Arquitectura Abierta de un Robot Industrial Stäubli RX60

J.Gámez García*, J. Gómez Ortega*, M. Vargas**, F.R. Rubio**

*Grupo de Robótica, Visión por Computador y Automática
Escuela Politécnica Superior de la Universidad de Jaén
juango@ujaen.es

**Ingeniería de Sistemas y Automática
Escuela Superior de Ingenieros de la Universidad de Sevilla

frrubio@esi.us.es

RESUMEN

En este trabajo se presentan los elementos esenciales de una arquitectura abierta para el control de cada una de las articulaciones de un robot industrial Stäubli RX60. Esta plataforma de experimentación se utilizará para el desarrollo de algoritmos de control para robots manipuladores con movimientos restringidos, utilizando para ello la información generada por un sensor de fuerza y par.

1. INTRODUCCIÓN

En el campo de la robótica de manipuladores el problema del control de los movimientos del brazo articulado es, probablemente, uno de los aspectos que más trabajos y bibliografía ha generado desde los inicios de la investigación en esta disciplina.

En este sentido, uno de los principales inconvenientes con el que se suelen encontrar la mayoría de los grupos de investigación es la imposibilidad de implementar nuevos algoritmos de control sobre las arquitecturas cerradas que presentan la práctica totalidad de los robots industriales del mercado.

La mayoría de los robots utilizados en tareas de investigación pueden encuadrarse en una de las tres siguientes categorías:

- Aquellos que han sido construidos para satisfacer globalmente la demanda de mercado. Estos suelen ser los más económicos pero también los más limitados en cuanto a flexibilidad de desarrollo se refiere.

- Aquellos en los que la parte del sistema de cálculo ha sido sustituida por computadores externos que permiten utilizar software de control diferente al original [5].

- Robots específicamente diseñados y construidos, a veces incluyendo soluciones mecánicas especiales [3], pero más a menudo como sistemas mecánicamente sencillos como es el caso de los robots de accionamiento directo.

En este trabajo se presentan los elementos esenciales de una arquitectura abierta para el control de un robot industrial Stäubli RX60. Esta plataforma de experimentación se utilizará para el desarrollo de algoritmos de control para robots manipuladores con movimientos restringidos, utilizando para ello la información generada por un sensor de fuerza y par.

El robot aquí descrito se considera dentro de aquellos que son utilizados normalmente en la industria, pero que permite acceso total a las funciones de control y programación de las articulaciones, aún manteniendo la funciones originales y de seguridad propias. En la siguiente sección se muestra una visión general del sistema, detallando aquellos aspectos significativos de la plataforma de experimentación.

Este trabajo ha sido desarrollado en el contexto del proyecto coordinado CONPOS, que está siendo desarrollado entre el grupo de Robótica, Visión por Computador y Automática de la Universidad de Jaén, y el Grupo de Robótica y Automática de la Universidad de Sevilla.

2. DESCRIPCIÓN DEL SISTEMA

El objetivo principal de diseño ha sido el desarrollo de una plataforma de experimentación para la implementación de algoritmos de control para robots manipuladores con realimentación del esfuerzo. La arquitectura general del sistema es la que se muestra en la figura 2.

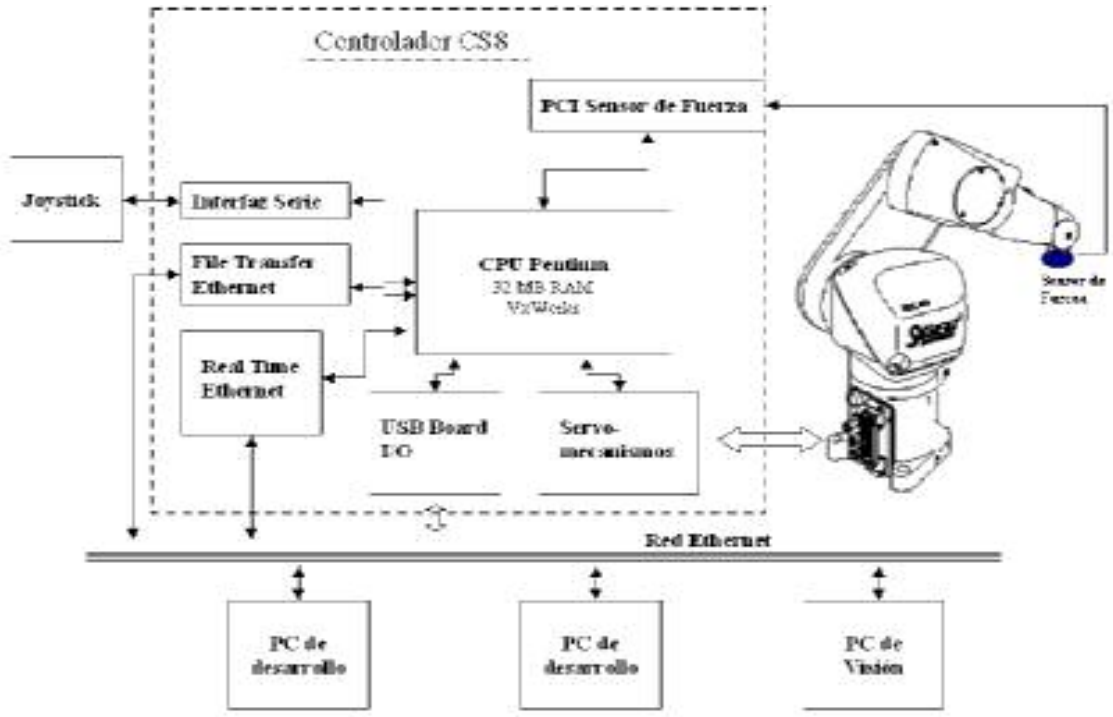

Figura 2. Arquitectura del sistema

El sistema está compuesto por el robot industrial Stäubli RX60B, un brazo articulado de seis grados de libertad, junto con el controlador CS8 del mismo fabricante. Este nuevo controlador está basado en un computador PC-Pentium, con una velocidad de procesamiento de 450 MHz. Además, dentro de la placa base cuenta con una serie de puertos PCI libres, utilizándose uno de ellos para la incorporación del sensor de fuerza y par de seis grados de libertad, necesario para la implementación de las estrategias de control de fuerza. El sistema Operativo con el que cuenta el robot Stäubli es el VxWorks, que garantiza la ejecución de los algoritmos de control en tiempo real.

Para el diseño y la implementación del software de control se han utilizado otros computadores de desarrollo, siendo las herramientas de trabajo utilizadas, por un lado, Tornado II – que es el sistema de desarrollo de VxWorks - y por otro, Microsoft Visual Studio como entorno de edición.

Para poder acceder directamente al control de las diferentes articulaciones del robot se ha utilizado un juego de rutinas específicas diseñadas especialmente por Stäubli, denominadas LLI (Low Level Interface), que permiten generar referencias en posición, en velocidad o en par para cada uno de los lazos de control de las seis articulaciones del robot.

También se han utilizado las herramientas Matlab y Simulink para la simulación fuera de línea de los algoritmos de control desarrollados y para la generación, a partir del paquete de software Real Time Workshop de Matlab, del código en C de los algoritmos para su posterior inclusión en el sistema real de control.

Otro de los objetivos para el que se pretende utilizar esta plataforma experimental es el del desarrollo de un sistema de teleoperación del robot manipulador, también para entornos con movimientos restringidos. En este sistema el operario dispone de información sensorial de la fuerza con la que el robot interactúa con el entorno a través de un dispositivo de telemanipulación motorizado. Estos motores permiten utilizar la información del sensor de fuerza y par para ejercer una resistencia al movimiento proporcional a la fuerza ejercida. Con este fin, se ha incluido en el sistema de control una conexión Etehrnet que permite transmitir la información del sensor entre el computador al que se conecta el mando de teleoperación y el controlador CS8.

2.1. Supervisión del Sistema

Dado que, al dejar de tener el software de control original la responsabilidad del control del robot, parte de las funciones de seguridad del control estándar CS8 han sido anuladas. Esto hace necesario el diseño y desarrollo de algunas funciones de supervisión, fundamentalmente orientadas hacia el mantenimiento de la seguridad, que

permitan tomar las acciones de emergencia adecuadas ante eventos como los que se describen a continuación:

1. Detección de sobrecalentamientos de los motores.

2. Supervisión de límites de posición y velocidades de cada uno de los ejes.

3. Contacto del brazo con un elemento extraño.

Además, el software de supervisión tiene como misión adicional reiniciar los motores y guardar información sobre errores que se produzcan (como es el caso referencias de par inaceptables, las cuales ocurren al utilizar ganancias demasiado elevadas o bien al excitar demasiado tiempo a los motores durante la realización de la identificación del sistema).

2.2. Control en Tiempo Real

Como ya ha sido comentado anteriormente, el sistema operativo utilizado en la plataforma es el VxWorks 5.2. Este sistema operativo, junto con el entorno de desarrollo Tornado II, ofrece una serie de herramientas especialmente útiles para la programación, depuración y monitorización de un sistema de control en tiempo Real.

El núcleo del RTOS (Real Time Operative System) de VxWorks , denominado Wind Microkernel, cuenta, entre otras, con la posibilidad de gestión de redes (TCP/IP), gestión de entradas y salidas, soporte de C++, etc. Además permite ejecución multitarea, soporte y gestión de las rutinas de interrupción y mecanismos de comunicación entre distintas tareas [1].

Como herramienta de desarrollo se ha utilizado el entorno Tornado [2], y como entorno de edición alternativo Microsoft Visual Studio. Diseñado para el desarrollo cruzado de software, la plataforma Tornado permite de un modo eficiente el desarrollo de aplicaciones, módulos objeto y de sistemas de tiempo real empotrados con un nivel mínimo de intrusión en el dispositivo final.

El entorno Tornado suministra un conjunto de herramientas independientemente de las necesidades del sistema objetivo. Las principales utilidades se ejecutan en un computador host con acceso compartido a una tabla de símbolos del sistema objetivo remoto. La Figura 3 muestra las relaciones entre los principales componentes interactivos de Tornado en el host y el sistema objetivo.

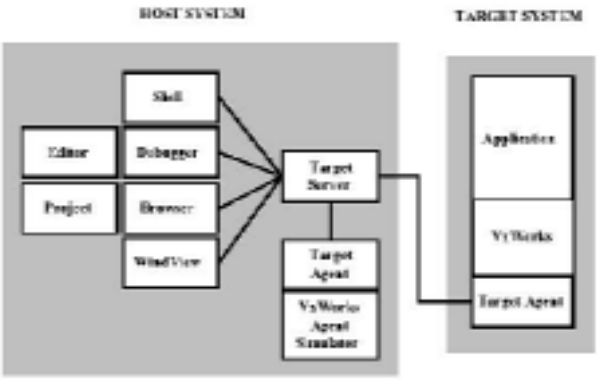

Figura 3. Interacción entre el Host y el sistema objetivo

Como herramienta de visualización dinámica de los cambios de contexto, los eventos que los originan, semáforos, watchdog times, etc., se ha utilizado el analizador de Tornado WindView.

Para la simulación inicial del sistema objetivo se utiliza el paquete VxSim, mientras que para la monitorización de los datos de las aplicaciones (por ejemplo las posiciones de las articulaciones) se ha utilizado el software StethoScope perteneciente a la misma plataforma.

3. ARQUITECTURA DEL SISTEMA DE CONTROL

Cuando se plantea el diseño de un sistema complejo desde el punto de vista de la programación es habitual dividir su estructura en capas o niveles. En la figura 4 se muestra la arquitectura de capas diseñada para el sistema de control del robot.

Figura 4. Arquitectura de capas del sistema de control

A continuación se analizan cada uno de los bloques de capas.

3.1. Capas del Usuario Final

En este nivel, el usuario debe definir la tarea que debe ser realizada por el robot, incluyendo no sólo los movimientos del mismo, sino también los esfuerzos que debe realizar y las direcciones en las que debe hacerlo al entrar en contacto con el entorno.

Sin embargo, el objetivo en la capa denominada nivel de tareas no es el conseguir un sistema de programación de tareas con un elevado nivel de abstracción, sino el de ejecutar simples operaciones donde se pongan de manifiesto las características de los algoritmos de control desarrollados.

3.2. Capas del Sistema

Se ha dividido la capa del sistema en dos niveles: el nivel inferior, identificado como la capa de aplicación, la cual estará encargada del control del movimiento del robot, y el nivel superior, identificado como la capa de ejecución, siendo su misión la de funcionar como interfase entre los distintos sensores y tareas a realizar, además de ejecutar el software de control en tiempo real.

En este bloque se distinguen los tres siguientes grupos de librerías:

1. Librerías de proceso (i.e. programación del nivel de prioridad de las interrupciones) y de los drivers para los dispositivos de entrada y salida y los sensores.

2. Librerías específicas del robot (i.e. sistema de supervisión) y lenguajes de programación específicos del robot.

3. Librerías para las aplicaciones de control de movimientos del robot (algoritmos de control de alto nivel).

3.3. Capas de los servos

La característica principal de este nivel es la estrecha relación que existe entre el hardware y el software. Por lo tanto el software ha de reflejar una estructura que coincida con la estructura hardware.

Teniendo en cuenta las características de control de un robot manipulador, este nivel se ha dividido en otros tres [4] , [6] (ver figura 5).

Figura 5. Estructura "Servo" en interacción con el sistema físico (J = articulación).

El sistema de control de movimientos proporciona un conjunto de funciones del robot que permiten al programador incorporar las funciones de control a más bajo nivel.

El sistema de control utilizado mantiene los controladores originales de bajo nivel de cada una de las articulaciones del robot. Sin embargo, a diferencia de lo habitual en los robots industriales, con esta arquitectura se pueden definir las referencias en posición, velocidad o para cada uno de ellos.

Esto se consigue a partir del software LLI (código C de acceso a bajo nivel), parte del cual ha sido desarrollado específicamente por Stäubli para esta aplicación. Las funciones de esta librería pueden clasificarse en dos tipos:

1. Funciones propias del control de los movimientos de las articulaciones.

2. Funciones encargadas de la adquisición de las variables de control.

Las funciones del tipo 2 pueden ser utilizadas en niveles superiores (como por ejemplo en la capa de aplicación) de manera que esta información puede utilizarse para la toma de decisiones de más alto nivel.

En la figura 6 se muestra un diagrama de bloques de este nivel de control

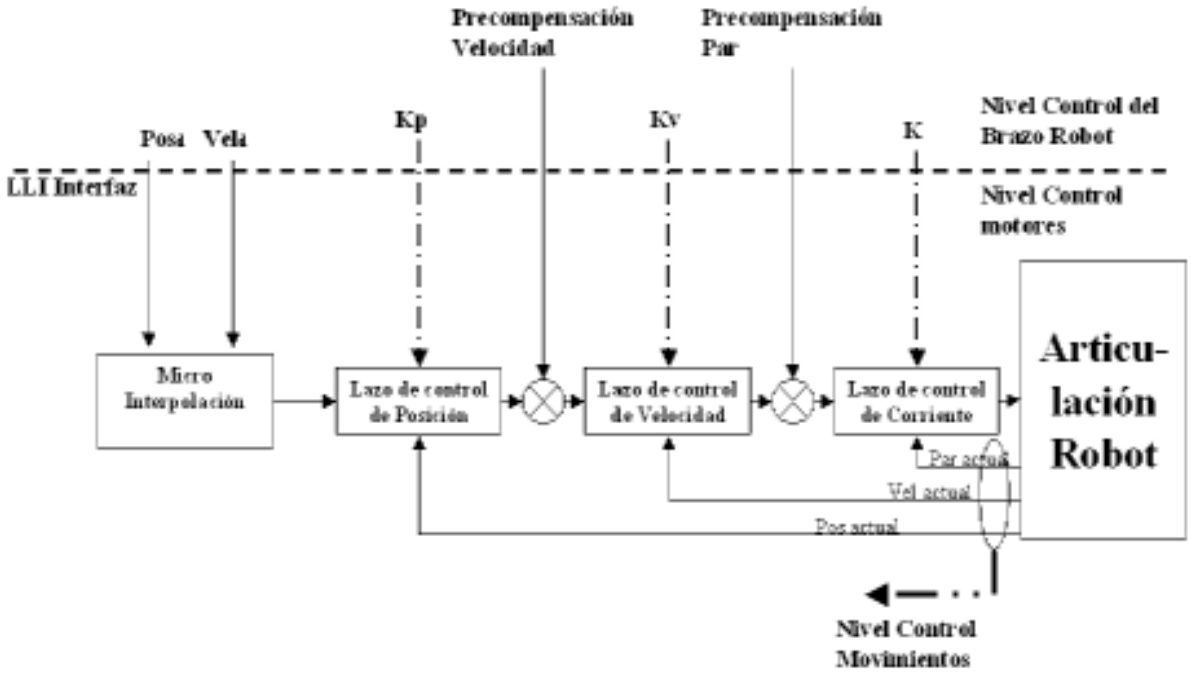

Figura 6. Estructura general del controlador de las articulaciones.

4. Ejemplo de Aplicación

Para una mejor comprensión de la arquitectura planteada, en la siguiente tabla se muestra un ejemplo completo de la realización de una tarea en la que se puede observar qué acciones corresponden a qué niveles.

Capa o Nivel	Acción
Nivel de Tarea	Lijado del borde de un tablero circular.
Nivel de Programación	Utilización del robot para indicar los puntos (programación OnLine).
Nivel de Ejecución	Ejecución en tiempo real del proceso de lijado. Lectura/Escritura de las I/O (i.e. Aspiradora de la viruta).
Control de la Aplicación	Utilización del Algoritmo de control multivariable diseñado (i.e. algoritmo de control no lineal con realimentación de fuerzas)
Control del Movimiento	Generación de trayectorias
Control del Brazo Robot	Configuración de los parámetros internos de control (Kp, Kv, K) de las articulaciones así como precompensación de la fricción. Manda los variables de control del robot al nivel "Control de la aplicación"
Control de los motores	Aplicación de las señales físicas a los motores. Lazos internos de control. Lectura de los sensores internos (i.e. encoder de posición).

5. Conclusiones.

A la hora de desarrollar una plataforma de experimentación adecuada para la implementación y prueba de algoritmos avanzados de control de robots manipuladores no resulta fácil encontrar un robot industrial con una arquitectura abierta, por lo que al final es necesario realizar una serie de adaptaciones ya sea en el Hardware como en el Software. Esta adaptaciones suelen ser complejas y costosas en tiempo y en financiación.

En este trabajo se han planteado las características más destacables de una arquitectura abierta para controlar un robot industrial RX60.

Cabe destacar que la colaboración de la empresa fabricante del robot ha sido de especial importancia, habiendo desarrollado un software específico para esta aplicación.

Bibliografía.

[1]**GUÍA DE REFERENCIA VXWORKS 5.2**. WindRiver.

[2] **GUÍA DE REFERENCIA TORNADO.** WindRiver.

[3] **INTEGRATED MOTIONS, INC**., 758 Gilman St., Berkeley, CA 94710, USA. Zebra-ZERO Force Control Robot, 1992.

[4] **K. NILSSON.** Industrial Robot Programming. PhD thesis ISRN LUTFD2/TFRT--1046--SE, May 1996. Department of Automatic Control, Lund Institute of Technology, Sweden

[5] **M.G. Ortega, F.R. Rubio and M. Román**, "Nonlinear H_infinity Control with PID Structure for Robot Manipulators." 15th World Congress International Federation of Automatic Control (IFAC-02), 21-26 July, Barcelona, España, 2002.

[6] **A. ROBERTSSON, JOHAN EKER, KLAS NILSSON, ROLF JOHANSSON**. Application-Specific Control for Industrial Robots. In Proc. 3rd Portuguese Conference on Automatic control---Controlo98, volume 2, pp. 877, Coimbra, Portugal, 1998