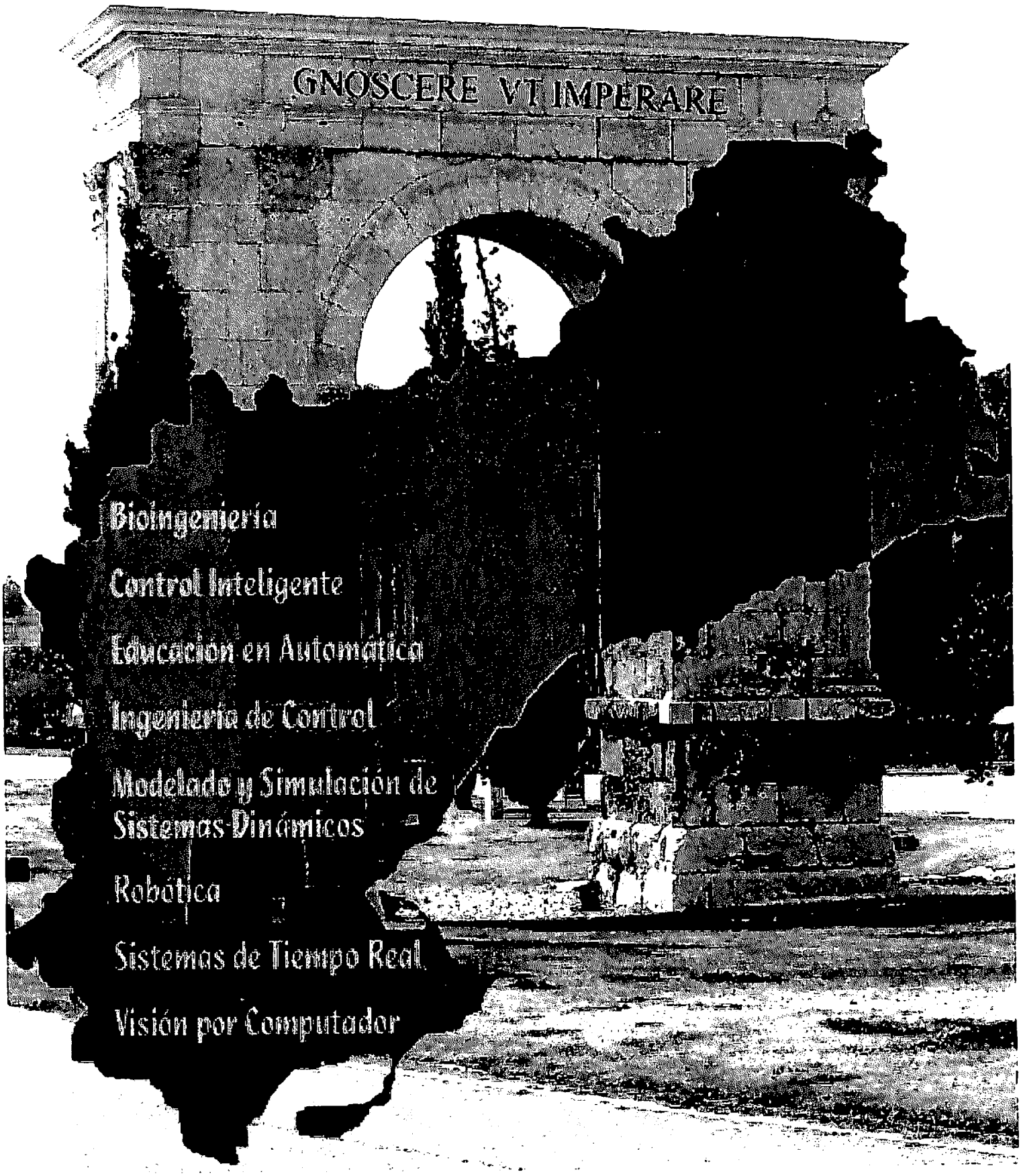
GNOSCERE VT IMPERARE
Bioingeniería
Control Inteligente
Educación en Automática
Ingeniería de Control
Modelado y Simulación de Sistemas Dinámicos
Robótica
Sistemas de Tiempo Real
Visión por Computador

IMPLEMENTACIÓN DE UN LABORATORIO REMOTO PARA LA REALIZACIÓN DE PRÁCTICAS DE ROBÓTICA MÓVIL

A. Sánchez, O. Reinoso, L. Payá, D. Ubeda, M. Juliá, A. Gil
Departamento de Ingeniería de Sistemas Industriales
Universidad Miguel Hernández. Avda. de la Universidad s/n
Ed. Torreblanca. 03202 Elche (Alicante) España
alex@emotica.net, o.reinoso@umh.es, lpaya@umh.es

Resumen

En este trabajo se presenta una plataforma distribuida que permite la realización de prácticas con robots móviles a través de Internet. El objetivo final consiste en el desarrollo de una arquitectura que permita al estudiante acceder de forma transparente a un conjunto de robots, independientemente del sistema operativo que utilice y con la mínima instalación por su parte. En el artículo se detallan todos los procedimientos utilizados para la implementación del sistema. Asimismo, se presenta el modelo de prácticas propuestas que permiten a los alumnos profundizar en algunos conceptos de robótica móvil y visión artificial, con las ventajas que supone el acceso a través de Internet. Por último, se plantean las medidas de seguridad adoptadas para garantizar un uso adecuado del sistema.

Palabras Clave: Robótica móvil, laboratorio remoto, arquitectura CORBA, Control de robots

1. INTRODUCCION

Hoy en día, Internet se ha convertido en una potente herramienta en el campo de la educación. Los estudiantes se encuentran altamente motivados para usar los recursos disponibles en entornos remotos a través de Internet. Asimismo, gracias a los laboratorios remotos, el estudiante se puede habituar al trabajo con equipos reales, a los cuales puede acceder desde su hogar y con una amplia flexibilidad de horarios.

El concepto de laboratorio remoto adquiere especial relevancia ante el cambio de escenario que está sufriendo la educación en nuestro país, con la adaptación de las enseñanzas al Espacio Europeo de Educación Superior. El nuevo sistema se centra en el aprendizaje del alumno, potenciando su autonomía y disminuyendo el tiempo dedicado a la exposición de conocimientos conceptuales por parte del profesor, en un sistema menos rígido en cuanto a clases y horarios. Los laboratorios remotos van a ayudar decididamente a la implantación de este nuevo sistema en las carreras técnicas.

Las tecnologías de la información y las comunicaciones se han venido aplicando en los últimos tiempos de forma exitosa a la docencia en control [1], [2]. Una de las posibles aplicaciones son los laboratorios remotos, entornos distribuidos que permiten a los alumnos acceder remotamente a través de Internet a los equipos disponibles en el laboratorio. En este trabajo se presenta un laboratorio remoto que ha sido desarrollado con el objetivo de que los alumnos puedan hacer uso de los robots móviles disponibles en el laboratorio a través de Internet. Dentro del área de robótica, podemos encontrar varios trabajos que presentan mecanismos para manejar robots en un entorno remoto. Candelas et al. presentan en [3] un laboratorio virtual para realización de prácticas en robótica que permite tanto la simulación de un brazo robot como la teleoperación del robot real equivalente. Thamma et al. describen en [4] una arquitectura cliente/servidor para el control remoto de un robot manipulador, haciendo uso de programación en Java utilizando TCP y sockets y accediendo al servidor usando cualquier navegador web. El trabajo [5] presenta un laboratorio remoto que permite el control de un robot móvil Lego a través de un navegador Web. El alumno puede diseñar diversos reguladores usando un entorno MATLAB/Simulink orientados a la planificación de trayectorias y testarlos sobre un robot real. Khamis et al. han desarrollado la arquitectura 'Developer' para la teleoperación y control del robot B21r [6] y por último, Siegwart y Saucy presentan y discuten varias aplicaciones que manejan robots móviles a través de Internet en EPFL, en Lausanne [7].

En este trabajo se presenta una plataforma distribuida que permite la monitorización y control de un equipo de robots móviles a través de Internet. Para ello, se ha diseñado un protocolo de comunicación entre los diferentes elementos del sistema usando el modelo de referencia CORBA (Control Object Request Broker

Architecture), cuya robustez y eficiencia para la implementación de aplicaciones distribuidas ha sido mostrada en trabajos previos [8], [9], [10]. Se pretende desarrollar una plataforma que permita a los usuarios hacer uso de los robots de forma transparente, sin necesidad de conocer con detalle la arquitectura del robot y que, en última instancia, ofrezca acceso a diversos tipos de robots utilizando la misma interfaz.

El resto del trabajo se estructura del siguiente modo. A continuación se presenta la arquitectura física de la plataforma desarrollada. En la sección 3 se muestran los flujos de comunicación existente entre los elementos de la plataforma. Por último se comentan las prácticas disponibles en el entorno y los mecanismos de seguridad implementados, para finalizar con las conclusiones y trabajos futuros.

2. ARQUITECTURA FISICA

El objetivo final del trabajo consiste en el diseño e implementación de una plataforma mediante la cual los alumnos puedan probar sus algoritmos de control mediante una interfaz sencilla, programando a alto nivel y sin necesidad de conocer en profundidad la arquitectura del robot utilizado. Con este objetivo, se han creado 3 entornos de trabajo que permitirán acceder simultáneamente hasta 3 usuarios para realizar las prácticas.

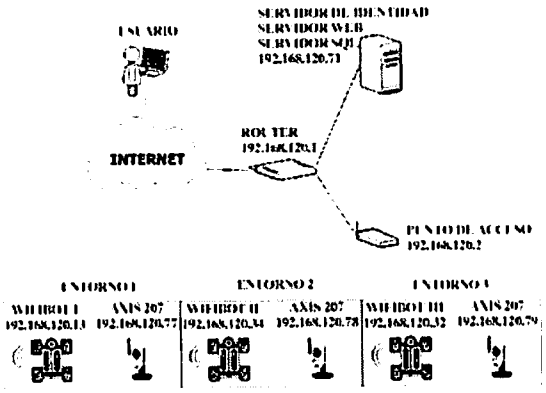

Figura 1: Entornos de trabajo accesibles remotamente por los estudiantes

En cada entorno de trabajo tenemos un robot WifiBot que se conecta vía wifi al punto de acceso y una cámara Axis 207W (también wifi) colocada en el techo, que nos dará una visión del entorno y nos permitirá calcular la posición del robot por medio de una marca circular situada encima del mismo. Cada robot WifiBot cuenta con una cámara color y dos sensores infrarrojos de distancia, sistema de odometría y cuatro motores independientes. Además, han sido equipados con un PC a bordo.

Asimismo, existe un PC conectado a la red en el laboratorio en el cual se ejecutan los siguientes servicios, que se comentarán con mayor grado de detalle en los apartados siguientes.

- Servidor Web, mediante el cual los alumnos pueden descargar la aplicación cliente.
- Servidor de identidad, base de las comunicaciones en CORBA.
- Servidor SQL, usado para implementar los mecanismos de seguridad necesarios, consistentes en la restricción de acceso y registro de las acciones llevadas a cabo por los usuarios.

Cabe destacar que, con el objetivo de que los estudiantes puedan acceder a la plataforma de prácticas desde cualquier ubicación en Internet y no solo desde los laboratorios de prácticas, es necesario disponer de un acceso total al router para poder abrir y redirigir por los puertos correspondientes, teniendo en cuenta que, además de los puertos necesarios para los protocolos HTTP, SSH y MySQL, necesitamos que los paquetes CORBA no sean filtrados por el firewall [11].

3. COMUNICACIONES

Para definir el flujo de comunicaciones que se produce entre los diferentes elementos del sistema, es necesario diferenciar entre 2 tipos de comunicaciones.

3.1 Comunicación CORBA

La comunicación básica entre los diferentes elementos del sistema esta basada en la arquitectura CORBA [12]. Los elementos principales de la red de comunicaciones se detallan a continuación:

Servidores a bordo de los robots: A bordo de cada robot, se ejecutan los servidores que acceden directamente a los sensores y actuadores de los robots (cámaras, sensores infrarrojos, motores, etc...). Estos servidores proporcionan una serie de servicios, que se implementan mediante interfaces CORBA. En concreto, cada interfaz está implementada mediante un objeto C++, que accede a los sensores del robot o comanda sus movimientos. El estándar CORBA permite acceder a cada servicio a través de una cadena de texto llamada IOR (Inter-operable Object Reference). En la actualidad se encuentran implementadas las siguientes interfaces:

- **Interfaz RobotBasics:** implementa los comandos de movimiento básicos (ponerse en marcha, establecer velocidad lineal/angular, detener el movimiento), que pueden ser realizados por todos los robots. También incluye la lectura de los sensores infrarrojos y de la odometría del robot.

- **Interfaz SingleCamera:** permite a un cliente obtener imágenes de la cámara integrada en cada robot. También permite obtener imágenes de la cámara del entorno y nos proporciona información sobre la posición del robot en el entorno mediante la marca circular que el robot lleva encima.

- **Interfaz StereoCamera:** se ejecuta en aquellos robots que están dotados de una cámara estéreo.

- **Interfaz LaserServer:** ofrece datos de un sensor de distancia láser (disponible únicamente para robots que dispongan de este tipo de sensor).

Servidor de identidad y ejecución: se ejecuta en un PC conectado a la red. La dirección IP de dicho PC es conocida por todos los elementos del sistema y es el único dato que es necesario configurar para establecer las comunicaciones en el sistema. Este servidor tiene dos funciones principales. La primera consiste en mantener un listado de todos los robots y cámaras del entorno que se encuentran activos en el sistema, y qué interfaces ofrece cada uno de ellos. En concreto, para cada robot, el servidor de identidad y ejecución almacena una cadena de texto que identifica al robot (Wifi1, Wifi2,...) junto con una lista de los servicios que proporciona (cámara, sensores infrarrojos, sonar, etc....) almacenando también una cadena de texto y una cadena IOR para cada uno de ellos. La segunda función principal es ofrecer un control de acceso de las aplicaciones cliente a los robots. Esto significa que cuando un usuario accede a un entorno de prácticas que se encuentra libre para comandar un robot, obtiene su control en exclusiva y el resto de usuarios no puede acceder al entorno hasta que el usuario lo abandone. Además, para evitar que un único usuario pueda acaparar el entorno indefinidamente, se ha establecido un sistema de reservas mediante el cual cada alumno puede reservar una franja horaria determinada (con un tiempo máximo limitado a una hora en la implementación actual). Dentro de ese intervalo de tiempo, únicamente el estudiante que ha formalizado la reserva tiene acceso al entorno de trabajo. Dado que existen tres entornos de trabajo, pueden estar tres estudiantes utilizando el sistema simultáneamente.

Aplicación cliente: se puede ejecutar en cualquier maquina independientemente de su arquitectura y/o sistema operativo. Asimismo la aplicación permite el acceso a los entornos de los robots para realizar las prácticas independientemente de la ubicación desde la que se ejecute, ya sea en los laboratorios de prácticas o desde cualquier ubicación en Internet.

La aplicación cliente se ha desarrollado íntegramente utilizando el lenguaje Java, puesto que nos permite la independencia que buscamos tanto en arquitectura como de sistema operativo. Además Java soporta nativamente CORBA sin necesidad de utilizar librerías suministradas por terceros. El soporte de CORBA en Java esta definido por OMG (Object Management Group), por lo tanto es capaz de interactuar con los servicios CORBA desarrollados en C++ que ofrecen tantos los robots, las cámaras del entorno y el servidor de identidad.

La implementación de la aplicación cliente ha sido llevada a cabo mediante Java Web Start, que es la implementación de referencia de la especificación JNLP (Java Networking Launching Protocol) y está desarrollada por Sun Microsystems. Mediante esta tecnología es posible arrancar aplicaciones Java de escritorio que están en un servidor web de aplicaciones, comprobando previamente si el cliente tiene la versión actualizada de dicha aplicación. Si no es así, se descargará la última versión y se ejecutará en modo local. El arranque de dichas aplicaciones puede ser efectuado mediante enlaces en una página web o bien a través de enlaces en el escritorio cliente. Mediante está tecnología se asegura que una aplicación es distribuida siempre en su última versión.

La información sobre dónde se encuentra la aplicación, versión, etc... está contenida en un fichero de configuración con extensión ".jnlp", de modo que únicamente es necesario colgar el archivo ".jnlp", que apenas ocupa unos pocos kilobytes, en una web desde la que los alumnos puedan descargarlo y posteriormente ejecutarlo. Por su parte, cualquier persona que quiera hacer uso de la aplicación también deberá tener instalada en su maquina la ultima distribución de JRE (Java Runtime Environment), teniendo en cuenta que, actualmente, Java Web Start viene incluido en el JRE.

Si bien también se barajó inicialmente la opción de integrar la aplicación en un Applet en Java, de modo que los usuarios sólo tuvieran que acceder a una página web determinada para utilizar la aplicación, finalmente, las restricciones de seguridad propias de los Applets presentaron dificultades tanto en el momento de trabajar con archivos (cargar/grabar) en la maquina local del usuario como al crear

comunicaciones bidireccionales en CORBA con los servicios de los robots, y se tuvo que optar por el uso de Java Web Start.

Flujo de comunicaciones: el primer elemento que debe estar activo en el sistema es el servidor de identidad y ejecución. A continuación se deben activar los servidores a bordo de los robots. Cuando un servidor se activa, publica el nombre del robot y las interfaces que implementa (junto con sus IOR correspondientes) en el servidor de identidad y ejecución. Los diferentes robots en el sistema se guardan en entradas distintas en el servidor de identidad y ejecución.

Cuando una aplicación cliente desea acceder a los servicios de un robot, entonces realiza una petición al servidor de identidad y ejecución, indicando el nombre simbólico del robot y una cadena con las interfaces a las que debe acceder. Si el robot se encuentra disponible, el servidor de ejecución le devuelve las referencias a las interfaces que implementa el robot. A partir de ese momento, la comunicación se realiza bidireccionalmente entre la aplicación cliente y el servidor a bordo del robot. Típicamente, la aplicación cliente envía peticiones de captura de datos de los sensores y comandos de movimiento de los motores. Todo este proceso se resume en la figura 2.

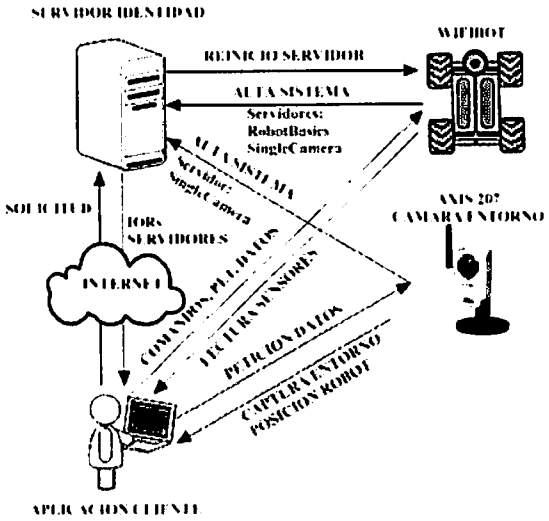

Figura 2: Flujo de comunicaciones CORBA en el laboratorio remoto

La figura 3 muestra el aspecto de la aplicación cliente. Desde la ventana principal de dicha aplicación, el estudiante puede visualizar las imágenes capturadas por la cámara del techo, y las velocidades de avance y de giro del robot, comandar el robot de forma básica y decidir qué datos desea guardar durante la ejecución del programa y en qué ficheros.

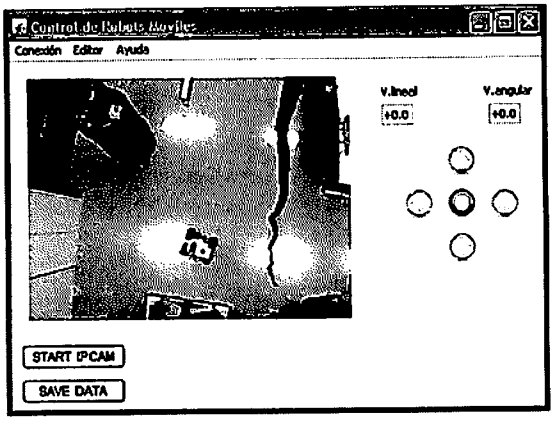

Figura 3: Interfaz gráfica de la aplicación cliente

3.2 Comunicación SSH

Este nivel de comunicación es empleado para la implementación de algoritmos en los Wifibots que nos permitan realizar las prácticas. El objetivo es que el alumno pueda enviar el programa que ha creado al robot correspondiente, compilarlo y ejecutarlo. Para su implementación se ha utilizado JSch, librería escrita completamente en JAVA desarrollada por la empresa JCraft bajo licencia GPL que implementa el protocolo SSH2. Gracias a esta librería podemos integrar las funcionalidades del protocolo SSH2 en nuestros programas JAVA.

SSH ofrece soporte para acceso remoto seguro, transferencia de archivos segura, redireccionamiento seguro de TCP/IP y X11. Puede automáticamente encriptar, autentificar y comprimir los datos transmitidos.

En este trabajo, JSch es utilizado para establecer conexiones remotas con un daemon SSH que se encuentra instalado en los robots con el objetivo de conexiones entrantes. Más concretamente, la plataforma hace uso de las siguientes funciones de JSch:

- Scpto: para enviar el archivo con el código fuente desde la maquina del usuario a la maquina donde se compilará y ejecutará.
- Exec: permite ejecutar comandos de forma remota. En el presente trabajo, es utilizado para compilar el código fuente en C mediante un "makefile" en la maquina remota y posteriormente ejecutarlo. Esta

función también nos muestra por pantalla tanto la salida estándar (stdout) como la salida estándar de error (stderr) de la maquina remota. Si la "stderr" no devuelve nada es que la compilación ha sido correcta, en caso contrario tendríamos que modificar el código fuente y volverlo a enviar.

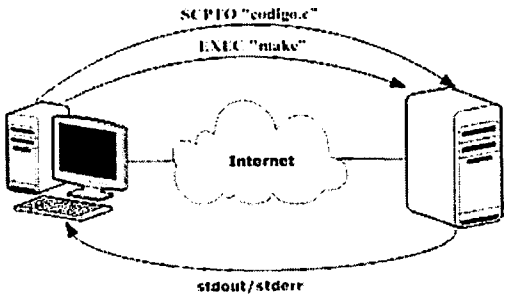

Figura 4: Flujo de comunicaciones SSH en el laboratorio remoto

Con esta forma de realizar las practicas tenemos la ventaja que el usuario no debe tener instalada ninguna librería en su maquina y la desventaja que depurar los errores de compilación en el código fuente se convierte en una tarea mas lenta, por tener que reenviar el código fuente cada vez. El proceso de comunicación se muestra en la figura 4. En la figura 5 se muestra el aspecto de la ventana de edición de código, accesible desde la ventana principal de la aplicación cliente. Desde la ventana de edición, el estudiante puede editar el código fuente de su programa de control, enviarlo al servidor, compilarlo y comprobar si ha existido algún error.

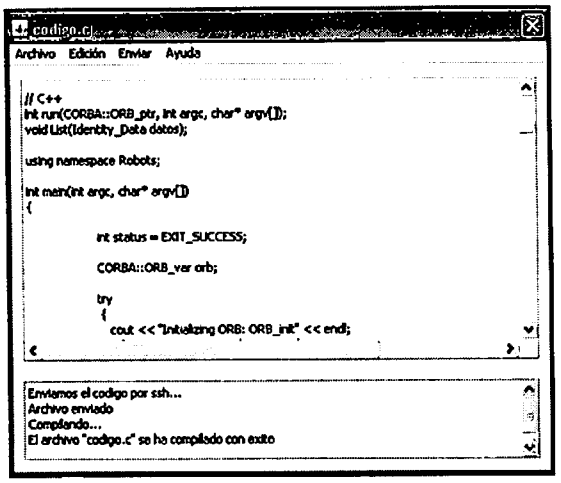

Figura 5: Ventana de edición del programa

4. PRACTICAS IMPLEMENTADAS

Hasta el momento, se ha implementado una práctica que consta de dos ejercicios mediante los cuales el alumno puede poner en práctica sus conocimientos en cuanto a los sensores y actuadores del robot, cinemática y control reactivo del mismo mediante coordinación competitiva de comportamientos. A continuación se describen con detalle los objetivos de dichos ejercicios.

4.1 Ejercicio 1.

El alumno debe escribir un programa que permita mover un robot en línea recta, evitando los obstáculos que encuentre en su camino, hasta que el robot haya recorrido una distancia determinada. Para ello, se debe hacer uso de los sensores de infrarrojos para detectar la presencia de un obstáculo y se debe leer la odometría del robot para calcular la posición del mismo en cada momento y la distancia total recorrida. El alumno también debe usar la información de la cámara fija para conocer en cada momento la posición del robot y compararla con los resultados suministrados por la odometría.

En general, el robot debe moverse en línea recta, describiendo un arco de 90° cuando encuentra un obstáculo hacia el lado más distante del obstáculo. Asimismo, el robot debe detenerse cuando haya alcanzado una determinada distancia recorrida. Para ello, el alumno debe hacer uso de una función que integra los datos de la odometría y que devuelve las coordenadas del robot respecto a la posición inicial, contenida en la interfaz RobotBasics.

Tras la simulación, el alumno debe representar gráficamente la trayectoria seguida por el robot, a partir de los datos de la odometría, y esta misma trayectoria pero obtenida a partir de la información aportada por la cámara externa, realizando una comparación entre ambas y un análisis crítico de los resultados.

4.2 Ejercicio 2.

El alumno debe modificar el programa anterior de forma que el robot se mueva desde la posición inicial hasta un punto destino predeterminado evitando los obstáculos que aparezcan a su paso.

En cada momento, el robot debe moverse en línea recta hacia el punto destino girando cuando encuentra un obstáculo. Una vez superado, el robot debe volver a orientarse hacia el punto destino y tender hacia él en línea recta. Para conocer la posición actual del robot se debe hacer uso de la función que calcula la posición del robot usando la cámara del techo.

Para resolver el problema, se deben implementar los comportamientos **Ir a destino** y **Evitar obstáculo**. En cada instante, únicamente uno de los dos comportamientos debe estar activo, en función de la lectura de los infrarrojos. Una vez ejecutado el programa, el alumno debe representar gráficamente la trayectoria seguida por el robot, obtenida a partir de los datos aportados por la cámara situada en el techo.

4.3 Mecanismos de seguridad

Como mecanismo de seguridad a la hora de realizar las prácticas se ha creado una base da datos SQL (MySQL) corriendo en la misma maquina Linux donde esta instalado el servidor Web y el servidor de identidad, que posee una doble funcionalidad. En primer lugar, limita el uso de la aplicación solo a usuarios registrados previamente en la base de datos mediante un nombre de usuario y una contraseña. Esto se realizará justo antes de consultar los robots disponibles para realizar las prácticas. Asimismo, también se encarga de registrar en todo momento las acciones de los usuarios guardándolas en una tabla de la base de datos, desde que un usuario se da de alta en un sistema y escoge un entorno para la practica hasta que acaba la practica, se da de baja en el sistema y libera el entorno.

Para el desarrollo del sistema necesitamos tener instalado un servidor MySQL en cualquier maquina de la intranet que consta de dos tablas. La primera de ellas, la correspondiente a los usuarios, contiene tres campos, donde se almacena el DNI del usuario, la contraseña encriptada en formato SHA1 y el nombre del mismo. La segunda tabla está destinada a almacenar el historial de operaciones realizadas por un usuario durante el periodo en que ha estado ocupando el sistema. Básicamente almacena la fecha, hora, IP y acción que realiza cada usuario en cada momento. Dado que es la aplicación cliente la que se encarga de realizar las inserciones en las tablas, para la implementación de este mecanismo de seguridad ha sido necesario usar el conector de MySQL para Java "MySQL Connector/J".

Además de este mecanismo de seguridad en los robots se ha implementado otro preventivo a modo de hilo de ejecución que se ejecuta continuamente junto con el programa de control del usuario, y que está encargado de monitorizar en todo momento las velocidades lineales y angulares del robot además de los sensores de infrarrojos, de modo que puede ordenar que se detenga el robot en el caso que detecte un objeto frontal a una distancia corta del robot o detectar y limitar velocidades elevadas del mismo.

5. CONCLUSIONES

Como conclusión, cabe destacar que se ha implementado una plataforma de trabajo para la realización de prácticas, a la que los usuarios pueden acceder con independencia de horario y ubicación. En esta plataforma los usuarios podrán probar los algoritmos que diseñan sobre robots reales y pueden acceder a todos los servicios que ofrecen los robots, permitiendo profundizar en el aprendizaje del funcionamiento de los sensores básicos en robótica móvil y las estrategias de control de un robot.

Gracias a los servidores implementados, los usuarios pueden hacer uso de los robots de forma transparente. A pesar de que todos los servicios se han expuesto de forma aplicada a un modelo de robot concreto (WifiBot), las interfaces ofrecidas por los robots son iguales para cualquier robot en las que se hayan implementado. De esto modo, se podrían añadir nuevos entornos o sustituir algún robot de algún entorno sin tener que hacer ninguna modificación de ningún tipo en el servidor de identidad, ni en la aplicación cliente. Por tanto, la plataforma permite el control remoto de un equipo heterogéneo de robots de forma transparente.

En lo que respecta a la aplicación cliente gracias a la utilización de Java Web Start se han conseguido las siguientes ventajas:

1. Independencia de la arquitectura y/o sistema operativo.
2. No es necesario que usuario instale nada en su maquina, ni necesite librerías adicionales, ni tenga que compilar la aplicación.
3. El usuario siempre dispone de la última versión de la aplicación cliente.

Agradecimientos

Este trabajo ha sido subvencionado por el Ministerio de Educación y Ciencia a través del proyecto DPI2007-61197 'Sistemas de percepción visual móvil y cooperativo como soporte para la realización de tareas con redes de robots'.

Referencias

[1] Sánchez, J., (2001) Un Nuevo enfoque metodológico para la enseñanza a distancia de asignaturas experimentales: análisis, diseño y desarrollo de un laboratorio virtual y remoto para el studio de la automática a través de Internet, Tesis Doctoral, UNED.

[2] Dormido, S., (2004) Control learning: present and future, Annual Reviews in Control, 28(1), pp. 115-136.

[3] Candelas, F. A., Puente, S. R., Torres, F., Ortiz, F. Gil, P., Pomares, J. (2003). A virtual laboratory for teaching robotics. *International Journal of Engineering Education*, vol. 19, No 3. pp. 363-370

[4] Thamma, R., Huang, L. H., Lou, S. J., Diez, C. R. (2004). Controlling robot through Internet using Java. *Journal of Industrial Technology*, vol. 20, No 3

[5] Carusi, F., Casini, M., Prattichizzo, D., Vicino A., (2004) Remote control of a Lego mobile robot through the web. Workshop of Internet Based Control Education, Grenoble, Francia

[6] Khamis, A., Rivero, D. M., Rodríguez, F., Salichs, M. (2003). Pattern-based architecture for building mobile robotics remote laboratories. *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 3, pp 3284-3289

[7] Siegwart, R. and Saucy, P. (1999). Interacting mobile robots on the Web. *Workshop Proceedings of the IEEE International Conference on Robotics and Automation*

[8] Payá, L., Gil, A., Reinoso, O., Riera, L., Jiménez, L.M. (2006). Distributed platform for the control of the WiFiBot robot through Internet. *ACE 2006. 7th IFAC Symposium on Advances in Control Education*

[9] Payá, L., Juliá, M., Reinoso, O., Gil, A., Jiménez, L.M. (2006). Behaviour-based multi-robot formations using computer vision. Aceptado para publicación en *6th IASTED International Conference on Visualization, Imaging and Image Processing*

[10] Utz, H., Stulp, F., Moeld, A. (2004). Sharing belief in teams of heterogeneous robot. *Proc. of RoboCup-2004 symposium*, pp. 508-515

[11] Theophilo, A., Endler, M., Cerqueira, R., (2005) Evaluation of Three Approaches for CORBA Firewall/NAT Traversal. International Symposium in Distributed Objects and Applications, Chipre

[12] OMG, Object Management Group, Common Object Request Broker: Architecture and Specification. Revision 2.0, 1995.