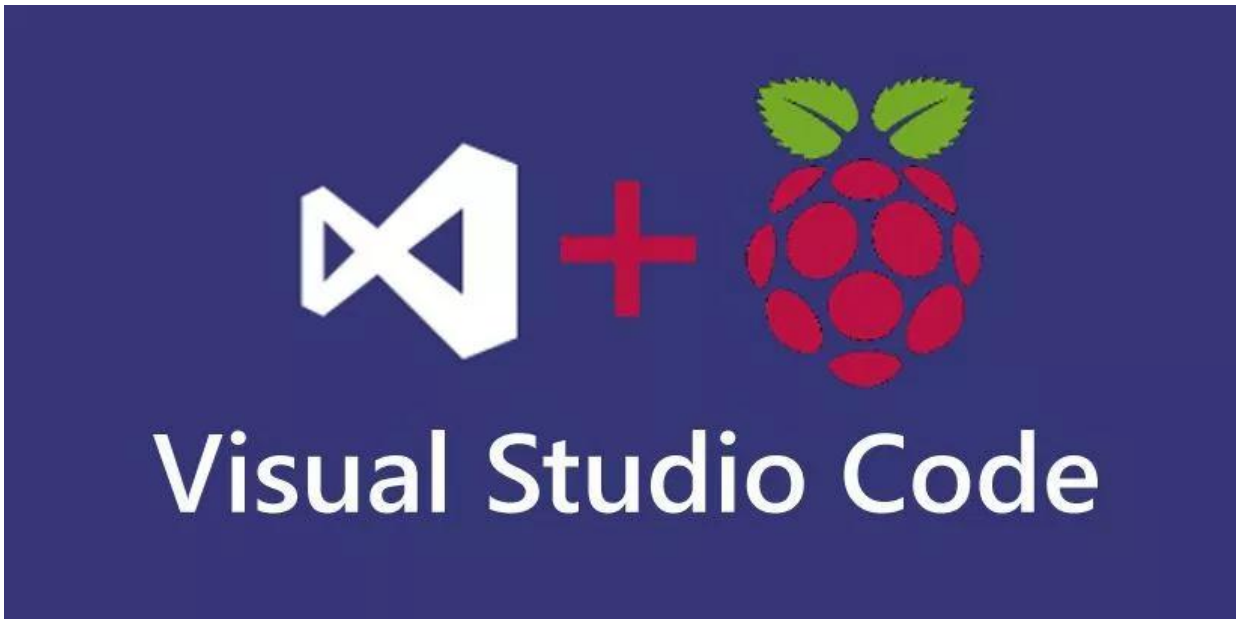
Práctica autónoma - La bombilla.

Sistemas electrónicos interactivos.



Gonzalo Bueno Santana

29 - mayo - 2019

La bombilla, proyecto realizado por:
Gonzalo Bueno

La bombilla, proyecto realizado por:
Gonzalo Bueno

1 - Introducción.

El objetivo de este proyecto era crear una interfaz de control natural, usando reconocimiento del esqueleto, para controlar mediante gestos y voz una bombilla conectada a una RaspBerry. La propuesta está pensada para personas con alguna dificultar en el control de la motricidad fina, centrándonos en gestos realizables usando motricidad gruesa, por ello el interfaz de control la hemos creado lo más sencillo posible. Además, nos hemos volcado desde el principio en la parte de comunicación, proponiendo un sistema de comunicación inalámbrico usando el protocolo JSON para hacer fácilmente escalable el proyecto a más de una RaspBerry. Por otra parte, una sola RaspBerry puede controlar fácilmente más de una luz.

Tras muchos ensayos el sistema de reconocimiento de esqueleto y la interfaz de control ha sido un éxito, nos decantamos por un sistema visual en el que se muestra el color y el brillo puesto en la propia pantalla del ordenador, además, los gestos van de manera fluida. Al final, en lugar de mostrar una imagen encima de la mano con las posibles opciones de selección, hemos cambiado el color de la propia mano, lo cual hace mucho más intuitivo el sistema.
Por otra parte, el sistema de comunicación hemos tenido que cambiarlo por completo y, en lugar de usar JSON como estaba previsto hemos acabado adaptado un servidor/cliente TCP, por lo que el proyecto es menos escalable, aunque da una respuesta rápida y fluida en lo que a intercambio de datos se refiere.

2 - Especificaciones iniciales.

Dividimos nuestro proyecto en dos partes siguiendo la planificación de la asignatura:

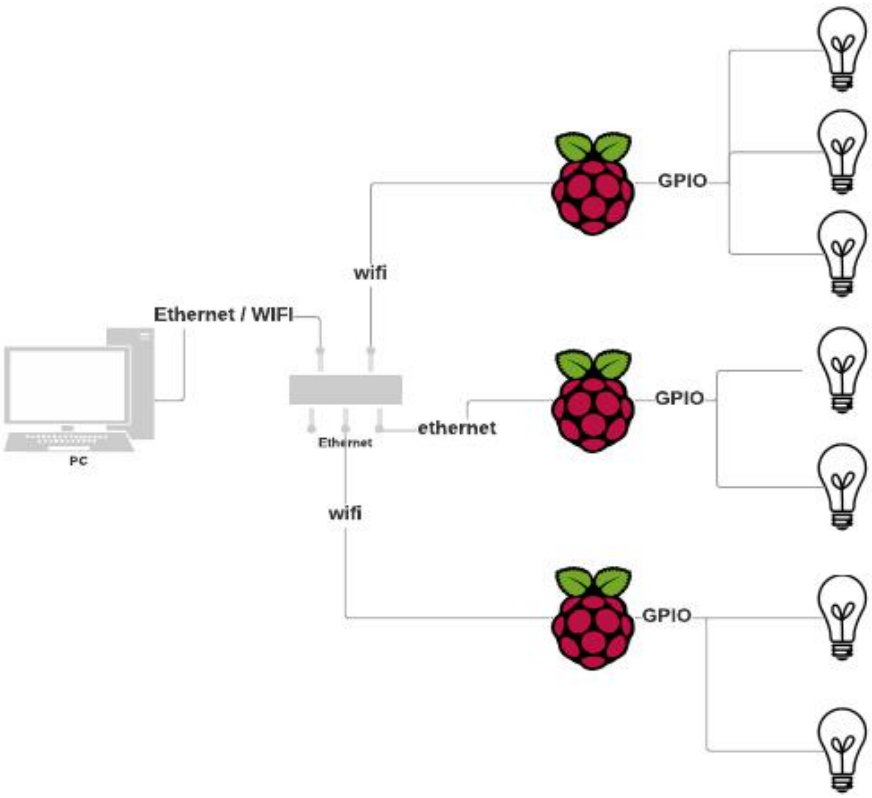
Las especificaciones mínimas, que se componen de 3 objetivos principales:

1. Reconocer varios comandos de voz para encender y apagar la bombilla.
2. Reconocer la posición del esqueleto para crear una interfaz de control que pueda manejar el brillo y el color de la bombilla
3. Aprender a usar Windows IoT core para configurar remotamente la RaspBerry.

Las especificaciones avanzadas, que consisten en cambiar el SO de la RaspBerry a software libre y usar el protocolo JSON para hacer sencilla la escalabilidad del proyecto, y así poder controlar tantas bombillas como se quiera.

3 - Diseño.

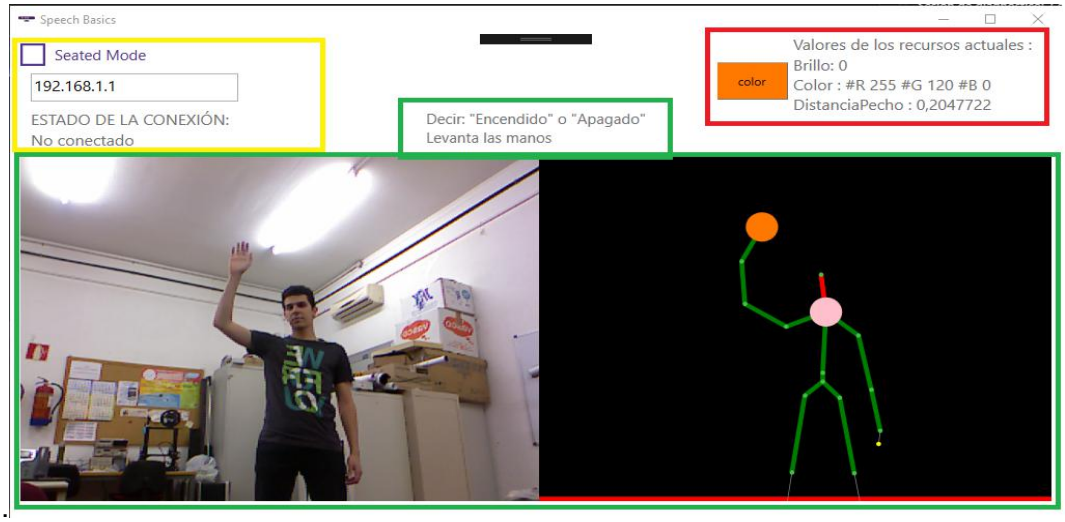
En el diagrama 3.1 podemos ver el esquema del sistema completo. Este sistema puede constar de numerosas Raspberries y del control de hasta 24 bombillas por RaspBerry, correspondientes al número de puertos GPIO disponibles. Además, el proyecto se podría escalar controlando más bombillas por puerto usando Arduino.



3.1 Ejemplo de diagrama del sistema completo.

El interfaz mostrado en la imagen 3.2 funciona en dos bloques:

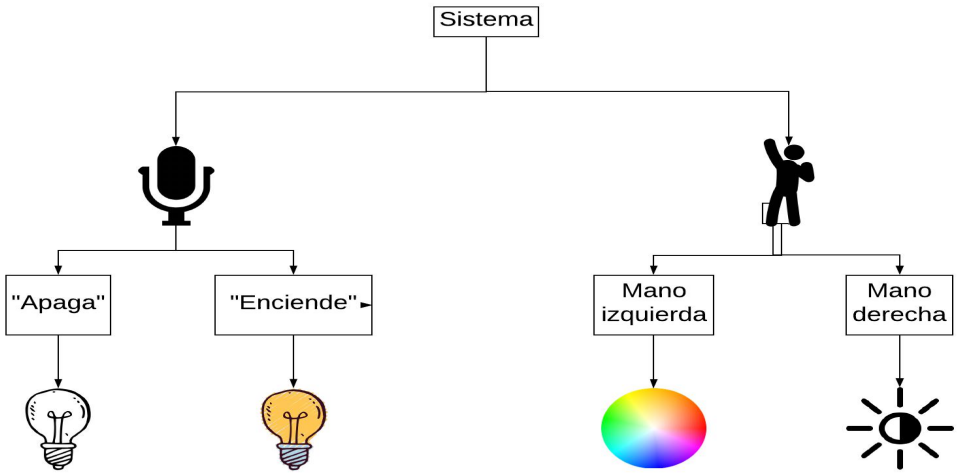
- Por una parte, **está la zona de desarrollo y conectividad**: esta parte corresponde a la zona superior del panel y muestra los valores enviados en tiempo real (marcado en rojo en la imagen 3.2). Además, en la parte izquierda del panel también se muestra una entrada de texto para conectar la IP de la RaspBerry (marcada en amarillo en la imagen 3.2). Justo abajo hay un panel que informa si hay algún error en la conexión.
- En la parte inferior está **la interfaz de control** de la bombilla (marcada en verde en la imagen 3.2), en esta se distinguen 3 bloques principales:
    - Un vídeo en tiempo real que graba al usuario
    - Un vídeo que muestra el reconocimiento de esqueleto del usuario y el control de luminosidad y color de la bombilla
    - Un bloque de texto que se ilumina cuando la bombilla está encendida o apagada y que da instrucciones sobre cómo proceder



3.2 Captura del interfaz

Esta interfaz funciona siguiendo el siguiente esquema de control:

- Lo primero es realizar la conexión con la RaspBerry, para ello se introduce la IP en el recuadro y al finalizar se hace doble clic en el mismo.
- Si la conexión se realiza correctamente debajo debería aparecer el mensaje “conectado”.
- Al comenzar nuestro sistema está apagado por lo que se debe encender utilizando el comando de voz “Encendido” o “Lumus”.
- Para apagarlo podrá usar los comandos de vos “apagado” o “Nox”.
- Por defecto la luz se encenderá en blanco con el brillo al máximo.
- Si se desea modificar el color o el brillo deberá colocarse en el campo visual de la Kinect, una vez esta reconozca su esqueleto podrá usar sus brazos para controlar dicha funcionalidad
    - Su brazo izquierdo puede controlar el color
    - El derecho el brillo
    - Para entrar en modo “selección” debe subir la mano por encima de la cabeza, y la distancia con respecto a su pecho determinará el color y el brillo.



3.3 Diagrama de control.

4 - Implementación.

Para la implementación del proyecto hemos usado una RaspBerry Pi 3b, una Kinect Xbox 360, y un pc portátil Lenovo ideapad 320, sin tarjeta gráfica dedicada, con 8gb de memoria ram y un procesador Intel i5 8250U, con Windows 10 Home de 64-bits. Hemos usado la versión beta del Visual Studio 2019, versión 16.0.4, y .Net Framework 4.8.03761, para la programación en C# y el cliente TCP. Además, hemos programado en C para el SO Raspbian 2019 el código de nuestro servidor y controlado el hardware de la RaspBerry desde el mismo código.

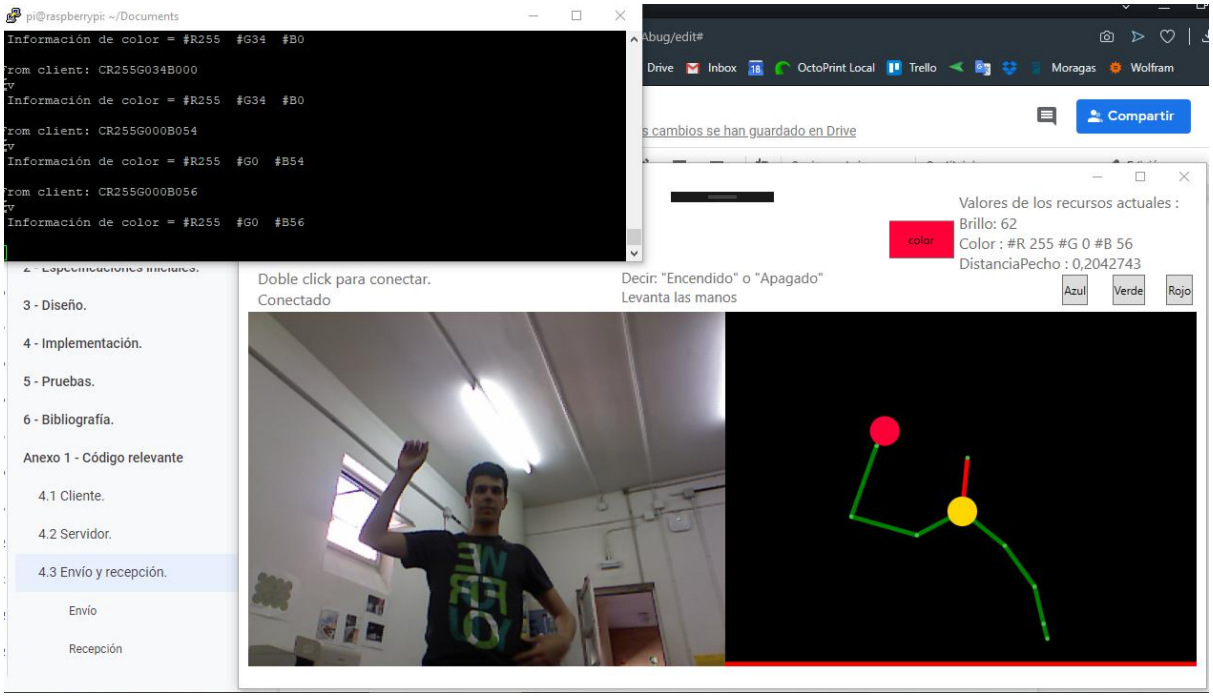
Nuestro código tiene fundamentalmente 6 partes relevantes (añadidas en el anexo 1), a saber:

1. El código relacionado con el establecimiento de la conexión TCP del cliente.
2. El código relacionado con el establecimiento de la conexión TCP del servidor.
3. El envío y recepción de los datos y la codificación de los mismos entre cliente y servidor
4. Lo relacionado con la interfaz visual
    a. El esqueleto
    b. La imagen de la cámara
    c. Los botones
    d. El reconocimiento de la intensidad de luz y el color a seleccionar
    e. La introducción de la IP destino de la RaspBerry.
5. El control de la RaspBerry.
6. El reconocimiento de voz.

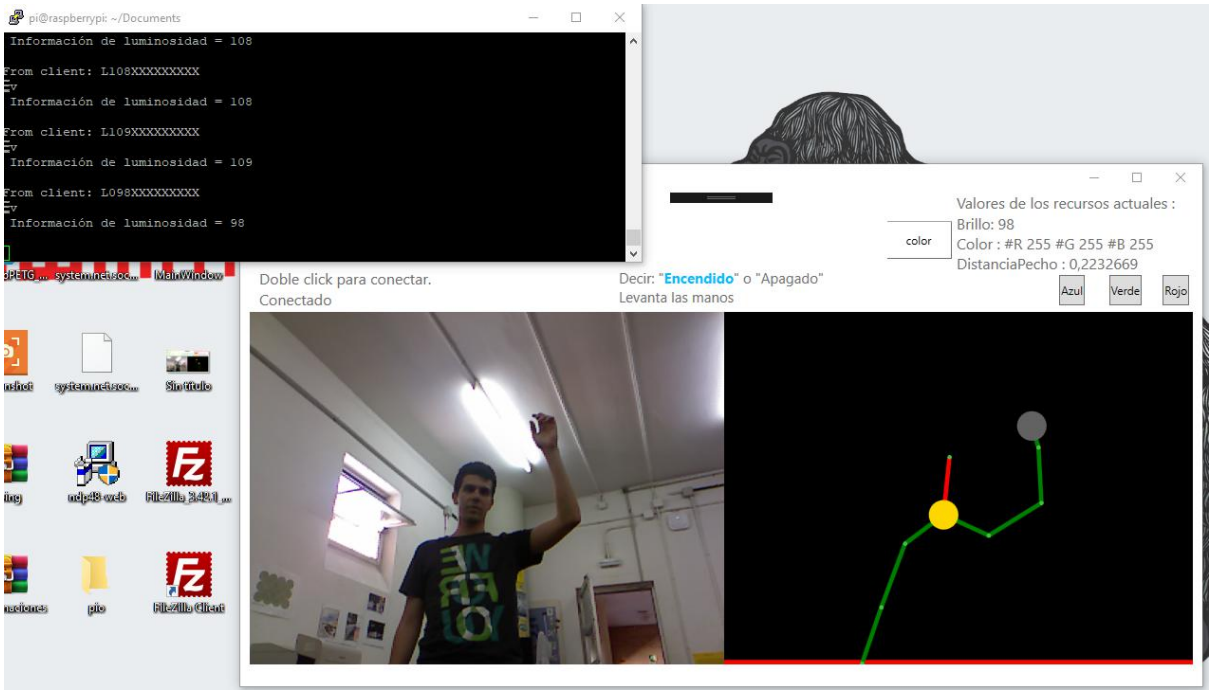
Para la cómoda realización de pruebas mostramos por la consola SSH de la RaspBerry los datos recibidos. Además, también se muestran en la zona de depuración de la propia ventana del PC.

5 - Pruebas.

Para las pruebas se ha puesto todo el sistema en funcionamiento y se han enviado diferentes códigos, tanto de color como de luminosidad. Se ha comprobado mediante SSH si se recibe en la RaspBerry de forma correcta. Hemos visualizado por pantalla el código que recibe el otro extremo de la comunicación simultáneamente puesto que en nuestra aplicación, en la zona de depuración, se guarda el último dato enviado y se muestra.



4.1 Imágenes de testeo de colores



4.2 Imágenes de testeo de luminosidad

Para el cierre de la comunicación se ha probado a hacerlo tanto con una señal desde el servidor como por Ctrl+C y se ha comprobado que ambas funcionen correctamente.

También se ha probado a seleccionar una dirección que no es correcta. Al hacer esto nuestra aplicación intenta conectar y, al no recibir respuesta después de un tiempo, cierra la conexión. Durante este tiempo el programa se queda congelado esperando la conexión.

También hemos implementado los botones para enviar los valores “rojo”, “verde” y “azul” puros al sistema, y comprobar si la señal PWM se genera correctamente en cada uno de los pines GPIO.
Gracias a esto descubrimos que el LED RGB por defecto tiene más luminosidad en los pines azul y verde. Por esto hemos reajustado la fórmula para equilibrar los 3 colores y que se ajuste más al color deseado. Además, hemos limado el exterior del LED para que se mezclen mejor los colores en su interior. Este proceso no sería necesario con una bombilla RGB, aunque probablemente fuese necesaria una etapa de potencia para conseguir la intensidad deseada. También se podría usar un relé de estado sólido para controlarla.

Adjuntamos dos vídeos probando el proyecto aquí y aquí.

6 - Bibliografía.

1. https://www.nuget.org/packages/System.Net.Sockets/4.3.0
2. https://www.xataka.com/makers/que-puedes-y-que-no-puedes-hacer-con-windows-10-iot-core-en-las-raspberry-pi-2
3. https://docs.microsoft.com/es-es/windows/iot-core/downloads
4. https://www.hackster.io/arioobarzan/connect-raspberry-pi-and-pc-with-tcp-ip-using-csharp-a299ed
5. https://windowsinstructed.com/windows-iot-raspberry-communication-server-client/
6. https://www.codeproject.com/Articles/1079341/Simple-Connection-Between-Raspberry-Pi-with-Window
7. https://stackoverflow.com/questions/51514526/tcp-communication-between-raspberry-pi-with-win10-iot-and-a-xamarin-android-appl
8. https://blogs.msdn.microsoft.com/robep/2017/06/21/desarrollando-con-visual-studio-code-y-raspberry-pi/
9. http://www.programmingapps.net/2015/10/windows-iot-hola-mundo/
10. https://books.google.es/books?id=YQQXHEj6O4QC&pg=PR11&lpg=PR11&dq=socket+not+contain+tcplistener&source=bl&ots=AvC5QWfROn&sig=ACfU3U08yCuaysUiDrRAAfqW2dSoq8yhiA&hl=es&sa=X&ved=2ahUKEwjJzr6p97jiAhXBAmMBHacbD1UQ6AEwA3oECAkQAQ#v=onepage&q=socket%20not%20contain%20tcplistener&f=false
11. https://www.nuget.org/packages/System.Net.Sockets/4.3.0
12. https://www.admfactory.com/rgb-led-on-raspberry-pi-using-c/
13. https://www.admfactory.com/breathing-light-led-on-raspberry-pi-using-c/

La mayoría de la bibliografía aquí adjuntada está relacionada con el control de la RaspBerry usando Windows IoT y para crear el servidor TCP y el protocolo de comunicación puesto que es lo que más trabajo ha requerido de todo el proyecto.

Anexo 1 - Código relevante

1.1 Cliente.

1 referencia
```
private void TextInput_MouseDoubleClick(object sender, System.Windows.Input.MouseButtonEventArgs e)
{
        try
        {
            string ip;
            /*using (System.IO.StreamReader file = new System.IO.StreamReader("conf.ini"))
            {
                ip = file.ReadLine();
            }*/
            ip = textInput.Text;
            if (ip.Length < 16 && ip.Length > 10)
            {
                IPAddress ipAd = IPAddress.Parse(ip);
                //this.estadoTCP.Text = "Conectando";
                estadoTCP.Text = "Conectando";
                tcpclnt = new TcpClient();
                estadoTCP.Text = "Conectando con " + ip + " por el puerto 8010";
                tcpclnt.Connect(ip, 8010);
                //texto1.Text = "oatata";
                if (tcpclnt.Connected)
                {
                    estadoTCP.Text = "Conectado";
                    conectadoTCP = true;
                }
                else
                {
                    estadoTCP.Text = "No conectado";
                    conectadoTCP = false;
                }
            }
                }
                else
                {
                    //Caracteres insuficientes
                    estadoTCP.Text = "Error de formato";
                }
            }
            catch (SocketException err)
            {
                estadoTCP.Text = "Error: " + err.StackTrace;
            }
            catch (Exception err)
            {
                estadoTCP.Text = "Error: " + err.StackTrace;
            }
            //TCP
}
```

1.2 Servidor.

```
        softPwmCreate(LedPinRed,  0, 100);  //create a soft pwm, original duty
        softPwmCreate(LedPinGreen,0, 100);
        softPwmCreate(LedPinBlue, 0, 100);
        // socket create and verification
        sockfd = socket(AF_INET, SOCK_STREAM, 0);
        if (sockfd == -1) {
                printf("Fallo creano el socket...\n");
                exit(0);
        }
        else
                printf("Socket creado correctamente..\n");
        bzero(&servaddr, sizeof(servaddr));

        signal(SIGINT,manejador);

        // assign IP, PORT
        servaddr.sin_family = AF_INET;
        servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
        servaddr.sin_port = htons(PORT);

        // Binding newly created socket to given IP and verification
        if ((bind(sockfd, (SA*)&servaddr, sizeof(servaddr))) != 0) {
                printf("socket bind failed...\n");
                exit(0);
        }
        else
                printf("Socket successfully binded..\n");

        // Now server is ready to listen and verification
        if ((listen(sockfd, 5)) != 0) {
                printf("Listen failed...\n");
                exit(0);
        }
        else
                printf("Server listening..\n");
        len = sizeof(cli);

        // Accept the data packet from client and verification
        connfd = accept(sockfd, (SA*)&cli, &len);
```

```
        if (connfd < 0) {
                printf("server acccept failed...\n");
                exit(0);
        }
        else
                printf("server acccept the client...\n");

        // Function for chatting between client and server
        func(connfd);

        // After chatting close the socket
        close(sockfd);
}
```

1.3 Envío y recepción.

1.3.1 Envío

```
if (conectadoTCP && estadoBombilla){
    //"Enter the string to be transmitted : ");
    int longitudm = 0;
        String str = "C" + "R" ; //Console.ReadLine();
    if (rActual < 10){
        str += "00" + rActual.ToString();
    }
    else if (rActual < 100){
        str += "0" + rActual.ToString();
    }
    else{
        str += rActual.ToString();
    }

    str += "G";
    if (gActual < 10){
        str += "00" + gActual.ToString();
    }
    else if (gActual < 100){
        str += "0" + gActual.ToString();
    }
    else{
        str += gActual.ToString();
    }
    str += "B";
    if (bActual < 10){
        str += "00" + bActual.ToString();
    }
    else if (bActual < 100){
        str += "0" + bActual.ToString();
    }
    else{
        str += bActual.ToString();
    else{
        str += bActual.ToString();
    }
    Stream stm = tcpclnt.GetStream();

    str += '\n';
    ASCIIEncoding asen = new ASCIIEncoding();
    byte[] ba = asen.GetBytes(str);
    //Console.WriteLine("Transmitting...");

    stm.Write(ba, 0, ba.Length);

    /* byte[] bb = new byte[100];
    int k = stm.Read(bb, 0, 100);*/

}
```

1.3.2 Recepción

```
void func(int sockfd)
{
        char buff[MAX];
        int n;
        // infinite loop for chat
        int r=255,g=255, b=255;
        int n1 = 0, n10=0, n100=0;
        int lum=255;
        short encendido = 0;
        for (;;) {
                bzero(buff, MAX);

                // read the message from client and copy it in buffer
                readn(sockfd, buff, sizeof(buff));
                // print buffer which contains the client contents
                printf("From client: %s\t \n ", buff);

                if(buff[0]=='E'){
                        encendido = 1;
                        softPwmWrite(LedPinRed,   (int)(r*((float)lum)/255));   //$
                        softPwmWrite(LedPinGreen, (int)(g*((float)lum)/265));
                        softPwmWrite(LedPinBlue,  (int)(b*((float)lum)/270));
                }
                else if(buff[0]== 'A'){
                        encendido = 0;
                        softPwmWrite(LedPinRed,   0);  //change duty cycle
                        softPwmWrite(LedPinGreen, 0);
                        softPwmWrite(LedPinBlue,  0);
                }
                if(buff[0]=='C' && encendido==1){
                        /*r_val = (color & 0xFF0000) >> 16;  //get red value
                        g_val = (color & 0x00FF00) >> 8;   //get green value
                        b_val = (color & 0x0000FF) >> 0;   //get blue value*/

                        n1 = buff[4]-48;
                        n10 = buff[3]-48;
                        n100 = buff[2]-48;
                        r = n100*100 + n10*10 + n1;
                        n1 = buff[8]-48;
                        n10 = buff[7]-48;
                        n100 = buff[6]-48;
                        g = n100*100 + n10*10 + n1;
                        n1 = buff[12]-48;
                        n10 = buff[11]-48;
                        n100 = buff[10]-48;
                        b = n100*100 + n10*10 + n1;
                      printf("Información de color = #R%d  #G%d  #B%d \n\n", r, g, b);

                        r = r*100/255 ;    //change a num(0~255) to 0~10
                        g = g*100/255;
                        b = b*100/255;
                       if(lum>=0){
                                softPwmWrite(LedPinRed,   (int)(r*((float)lum)/255));  //chan
                                softPwmWrite(LedPinGreen, (int)(g*((float)lum)/265));
                                softPwmWrite(LedPinBlue,  (int)(b*((float)lum)/270));
                        }
                }else   if(buff[0]=='L' && encendido==1){
                        n1 = buff[3]-48;
                        n10 = buff[2]-48;
                        n100 = buff[1]-48;
                        lum = n100*100 + n10*10 + n1;
                        printf("Información de luminosidad = %d\n\n", lum);
                        softPwmWrite(LedPinRed,   (int)(r*((float)lum)/255));  //change duty
                        softPwmWrite(LedPinGreen, (int)(g*((float)lum)/255));
                        softPwmWrite(LedPinBlue,  (int)(b*((float)lum)/255));
                }
```

1.4 Interfaz.

1.4.1 Esqueleto.

Código de la mano derecha.

```
if (joint.JointType == JointType.HandRight){
    miAlturaDer = joint.Position.Y;
    if (miAlturaDer > miAlturaCabeza){
        //drawBrush = this.brushManoDer;
        miJointThickness = JointThicknessOro;
        float temporalDer = (joint.Position.X - xPecho)/(distanciaPechoCabeza * (float)2);
        if (temporalDer < 0.3){
            distanciaDer = 0;
        }
        else if (temporalDer > 1.3){
            distanciaDer = 1;
        }
        else{
            distanciaDer = temporalDer - (float)0.3;
        }
        BrilloValue.Text = ((int)(distanciaDer * 255)).ToString();
        drawBrush = new SolidColorBrush(Color.FromRgb((byte)(distanciaDer * 255), (byte)(distanciaDer * 255),
            (byte)(distanciaDer * 255)));

            if (conectadoTCP && estadoBombilla){
            //"Enter the string to be transmitted : ");
                int longitud = 0;
                String str = "L" ; //Console.ReadLine();
            if (distanciaDer*255 < 10){
                str += "00" + ((byte)(distanciaDer*255)).ToString();
            }
            else if (distanciaDer *255 < 100){
                str += "0" + ((byte)(distanciaDer * 255)).ToString();
            }
            else{
                str += ((byte)(distanciaDer * 255)).ToString();
            }
            longitud = 14 - str.Length;
            while (longitud > 1){
                longitud--;
                str += "X";
            }
            str += '\n';
                Stream stm = tcpclnt.GetStream();

                ASCIIEncoding asen = new ASCIIEncoding();
                byte[] ba = asen.GetBytes(str);
                //Console.WriteLine("Transmitting...");

                stm.Write(ba, 0, ba.Length);
```

Mano Izquierda

```
double miJointThickness = JointThickness;
if (joint.JointType == JointType.HandLeft)
{
    miAlturaIzqda = joint.Position.Y;
    if (miAlturaIzqda > miAlturaCabeza)
    {
        //drawBrush = this.brushManoIzq;
        miJointThickness = JointThicknessAzul;
        // this.Imagen.Visibility = Visibility.Visible;
        float temporalIzq = (xPecho - joint.Position.X)/(distanciaPechoCabeza * (float)2);
        if (temporalIzq < 0.3)
        {
            distanciaIzq = 0;
        }
        else if (temporalIzq > 1.3) {
            distanciaIzq = 1;
        }
        else
        {
            distanciaIzq = temporalIzq - (float)0.3;
        }

        //Condiciones brushese brochas
        if (0 >= distanciaIzq && distanciaIzq > 0.25)
        {
           // drawBrush = this.brushBlanco;
            rActual =255;
            gActual =255;
            bActual =255;
        }
        else if (0.25 <= distanciaIzq && distanciaIzq < 0.375) {
            //drawBrush = this.brushRojo;
            rActual = 255;
            gActual = 0;
            bActual = (byte)(255 - (((distanciaIzq - 0.25) * 255) / 0.125));
        }
        else if (0.375 <= distanciaIzq && distanciaIzq < 0.5) {
            //drawBrush = this.brushAmarillo;
            rActual = 255;
            gActual = (byte)(((distanciaIzq - 0.375) * 255) / 0.125);
            bActual = 0;
        }
        else if (0.5 <= distanciaIzq && distanciaIzq < 0.625){
            //drawBrush = this.brushVerde;
            rActual = (byte) (255 -(((distanciaIzq - 0.5)*255)/ 0.125));
            gActual = 255;
            bActual = 0;
        }
```

```
}
else if (0.625 <= distanciaIzq && distanciaIzq < 0.75){
    //drawBrush = this.brushAguamarina;
    rActual = 0;
    gActual = 255;
    bActual = (byte)(((distanciaIzq - 0.625) * 255) / 0.125);
}
else if (0.75 <= distanciaIzq && distanciaIzq < 0.875){
    //drawBrush = this.brushAzul;
    rActual = 0;
    gActual = (byte)(255 - (((distanciaIzq - 0.75) * 255) / 0.125));
    bActual = 255;
}
else if (0.875 <= distanciaIzq && distanciaIzq < 1){
    //drawBrush = this.brushRosa;
    rActual = (byte)(((distanciaIzq - 0.875) * 255) / 0.125);
    gActual = 0;
    bActual = 255;
}
else{
   // drawBrush = this.brushBlanco;
    rActual = 255;
    gActual = 255;
    bActual = 255;
}
//LeftValue.SetCurrentValue(MainWindow,distanciaIzq.ToString());
colorValueButton.Background = new SolidColorBrush(Color.FromRgb(rActual,gActual,bActual));
drawBrush= new SolidColorBrush(Color.FromRgb(rActual, gActual, bActual));
ColorValueR.Text = rActual.ToString();
ColorValueG.Text = gActual.ToString();
ColorValueB.Text = bActual.ToString();
//Aquí los datos de color

if (conectadoTCP && estadoBombilla){
    //"Enter the string to be transmitted : ");
    int longitudm = 0;
        String str = "C" + "R" ; //Console.ReadLine();
    if (rActual < 10){
        str += "00" + rActual.ToString();
    }
    else if (rActual < 100){
        str += "0" + rActual.ToString();
    }
    else{
        str += rActual.ToString();
    }

    str += "G";
    if (gActual < 10){
```

```
            str += "00" + gActual.ToString();
        }
        else if (gActual < 100){
            str += "0" + gActual.ToString();
        }
        else{
            str += gActual.ToString();
        }
        str += "B";
        if (bActual < 10){
            str += "00" + bActual.ToString();
        }
        else if (bActual < 100){
            str += "0" + bActual.ToString();
        }
        else{
            str += bActual.ToString();
        }
        Stream stm = tcpclnt.GetStream();

        str += '\n';
        ASCIIEncoding asen = new ASCIIEncoding();
        byte[] ba = asen.GetBytes(str);
        //Console.WriteLine("Transmitting...");

        stm.Write(ba, 0, ba.Length);

        /* byte[] bb = new byte[100];
        int k = stm.Read(bb, 0, 100);*/

    }


}
else{
    //this.Imagen.Visibility = Visibility.Hidden;

}
```

1.4.2 Imagen cámara.

```
void myKinect_ColorFrameReady(object sender, ColorImageFrameReadyEventArgs e)
{
    using (ColorImageFrame ColorFrame = e.OpenColorImageFrame())
    {
        if (ColorFrame == null) return;
        byte[] ColorData = new byte[ColorFrame.PixelDataLength];
        ColorFrame.CopyPixelDataTo(ColorData);
        Imagen.Source = BitmapSource.Create(
            ColorFrame.Width, ColorFrame.Height,
            500, 500,
            PixelFormats.Bgr32,
            null,
            ColorData,
            ColorFrame.Width * ColorFrame.BytesPerPixel
            );
    }
}
```

1.4.3 Botones.

```
private void BotonAzul_Click(object sender, RoutedEventArgs e)
{
    if (conectadoTCP)
    {
        //"Enter the string to be transmitted : ");
        estadoBombilla = true;
        String str; //Console.ReadLine();
        str = "E" + "XXXXXXXXXXX";
        str += '\n';
        Stream stm = tcpclnt.GetStream();
        ASCIIEncoding asen = new ASCIIEncoding();
        byte[] ba = asen.GetBytes(str);
        //Console.WriteLine("Transmitting...");
        stm.Write(ba, 0, ba.Length);


        str = "L255XXXXXXXXX";
        str += '\n';
        // asen = new ASCIIEncoding();
        ba = asen.GetBytes(str);
        stm.Write(ba, 0, ba.Length);

        str = "CR000G000B255";
        str += '\n';
        ba = asen.GetBytes(str);
        stm.Write(ba, 0, ba.Length);
    }
}
```

1.4.4 Selección de color y luminosidad.

Luminosidad:

```
if (miAlturaDer > miAlturaCabeza){
    //drawBrush = this.brushManoDer;
    miJointThickness = JointThicknessOro;
    float temporalDer = (joint.Position.X - xPecho)/(distanciaPechoCabeza * (float)2);
    if (temporalDer < 0.3){
        distanciaDer = 0;
    }
    else if (temporalDer > 1.3){
        distanciaDer = 1;
    }
    else{
        distanciaDer = temporalDer - (float)0.3;
    }
    BrilloValue.Text = ((int)(distanciaDer * 255)).ToString();
    drawBrush = new SolidColorBrush(Color.FromRgb((byte)(distanciaDer * 255), (byte)(distanciaDer * 255),
        (byte)(distanciaDer * 255)));

        if (conectadoTCP && estadoBombilla){
        //"Enter the string to be transmitted : ");
            int longitud = 0;
            String str = "L" ; //Console.ReadLine();
        if (distanciaDer*255 < 10){
            str += "00" + ((byte)(distanciaDer*255)).ToString();
        }
        else if (distanciaDer *255 < 100){
            str += "0" + ((byte)(distanciaDer * 255)).ToString();
        }
        else{
            str += ((byte)(distanciaDer * 255)).ToString();
        }
        longitud = 14 - str.Length;
        while (longitud > 1){
            longitud--;
            str += "X";
        }
        str += '\n';
            Stream stm = tcpclnt.GetStream();
```

Mano izquierda

```
miAlturaIzqda = joint.Position.Y;
if (miAlturaIzqda > miAlturaCabeza)
{
    //drawBrush = this.brushManoIzq;
    miJointThickness = JointThicknessAzul;
    // this.Imagen.Visibility = Visibility.Visible;
    float temporalIzq = (xPecho - joint.Position.X)/(distanciaPechoCabeza * (float)2);
    if (temporalIzq < 0.3)
    {
        distanciaIzq = 0;
    }
    else if (temporalIzq > 1.3) {
        distanciaIzq = 1;
    }
    else
    {
        distanciaIzq = temporalIzq - (float)0.3;
    }

    //Condiciones brushese brochas
    if (0 >= distanciaIzq && distanciaIzq > 0.25)
    {
       // drawBrush = this.brushBlanco;
        rActual =255;
        gActual =255;
        bActual =255;
    }
    else if (0.25 <= distanciaIzq && distanciaIzq < 0.375) {
        //drawBrush = this.brushRojo;
        rActual = 255;
        gActual = 0;
        bActual = (byte)(255 - (((distanciaIzq - 0.25) * 255) / 0.125));
    }
    else if (0.375 <= distanciaIzq && distanciaIzq < 0.5) {
        //drawBrush = this.brushAmarillo;
        rActual = 255;
        gActual = (byte)(((distanciaIzq - 0.375) * 255) / 0.125);
        bActual = 0;
    }
```

```
}
else if (0.5 <= distanciaIzq && distanciaIzq < 0.625){
    //drawBrush = this.brushVerde;
    rActual = (byte) (255 -(((distanciaIzq - 0.5)*255)/ 0.125));
    gActual = 255;
    bActual = 0;
}
else if (0.625 <= distanciaIzq && distanciaIzq < 0.75){
    //drawBrush = this.brushAguamarina;
    rActual = 0;
    gActual = 255;
    bActual = (byte)(((distanciaIzq - 0.625) * 255) / 0.125);
}
else if (0.75 <= distanciaIzq && distanciaIzq < 0.875){
    //drawBrush = this.brushAzul;
    rActual = 0;
    gActual = (byte)(255 - (((distanciaIzq - 0.75) * 255) / 0.125));
    bActual = 255;
}
else if (0.875 <= distanciaIzq && distanciaIzq < 1){
    //drawBrush = this.brushRosa;
    rActual = (byte)(((distanciaIzq - 0.875) * 255) / 0.125);
    gActual = 0;
    bActual = 255;
}
else{
   // drawBrush = this.brushBlanco;
    rActual = 255;
    gActual = 255;
    bActual = 255;
}
//LeftValue.SetCurrentValue(MainWindow,distanciaIzq.ToString());
colorValueButton.Background = new SolidColorBrush(Color.FromRgb(rActual,gActual,bActual));
drawBrush= new SolidColorBrush(Color.FromRgb(rActual, gActual, bActual));
ColorValueR.Text = rActual.ToString();
ColorValueG.Text = gActual.ToString();
ColorValueB.Text = bActual.ToString();
//Aquí los datos de color
```

1.4.5 Cuadro de texto para introducir la IP.

```
1 referencia
private void TextInput_MouseDoubleClick(object sender, System.Windows.Input.MouseButtonEventArgs e)
{
        try
        {
            string ip;
            /*using (System.IO.StreamReader file = new System.IO.StreamReader("conf.ini"))
            {
                ip = file.ReadLine();
            }*/
            ip = textInput.Text;
            if (ip.Length < 16 && ip.Length > 10)
            {
                IPAddress ipAd = IPAddress.Parse(ip);
                //this.estadoTCP.Text = "Conectando";
                estadoTCP.Text = "Conectando";
                tcpclnt = new TcpClient();
                estadoTCP.Text = "Conectando con " + ip + " por el puerto 8010";
                tcpclnt.Connect(ip, 8010);
                //texto1.Text = "oatata";
                if (tcpclnt.Connected)
                {
                    estadoTCP.Text = "Conectado";
                    conectadoTCP = true;
                }
                else
                {
                    estadoTCP.Text = "No conectado";
                    conectadoTCP = false;
                }
            }
            else
            {
                //Caracteres insuficientes
                estadoTCP.Text = "Error de formato";
            }
        }
        catch (SocketException err)
        {
            estadoTCP.Text = "Error: " + err.StackTrace;
        }
        catch (Exception err)
        {
            estadoTCP.Text = "Error: " + err.StackTrace;
        }
        //TCP
```

1.5 El control de la RaspBerry

Código para el control de los LEDs en función de los datos recibidos.

```
if(buff[0]=='E'){//Encendido
        encendido = 1;
        softPwmWrite(LedPinRed,   (int)(r*((float)lum)/255));  //change duty cycle
        softPwmWrite(LedPinGreen, (int)(g*((float)lum)/265));
        softPwmWrite(LedPinBlue,  (int)(b*((float)lum)/270));
}
else if(buff[0]== 'A'){//Apagado
        encendido = 0;
        softPwmWrite(LedPinRed,   0);  //change duty cycle
        softPwmWrite(LedPinGreen, 0);
        softPwmWrite(LedPinBlue,  0);
}
if(buff[0]=='C' && encendido==1){//Dato de color
        /*r_val = (color & 0xFF0000) >> 16;  //get red value
        g_val = (color & 0x00FF00) >> 8;   //get green value
        b_val = (color & 0x0000FF) >> 0;   //get blue value*/

        n1 = buff[4]-48;
        n10 = buff[3]-48;
        n100 = buff[2]-48;
        r = n100*100 + n10*10 + n1;
        n1 = buff[8]-48;
        n10 = buff[7]-48;
        n100 = buff[6]-48;
        g = n100*100 + n10*10 + n1;
```

```
        n1 = buff[12]-48;
        n10 = buff[11]-48;
        n100 = buff[10]-48;
        b = n100*100 + n10*10 + n1;
        printf("Información de color = #R%d  #G%d  #B%d \n\n", r, g, b);
        r = r*100/255 ;    //change a num(0~255) to 0~10
        g = g*100/255;
        b = b*100/255;
       if(lum>=0){
                softPwmWrite(LedPinRed,   (int)(r*((float)lum)/255));  //change duty cycle
                softPwmWrite(LedPinGreen, (int)(g*((float)lum)/265));
                softPwmWrite(LedPinBlue,  (int)(b*((float)lum)/270));
        }
 }else   if(buff[0]=='L' && encendido==1){//Dato de luminosidad
        n1 = buff[3]-48;
        n10 = buff[2]-48;
        n100 = buff[1]-48;
        lum = n100*100 + n10*10 + n1;
        printf("Información de luminosidad = %d\n\n", lum);
        softPwmWrite(LedPinRed,   (int)(r*((float)lum)/255));  //change duty cycle
```

Código para el correcto cierre del socket al finalizar el programa

```
int sockfd;

void manejador(int i){
        close(sockfd);
        exit(1);
}
```

1.6 Reconocimiento de voz

```
if (null != ri)
{
    recognitionSpans = new List<Span> { forwardSpan, backSpan };

    this.speechEngine = new SpeechRecognitionEngine(ri.Id);

    /****************************************************************
    *
    * Use this code to create grammar programmatically rather than from
    * a grammar file.
    */
     var directions = new Choices();
     directions.Add(new SemanticResultValue("luz", "FORWARD"));
     directions.Add(new SemanticResultValue("lus", "FORWARD"));
     directions.Add(new SemanticResultValue("lumus", "FORWARD"));
    directions.Add(new SemanticResultValue("enciende", "FORWARD"));
    directions.Add(new SemanticResultValue("encender", "FORWARD"));
    directions.Add(new SemanticResultValue("encendido", "FORWARD"));
    directions.Add(new SemanticResultValue("encendio", "FORWARD"));

   // directions.Add(new SemanticResultValue("apagar", "BACKWARD"));
     directions.Add(new SemanticResultValue("apagao", "BACKWARD"));
    directions.Add(new SemanticResultValue("apagado", "BACKWARD"));
   // directions.Add(new SemanticResultValue("apaga", "BACKWARD"));
   // directions.Add(new SemanticResultValue("abajo", "BACKWARD"));


    directions.Add(new SemanticResultValue("turn left", "LEFT"));
    directions.Add(new SemanticResultValue("izquierda", "LEFT"));

    directions.Add(new SemanticResultValue("derecha", "RIGHT"));

     var gb = new GrammarBuilder { Culture = ri.Culture };
     gb.Append(directions);

     var g = new Grammar(gb);
    /*
    ****************************************************************/

    // Create a grammar from grammar definition XML file.
    using (var memoryStream = new MemoryStream(Encoding.ASCII.GetBytes(Properties.Resources.SpeechGrammar)))
    {
        // var g = new Grammar(memoryStream);
         speechEngine.LoadGrammar(g);
    }

    speechEngine.SpeechRecognized += SpeechRecognized;
```

```
if (e.Result.Confidence >= ConfidenceThreshold)
{
    switch (e.Result.Semantics.Value.ToString())
    {
        case "FORWARD":
            forwardSpan.Foreground = Brushes.DeepSkyBlue;
            forwardSpan.FontWeight = FontWeights.Bold;
            string temporal = "Derecha :\t" + distanciaDer.ToString() + "\n" + "Izquierda :\t" + distanciaIzq.ToStr
            if (depurando)  MessageBox.Show(temporal);

            if (conectadoTCP)
            {
                //"Enter the string to be transmitted : ");
                int longitud = 0;
                String str = "E" + "XXXXXXXXXXX"; //Console.ReadLine();
                str += '\n';
                Stream stm = tcpclnt.GetStream();

                estadoBombilla = true;
                ASCIIEncoding asen = new ASCIIEncoding();
                byte[] ba = asen.GetBytes(str);
                //Console.WriteLine("Transmitting...");

                stm.Write(ba, 0, ba.Length);

            }
            break;
        case "BACKWARD":
            backSpan.Foreground = Brushes.DeepSkyBlue;
            backSpan.FontWeight = FontWeights.Bold;
            if (depurando) MessageBox.Show("Apagado");
            if (conectadoTCP)
            {
                //"Enter the string to be transmitted : ");
                int longitud = 0;
                String str = "A" + "XXXXXXXXXXX"; //Console.ReadLine();
                str += '\n';
                Stream stm = tcpclnt.GetStream();
                estadoBombilla = false;
                ASCIIEncoding asen = new ASCIIEncoding();
                byte[] ba = asen.GetBytes(str);
                //Console.WriteLine("Transmitting...");

                stm.Write(ba, 0, ba.Length);


            }
            break;
        case "LEFT":
            if (depurando) MessageBox.Show(distanciaIzq.ToString());
            /*RightSpan.Foreground = Brushes.DeepSkyBlue;
            RightSpan.FontWeight = FontWeights.Bold;*/
            //RightValue.SetValue(distanciaDer.ToString());

            break;
        case "RIGHT":
            /*LeftSpan.Foreground = Brushes.DeepSkyBlue;
            LeftSpan.FontWeight = FontWeights.Bold;*/
            //textValues.Text.ToString;

            if (depurando) MessageBox.Show(distanciaDer.ToString());

            break;
        default:
            if (depurando) MessageBox.Show("DEFAULT");
            break;
    }
```