Sistema de intercambio de información basado en memoria compartida distribuida entre un PC y varios microcontroladores utilizado para el control de un robot móvil autónomo.

Oscar González, Julio Pastor, Enrique Moraleja, Diego Alonso, Pedro Revenga, José Manuel Gómez
Departamento de Electrónica - Universidad de Alcalá
Alcalá de Henares – Madrid – Spain
pastor@depeca.uah.es

***Abstract*—Este artículo presenta un método de intercambio de información entre una o varias aplicaciones de alto nivel ejecutadas en un sistema Linux, con varios microcontroladores encargados de realizar tareas de control de bajo nivel de hardware externo, mediante un sistema de intercambio de información basado en memoria compartida distribuida. Este sistema ha sido utilizado en el control de un robot móvil autónomo**

Robot móvil, comunicación, memoria compartida, I²C, sistema de control distribuído.

I. INTRODUCCIÓN

Cuando se plantea el diseño de un robot móvil con suficiente complejidad para necesitar un ordenador empotrado para implementar el control de alto nivel, se plantea el problema de comunicar dicho control de alto nivel con el hardware de bajo nivel (motores, sensores, actuadores, automatismos, etc.) para lo que es necesario tener en cuenta la arquitectura hardware que se utiliza, los protocolos de comunicación empleados, y su implementación.

En el presente artículo se presenta, como solución a este problema, una arquitectura de comunicación basada en memoria compartida distribuida, con la que se consigue una abstracción de todo el proceso de comunicación con el hardware.

En primer lugar se presenta una visión general del sistema, para pasar a comentar en detalle la comunicación entre los procesadores periféricos y el ordenador principal (comunicación de bajo nivel) y los procesos encargados de mantener en el alto nivel la integridad de la memoria compartida (comunicación de alto nivel). Tras explicar la arquitectura interna del sistema se describe la plataforma en la que se han realizado las pruebas.

II. DESCRIPCIÓN GENERAL DEL SISTEMA

En este artículo se presenta la arquitectura hardware y software que posibilita la comunicación entre aplicaciones o procesos de alto nivel ejecutadas sobre un ordenador local o remoto con un sistema hardware formado por varios microcontroladores y periféricos. Las aplicaciones de alto nivel podrán monitorizar o controlar los diferentes elementos conectados.

Esta arquitectura se ha implementado como sistema de control de un robot móvil autónomo donde el hardware está formado por un PC empotrado (VIA – EPIA) donde se ejecutan los procesos de alto nivel, y al cual se conectan varios dispositivos externos mediante el bus I^2C. Estos dispositivos pueden ser microcontroladores o simples elementos hardware compatibles con I^2C.

La arquitectura software implementada para conseguir una abstracción de las comunicaciones entre las aplicaciones de alto nivel y los diferentes elementos hardware, se basa en un sistema que simula el intercambio de información mediante memoria compartida.

Desde un punto de vista general se puede ver el sistema como se indica en la Fig. 1 donde, por un lado se encuentra el hardware a controlar (entradas y salidas digitales y analógicas) y por otro los procesos de control y monitorización de alto nivel. Estos procesos podrían estar situados en el ordenador principal (PC empotrado) y/o en un ordenador remoto conectados mediante un protocolo estándar de comunicación entre ordenadores.

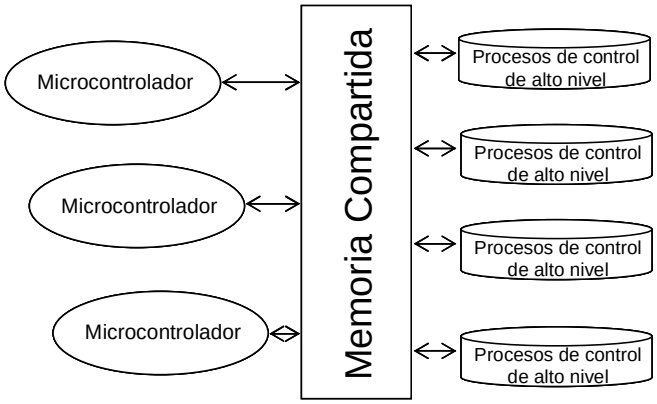

Figura 1: Descripción del sistema de memoria compartida

A. *Arquitectura general de control*

La arquitectura general que se plantea para el control de un robot móvil autónomo está compuesta por procesos de alto nivel y de bajo nivel. Los procesos de alto nivel tendrán funciones de control, monitorización, comunicaciones remotas, etc. y los procesos de bajo nivel serán los encargados del control de motores, de automatización de procesos secuenciales, acciones de control reactivo, etc.

La arquitectura propuesta de comunicaciones entre todos estos procesos está basada en lo que denominamos *Memoria Compartida Distribuida*. Desde el punto de vista de los diferentes procesos implicados en el sistema, la comunicación se establece mediante una única memoria compartida. El sistema propuesto oculta un complejo mecanismo de comunicaciones entre los diferentes elementos, haciendo transparente a los procesos de alto nivel los accesos a sus distintas regiones de memoria, además de mantener en todo momento la integridad de su información.

En cada uno de los dispositivos conectados por I^2C se definen unas zonas de memoria que van a intervenir en la comunicación. Estas zonas, son proyectadas en una región de memoria en el ordenador principal a la cual accederán los procesos de alto nivel.

Para ello el sistema se divide en dos niveles de comunicación: el *nivel bajo* se encarga de proporcionar un conjunto de funciones que permiten, desde una aplicación bajo Linux o Windows, acceder a la memoria de los procesadores periféricos; el *nivel alto* (actualmente desarrollado sobre Linux) se encarga de realizar llamadas a las funciones de *bajo nivel* para actualizar la información de la proyección de las zonas de memoria de los dispositivos externos en la memoria del ordenador principal, a la vez que asegura su integridad y la exclusión mutua en su acceso.

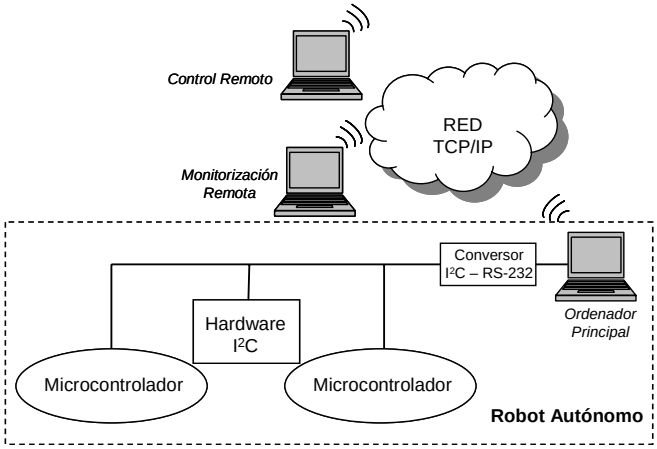

Figura 2: Arquitectura General del Sistema de Comunicaciones

III. SISTEMA DE COMUNICACIONES DE BAJO NIVEL

El control de bajo nivel de un robot móvil hace necesario que exista uno o varios procesadores específicos dedicados a las actividades que requieren una interacción constante con el hardware, como es el caso de los controladores de los motores, sistemas reactivos que se activan al detectar colisiones, control automático de sistemas mecánicos específicos (por ejemplo, en un robot que incorpore un sistema mecánico elevador de materiales es conveniente que el automatismo de bajo nivel que lo controla se ejecute siguiendo las directrices del control de alto nivel pero de forma independiente al mismo para asegurar la respuesta inmediata a finales de carrera, detectores de proximidad, etc.)

El protocolo principal de comunicaciones que se ha decidido utilizar entre el PC y los procesadores periféricos es el bus estándar de comunicaciones I^2C, bus síncrono cada vez más utilizado para el intercambio de información entre microcontroladores o entre un microcontrolador y sus periféricos. Para los ordenadores PC que no incorporan bus I^2C (actualmente hay varios que lo soportan), se utiliza un adaptador RS-232 – I^2C que permite controlar desde un PC cualquier hardware I^2C. (Fig. 3) implementado mediante un microcontrolador dedicado.

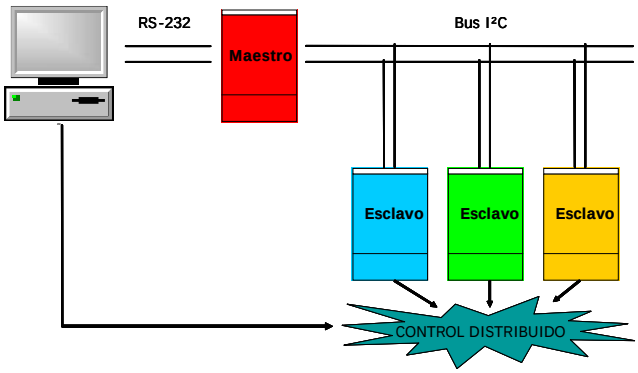

Figura 3: Descripción de los hardware del sistema de control distribuido

La filosofía del intercambio de datos entre el PC y los procesadores periféricos conectados al bus I^2C estriba en que todos los periféricos del bus se comporten como una memoria serie. Los comandos del bus I^2C serán, por tanto, los de acceso a una memoria serie. Esto requiere que en los microcontroladores se implemente una arquitectura software (Fig. 4) donde exista como eje de las comunicaciones una memoria que es compartida por al menos dos procesos: por un lado está el proceso o las tareas de control que leerán los parámetros que sean necesarios de la memoria y escribirán en ella sus resultados, medidas y estado. Al otro lado de la memoria está el proceso o tarea de comunicaciones que se encarga de controlar las comunicaciones, permitiendo la lectura y/o escritura remota de la memoria interna compartida. Este proceso de gestión de las comunicaciones es una máquina de estados situada en la rutina de atención a la interrupción provocada por los subsistemas internos de comunicaciones I^2C configurados como esclavos de I^2C.

Muchos dispositivos preparados para ser controlados por I^2C (sensores, drivers de potencia, etc.) se comportan como si fueran memorias serie (acciones de lectura y escritura), por lo que en el bus pueden situarse, no sólo microcontroladores que

emulan memorias, sino cualquier otro dispositivo que se comporte de la misma manera.

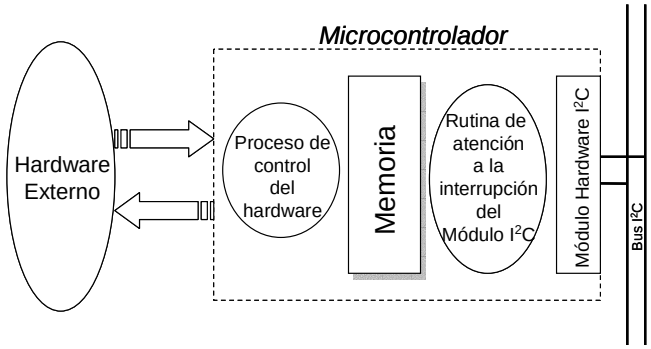

Figura 4: Descripción de la estructura interna de un procesador periférico

Si el PC dispone de subsistemas hardware que le permitan conectarse directamente al bus I^2C, se configurará como maestro del bus y accederá a cada dispositivo con su dirección y los comandos que corresponda como si de memorias serie se tratara.

Si el PC no dispone de esta posibilidad, es necesario introducir un conversor RS-232 – I^2C que simplemente consiste en un microcontrolador que se encarga de traducir los comandos enviados por el PC solicitando lectura y/o escritura en comandos I^2C a la vez que responde con la respuesta.

Desde el punto de vista del PC, se definen unas funciones que permiten enviar un comando y recibir su respuesta.

IV. SISTEMA DE COMUNICACIONES DE ALTO NIVEL

Para la implementación del *Sistema de Memoria Compartida de Comunicaciones Distribuida* de alto nivel se ha empleado un ordenador empotrado modelo VIA EPIA M-6000 gobernado por un sistema operativo GNU/Linux con un Kernel reducido (v.2.4.26) adaptado, con funcionalidades básicas para dar soporte a los distintos módulos de programa que componen toda la aplicación de alto nivel (ver Figura 5) dada a conocer en este artículo.

La parte de la arquitectura software que se expone a continuación consta de diferentes aplicaciones modulares independientes, que pueden compartir información entre ellas y el hardware de bajo nivel por medio de una región de memoria compartida de acceso controlado, siendo su contenido la proyección de los datos presentes en unas determinadas regiones de memoria de cada uno de los microcontroladores conectados al sistema.

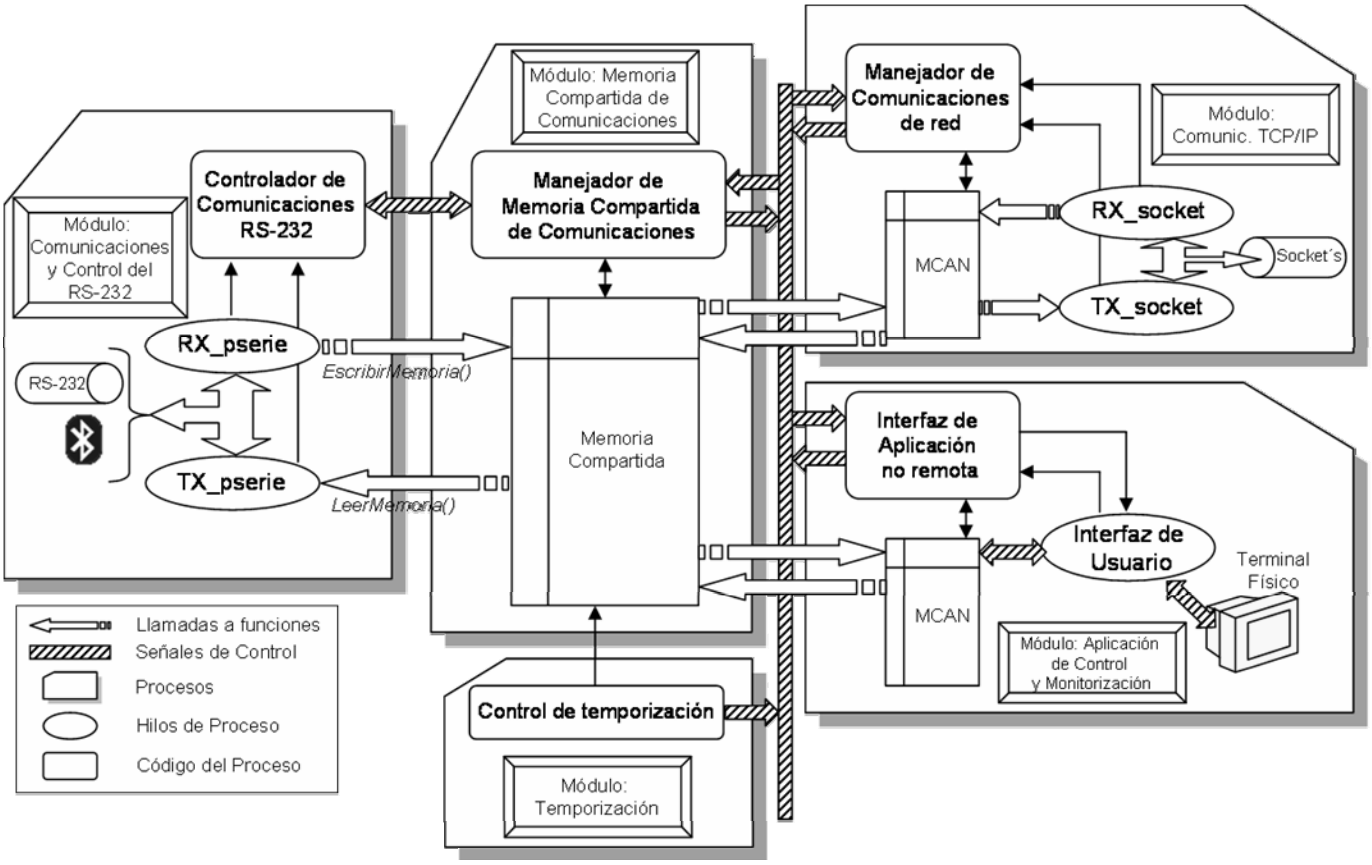

Figura 5: Arquitectura software de alto nivel

A continuación se exponen los diferentes bloques de los que consta este sistema de comunicaciones y control de alto nivel.

A. Memoria Compartida

Este es uno de los elementos principales en cual se basa toda la filosofía de la arquitectura que se está dando a conocer en este artículo. Su funcionalidad principal es abstraer todos los datos y variables de control de los elementos de bajo nivel, haciendo una traslación equivalente de sus contenidos a una región habilitada de memoria compartida del ordenador principal. Uno de los problemas que plantea un sistema dotado de recursos compartidos, es controlar el acceso y ejecutar las diferentes peticiones provenientes de varias fuentes, asegurando continuamente la integridad de los datos, ya sea a la hora de escribirlos o de leerlos. Otro de los problemas consecuencia de la compartición de un recurso por varios procesos, es el denominado interbloqueo o abrazo mortal. Como solución a estos problemas, se ha planteado la creación de un *Manejador de Memoria Compartida de Comunicaciones*, que es un receptor multicola con escala de prioridades, el cual atiende las diversas peticiones de lectura y escritura en la memoria compartida del resto de aplicaciones de alto nivel, evitando, por una parte, que las aplicaciones modifiquen los datos directamente sin control, y por otra, consiguiendo así la abstracción de las memorias de los microcontroladores, facilitando su acceso, haciéndolo transparente de cara al programador, ya sea para programar módulos o aplicaciones, empleando un lenguaje de alto nivel.

Aunque este módulo funciona de forma independiente, existe una cierta simbiosis con el módulo encargado de comunicarse con todos los elementos de bajo nivel, el cual mantiene una comunicación continua, que transfiere todos los datos de las memorias de los diferentes microcontroladores a esta memoria compartida, y viceversa, siendo la velocidad de actualización programable. Esta característica facilita la adaptación a PC´s con distintos tipos de prestaciones, donde el uso del procesador por parte de estas tareas, está limitado por el cómputo necesario para otras, como puede ser el tratamiento digital de imágenes en el caso en el que el sistema disponga de visión artificial.

Todas las aplicaciones de alto nivel que se programen dispondrán de una librería de funciones con las que podrán realizar acciones de escritura y lectura. El manejador de memoria además ofrecerá un conjunto de señales de control y supervisión a estas aplicaciones, por lo que podrán ser informadas de las acciones realizadas, de errores, etc.

Puesto que los accesos a memoria pueden ser realizados por diversas aplicaciones, intentando acceder de forma concurrente. Sería necesario definir una escala de prioridades de peticiones para el acceso entre las diversas aplicaciones, dando mayor prioridad a las que sean más importantes: conexiones de monitorización remotas, control de motores, detección de obstáculos, etc. y menor prioridad a las menos importantes o a las que el tiempo de resolución de peticiones pueda ser mayor.

El número de peticiones que el manejador es capaz de admitir sin ver afectado el rendimiento global del sistema, viene determinado por la velocidad del microprocesador, la velocidad de acceso a la memoria RAM y el grado de computo (accesos a la memoria) del resto de las aplicaciones, como factores más relevantes.

B. Módulo de Conexión con el Sistema de Comunicaciones de Bajo Nivel de Bajo Nivel: RS-232

Este módulo se encarga de recibir y enviar los datos entre la memoria compartida del ordenador principal y las memorias de los distintos microcontroladores, manteniendo así la coherencia de información entre todas ellas. La comunicación entre los diferentes microcontroladores se realiza a través de un microcontrolador maestro que por un lado se encuentra conectado al puerto serie del PC, empleando el correspondiente protocolo de transferencia serie RS-232 y por el otro, está conectado al resto del sistema por I^2C.

El módulo se compone de un proceso principal, denominador *Controlador de Comunicaciones RS-232*, y dos subprocesos que se ejecutan de forma alternada, uno para recoger los datos recibidos por el puerto serie y colocarlos en la memoria compartida (*RX_pserie)*, y el otro para tomar los datos que se vayan a transferir y enviarlos por el puerto serie (*TX_pserie)*. El proceso principal es el encargado de controlar la ejecución ordenada de estos subprocesos y su acceso al recurso compartido como es el puerto serie del ordenador principal, teniendo además la función de comunicarse con el *Manejador de Memoria Compartida de Comunicaciones* para tramitar las peticiones de acceso de escritura o lectura de la memoria compartida. Dicho acceso se realiza por medio de un API de funciones con las que se leen o escriben una determinada región de la memoria compartida.

La idea de dividir en dos subprocesos la comunicación por el puerto serie, uno para escribir en su buffer de salida y otro para leerlo, evitando una posible inanición o bloqueo de la funcionalidad del módulo en caso de que se produzcan errores durante la transmisión o recepción, posibilitando su independencia. Gracias a esto, se han aplicado métodos para temporización, monitorización y aplicación de acciones de recuperación, en caso de que sea posible, y en caso negativo, realización de una parada segura del sistema. Este último punto es importante, puesto que si falla alguno de estos subprocesos se rompe, como consecuencia de la comunicación, la coherencia entre todas las memorias, apareciendo la problemática de que, por un lado, el hardware de bajo nivel quede descontrolado, y por otro, que sea necesario avisar de algún modo a las aplicaciones de alto nivel que usan la memoria compartida, por lo que estas se deberán parar o finalizar de una forma segura.

A la hora de implementar el nivel físico de este módulo existen dos opciones, o bien emplear un cable serie de tres hilos o bien, usar tecnología inalámbrica que soporte

comunicaciones serie, como Bluetooth.

C. Comunicaciones de alto nivel empleando Sockets

Al integrar todo este sistema de control y memoria compartida distribuida en un sistema móvil autónomo, aparece la necesidad de comunicar el sistema con el exterior para disponer de la capacidad de añadir más elementos, de una forma fácil y segura, al sistema distribuido, ya sea para realizar funciones de monitorización del estado de todos los componentes, variables, etc. de forma remota, repartir la carga de trabajo y uso del procesador en otros equipos, etc. aumentando así la eficiencia de todo el sistema móvil.

El módulo encargado de aportar todas estas funcionalidades a la hora de establecer comunicaciones con el exterior es el *Manejador de Comunicaciones de Red*. Las conexiones que se establecen con sistemas remotos utilizan el protocolo TCP/IP (sockets a nivel de aplicación en Linux) teniendo la posibilidad de ser extensibles fácilmente a otros protocolos orientados a conexión.

Cada módulo dispone de una región de *Memoria Compartida de Alto Nivel* individual (MCAN), y dos subprocesos, uno encargado de transmitir los datos, (*TX_socket)*, y otro de recibirlos, (*RX_socket)*. Estos subprocesos hacen uso de la MCAN de la cual tomarán los datos a enviar o volcarán los recibidos. Cuando sea necesario acceder a la Memoria Compartida de Comunicaciones, será el manejador de Comunicaciones, el encargado de tramitar las peticiones al Manejador de Memoria Compartida, para realizar la transferencia de datos entre ambas memorias.

Este módulo aporta unas librerías de funciones que permiten la monitorización, configuración, control y aviso de pérdidas de conexión, situación especial y en ocasiones peligrosa, ante la cual se debe disponer de un protocolo de actuación, ya que puede verse afectada la integridad del sistema.

Las funcionalidades que ofrece este proceso pueden ser utilizadas por el resto de las aplicaciones internas del ordenador principal, que necesiten conectarse a otros sistemas remotos, o en el caso contrario, donde existan aplicaciones remotas que se conecten a él. Un ejemplo de este tipo de aplicación remota es la creada como aplicación para telemetría y control en tiempo real, además de para dirigir el sistema robótico móvil donde se ha implantado esta arquitectura, "Consola", de la cual se habla más adelante.

V. PLATAFORMA DE PRUEBAS

La plataforma de pruebas en la que se ha aplicado de forma práctica toda la arquitectura comentada, ha sido en un robot móvil autónomo creado para participar en la Competición Europea de Robots, Eurobot 2004 [5], en la prueba denominada "Rugby del Coco". Dicho juego requería de un robot autónomo dotado con diversos subsistemas sensoriales, de actuación y motrices, siendo controlados a bajo nivel por una red de microcontroladores conectados a un PC empotrado, que realizaba el control a alto nivel del sistema completo.

La estructura que se ha empleado en el robot ha sido de tres microcontroladores, uno encargado del control de los motores para el movimiento del mismo e implementación del sistema de sensores de contacto, otro encargado del manejo de la electrónica relativa al dispositivo de disparo (cañón para lanzar pelotas), y un tercero encargado de la recopiación de datos relativos al posicionamiento y orientación tomados del sistema de triangulación por balizas infrarrojas. A su vez, estos microcontroladores son controlados por un PC empotrado, conectados a través de un microcontrolador maestro, a su puerto serie RS-232. El hecho de emplear este tipo de arquitectura permite crear aplicaciones que engloban conjuntos de simples instrucciones de bajo nivel en otras instrucciones de alto nivel más completas y complejas, consiguiendo así un mejor control, pudiendo llegar a ser un robot capaz de realizar funciones y acciones más complicadas.

Por último, se ha creado una aplicación gráfica multiplataforma (programada haciendo uso de las librerias Qt) para acceder remotamente al robot, ya sea para realizar funciones de monitorización, telecontrol y un pequeño programador de conductas o acciones que el robot es capaz de generar de forma autónoma.

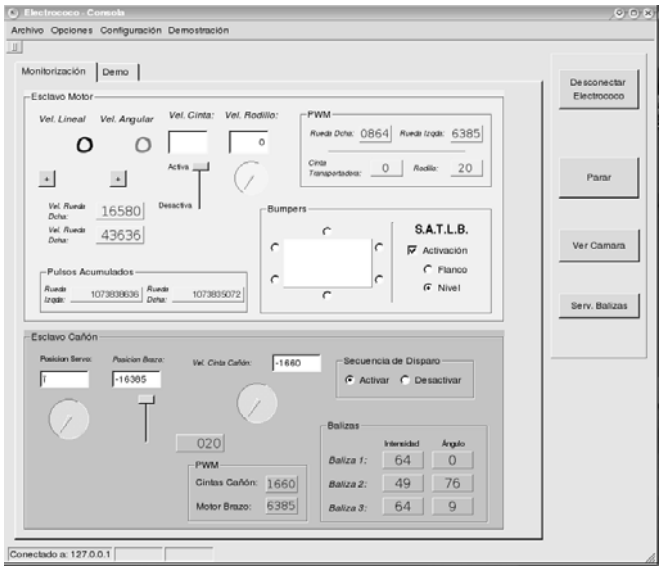

Figura 6: Interfaz Gráfico de la aplicación de control remoto "Consola"

AGRADECIMIENTOS

El presente trabajo se ha realizado con el apoyo económico del Departamento de Electrónica de la Universidad de Alcalá para el desarrollo de un robot que participara en la competición de internacional robots Eurobot 2004.

También se agradece la participación en el diseño del robot del resto de componentes del equipo: Javier Baliñas, Diego Alonso, Saleta Nistal, Till Zenthöfer, Noelia Requena, Santiago Castillo, Raúl Sánchez, David Mollejo, Jorge Fernández, Mario

Inglés, Jordi Polo, Javier Rodríguez, César Guerra, César Lozano, Jorge Pérez y Javier Antunez.

REFERENCIAS

[1] Dominique Paret and Carl Fenger, "The I^2C Bus From Theory to Practice", John Wiley & Sons Ltd., 1997

[2] Hennessy Patterson. "Computer Arquitecture". 2002. ISBN 1.55860.596.7

[3] Karim Yaghmour, "Building Embedded Linux Systems". 2003. O´Reilly.

[4] Bill O. Gallmeister, "POSIX 4: Programming For Real World". 1995, O´Reilly.

[5] Competición Internacional de Robots EUROBOT: http://www.eurobot.org