Documento técnico funcional

Sumo de robots
Robotgroup Multiplo N6 Max
Expocarreras 2016

Integrantes:

WIEILLY, Alan
MACULA, Alejandro
AMARO, Marcos
DIAZ BURCET, Nehuén

Sistemas Embebidos

Licenciatura en Sistemas

2016

CONTENIDO

INTRODUCCIÓN

En el presente documento se detallarán los principios básicos del Sumo de robots y la especificación técnica funcional del desarrollo realizado para la Expocarreras 2016. Se abarcarán tanto las reglas del juego, los algoritmos principales programados por el grupo y una guía con los pasos necesarios para preparar el ambiente de desarrollo del robot Multiplo N6 Max utilizado para la exposición.

Sumo de robots

El sumo de robots es una competencia en donde se enfrentan dos robots en una pista, el objetivo es sacar completamente de la pista al robot adversario.

La pista es una circunferencia, donde en la parte central existen dos franjas grises, que es desde los robots parten en el inicio de la competencia. También cuenta con una franja exterior blanca para indicar al robot que está fuera, y el interior de la pista es negra, que es donde se desarrolla la competencia.

Se deben esperar al menos 5 segundos para iniciar el primer movimiento.

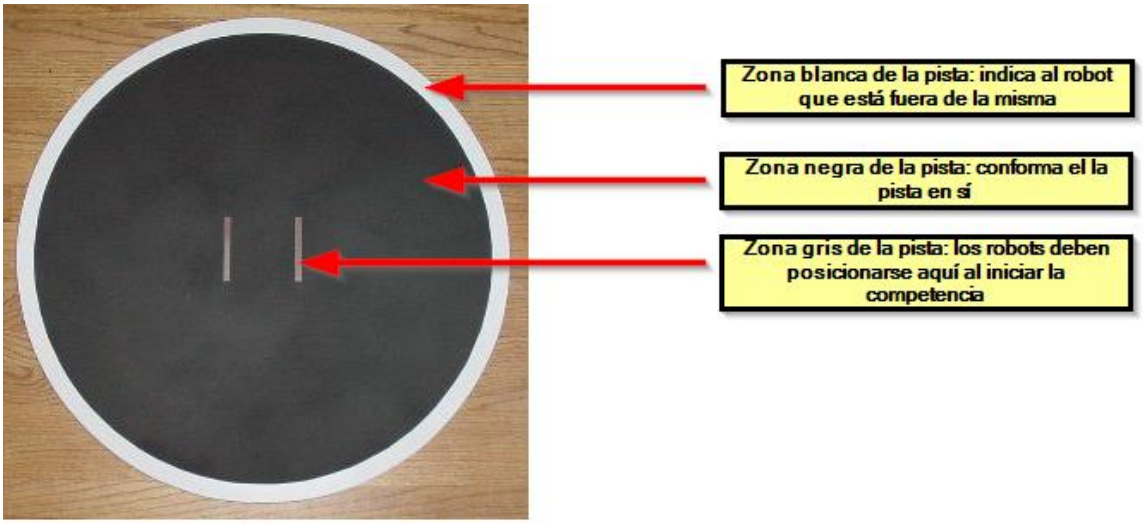

Figura 1: pista del sumo de robots.

INSTALACIÓN

Requisitos

Se debe contar con una computadora con un sistema operativo Windows XP o versión superior. El fabricante también provee drivers e IDE compatibles con MAC OS X.

El robot opera con 3 pilas AA y usa una conexión mediante un cable mini USB para la entrada al robot y estándar para conectar en la computadora.

Instalación de drivers

Para empezar a desarrollar con el robot, es necesario primero instalarse los drivers provistos por Robotgroup dentro del DuinoPack incluido en el CD, dentro de la carpeta **"Entorno"**, como alternativa también se pueden descargar del siguiente enlace: http://www.robotgroup.com.ar/archivos/software/Duinopack-v1-3-win.zip

Antes de encender el robot, hay que asegurarse que tenga un jumper conectado en las patitas para CDC, tal como se muestra en la figura 2, en caso contrario, no se podrá subir el código al robot.

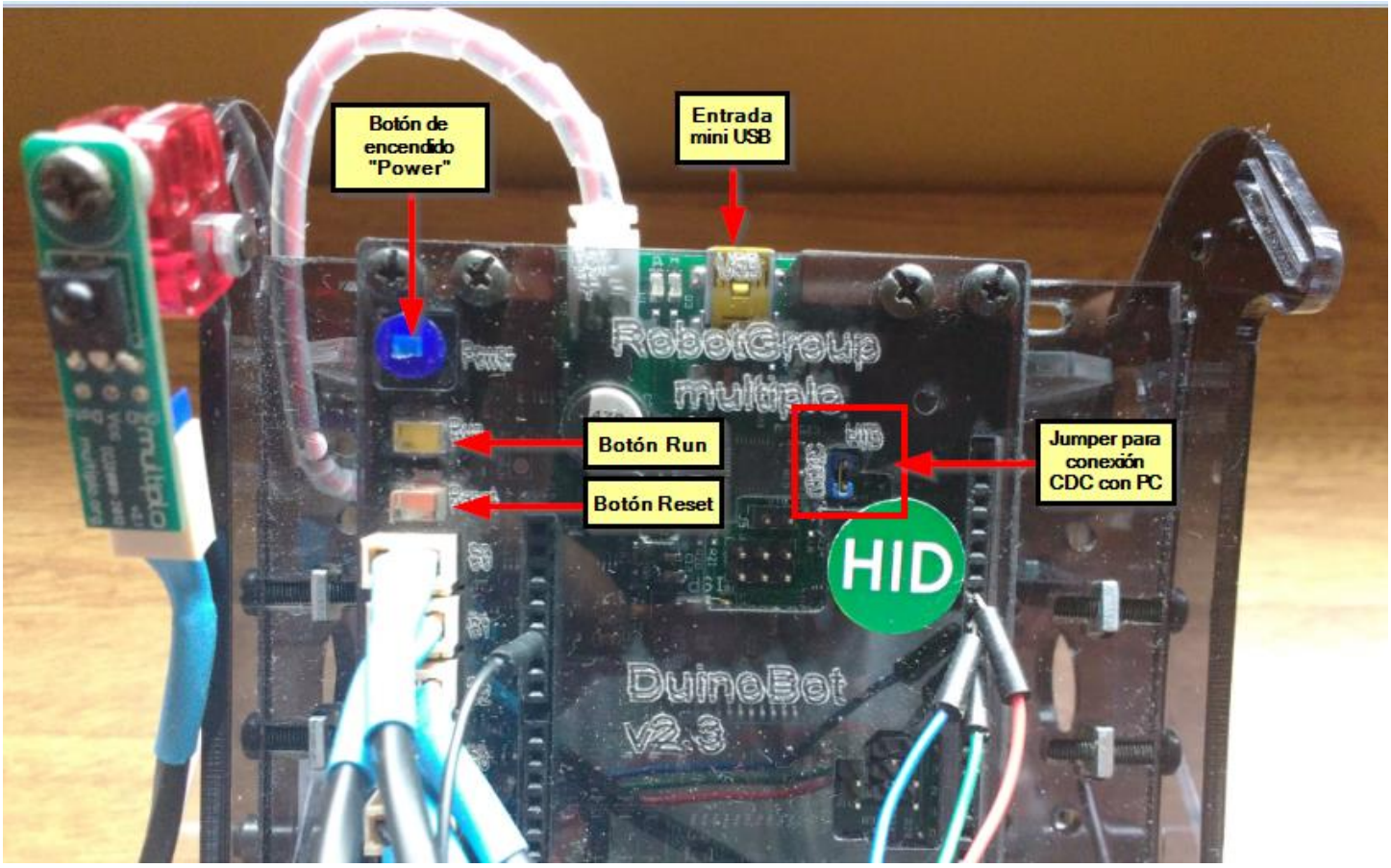

Figura 2: conexión de jumper a patas CDC

Conectamos el cable **mini usb** al robot y el otro al conector a la computadora. Asegurarse que tiene las pilas insertadas y encender el robot.

Una vez realizado el paso anterior, esperamos que el sistema operativo busque el driver del robot y lo instale. Es posible que el sistema instale los drivers equivocados en un sistema Windows, y quede en un estado en que

no pueda usarse, como se ve en la figura 3, por lo que es recomendable verificar dentro del Administrador de dispositivos si se instaló bien.

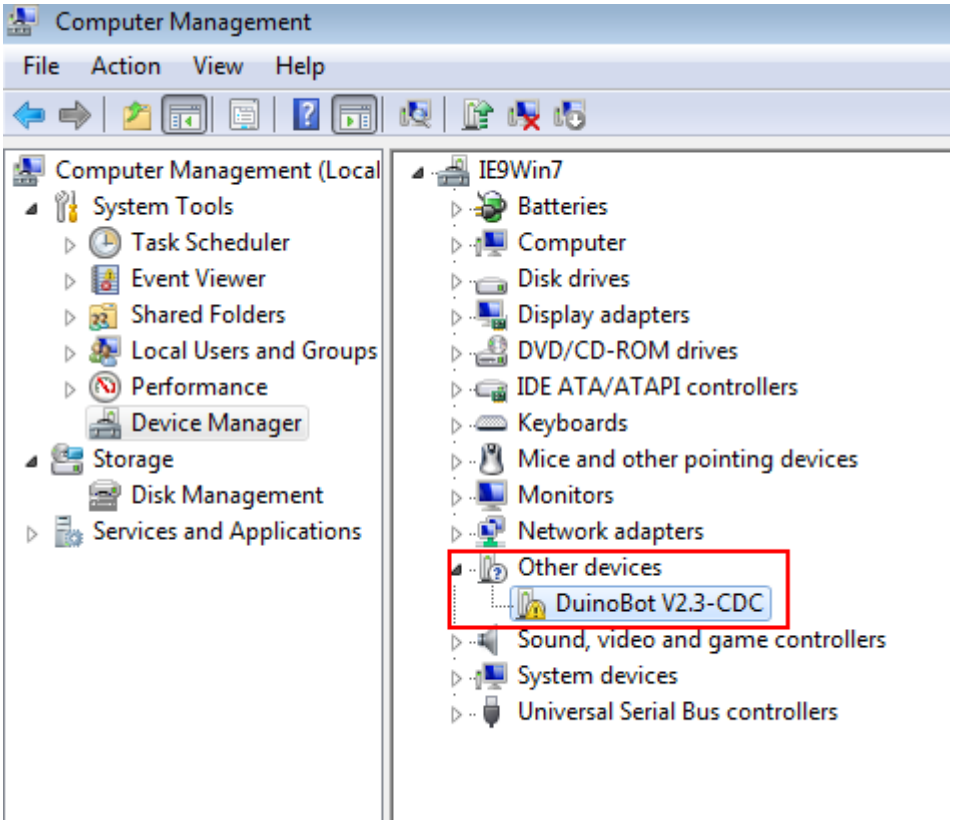

Figura 3: driver instalado incorrectamente

Para solucionar este problema, vamos al **Administrador de dispositivos → Otros dispositivos**, click derecho y **"Actualizar controlador"**. Elegir la instalación manual y buscar dentro de los drivers provistos anteriormente, la carpeta **Entorno → Duinopack → drivers.**

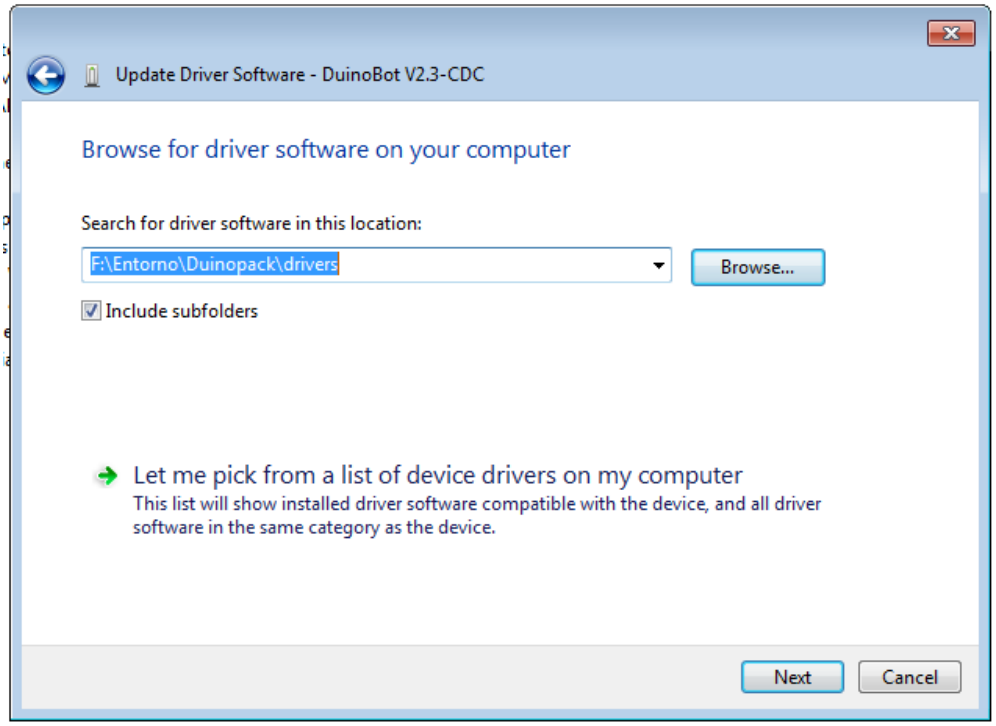

Figura 4: instalación manual de los drivers en Windows

Los drivers incluidos no están firmados, por lo que saltará la siguiente advertencia, clickear en Instalar de todos modos para seguir con la instalación.

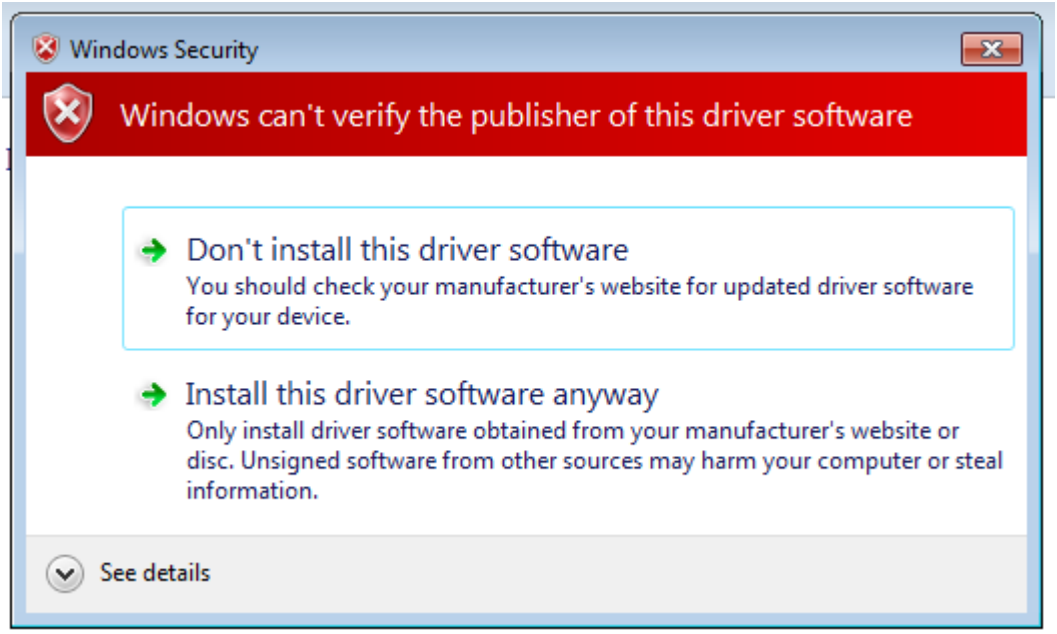

Figura 5: drivers no firmados

Una vez finalizado el proceso, verificar en el **Administrador de dispositivos** que el robot quedó correctamente instalado en la computadora. Deberá tener asignado alguno de los puertos **COM**.

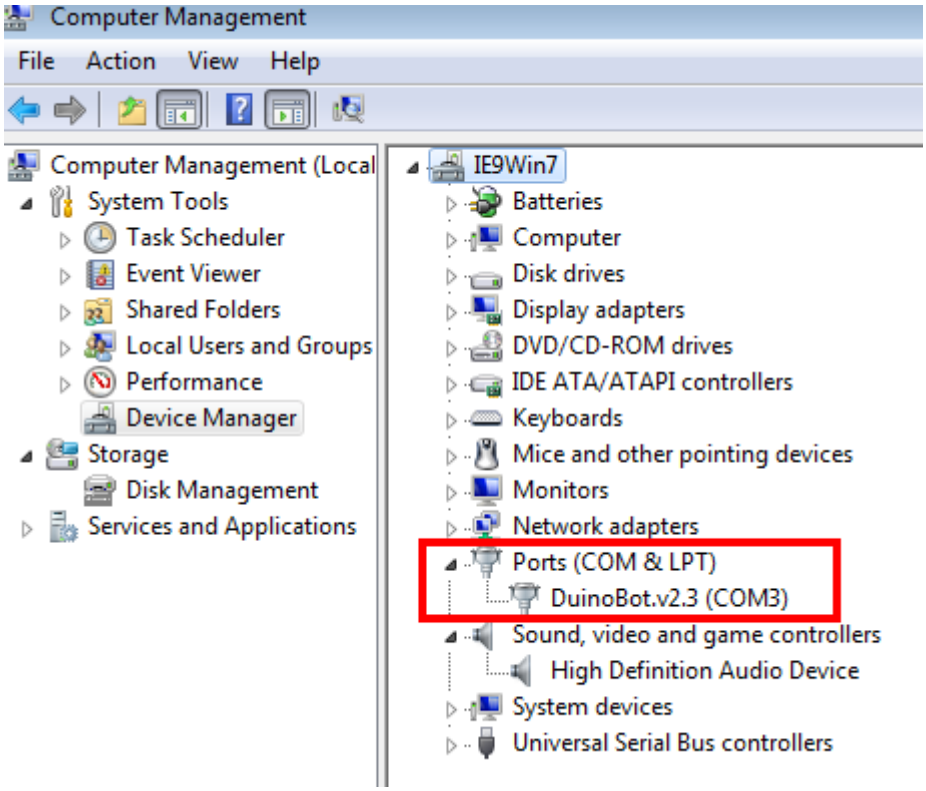

Figura 6: Robot instalado en el puerto COM3

Importante: es posible que en sistemas Windows 8 en adelante no deje instalar los drivers debido a que no se encuentran firmados, para esto, reiniciar el sistema con opciones de arranque que permitan instalar drivers no firmados, siguiendo estos pasos:

1. Desplegamos el menú que aparece al acercar el cursor a una de las esquinas derechas de la pantalla y pulsamos en Configuración:

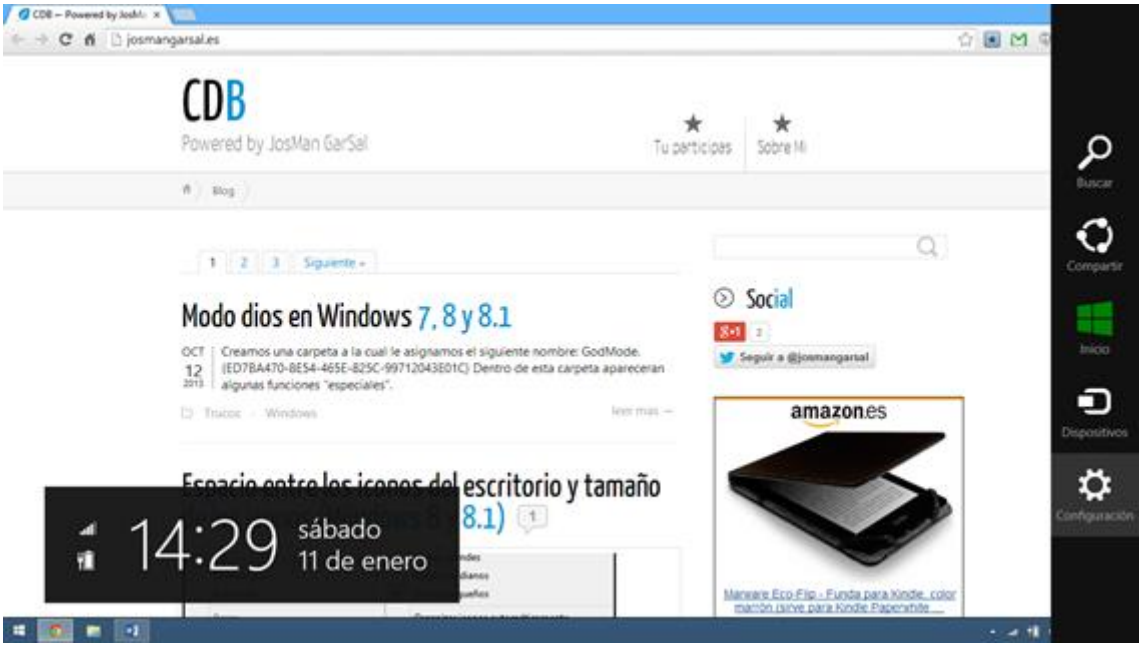

Figura 7: menú lateral derecho de Windows 8+

2. Se nos desplegará una barra en la derecha con varias opciones, pulsamos sobre **Cambiar configuración de PC** en la parte inferior.

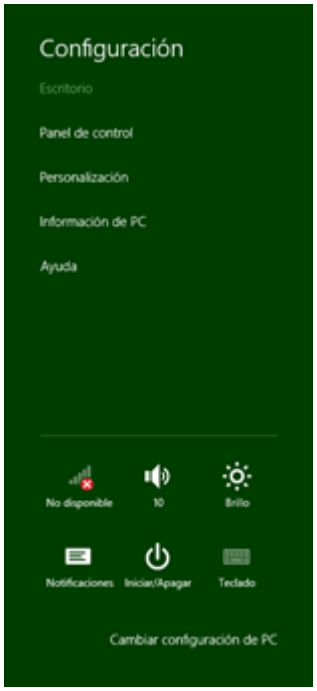

Figura 8: menú Configuración

3. Se abrirá una aplicación Modern UI para configurar la PC, en el menú de la izquierda la opción que buscamos es **Actualizar y recuperar**:

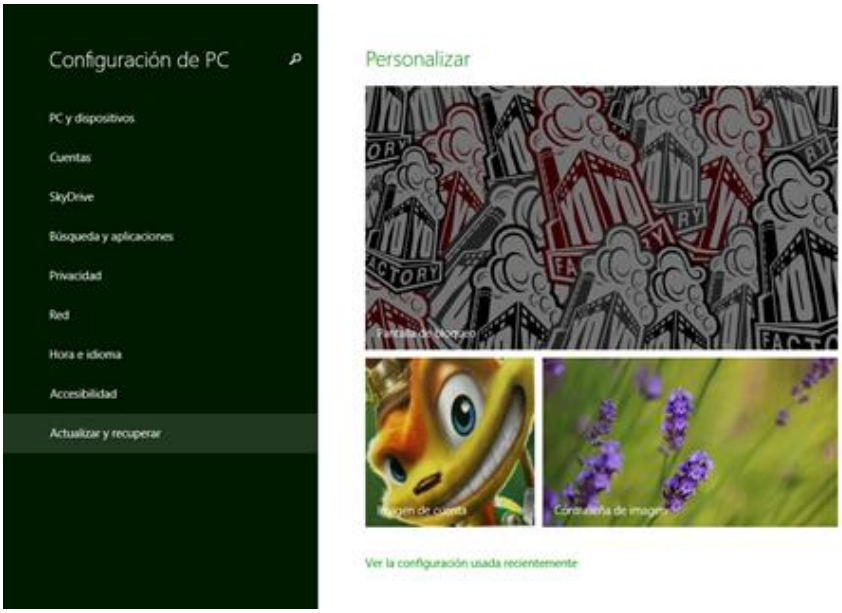

Figura 9: Opción Actualizar y Recuperar

4. Ahora pulsamos en **Recuperación**, en el menú de la izquierda y en el apartado **Inicio avanzado** pulsamos en **Reiniciar ahora**. El sistema se reiniciará y entrará en el menú de inicio avanzado.

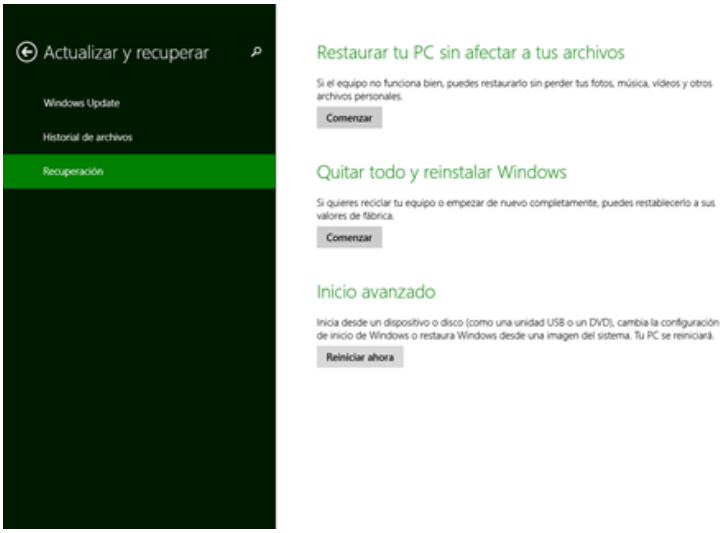

Figura 10: Opciones de recuperación

5. Ahora pulsaremos en **Solucionar problemas**.

Figura 11: Solucionar problemas

6. En la siguiente pantalla pulsamos sobre **Opciones avanzadas**.

Figura 12: Opciones avanzadas

7. Y por último, en la pantalla que aparece de **Opciones avanzadas**, pulsamos sobre **Configuración de inicio**.

Figura 12: Configuración de inicio

8. Nos aparecerá una nueva pantalla indicando las características que podremos modificar. Pulsar en **Reiniciar** para acceder al menú de configuración.

Figura 13: Opciones de la configuración de inicio

9. Para finalizar, pulsamos la tecla 7 que es la opción que nos interesa. El sistema se reiniciará y ya podremos instalar drivers no firmados.

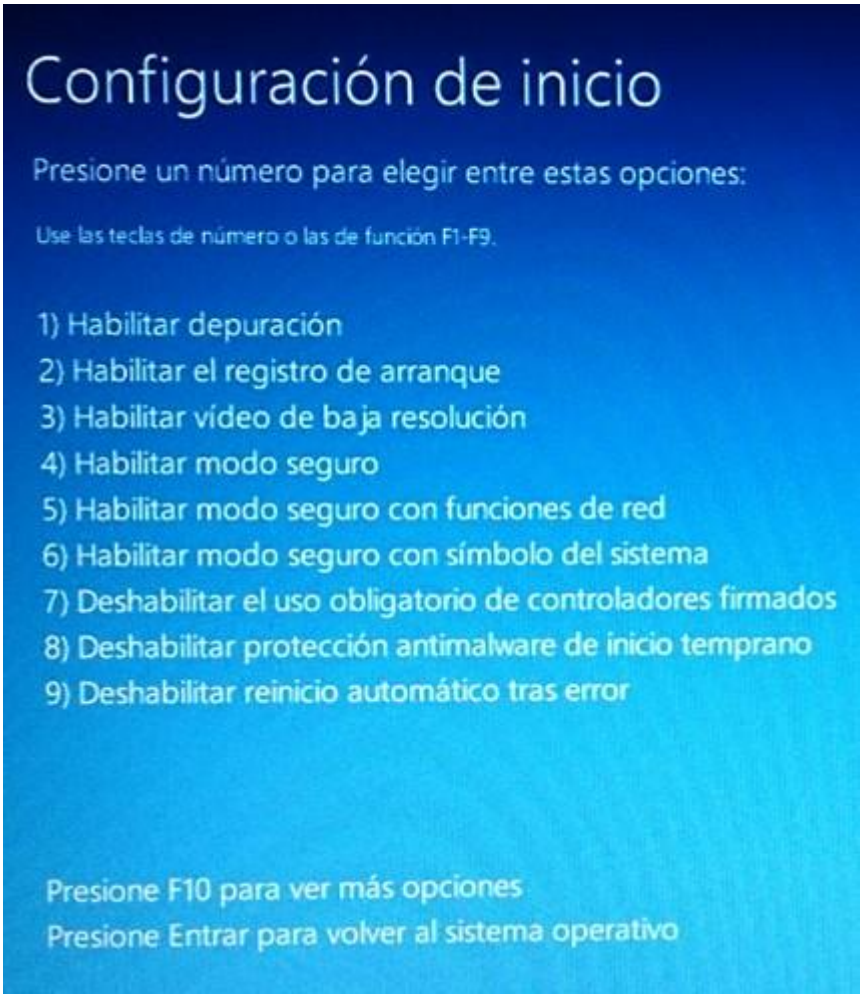

Figura 14: Reinicio para deshabilitar el uso obligatorio de controladores firmados

10. Una vez reiniciado el sistema, seguir los pasos de instalación de drivers mencionados al principio.

ENTORNO DE DESARROLLO “ARDUINO”

En el CD adjunto, se encuentra el entorno utilizado para el desarrollo del algoritmo de Sumo de robots.

Dirigirse a la carpeta **Entorno → DuinoPack** y ejecutar el archivo “**arduino.exe**”.

El IDE trae varios ejemplos, como estado de la batería del robot, manejo del control remoto, etc. Para acceder a ellos nos dirigimos a **Archivo →Multiplo → Ejemplos**.

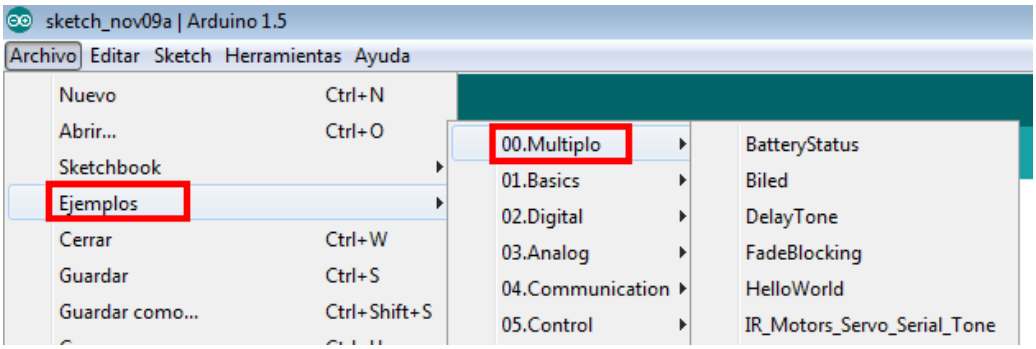

Figura 15: Ejemplo para Multiplo N6

Para poder cargar el programa tenemos que configurarlo con la placa que posee el robot. Para ello nos dirigimos a **Herramientas → Tarjeta**, y elegimos la opción que dice **Duinobot.v2.3.CDC W/Atmega 1284**.

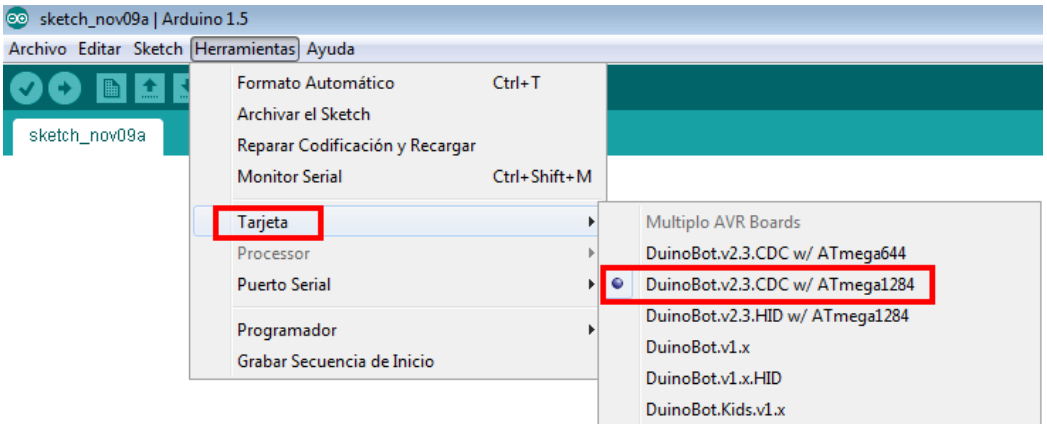

Figura 16: Elección de tarjeta

Luego de seguir los pasos anteriores y haber conectado el robot, vamos a **Herramientas → Puerto Serial**, y nos fijamos que el puerto donde está conectado el robot esté tildado, en caso de no estarlo, lo tildan.

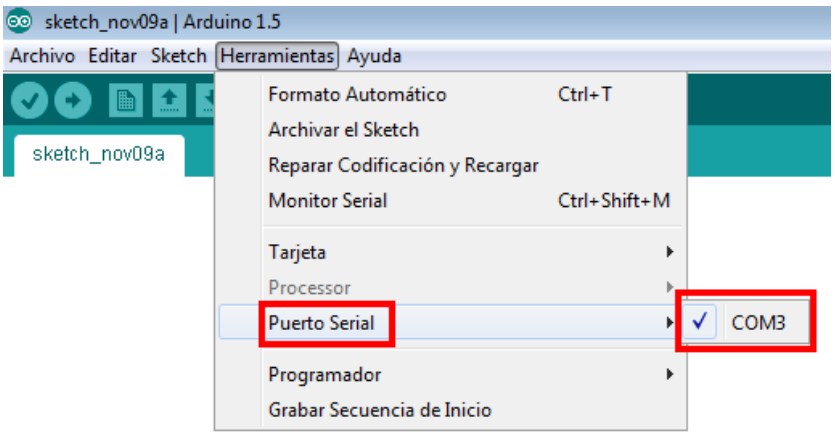

Figura 16: Elección del puerto serial

Test código de ejemplo

Siguiendo los pasos anteriores, se puede verificar que todo se haya instalado correctamente, instalando algún código de ejemplo en el robot. Un ejemplo bastante sencillo es el "**DelayTone**", el cual prueba el módulo de sonido del robot.

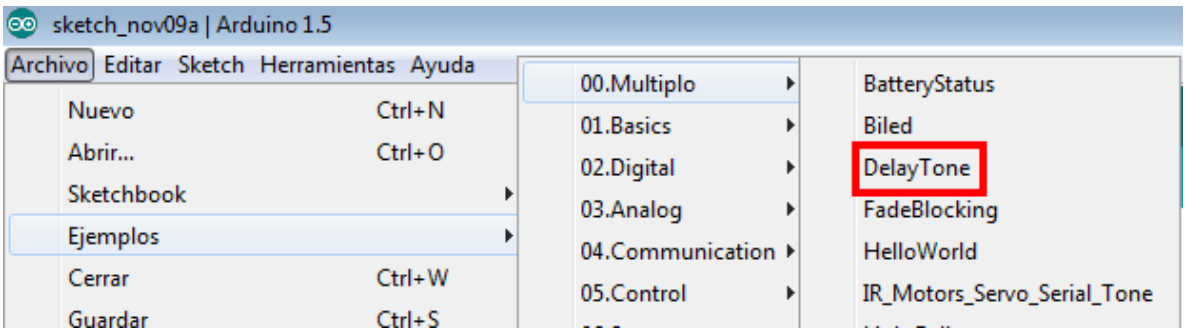

Figura 17: Ejemplo DelayTone

Una vez elegido el algoritmo de ejemplo, se abrirá otra ventana que contiene el código fuente, para cargarlo al robot, ejecutar los siguientes pasos:

1. **Verificar**: esta opción permite verificar que el código escrito compile correctamente. En caso que tenga algún error, lo mostrará en la consola inferior.

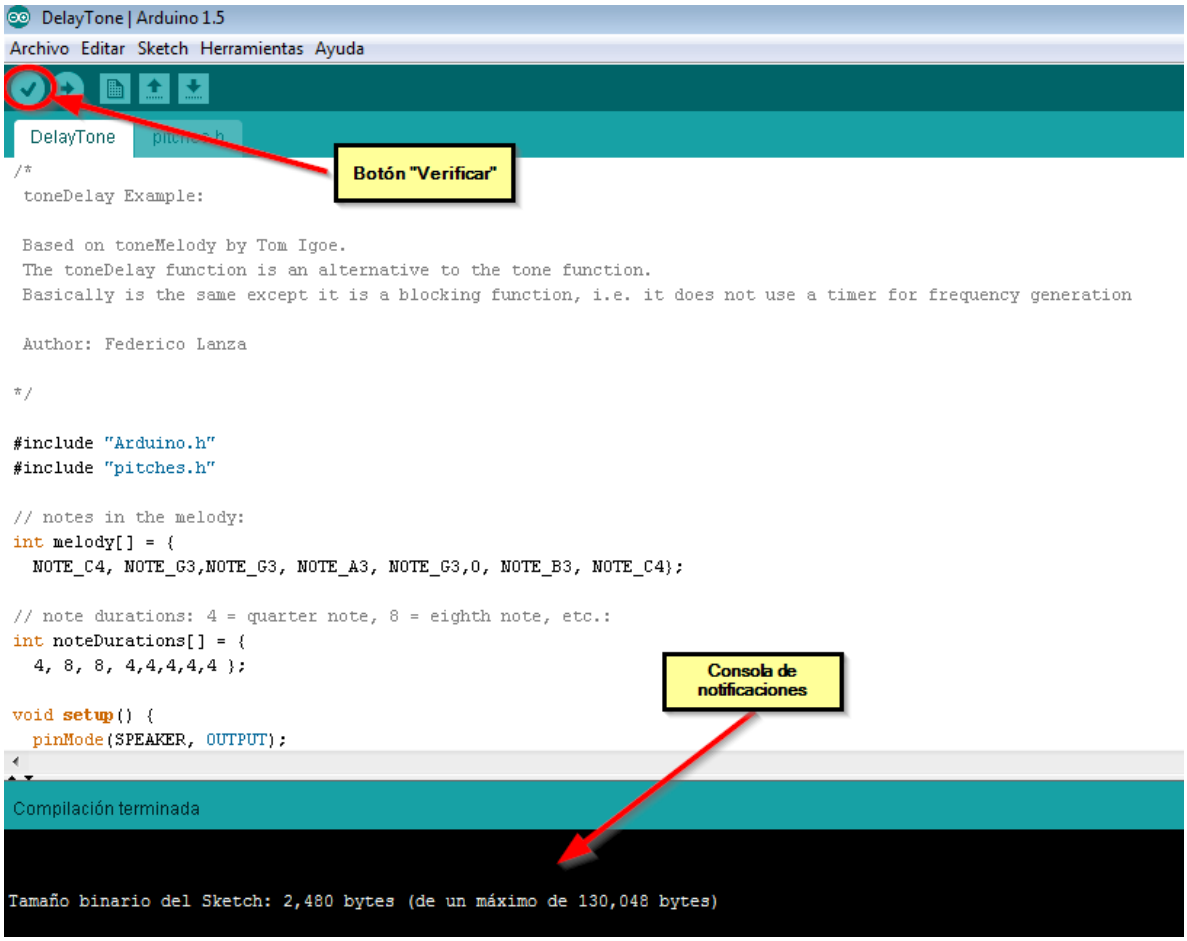

Figura 18: Verificar código

2. **Cargar**: una vez verificado que el código compile correctamente, se puede proceder a cargar el programa en el robot, para ello hacer click en "Cargar" esperar a que el proceso finalice.

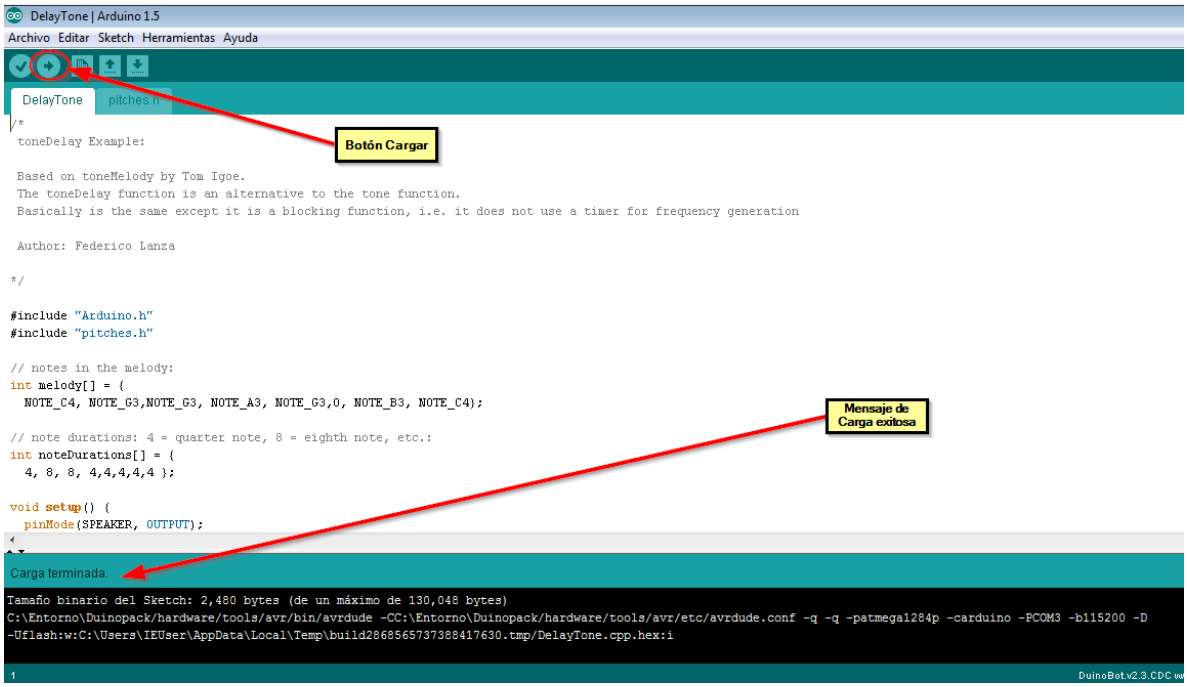

Figura 19: carga del programa en el robot

3. Para empezar a correr el programa en el robot, solo basta con apretar el botón "**Run**", y para detenerlo pulsar el botón "**Reset**".

Monitor serial

Dentro del entorno de desarrollo **Arduino**, también es posible imprimir líneas hacia el monitor que trae integrado. Esto es muy útil para depurar código, ver los valores que devuelven los sensores, etc. Para esto, hay que cargar el código al robot introduciendo lo que se quiere imprimir con la sentencia **Serial.println("texto o variable");**

Uno de los ejemplos que trae el IDE para probar el monitor serial, es el llamado "**TestSensor**", el cual hace una lectura del puerto **A1** e imprime los valores leídos. En el caso del robot del grupo, el sensor derecho está conectado al puerto **A1**.

Una vez cargado el código, ir a **Herramientas → Monitor Serial**. Luego de este paso, es necesario presionar el botón **"Run"** del robot para que empiece a correr el código cargado dentro de él.

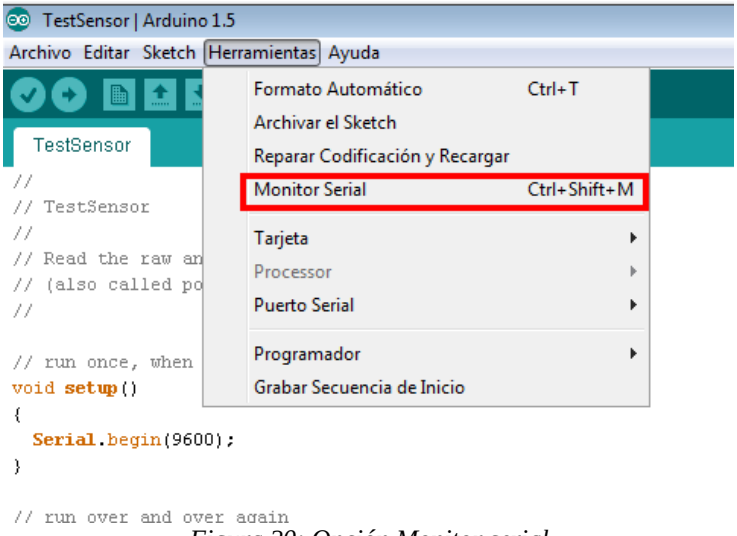

Figura 20: Opción Monitor serial

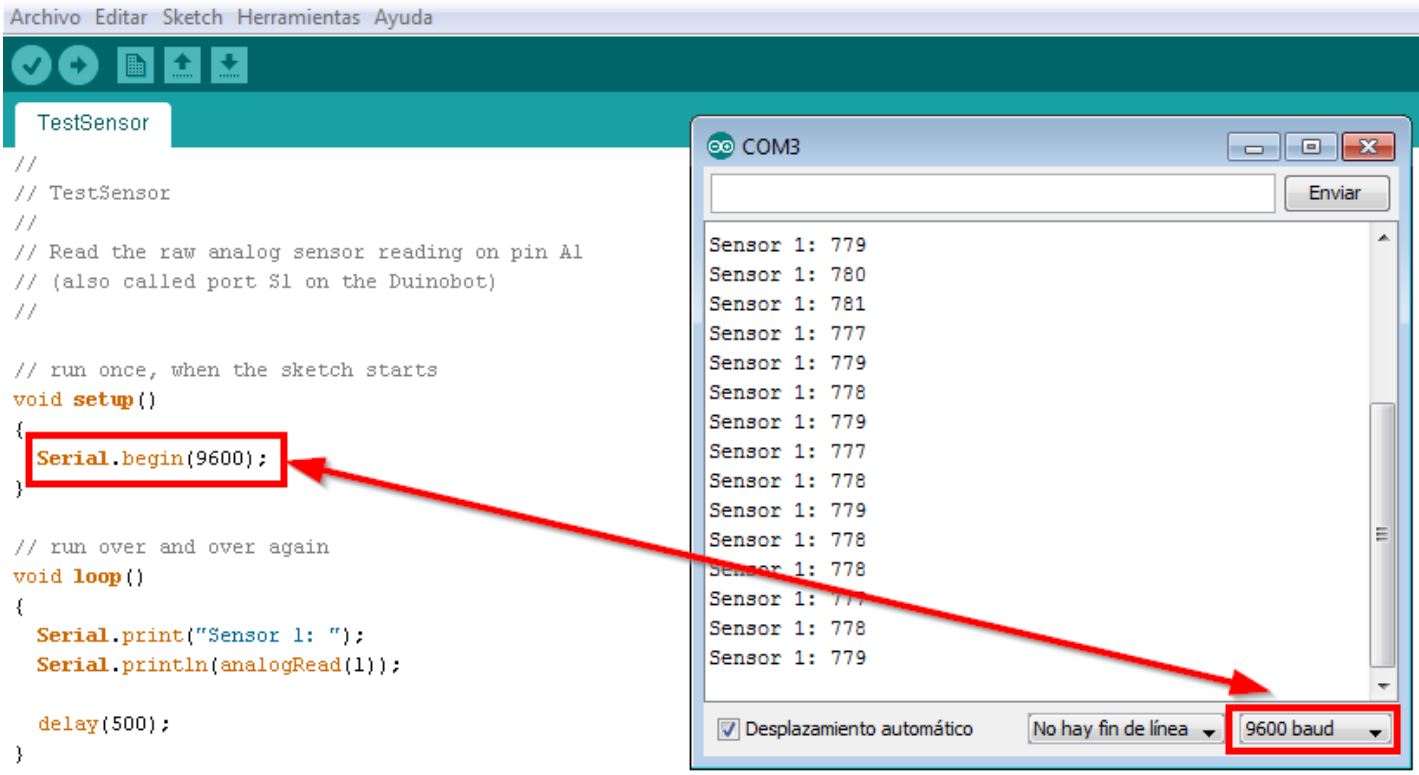

Figura 21: ventana del monitor serial, debe estar configurado con los mismos bauds que lo codificado

DESARROLLO DE SUMO DE ROBOTS

Antes de detallar el programa realizado, es importante remarcar que hay dos funciones principales que deben existir siempre en cualquier programa, la primera es **void setup()** y la segunda **void loop()**

1. **setup:** en esta función se pueden inicializar variables, setear valores, etc. al comenzar el programa, tales como los baudios a utilizar.
2. **loop:** está función, tal lo indica su nombre, es el loop en el cual el programa ejecutará las rutinas que le indiquemos, por ejemplo: hacer avanzar el robot, leer el estado del control remoto, etc. Esta función se ejecuta reiteradamente.

Funciones principales

Como parte del trabajo para la **Expocarreras 2016**, se desarrolló el siguiente algoritmo para la demostración. El mismo abarca 4 funcionalidades principales, las cuales se detallan a continuación:

1. Moverse

Lo que hace esta función es recorrer la pista sin salirse del borde blanco, esto se logra con los tres sensores infrarrojos que están instalados en el robot, que miden el color cada vez que hace algún movimiento, como por ejemplo, avanzar, retroceder o girar (tiene tres giros 90, 180 y 360 grados), para que no se vaya de la misma. El botón del control remoto que permite elegir esta opción es el número 9.

2. Atacar

En este algoritmo se busca un objetivo para sacarlo de la pista, esto se logra mediante el sensor de ultrasonido instalado en el robot, que mide la distancia a la que se encuentra el adversario. La distancia máxima de cercanía entre ellos es de 3000 (cuando el oponente se encuentra pegado al sensor del robot que está atacando), la mínima indicada por nosotros para el alcance de búsqueda es de 30. El botón del control remoto que permite elegir esta opción es el número 7.

3. Defenderse

El módulo de defensa consiste en escapar cada vez que encuentra un objetivo cerca a través de la medición hecha por parte del sensor de ultrasonido mencionado anteriormente en la función de atacar. El botón del control remoto que permite elegir esta opción es el número 8.

4. Aumentar/disminuir la velocidad

Por medio de esta función se logra aumentar o disminuir la velocidad del robot cambiando el valor de la velocidad de los dos motores que posee el mismo. Al robot le agregamos tres velocidades: baja, media y alta. La velocidad baja es de 60, la media de 75, y la alta de 90. El botón del control remoto que permite elegir estas opciones son: el número 1 para la velocidad alta, el 2 para la media, y el 3 para la baja

El código para las funciones explicadas anteriormente, es el siguiente:

```
/*
       El ciclo continuo del robot. Aquí se lee constantemente la orden dada desde el control remoto.
       Según la orden especificada, se cambiará la velocidad del robot o aplicará una estrategia.
       En cualquier caso, el robot estará moviendose.
*/
void loop() {

    sensorDerecho = analogRead(1);
    sensorIzquierdo = analogRead(2);
    sensorTrasero = analogRead(4);

    int codigoControlRemoto = codigoControlIR.getIRRemoteCode();

    switch (codigoControlRemoto) {
    case 1:
        velocidadActiva = velocidadMaxima;
        break;
    case 2:
        velocidadActiva = velocidadMedia;
        break;
    case 3:
        velocidadActiva = velocidadBaja;
        break;
    case 7:
        estrategiaActiva = "ataque";
        break;
    case 8:
        estrategiaActiva = "defensa";
        break;
    case 9:
        estrategiaActiva = "pasivo";
        break;
    }

    moverse(velocidadActiva, estrategiaActiva);
}



/*
 *  Parametros: velocidad a la que va a funcionar, estrategia: "ataque", "defensa"
 *  Comportamiento: el robot va a avanzar en linea recta mientras nos estemos acercando al objetivo
detectado,
 *    en caso de alejarnos respecto a la ultima distancia grabada sale de la funcion
 */
void moverse(int velocidad, String estrategia) {
    contadorGeneral++;
    int sobreLineaBlancaFlag = 0;
    sensorDerecho = analogRead(1);
    sensorIzquierdo = analogRead(2);
    sensorTrasero = analogRead(4);

    if(((sensorDerecho<refraccionColorNegroDerecho)  && (sensorIzquierdo<refraccionColorNegroIzquierdo) &&
!flagSaliendoDelBorde)) {
        if(objetivoCerca() && estrategiaActiva != "pasivo") {
            if (estrategiaActiva == "ataque") {
                analogWrite(ledRojo,  255);
                analogWrite(ledVerde,  0);
                analogWrite(ledAzul,  255);
                avanzar(100);
            }
            else if( estrategia == "defensa") {
                analogWrite(ledAzul,  255);
                analogWrite(ledVerde,  255);
                analogWrite(ledRojo,  0);
                girarDerecha(velocidadMaxima, tiempoGiro90Grados);
            }
        }
        else {
            analogWrite(ledVerde,  255);
            analogWrite(ledAzul,  0);
            analogWrite(ledRojo,  0);
            if (((contadorGeneral % frecuenciaBusquedaContrincante) == 0) && estrategiaActiva == "ataque") {
                girarHastaEncontrarDireccionObjetivo(velocidad);
            }
            else {
                avanzar(velocidad);
            }
        }
    }
```

```
    else {
        flagSaliendoDelBorde = true;
        sobreLineaBlancaFlag = sobreLineaBlanca();
        analogWrite(ledRojo,  255);
        analogWrite(ledVerde,  0);
        analogWrite(ledAzul,  0);
        //contraresto la inercia del robot para que no se vaya de la pista
        retroceder(70);
        delay(400);
        frenar();
        //

        int tiempo;
        int flag = random(0,2);

        if( flag == 0 ) tiempo = tiempoGiro90Grados;
        else if ( flag == 1 ) tiempo = tiempoGiro180Grados;

        if (sobreLineaBlancaFlag == 1) { // con la rueda derecha sobre la linea blanca
            if(velocidad > 61 ) {
                retroceder(100);
                delay(150);
                if(girarIzquierda((velocidad*2)/3, tiempo)) {
                    flagSaliendoDelBorde = false;
                }
            }
            else {
                if(girarIzquierda(velocidad, tiempo)) {
                    flagSaliendoDelBorde = false;
                }
            }
            frenar();
        }
        else if (sobreLineaBlancaFlag == 2) { // con la rueda izquierda sobre la linea blanca
            if(velocidad > 61 ) {
                retroceder(100);
                delay(150);
                if(girarIzquierda((velocidad*2)/3, tiempo)) {
                    flagSaliendoDelBorde = false;
                }
            }
            else {
                if(girarIzquierda(velocidad, tiempo)) {
                    flagSaliendoDelBorde = false;
                }
            }
            frenar();
        }
        else {
            // si por algun motivo sigo tocando la linea blanca retrocedo para asegurarme que no se va a
salir
            retroceder(100);
            delay(300);
            frenar();
            //
            if(velocidad > 61 ) {
                girarSobreEje((velocidad*2)/3, tiempoGiro90Grados);
            }
            else girarSobreEje(velocidad, tiempoGiro90Grados);
            frenar();
            flagSaliendoDelBorde = false;
        }
    }
}
```

Funciones auxiliares

A continuación se listan otras funciones auxiliares desarrolladas para dar soporte a las funciones principales expuestas anteriormente:

- ✓ Girar a la derecha o a la izquierda
- ✓ Girar 360 grados
- ✓ Girar 180 grados
- ✓ Girar 90 grados
- ✓ Leer sensor de la derecha, izquierda y atrás
- ✓ Detectar objetivo cerca
- ✓ Frenar
- ✓ Detectar blanco
- ✓ Girar hasta encontrar la dirección del objetivo

Comportamiento del robot (leds RGB)

- Al iniciar, el robot espera 5 segundos antes de hacer el primer movimiento.
- Cuando **empieza a avanzar**, el led **RGB** toma color **blanco**.
- Mientras está **dentro de la pista y andando**, el led RGB del robot se muestra en color **verde**.
- Si el robot se encuentra **pisando la franja blanca**, el led RGB toma el color **rojo** hasta acomodarse dentro de la pista nuevamente.
- Dentro del **modo defensivo**, si detecta un **objetivo**, el led RGB se torna de color **verde + azul**.
- En el **modo de ataque**, cuando detecta al **objetivo**, el led cambia a color **rojo + azul**.

CONEXIONES

En la figura 22 y 23, se grafica cada componente conectado al robot y en la tabla 1 se detallan las funciones programadas para el control remoto y en la tabla 2 se especifican los puertos y pines conectados.

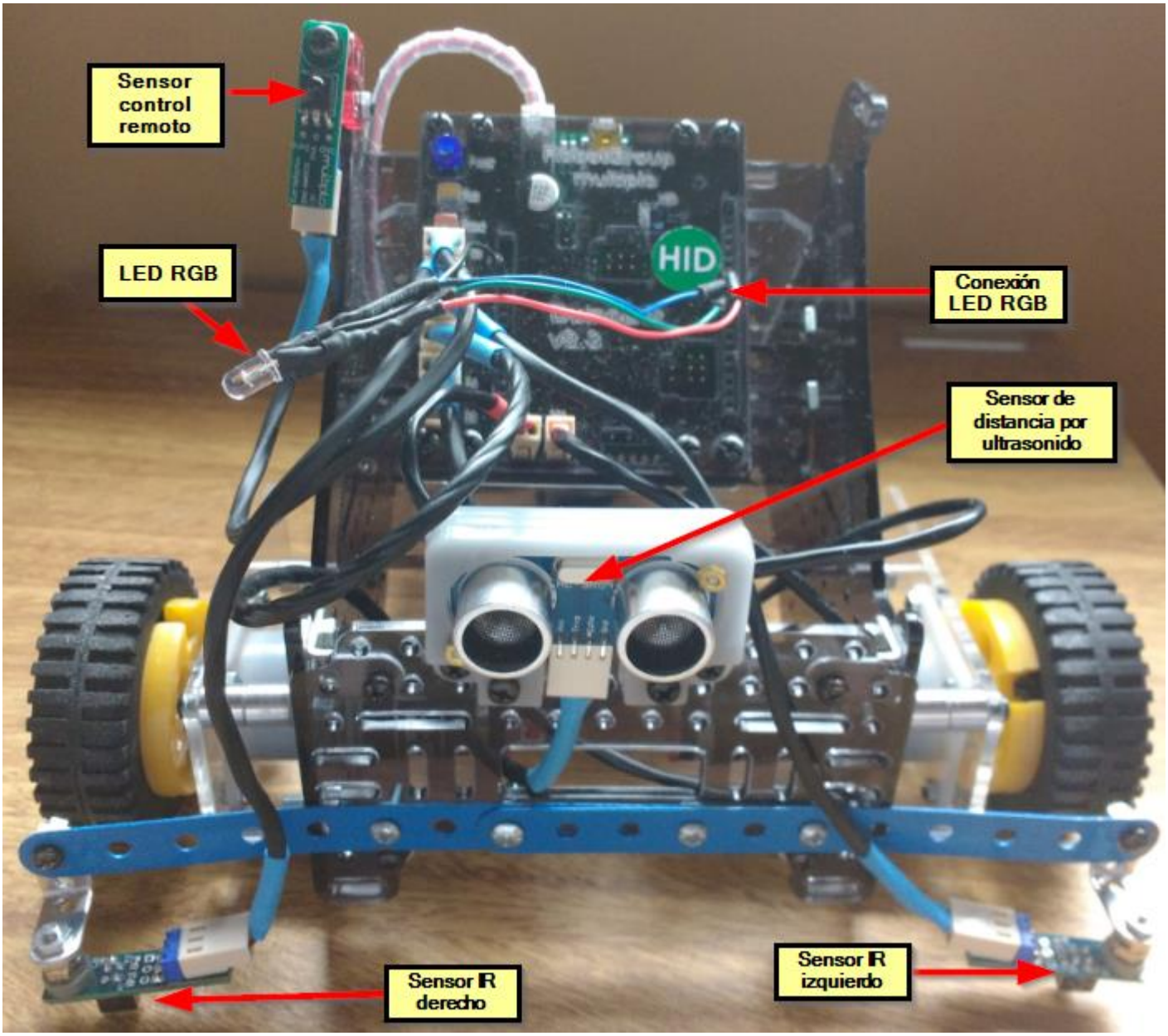

Figura 22: componentes conectados (frente)

Función	Botón
Velocidad baja (55)	3
Velocidad media (75)	2
Velocidad máxima (90)	1
Atacar	7
Defenderse	8
Moverse (pasivo)	9

Tabla 1: funciones del control remoto

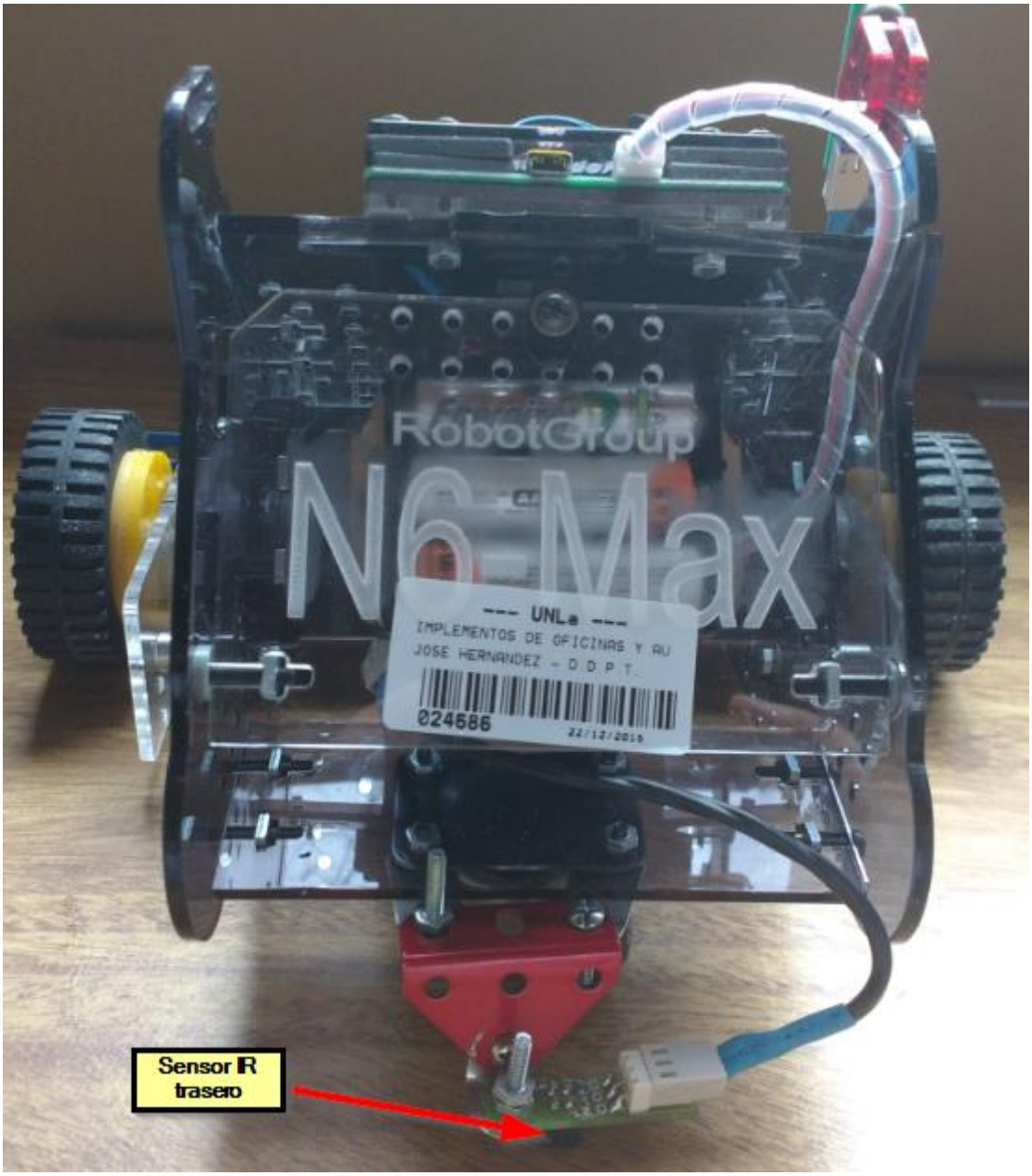

Figura 23: componentes conectados (trasero)

Componente	Puerto	Pin
Sensor IR izquierdo	A2 (S2)	-
Sensor IR derecho	A1 (S1)	-
Sensor IR trasero	A4 (S4)	-
Sensor control remoto	A0 (S0)	-
Sensor distancia ultrasonido	A3 (S3)	-
LED RGB (Entrada)	-	25
LED RGB (Salida): Rojo	-	11
LED RGB (Salida): Azul	-	10
LED RGB (Salida): Verde	-	9

Tabla 2: Conexiones