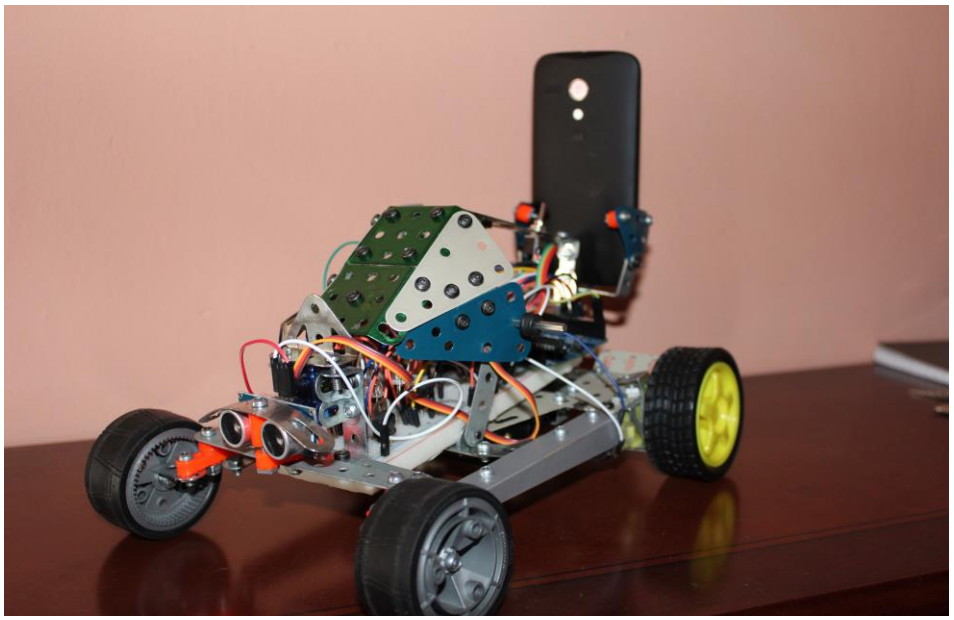
1/1/2015

Proyecto robot-RC con retransmisión de video en tiempo real.



Proyecto Sistemas Electrónicos Digitales (SED)
3ºTelecomunicaciones

Abstract

Elaboración de un Robot controlado mediante Bluetooth desde un ordenador, mediante una aplicación elaborada en lenguaje Java. Así mismo, esta aplicación recibe datos de video vía WIFI desde un dispositivo móvil colocado en el robot. Mostrando el video en tiempo real desde esta aplicación.

Manuel Moya Ferrer mmoyaferrer@gmail.com
Ismael Yeste Espín iye1994@gmail.com

Proyecto Robot-RC con retransmisión de video en tiempo real - SED

Introducción

La motivación por la que hemos realizado este proyecto, ha sido la atracción hacia la tecnología y las diferentes opciones que aporta. Desde que comenzamos la carrera de Ingeniería en Telecomunicaciones, se aprenden y dominan diferentes formas en las que la tecnología ha evolucionado, tales como microprocesadores en electrónica, montaje de circuitos, programación a un mayor nivel como Java, redes, tratamiento de señales, etc. Así como otros muchos aspectos que no incluimos en la realización de este proyecto.

El hecho de conocer todas estas ventajas de la tecnología, combinar programación de alto nivel como java y electrónica, así como el gusto por crear objetos inteligentes desde cero, es lo que nos ha llevado a la realización de este proyecto.

Para la realización del robot, hemos utilizado un juego de piezas de mecano, para ahorrar en gastos y poder construirlo desde cero. Para el control de éste, se ha utilizado "Arduino", un microcontrolador programable en C, con el cual el robot recibirá las señales de control de movimiento mediante un módulo Bluetooth. Este microcontrolador tendrá a su disposición motores de corriente continua, así como servos, leds, etc. Sin embargo, éste no influye en la transmisión de video, puesto que para tener una mayor comodidad y ahorro en gastos, hemos utilizado como transmisor de video un smartphone que se sitúa en el robot. El cual actúa como cámara IP, recibiendo los datos de Introducción a la realización del robot.

Para la realización de la aplicación en el ordenador, hemos utilizado lenguaje Java, realizando una aplicación sencilla. Hemos desarrollado un menú de control de acciones del robot, tales como avanzar hacia adelante, atrás, a los lados, encender las luces o apagarlas, así como un modo automático en el que el robot se mueve bajo su propio control. En la aplicación disponemos de una pequeña pantalla en la que podemos ver lo que el robot ve desde su posición, de manera que podemos controlarlo sin necesidad de verlo físicamente.

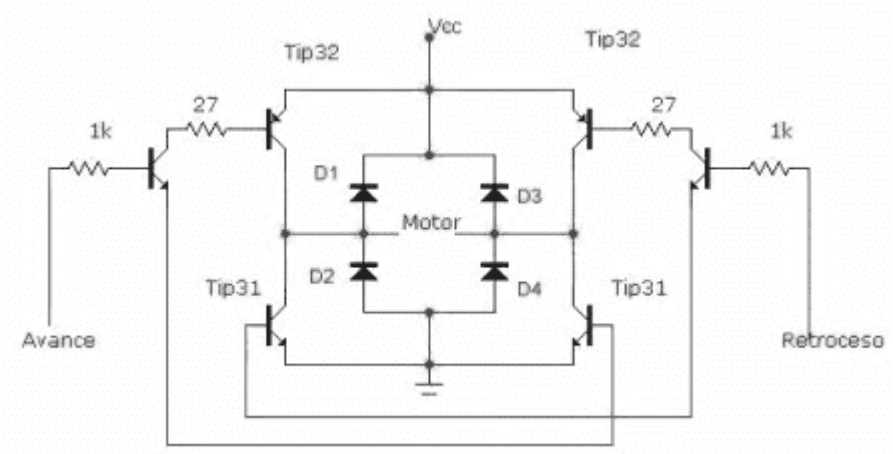
ÍNDICE:

1 Puente en H - Funcionamiento Motores

Necesidad de circuito puente en H: El primer pensamiento a la hora de hacer el movimiento de un robot con Arduino y mediante motores dc, es el de conectar los motores directamente a las patillas del micro-controlador Arduino, y de esta manera ya tenemos establecido el movimiento adelante-atrás. Esto no es posible debido a que las patillas de un micro-controlador, típicamente tienen un límite de intensidad a suministrar, como es en nuestro caso 40mA con Arduino UNO. Este límite de intensidad hace que los motores no funcionen, puesto que necesitan una intensidad mucho más alta, por ello realizamos este circuito. Obtendremos una intensidad en los motores desde 90 a 200 mA, dependiendo de la tensión que establezcamos como fuente.

Este circuito nos permitirá hacer circular la intensidad en los motores mediante dos pines de nuestro Arduino, cuando uno de los pines se active, una parte de los BJT conducirá de manera que circulará la intensidad por el motor. Con un pin en alta y otro en baja, los motores funcionará en un sentido y a la inversa.

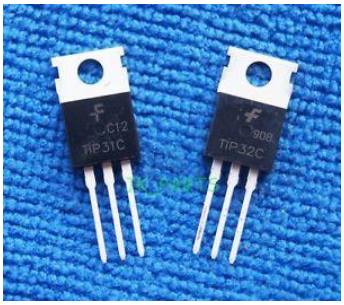
El diseño del puente en H que hemos construido es el siguiente (existen otros más sencillos, pero este es el que mejor funcionamiento nos ha mostrado):



1.1 Elementos de nuestro puente en H:

Para los **transistores** N, hemos utilizado 4xTIP 31C, tal como vemos en el esquema, así como hemos utilizado 2xTIP 32C como transistores de tipo P. Hemos de usar transistores de esta categoría puesto que otro tipo de BJT podría no soportar las intensidades o voltaje que se manejan y quemarse (en el "datasheet" de éstos podemos ver como cumplen las condiciones).

Estos son de un tamaño mayor que los BJT cotidianos para pequeños circuitos de amplificación, así mismo incluyen un agujero para posibilidad de anclarlos a un disipador de calor, en nuestro caso no es necesario ya que no alcanzarán altas temperaturas con las intensidades que circularán en el circuito.



Como **diodos**, hemos utilizado diodos led de diferentes colores, los cuales nos sirven y no es necesario comprar diodos específicos. Además, cuando utilicemos el robot podremos observar los picos de tensión que absorben, protegiendo así nuestros transistores.

Como **resistencias**, hemos usado las 2 resistencias de 1kΩ que observamos en el circuito, y en lugar de las de 27Ω, hemos utilizado de 100Ω, lo cual es prácticamente indiferente.

Como **fuente** que alimenta a nuestro puente en H, utilizamos 6 pilas en serie de tipo AA. Estas nos proporcionarán una tensión de 1.5*6=9Voltios, la cual teniendo en cuenta los motores que vamos a describir a continuación, es más que suficiente.

Motores: Hemos usado dos motores de corriente continua, estos conectados en paralelo a los terminales del esquema donde observamos escrito "motor".

El hecho de usar dos nos ayuda a la fuerza que ejercerá para el movimiento del robot, puesto que usando uno sólo, no hay fuerza suficiente para un buen movimiento del robot.

Los motores que hemos utilizado son los siguientes:

*Podemos ver que incluyen las ruedas, lo que nos facilita el montaje, así mismo podemos separar ambas partes.

2 Dirección

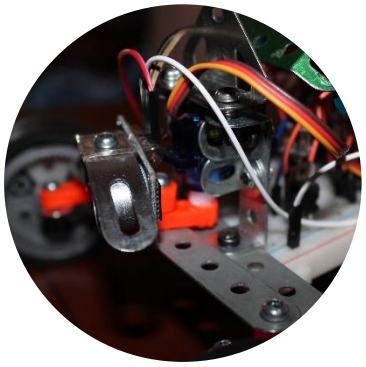
Para el movimiento hacia los lados, hemos de instalar algún dispositivo que haga girar las ruedas. En este proyecto, usamos un micro-servo 9g, que nos proporcionará una fuerza de 1,6 kg.

Mediante este dispositivo creamos un mecanismo que gira las ruedas delanteras.

Podemos verlo en la siguiente imagen:

2.1 Funcionamiento

Este servo moverá en sentido horizontal una barra de mecano, la cual estará unida a las ruedas y estas girarán. Lo mostramos en una imagen:



El servo gira la pieza naranja, haciendo girar la barra a la que están ancladas las ruedas.

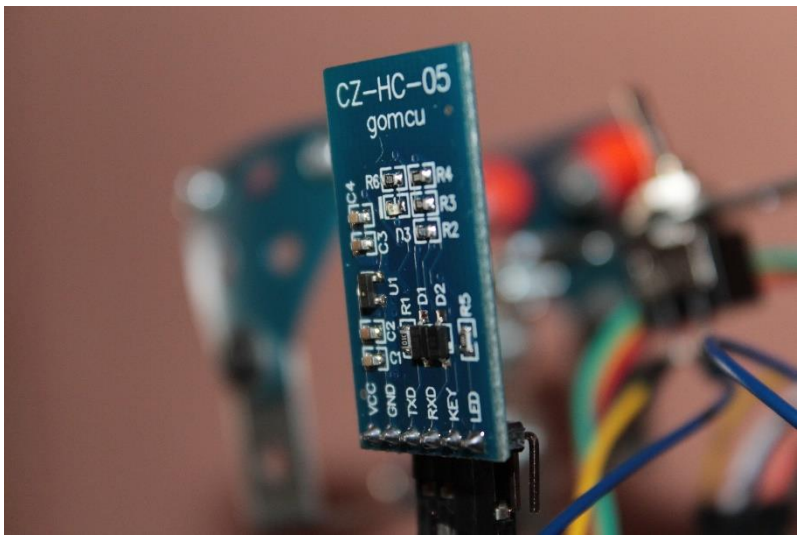
2.2 Conexiones con Arduino

Como vemos en la imagen, el Servo dispone de 3 patillas, las cuales serán:

- Cable rojo: Patilla de voltaje de Arduino (5 V)
- Cable marrón: Conexión a tierra (GND)
- Cable naranja: Conexión a pin de Arduino, mediante el cual le mandaremos el ángulo de giro que ha de establecer, para así girar la barra y consecuentemente ambas ruedas delanteras.

La programación de este dispositivo la veremos en el apartado Programación de Arduino.

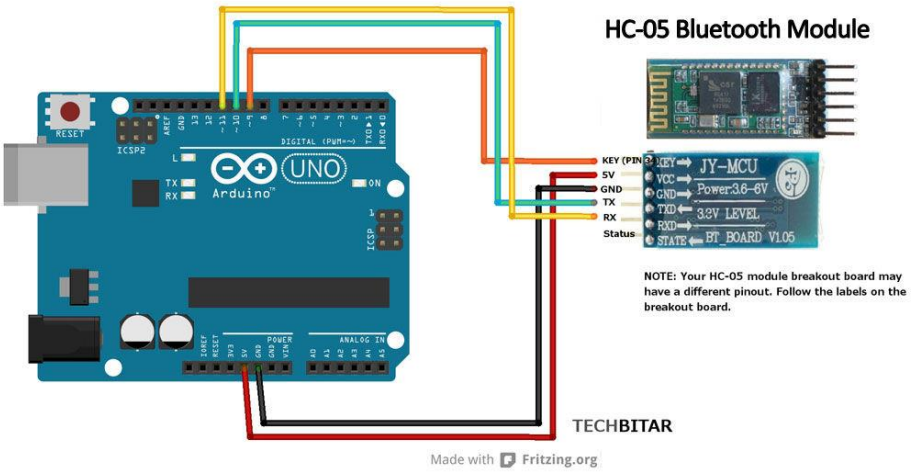
3 Módulo Bluetooth



Para la comunicación con el Robot usamos el módulo Bluetooth HC-05. La configuración para la comunicación se resume en lo siguiente:

1º Asociar el dispositivo a nuestro ordenador:

Como primer paso, conectamos el Bluetooth a Arduino, de la siguiente manera:



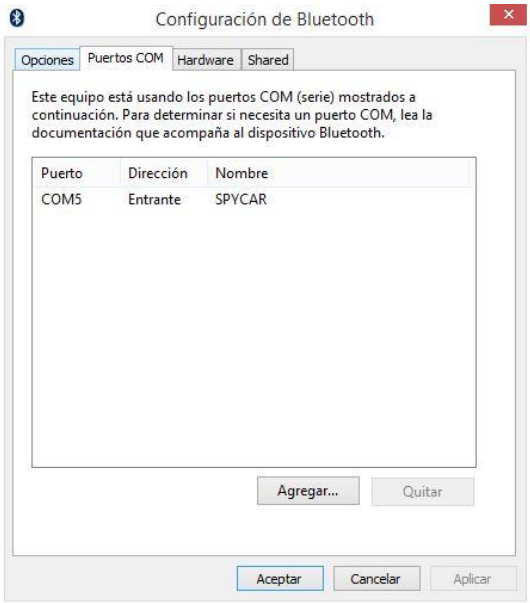
*Puesto que no vamos a configurar el nombre del Bluetooth ni su contraseña, no conectamos la patilla Key, en este dibujo se ve conectada puesto que se procede a su configuración, nosotros solo queremos activar el dispositivo para sincronizarlo con el ordenador.

Buscamos el dispositivo con nuestro ordenador mediante “dispositivos Bluetooth”, y lo asociamos.

2ºAsociar el dispositivo a un puerto COM (será un puerto COM virtual, ya que la no existe un cable físico que conecte el dispositivo al ordenador):

En nuestro ordenador, accedemos a :

“Configuración de Bluetooth” → “Puertos Com” → Agregar → Dirección Saliente(El equipo inicia la conexión mediante la aplicación)



3º Ya tenemos nuestro circuito para la comunicación establecido, en la aplicación tendremos que decir mediante que puerto COM se establece la comunicación, lo cual veremos en “Configuración de Bluetooth” una vez que hayamos realizado los dos primeros pasos.

4 Programación de Arduino

La programación viene totalmente explicada en el mismo sketch del programa, por lo que podemos consultarla ahí, aún asi pondremos unas capturas del código para una visión de su funcionamiento sin recurrir al archivo:

```
//El funcionamiento básico del programa es el siguiente:
//Tenemos dos posibilidades, tener el modo autopiloto activado o no, a partir de esto se formará nuestro programa, de la siguiente manera:
//
//1º Si el modo autopiloto está desactivado (autopiloto==false)  :  Lo que haremos será recibir nuestros comandos bluetooth, los cuales nos indicarán
//   la acción a realizar; hacia delante, hacia la izquierda(los cuales se realizan mediante la funcion mover, que explicaremos a continuación),...() O así mismo el comando para activar el modo autopiloto, por lo que autopiloto
//   sería un valor true y por tanto pasaremos a la otra opción.
//
//2º Si el modo autopiloto está activado (autopiloto == true)  :  El programa activará las funciones de nuestro sensor de ultrasonidos, el cual nos dará la distancia
//   que hay entre el obstáculo frontal y el coche mediante la funcion medir_distancia que explicaremos a continuación. Si esta distancia es menor a 30 cm, el coche rectificará
//   su movimiento echando marcha atrás y torciendo. En caso de que la distancia sea mayor, no habrá obstaculos cercanos y por tanto el robot podrá continuar su camino sin girar.
//
//   Así mismo, en este modo seguiremos leyendo datos recibidos por Bluetooth, puesto que nos pueden mandar el comando para desactivar el modo autopiloto.

//Función mover: Mediante esta función, controlaremos el movimiento del coche tanto en direccion delantera-trasera como hacia los lados.
//               Los argumentos que recibe como entrada son un byte de dirección(comprendido en el rango 0-180 (grados) los cuales nos permite el Servo),
//               Y un dato entero que será velocidad(1 si va hacia delante, -1 hacia atras, en otro caso los motores pararán.)

//Función medir_distancia: El sensor de ultrasonidos mandará una serie de pulsos que posteriormente volverá a recibir en su patilla echo, mediante el tiempo que ha pasado entre
//                         que los pulsos van y vuelven(dato que nos suministra este sensor), podemos calcular el espacio que hay , puesto que sabemos la velocidad del sonido, por
//                         lo que la salida de esta función serán los centímetros de distancia que separan el frontal del coche con un obstáculo. Para mas información acerca de este
//                         sensor, podemos ver su datasheet en internet.


//*Los datos que nos envia el ordenador mediante Bluetooth, los recibimos por el puerto serial del Arduino (patillas Rx y Tx)
```

Figura 1: Explicación programa

```
void mover(byte dir, int v){

  direccion=dir;
  velocidad=v;
  if(v>0){
    digitalWrite(pin_adelante,HIGH);
    digitalWrite(pin_atras,LOW);
  }
  else if(v<0){
    digitalWrite(pin_adelante,LOW);
    digitalWrite(pin_atras,HIGH);
  }
  else{
    digitalWrite(pin_adelante,LOW);
    digitalWrite(pin_atras,LOW);
  }
  miServo.write(dir);
}

int medir_distancia(){

  digitalWrite(pin_trig,LOW);
  delayMicroseconds(5);
  digitalWrite(pin_trig,HIGH);
  delayMicroseconds(10);
  duracion=pulseIn(pin_echo,HIGH);  //LA SEÑAL RETORNA AL SENSOR

  cm=int (0.034*(duracion/2)) ;  //Velocidad = cm/us  Tiempo = us

  if(cm==0 || cm<0){
  cm=100;
  }

  return cm;
}
```

Figura 2: Funciones mover y medir_distancia

```
if(!autopiloto){

  switch(comando){

    case 'f':                        //Comando avanzar
      velocidad=1;
      mover(direccion, velocidad);
      break;

    case 'b':                        //Comando retroceder
      velocidad=-1;
      mover(direccion, velocidad);
      break;

    case 'l':                        //Comando girar a la izquierda
    direccion=20;
      mover(direccion, velocidad);
      break;

    case 'r':                        //Comando girar a la derecha
    direccion=160;
      mover(direccion, velocidad);
      break;

    case 's':                        //Se ha soltado la tecla W ó S, por lo que paramos el motor.
      velocidad=0;
      mover(direccion, velocidad);
      break;

    case 'c':                       //Se ha soltado la tecla de dirección (A ó D), por lo que el servo toma el valor 90°(no hay giro)
      direccion=90;
      mover(direccion, velocidad);
      break;

    case 'a':                       //Comando para activar el modo autopiloto.
      autopiloto=true;
      break;
  }
}
```

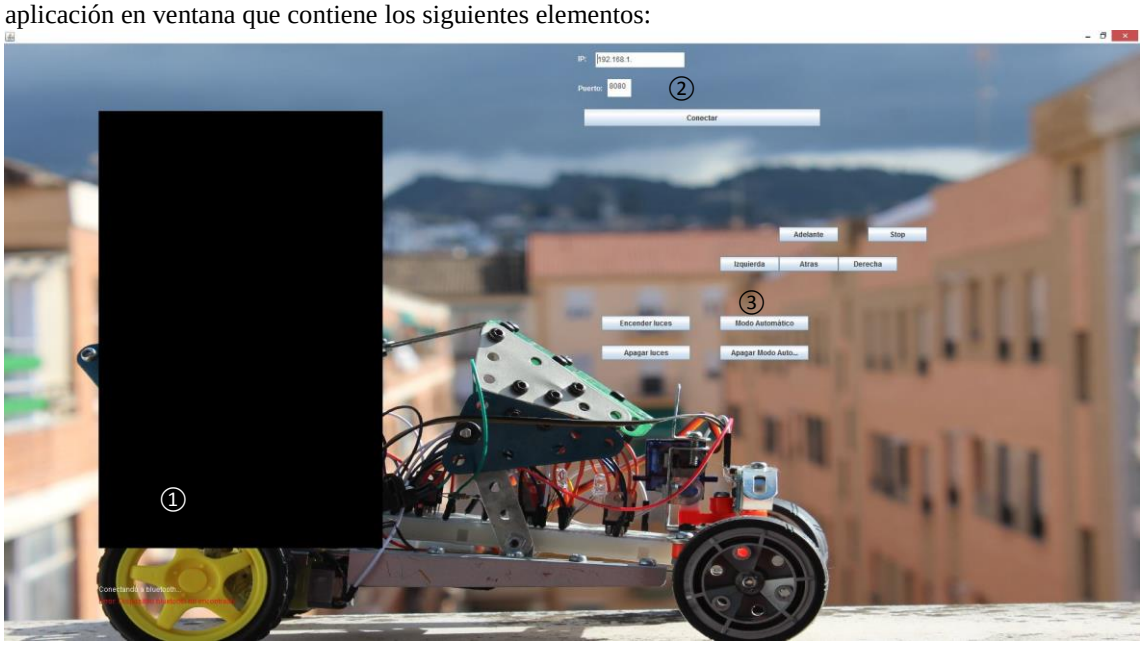
Figura 3: Modo Manual

```
                    else{     //MODO AUTOPILOTO

                        while(autopiloto){

                             distancia=medir_distancia();

                             if(distancia<30){
                                  mover(160, -1);
                                  delay(2000);
                                  mover(20, 1);
                                  delay(500);
                             }
                             if(distancia>30){
                                mover(90,1);
                             }

                            if(Serial.available()){
                               comando=Serial.read();

                               if(comando=='m'){       //Comando para desactivar el modo autopiloto.
                               autopiloto=false;
                               direccion=160;
                               velocidad=0;
                               mover(direccion,velocidad);
                               }
                               else{
                               autopiloto=true;
                               }
                             }
                        }

                    }
```

Figura 4: Modo autopiloto

5 Programación Aplicación Java

En este apartado vamos a describir la aplicación utilizada para el control del coche. Se trata de una aplicación en ventana que contiene los siguientes elementos:



- 1.-Cuadro de la imagen: muestra la imagen de video y mensajes de información, como errores de conexión.
- 2.- Formulario para conectar video: pide la ip de un teléfono móvil que utilizamos para transmitir video y el puerto que utiliza la aplicación.
- 3.- Controles de movimiento y luces.
-

Al iniciar la aplicación se conecta automáticamente al dispositivo bluetooth, siempre que hayamos asociado el dispositivo con el puerto serie COM5. Una vez conectado permite el control a través de los botones de movimiento y las teclas W (adelante) A (atrás) S (izquierda) y D (derecha). Para los controles de luces es necesario establecer conexión con el teléfono, ya que utilizaremos el flash de este.

5.1 Receptor de video

Para la transmisión de video se hará uso la aplicación 'IP Webcam' disponible para Android. Esta aplicación emite en streaming la imagen de la cámara del móvil. Además crea un servidor HTTP para poder controlar algunas funciones como el flash, el zoom, etc.
Para ver el video emitido utilizamos la clase MjpegRunner de Java, publicada en:

http://thistleshrub.net/Joomla/index.php?option=com_content&view=article&id=115:displaying-streamed-mjpeg-in-java&catid=43:robotics&Itemid=64.

A continuación se muestra el código fuente de la clase.

```
import java.awt.Color;
import java.awt.image.BufferedImage;
import java.io.ByteArrayInputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.StringWriter;
import java.net.SocketTimeoutException;
import java.net.URL;
import java.net.URLConnection;

import javax.imageio.ImageIO;
```

```
/**
 * Given an extended JPanel and URL read and create BufferedImages to be displayed from a MJPEG stream
 * @author shrub34 Copyright 2012
 * Free for reuse, just please give me a credit if it is for a redistributed package
 */
public class MjpegRunner implements Runnable
{
	private static final String CONTENT_LENGTH = "Content-Length: ";
	private static final String CONTENT_TYPE = "Content-Type: image/jpeg";
	private MJpegViewer viewer;
	private InputStream urlStream;
	private StringWriter stringWriter;
	private boolean processing = true;
	private URLConnection urlConn;


	public MjpegRunner(MJpegViewer viewer, URL url) throws IOException
	{
		this.viewer = viewer;
		urlConn = url.openConnection();
		// change the timeout to taste, I like 1 second
		urlConn.setReadTimeout(500);
		urlConn.connect();
		urlStream = urlConn.getInputStream();
		stringWriter = new StringWriter(128);
	}

	/**
	 * Stop the loop, and allow it to clean up
	 */
	public synchronized void stop()
	{
		processing = false;
	}
	/**
	 * Keeps running while process() returns true
	 *
	 * Each loop asks for the next JPEG image and then sends it to our JPanel to draw
	 * @see java.lang.Runnable#run()
	 */
	@Override
	public void run()
	{

		while(processing)
		{
			try
			{
				byte[] imageBytes = retrieveNextImage();
				ByteArrayInputStream bais = new ByteArrayInputStream(imageBytes);

				BufferedImage image = ImageIO.read(bais);
				viewer.setBufferedImage(image);
				viewer.repaint();
			}catch(SocketTimeoutException ste){
				viewer.mensaje("Mala conexion", Color.red);
			}catch(IOException e){
			}
		}
		// close streams
		try
		{
			urlStream.close();
		}catch(IOException ioe){
			System.err.println("Failed to close the stream: " + ioe);
		}
	}
```

```
	/**
	 * Using the <i>urlStream</i> get the next JPEG image as a byte[]
	 * @return byte[] of the JPEG
	 * @throws IOException
	 */

	private byte[] retrieveNextImage() throws IOException
	{
		boolean haveHeader = false;
		int currByte = -1;

		String header = null;
		// build headers
		// the DCS-930L stops it's headers
		while((currByte = urlStream.read()) > -1 && !haveHeader)
		{
				stringWriter.write(currByte);

				String tempString = stringWriter.toString();
				int indexOf = tempString.indexOf(CONTENT_LENGTH);
				int last=tempString.charAt(tempString.length()-1);

				if(indexOf > 0 && last=='\n')
				{
					haveHeader = true;
					header = tempString;
				}
			}


			// 255 indicates the start of the jpeg image
			while((urlStream.read()) != 255)
			{
				// just skip extras
			}

			// rest is the buffer
			int contentLength = contentLength(header);
			byte[] imageBytes = new byte[contentLength + 1];
			// since we ate the original 255 , shove it back in
			imageBytes[0] = (byte)255;
			int offset = 1;
	    int numRead = 0;
	    while (offset < imageBytes.length
	        && (numRead=urlStream.read(imageBytes, offset, imageBytes.length-offset)) >= 0) {
	       offset += numRead;
	    }

				stringWriter = new StringWriter(128);

			return imageBytes;
	}
	// dirty but it works content-length parsing
	private static int contentLength(String header)
	{
		int indexOfContentLength = header.indexOf(CONTENT_LENGTH);
		int valueStartPos = indexOfContentLength + CONTENT_LENGTH.length();
		int indexOfEOL = header.indexOf('\n', indexOfContentLength);

		String lengthValStr = header.substring(valueStartPos, indexOfEOL).trim();

		int retValue = Integer.parseInt(lengthValStr);

		return retValue;
	}
}
```

Esta clase implementa un hilo de ejecución para leer el video. Al instanciar la clase se establece una conexión con el servidor http correspondiente a la URL pasada como argumento.

El hilo de ejecución implementado en el método run lee los bytes del stream de forma secuencial, identifica las cabeceras que indican el tipo de contenido y el tamaño de cada imagen, y obtiene el array de bytes de la imagen (llamada a la función 'retrieveNextImage()'). A partir de este array crea una imagen y se la pasa a un objeto MjpegViewer (heredado de JComponent) a través del procedimiento setBufferedImage().

Para utilizar la clase hay que crear un objeto MjpegRunner, pasándole como argumentos un objeto URL y otro de tipo MJpegViewer (cuya clase debemos implementar). A partir de este objeto se crea un Thread (hilo), el cual se puede ejecutar en paralelo a nuestro programa.

Como se ha dicho, debemos implementar la clase MJpegViewer, que es la encargada de mostrar la imagen. Nuestra implementación de la clase MjpegViewer es la siguiente:

```
public class MJpegViewer extends JComponent{

	AffineTransform rotacion=new AffineTransform();//Se usa para rotar la imagen 90º, ya que colocaremos el teléfono en posición vertical.
	Dimension dim;//Dimension de la ventana
	BufferedImage imagen;//Frame actual


	MJpegViewer(JFrame padre){
		dim=padre.getSize();
		rotacion.rotate(Math.PI/2) ;//Rota la imagen 90º
		rotacion.translate(0, -dim.width/4);//Al rotar con centro en origen la
                                                    //imagen quedaría fuera de la pantalla.
                                                    //Esta translación lo corrige
	}

	@Override
	/*
	 * Cada vez que se actualiza la imagen con repaint()
	 * se hace una llamada a este procedimiento.
	 * En el se aplica la transformación "rotacion" y se
	 * dibuja la imagen. Si la imagen no ha sido inicializada
	 * dibuja un rectangulo negro.
	 */
	public void paint(Graphics g) {
		Rectangle bounds=getBounds();
		((Graphics2D)g).setTransform(rotacion);

		if(dim.width/2!=bounds.getWidth()){
			bounds.width=dim.width/4;
			setBounds(bounds);
		}
		if(dim.height/2!=bounds.getHeight()){
			bounds.height=dim.width/3;
			setBounds(bounds);
		}

		if(imagen==null){
			g.setColor(Color.BLACK);
			g.fillRect(0, 0, bounds.height, bounds.width);
		}
		else{
			g.drawImage(imagen, 0, 0, bounds.height, bounds.width,null);
		}
	}
```

```
	/*
	 * Este método sirve para que un objeto MjpegRunner
	 * pase una imagen leida.
	 */
	void setBufferedImage(BufferedImage bf){
		imagen=bf;
	}


	/*
	 * Cuando se llama al siguiente método
	 * se muestra en el centro de la imagen
	 * el mensage: "Mala conexion".
	 */
	void conexionMala(){
		Graphics g=getGraphics();
		g.setColor(Color.red);
		Rectangle r=this.getBounds();
		g.drawString("Mala conexión", r.width/2, r.height/2);
	}
}
```

5.2 Conexión bluetooth

Para la comunicación con el módulo bluetooth CZ-HC-05, se asocia un pueto COM (serie). Este puerto se comunica con el programa a través de la biblioteca jssc, que podemos encontrar en:

http://search.maven.org/#search|ga|1|jssc.

Esta biblioteca contiene clases y funciones que permiten comunicación con distintos tipos de puertos serie como COM y USB. En nuestro caso usaremos un puerto COM y lo que nos interesa de esta librería es la clase SerialPort y sus funciones openPort, setParams y writeBytes.

SerialPort: implementa un puerto serie. Para inicializarla basta con pasarle un String con el nombre del puerto.

SerialPort.openPort(): abre el puerto. Si está ocupado o no existe lanza la excepción SerialPortException.

SerialPort.setParams(int baudrate, int databits, int stopbits, int parity): define los parámetros tasa de baudios, número de bits de datos, número de bits de parada, y tipo de paridad.

SerialPort.writeBytes(byte[] b): envía por el puerto la secuencia de bytes b.

5.3 Clase principal

A continuación se describen cada uno de los elementos de la clase principal.

En primer lugar se importan las bibliotecas:

```
import java.awt.*;
import java.net.URL;
import java.net.URLConnection;
import jssc.SerialPort;
import jssc.SerialPortException;
import javax.swing.*;
import java.awt.event.*;
```

A continuación declaramos las variables de clase:

```
static MJpegViewer visor;//Objeto heredado de JComponent usado para
                         //mostrar la imagen.
static Thread lecturaVideo;//Hilo de ejecucion obtenido a partir de
                           //un objeto de la clase MjpegRunner
static JFrame ventana=new JFrame();
```

```
static Dimension tam=Toolkit.getDefaultToolkit().getScreenSize();//Tamaño de la pantalla
static SerialPort serialPort = new SerialPort("COM5");//Puerto serie
                                                        //emulado, asociado al módulo bluetooth
static  JTextPane ip=new JTextPane(),puerto=new JTextPane();//Entradas del formulario que se utilizará para
conectar con el servidor http.
```

La función principal es:

```
public static void main(String args[]) throws Exception{

                        ////////////////////////////////////////////////////////////////////

                        //Definicion de los elementos de la ventana
                        Container cBotones=new Container();
                        Container filas[]=new Container[20];


                        JLabel lip=new JLabel("IP:"),
                                        lpuerto=new JLabel("Puerto:");

                        JButton conectar=new JButton("Conectar"),
                                                controles =new JButton("Controles"),
                                                adelante=new JButton("Adelante"),
                                                atras=new JButton("Atras"),
                                                stop=new JButton("Stop") ,
                derecha=new JButton("Derecha"),
                izquierda=new JButton("Izquierda"),
                encenderluces=new JButton("Encender luces"),
                apagarluces=new JButton("Apagar luces"),
                modoautomatico=new JButton("Modo Automático"),
                apagarmodoautomatico=new JButton("Apagar Modo Automático");

                        ////////////////////////////////////////////////////////////////////


                        ////////////////////////////////////////////////////////////////////

                        //Se añaden los elementos a la ventana
                        ventana.setSize(tam.width, tam.height);
        ventana.setContentPane(new ImagenFondo());
        ventana.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                        ventana.setLayout(new BoxLayout(ventana.getContentPane(), BoxLayout.X_AXIS));
                        visor=new MJpegViewer(tam.width/12, tam.height/2-2*tam.width/9, tam.width/3,
4*tam.width/9);
                        ventana.add(visor);
                        ventana.add(cBotones);
                        ventana.addMouseListener(new MouseAdapter(){
                                @Override
                                        public void mouseClicked(MouseEvent e) {
                                                ventana.requestFocusInWindow();
                                        }
                        });

                        ventana.addKeyListener(TeclasPulsadas);
                        ////////////////////////////////////////////////////////////////////


                        ////////////////////////////////////////////////////////////////////
                        //Se ajustan los atributos de los botones
                        cBotones.setLayout(new BoxLayout(cBotones, BoxLayout.Y_AXIS));
```

```
for(int i=0;i<20;i++){

                        filas[i]=new Container();
                        filas[i].setLayout(null);

                        if(i==0){
                                filas[i].add(lip);
                                filas[i].add(ip);
lip.setBounds(10, 10, 40, 30);
ip.setBounds(40,15,150,25);
ip.setText("192.168.1.");
lip.setForeground(Color.white);
                        }
                        else if(i==1){
                                filas[i].add(lpuerto);
                                filas[i].add(puerto);
lpuerto.setBounds(10, 10, 100, 30);
                                puerto.setBounds(60,10,40,30);
                                puerto.setText("8080");
                                lpuerto.setForeground(Color.white);

                        }
                        else if(i==2){
                                filas[i].add(conectar);
                                filas[i].add(controles);
                                conectar.setBounds(20, 10, 400, 30);
                        }

                        else if(i==6){
                                filas[i].add(adelante);
                                adelante.setBounds(350, 10, 100, 25);

                                filas[i].add(stop);
                                stop.setBounds(500, 10, 100, 25);
                        }

                        else if(i==7){
                                filas[i].add(izquierda);
                                izquierda.setBounds(250, 10, 100, 25);
                                filas[i].add(atras);
                                atras.setBounds(350, 10, 100, 25);
                                filas[i].add(derecha);
                                derecha.setBounds(450, 10, 100, 25);
                        }
                        if(i==9){
                                filas[i].add(encenderluces);
                                encenderluces.setBounds(50, 10, 150, 25);

                                filas[i].add(modoautomatico);
                                modoautomatico.setBounds(250, 10, 150, 25);
                        }
                        if(i==10){
                                filas[i].add(apagarluces);
                                apagarluces.setBounds(50, 10, 150, 25);

                                filas[i].add(apagarmodoautomatico);
                                apagarmodoautomatico.setBounds(250, 10, 150, 25);
                        }
                        cBotones.add(filas[i]);
                }

                ////////////////////////////////////////////////////////////////////
```

```
                ////////////////////////////////////////////////////////////////
                // Se definen las funciones de los botones
                conectar.addActionListener(new ActionListener() {
                @Override
                public void actionPerformed(ActionEvent e){
                        URL urlVideo;
                        try{
                                urlVideo=new
                                URL("http://"+ip.getText()+":"+puerto.getText()+"/videofeed");

                                if(lecturaVideo!=null)
                                        lecturaVideo.stop();
                                lecturaVideo=new Thread(new MjpegRunner(visor, urlVideo));
                                lecturaVideo.start();
                        }
                        catch(Exception ex){}
                }
                });

        adelante.addActionListener(new ActionListener() {
            @Override
           public void actionPerformed(ActionEvent e){
                        comandoBTooth(e.getActionCommand());
            }
        });
        adelante.setActionCommand("f");

        atras.addActionListener(adelante.getAction());
        atras.setActionCommand("b");

        stop.addActionListener(adelante.getAction());
        stop.setActionCommand("s");

        izquierda.addActionListener(adelante.getAction());
        izquierda.setActionCommand("l");

        derecha.addActionListener(adelante.getAction());
        derecha.setActionCommand("r");

        modoautomatico.addActionListener(adelante.getAction());
        modoautomatico.setActionCommand("a");

        apagarmodoautomatico.addActionListener(adelante.getAction());
        apagarmodoautomatico.setActionCommand("m");

        encenderluces.addActionListener(new ActionListener(){
            @Override
            public void actionPerformed(ActionEvent e){
                        try{
                                URLConnection con=new
URL("http://"+ip.getText()+":"+puerto.getText()+e.getActionCommand()).openConnection();
                                con.setReadTimeout(500);
                                con.connect();
                                con.getInputStream().read();
                        }
                        catch(Exception ex){}

            }

        });
        encenderluces.setActionCommand("/enabletorch");

        apagarluces.addActionListener(encenderluces.getAction());
        apagarluces.setActionCommand("/disabletorch");
        ////////////////////////////////////////////////////////////////
```

```
                ventana.setVisible(true);//Muestra la ventana

                ////////////////////////////////////////////////////////////
                // Se establece la conexión con el puerto serie asociado al
                // módulo bluetooth.
                visor.mensaje("Conectando a bluetooth...", Color.white);
                try{
                        serialPort.openPort();
                        serialPort.setParams(SerialPort.BAUDRATE_9600, SerialPort.DATABITS_8,
                        SerialPort.STOPBITS_1,SerialPort.PARITY_NONE);
                        visor.mensaje("Conectado a bluetooh", Color.GREEN);
                }
                        catch(Exception ex){
                        visor.mensaje("Error: Dispositivo bluetooth no encontrado", Color.RED);
    }
        }//Fin del main
```

Las teclas se captan con el KeyListener:

```
static KeyListener TeclasPulsadas=new KeyListener()  {
                /*
                 * Este objeto se encarga de ejecutar las funciones correspondientes
                 * cuande se pulse o se suelte una tecla. Para cada tecla definida envía
                 * un byte por el pueto serie.
                 */

            @Override
            public void keyTyped(KeyEvent e) {
            }

    @Override
    public void keyPressed(KeyEvent e) {

       if("W".equals(KeyEvent.getKeyText(e.getKeyCode())))
          comandoBTooth("f");

       if("S".equals(KeyEvent.getKeyText(e.getKeyCode())))
          comandoBTooth("b");

       if("A".equals(KeyEvent.getKeyText(e.getKeyCode())))
          comandoBTooth("l");

       if("D".equals(KeyEvent.getKeyText(e.getKeyCode())))
          comandoBTooth("r");
    }

    @Override
    public void keyReleased(KeyEvent e) {
        if("W".equals(KeyEvent.getKeyText(e.getKeyCode())))
                comandoBTooth("s");

        if("S".equals(KeyEvent.getKeyText(e.getKeyCode())))
                comandoBTooth("s");

        if("A".equals(KeyEvent.getKeyText(e.getKeyCode())))
                comandoBTooth("c");

        if("D".equals(KeyEvent.getKeyText(e.getKeyCode())))
                comandoBTooth("c");
    }
        };
```

6 Enlaces hacia vídeos

A partir de este enlace podemos ver los videos que iremos subiendo:

http://youtu.be/FOd9rXpoTz4

7 Presupuesto

Componente	Precio ud	Cantidad	Precio
Módulo bluetooth CZ-HC-05	11,99 €	1	11,99 €
Sensor ultrasónico HC-SR-04	3,00 €	1	3,00 €
Tower pro micro servo SG90	4,90 €	1	4,90 €
BJT TIP 31-C (NPN)	1,20 €	4	4,80 €
BJT TIP 32-C (PNP)	1,20 €	2	2,40 €
Pack x2 Motor DC con reductora + rueda	6,90 €	1	6,90 €
***Arduino UNO (atmega 328)**	17,45 €	1	17,45 €
***Resistencias 100Ohm 0,25W**	0,03 €	2	0,06 €
***Resistencias 1kOhm 0,25W**	0,03 €	2	0,06 €
***LED**	0,15	4	0,60 €
TOTAL			**52,16 €**