Brutus v1.0
Robot para experimentación y aprendizaje

Alejandro Alonso Puig – mundobot.com
Febrero 2004

Introducción

Brutus nació de la idea de construir un robot modular, capaz de ir creciendo en capacidad con el tiempo y en el que poder probar nuevos módulos experimentales.

La versión 1.0 que se muestra en este informe técnico se refiere a la versión más sencilla de la implementación del robot, en la que dispone de control remoto de navegación, así como de un sencillo programa de navegación autónoma.

Algunas características básicas que definen a Brutus v 1.0 son las siguientes:

- Medidas (cm): 120 x 85 x 50
- 4 ruedas
- Sistema de tracción trasera mediante dos motores-reductores de buen par
- Amortiguación trasera
- Dirección delantera controlada por servomotor de buen par
- Sonar delantero orientable
- Parachoques-bumper delantero con detección electrónica de golpes
- Ordenador de abordo (Pentium) con programa de control en Visual C++
- Módulo "Ruso" para la gestión de los dispositivos del robot
- Comunicación inalámbrica WIFI con otro ordenador externo al robot.
- 3 baterías de 12v
- Sistema íntegramente controlado por bus I^2C

En este trabajo se hará mención a todos los aspectos Mecánicos, electrónicos y de programación, así como conclusiones de la experiencia.

Mecánica

El chasis de Brutus consiste en una "cuna" de material plástico de 4 mm de espesor con refuerzo de barras de aluminio.

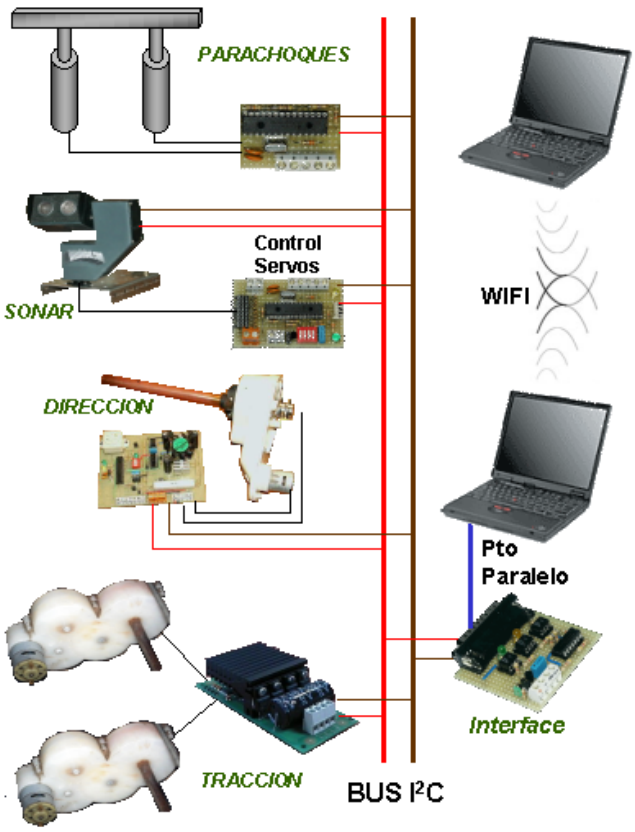
La parte trasera está formada por dos ruedas con sendos motores-reductores acoplados a las mismas y sistema de amortiguación del chasis.

La parte delantera está formada por la dirección, controlada por un servomotor de buen par. Adicionalmente está dotado de un parachoques de aluminio con sistema de amortiguación y absorción de golpes, sin efecto torsión en golpes no centrados, así como con sistema de doble conmutación para la detección electrónica de golpes.

El robot lleva un ordenador portátil como cerebro. Dicho ordenador está colocado en una estructura que permite un movimiento amortiguado del mismo.

Electrónica

Ante todo Brutus es un robot modular, diseñado para crecer. Los dispositivos electrónicos que utiliza Brutus están intercomunicados por un bus I^2C.



El cerebro del robot lo compone un ordenador portátil con procesador Pentium y dispositivo wireless WIFI.

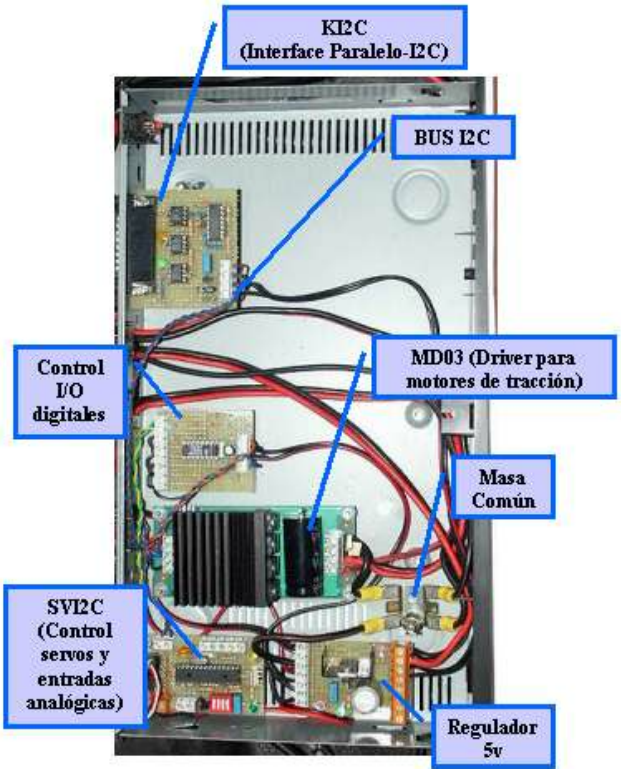
Dicho ordenador está conectado a un módulo denominado "Ruso", que actuará sobre los diferentes dispositivos del robot.

El Ordenador Portátil posee un programa en Visual C++ 6.0 que se ocupará de entablar la comunicación con el módulo "Ruso" y por tanto con todos los dispositivos del Robot.

En la siguiente sección hablaremos más detenidamente de este programa.

El módulo "Ruso" contiene fundamentalmente los siguientes dispositivos:

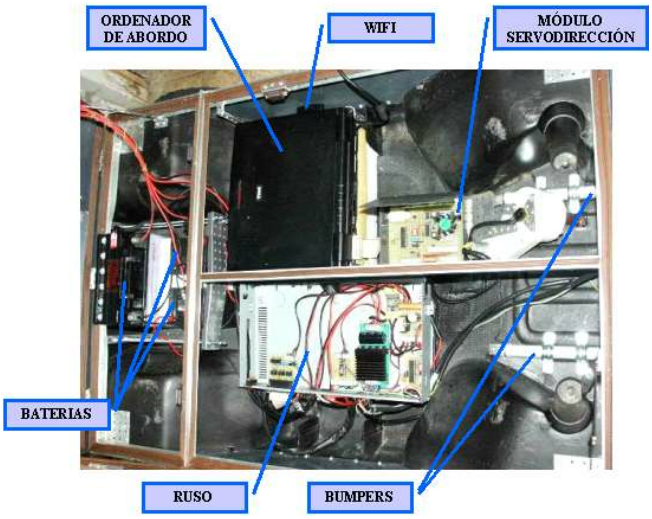
- Unidad de regulación a 5v
- Interface puerto paralelo – Bus I^2C (KI2C), que permite comunicar el PC con el resto de los dispositivos del robot
- Unidad MD03 de Devantech, que actúa de driver para el control de los motores de tracción
- Unidad SVI2C para el control de 8 servomotores y 5 entradas analógicas. En esta versión de Brutus solo se utilizan dos de las salidas de control de servos para controlar los movimientos del sonar, del que hablaremos un poco más adelante.
- Unidad de entradas/salidas digitales (8). Consiste simplemente en la utilización del chip PCF8575. En esta versión de Brutus solo se utilizan dos entradas para el control del estado del parachoques, que funciona como un doble bumper con resistencias pull-up.



Para obtener más información sobre los elementos que componen el módulo "Ruso" puede accederse a las siguientes páginas del autor:

- KI2C: http://mundobot.com/tecnica/ki2c/spki2c.htm
- MD03: http://mundobot.com/tecnica/md03/spmd03.htm
- SVI2C: http://mundobot.com/tecnica/svi2c/spsvi2c.htm

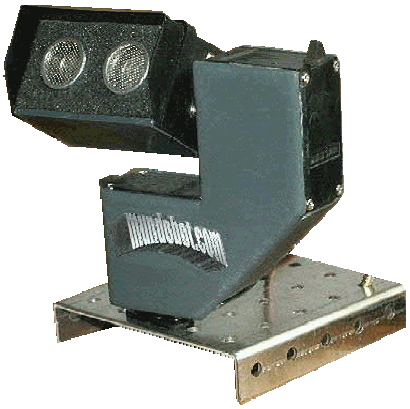
Fue mi sobrino Manuel, de 9 años, listo como él solo y con intensos deseos de convertirse en inventor, el que bautizó a este módulo como "Ruso". Al volver del colegio me veía a menudo ensimismado "pegándome" con los circuitos electrónicos para hacerlos comunicarse adecuadamente por bus I^2C. Un día me vio comprobando las señales del bus I^2C en el osciloscopio y me preguntó que qué hacía. Yo le explique que estaba intentando hacer que se entendiesen los diferentes circuitos entre sí. Él me preguntó que en qué idioma hablaban (¡niños!). Le dije que en Ruso. Se le abrió la boca de entusiasmo. Teníais que haberle visto. Cuando finalmente puse todos los circuitos que se interconectaban por bus I^2C en una caja negra y la metí en Brutus, él me preguntó "¿Qué es eso?. ¿El Ruso?". Obviamente le dije que sí y de ahí nació el nombre.



Vista del interior del robot

Adicionalmente al módulo "Ruso" existen los siguientes módulos:

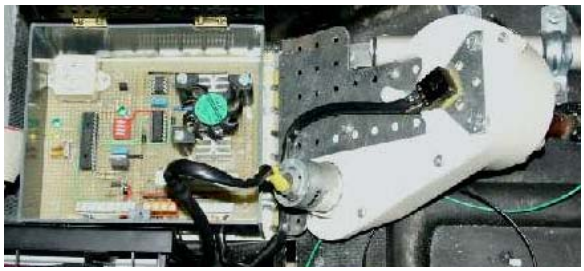
1. Un sistema de sonar orientable (eje X y eje Y) en la parte frontal del robot, gobernado también por bus I^2C, con un alcance de detección de hasta 6 metros.



Para obtener información completa sobre este módulo puede accederse a la página del autor: http://mundobot.com/tecnica/sonar/spsonar.htm

2. Un sistema de servomotor (SVD01) para el control de la dirección. Dicho módulo permite el control por bus I^2C de un motor de gran par controlándolo para que haga las funciones de servo. Adicionalmente permite obtener por bus I^2C medidas de posición, consumo y temperatura.

   Para obtener información completa sobre este módulo puede accederse a la página del autor: http://mundobot.com/tecnica/svd01/spsvd01.htm



Todo el conjunto está alimentado por tres baterías de 12v de plomo-ácido:

- Una para la alimentación de los motores de tracción, con una capacidad de 8Ah
- Una para la alimentación de la electrónica del robot de 4Ah
- Una para la alimentación del servomotor de la dirección de 4Ah

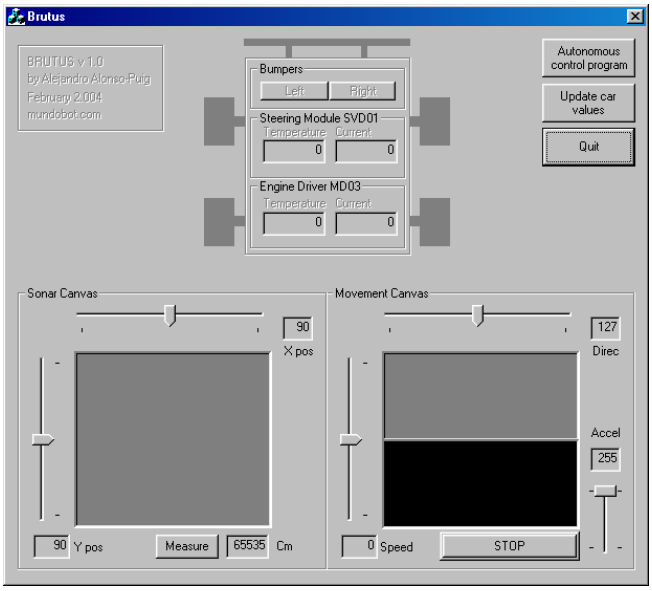
Adicionalmente el propio ordenador de control lleva su propia batería de Li-Ion

Programación

Para el control del conjunto completo se ha generado la aplicación "Brutus". Dicha aplicación se encuentra localizada en el ordenador de abordo. Mediante el enlace WIFI y la utilización del software de gestión remota de PCs VNC, se maneja el programa Brutus desde un PC externo a Brutus.

Utilización de la aplicación

El interface con el usuario consiste básicamente en un cuadro de diálogo como el que se muestra a continuación.



El cuadro de diálogo muestra una serie de controles que permiten manejar al robot de forma manual. Adicionalmente incluye un botón "*Autonomous control program*" que ejecuta el programa de control para navegación autónoma del robot.

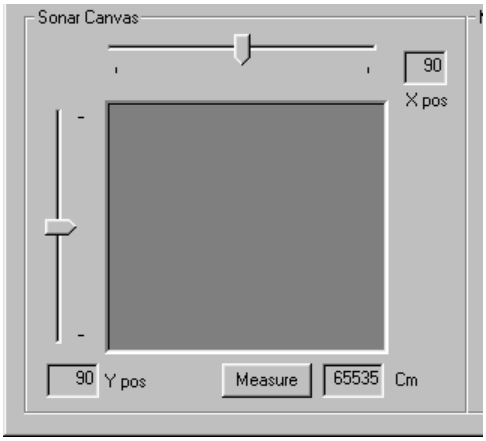
Control Manual

El control manual del robot es muy sencillo. Básicamente se hace desde tres áreas de la pantalla:

- "Canvas" de control del sonar
- "Canvas" de control de la tracción y dirección
- Área de valores de estado

Canvas de control del sonar

El Canvas de control del sonar permite mover el sonar frontal hacia donde deseemos con tan solo arrastrar por el mismo el ratón con el botón izquierdo pulsado.

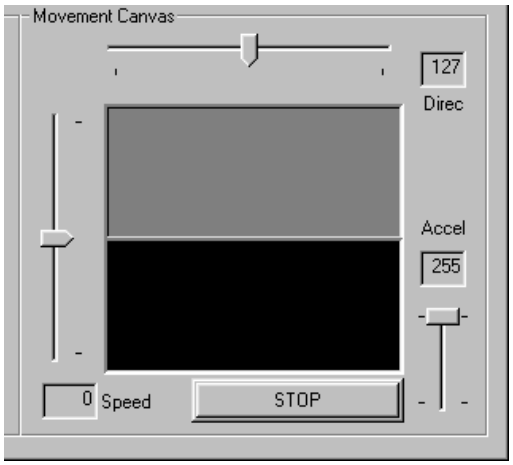


Los campos "*X pos*" e "*Y pos*" indican la posición del sonar en grados.
El campo "*Cm*" indica el valor de distancia al obstáculo más cercano medido por el sonar en centímetros. Dicho valor se va actualizando automáticamente según se mueve el ratón por el canvas, pero puede forzarse una medición en un momento dado pulsando el botón "*Measure*"

Canvas de control de la tracción y dirección

Al igual que en el canvas anterior, el control se hace mediante el movimiento del ratón sobre el mismo con el botón izquierdo pulsado. El área gris y el área negra indican los movimientos hacia delante a hacia atrás. De esta manera el ratón colocado en la frontera entre ambas secciones implicará un movimiento nulo del robot. Un desplazamiento hacia arriba en el área gris o hacia abajo en el área negra implicará un aumento de la

velocidad. La velocidad viene representada por un valor entre –240 y 240, siendo –240 la velocidad máxima hacia atrás y 240 la velocidad máxima hacia delante. El valor de la velocidad vendrá indicado en el campo "*Speed*".

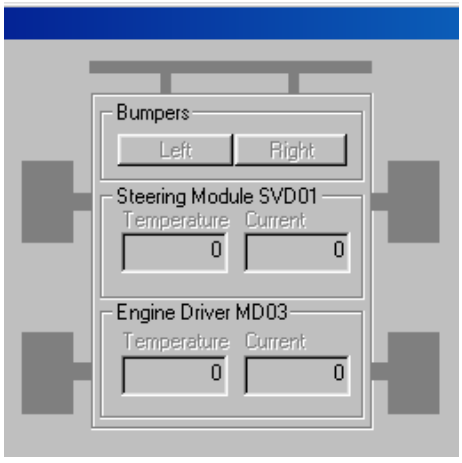


Por otra parte el movimiento del ratón hacia la derecha e izquierda del canvas, tanto en la zona gris como negra implicará un movimiento hacia derecha o izquierda respectivamente de la servodirección.

El control deslizante "*Accel*" permite establecer el valor de aceleración del robot, con lo que se pueden limitar los movimientos bruscos al cambiar la velocidad del mismo.

El botón "*STOP*" permite realizar una parada inmediata del robot.

Área de valores de estado



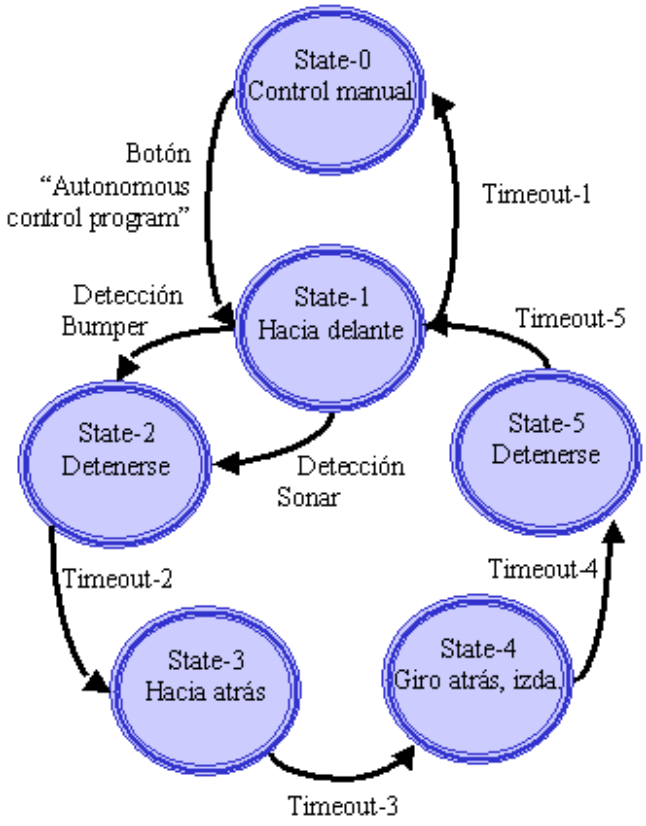
Esta área permite mostrar ciertos valores del estado del robot como son la temperatura y consumo eléctrico de los módulos de tracción y dirección o el estado de colisión del parachoques.

Los datos se actualizan manualmente con el botón "*Update car values*"

Control Autónomo

El control autónomo se activa simplemente mediante la pulsación del botón "*Autonomous control program*".

Este botón activa temporalmente un programa muy básico que hará moverse al robot hacia delante, midiendo constantemente la distancia a posibles obstáculos con el sonar. De encontrar un obstáculo hará una maniobra para evitarlo, tomando otra dirección. Dicho comportamiento se muestra a continuación como una máquina de elementos finitos:



Diseño de la aplicación

La aplicación se ha generado en Visual C++ 6.0. Utiliza una serie de librerías para el control de la tarjeta KI2C y poder así utilizar su bus I^2C, al que se encuentran conectados los demás elementos del robot.

Para más información sobre el manejo de la mencionada librería puede consultarse el trabajo del autor existente en la dirección: http://mundobot.com/tecnica/ki2c/spki2c.htm

El interface con el usuario consiste básicamente en un cuadro de diálogo asociado a una clase *CBrutusDlg* dentro de la cual se encuentran todas las funciones y variables utilizadas para el control del robot.

Los ficheros fuente de la aplicación contienen numerosos comentarios explicativos y pueden ser accedidos en la dirección: http://mundobot.com/projects/brutus/brutusprg.zip

No obstante a continuación explicaremos la estructura general de la aplicación.

Las direcciones que se han utilizado para los dispositivos I^2C se han establecido en las siguientes variables:

```
cSRF08Add =224;// Direccion I2C modulo sonar SRF08
cSVI2Cadd =118;// Direccion I2C modulo SVI2C
cMD03_1Add=176;// Direccion I2C modulo MD03
cSVD01Add =120;// Direccion I2C modulo SVD01
cPORTAdd  =112;// Direccion I2C modulo I/O Digitales
```

Las funciones de esta aplicación para el manejo de los dispositivos I^2C son:

Sonar

void CBrutusDlg::SonarPosition(int iX, int iY)

Envía los comandos I^2C adecuados al módulo svi2c para mover los servos a las posiciones especificadas por *iX* e *iY*. El valor de ambas variables estará entre 0 y 255.

int CBrutusDlg::DistMeasured(void)

Ordena por I^2C al módulo SRF08 hacer una medición de distancia tomando el valor del primer eco y entregando la distancia en cm al objeto por tanto más cercano. Un valor de cero indica que el objeto estaba muy alejado y el eco no regresó.

MD03 (Driver motores tracción)

void CBrutusDlg::SetSpeedMD03_1(unsigned char cSpeed)

Establece la velocidad de los motores según los valores admitidos por el módulo MD03 de Devantech.

void CBrutusDlg::SetDirecMD03_1(unsigned char cDirec)

Establece el sentido de giro de los motores según los valores admitidos por el módulo MD03 de Devantech.

void CBrutusDlg::SetAccelMD03_1(unsigned char cAccel)

Establece la aceleración de los motores según los valores admitidos por el módulo MD03 de Devantech.

unsigned char CBrutusDlg::ReadTempMD03_1()

Lee la temperatura del módulo MD03 de Devantech.

unsigned char CBrutusDlg::ReadCurrMD03_1()

Lee la corriente del módulo MD03 de Devantech (que circula por los motores de tracción).

Servodirección (SVD01)

void CBrutusDlg::SetPosSVD01(unsigned char cPos)

Establece la posición del eje de servodirección y por tanto la inclinación de las ruedas directoras. Admite valores entre 0 y 255.

unsigned char CBrutusDlg::ReadTempSVD01()

Lee la temperatura del módulo SVD01.

unsigned char CBrutusDlg::ReadCurrSVD01()

Lee la corriente del módulo SVD01 que circula por el motor de servodirección).

Modulo de I/O digitales

unsigned char CBrutusDlg::cReadDPort()

Lee los 8 bits del módulo de I/O digitales.

En el caso del control manual, basado en estas funciones y dependiendo de los valores que se establezcan manualmente en los diferentes controles del cuadro de diálogo se controlará adecuadamente el funcionamiento del robot.

El proceso de control autónomo (Autonomous control program) se lleva a cabo mediante la sencilla función que se muestra a continuación, que materializará la máquina de elementos finitos mostrada anteriormente.

```
void CBrutusDlg::OnAuto1()
//Run the autonomous control process
{
   int    iTimeout1, iTimeout2=200, iTimeout3=1000,
          iTimeout4=2000, iTimeout5=200;
   unsigned char cDPort, cDistSonar;

   //State-1. Go ahead until iTimeout1
   for (iTimeout1=0; iTimeout1<=100; iTimeout1++)
   {
      SonarPosition(128, 128);    //Sonar looking to the front
      SetPosSVD01(128);           //Steering centered
      SetAccelMD03_1(128);        //Establish a medium acceleration value
      SetDirecMD03_1(2);          //Wheels rotation: Move robot to front
      SetSpeedMD03_1(50);         //Moderate speed, so could react properly
      cDistSonar=DistMeasured();//Measure distance from sonar
      cDPort=cReadDPort();        //Read Digital port (Check bumpers)
      if (((cDistSonar!=0) && (cDistSonar<50)) ||       //Obstacle < 50cm
         (cDPort!=0) && (cDPort<=3))                    //Or bumpers hit
      {
         //State-2.Stop until iTimeout2
         SetDirecMD03_1(0);       //Wheels stopped
         Sleep (iTimeout2);       //Time to really stop

         //State-3.Go back until iTimeout3
         SetDirecMD03_1(1);       //Wheels rotation: Move robot back
         Sleep (iTimeout3);       //go back during a while

         //State-4. Go back turning until iTimeout4
         SetPosSVD01(0);          //Steering turned to one side
         Sleep (iTimeout4);       //turn during a while

         //State-5.Stop until iTimeout5
         SetDirecMD03_1(0);       //Wheels stopped
         Sleep (iTimeout5);       //Time to really stop
      }
   }
}
```

Conclusiones y líneas futuras

Como se explicó al principio de este informe, Brutus v1.0 solo pretende ser una plataforma de aprendizaje y experimentación. Su estructura modular ha permitido durante su construcción, detectar y subsanar problemas del diseño inicial, pudiendo aislar perfectamente el módulo con funcionamiento inadecuado y facilitando así la identificación de las fuentes de dichos problemas.

Tras numerosas pruebas el robot ha demostrado ser tremendamente robusto, pudiendo bajar por escaleras o moverse por áreas muy abruptas sin sufrir daños.

Obviamente tiene muchas limitaciones, como un único sonar, localizado en la parte frontal o un algoritmo de navegación extremadamente básico, pero a su vez da pie, como se pretende, a ampliaciones, siguiendo la estructura modular detallada en este estudio.

Bibliografía y enlaces de interés

- Mobile Robots: Inspiration to implementation. *Joseph L.Johnes & Anita M.Flyn*. Ed A K Peters, Ltd

- Diseño de un servomotor controlado por bus I^2C mediante microcontrolador PIC de gama media. *Alejandro Alonso Puig*: http://mundobot.com/tecnica/svd01/spsvd01.htm
- Diseño de un módulo para control de 8 servos y 5 canales analógicos por bus I^2C. *Alejandro Alonso Puig*: http://mundobot.com/tecnica/svi2c/spsvi2c.htm
- Control por bus I^2C de drivers de motores MD03 de Devantech desde un microcontrolador PIC de gama media. *Alejandro Alonso Puig*: http://mundobot.com/tecnica/md03/spmd03.htm
- Tarjeta interface KI2C para comunicaciones entre Puerto Paralelo y Bus I^2C: http://mundobot.com/tecnica/ki2c/spki2c.htm
- Diseño y aplicación de un Sonar de barrido. *Alejandro Alonso Puig*: http://mundobot.com/tecnica/sonar/spsonar.htm
- Comunicaciones I^2C. Página de Philips: http://www.semiconductors.philips.com/buses/i2c/