Ingeniería Mecánica. Vol. 15. No. 3, septiembre-diciembre, 2012, p. 233-242 ISSN 1815-5944

Artículo Original

Plataforma de desarrollo para el control en tiempo real de estructuras cinemáticas con realimentación visual

Platform to develop real time visual servoing control in kinematics systems

René González-Rodríguez[I], Luis Hernández-Santana[II]

I. Empresa de Automatización Integral CEDAI. Santa Clara, Cuba.
Correo electrónico: voltus@cedai.com.cu
II. Universidad Central Marta Abreu de Las Villas. Facultad de Ingeniería Eléctrica. Cuba

Recibido: 14 de febrero de 2012 Aceptado: 13 de junio de 2012

Resumen

En este trabajo se presenta una plataforma de desarrollo para el control en tiempo real de estructuras cinemáticas con realimentación visual. Se ha diseñado una configuración genérica que permite la implementación de cualquier variante de control visual. Para el procesamiento de la imagen se ha propuesto una estrategia que permite el uso de diferentes herramientas comerciales o algoritmos propios para la captura y extracción de características de la imagen. El uso de *Real Time Work Shop* y *Real Time Windows Target* en el lazo de control interno brinda la posibilidad de implementar algoritmos de control servovisual en tiempo real. Al final del trabajo se presentan los resultados de un esquema de control servovisual aplicado en un manipulador industrial. La plataforma propuesta constituye una herramienta de desarrollo para aplicaciones industriales de control servovisual y sirve de apoyo a la enseñanza de la mecatrónica en pregrado y postgrado.

Palabras claves: control servovisual, control en tiempo real, estructuras cinemáticas.

Abstract

In this work we propose a platform to develop visual servoing control systems. The platform has a generic design with the possibility to implement direct or look and move visual servoing systems. For the image processing we present a generic design allowing the use of any image processing library like Matrox MIL, Intel IPP, OpenCV or any algorithms for image capture and target characteristics extraction. The uses of Real Time Work Shop and Real Time Windows Target in the internal loop permits modify the control structure in SIMULINK very easy.

Key words: visual servoing, real time control, kinematics systems.

Introducción

El control de estructuras cinemáticas con realimentación visual es un tema de mucho interés en la comunidad científica actual. La necesidad de brindarle a las máquinas cada vez mayor independencia, ha provocado que reconocidos investigadores y centros de investigación a nivel mundial dediquen tiempo y recursos en crear sistemas autónomos. La cámara de video se ha convertido en el sensor que más información brinda del ambiente sin interactuar con éste. Esto provoca que los sistemas con realimentación visual,*visualservoing* como se conocen en inglés, estén en constante desarrollo. La literatura reporta gran cantidad de trabajos de aplicaciones y estrategias de control servovisual. La mayoría de los artículos están enfocados en aplicaciones con configuraciones específicas de robots manipuladores [1, 3] o robots móviles [4, 5].

Para una institución que se dedique al desarrollo de sistemas de control con realimentación visual es fundamental tener el banco de trabajo o plataforma experimental que permita la implementación y puesta a punto de los sistemas en estudio. Una problemática existente en la actualidad es la falta de plataformas o bancos de trabajo genéricos reutilizables que permitan el desarrollo de sistemas de control servovisual ya sea con fines industriales o docentes.

La literatura reporta ejemplos de plataformas de desarrollo. Marchand y colaboradores presentan *ViSP: Visual ServoingPlatform* consistente en una plataforma genérica con una amplia gama de controladores para robots manipuladores. El trabajo consiste en un ambiente de desarrollo hecho en C^{++} sobre Linux con las posibilidades de portabilidad, independencia de hardware y simplicidad. Esta plataforma brinda una extensa biblioteca de tareas elementales con varias características visuales que se pueden combinar entre ellas para obtener el sistema deseado. Posibilita el uso de una biblioteca de procesamiento de imagen que permite el seguimiento de señales visuales a la razón de muestreo del video, un simulador, una interfaz con varias tarjetas de adquisición de imágenes [6].

Otro reporte de plataforma de trabajo es presentado por Soria y colaboradores, quienes defienden una arquitectura para el desarrollo rápido de prototipos para control servovisual basado en productos estándares de hardware y software. El esquema presentado permite controlar una amplia variedad de sistemas servovisuales incluyendo robots manipuladores. El ambiente está basado en *MATLAB/SIMULINK*. Se presentan experimentos que validan el funcionamiento de la plataforma [7].

Por su parte, Corke presenta un *Toolbox* para MATLAB sobre visión y controles basados en visión nombrado *Machine VisionToolbox MVT*. Si bien no permite la implementación en sistemas reales, sí contribuye a la prueba de algoritmos en simulación [8].

Después de un profundo estudio de los sistemas reportados en la literatura se decidió diseñar e implementar toda la infraestructura de hardware y software que permita la realización de los experimentos y el desarrollo de futuros trabajos en el tema de control servovisual de estructuras cinemáticas. En el presente trabajo se propone una plataforma para el desarrollo de sistemas de control con realimentación visual. La plataforma tiene un diseño genérico que permite la implementación de cualquier estructura de control visual directo [9] o con control articular [10]. Para el procesamiento de la imagen se realiza un diseño que permite el uso de diferentes herramientas de procesamiento como las bibliotecas de la *Matrox MIL, Intel IPP*, de código abierto OpenCV o algoritmos propios para la captura y extracción de características de la imagen. El uso de *Real Time Work Shop* y *Real Time Windows Target* de MATLAB en el lazo de control interno brinda la posibilidad de implementar algoritmos de control servovisual directos o del tipo Vea y Mueva dinámicos en tiempo real. Además, las estructuras de control se pueden variar con relativa facilidad desde el ambiente gráfico del *SIMULINK*. Para validar el funcionamiento del sistema se presentan los resultados de una estructura de control del tipo Vea y Mueva con compensación cinemática en un robot manipulador industrial.

Método

Existen varios modelos de controladores para robots industriales según el tipo de accionamiento, sensores y fabricantes. Por ejemplo, la firma *Kuka* fabrica controladores KRC y KMC, con tecnología *Devicenet*, que pueden ser simulados y programados desde ambiente *Microsoft Windows*. ABB presenta controladores muy eficientes como el S4CPlus en el que se combinan una alta aceleración y el control de movimiento de ABB *Quickmove* con una precisión de $\pm 0{,}03mm$.

La mayoría de los robots industriales no permiten modificar el algoritmo de control articular, pero dan la posibilidad de comunicación vía RS232 para programar los valores deseados de los controladores, tanto articulares como cartesianos. Esta limitante es tomada como premisa para el diseño de la plataforma de desarrollo, donde se propone una unidad lógica de procesamiento para el control articular (análoga al control

cerrado del manipulador) que recibe los valores deseados desde otra unidad de procesamiento como se muestra en la figura 1.

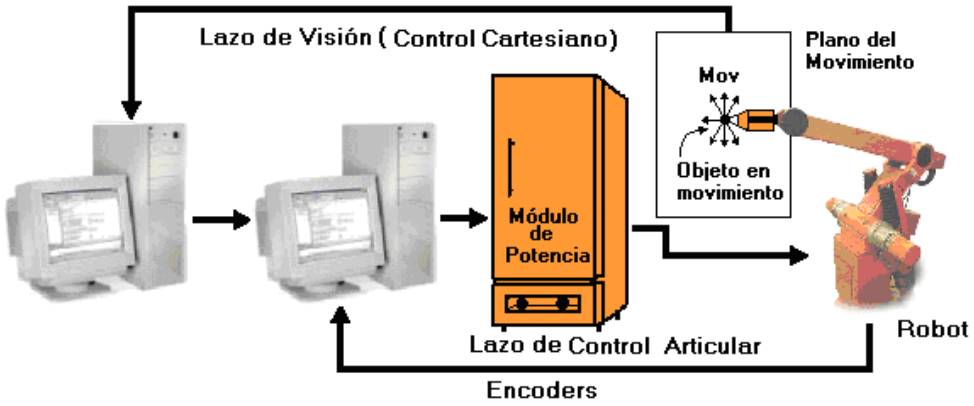

Fig. 1. Esquema de control de la plataforma de desarrollo propuesta

Esta configuración permite realizar experimentos bajo las mismas condiciones de los robots manipuladores industriales comerciales. Para el control articular se decidió usar una PC, en lugar de un controlador industrial comercial, con el objetivo de permitir la implementación de diferentes algoritmos de control de robots. De esta forma, se obtiene, como valor agregado, una herramienta experimental para la docencia y la investigación en teoría de control, accionamiento eléctrico y robótica.
Para el lazo de visión se propone una PC con el software de captura y procesamiento de imagen. La figura 2 muestra el diseño por capas del software de visión. Como se puede ver, el diseño tiene un enfoque genérico que posibilita el uso de diferentes bibliotecas para la captura y procesamiento de la imagen. Las capas están encapsuladas en clases relacionadas entre sí que abstraen el diseño de la implementación.

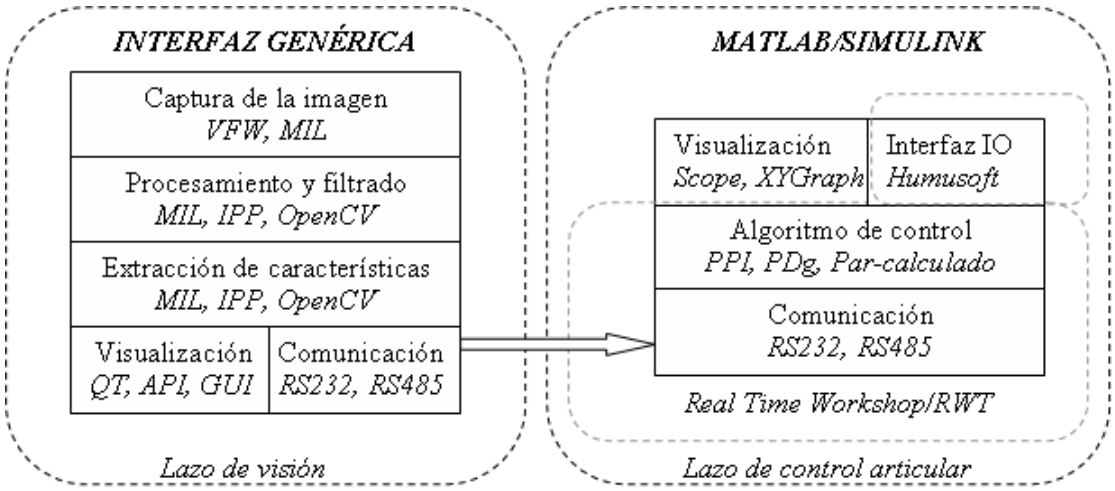

Fig. 2. Diagrama por capas del diseño propuesto

El lazo de visión consta de varias clases con relaciones de uso entre ellas. Por ejemplo, en la figura 2 se muestra la capa para la captura de imagen que es modelada en una clase, *imageCapture,*que encapsula las funcionalidades de captura. La adquisición puede realizarse usando la biblioteca *Video for Windows, MIL* ocualquier otra escogida por el usuario. La interfaz de la clase *imageCapture* es la misma, por lo que no se requiere recompilar el código completo, solamente el módulo que se desee variar según el hardware que se esté usando. Bajo esta filosofía se pueden implementar todas las capas del lazo de visión sin tener que modificar el código completo de la aplicación. Con esta herramienta, el usuario consta de un SDK (*software developer kit*) orientado al desarrollo de aplicaciones de procesamiento de imagen con las prestaciones comúnmente utilizadas en los sistemas de control con realimentación visual y la posibilidad de ampliarlas según las necesidades de la aplicación. El caso de la última capa tiene un tratamiento especial.

Está dividida en dos clases: *visualizationLayer* y *communicationLayer.* La primera es abstracta por lo que siempre debe ser implementada por el usuario en dependencia de la forma de visualizar la información. Es una tarea que lleva recursos y se le ha dado baja prioridad de ejecución para garantizar el tiempo real en las otras tareas de captura y procesamiento. Se pueden usar las APIs clásicas de Windows, QT, o cualquier interfaz gráfica según las necesidades de la aplicación.

Tiempo de muestro lazo de visión

El tiempo de muestreo es una variable fundamental en cualquier sistema de control. La figura 1 muestra dos lazos de control en cascada. El tiempo de muestreo del lazo externo, donde se ejecutan las tareas de captura y procesamiento de la imagen, depende de la potencialidad del sistema de adquisición de imágenes que se use. Por ejemplo, si se tiene una tarjeta de adquisición estándar con sistema NTSC a 30fps (cuadros por segundo) se puede lograr un tiempo de muestreo de 33.3 ms o 40ms en sistema estándar PAL. En este tiempo se debe procesar y filtrar la imagen, extraer las características y enviar los puntos vía RS232 al control articular. Si los algoritmos de procesamiento y extracción de características consumen más tiempo que la frecuencia de captura del sistema de visión, entonces se perderán algunos cuadros y el tiempo de muestreo estará dado por la suma de todos los tiempos de procesamiento de la imagen más el tiempo de transmisión de los datos.

Bibliotecas para la captura y procesamiento de imagen

En la actualidad, existen varias bibliotecas para el procesamiento de imagen. Por las prestaciones que brindan se destacan:

Open SourceComputerVisionOpenCV: es una biblioteca de visión artificial basada en código abierto. Originalmente fue desarrollada por Intel. Desde que apareció su primera versión, alfa, en el mes de enero de 1999, se ha utilizado en infinidad de aplicaciones como sistemas de seguridad con detección de movimiento o aplicaciones de control de procesos donde se requiere reconocimiento de objetos. *OpenCV* es multiplataforma, existiendo versiones para Linux, Mac OS X y Windows. Contiene más de 500 funciones que abarcan una gran gama de áreas en el proceso de visión, como reconocimiento de objetos, reconocimiento facial, calibración de cámaras, estéreo visión y visión robótica. El proyecto *OpenCV* pretende proveer un *Tool-Kit* o marco de desarrollo fácil de utilizar y altamente eficiente. Esto se ha logrado realizando su programación en código C y C^{++} optimizados, aprovechando las capacidades que proveen los procesadores multi núcleo. OpenCV puede, además, integrarse con la Intel IPP para aumentar su rendimiento en procesadores Intel.

Intel IPP: IPP es el acrónimo en inglés de *Integrated Performance Primitives*. *Intel IPP* está formado por una biblioteca software de funciones para el procesamiento de imágenes, señales, y operaciones matemáticas, que han sido optimizadas para múltiples sistemas operativos y procesadores. Se sirve de una biblioteca de bajo nivel que extrae la funcionalidad del procesador, de ahí que haya una versión de la biblioteca específica para cada procesador. Tiene un API común, es decir, que el aspecto del código es el mismo para todas las versiones, mientras que la puesta en práctica de la función subyacente tiene en cuenta las variaciones de las arquitecturas del procesador. Intel IPP está actualmente disponible para Windows y Linux y sólo es válida en procesadores Intel.

MatroxImaging Library (MIL): es una colección de herramientas de software para desarrollar aplicaciones en procesamiento de imágenes, visión por computador, imágenes médicas, y análisis de videos. La MIL está compuesta por programas interactivos y funciones para la captura de imagen, procesamiento, análisis, anotaciones, visualización y almacenamiento. Se integra con la mayoría de las tarjetas de captura, *framegrabbers*, aunque obtienen el mayor rendimiento con las tarjetas propias de la *Matrox*.

Software para el control articular

Para el lazo de control articular se decidió usar el *MATLAB* como sistema base. De esta forma se obtendrían las ventajas de cálculo, simulación y herramientas para el control automático que brinda el *MATLAB*.

El tiempo de muestreo del control articular está limitado por el tiempo de los algoritmos de lectura de los sensores, por la complejidad del algoritmo de control y por el tiempo de ejecución de las operaciones de salida hacia los actuadores. Normalmente los sistemas de adquisición de datos trabajan de forma paralela al procesamiento (*pipeline*) por lo que las entradas y salidas se pueden realizar con un retardo de muestreo, quedando como tiempo determinante en la subrutina de tiempo real el tiempo que demora en ejecutarse el algoritmo de control. Para garantizar un tiempo de muestreo estable en un sistema operativo que no es de tiempo real como Windows, se realiza el diseño para el control articular basado en los *Toolboxs*: *Extended Real-Time Workshop* y *Real-Time Windows Target* de *MATLAB*, los cuales permiten realizar la ejecución en

tiempo real de un esquema *SIMULINK* sobre un determinado sistema de adquisición de datos, logrando la interacción sobre el sistema físico conectado a él.Por una parte, *Real-Time Workshop* proporciona la conexión en tiempo real con el sistema de adquisición de datos, mientras que *Real Time Windows Target* permite la ejecución de esquemas *SIMULINK* sobre *Real Time Workshop*. Con esta configuración se pueden bajar los tiempos de muestreo en el lazo de control interno hasta 1ms.

El hecho de ejecutar directamente un esquema *SIMULINK* supone una ventaja añadida ya que el tiempo y la complejidad de trabajar directamente en su ambiente gráfico se reducen en gran medida, permitiendo una fácil creación y modificación de esquemas.

Para el uso de hardware no convencional, que no esté en las bibliotecas que por defecto tiene *MATLAB*, o para la implementación de algoritmos específicos de control articular, se proponen plantillas para la programación y ejecución en tiempo real. Las plantillas son *S-Functions* que permiten cargarse en tiempo de corrida y su ejecución en tiempo realpara adicionar funcionalidad a nivel de bloque al *SIMULINK*. Debido a las exigencias y limitantes de las *S-Function* que se ejecutan en tiempo real, se crea una interfaz para facilitar el diseño con el uso de *Scripts* que automatizan el proceso de implementación. Como se puede ver en la figura 3, la interfaz para la creación de lãs *S-Functions* está dividida en:

- *Datos de la S-Function*. Nombre de la función, número de variables de entradas, número de variables de salidas.
- *Código de la función principal*. En este Script se implementa en C el algoritmo principal de la *S-Function*. Se especifican las entradas como u[0], u[1]...u[n] y las salidas como y[0], y[1]...y[n].
- *Variables globales y funciones auxiliares*. Las variables globales declaradas en este *Script*son visibles y pueden ser usadas desde cualquier lugar de la *S-Function*, las funciones auxiliares son fragmentos de código que se necesiten ejecutaren cualquiera de las partes de la *S-Function*, por ejemplo el acceso a puertos y zonas de memoria, cálculos auxiliares, etc.
- *Función de inicialización*. Se ejecuta una sola vez cuando se inicia el proceso de corrida. Se usa para configurar e inicializar puertos, *timers*, *hardware*, habilitar *buffers.*
- *Función de terminación*. Se ejecuta una sola vez al terminar la ejecución. Se usa para restablecer las configuraciones de los puertos y hardware en general, limpiar *buffers*, almacenar resultados *offline* o cualquier tarea de terminación.

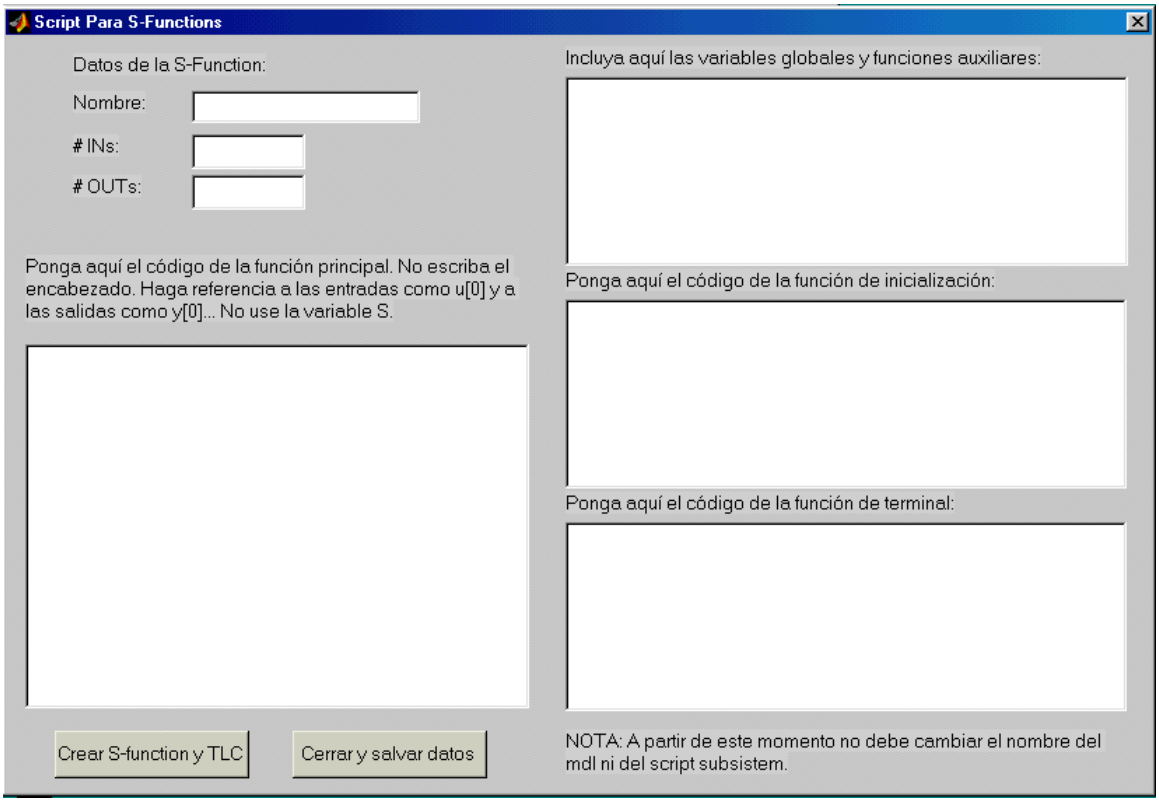

Fig. 3. Interfaz para generación de *S-Functions* a partir de *Scripts*.

La interfaz para la generación de *S-Function*a partir de *Scripts*fue utilizada para crear el módulo de comunicación vía RS232/485 con la PC que ejecuta la captura, procesamiento y extracción de características de la imagen.

Implementación

Para demostrar el funcionamiento de la plataforma se realizaron varios experimentos en un robot *ASEA IRB6* con arquitectura de control abierta. El lazo interno es implementado en una PC *Intel Pentium* III a 500MHz conectada al robot a través de una tarjeta de adquisición *Humusoft* MF624 que se encarga de leer de los encoders la posición articular de cada articulación y dar la señal de mando, a los drivers de corriente directa del motor, generados por el algoritmo de control con un tiempo de muestreo de 1ms. La señal de video es tomada a través de una tarjeta de captura (*framegrabber*) EZ-*Capture* con *chipset* BT878 montada en una segunda computadora *Intel Celeron* a 2.0G Hz la cual procesa la imagen, extrae las características del objeto, resuelve el problema cinemático inverso y envía los datos vía RS232 a la primera PC cada 50 ms como lo muestra la figura 1.

Software para la captura y procesamiento de imagen

Para el sistema de visión se usó una cámara monocromática JAI CV-252. Se desarrolló un software de captura, procesamiento, control del lazo externo, cálculo de la cinemática inversa y módulo de comunicación. La figura 4 muestra una de las ventanas de parametrización del software desarrollado. Para optimizar el procesamiento de la imagen en tiempo real se decidió usar las bibliotecas *MatroxImaging Library*.

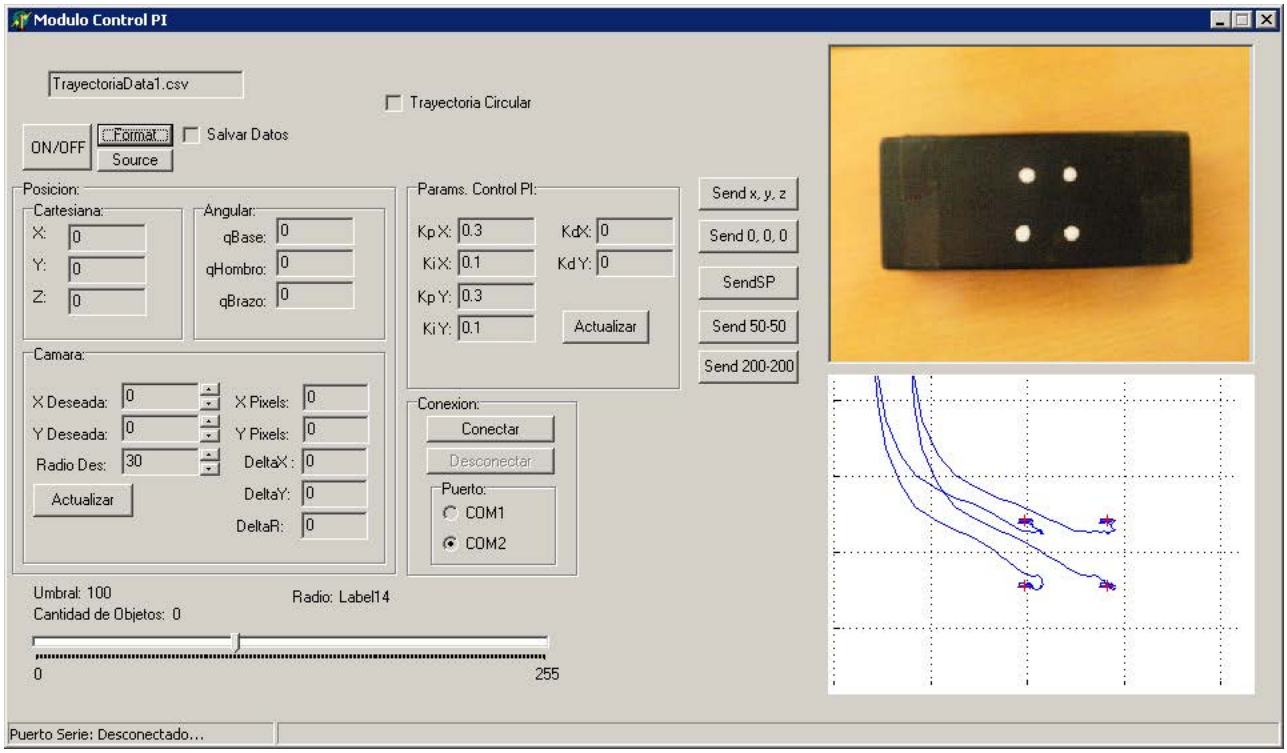

Fig. 4. Ventana de parametrización del software desarrollado para el sistema de visión

Una importante operación en el control de robots mediante retroalimentación visual es La determinación de las coordenadas de un punto característico de la imagen. Generalmente, esta operación es realizada a través del cálculo del centroide.

Para la extracción de características del objeto se usaron varias técnicas de procesamiento de imágenes para el filtrado y mejoramiento de la misma. Se implementó un algoritmo capaz de filtrar y detectar el centroide de un objeto, el área y la distancia entre dos puntos en un rango determinado. Usando la biblioteca MIL, el algoritmo en forma secuencial queda de la siguiente forma:

1. Binarizar la imagen.
2. Quitar las partículas pequeñas y luego los orificios (*Close Open*).
3. Asignar espacio para la lista de características
4. Asignar un buffer para el resultado.

5. Especificar el área deseada.
6. Calcular las características seleccionadas para cada objeto.
7. Excluir objetos cuya área sea muy pequeña.
8. Marcar el centro de gravedad.
9. Seleccionar el centro de gravedad como la característica de interés.
10. Calcular el área y la distancia entre dos puntos del objeto para el experimento 3D.

El sistema de visión usando el modelo *Pinhole*dela cámara ha sido descrito por varios autores [11, 12]. El modelo implementado en el experimento es el presentado por Hernández en [10]:

$$\delta\boldsymbol{\xi}=\begin{bmatrix}\delta u\\ \delta v\\ \delta d\end{bmatrix}=\alpha\frac{\lambda}{P^C_{zo}}\begin{bmatrix}C\psi & S\psi & 0\\ -S\psi & C\psi & 0\\ 0 & 0 & -\dfrac{d_o}{(P^C_{zo})}\end{bmatrix}\begin{bmatrix}\delta P^C_{xo}\\ \delta P^C_{yo}\\ \delta P^C_{zo}\end{bmatrix} \quad (1)$$

Donde δ significa variaciones alrededor del punto de operación, $[u\ v\ d\ ]^T$ es el vector de características de la imagen (posición del centroide y distancia entre dos puntos en el plano imagen), α es el factor de escala del lente, λ la distancia focal, $[P^C_{xo}\ P^C_{yo}, P^C_{zo}]$ el vector de posición del objeto con respecto a la cámara y $C\psi$ y $S\psi$ son los cosenos y senos del ángulo de rotación ψ entre el eje coordenado de la cámara y el robot.

Resultados

Con el objetivo de validar el sistema propuesto se realizó el experimento de posicionar el robot ABB IRB6 con respecto a un objeto que se mueve en el espacio. Dentro de las estructuras de control servovisual, reportadas en la literatura [13, 15] se decidió implementar una aproximación de la variante de Control Servovisual Basado en Imagen con compensación cinemática que se muestra en la figura 5 y que fue presentada por González-Rodríguez y colaboradores [16]. En este caso con varios puntos característicos.

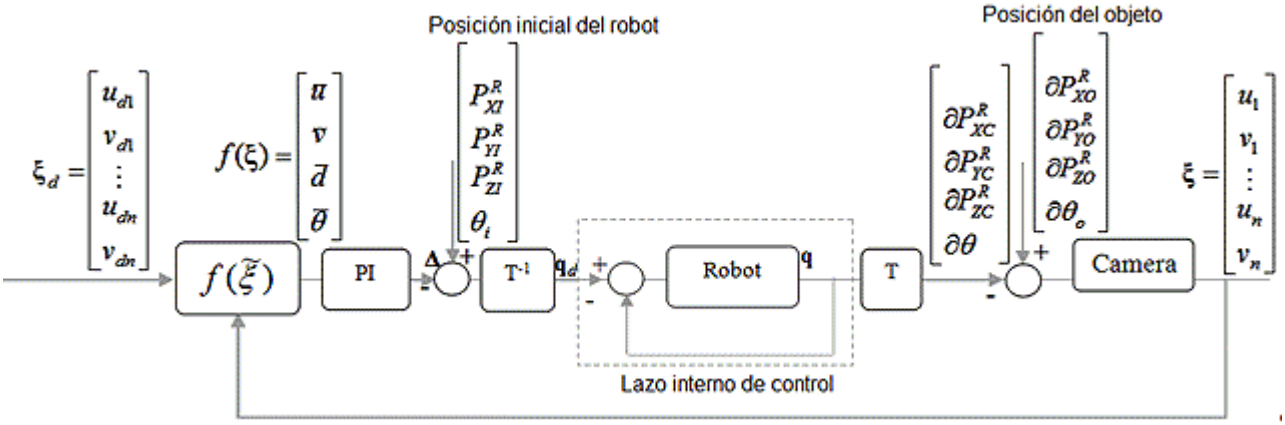

Fig. 5. Diagrama de control servovisual implementado.

La función $f(\xi)$ se encarga de convertir los puntos característicos del vector de la imagen ξ en movimientos por cada grado de libertad. Esta estructura brinda la posibilidad de ser un sistema de muy fácil implementación en robots manipuladores industriales con control articular cerrado sin la necesidad de usar el modelo del Jacobiano de la imagen. El vector de estado del objeto solo puede ser medido por la cámara, por lo que, el conocimiento directo de los valores deseados de los ángulos de las articulaciones, $\mathbf{q}_d$, no es conocido. Sin embargo, el vector de posiciones articulares puede ser obtenido como resultado de la estimación de la señal de control Δ y la solución del problema cinemático.El control interno está desarrollado con una arquitectura abierta donde puede ser implementado cualquier tipo de controlador. Una posibilidad es usar un control no lineal en el estado de las variables conocido como torque-calculado con la siguiente ecuación:

$$\boldsymbol{\tau} = \boldsymbol{M}(\boldsymbol{q})\left[\ddot{\boldsymbol{q}}_d + \boldsymbol{K}_{vi}\dot{\tilde{\boldsymbol{q}}} + \boldsymbol{K}_{pi}\tilde{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q})\right] \quad (2)$$

Donde$\boldsymbol{M}(\boldsymbol{q})$es la matriz de inercia, $\mathbf{q}$ el vector de movimientos articulares, $\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})$vector de fuerza centrípeta y coriolis, $\boldsymbol{g}(\boldsymbol{q})$vector de torque gravitacional, $\boldsymbol{\tau}$,vector de torque aplicado al robot, $\boldsymbol{K_p} \in \Re^{n\times n} y \boldsymbol{K}_v \in \Re^{n\times n}$ matrices simétricas definidas positivas y $\tilde{\mathbf{q}} = \mathbf{q} - \mathbf{q}_{\boldsymbol{d}}$. En [17] se demuestra que con esta configuración el sistema enlazo cerrado se comporta como un sistema lineal multivariable desacoplado en cada una de las articulaciones del robot, sugiriendo que las matrices pueden ser expresadas como:$\boldsymbol{K_{pi}} = diag\{\omega_1^2, \cdots, \omega_n^2\}y\boldsymbol{K_{vi}} = diag\{\omega_1, \cdots, \omega_n\}$.De esta forma cada articulación se comporta como un sistema lineal críticamente amortiguado, de segundo orden, con ancho de banda ω_i que determina la velocidad de respuesta de cada articulación. En tal sentido, el efecto dinámico del lazo interno es independiente con respecto al lazo externo, siempre que se cumpla la condición $\mathbf{q} \cong \mathbf{q}_d \; \forall t > 0$En este artículo se considera una simple aproximación que consiste en usar el vector de características de la imagen $\tilde{\boldsymbol{\xi}}$, como la diferencia entre el centro del plano imagen $\boldsymbol{\xi}_d$ y el centro de gravedad del objeto en el espacio de la imagen $\boldsymbol{\xi}$; y la diferencia entre una distancia deseada en el plano imagen d_d y la correspondiente distancia entre dos puntos del objeto en el plano imagen *d*.

La descripción geométrica y el cálculo de la cinemática que se utilizó para los experimentos es presentada en [18]. La figura 6 muestra la plataforma de desarrollo implementada en un robot *ASEA IRB6*.

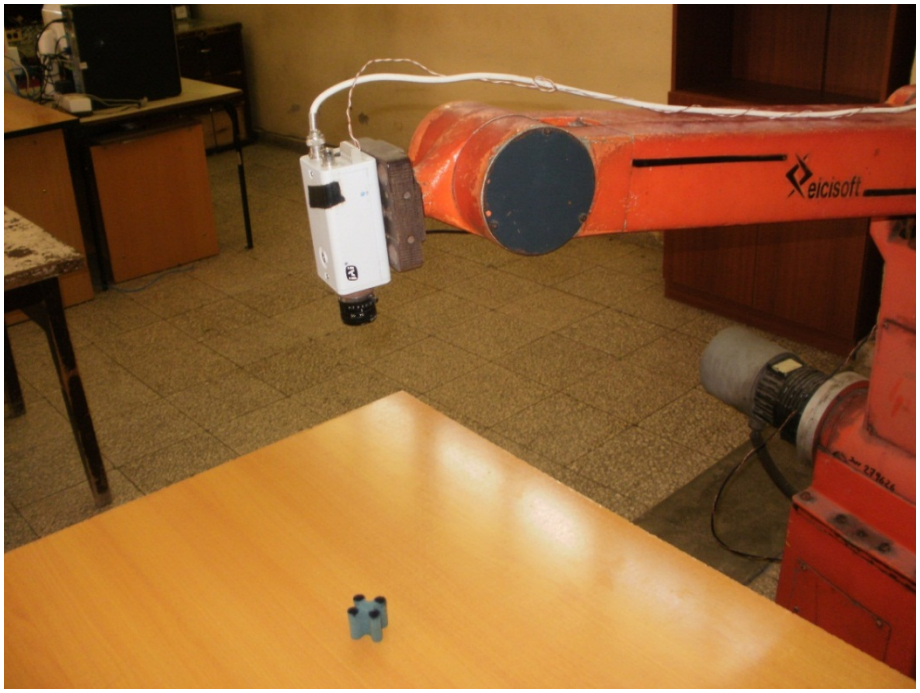

Fig. 6. Estación experimental

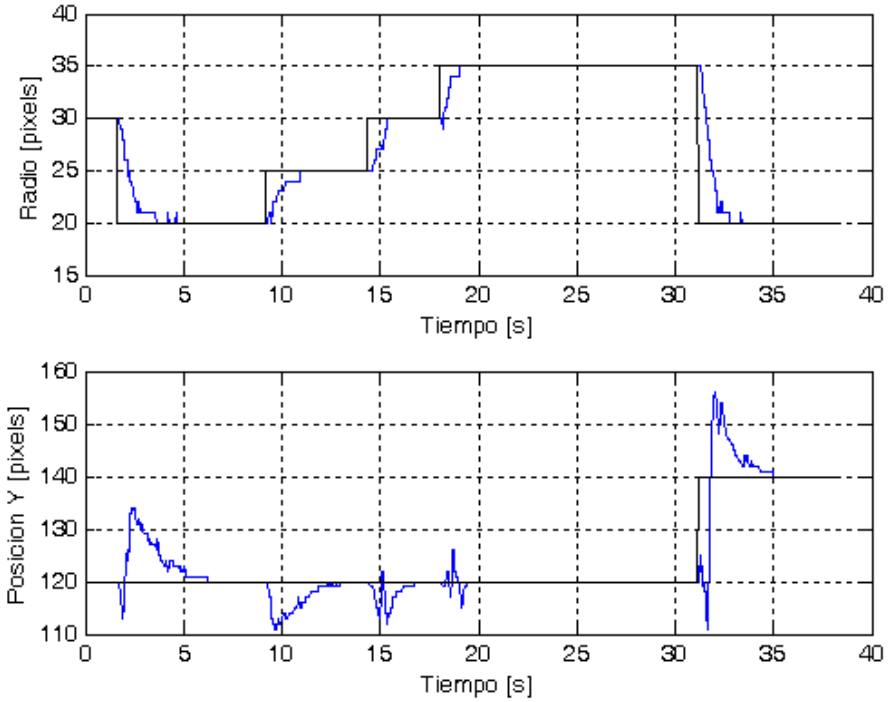

Fig. 7. Respuesta del control de profundidad (distancia) frente a perturbaciones

Para corroborar los resultados teóricos se realizan tres experimentos:entradas tipo escalón en las coordenadas de la imagen, mover el extremo operativo del robot alrededor del objeto en forma de círculo y crear perturbaciones en coordenadas cartesianas para analizar su influencia en las otras coordenadas y graficar la robustez del sistema. La figura 7 muestra los efectos que se tienen sobre el control de profundidad cuando se aplican perturbaciones en el manipulador. Se puede apreciar el buen comportamiento del sistema demostrándose la estabilidad y robustez del mismo.

Discusión

Como se pudo apreciar en la sección anterior, la plataforma propuesta permite la implementación de sistemas servovisuales con relativa facilidad. Los tiempos de muestreo requeridos para estos sistemas son garantizados. El lazo externo depende del sistema de adquisición de imágenes y los algoritmos de procesamiento. La estructura propuesta permite la utilización de hardware estándar (30 ms para NTSC a 40 ms PAL) o sistemas de adquisición especiales con mayor velocidad en la captura de la imagen (fps). Para el caso del lazo de control interno, control articular, se pueden lograr tiempos de muestreo de hasta 1ms.

El uso de las interfaces para la creación de las *S-Function*brinda al usuario uma herramienta que humaniza y agiliza el trabajo con el hardware, la interacción con las interfaces de entrada salida y la creación de algoritmos de control que deben correr en tiempo real.

Conclusiones

La plataforma presentada facilita el diseño de los experimentos para estructuras cinemáticas con realimentación visual a través de herramientas de visión artificial y un ambiente gráfico (*SIMULINK*) que permite el cambio de estructura y su posterior implementación en tiempo real. El diseño de una interfaz genérica para el lazo de visión brinda al usuario una herramienta de desarrollo (SDK) para aplicaciones que requieran captura, procesamiento y extracción de características de imágenes con la posibilidad de usar cualquier biblioteca sin tener que modificar todo el código. Se muestran los resultados de algunos experimentos realizados en un robot ASEAIRB6 donde se corrobora la estabilidad y el buen funcionamiento en tiempo real de la plataforma propuesta. Esta plataforma constituye una herramienta de desarrollo para aplicaciones industriales de control servovisual y sirve de apoyo a la enseñanza de la mecatrónica en pregrado y postgrado.

Referencias

1. Kosmopoulos, D.I., "Robust Jacobian matrix estimation for image-based visual servoing". *Robotics and Computer-Integrated Manufacturing*, 2010. vol. 27, nº. 1, p. 82-87.. ISSN 0736-5845.
2. Hernandez, L., Sahli, H. y González, R. “Vision-based 2D and 3D Control of Robot Manipulators”. En: Robot Manipulators Trends and Development. InTech. 2010. p. 441-462. ISBN:978-953-307-073-5.
3. Cid, J. and F. Reyes. "Visual Servoing Controller for Robot Manipulators". En: *International Conference on Electrical, Communications and Computers CONIELECOMP*. 2009. ISBN 978-0-7695-3587-6.
4. de la Fuente, M.I., Echanobe, J., del Campo, I. *et al.* "Hardware Implementation of a Neural-Network Recognition Module for Visual Servoing in a Mobile Robot.". DEXA Workshops. 2010. ISBN 978-0-7695-4174-7.
5. Hadj-Abdelkader, H., Mezouar, Y., Martinet, P. *et al.* "Catadioptric Visual Servoing From 3-D Straight Lines". *IEEE Transactions on Robotics*. 2008. vol. 24, nº. 3, p. 652-665. ISSN 1552-3098.
6. Marchand, E., Spindler, F. and Chaumette, F. "ViSP for visual servoing: a generic software platform with a wide class of robot control skills". *IEEE Robotics and Automation Magazine*, 2005. vol. 12, nº. 4, p 40-52. ISSN 1070-9932.
7. Soria, A., Garrido, R., Vázquez, I. *et al.* "Architecture for rapid prototyping of visual controllers". *Robotics and Autonomous Systems*, 2005. vol. 54, nº. 6, p. 486-495. ISSN 0921-8890.
8. Corke, P.I. "The Machine Vision Toolbox: A MATLAB Toolbox for Vision and Vision-Based Control". *IEEE Robotics and Automation Magazine*, 2005. vol. 12, nº. 4, p. 16-25. ISSN 1070-9932.
9. Kelly, R., Bugarin, E., Cervantes, I. *et al.* "Monocular direct visual servoing for regulation of manipulators moving in the 3D Cartesian space". En: *45th IEEE Conference on Decision and Control*. San Diego, CA, USA. 2006.
10. Hernández, L., González, R., Sahli, H. *et al*. "Simple Solution for Visual Servoing of Camera-in-hand Robots in the 3D Cartesian Space". En: *10th International Conference on Control, Automation, Robotics and Vision*. Hanoi, Vietnam. 2008. ISBN 978-1-4244-2286-9.
11. Hutchinson, S., Hager, G.D. y Corke, P.I. "A tutorial on visual servo control". *IEEE Transactions on Robotics and Automation*, 1996. vol. 12, nº. 5, p. 651-670. ISSN 1042-296X.
12. 12. Corke, P.I. y Hutchinson, S. "Real-Time Vision, Tracking and Control". En: *International Conference on Robotics and Automation*. 2000. ISBN 0-7803-5889-9.
13. Kelly, R., "Robust Asymptotically Stable Visual Servoing of Planar Robots". *IEEE Transactions on Robotics and Automation*, 1996, vol. 12, nº. 5. ISSN 1042-296X.
14. Kelly, R., Carelli, R., Nasisi, O. B. "Stable Visual Servoing of Camera-in-Hand Robotic Systems". *IEEE/ASME Transactions on Mechatronics*, 2000. vol. 5, nº. 1, p. 39-48. ISSN 1083-4435.
15. Chaumette, F. y Hutchinson, S. "Visual Servo Control. Part I Basic Approaches". *IEEE Robotics and Automation Magazine*, 2006. vol. 13, nº.. 4, p. 82-90. ISSN 1070-9932.
16. González-Rodríguez, R. y Santana, L.H. "Control monocular 3D dinámico basado en imagen", *Revista de Ingeniería Electrónica, Automática y Comunicaciones.* 2011. vol. 32, nº. 2, p. 15-30. ISSN 1815-5928.
17. Kelly, R. y Santibáñez, V. *Control de Movimiento de Robots Manipuladores*. Madrid: Pearson Education.. 2003. ISBN:84-205-3831-0.
18. Gonzalez-Rodríguez, R., Santana, L.H., Izaguirre, E. *et al.* "Estrategia de control para robots manipuladores con realimentación visual y plataforma electro-neumática de 3gdl", *Ingeniería Mecánica*. 2011. vol. 14, nº. 3, p. 245-257. ISSN 1815-5944.