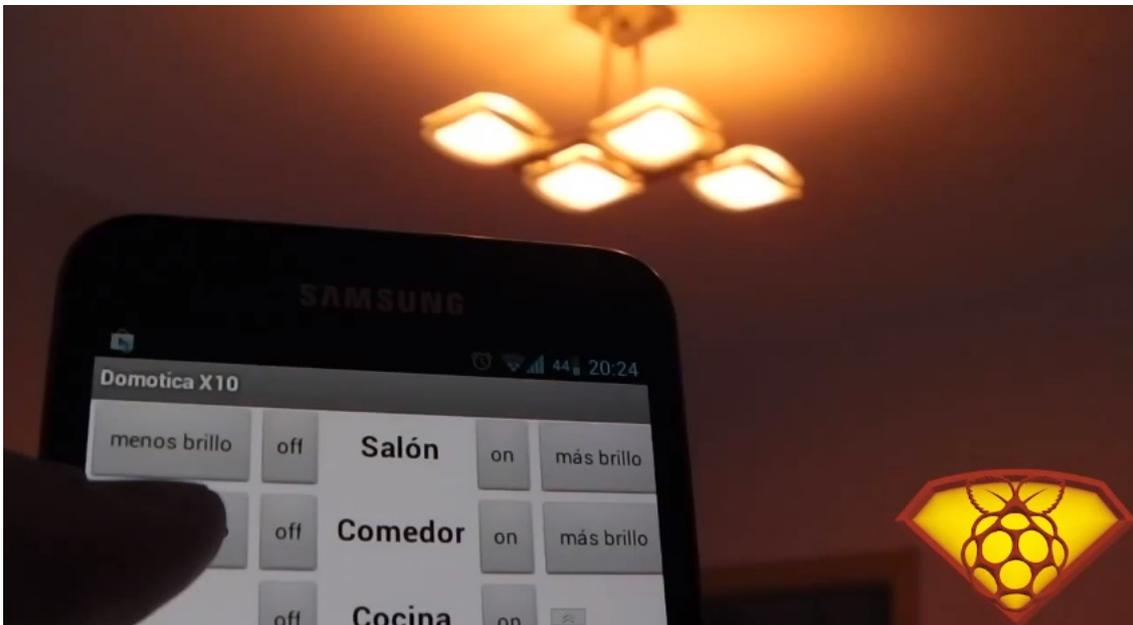


Figura 2.2: Domótica Raspberry Pi + Android.

- WeMo ([2]): Se trata de un enchufe inteligente de la empresa Belkin que tiene un precio de unos 50 euros. Es posible controlarlo mediante un teléfono con Android o con un iPhone, vía WiFi o la red de telefonía móvil. Además lo podemos programar mediante servicios IFTTT (If This Then That), es decir, "Si pasa Esto, hacer Aquello".



Figura 2.3: Interruptor domótico y la aplicación Android WeMo de Belkin.

- Android@Home([4]): Es un proyecto de la empresa Google para introducir su sistema operativo Android en el hogar. Se presentó en el año 2011 junto con un framework para que los desarrolladores pudieran programar sus proyectos, pero no ha tenido una gran expansión en el mercado ni han aparecido dispositivos compatibles.

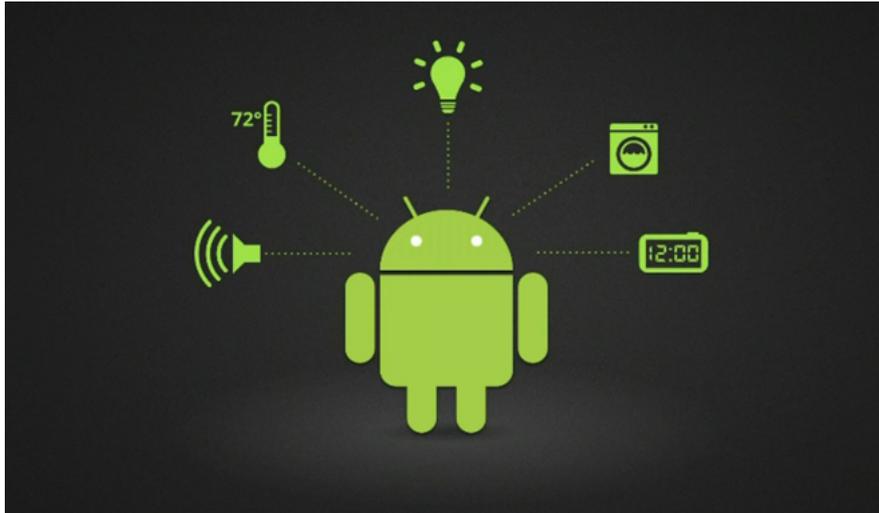


Figura 2.4: Android@Home.

- Smart ThinQ ([4]): Corresponde a la tecnología domótica que incorporan los nuevos electrodomésticos de la marca LG. Gracias a esta tecnología, los dispositivos son capaces de interconectarse mediante una red inalámbrica WiFi. Podemos controlar estos elementos a través de un ordenador, un teléfono móvil o un televisor.



Figura 2.5: LG Smart ThinQ.

El proyecto que presentaremos a continuación es diferente a los expuestos anteriormente, ya que hace uso de una cámara Kinect para el reconocimiento de gestos y de voz. A su vez, cuenta con un presupuesto ligeramente inferior a los dos primeros.

2.3. Elementos empleados.

La presente sección vamos a comentar los principales elementos de hardware que vamos a emplear para el proyecto y comentar brevemente la historia de cada uno de ellos. En primer lugar comentaremos la cámara Kinect, y finalmente la placa microcontroladora Arduino.

2.3.1. Kinect.

La cámara Kinect (figura 2.6) fué fabricada por la empresa israelí PrimeSense, y lanzada por Microsoft a finales del año 2010 como periférico para la videoconsola Xbox360. Obtuvo el récord Guinness de "Dispositivo de Electrónica de Consumo que más rápido se ha vendido en todos los tiempos", al vender ocho millones de unidades en dos meses, y hasta el año 2013 se habían vendido más de 24 millones de unidades.

Este dispositivo es una cámara que combina un sensor de color RGB (Rojo, verde y azul), un emisor de infrarrojos y un sensor de infrarrojos que mediante hardware calcula la distancia a la que se encuentra cada punto infrarrojo. Además esta cámara cuenta con un array de cuatro micrófonos con los que podemos emular un micrófono directivo y un motor con el que podemos subir o bajar el Kinect.



Figura 2.6: Cámara Kinect.

Durante el año 2011 se publicaron varios kits de desarrollo (SDK) del Kinect para ordenador, y entre los más populares encontramos el SDK OpenNI, el cual es un software abierto, y el SDK oficial de Microsoft, con el que podemos hacer aplicaciones no comerciales. Cada uno de ellos tiene unas ventajas y unos inconvenientes.

Esta cámara se puede encontrar en packs con la Xbox360 o de manera individual. Para poder utilizarla en el ordenador, es necesario comprarla por separado, ya que viene también con el cable USB para poder conectarla al PC.

2.3.2. Arduino.

Arduino (figura 2.7) es una plataforma de electrónica abierta, es decir, cualquiera puede utilizar o modificar el hardware y el software de esta placa microcontroladora. El proyecto lo iniciaron unos estudiantes italianos en el año 2005 para poder abaratar el coste de diseño de las placas controladoras, y desde entonces han aparecido diferentes versiones y revisiones de la placa. Además existen multitud de 'shield'

que se acoplan a esta placa y permiten añadir funcionalidades como Bluetooth, Wifi, pantallas LCD, ...

La Arduino se programa en Processing, un lenguaje de programación basado en Wiring, con el que se escriben los programas y posteriormente se cargan en la placa. Se trata de un lenguaje muy intuitivo y sencillo de utilizar. En la página de Arduino ([\[1\]](#)) se puede descargar toda la información y el software de dicha placa. Se ha creado una gran comunidad de desarrolladores de esta placa, y se pueden encontrar multitud de proyectos que la utilizan.



Figura 2.7: Placa Arduino.

Capítulo 3

Análisis

En este capítulo en primer lugar haremos una descripción de la solución a nuestro proyecto de domótica. En segundo lugar describiremos como fluye la información por el sistema domótico propuesto. Terminaremos estableciendo los requisitos que debe cumplir nuestro sistema.

3.1. Descripción de la solución

La solución esquemática de domótica de nuestro proyecto se muestra en la figura 3.1. Como primer componente tenemos la cámara Kinect, la cual capturará los gestos y la voz del usuario. El ordenador se encargará de ejecutar el programa que procesa los datos extraídos por la cámara y le da órdenes al robot, formado por la placa Arduino, una fuente de alimentación, unos sensores y unos actuadores. La placa Arduino obtendrá información de los sensores y actuará según las instrucciones del PC. Con el objetivo de controlar los picos altos de corriente, que podrían estropear la electrónica de la placa Arduino, utilizaremos una fuente de alimentación para los motores. Para dar salida gráfica a la interfaz de usuario contaremos con un monitor.

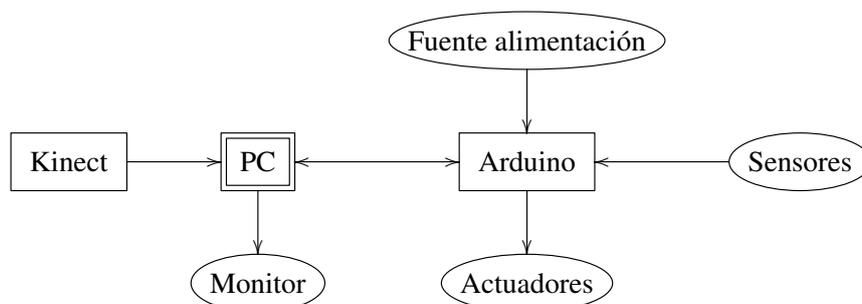


Figura 3.1: Esquema solución.

Los elementos que forman parte de la solución de domótica, tal como hemos comentado anteriormente, son todos ellos de bajo coste, lo que favorece su comercialización.

3.2. Flujo de trabajo

La solución domótica consta de dos programas. Uno de ellos se ejecutará en el ordenador, y el otro en la placa Arduino.

El programa que ejecutaremos en el ordenador será el más complejo. La cámara Kinect capturará los gestos y la voz del usuario, y dicho programa lo procesará. Recibirá información de los sensores de la placa arduino y cuando ésta le haga preguntas, responderá con información requerida. Además este software se encargará de la interfaz de usuario. En el esquema 3.2 podemos ver una simplificación del programa.

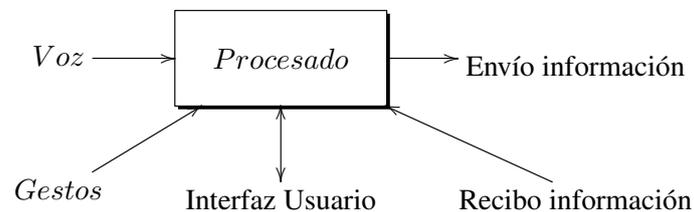


Figura 3.2: Esquema programa PC.

El programa que cargaremos a la placa microcontroladora arduino se encargará de preguntar al programa del PC por el estado de ejecución. El programa será como una máquina de estados. Empezará en un estado 'Desconectado' y pasará a un estado 'Conectado' al recibir la secuencia de conectar (figura 3.3). En el estado 'Conectado' estará comprobando si recibe la orden de desconectar. Este estado de funcionamiento será un bucle en el que preguntaremos por la posición de cada una de las partes del cuerpo y enviaremos la lectura de los sensores. Además, tendremos un protocolo de retransmisión y un circuito de 'watch dog' en funcionamiento por si no recibimos información del ordenador. Pasado un tiempo de temporizador, reiniciaremos el programa. De esta forma controlaremos que el usuario no ha desconectado el cable o hemos tenido un error de transmisión.

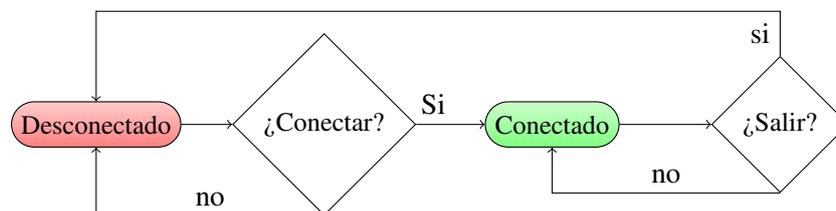


Figura 3.3: Esquema programa arduino.

Los programas del ordenador y de la placa Arduino, compartirán información entre ellos y se ejecutarán en paralelo. Para el intercambio de mensajes entre ambos programas, utilizaremos un sistema de paso de mensajes 'Polling', es decir, de consulta permanente mediante pregunta y respuesta. Esta decisión se ha tomado por varios motivos. Principalmente, por la gran diferencia entre ambos sistemas, ya que el ordenador trabaja a una velocidad mucho mayor que la placa controladora, por lo que es ésta la que nos define la velocidad máxima a la que se pueden comunicar. La placa arduino será la encargada de preguntar al PC por el estado de ejecución. Otro de los motivos es el hecho de que la comunicación se realiza a través de un puerto serie, lo que impide tener problemas de saturación, al tratarse de un medio exclusivo para nuestra aplicación. El paso de mensajes mediante 'Polling' es por tanto una solución adecuada.

3.3. Requisitos de funcionamiento del sistema domótico.

Para el funcionamiento de los elementos que forman parte del sistema domótico, se han de cumplir los siguientes requisitos de hardware y software:

- El ordenador ha de contar con un procesador dual-core a 2.66 GHz o superior, con 2 GB de RAM como mínimo y con el sistema operativo Windows 7, a fin de ser compatible con los drivers de Kinect for Windows 1.5. Las versiones anteriores o posteriores del software, por tanto, no serán compatibles con la solución domótica de nuestro proyecto.
- La placa controladora Arduino debe ser el modelo UNO o similar, con los drivers correspondientes. Nuestra solución aprovecha todas las salidas de esta placa. Si quisiéramos ampliar el número de motores o luces, deberíamos utilizar el modelo MEGA que es más potente y cuenta con un mayor número de entradas y salidas.
- La fuente de alimentación tiene que ser capaz de proporcionarnos 4,5-5 Voltios y con limitación de 2 Amperios, ya que las ferritas de los motores hacen que tengamos picos de corriente altos que podrían dañar la placa.
- El reconocimiento de voz se hará a través del paquete KinectSpeechLanguagePack_esES.

Capítulo 4

Diseño

En este capítulo nos disponemos a describir los módulos principales del sistema de domótica construido. En primer lugar, detallaremos el programa y funcionamiento del ordenador. Posteriormente se describirá el software utilizado en la placa Arduino así como su funcionamiento. Finalmente expondremos los componentes de hardware que forman parte del robot y las conexiones que hemos de realizar.

4.1. Diseño del programa del ordenador.

Nuestro programa del PC está realizado con lenguaje C#. Se trata de un lenguaje orientado a objetos que nos va a permitir estructurar el programa del PC en diferentes clases más sencillas. El código usado se basa en el creado por los autores Webb y Ashley [11] y en la página de recursos web para desarrolladores de Microsoft [7]. La interfaz de usuario la vamos a realizar en XAML. Este lenguaje está basado en XML y nos permite programar esta interfaz de forma sencilla.

A continuación, pasamos a describir las clases que conforman el programa del ordenador.

4.1.1. Conexión Kinect

Nuestro programa del ordenador, al iniciarse, lo primero que hará será ver el estado de la cámara Kinect. Ésta puede estar en diversos modos: iniciando, conectado, sin conectar a la red eléctrica, no preparada, dispositivo no original o desconectado. En caso de que la cámara esté conectada y preparada podremos trabajar con ella. Tendremos que inicializar el seguimiento de esqueleto (Skeletontracking) y el reconocimiento de voz en castellano (Speech). En los capítulos correspondientes de [11] o en la web [7] encontramos la explicación y ejemplos de uso de dichos objetos.

En caso de que la cámara no esté conectada, o esté en otro modo de funcionamiento, tendríamos que avisar al usuario para que conectase la cámara correctamente y continuar con el programa pero sin estas funcionalidades.

4.1.2. Reconocimiento de voz

El Kinect tiene dos modos de reconocimiento de voz. En uno de ellos, el reconocimiento se hace mediante transcripción de lo dictado por el usuario, mientras que en el otro, el reconocimiento se realiza por similitud entre las frases dictadas por el usuario y las que forman parte de un diccionario definido previamente. En nuestra aplicación, utilizaremos este segundo modo por ser mucho más preciso que el primero y más adecuado para nuestras necesidades, ya que necesitamos que reconozca solo determina-

das órdenes.

Para poner en funcionamiento el segundo modo de reconocimiento de voz del Kinect, definimos un diccionario con las palabras que el usuario puede pronunciar, y un diccionario de frases formadas con las palabras del diccionario anterior. En la tabla 4.1 presentamos un ejemplo de frases y las acciones que realizaría el Kinect a través de estas frases. En el apéndice A, manual de usuario, se define el diccionario de frases completo que utiliza nuestro programa. A través del paquete KinectSpeechLanguagePack_es-ES, se creará la fonética de cada sonido en castellano, y se generará un objeto con el que el Kinect establecerá comparaciones entre las frases emitidas por el usuario y las que contienen el diccionario de frases. Esta comparación permitirá que el programa ignore una palabra suelta o una frase incorrecta. En el caso de que pronunciemos correcta y claramente una frase, el Kinect la evaluará y si está por encima del umbral definido previamente, actuará en consecuencia.

Frases	Acción
Conecta arduino.	Conecta/ Desconecta el robot del PC.
Kinect enciende blanco ahora .	Se enciende el led de color blanco.
Kinect enciende rojo ahora.	Se enciende el led de color rojo.

Figura 4.1: Frases que admite.

A fin de evitar que el usuario le dicte órdenes al Kinect antes de que los archivos necesarios de reconocimiento de voz se hayan cargado correctamente, hemos añadido una barra de progreso que nos indicará, a través del desplazamiento de un cursor, si la carga se ha completado o no.

4.1.3. Reconocimiento de gestos

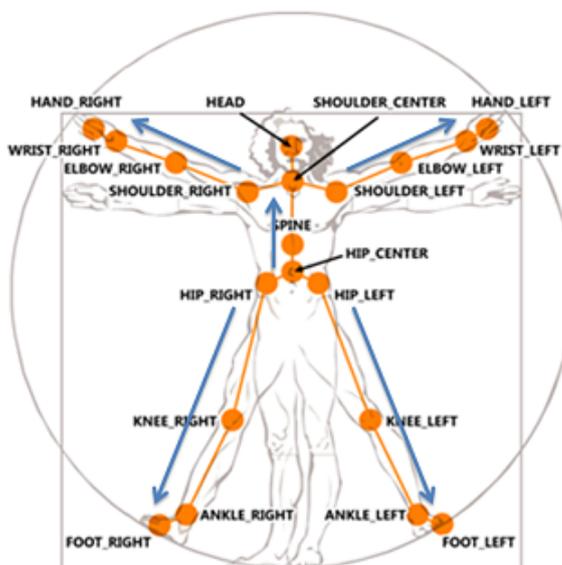


Figura 4.2: Puntos esqueleto [6]

Una vez inicializado el seguimiento de esqueleto, el Kinect nos va a enviar información de la posición de las articulaciones presentadas por el usuario. La cámara Kinect reconoce un total de 20 puntos del

cuerpo (figura 4.2). En este proyecto solo utilizaremos los puntos correspondientes a la posición de las extremidades superiores y cintura (manos, codos y hombros) (figura 4.3) para interactuar con el robot, o bien:

- Encendiendo y apagando luces del robot, o
- Controlando los motores del robot (articulación del robot).



Figura 4.3: Puntos de esqueleto que vamos a utilizar.

La interacción que permite encender y apagar leds se realiza con las manos. Si éstas se encuentran sobre cualquiera de los botones gráficos que representan los distintos colores de los leds del robot, éstos se encenderán (figura 4.4). Esta forma de interacción es incompatible con la de controlar motores.

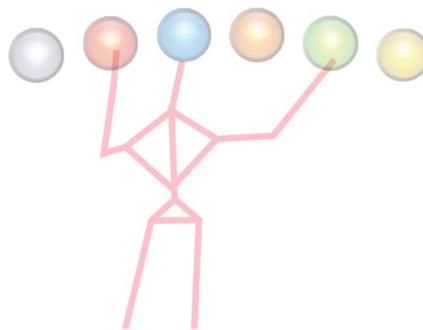


Figura 4.4: Botones gráficos luces.

La segunda forma de interacción permite el movimiento de los cinco motores que conforman el robot, y representan a cada una de las articulaciones de las extremidades superiores y cintura. Un motor para la cintura, otro para cada uno de los hombros y uno para cada codo. Dado que los motores utilizados tienen un recorrido de 0 a 180 grados y los codos habitualmente se doblan hacia arriba, obligaremos a que el movimiento de los motores de esta articulación sea únicamente hacia la cabeza. La articulación del robot la determinaremos a través de distintas fórmulas. Para obtener las fórmulas que representan a cada una de las posiciones de los motores, debemos calcular (figura 4.5):

- Los ángulos (α_i) que forman los antebrazos con respecto al cuerpo, a través de la posición de los codos y los hombros.
- Los ángulos (γ_i) que forman los antebrazos con respecto a los brazos, a través de la posición de los hombros, los codos y las manos.

- El ángulo (β) que forman los hombros con respecto a la cámara Kinect, a través de la posición de los hombros.

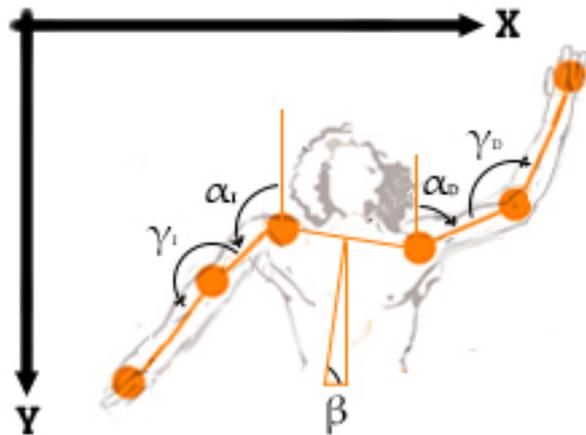


Figura 4.5: Ángulos α_i , (β_i) y (γ_i) calculados.

Al calcular los ángulos de los motores, hemos de tener en cuenta también, que los motores se desplazan en el sentido de las agujas del reloj (figura 4.6). El motor izquierdo se desplazará desde abajo hacia arriba, y el derecho se desplazará desde arriba hacia abajo. Por tanto, en el caso del motor izquierdo, tendremos que añadirle o quitarle 180° al ángulo calculado anteriormente.

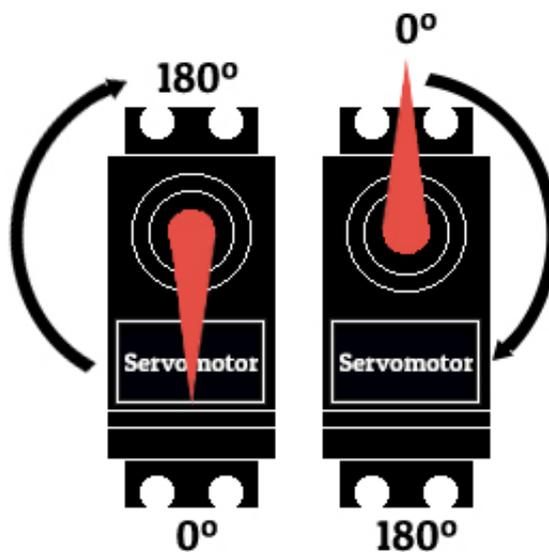


Figura 4.6: Servomotor izquierdo y derecho.

El recorrido de los motores nos obliga a limitar los ángulos anteriores, considerando los ángulos negativos como 0 grados, y los ángulos mayores de 180 grados como 180.

Tendremos en cuenta también que el programa será capaz de funcionar, o bien con el robot de cara al usuario, como si fuera un espejo, o con el robot de espaldas al usuario, por lo que cambiará la lateralidad de los motores, y en consecuencia las fórmulas para calcular los ángulos.

Por ejemplo, los ángulos correspondientes a los hombros derecho e izquierdo se calculan con las fórmulas 4.1 y 4.2.

$$\alpha_{HombroDerecho} = \arctan\left(\frac{CodoDerecho_Y - HombroDerecho_Y}{CodoDerecho_X - HombroDerecho_X}\right) \quad (4.1)$$

$$\alpha_{HombroIzquierdo} = 180 + \arctan\left(\frac{CodoIzquierdo_Y - HombroIzquierdo_Y}{CodoIzquierdo_X - HombroIzquierdo_X}\right) \quad (4.2)$$

Una vez calculados todos los ángulos, el valor de los mismos será enviado a la placa Arduino, a través del puerto serie, en contestación a las preguntas realizadas sobre los mismos por ésta. Una vez recibida esta información, el robot moverá la articulación correspondiente.

4.1.4. Puerto Serie

Entre el ordenador y la placa arduino hemos establecido una comunicación a través del puerto serie. El usuario tendrá que definir el número de puerto COM al que está conectado el robot. La transmisión será de 9600 bits por segundo, más que suficiente para nuestra aplicación.

Como el driver de Arduino utiliza un bit de paridad para la transmisión, de cada byte tendremos disponibles siete bits para nuestro propósito. Por este motivo el mayor número que podemos transmitir es el 127 ($2^7 - 1$). Por tanto, el valor de los ángulos necesarios para la articulación del robot (apartado anterior), no podrá ser superior a 127 grados. Para solucionar este inconveniente hemos optado por dividir entre dos los valores de los ángulos calculados.

Para evitar o detectar cualquier incidencia, como desconexión del cable, pérdida de información, ... , se ha incluido un mecanismo de retransmisión que efectúa nuevas transmisiones tras un período de inactividad. Si realizadas varias retransmisiones no se obtiene respuesta, el programa entenderá que el robot está inaccesible o se ha interrumpido la comunicación entre el ordenador y el robot.

4.1.5. Temporizador

Hemos definido un objeto *time* para poder manejar todos los eventos relacionados con el tiempo. Las funciones entre otras de este objeto *time* son las que presentamos seguidamente:

- Asegurarnos que todos los archivos se cargan adecuadamente. Para ello, mostraremos una imagen barra de progreso. Finalizada la carga aparecerá la interfaz y desaparecerá la imagen en el caso de que la cámara Kinect esté correctamente conectada.
- Controlar las retransmisiones de la información del puerto serie.
- Actualizar la hora y comprobar si hay alarmas activas.

4.1.6. Sonidos

En el sistema domótico del proyecto, hemos incluido una función que nos permite saber a través de sonidos grabados previamente la hora actual y la temperatura ambiente. Para la reproducción de los sonidos hemos utilizado un objeto *SoundPlayer* que los reproduce asincrónicamente, es decir, se reproducirán en otro hilo y no interferirán en el funcionamiento del resto del programa. Los sonidos se guardarán en formato .wav y estarán en la carpeta sonidos, por lo que si queremos cambiarlos simplemente los sobre-escribiremos.

Como la reproducción de la hora se realiza a través de varios audios seguidos, esto es, horas y minutos, hemos modificado el objeto *SoundPlayer* para que funcione de forma síncrona. Para esto, hemos creado una función que nos permite reproducir un sonido tras la finalización del otro. Para simplificar redondeamos los minutos de cinco en cinco.

4.2. Programa Arduino

La figura 3.3 comentada en el capítulo anterior nos muestra el esquema del programa que cargaremos en la placa Arduino. Cuando el programa esté en estado 'Conectado', mostrado a través de dos leds verdes, iremos haciendo 'Polling' al ordenador, preguntándole por (a) el estado de los otros seis leds de diferentes colores, conectados a las salidas digitales, (b) del estado de los cinco motores, conectados a las salidas de modulación de pulsos (PWM) o "salidas analógicas", y, por último, respondiéndole con el valor de la temperatura que hemos leído del sensor, y que ha sido calculada con la fórmula siguiente:

$$temperatura^{\circ}C = \frac{500 \times valorLectura}{1024} \quad (4.3)$$

En relación a la fórmula de la temperatura (4.3), hemos de tener en cuenta que la placa Arduino tiene una resolución de 10 bits, o 1024 valores, y la máxima tensión que puede medir son 5V. El sensor que vamos a utilizar nos devuelve 10mV por cada grado centígrado, por tanto, con 5V tendríamos 500°C, por lo que multiplicando por 500 y dividiendo entre 1024 se obtiene el valor real de la temperatura. Para obtener una medida mas fiable, haremos varias mediciones y promediaremos, de esta forma eliminamos el posible ruido del sensor.

En relación a las salidas de modulación de pulsos, cabe señalar que éstas cambian el ciclo de trabajo y con ellas podemos modificar el ángulo del servomotor (figura 4.7) a través de la librería 'Servo.h'. Esta librería nos calcula el ciclo de trabajo a partir del valor de un ángulo expresado en grados.

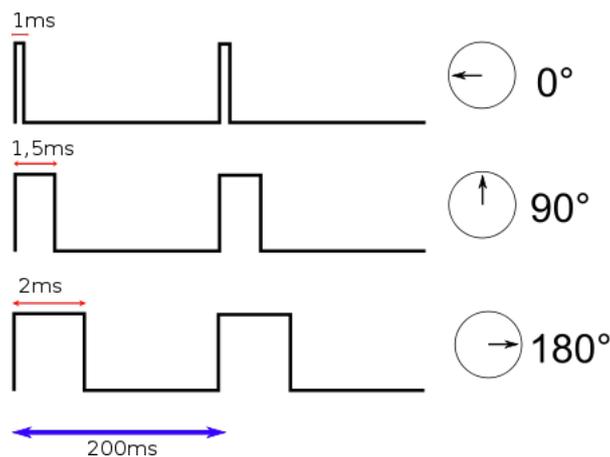


Figura 4.7: Ciclo de trabajo y ángulo de servomotores [12].

Igual que en el caso del programa del ordenador, ha sido necesario incluir un mecanismo de retransmisión de tramas para evitar o detectar cualquier incidencia como bloqueos del ordenador, errores de transmisión, ..., que nos permitirá desconectarnos a través de la orden recibida desde el ordenador o pasado un tiempo intentando transmitir.

4.3. Hardware Robot

Ahora describiremos los componentes y circuito eléctrico necesario y cómo estos se han cohesionado para obtener la estructura completa del robot.

4.3.1. Componentes Robot.

En la construcción del robot necesitamos los siguientes elementos de hardware:

- Una placa Arduino.
- Ocho leds de colores y resistencias.
- Cinco servomotores.
- Placa perforada, un pulsador y un sensor de temperatura.
- Fuente de alimentación.

La placa que hemos usado ha sido el modelo Arduino UNO rev3 con un procesador Atmega 16U2 a 16 Mhz. En la página web de Arduino [1] podemos ver el resto de características de esta placa . También podríamos haber optado por cualquier modelo con estas características.

De los ocho leds de colores usados, dos se han utilizado para los ojos, y los otros seis para mostrar que el robot es capaz de realizar diferentes actividades, simbolizadas por leds de diferentes colores. Para obtener seis colores diferentes, hemos utilizado cinco leds con los colores básicos: rojo, verde, azul, naranja y blanco. Para el sexto color hemos optado por un led bicolor, rojo y verde, que simulará el color amarillo. También se podría utilizar un led tricolor y modular la energía de cada led con las salidas PWM para obtener el color deseado. En este proyecto no hemos usado el led tricolor por tener todas las salidas de la placa Arduino en uso. Para que los leds funcionen correctamente hemos necesitado resistencias cuyos valores se han calculado con la siguiente fórmula:

$$R = \frac{V_{cc} - V_d}{I_d} \quad (4.4)$$

Siendo V_{cc} la tensión de alimentación, V_d la tensión de funcionamiento del led e I_d la corriente de funcionamiento del led. La tensión de nuestra placa es de 5V y la corriente típica de este tipo de leds es de unos 20mA. Además cada color de led tiene un valor típico de tensión de funcionamiento. Normalmente, los leds rojos, verdes y naranjas tienen una tensión típica de 2.1 Voltios aproximadamente, mientras que, los azules, los blancos y los bicolors habitualmente trabajan correctamente a unos 3.6 Voltios. Por lo tanto, necesitaremos resistencias de 150 Ohmios y de 82 Ohmios, respectivamente. Para los dos leds verdes en serie necesitaremos una resistencia de 47 Ohmios. En la tabla 4.8 podemos ver los datos de los leds y las resistencias.

Color Led	Voltaje(V)	Resistencia necesaria (Ω)
Rojo	2.1	150
Verde	2.1	150
Naranja	2.1	150
Amarillo	2.1	150
Azul	3.6	82
Blanco	3.6	82
Verdes serie	4.2	47
Bicolor verde-rojo	3.5	82

Figura 4.8: Tabla leds y resistencias

Los cinco servomotores que hemos empleado en la construcción del robot, son de tipo estándar de la marca Hitec modelo HS-322hd. Dichos motores tienen un recorrido de 180 grados y soportan 3.0 kg/cm, más que suficiente para esta aplicación. Los servomotores de esta marca tienen tres cables, rojo, negro y amarillo. El rojo lo hemos de conectar a VCC, el negro a tierra, y el amarillo es el que tenemos que conectar a la salida PWM correspondiente.

En la placa perforada con separación una décima de pulgada se han soldado mediante unos conectores macho con separación 2.54 mm, el pulsador 6x6x2.5, que permite resetear el programa de forma sencilla, el sensor de temperatura y los cables correspondientes al resto de componentes. El sensor de temperatura que hemos utilizado corresponde al modelo LM35DZ de National Semiconductor. El tipo de encapsulado del sensor y sus conexiones se muestran en la figura 4.9.



Figura 4.9: Sensor LM35DZ [10]

En la figura 4.10 se muestra la conexión de esta placa perforada con la placa arduino. En el centro de la placa perforada se puede observar el pulsador de reset y el sensor de temperatura. En la parte superior las soldaduras de los cables de los leds y los motores.

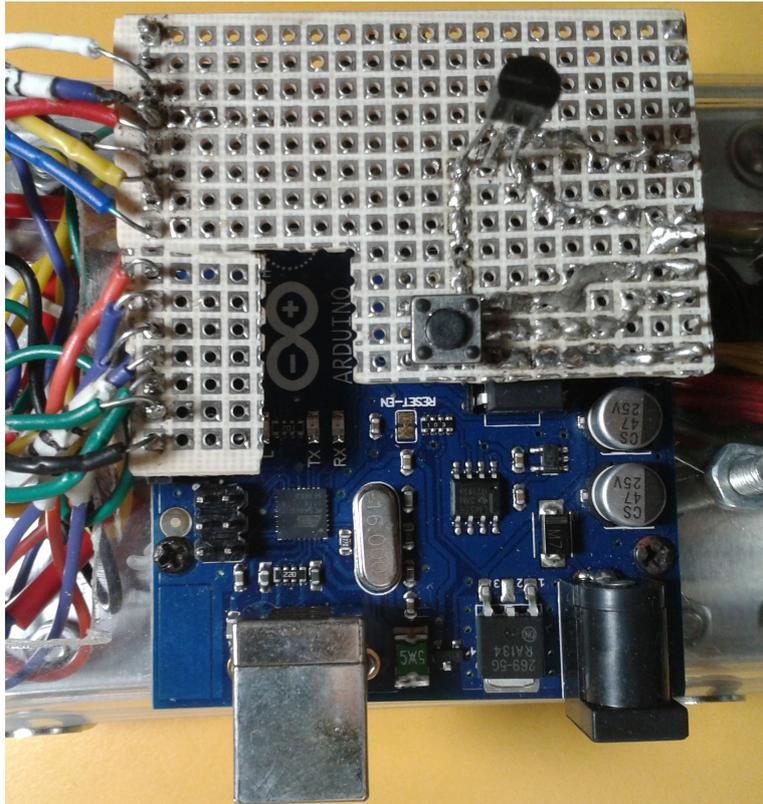


Figura 4.10: Arduino y componentes.

La fuente de alimentación que hemos utilizado para los servomotores con el objetivo de controlar los picos de corriente que producen las ferritas, tal como se ha descrito en el apartado 3.3 del capítulo anterior, ha sido el modelo MW2122A, que nos proporcionará una tensión de 4.5 Voltios y una corriente máxima de 2 Amperios.

4.3.2. Circuito eléctrico.

El circuito eléctrico de la placa Arduino (figura 4.11), está compuesto por catorce salidas, ocho digitales y seis analógicas, y de seis entradas analógicas. Las salidas D0 y la D1 se utilizan para comunicación con el puerto serie, de ahí, que no podamos utilizarlas para conectar otros elementos. Las salidas PWM de la placa que utilizaremos están en los pines D3, D5, D6, D9, D10, y D11, y en cinco de ellas conectaremos los cinco servomotores, que deberán estar también conectados a la fuente de alimentación y a la tierra de la placa. Los leds que simulan las distintas actividades que puede realizar el robot estarán conectados a los pines D2, D4, D6, D7, D8 y D12. Al pin D13 conectaremos en serie los dos leds que muestran si el Arduino está conectado o no. Al estar conectada en paralelo a esta salida una resistencia y un led soldados a la placa Arduino, nos podrá proporcionar una corriente máxima menor. En el extremo inferior de la placa, tenemos la zona de entradas analógicas. En el puerto A0 hemos conectado el sensor de temperatura. Éste también debe ser conectado a tierra y a Vcc. El pulsador lo hemos conectado a GND y a la entrada de Reset, que es activa a nivel bajo.

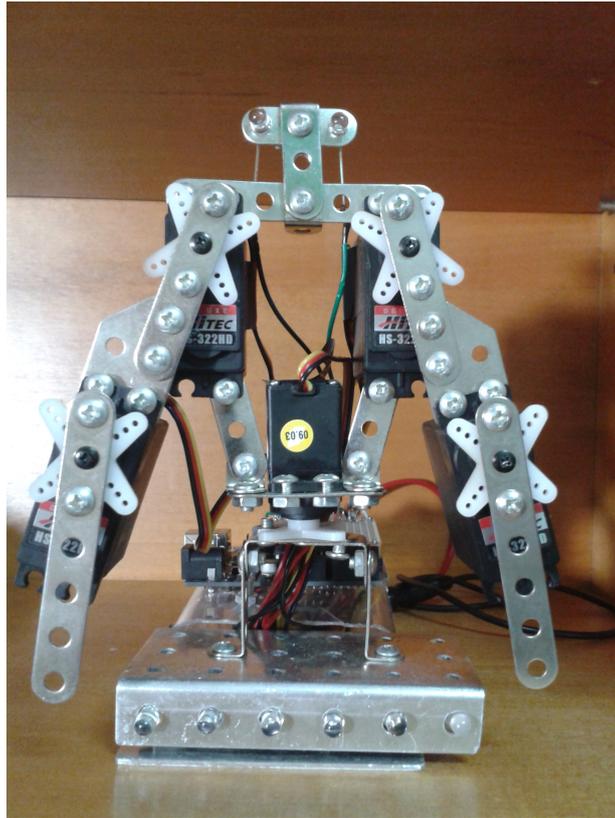


Figura 4.12: Robot de frente.

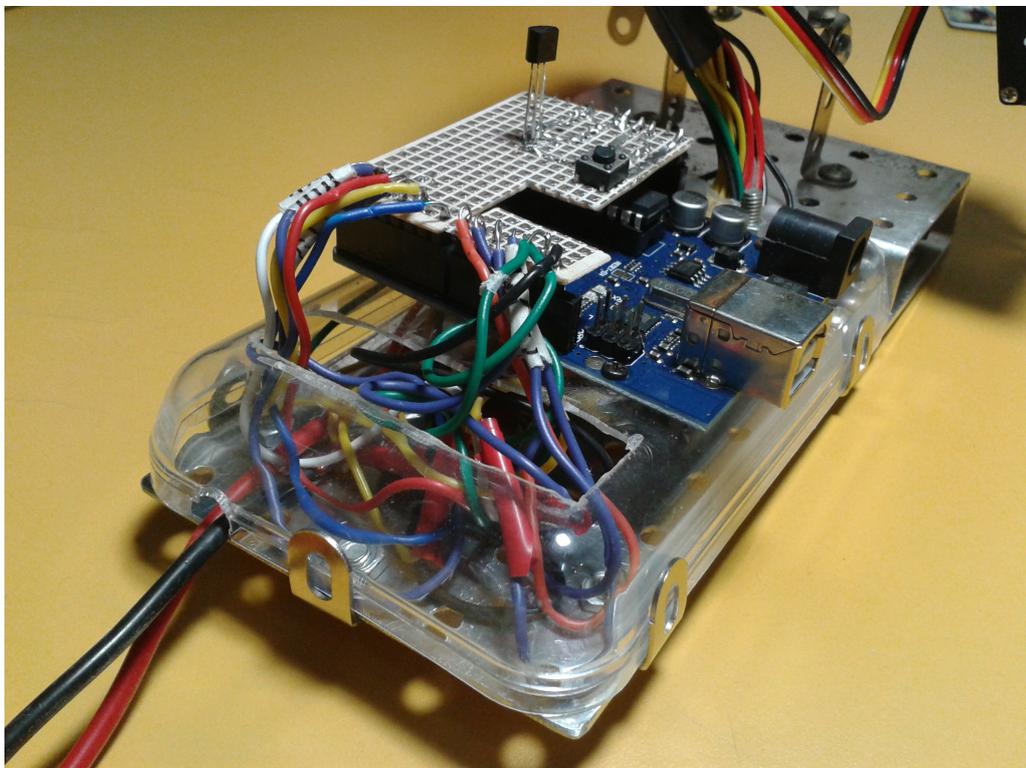


Figura 4.13: Carcasa transparente.

4.4. Interfaz Usuario

Se ha optado por una interfaz muy sencilla, donde al ejecutar el programa aparece un menú de carga. Pasado el tiempo, se muestra en la pantalla del PC la imagen de la figura 4.14, en la que encontramos los siguientes elementos.

- Un interruptor con el título 'debug' para testear el funcionamiento interno de la aplicación del sistema domótico. Al activarlo aparecerán en la parte derecha de la pantalla datos de funcionamiento del programa, como el ángulo de las articulaciones o dónde se encuentra el error en caso de que el programa no funcione correctamente. También nos permite activar y desactivar manualmente los leds o los motores.
- Un rótulo en la parte superior derecha, donde se muestra la temperatura leída por el sensor del robot, la fecha y la hora actual.
- Botones gráficos que se encuentran en la parte central con los que el usuario podrá interactuar colocando sus manos sobre el centro de cada uno de ellos. El programa tiene en cuenta ambas manos por lo que se pueden encender simultáneamente dos luces.
- Un rótulo o un icono en la parte derecha muestra el estado de la alarma y la hora en la que sonará ésta, en caso de haberla activado.
- Un interruptor deslizable, que también se encuentra a la derecha, con el que podremos subir o bajar el motor del Kinect.
- Un botón de conectar o desconectar a la izquierda y un cuadro de diálogo en el que podremos seleccionar el puerto COM al que está conectada la placa Arduino. En la misma línea, aparecen datos sobre las articulaciones.
- Iconos de actividad (figura 4.15), en la parte inferior de la imagen, que permiten saber el estado del robot. El primero se activará al conectar la placa. Los seis siguientes mostrarán qué color ha encendido el usuario y el último si los motores están conectados.

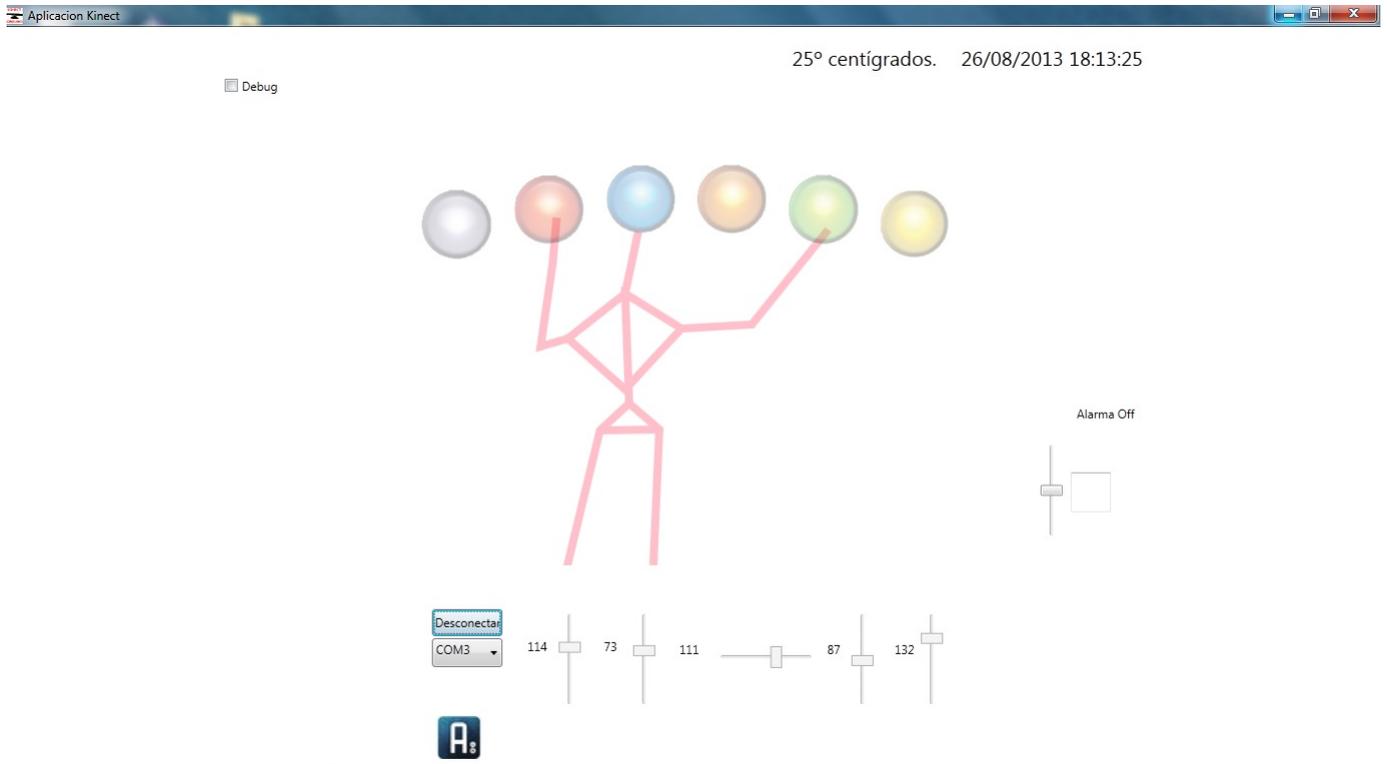


Figura 4.14: Interfaz Usuario.



Figura 4.15: Iconos de actividad.

Capítulo 5

Resultados

5.1. Introducción.

En este capítulo ejemplificaremos el funcionamiento de nuestro proyecto. Además comprobaremos si hemos cumplido con los objetivos descritos en el capítulo 1.

5.2. Ejemplo de aplicación.

En esta sección vamos a mostrar el funcionamiento de nuestro proyecto de domótica, a través de un robot, que mediante el uso de diferentes sensores y actuadores simulará el funcionamiento de los elementos de una casa, como persianas, puertas, luces...

Al iniciar el programa esperamos unos segundos para que los archivos necesarios se carguen. Posteriormente, si el Kinect se ha conectado correctamente al puerto USB y a la corriente eléctrica, desaparece de la pantalla del ordenador la imagen central correspondiente a la foto de la cámara. En caso contrario, se nos estará indicando que ha ocurrido algún error con la cámara Kinect, aunque podremos seguir utilizando la aplicación sin las funcionalidades que nos aporta dicha cámara. En la figura 4.14 podemos observar como ha quedado el programa al iniciarlo correctamente. Debemos seleccionar el puerto COM al que está conectada la placa Arduino, normalmente, estará en el puerto COM3 o COM4. No obstante, es posible que se encuentre en otro puerto. En caso de no saber en qué puerto está conectada deberíamos hacer uso del Panel de control y Administrador de Dispositivos. En el panel de control buscaremos la placa Arduino UNO. Seleccionado el puerto correcto, le damos al botón 'Conectar', o damos la orden correspondiente (ver Apéndice A). Como todo funciona correctamente observamos que se han encendido los dos leds verdes del robot. Si ocurriese algún error nos saltará un mensaje advirtiéndonoslo. El estado inicial del programa es con todos los elementos apagados.

Una vez completada la fase de iniciación del sistema, ya podremos ejecutar el mismo. Se puede realizar de dos modos, por reconocimiento de voz o de gestos. En primer lugar, vamos a comprobar que funciona correctamente la ejecución del programa por activación por gestos. Nos situamos a una distancia entre 2 y 3 metros de la cámara Kinect, distancia adecuada para que el programa nos reconozca y proyecte nuestra silueta en el centro de la pantalla del ordenador. Aunque el programa reconoce dos usuarios, se recomienda trabajar con un único usuario dado que es posible que el sistema no funcione adecuadamente al aparecer dos en pantalla. Una vez que el programa ha reconocido nuestra silueta, vamos a interactuar con el robot de dos maneras:

- Gestos: colocando las manos sobre el centro de los botones superiores que se encuentran en la

pantalla del ordenador (Figura 4.14). Estamos colocando las manos en uno, dos botones... Observamos que se van encendiendo los leds rojo, azul, ... del robot que como hemos indicado representan el encendido de luces, calefacción del hogar, etc. (Figura 5.1).

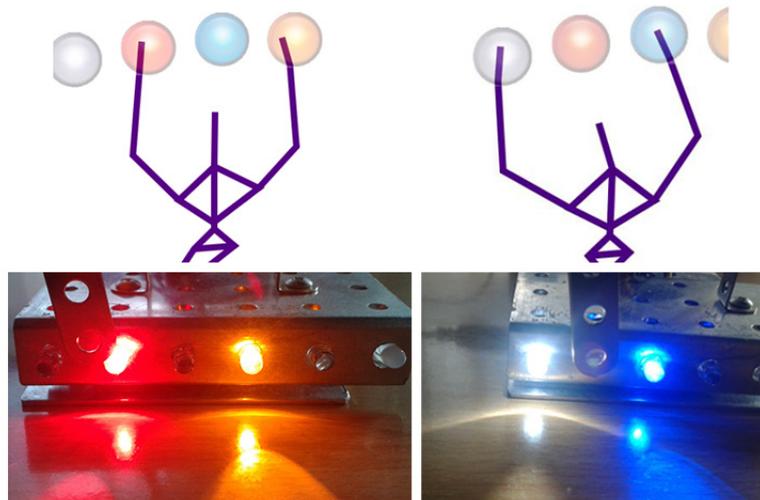


Figura 5.1: Interfaz de gestos.

- Corporal: moviendo las articulaciones superiores (brazos y cintura), una vez encendidos los motores. El movimiento de cada una de las articulaciones a través de los servomotores simulan cómo este sistema domótico proyectado me permite abrir las puertas y/o las ventanas de una casa. Estoy moviendo el brazo derecho y observo que los motores correspondientes cambian también (figura 5.2). En caso de que los servomotores no se muevan, hemos de comprobar que el robot está correctamente conectado a la fuente de alimentación.

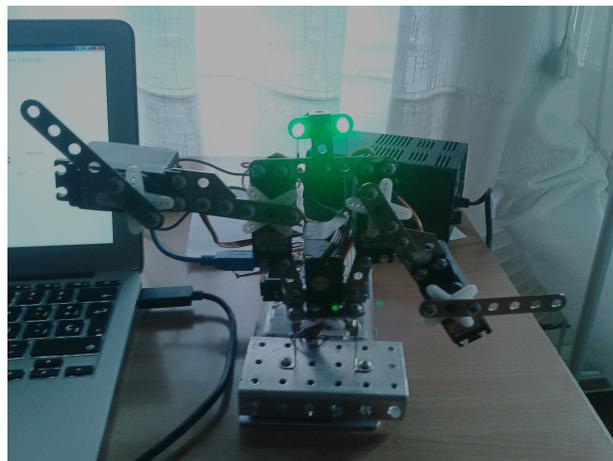


Figura 5.2: Motores en movimiento.

En la ejecución por reconocimiento de voz, el usuario podrá ejecutar las mismas funciones que en la activación por gestos, encender y apagar los leds y los servomotores, pero en este caso dando las órdenes pertinentes. Por ejemplo, “Kinect enciende rojo ahora”, observamos como se enciende el led rojo (Figura 5.3). También el usuario puede (a) conocer la hora, la temperatura o conectar y desconectar la alarma y (b) apagar la cámara, el robot y finalizar el programa, diciendo las órdenes adecuadas de forma

clara. Los datos horarios, de temperatura y alarma también se pueden obtener a través de los datos que muestran los sensores en la pantalla (Figura 4.14).

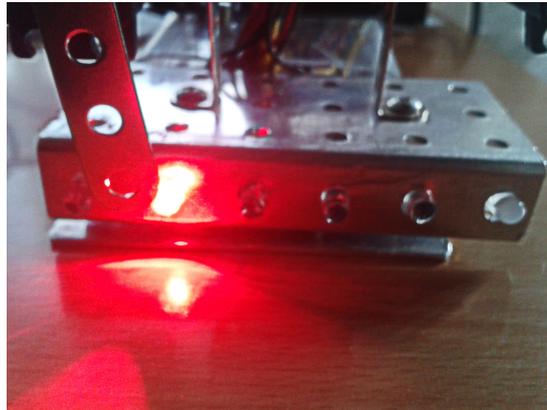


Figura 5.3: Led rojo encendido.

Los dos modos de ejecución son incompatibles. Si alguno de los elementos no funcionase correctamente podremos activar el modo 'debug' con el botón situado en la zona superior izquierda de la imagen proyectada en el ordenador. En la parte derecha de ésta aparecerán unos botones para testear cada uno de los elementos. De esta forma podremos detectar el problema.

5.3. Presupuesto.

La tabla 5.4 nos muestra el presupuesto del proyecto de domótica presentado. El precio de algunos elementos como placas de aluminio y perforadas y caja de metacrilato, es estimado, dado que estos han sido reciclados.

Producto	Precio unidad(€)	Cantidad	Precio Total(€)
Kinect	94.90	1	94.9
Arduino UNO	15	1	15
Servomotores Hitec	11	5	55
Leds	0.12	7	0.84
Leds bicolor rojo-verde	1.10	1	1.10
Resistencias 1/4W	0.10	7	0.70
Sensor LM35DZ	2.90	1	2.90
Pulsador	0.10	1	0.10
Cables y estaño	1	1	1
Placas perforada	2.91	1	2.91
Placas aluminio	6	1	6
Caja metacrilato	1	1	1
Conectores macho 2.54	1	1	1
Fuente de alimentación	20	1	20
TOTAL			202.45

Figura 5.4: Presupuesto del proyecto

El presupuesto final del proyecto ha sido de 202.45€. De este presupuesto, 94,9€, corresponde a la cámara Kinect. El robot, sin contar la cámara, ha costado 107,55€, y más de la mitad corresponde al coste de los 5 servomotores, 55€. Podríamos haber rebajado el precio de los servomotores comprándolos más sencillos pero hemos optado por estos por ser más fiables y silenciosos. El resto de elementos supone un gasto total de 52.55€, de los cuáles 15€ corresponde a la placa Arduino, 7€ a la estimación del material reciclado, y la fuente de alimentación tiene un coste de 20€.

5.4. Cumplimiento de objetivos.

En esta sección vamos a comprobar si hemos cumplido con los objetivos del proyecto. El objetivo principal del proyecto, plantear un sistema de domótica de bajo coste, se ha cumplido ya que éste se ha realizado con poco más de 200€ tal como analizamos en la sección 5.3 de este capítulo. Además se ha dotado al robot de diferentes sensores y actuadores con los que podemos interaccionar. Por lo tanto, podemos decir que se han cumplido los tres objetivos propuestos para este proyecto.

Capítulo 6

Conclusiones

6.1. Desarrollo del proyecto.

En esta sección vamos a describir los inconvenientes que nos han surgido al realizar nuestro proyecto así como las soluciones que hemos dado a estos. Estas dificultades las hemos encontrado principalmente en la segunda fase del proyecto. El proyecto ha tenido una duración aproximada de un año. Se ha realizado en dos fases. La primera fase corresponde a documentación y aprendizaje de programación de la cámara Kinect, donde se decidió utilizar el SDK oficial de Kinect for Windows y el lenguaje de programación c#. Esta fase fue la de mayor duración. La segunda fase, a la que nos referiremos en los párrafos siguientes, corresponde al diseño del proyecto.

La fase de diseño se realizó en tres etapas. La primera etapa tuvo como objetivo sincronizar el programa del ordenador y el de la placa Arduino a fin de que se enviaran mensajes correctamente. Para alcanzar el objetivo propuesto tuvimos que documentarnos sobre el uso del puerto serie del ordenador y tomar decisiones sobre el mismo. Se optó por un tipo de paso de mensajes mediante “polling” al ser el que mejores resultados nos ofreció, por utilizar una tasa de bits determinada, bits de paridad, retransmisiones, etc.

En la segunda etapa, una vez superadas las dificultades de la primera y comprobado que la sincronización de los programas del ordenador y la placa Arduino era adecuada, se introdujo el Kinect como fuente de datos. Esto supuso un reto ya que el software de desarrollo para el Kinect estaba en constante actualización. En el momento de iniciar esta etapa la versión del Kinect era la “beta del SDK”. No habíamos terminado el desarrollo del programa de la fuente de datos, cuando esta versión se actualizó. Finalmente y tras varias reestructuraciones del código se optó por utilizar la versión 1.5 de los drivers de Kinect for Windows. Con estos drivers se hizo el programa definitivo. Posteriormente, aparecieron nuevas versiones del SDK que han resultado incompatibles con programa del proyecto, por lo que hemos impuesto como restricción del mismo, para garantizar su correcto funcionamiento, el uso de la versión 1.5.

En la tercera etapa se testeó el programa para corregir todos los fallos que nos iban apareciendo.

Por último, se pasó al montaje de los robots. Se montaron dos robots similares, con una duración aproximada de un mes cada uno. En el segundo robot se corrigieron algunos fallos de diseño o de montaje que se detectaron en la primera versión. Al segundo robot se le añadió un pulsador para poder reiniciar de forma fácil el programa. También se cambiaron los servomotores por unos superiores, se corrigió algún valor de resistencia incorrecto y se cambió el modelo de placa Arduino por una más potente.

6.2. Conclusiones trabajo realizado.

En el desarrollo de este proyecto se han tratado muchos aspectos vistos en la carrera de Ingeniería de Telecomunicaciones. Se han utilizado diversos lenguajes de programación, como C# y XAML para la creación del programa de ordenador, y Processing para programar la placa microcontroladora Arduino. Se han visto conceptos de electrónica para el diseño de los circuitos electrónicos, y ha hecho falta calcular los valores de las resistencia para el correcto funcionamiento de los leds. También se han revisado conceptos de telemática para el paso de mensajes entre los diferentes elementos. Por tanto, ha sido un proyecto donde se han tratado diversos temas de la carrera.

6.3. Futuras ampliaciones.

En este proyecto hemos utilizado todas las salidas de la placa microcontroladora Arduino. Podríamos ampliar el numero de dispositivos conectados, como motores, leds, ..., utilizando una placa Arduino modelo mega, que cuenta con un mayor número de salidas. Si quisiéramos en vez de utilizar un robot, aplicar los conceptos vistos para el control domótico de un hogar podríamos utilizar algún dispositivo como el 'MICASAVERDE VERALITE HOME CONTROLLER' [3], que nos permitiría acoplar diferentes accesorios para automatizar el hogar. Además se podría utilizar un dispositivo de bajo coste y bajo gasto energético, como una Raspberry Pi, para sustituir al PC y poder manejar el sistema a través de Internet.

Manual de usuario

6.4. Manual usuario.

En esta sección presentamos de forma breve unas orientaciones de funcionamiento del sistema de domótica propuesto a fin de que este pueda ser manejado por cualquier usuario.

- **Instalación del software necesario.** Lo primero que se debe hacer para poder ejecutar la aplicación es instalar el software necesario en nuestro ordenador. Necesitamos un ordenador con sistema operativo Windows 7 cuyo procesador sea dual-core a 2,66 GHz o superior, y con al menos 2 GB de memoria RAM. Instalaremos los drivers de Kinect for Windows versión 1.5 y el paquete de reconocimiento de voz en castellano KinectSpeechLanguagePack_esES. Una vez tenemos todo el software podemos pasar a ejecutar el programa del proyecto.

- **Conexión de todos los elementos del sistema.**

Antes de iniciar la aplicación se deberán conectar correctamente todos los elementos del sistema. Se conectará la cámara Kinect al puerto USB y a la red eléctrica, y el robot a otro puerto USB y a la fuente de alimentación. Realizada la conexión, se debe seleccionar el puerto donde tenemos conectada la placa Arduino y darle al botón de conectar (Figura 6.1). Pasados unos segundos, se encenderán los dos leds verdes del robot. En caso de no saber el puerto al que se encuentra conectado el robot podemos consultarlo en Panel de control y Administrador de Dispositivos.



Figura 6.1: Conectar arduino.

- **Interacción con el robot.** La interacción con el robot permite encender los leds o los servomotores. La interacción gestual se realiza situando las manos sobre los botones superiores de la pantalla, que activarán los leds correspondientes del robot. El encendido de los servomotores se realiza a través del dictado de órdenes que el software interpretará. Una vez encendidos los servomotores, podemos interactuar con el robot corporalmente moviendo los brazos. El encendido de leds y la consulta de la hora y temperatura actual también se pueden realizar mediante el dictado de órdenes. En la tabla 6.3 se muestran las órdenes necesarias para el encendido de servomotores y leds, así como la consulta horaria y de temperatura y el cerrado de la aplicación. Las combinaciones que admite la orden "Despertador 'hora' 'minutos'" se muestran en la tabla 6.4 donde los minutos se presentan de cinco en cinco, y que medianoche son las 24 horas. La orden de 'Cierra aplicación' es necesario decirla de forma muy clara para evitar posibles cierres involuntarios.

Al encender los diferentes elementos como las luces o los motores nos aparecerán los iconos que se ven en la figura 6.2. El primer icono simboliza que la placa arduino está conectada. Los seis siguientes

iconos con forma de círculos significan que los leds de cada color están encendidos. El último icono es el que indica si los motores están activos.



Figura 6.2: Iconos programa.

Orden	Acción
Conecta arduino.	Conecta/ Desconecta el robot del PC.
Kinect enciende blanco ahora .	Se enciende el led de color blanco.
Kinect enciende rojo ahora.	Se enciende el led de color rojo.
Kinect enciende azul ahora.	Se enciende el led de color azul.
Kinect enciende naranja ahora.	Se enciende el led de color naranja.
Kinect enciende verde ahora.	Se enciende el led de color verde.
Kinect enciende amarillo ahora.	Se enciende el led de color amarillo.
Kinect enciende luces ahora.	Se encienden todos los leds.
Kinect apaga luces ahora.	Se apagan todos los leds.
Kinect enciende motor ahora.	Se enciende los servomotores.
Qué hora es.	El programa dice la hora actual de cinco minutos en cinco minutos.
Qué temperatura.	El programa dice la temperatura leída por la placa Arduino.
Despertador 'hora' y 'minutos'.	Se activa la alarma a la hora indicada. Ver tabla 6.4
Despertador desconectar	Se desactiva la alarma.
Cierra aplicación	Se cierra el programa.

Figura 6.3: Ordenes que admite.

En la parte superior izquierda del programa podemos ver la temperatura leída por el sensor de la placa Arduino y fecha y la hora actuales. Podremos saberlas con las órdenes vistas en la tabla 6.3.

25° centígrados. 28/08/2013 18:35:11

Figura 6.5: Temperatura, fecha y hora.

En la figura 6.6 podemos ver el estado inicial de la alarma desconectada y cuando la conectamos. Aparece un icono de una campana y la hora a la que va a sonar.

Alarma Off  18:40

Figura 6.6: Alarma apagada y encendida.

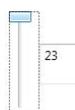


Figura 6.7: Subir o bajar Kinect.

Hora	Minutos
Una.	En punto
Dos.	Y cinco
Tres.	Y diez
Cuatro.	Y cuarto
Cinco.	Y veinte
Seis.	y veinticinco
Siete.	Y media
Ocho.	Y treinta y cinco
Nueve.	Y cuarenta
Diez.	Y cuarenta y cinco
Once.	Y cincuenta
Doce.	Y cincuenta y cinco
Trece.	
Catorce.	
Quince.	
Diez y seis.	
Diez y siete.	
Diez y ocho.	
Diez y nueve.	
Veinte	
Veintiuna	
Veintidós	
Veintitrés	
Veinticuatro	

Figura 6.4: Horas y minutos.

6.5. Preguntas frecuentes.

La foto del Kinect no desaparece al terminar el tiempo de carga. Solución: Comprueba que el Kinect está conectado correctamente al ordenador, a la red eléctrica y que se tienen instalados los drivers adecuados.

El programa se cierra al abrirlo. Solución: Comprueba que se tienen instalados los drivers de Kinect for Windows versión 1.5.

Al darle a conectar el robot da error. Solución: Comprueba que el número de puerto COM es el correcto y que están instalados los drivers de Arduino.

El Kinect no me reconoce la silueta adecuadamente. Solución: Sitúate a una distancia de entre 1.5 metros y 2.5 metros de la cámara. Comprueba no reconoce varios esqueletos.

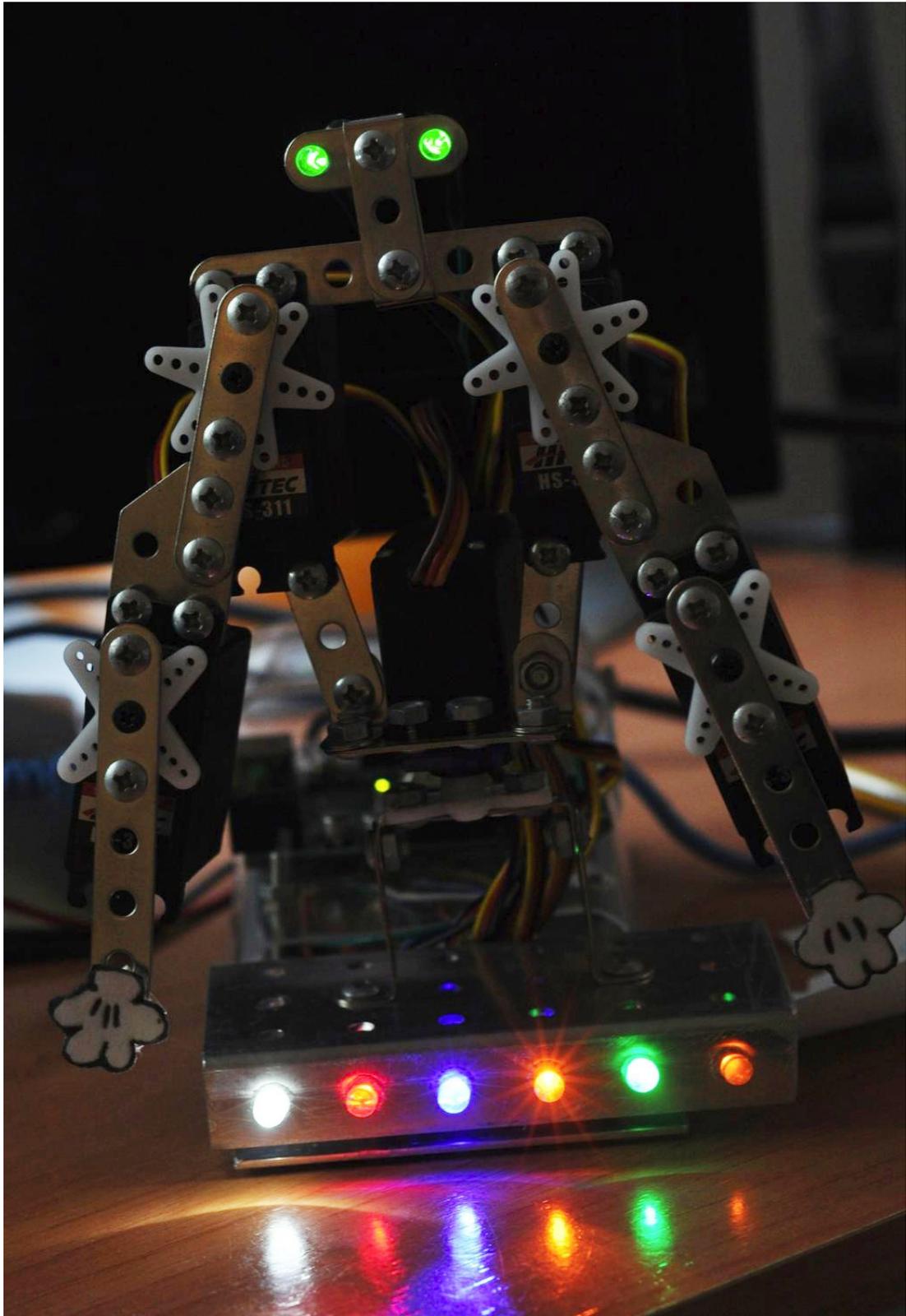


Figura 6.8: Robot en funcionamiento.

Bibliografía

- [1] Arduino. Arduino uno. <http://arduino.cc/en/Main/arduinoBoardUno>.
- [2] Belking. Smart home automation from belking. <http://www.belkin.com/us/wemo>.
- [3] dkcautomation. Micasaverde veralite home controller. <http://www.dkcautomation.com/products/mi-casa-verde-veralite-home-controller>.
- [4] Google. Android@home. <http://www.intomobile.com/2011/05/10/android-home-green-robots-running-house-becoming-reality/>.
- [5] Doug Gregory. (dkc automation) voice-controlled home automation with android (tasker, autovoice, and more... <http://youtu.be/RjTj0ymhbBw>.
- [6] MSDN Library. Skeleton position and tracking state. <http://msdn.microsoft.com/en-us/library/jj131025.aspx>.
- [7] Microsoft. Resources and documentation. <http://www.microsoft.com/en-us/kinectforwindows/develop/learn.aspx>.
- [8] Juan Mol. Control domótico con android. <http://rsppi.blogspot.com.es/2013/05/control-domotico-con-android.html>.
- [9] Juan Mol. Domótica en raspberrypi (parte 1). <http://rsppi.blogspot.com.es/2012/06/domotica-en-raspberrypi-parte-1.html>.
- [10] National Semiconductor. Precision centigrade temperature sensor. http://www.datasheetcatalog.com/datasheets_pdf/L/M/3/5/LM35D.shtml.
- [11] Jarrett Webb and James Ashley. *Beginning Kinect Programming with the Microsoft Kinect SDK*. Apress, 2012.
- [12] Wikipedia. Servomotor de modelismo. http://es.wikipedia.org/wiki/Servomotor_de_modelismo.
- [13] Wikipedia. Domotica, 9 2013. <http://es.wikipedia.org/wiki/Domotica>.