Implementación de una arquitectura para control y verificación de un sistema de teleoperación por medio de LabVIEW

Isidro González Tobías[1], Manuel Nandayapa Alfaro[1], Osslan Osiris Vergara Villegas[1], Ángel Flores Abad[1], Raúl Ñeco Caberta[1]

[1]Universidad Autónoma de Ciudad Juárez.

Resumen

El control bilateral es un método de teleoperación con aplicaciones en diferentes áreas como cirugías mínimamente invasivas, exploración espacial, manejo de materiales peligrosos, entre otras. En un sistema de teleoperación convencional, el flujo de información es unidireccional, solo de maestro a esclavo. Por otro lado, con la implementación del control bilateral el esclavo puede retroalimentar fuerza hacia el maestro. La retroalimentación de fuerza permite al maestro sentir las características del ambiente remoto a través del esclavo. Por lo tanto, en el control bilateral se requiere un alto grado de fidelidad o transparencia entre maestro y esclavo. En este proyecto de se implementa una arquitectura para la implementación y control de un sistema bilateral, a través de la integración de la plataforma LabVIEW Real Time y un FPGA Virtex 5, utilizando también un novedoso método de estimación de velocidad y una estrategia matemática denominada observador de perturbaciones para estimar las fuerzas a partir de la velocidad obtenida. Al utilizar una plataforma FPGA, se toma ventaja de las altas frecuencias de procesamiento para crear un algoritmo robusto para estimación de velocidad y aprovechando las bondades de un sistema operativo en tiempo real, se obtiene una arquitectura estable y la vez robusta.

Palabras clave: Control bilateral, Teleoperación, LabVIEW.

Introducción

La necesidad de crear sistemas con un alto nivel de precisión y que además tengan cierto grado de independencia, ha devenido en el desarrollo de diversas técnicas destinadas al control de dichos sistemas. La importancia del control radica en la necesidad de crear sistemas que sean robustos y estables en determinadas condiciones de trabajo, esto con el fin de minimizar las perturbaciones que puedan interferir con el desarrollo de la tarea para la que fueron creados. La velocidad y la aceleración se han implementado como estrategias de control de movimiento de manera conjunta con otros procedimientos, como por ejemplo en (Ji y Sul, 1995) donde se combina el uso de un filtro Kalman con un observador, el cual es un modelo con el que se pueden estimar diferentes variables de un sistema, en este caso se estima la velocidad para utilizarla en el control de movimiento. También en trabajos como en

(Nandayapa, Mitsantizuk y Ohishi, 2011) donde se realiza el control obteniendo una estimación de la velocidad y combinando con un observador.

El desarrollo de nuevas tecnologías ha permitido crear sistemas robóticos autónomos. Mediante el uso de diversos sensores y complejos algoritmos de programación, los sistemas autónomos requieren muy poca o ninguna intervención humana para operar. Sin embargo, algunas aplicaciones siempre necesitarán control directo del humano. Por lo tanto, aun cuando se tengan sistemas autónomos, la teleoperación siempre será una modalidad de operación tanto retadora como indispensable. Estos sistemas teleoperados se componen de un teleoperador (maestro) y un teleoperado (esclavo). El teleoperador es un dispositivo que permite a un operador humano manipular y sentir objetos a distancia. En contraste, el teleoperado recibe órdenes del teleoperador y las ejecuta en tiempo real (Cui, Tosunoglu, Roberts, Moore y Repperger, 2003). Las aplicaciones prácticas de estos sistemas van dirigidas a aplicaciones: medicas, como cirugías teleoperadas (Kasahara, Kawana, Usuda y Onishi, 2012; Okamura, 2009); farmacéuticas y de laboratorios, como la nanomanipulación (Onal, Pawashe y Sitti, 2007; Daunay y Régnier, 2009); o industriales, para aplicaciones en ambientes hostiles. Una ventaja de estos sistemas radica en el nivel de intrusividad tan bajo con el que cuentan, esto se debe a que no dependen de sensores de retroalimentación de la fuerza de reacción del ambiente (Kosugi y Katsura, 2012; Susa, Shimono, Takei, Atsuta, Shimojima, Ozawa, Morikawa y Ohnishi, 2008), independiente de la aplicación, la transmisión de las sensaciones táctiles es absolutamente imperativa para lograr teleoperación robótica precisa. Una solución prometedora para atender este problema es un método de teleoperación llamado control bilateral. En el control bilateral, un operador manipula el maestro, y el esclavo retroalimenta la información táctil que se encuentra sujeta a la ley de acción y reacción. Además, es posible extenderse a un sistema de control multilateral si se cuenta con múltiples maestros y/o esclavos (Yamanouchi, Yajima y Katsura, 2012).

Métodos

Para lograr la implementación del sistema de control bilateral se siguieron una serie de pasos mostrados en la Figura 1, los cuales contienen una parte específica de bloques de programación que en conjunto conforman la arquitectura planteada para lograr los resultados obtenidos y se explican más a detalle en las siguientes secciones:

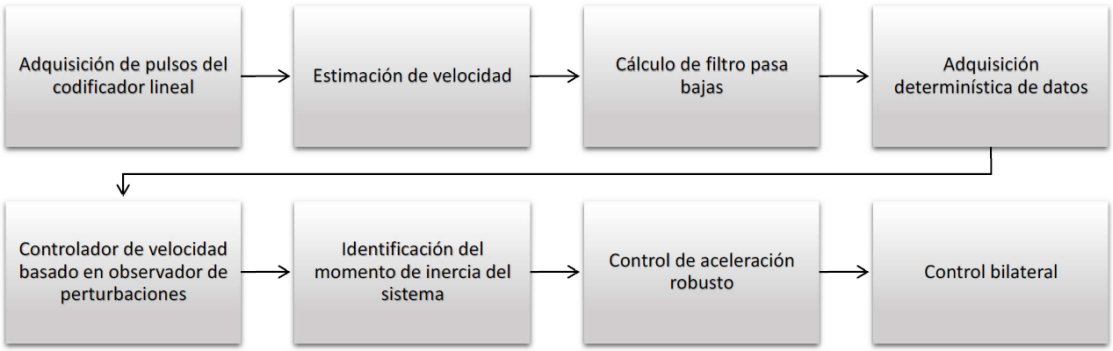

Figura 1. Fases seguidas en el desarrollo del proyecto.

Adquisición de pulsos del codificador lineal.

Para el cálculo de la posición y velocidad del sistema se implementó un algoritmo para la lectura de pulsos del codificador lineal del servomotor, el cual cuenta con una resolución de 2000 segmentos por revolución. Dicho algoritmo adquiere los pulsos provenientes del codificador y las compara con su estado previo, por ejemplo, si ocurre primero un pulso en A y luego en B, el disco está girando en sentido del reloj. Si tiene lugar primero un pulso en B y luego en A, entonces el disco está rotando en el sentido inverso a las agujas del reloj. Por lo tanto, si se monitorea tanto el número de pulsos como la fase relativa de las señales A y B, se puede hacer un seguimiento de la posición y de la dirección de la rotación. Además el algoritmo cuenta con un filtro digital, el cual se asegura que los valores de las señales sean constantes antes de tomarlos como válidos. La programación del algoritmo de lectura de pulsos fue realizada en lenguaje VHDL, una vez terminado se insertó en el programa de LabVIEW. En la Figura 2 se observa la sección del programa encargada del conteo de pulsos del codificador lineal de los dos servomotores, la velocidad de ejecución se realiza dentro del FPGA a una frecuencia de 40MHZ, lo que significa que la adquisición de las señales se realiza cada 25ns.

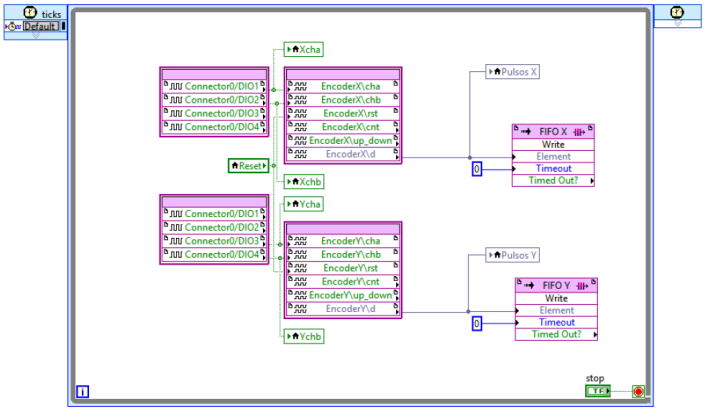

Figura 2. Sección del programa de decodificación de pulsos del codificador.

Estimación de velocidad

Una vez adquirida la cuenta de pulsos del codificador, se utilizó el llamado método N de estimación de velocidad, el cual consiste en comparar el valor de la posición actual y el valor de posición de m muestreos anteriores, la Ecuación 1 muestra la fórmula para estimación de la velocidad, dicha ecuación involucra al total de ranuras de pulsos del codificador lineal eP, a la posición, al valor de muestreo actual k, al valor de muestro m, y al tiempo de muestreo Tp. El tiempo de muestreo Tp es mucho más corto que el tiempo conocido Ts. El resultado de multiplicar $(Tp)(m)$ equivale al tiempo de muestreo Ts. Después se calcula el factor de conversión a radianes y se obtiene la velocidad en unidades de rad/s.

$$\hat{\theta} = \frac{2\pi(2^n)(\theta(k)-(\theta(k-m)}{(E_r)(eP)(m)} \quad (1)$$

En la figura 3 se muestra el algoritmo creado en LabVIEW encargado de obtener la diferencia entre el muestreo de pulsos actual y el anterior, dicho algoritmo se ejecuta dentro del FPGA al igual que el algoritmo de adquisición de pulsos del codificador de cuadratura, pero a una frecuencia de reloj distinta, por lo que fue necesario utilizar un método para la transmisión de datos entre los diferentes bucles control que corren de manera paralela dentro del FPGA.

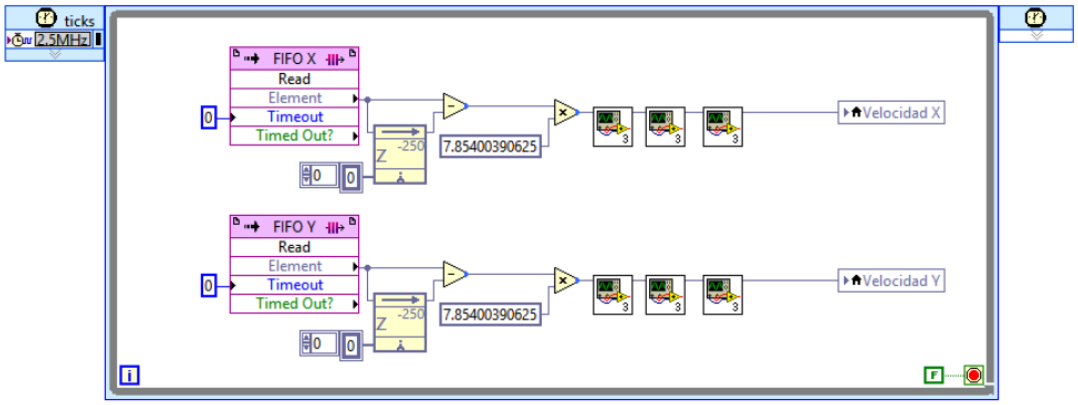

Figura 3. Sección de programa de estimación de velocidad.

El método seleccionado para realizar dicha tarea fue el *first input - first output* (FIFO, por sus siglas en inglés), el cual consiste en crear un buffer capaz de contener todos los datos generados en el bucle de control corriendo a velocidad más alta y leerlos en bucle de control la menor velocidad, un aspecto importante a considerar es la definición del tamaño del buffer a crear, ya que este debe ser suficientemente grande para impedir su desbordamiento y por consecuencia pérdida de datos, pero a la vez no ocupar espacio innecesario, optimizando el uso de los recursos de memoria.

Implementación de filtro pasa bajas

Con el fin de aislar los ruidos que pudiesen presentarse en el sistema, en distintas partes del programa se implementó un filtro digital de primer orden *Infinite impulse response* (IIR, por sus siglas en inglés) cuya función de transferencia se muestra en la Ecuación 2.

$$y = a(I + I^{z-1}) + b^{z-1} \qquad (2)$$

La salida de los filtros IIR depende de las entradas actuales y pasadas, y además de las salidas en instantes anteriores. Esto se consigue mediante el uso de retroalimentación de la salida, el filtro se encuentra en el dominio discreto y el valor de z^{-1} corresponde al muestreo obtenido en la ejecución anterior del bucle de control del programa. En la Figura 4 se observa el algoritmo implementado en LabVIEW para el filtro digital, donde solo resta calcular el valor de los coeficientes a y b, para lo cual se realizó un script en MATLAB el cual crea una función de transferencia para el filtro a partir de los valores de frecuencia de corte $gdis$ y tiempo de ejecución del programa ts.

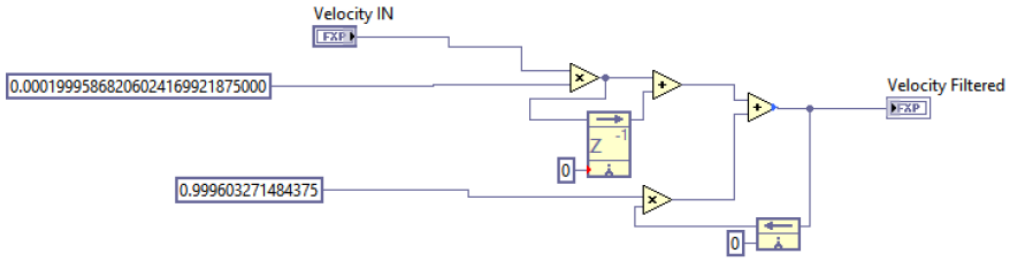

Figura 4. Sección de programa de implementación de filtro

Debido a que el programa corre a frecuencias de reloj diferentes, un filtro fue calculado para las operaciones realizadas en el FPGA y otro para las ejecutadas en el controlador de tiempo real RT, las cuales se ejecutan a frecuencias de 2.5MHZ y 1MHZ respectivamente.

Adquisición determinística de datos

La comunicación y transferencia de datos es uno de los factores más importantes a considerar en el diseño de un sistema embebido, LabVIEW ofrece la opción de seleccionar entre varios mecanismos diferentes para la transferencia de datos entre los procesos en un solo objetivo, así como los procesos que se comunican a través de objetivos, debido a que la arquitectura propuesta para el control del sistema trabaja a distintas frecuencias de reloj y en diferentes hardwares fue necesaria la utilización de distintos métodos para comunicar y transmitir información. En la figura 5 se aprecia el segmento del programa encargado de escribir los datos de prueba, consiste en dos bucles de control corriendo a diferentes velocidades dentro del *RTOS* (sistema operativo de tiempo real, por sus siglas en ingles). El primer bucle corre a una velocidad de $100\mu s$, durante 5 segundos, y en él se escriben los valores provenientes del monitoreo del sistema y se crea un arreglo con las 6 variables que se decidieron almacenar (velocidad, posición y torque para cada motor) obteniendo un total de 50,000 datos por cada variable. Posteriormente, el

arreglo es leído por el siguiente bucle a una velocidad no determinística ya que ésta depende de la disponibilidad de los procesadores del controlador y por último el conjunto de datos es escrito a un archivo dentro del disco duro del RTOS.

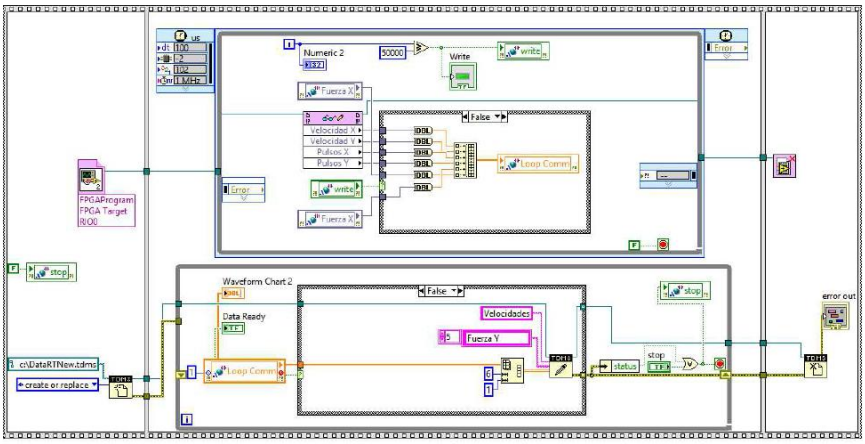

Figura 5. Sección de programa de adquisición determinística de datos.

Controlador de velocidad basado en observador de perturbaciones

Una vez que se cuenta con el valor de la velocidad estimada $\hat{\theta}$ de los servomotores y teniendo la certeza que los datos obtenidos se encuentran libres de ruido, se procedió a implementar un control de velocidad basado en un observador de perturbaciones (DOB, por sus siglas en inglés). La lógica de operación del DOB hace una comparación entre la entrada nominal de la planta o sistema a controlar y su salida nominal. Los valores reales recaen explícitamente en los valores tangibles del motor. Por otro lado, los valores nominales se pueden obtener de los manuales del fabricante del motor. Para este caso particular la constante de par Ktn y el momento de inercia Jn se obtuvieron de la hoja de especificaciones del motor. A la salida del DOB se obtiene el par compensado τ, el cual es una estimación bastante cercana a la perturbación real del sistema. En la figura 6 se ejemplifica el diagrama del control de velocidad basado en el DOB.

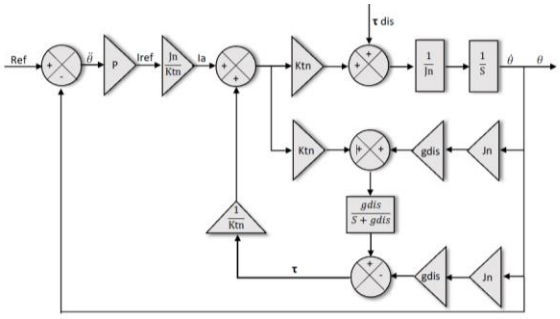

Figura 6. Diagrama de controlador de velocidad, basado en un DOB.

En la figura 7 se muestra la sección del programa de LabVIEW donde fue implementado el control de velocidad basado en un observador de perturbaciones, para realizar las pruebas se incluyó también en el código una sección encargada de generar señales cuadradas y triangulares, y de esta manera evaluar si el sistema tiene la capacidad de seguir la referencia de velocidad que se le está aplicando.

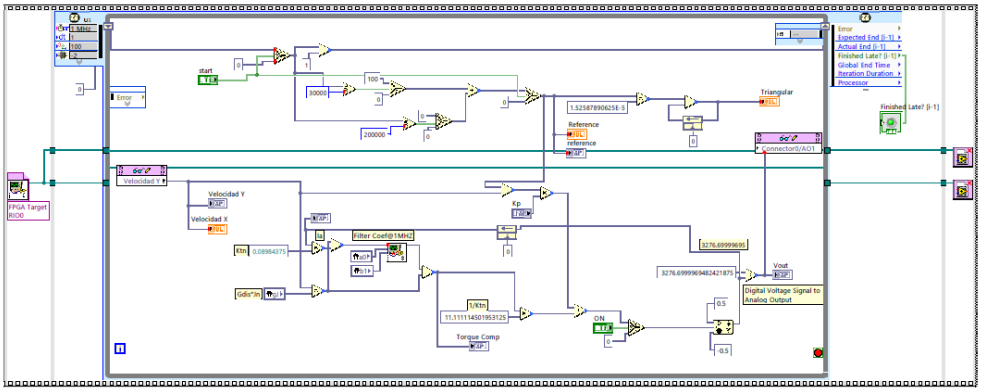

Figura 7. Sección de programa de controlador de velocidad.

Verificación del controlador de velocidad

La prueba realizada al sistema consistió en la aplicación de una señal cuadrada yendo de $100\,rad/s$ a $-100\,rad/s$ y obteniendo el comportamiento descrito en la Figura 8 donde se aprecia la señal de referencia y la velocidad real adquirida por el sistema, y además se muestra el detalle del rápido tiempo de respuesta del controlador correspondiente a $5\,ms$.

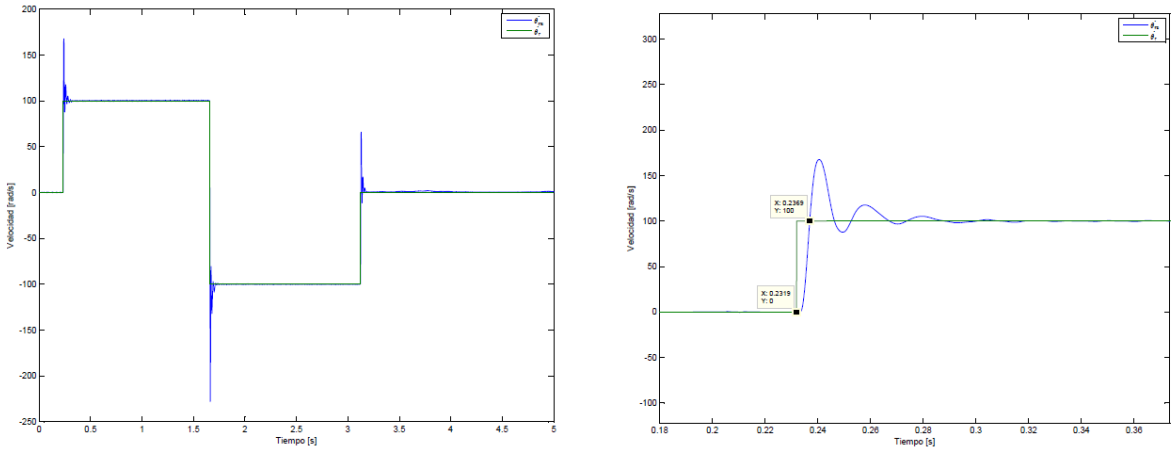

Figura 8. Comportamiento y tiempo de respuesta del controlador de velocidad.

Identificación del momento de inercia del sistema

Como se mencionó en la sección 2.4 los valores obtenidos de la hoja de especificaciones de los motores fueron

utilizados para realizar las pruebas de control de velocidad, sin embargo, al momento de incorporar más elementos de hardware al sistema como lo son los actuadores lineales implica que el valor del momento de inercia cambie significativamente, por lo que es necesario obtener el valor real de dicho parámetro para posteriormente incorporarlo al modelo de control bilateral. Para su obtención se utilizó la Ecuación 3, donde teniendo una aceleración constante y conocer el valor de par obtenido por el DOB, se puede obtener el valor real del momento de inercia Jn.

$$\mathrm{Jn} = \frac{\tau}{\ddot{\theta}} \tag{3}$$

Se realizaron distintas pruebas para validar el método propuesto, consistentes en colocar distintas cargas al motor con diferentes geometrías y aplicar una señal de referencia de velocidad incremental al motor, para obtener una aceleración constante, en la sección 2.5.1 se muestra una de dichas pruebas.

Verificación del método de obtención del momento de inercia

En la Figura 9 se muestra el sistema acoplado al servomotor para realizar los cálculos de inercia, se aplicó la misma señal de rampa de la prueba anterior y se tomaron los datos de respuesta del motor para velocidad y el par calculado por el DOB. En la figura 10 se observa el comportamiento de la velocidad del motor al realizar la prueba y se señala con un recuadro la parte de la pendiente de velocidad que fue utilizada para el cálculo del momento de inercia, dicho detalle se muestra más claramente y con él obtenemos la ecuación de la recta pendiente y por consecuentemente el valor de la aceleración.

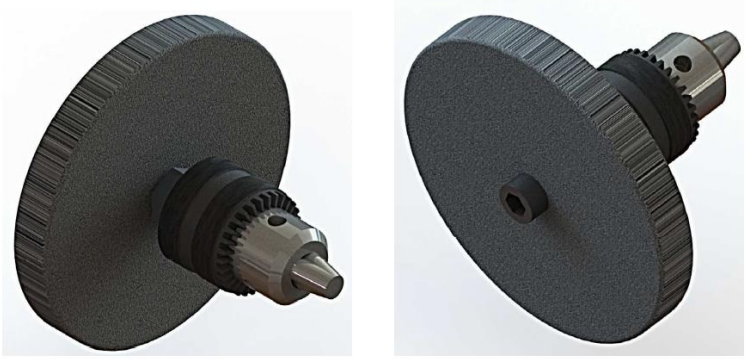

Figura 9. Carga acoplada al motor para realización de prueba.

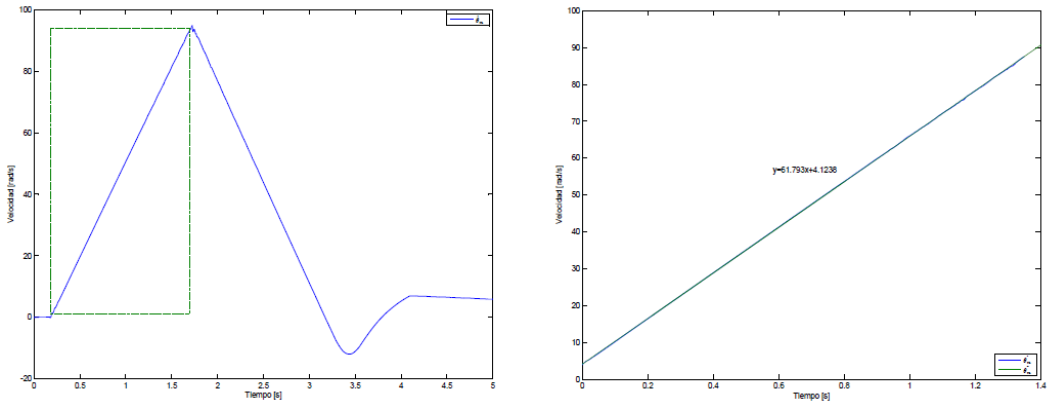

Figura 10 .Gráficas utilizadas para cálculo de aceleración.

De la misma manera fueron obtenidas las gráficas correspondientes al par, mostradas en la Figura 11, tomando el valor de par τ calculado.

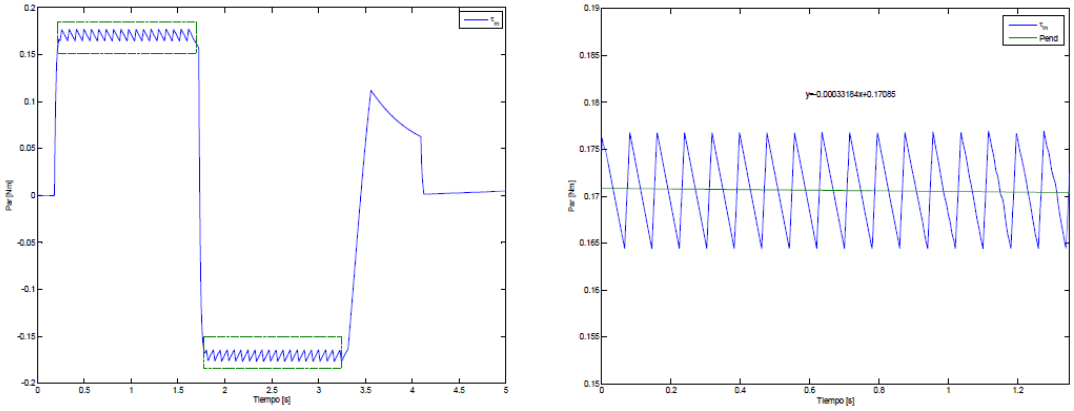

Figura 11. Gráficas utilizadas para cálculo de par de compensación.

Al tener los valores de aceleración y par del sistema se utilizó la Ecuación 3 para calcular el momento de inercia Jn del sistema, obteniendo lo siguiente (Ecuación 4):

$$\text{Jn} = \frac{0.17085\ Nm}{61.793\ rad/s^2} = 0.0027648\ \text{Kg m}^2 \quad (4)$$

Utilizando el software Solid Works, se dibujaron los elementos montados al motor, especificando los materiales de los que están conformados y su geometría, ya que de estos valores depende el momento de inercia de cualquier sistema, los cálculos realizados por el software de SolidWorks arrojaron resultados prácticamente iguales a los obtenidos.

Con los resultados obtenidos se observó que los valores del momento de inercia reales son bastante similares a los simulados, con lo que se concluyó que el método funciona correctamente y se procedió a utilizarlo para obtener el valor de inercia del sistema que incluye los actuadores lineales y será utilizado para el

control bilateral posteriormente. La Figura 12 corresponde a la pendiente de velocidad y al torque de compensación resultante de la prueba realizada al sistema con los actuadores lineales montados a los motores.

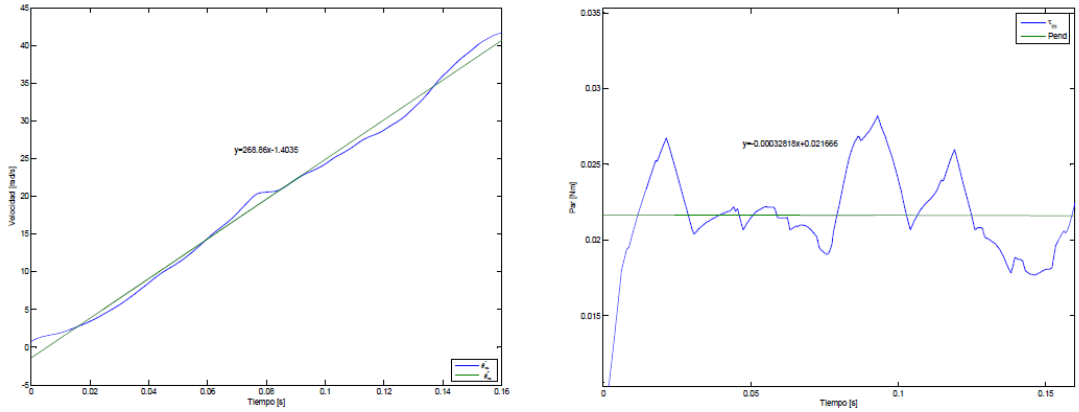

Figura 12. Gráficas utilizadas para cálculo de aceleración.

Al igual que en las pruebas anteriores se utilizaron los valores de aceleración y par del sistema y con la Ecuación 6 fue calculado el momento de inercia Jn del sistema, obteniendo el valor expuesto en la Ecuación 5.

$$\text{Jn} = \frac{0.021666\ Nm}{268.86\ rad/s^2} = 0.0027648\ \text{Kg m}^2 \quad (5)$$

Control robusto de aceleración

Con el algoritmo del DOB generado, se prosiguió a su implementación dentro del control de aceleración robusto. La figura 13 nos muestra un control de aceleración con el DOB incluido en el algoritmo como un sub programa. Como salida de esta parte del código se obtiene el torque de compensación calculado por el DOB y el voltaje a aplicar a los motores.

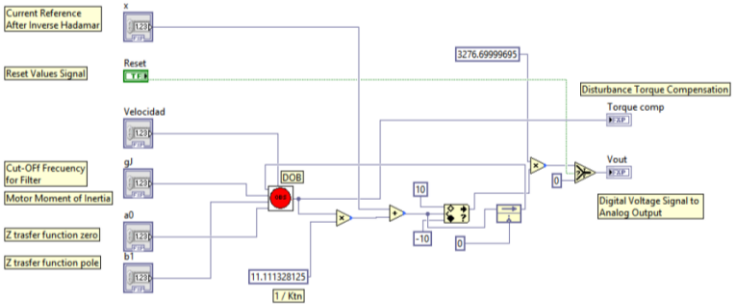

Figura 13. Sección de programa de control robusto de aceleración.

Control bilateral

Para lograr el control bilateral es necesario haber implementado el control robusto de

aceleración en cada uno de los motores que componen el sistema y después utilizar la matriz de Hadamard (H) y la matriz inversa de Hadamard (H^{-1}). El control de aceleración robusto previamente obtenido, nos entrega tres grupos de señales: posición, velocidad y par. Dado que en un control bilateral se necesitan dos sistemas como interfaces, un maestro y un esclavo, entonces cada sistema maneja su propio control de aceleración robusto. Por lo que se tienen un total de 6 grupos de señales: $\tau_m, \tau_e, \theta_e, \theta_m, \dot{\theta}_e, \dot{\theta}_m$ Para poder continuar con la consigna de ejecutar el control bilateral, se necesita obtener el error entre el maestro y el esclavo y luego compensar con una ganancia para ir reduciendo ese error. Para el caso de esta aplicación existen tres tipos de error: de fuerza, de posición, y de velocidad. Para obtener el valor de estos errores, se utiliza la matriz de Hadamard. (Ecuación 6)

$$\begin{bmatrix}\tau_c\\ \tau_d\end{bmatrix} = \begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} = \begin{bmatrix}\tau_m\\ \tau_s\end{bmatrix} = H\begin{bmatrix}\tau_m\\ \tau_s\end{bmatrix}$$
$$\begin{bmatrix}\theta_c\\ \theta_d\end{bmatrix} = \begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} = \begin{bmatrix}\theta_m\\ \theta_s\end{bmatrix} = H\begin{bmatrix}\theta_m\\ \theta_s\end{bmatrix} \quad (6)$$

Para el control bilateral se utiliza el modo común de par de torsión τ_c y el modo diferencial de posición θ_d para satisfacer esas ecuaciones. Es en base a esta matriz de Hadamard, se obtienen los valores de error en los sistemas maestro y esclavo. Después de la matriz de Hadamard, se trata de lograr que los modos común y diferencial converjan a cero. Por este motivo los factores proporcionales respectivos para fuerza KpF, posición KpP, y velocidad KpV son utilizados como forma de rastrear el error a cero. Finalmente, después de ajustar dicho error, se utiliza la matriz inversa de Hadamard (H^{-1}) para convertir nuevamente los valores al dominio de la aceleración y puedan ser introducidos al bucle de control de aceleración.

Para la implementación del control bilateral en LabVIEW se utilizaron distintos subprogramas creados anteriormente cómo el control robusto de aceleración, DOB y se agregaron los subprogramas correspondientes a las matrices de Hadamard normal e inversa, teniendo el diagrama completo mostrado en la Figura 14, donde se corre dentro del RTOS a una velocidad de 1MHz.

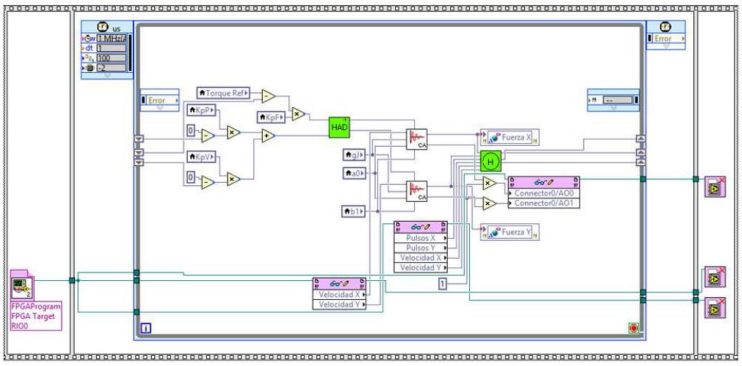

Figura 14. Sección de programa de control bilateral.

Resultados

Una vez implementado el programa de control bilateral, se realizaron dos pruebas diferentes para observar la respuesta del sistema; la primera llamada movimiento libre y la segunda movimiento con contacto.

Los valores de los parámetros utilizados en la realización de las pruebas fueron obtenidos de distintas fuentes, para así lograr un mejor desempeño. Las constantes de los motores se tomaron de la hoja de especificaciones del fabricante de los mismos, los valores de las constantes de posición, velocidad y fuerza se obtuvieron mediante la experimentación con el sistema y el momento de inercia se calculó anteriormente.

La primer prueba consistió en mover las masas de los actuadores lineales maestro y esclavo evitando contacto alguno entre alguno de los dispositivos con algún objeto. El segundo tipo de pruebas consistió en mover las masas de los actuadores lineales con el fin de que el dispositivo esclavo hiciera contacto con algún objeto, en este caso un bloque rígido de aluminio. Las gráficas resultantes de las pruebas para movimiento libre se observan en la Figura 15 y en la Figura 16, las correspondientes a la prueba con contacto.

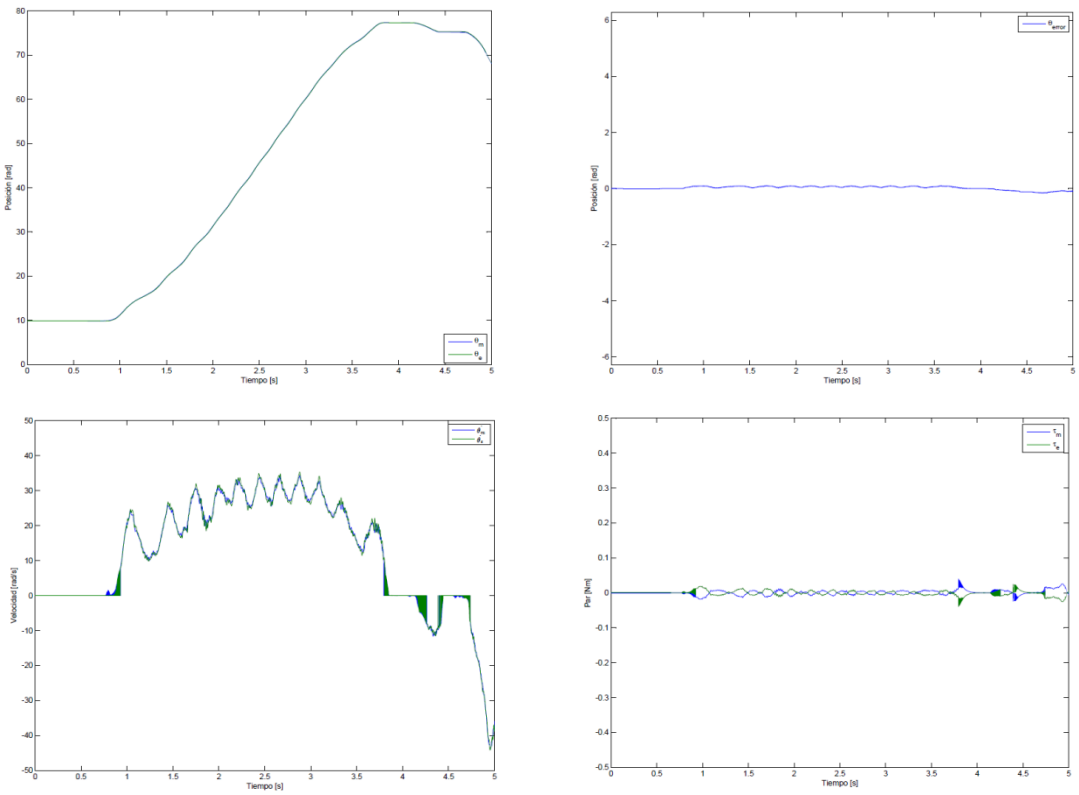

Figura 15: Resultados de prueba de movimiento libre

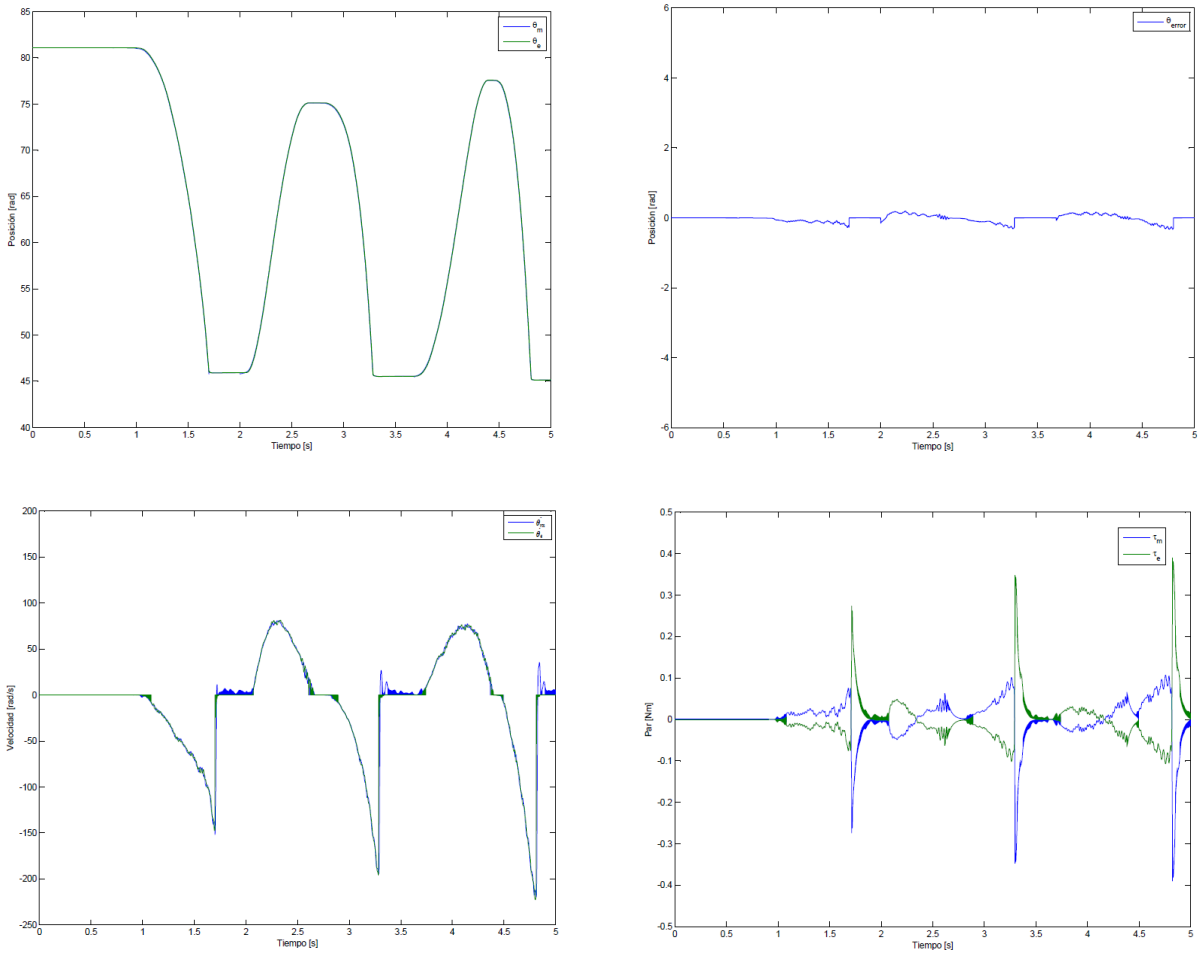

Figura 16. Resultados de prueba de movimiento libre

Conclusiones

Al finalizar el presente proyecto se puede concluir que el algoritmo de control bilateral propuesto tiene la capacidad de recrear sensaciones táctiles al realizar contacto con diferentes objetos.

El método N de estimación de velocidad permitió transmitir sensaciones táctiles con alta transparencia entre el sistema maestro - esclavo, y fue clave para el correcto funcionamiento del observador de perturbaciones y a su vez el control bilateral.

LabVIEW es una herramienta muy útil para el desarrollo de algoritmos de control, y que su versatilidad permite la integración de código alterno a su lenguaje nativo y permitió la programación de un FPGA mediante su sencillo y comprensible lenguaje de bloques.

Referencias

Ji J. y Sul, S. (1995). Kalman Filter and LQ Based Speed Conroller for Torsional Vobration Suppresion,» IEEE Transactions on Industrial Electronics, pp. 564-571.

Nandayapa, M. Mitsantizuk C. y Ohishi, K. (2011). High Performance Velocity Estimation for Controllers with Short Prossensing Time by FPGA, IEEE Journal of Industry Applications, pp. 55-61.

Cui, S. J. Tosunoglu, Roberts, R. Moore C. y Repperger, D. W. (2003). A Review Of Teleoperation System Control, Florida Conference on Recent Advances in Robotics, pp. 1-12.

Kasahara, Y. Kawana, H. Usuda S. y Onishi, K. (2012). Telerobotic-assisted Bone Drilling System Using Bilateral Control with Feed Operation Scaling and Cutting Force Scaling, The Internation Journal of Medical Robotics and Computer Assisted Surgery, pp. 211-229.

Okamura, A. M. (2009). Haptic Feedback in Robot-Assisted Minimally Invasive Surgery, Curr Opin Urol. Author manuscript,, pp. 102-107.

Onal, C. D. Pawashe C. y Sitti, M. (2007). A Scaled Bilateral Control System for Experimental 1-D Teleoperated Nanomanipulation Applications, EEE/RSJ International Conference on Intelligent Robots and Systems, pp. 483-488.

Daunay B. y Régnier, S. (2009). Stable Six Degrees of Freedom Haptic Feedback for Flexible Ligand-protein Docking, Computer-Aided Design, pp. 886-996.

Kosugi T. y Katsura, S. (2012). An Approach to Controller Design of Bilateral Control with Dimensional Scaling, The 12th IEEE International Workshop on Advanced Motion Control, pp. 1-6.

Susa, S. Shimono, T. Takei, T. Atsuta, K. Shimojima, N. Ozawa, S. Morikawa Y. y Ohnishi, K. (2008). Transimission of Force Sensation by Micro-Macro Bilateral Control with Scaling of Control Gains, 0th IEEE International Workshop on Advanced Motion Control, pp. 532-537.

Yamanouchi, W. Yajima S. y Katsura, S. (2012). A Novel Control Index of Bilateral Control for Master-Slave System with Diferent Motion Areas, IEEE International Symposium on Industrial Electronics (ISIE), pp. 1656-1661.