Instituto Tecnológico de Costa Rica

Carrera de Ingeniería Mecatrónica

Diseño de prototipo del myRIO demobox para la demostración de las cualidades y características del dispositivo desarrollado por National Instruments.

Informe de Proyecto de Graduación para optar por el título de Ingeniero en Mecatrónica con el grado académico de Licenciatura

Ignacio Alejandro Sandí Fonseca

Cartago, Noviembre de 2014

Índice General

Índice de figuras.

Resumen:

National Instruments se ha caracterizado por ser una empresa que provee a ingenieros y científicos con las herramientas necesarias tanto de software como de hardware para poder impulsar de una manera más rápida y eficiente el desarrollo de nuevas tecnologías.

Gracias a esta visión es que se ha desarrollado el dispositivo NI myRIO, el cual está orientado a que estudiantes de los últimos cursos de ingeniería lo utilicen para desarrollar sus proyectos de grado o bien proyectos finales en sus cursos. Esto para facilitarles la implementación de la solución y que así puedan crear aplicaciones con niveles de complejidad mayores.

Mostrar las múltiples capacidades de este dispositivo en ocasiones se vuelve complicado por el amplio rango de campos en el cual se puede desenvolver el myRIO, y es por esa razón que surge la necesidad de crear un dispositivo o aplicación totalmente portátil capaz de mostrar muchas de dichas funcionalidades utilizando la mínima cantidad de hardware posible. Con esto nace la concepción del myRIO demobox.

El mismo reúne un conjunto de aplicaciones que se pueden poner en funcionamiento utilizando al myRIO de manera que se puede mostrar su aplicabilidad en campos tan variados como lo son los sistemas de visión, control automático, sistemas embebidos, mecatrónica, entre otros.

En este documento se detalla el diseño del sistema del demobox, desde su concepción, demos posibles a implementar y el diseño final del primer prototipo.

Abstract:

National Instruments has been known for being a company that equips scientists and engineers with the necessary software and hardware tools to help improve and advance the development of new technologies in a faster and more efficient way.

Thanks to this approach is that the NI myRIO has been developed, a device which is geared to students in their final year of engineering to use it in their final class projects or even in their graduation projects,all of this with the aim of facilitating students to develop applications of greater complexity.

Because of the wide variety of fields in which the myRIO can be used it becomes difficult to show its capacities, that is why the necessity of creating a new device or application able to fully show many of these capabilities in a portable way arose. This new application or device had to use the less amount of hardware possible to work, and that is how the myRIO demobox was conceived.

The demobox gathers together many of the applications that can be developed using the NI myRIO and shows its applicability in fields such as machine vision, automatic control, embedded systems, mechatronics, etc.

This paper details the design of the myRIO demobox since it conception, possible apps to implement and the final design of the first prototype.

Introducción:

Actualmente en el mercado latinoamericano la plataforma NI myRIO es ampliamente demandada, al ser un producto competitivo en cuanto a costo, competencia y accesibilidad para el mercado local. Por esta razón se busca contar con un recurso que permita compartir, dar acceso e impactar a los prospectos, dando visibilidad de los alcances del producto, maximizando los entornos y aplicaciones en los cuales se puede aprovechar el producto.

Problema:

Como se mencionó anteriormente, a la hora de mostrar el producto se tienen problemas tanto de logística como de portabilidad con el mismo debido a la gran cantidad de conexiones posibles que se le pueden realizar al myRIO. Además, este posee sensores que no están diseñados para "heavy manipulation" lo que quiere decir que los mismos son propensos a dañarse si se manipulan sin cuidado o bien son sometidos a realizar acciones para las cuales no fueron diseñados.

De igual manera, debido a su complejidad y extensas capacidades el myRIO es un producto difícil de mostrar si no se posee una plataforma predispuesta para dicha acción, lo que implica que se deben estar creando aplicaciones dependiendo del entorno en el que se vaya a mostrar que generalmente no son capaces de demostrar la máxima capacidad del dispositivo, cuestión que dificulta a los posibles compradores observar y entender los beneficios que les puede brindar la compra de un myRIO.

Además, la cualidad del myRIO de ser controlado por medio de la aplicación desarrollada en IOS no se está utilizando a cabalidad y esta es una herramienta que simplificará mucho el proceso de muestreo y control.

Enfoque de la solución:

Para solucionar el problema planteado anteriormente se plantea hacer un diseño completo de un nuevo dispositivo en el cual se puedan integrar todas las funcionalidades tanto del myRIO como de los módulos existentes actualmente.

Para lograr esto primeramente se debe determinar el modo de funcionamiento, capacidades y similitudes de cada uno de los módulos a ser utilizados para la elaboración del myRIO demobox como lo son el kit de mecatrónica o el kit de embedded. Esto con el objetivo de determinar de una manera íntegra qué módulos pueden ser demostrados utilizando la misma pieza de hardware y así optimizar tanto el espacio como el costo del mismo.

Una vez determinados dichos factores se procederá a investigar y diseñar posibles aplicaciones que sirvan para mostrar las cualidades de varios módulos a la vez y que estas sean enfocadas a la academia debido a que los principales compradores del myRIO son universidades y colegios donde se imparten distintas ingenierías.

Ahora bien, una de las adversidades principales con el desarrollo de dichas aplicaciones es su nivel de complejidad, dado que, se podría caer en el error de diseñar aplicaciones de bajo nivel solo con el objetivo de cumplir el aspecto de mostrar las cualidades del myRIO, sin embargo esto no cumpliría el objetivo de convencer a posibles compradores de elegir el myRIO por sobre otras plataformas disponibles en el mercado.

Por esta razón se necesita un límite de complejidad de sistema para poder ser considerado aceptable, un ejemplo de una posible solución con un nivel de complejidad adecuado es; en el caso de aplicaciones de control automático la simulación de un sistema que mantiene una distancia específica respecto a un objeto. Esto implicaría implementar un control ya sea por medio de PID u otra aplicación para que el sistema responda según su distancia respecto a un objeto, por ejemplo: mover un motor que simule una rueda alejándose o acercándose a dicho objeto.

Dicha ejemplo aparte de tener un nivel de complejidad lo suficientemente alto como para ser utilizado en laboratorios de universidades, implica la aplicación de conocimientos en el área de control, de mecánica al diseñar la planta, control de motores y programación, con lo que utilizaría algunas de las ramas de la mecatrónica para poder ser desarrollado.

Junto a esto se deben tomar en consideración cuestiones como el mantenimiento del dispositivo y la protección que le va a brindar el encapsulado al mismo. Estas consideraciones son parte del diseño concurrente de sistemas por lo cual son de gran importancia a la hora de desarrollar este nuevo producto.

Entre las consideraciones a tomar en cuenta a la hora de realizar el diseño del sistema se tiene que el mismo debe estar dentro de una caja portátil que soporte golpes, que contenga un myRIO y diferentes tipos de sensores y actuadores con los que se pueda interactuar, debidamente sujetos a la unidad sin contener partes móviles que se puedan soltar o dañar. El sistema debe quedar perfectamente documentado para su escalabilidad y duplicación.

Entorno y Objetivos:

Entorno del Proyecto:

El proyecto se desarrolla en la empresa National Instruments ubicada en Lagunilla de Heredia.

Debido a la complejidad y diversidad de los módulos con los que puede trabajar el equipo myRIO, además de la versatilidad de programación que este presenta se vuelve complicado mostrar todas sus cualidades de una forma fácil ya sea en eventos universitarios o bien en laboratorios empresariales.

Generalmente el personal encargado de mostrar el dispositivo debe improvisar aplicaciones en campo para así demostrar sus cualidades, sin embargo esto comúnmente conlleva a que dichas aplicaciones no sean lo suficientemente complejas como para justificar el uso de un dispositivo de alto nivel como lo es el myRIO.

Esto implica ciertos inconvenientes a la hora de mostrar el dispositivo ya que se deben estar generando soluciones distintas para cada una de las aplicaciones a mostrar, situación que a largo plazo no es conveniente por la ineficiencia en la administración de tiempo y recursos.

Además al ser un producto con un alto nivel de demanda a nivel regional se vuelve importante encontrar una manera de mostrar sus funcionalidades de primera mano en una forma que sea rápida, eficaz y logísticamente simple para así optimizar el recurso humano utilizado en las demostraciones del producto.

Debido a lo mencionado anteriormente, este proyecto está enfocado en lograr una solución que integre tanto el dispositivo myRIO como los distintos módulos que se pueden unir a él, de una forma que su puesta en marcha sea simple y rápida. Por lo tanto se diseñará un nuevo producto que permita demostrar las principales cualidades del myRIO de una forma que justifique usar este dispositivo.

Para hacer esto se tomará como base un proyecto ya existente, el compactRIO demobox, el cual puede mostrar usando unas cuantas herramientas de hardware las cualidades de los distintos módulos del compactRIO. Por ejemplo, el mismo sistema de motor el cual cuenta con varios sensores así como con una cámara, es utilizado para la demostración de mediciones de aceleración y velocidad; al igual dicho sistemas es utilizado para la demostración de sistemas de visión, posicionamiento de precisión entre otros. Esto demuestra cómo se puede utilizar un mismo sistema para demostrar diferentes módulos no relacionados entre sí.

Objetivo General:

Diseñar una plataforma que permita mostrar el myRIO interactuando con elementos externos, tales como sensores, DIO, AIO que además sea visualmente atractivo para clientes y usuarios en general.
(Indicador: Modelado al 100% del dispositivo donde se aprecian sus características de conexión e interfaces con el usuario.)

Objetivos Específicos:

1. Investigar sobre las capacidades técnicas y funcionales además de las aplicaciones de mayor uso del dispositivo myRIO y sus módulos para así modelar el diseño a las funciones y necesidades más críticas de los usuarios.
   (Entregable: Informe técnico de las capacidades de hardware del myRIO)
2. Diseñar un sistema capaz de integrar el myRIO con distintos módulos desarrollados por National Instruments como el kit de mecatrónica o el kit embedded, de manera que el sistema sea portátil y pueda realizar todas las funciones del myRIO sin necesidad de conexiones complejas con otros equipos.
   (Indicador: planos y modelo CAD donde se aprecie el sistema en un porcentaje mínimo del 90% de componentes necesarios para su construcción y funcionamiento)
3. Diseñar un equipo que permita realizar simulaciones en los ambientes de la mecatrónica, sistemas embebidos y control automático.
   (Indicador: Software de control al 85% para las aplicaciones de mecatrónica, control automático y embedded)

4. Desarrollar un Manual de construcción del equipo donde se detallen los materiales y componentes necesarios para la producción en masa del sistema completo.
   (Indicador: Manual al 90% que brinde la información necesaria de planos mecánicos y eléctricos para la construcción del prototipo completo)

Marco Teórico

Ingeniería Mecatrónica:

"El campo de la mecatrónica concierne la aplicación sinérgica de la ingeniería mecánica, electrónica, de control y de la computación para el desarrollo de productos y sistemas electromecánicos a través de un enfoque de diseño integral." (De Silva).

Un sistema mecatrónico requiere un enfoque multidisciplinario para su diseño, desarrollo e implementación. En el desarrollo tradicional de un sistema electromecánico los componentes mecánicos y eléctricos son diseñados o seleccionados de manera separada y luego integrados, posiblemente con otros componentes de hardware y software. En contraste, con un enfoque mecatrónico el sistema electromecánico completo es tratado concurrentemente de una manera integrada por un equipo multidisciplinario de profesionales de ingeniería y otras áreas. Esto incrementa la eficiencia del sistema ya que hay una mayor compatibilidad entre las funciones de los componentes debido al enfoque unificado e integrado que se da durante el diseño y desarrollo del sistema. A su vez un sistema mecatrónico generalmente será más eficiente tanto en funcionalidad como en precio que un sistema no mecatrónico al cual se le dedica un esfuerzo similar durante su desarrollo.

Sistemas Mecatrónicos:

Un sistema mecatrónico típico consiste de un armazón metálico, sensores y actuadores, controladores, dispositivos de procesamiento de señales, unidades de procesamiento tanto en forma de hardware como de software, interfaces y fuentes de poder.

Distintos tipos de mediciones, adquisición y transferencia de información están involucradas entre la diversidad de componentes que integran un sistema mecatrónico.
Ahora bien, los sistemas mecatrónicos varían tanto en tamaño como en complejidad, desde un pequeño servomotor, hasta un robot de alta tecnología como lo es Asimo de la compañía Honda.

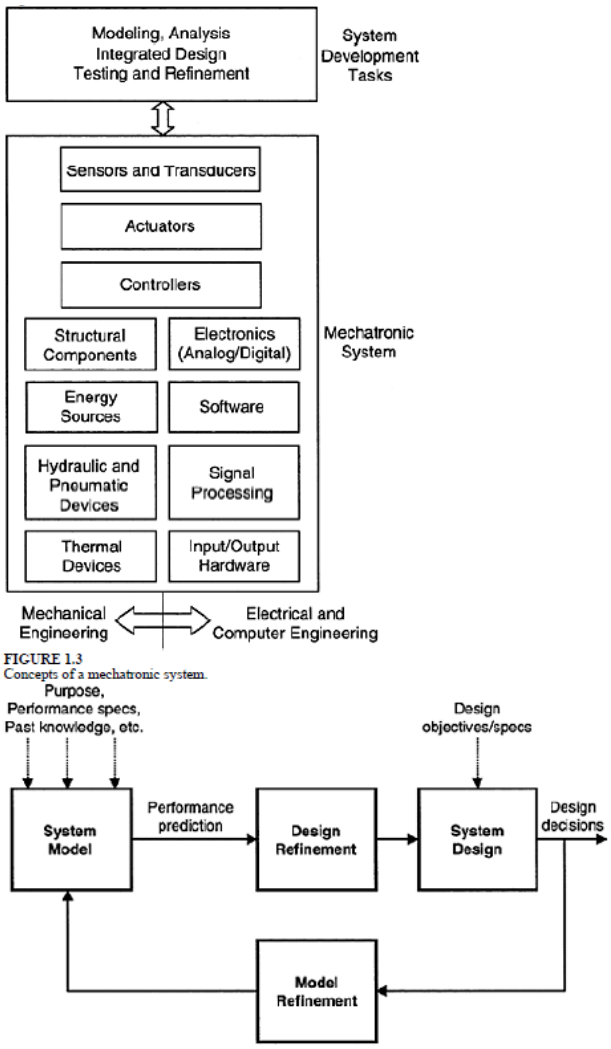


Figura 1. *Conceptos de un sistema mecatrónico.*[2]

Modelado y Diseño:

El diseño en ingeniería Mecatrónica toma en cuenta uno de los problemas más comunes en este ámbito: la optimización de modelos según su coste y rendimiento.

"El modelado y diseño pueden ir de la mano en una forma iterativa, al principio, al saber un poco de información del sistema como por ejemplo: funciones deseadas, especificaciones de rendimiento o bien conocimiento general de sistemas similares, y combinar esto con los objetivos de diseño es posible desarrollar un modelo lo suficientemente detallado y complejo del

sistema."(De Silva). Al analizar y llevar a cabo simulaciones del modelo por computador es posible generar información útil que guiará el proceso de diseño. De esta forma algunas decisiones de diseño pueden ser tomadas con mejor fundamento y el modelo puede ser redefinido o bien mejorado. Ésta relación iterativa entre modelado y diseño se muestra en la figura siguiente.

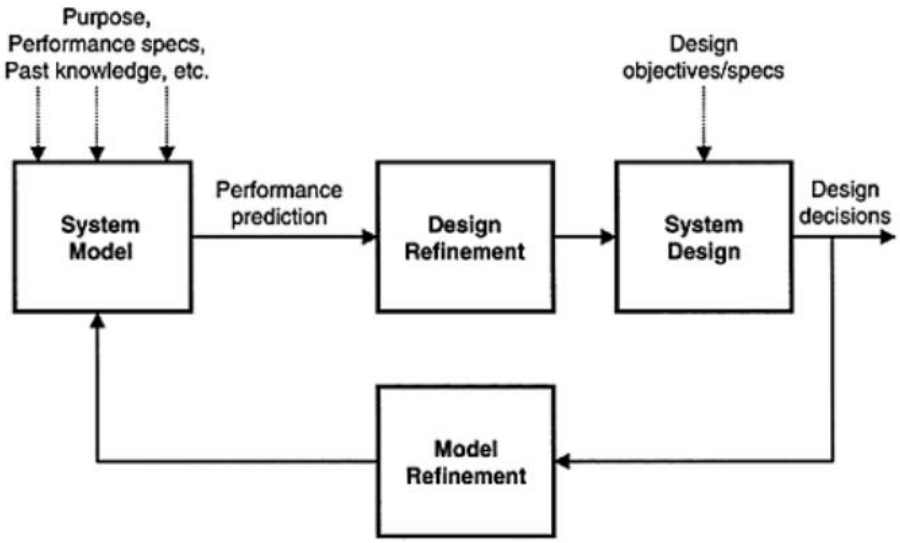


Figura 2. *Diseño de un sistema mecatrónico.*[2]

Conceptos de Diseño Mecatrónicos:

"Un sistema mecatrónico consiste de diferentes tipos de componentes y elementos interconectados. Como resultado de esto habrá conversiones de energía de una forma a otra, particularmente de energía eléctrica a energía mecánica. Esto permite utilizar a la energía como un concepto unificador en el análisis y diseño de un sistema mecatrónico." (De Silva)

En un sistema electromecánico existen interacciones entre dinámica eléctrica y dinámica mecánica. Específicamente la dinámica eléctrica afecta la dinámica mecánica y viceversa. Tradicionalmente se le da un enfoque secuencial al diseño de sistemas mixtos como los electromecánicos. Por ejemplo: la parte mecánica y estructural son diseñados primero, luego la parte eléctrica y electrónica y por último el sistema computacional es seleccionado e integrado al equipo y así sucesivamente. El acoplamiento dinámico entre varios componentes de un sistema dicta, sin embargo, que un diseño adecuado de un sistema debe considerar al sistema completo

como un todo en vez de diseñar los aspectos eléctricos/electrónicos y mecánicos por separado y de manera secuencial, ya que cuando distintos componentes diseñados independientemente son interconectados pueden surgir varios problemas:

- Sus características originales y sus condiciones de operación van a cambiar debido a interacciones de carga o de dinámica.
- No se puede asegurar un acople perfecto entre los componentes diseñados.
- Algunas de las variables externas de los componentes pueden convertirse en variables "ocultas" debido a las interconexiones, lo que puede resultar en problemas potenciales que no precisamente puedan ser monitoreados o controlados directamente.

Áreas de aplicación:

Las áreas de aplicación de la mecatrónica son numerosas, e involucran todas aquellas que conciernen sistemas mixtos y particularmente sistemas electromecánicos. Estas aplicaciones pueden incluir:

1. Modificaciones y mejoras a diseños convencionales al usar un enfoque mecatrónico.

2. Desarrollo e implementación de sistemas mecatrónicos originales e innovadores.

En cada categoría, las aplicaciones usarán mediciones, actuación, control, tratamiento de señales, interconexión e interface de componentes, y comunicaciones, generalmente usando herramientas mecánicas, eléctricas/electrónicas, computacionales y de ingeniería de control. Algunas áreas importantes se indican a continuación:

Transporte: En el transporte terrestre (automóviles, trenes, entre otros) se han desarrollado sistemas como las bolsas de aire, frenos ABS, sistemas de control crucero, suspensiones activas, etc.
En el transporte aéreo, sistemas como los simuladores de vuelo, control de navegación, control de vuelo e incluso los trenes de aterrizaje son resultado de la mecatrónica.

Manufactura: Robots de líneas de ensamblaje (para soldadura, pintura, ensamblaje, inspección, etc.) vehículos guiados automáticamente (AGVs), Computadores de control numérico (CNCs) entre otros.

Otras áreas de aplicación son la medicina, ambientes de oficina, hogares, industria de computadores, ingeniería civil, viajes espaciales y muchos más.

myRIO

Figura 3. *Placa NI myRIO*

El NI myRIO,ver figura 3, es un dispositivo embebido para diseño estudiantil, es ideal para hacer ingeniería real en un semestre. Dispone de un procesador programable ARM Cortex-A9 dual-core de 667 MHz y E/S personalizadas de FPGA Xilinx las cuales pueden ser usadas por estudiantes para empezar el desarrollo de sistemas y resolver problemas de diseño más rápido. El dispositivo NI myRIO permite interactuar con el Zynq-7010 Sistema en un Chip completamente programable (SoC) para liberar el poder del software de diseño de sistemas NI LabVIEW en aplicaciones de tiempo-real (real-time RT) y en nivel FPGA. De esta forma en lugar de invertir tiempo en la depuración de sintaxis de código o en desarrollar interfaces de usuario, los estudiantes pueden usar el paradigma de programación gráfica de LabVIEW para enfocarse en

construir sus sistemas y resolver sus problemas de diseño sin la presión agregada de una herramienta complicada.

NI myRIO es una herramienta de enseñanza reconfigurable y reusable que ayuda a los estudiantes a aprender una gran variedad de conceptos de ingeniería y al mismo tiempo completar sus proyectos de diseño. Las capacidades RT y FPGA junto con la memoria integrada y con WiFi incorporado permiten a los estudiantes desplegar aplicaciones remotas y ejecutarlas sin conexión a una computadora remota. Tres conectores (dos puertos de expansión [MXP] NI myRIO y un puerto NI miniSystems [MSP] que es idéntico al conector de NI myDAQ) envían y reciben las señales desde los sensores y circuito que los estudiantes necesitan en sus sistemas. Cuarenta líneas de E/S digitales en total con soporte de interfaz periférica de señal (SPI), modulación de ancho de pulsos (PWM), entrada de codificador de cuadratura, transmisor-receptor asíncrono universal (UART) y I^2C; ocho entradas analógicas de (composición única, independientes-single-ended); dos entradas diferenciales analógicas; cuatro salidas analógicas (composición única, independientes-single-ended); y dos salidas análogas con referencia a tierra que permiten conectividad a innumerables sensores, dispositivos y sistemas de control programáticos. Toda esta funcionalidad está construida y pre-configurada en la funcionalidad predeterminada FPGA lo que elimina la necesidad de expansión en las tablillas o "escudos" para agregar utilidad. Por último, estas características permiten a los estudiantes hacer ingeniería real de inmediato- desde vehículos controlados por radio hasta crear dispositivos médicos independientes.

El dispositivo es amigable (Interactivo) para los estudiantes y con capacidad completa (fuera de la caja-out of the box), el dispositivo NI myRIO es simple para instalarse, y los estudiantes pueden determinar fácilmente el estatus de operación. La personalidad FPGA completamente configurada de fábrica es desplegada en el dispositivo, por lo que los principiantes pueden empezar con un fundamento funcional sin tener que programar un FPGA para lograr el funcionamiento de sus sistemas. Sin embargo, el poder (funcionalidad) de las E/S (RIO) reconfigurables aparece cuando los estudiantes empiezan a definir la personalidad FPGA y moldear el funcionamiento del dispositivo para la aplicación. Con la escalabilidad del dispositivo, los estudiantes pueden usarlo desde clases de sistemas embebidos básicos hasta los

cursos de diseño del último año en áreas de control automático, sistemas mecatrónicos y robótica.

Figura 4. *NI myRIO.*[10]

LabVIEW:

Figura 5. *Pantalla inicial LabVIEW 2014.*

“LabVIEW posee aproximadamente 25 años en el mercado, en estos años se ha convertido en un estándar en el desarrollo de aplicaciones de test y medida, control de instrumentación y sistemas de adquisición de datos. Su versatilidad y potencia le ha hecho expandirse a otras áreas tales como visión artificial, PACs, control de movimiento, HMI y SCADAs para automatización industrial, análisis de ruido y vibraciones, gestión de información y generación de informes, etc. Hoy día LabVIEW ha incursionado en campos como Simulación, Diseño de Control, sistemas embebidos en tiempo real, algoritmos matemáticos avanzados, etc.” (Lajara & Pelegrí)

Durante este periodo de tiempo LabVIEW (Laboratory Virtual Instrument Engineering Workbench) ha tenido una gran expansión en la comunidad científica y académica, ya sea en universidades, colegios, centros de investigación entre otros debido a su versatilidad para la elaboración de prácticas de laboratorio, clases teóricas en las áreas de control, instrumentación, tratamiento digital de señales e incluso en proyectos de graduación.

LabVIEW comenzó su desarrollo en el Abril del año 1983, cuyo lanzamiento se daría en octubre de 1986 con el lanzamiento de LabVIEW 1.0 para Macintosh, luego en 1990 se liberó su versión 2.0. En la siguiente figura se pueden ver algunos de los hitos de LabVIEW a través de los años.

Fecha	Hito
Abril de 1983	Comienza el desarrollo de Labview
Octubre de 1986	Labview 1.0 para Macintosh
Enero de 1990	Labview 2.0
Septiembre de 1992	Labview para Windows
Octubre de 1992	Labview para Sun
Octubre 1993	Labview 3.0 multiplataforma
Abril de 1994	Labview para Windows NT
Octubre de 1994	Labview para Power Macintosh
Octubre de 1995	Labview para Windows 95
Mayo de 1997	Labview 4.0
Marzo de 1998	Labview 5.0
Febrero de 1999	Labview 5.1, LV para Linux y LV Real-Time
Agosto de 2000	Labview 6i
Enero de 2002	Labview 6.1
Mayo de 2003	Labview 7 Express, Labview PDA y FPGA
Mayo de 2004	Labview 7.1

Figura 6. *Versiones de LabVIEW hasta el 2004.*[4]

LabVIEW en sí mismo es el conjunto tanto del lenguaje gráfico de programación desarrollado por National Instruments así como de su entorno de desarrollo, el lenguaje de programación gráfica en LabVIEW se conoce como código G.

Una de las particularidades de LabVIEW es que el mismo está enfocado a la programación de instrumentación, es decir, a la recolección, análisis, monitoreo y visualización de mediciones de datos. Históricamente científicos, profesionales de ingeniería y otros han estado limitados a la compra o construcción de hardware instrumental para realizar mediciones de datos, LabVIEW presenta la opción de crear instrumentos de software personalizados y enfocados a las necesidades de la aplicación. Es por esta razón que los programas de LabVIEW son llamados VIs (Virtual Instruments).

Los VIs presentan la particularidad de constar de dos partes: el panel frontal y el diagrama de bloques. En el primero se aprecian todos los controles, indicadores y referencias utilizadas en el programa, mientras que en el diagrama de bloques se aprecia el código propiamente dicho.

Ahora bien, una de las características más fundamentales de LabVIEW es su capacidad de ejecución en paralelo; esto quiere decir que el código no se ejecuta de modo secuencial como sucede con los lenguajes de programación basados en texto. Sino que el mismo ejecuta todo el código al mismo tiempo siempre y cuando no dependa de alguna instancia de ejecución anterior.

Los programas de LabVIEW tienen las siguientes características:

- Naturaleza gráfica y compilada.
- Flujo de datos y/o programación basa en eventos.
- Capacidades multi-objetivo y plataforma.
- Flexibilidad orientada a objetos.
- Posibilidades de multithreading.

Aunque representado gráficamente con íconos y cables en vez de texto, el código G en el diagrama de bloques contiene los mismos conceptos de programación encontrados en la mayoría de lenguajes tradicionales. Por ejemplo, código G incluye tipos de datos, bucles, gestión de

eventos, variables y programación orientada a objetos. LabVIEW compila código G directamente a código de máquina para que el procesador pueda ejecutarlo.

Los programas de LabVIEW se ejecutan de acuerdo a las reglas de flujo de datos en lugar de la manera tradicional encontrada en la mayoría de lenguajes de programación basados en texto como C y C++. La ejecución mediante flujo de datos es dependiente de los datos. El flujo de datos entre los nodos en el código G determina el orden de ejecución.

La programación orientada a eventos extiende el concepto de flujo de datos de LabVIEW para permitir al usuario interacción directa con el programa. La programación basada en eventos también permite otras actividades asíncronas para influenciar la ejecución del código G en el diagrama de bloques.

Capacidades multi-objetivo y plataforma:

Con aplicaciones de LabVIEW se puede tener como objetivo procesadores multinucleo y otros tipos de hardware paralelos como FPGAs. También puede automáticamente escalar aplicaciones de LabVIEW a CPUs con dos, cuatro o más núcleos, usualmente sin programación adicional.

El código G con la excepción de algunas funciones específicas de algunas plataformas, es portable entre diferentes sistemas LabVIEW para diferentes sistemas operativos. Por lo tanto, se puede usar el mismo código de LabVIEW indistintamente de si se utiliza en sistemas Windows, Mac OS X o Linux.

Ahora bien, la programación orientada a objetos es un acercamiento popular a través de una gran variedad de lenguajes de programación. Permite una gran variedad de ítems similares, aunque diferentes, de ser representados como una clase de objetos en el software. LabVIEW provee herramientas y funciones para que se puedan utilizar técnicas de programación orientada a objetos en código G.

Paralelismo:

LabVIEW permite paralelismo (multithreading) automático a su código. En otros lenguajes si se requiere ejecutar código en paralelo, se deben manejar los hilos múltiples manualmente. El entorno LabVIEW, con el compilador y sistema de ejecución trabajando juntos, automáticamente ejecuta el código en paralelo siempre que sea posible.

Entorno de programación de LabVIEW:

Debido a que el entorno de programación varía considerablemente de los lenguajes basados en texto, se procederá a explicar brevemente el funcionamiento de su IDE para así poder entender de mejor manera la forma de programar con LabVIEW y su lógica.

La ventana inicial para crear o abrir un proyecto existente es la mostrada en la siguiente figura.

Figura 7. *Ventana para la creación de proyectos en LabVIEW.*

Explorador de Proyectos:

El explorador de proyectos el cual se aprecia en la figura 8 se utiliza para administrar los proyectos y VIs en LabVIEW.

Este cuenta con los siguientes ítems:

- Project root: Contiene todos los otros ítems del proyecto. La etiqueta en el project root incluye el nombre de archivo del proyecto.
- My Computer: Representa a la computadora local como destino del proyecto.
- Dependencies: Incluye VIs e ítems requeridos por dichos VIs.
- Build Specifications: Incluye las especificaciones de compilación para la distribución fuente, además de las especificaciones de compilación de otros módulos y herramientas de LabVIEW.

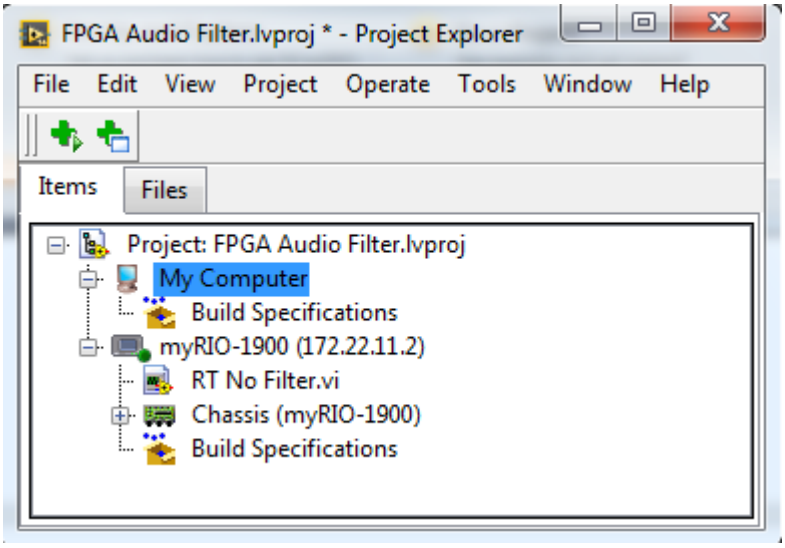


Figura 8. *Project Explorer en LabVIEW.*

Partes de un VI:

Panel Frontal

La ventana de panel frontal es la interfaz de usuario para el VI. Dicho panel se crea con controles e indicadores que son terminales interactivas de entradas y salidas para el VI respectivamente.

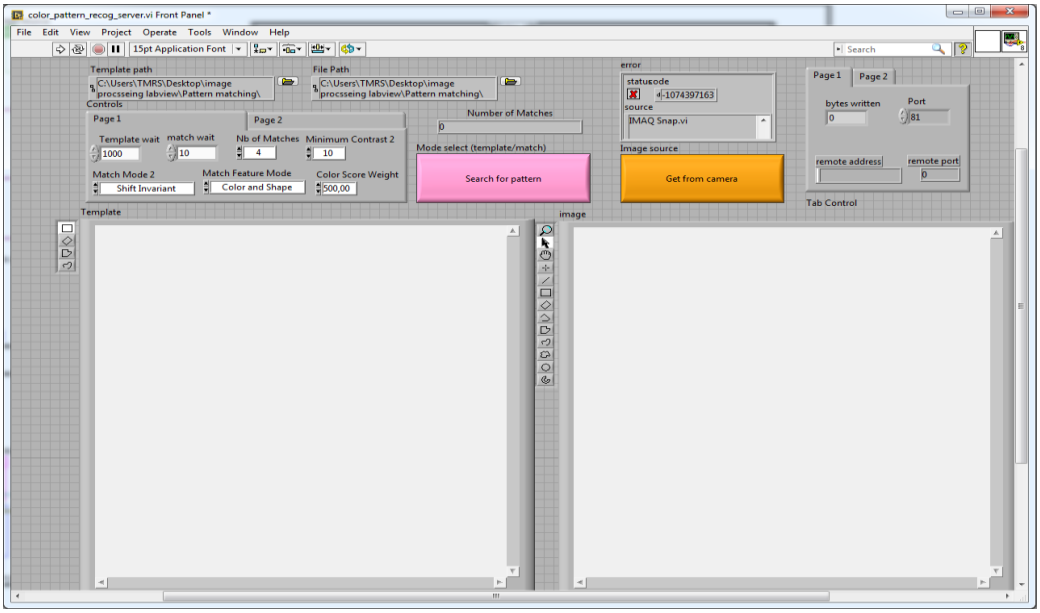


Figura 9. *Panel Frontal de un VI.*

Diagrama de Bloques:

Luego de crear el panel frontal, se agrega código usando representaciones gráficas de funciones para controlar los objetos del panel frontal. El diagrama de bloques contiene el código gráfico del VI y a su vez los objetos del panel frontal son representados por medio de terminales en el diagrama de bloques.

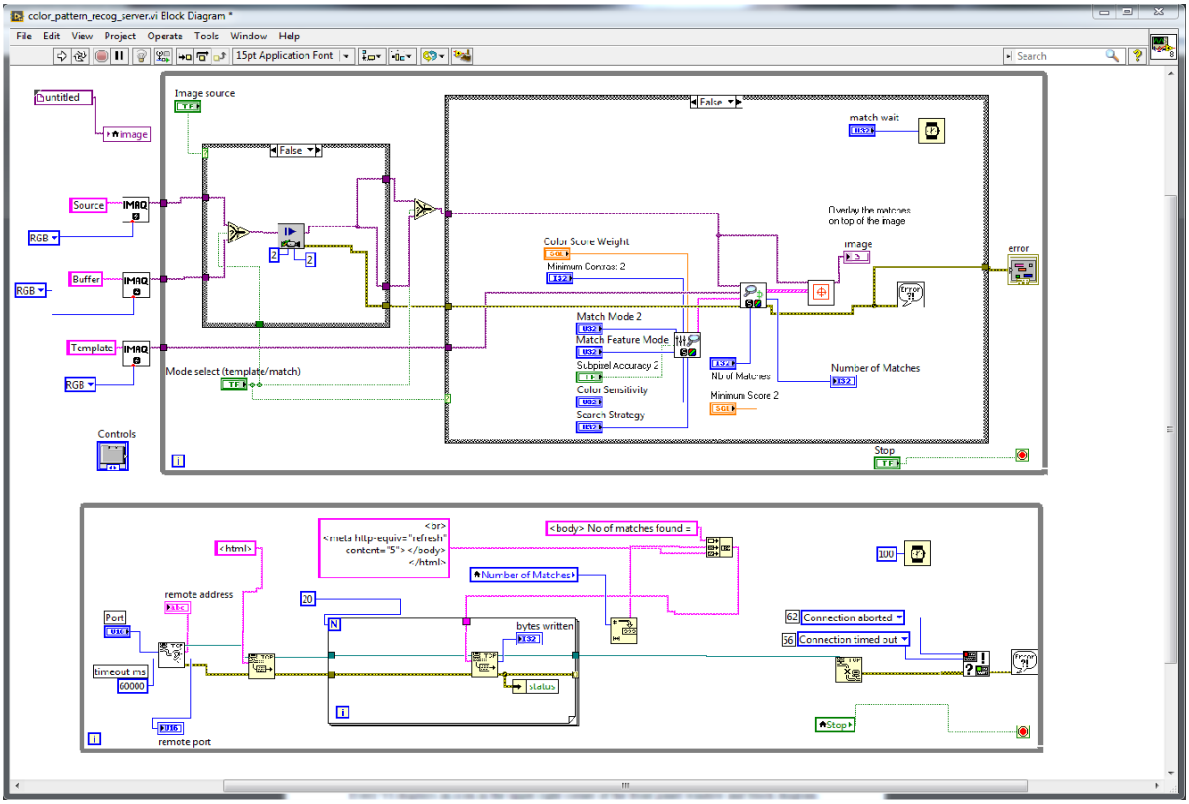


Figura 10. *Diagrama de bloques de un VI.*

FPGA:

Los FPGA (Field Programmable Gate Arrays) son chips de silicio reprogramables que poseen una alta velocidad, confiabilidad y versatilidad ya que pueden ejecutar software para sistemas basados en procesadores, pero no están limitados a la cantidad de núcleos disponibles.

Además a diferencia de los procesadores, los FPGAs son intrínsecamente paralelos, de esta manera las operaciones que se están llevando a cabo no deben competir por los mismos recursos para poder ser ejecutadas. Cada proceso independiente es asignado a una sección dedicada del chip y puede funcionar autónomamente sin influencia de otros bloques de lógica. Como resultado de esto el rendimiento de una parte de la aplicación no es afectado cuando se agregan otras tareas de procesamiento.

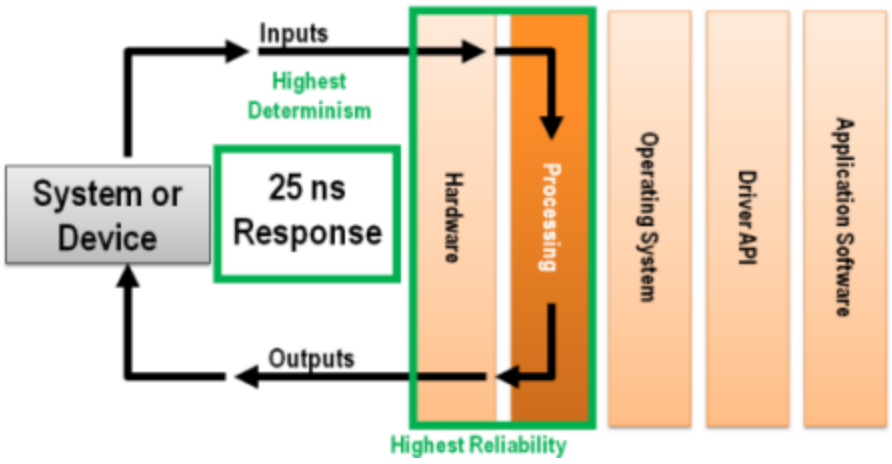


Figura 11. *Caracterización del procesamiento basado en hardware de los FPGAs.*[6]

Cada FPGA está hecho con un número finito de recursos predefinidos con conexiones programables para implementar circuitos digitales reconfigurables y bloques de E/S que le permiten al circuito interactuar con el exterior.

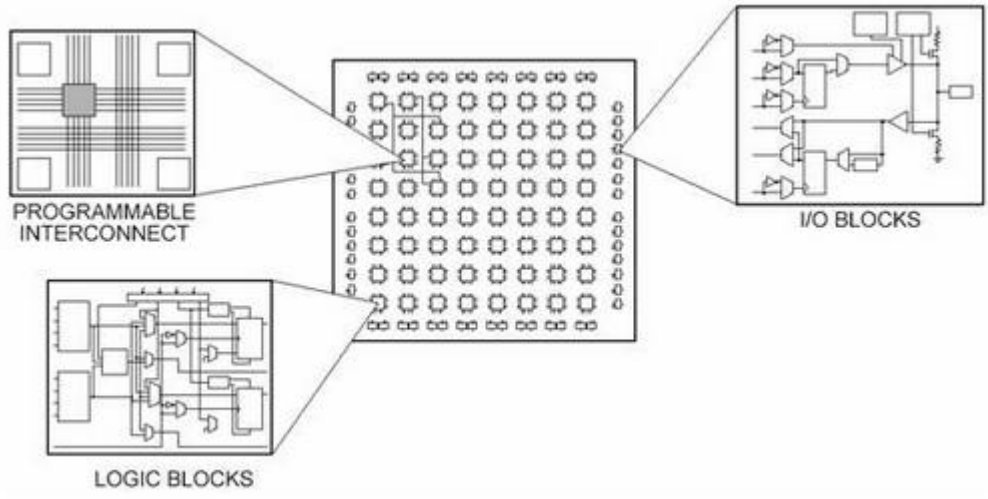


Figura 12. *Diagrama básico de un FPGA.*[6]

Control PID

Un controlador PID es una de las herramientas más comunes en las aplicaciones de control automático debido a su simplicidad y excelente rendimiento en la mayoría de aplicaciones, es por esta razón que el mismo es utilizado en aproximadamente un 95% de aplicaciones industriales de lazo cerrado.

El PID observa un punto de referencia definido y lo compara con el valor actual de la variable del proceso (VP). Al hacer esta comparación lo que se busca es encontrar una disparidad entre los valores de ambas señales, ya que esto implica que se deban aplicar acciones correctivas para disminuir el error de la variable del proceso.

Para hacer las correcciones necesarias a la señal de salida, el PID se basa en tres variables distintas: La ganancia proporcional Kp, la ganancia integrativa Ki, la ganancia derivativa Kd. Las cuales se utilizan para modificar los aspectos dinámicos del sistema como lo son el tiempo de subida, la sobreenlongación, tiempo de asentamiento y el error de estado estacionario.

Respuesta	Tiempo de Subida	Sobreenlongación	Tiempo de Asentamiento	Error Estado Estacionario
Kp	Aumenta	Disminuye	No afecta	Disminuye
Ki	Disminuye	Aumenta	Aumenta	Elimina
Kd	No afecta	Disminuye	Disminuye	No afecta

Al modificar los valores de dichas ganancias se puede ajustar el PID de forma que el sistema a controlar llegue a una respuesta dentro de los parámetros deseados para el proceso que se lleve a cabo, sea este lograr estabilidad, o incluso aumento del tiempo de respuesta.
Uno de los métodos más utilizados para el ajuste de las ganancias de un controlador PID es el método de Zieger-Nichols

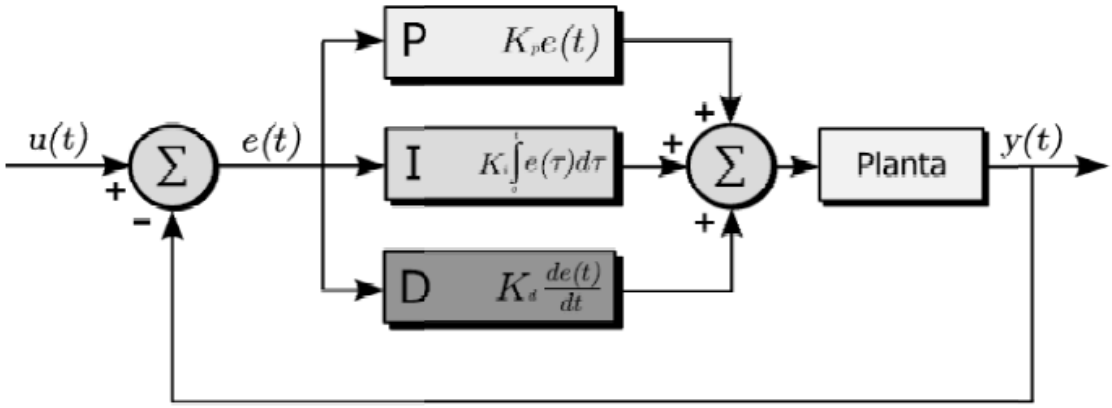


Figura 13. *Esquema básico de un PID.*[11]

Visión Por Computador:

El estudio y aplicación de sistemas de visión es un tema de especialización que generalmente requiere de varios cursos universitarios para poder obtener un buen entendimiento del mismo, en esta sección se limitará a expresar los aspectos más relevantes del pre-procesamiento de imágenes en sistemas de visión por computador.

Generalmente se tiene un concepto erróneo de lo que es visión por computador, y en muchos casos se cree que es algo similar a la adquisición de imágenes como se realiza en las fotografías digitales o bien en películas. Sin embargo los sistemas de visión por computador se encargan de adquirir imágenes para poder procesarlas mediante el uso de algoritmos computacionales y así poder extraer información de dichas imágenes, información que luego será utilizada en distintos procesos según sean las necesidades de la aplicación desarrollada.

Algunas de las aplicaciones más comunes de los sistemas de visión son:

1. Inspección Visual: Es decir, se analizan objetos se comparan con un patrón establecido para determinar si su forma, color, tamaño, orientación o alguna otra característica cumple los requisitos necesarios.
2. Monitoreo de Procesos: Este tipo de aplicaciones son aquellas en que se realiza un monitoreo de las condiciones de un sistema de producción o bien una etapa del proceso,

por ejemplo medición de temperatura haciendo uso del espectro infrarrojo en aplicaciones donde el uso de sensores como termocuplas no es tan adecuado.

3. Clasificación de objetos: Similar a la inspección visual, se procesan imágenes de objetos y se comparan con patrones establecidos para así determinar un sistema de clasificación para los mismos.

Los sistemas de visión por computador dependen de muchas variables ambientales, tales como las condiciones de iluminación, el medio en que se desplaza la luz lo cual afecta su índice de refracción, el ángulo de incidencia de la luz, entre otros.

Es por esta razón que los sistemas de visión están conformados por los siguientes componentes:

1. Sistema de iluminación
2. Filtros ópticos
3. Lentes ópticos
4. Cámara
5. Computador
6. Software

Iluminación:

La iluminación es uno de los factores de mayor importancia para las aplicaciones de visión, ya que de esta dependen factores como el contraste de la imagen, nitidez, entre otros. Es por esto que detalles como la posición de la fuente de luz influyen sobremanera en la aplicación, colocar la fuente de iluminación detrás del objeto puede ayudar a definir sus bordes pero se pierde nitidez en el centro del objeto, o bien, poner la iluminación sobre el objeto ayuda a definir ciertos detalles pero puede crear sombras que influyan en la detección de bordes o patrones.

Incluso el tipo de iluminación y su longitud de onda son cuestiones a tomar en cuenta para cualquier aplicación de visión, ya que ciertas longitudes de ondas (colores) ayudan a resaltar detalles en un objeto esconder otros no tan importantes.

Factores como la frecuencia de trabajo de las fuentes de iluminación también afectan, por ejemplo, los fluorescentes trabajan a una determinada frecuencia (se prenden y apagan) la cual

hace que dicho parpadeo de luz no sea detectable para el ojo humano, sin embargo algunas cámaras podrían captar el parpadeo lo que afectaría la aplicación.

Filtros Ópticos:

Los filtros ópticos se utilizan para permitir el paso solo de determinadas longitudes de onda, para así evitar interferencia de luz en la imagen. Estos se colocan justo frente a la lente de la cámara, los más comunes son los filtros de color y los polarizantes.

Lentes Ópticos:

Los lentes son aquellos encargados de dirigir la luz hacia los sensores ópticos de la cámara. Al igual que los sistemas de iluminación, las lentes son muy variadas y dependen de la aplicación en la cual se vayan a utilizar. Sin embargo presentan el inconveniente de que al ser objetos sujetos a un proceso de manufactura, pueden presentar defectos, mejor conocidos como aberraciones algunos de estos defectos son: miopía, astigmatismo, curvatura de campo entre otros.

Cámara:

Es el dispositivo que toma la imagen y la almacena en película fotográfica (cámaras analógicas) o en una memoria flash (cámaras digitales). La misma posee sensores que captan los haces de luz y los convierten en información de color para luego ser grabados en forma de bits en el caso de las cámaras digitales.

Los sensores utilizados por las cámaras digitales son CCD o bien CMOS, en el caso de los sensores CCD son foto-diodos que transforman los fotones en electrones y “hoyos” con carga positiva y luego los recolecta para formar una carga. Cada sensor tiene una limitante física de cuanto puede cargarse, así que debe transmitir su información a registros de lectura para así poder almacenar la información luminosa.

En el caso de los sensores CMOS también se utilizan foto-diodos para almacenar la carga, sin embargo esta no se transmite secuencialmente como en los CCD, sino que se puede acceder a la ubicación de cada foto-diodo para descargarlo individualmente, este comportamiento es muy similar al de la memoria de acceso aleatorio (RAM).

Computador y Software:

EL computador es quien va a aportar el poder de procesamiento para poder ejecutar la aplicación de visión requerida, y el software es el encargado de aplicar los algoritmos de procesamiento a las imágenes.

Procedimiento Metodológico:

Inicialmente se analizaron cada uno de los componentes incluidos en los distintos kits disponibles para el dispositivo NI myRIO, los cuales son el starter, embbedded y mechatronics kit.
Para esto se utilizó el documento "NI myRIO Project Essentials Guide" desarrollado por Ed Doering, el cual contiene un paso a paso de ejemplos a realizar con cada uno de los componentes de los kits anteriormente mencionados, así como los respectivos VIs a utilizar con el myRIO para ejecutar los ejemplos. Este proceso de pruebas tomó varios días antes de dar lugar a la etapa de desarrollo de los demos.

Una vez determinadas las capacidades del myRIO con cada uno de los componentes de los kits y sus posibles aplicaciones se determinaron más de 15 demos distintos los cuales se podrían implementar con el dispositivo, estos son:

Demos de nivel 1

- Control de motor a pasos.
- Control de servomotor.
- Medición de posición.
- Medición de torque.
- Medición de distancias.
- Medición de aceleración en 2 ejes.
- Captura de imágenes.

Demos de nivel 2

- Control de posición de disco.
- Control de movimiento en 2 ejes.
- Análisis de vibraciones.
- Evasión de objetos.

- Simulación Theremin.
- Reconocimiento de patrones en imágenes.

Demos de nivel 3

- Registro de datos de sensores.
- Despliegue de resultados, imágenes y videos en pantalla de LEDs RGB.
- Simulación de Control de péndulo invertido usando el cubo Quansar.

Demo Principal

- Sistema de seguimiento autoajustable en tiempo real de objetos mediante reconocimiento de patrones.

Todos estos demos a excepción del péndulo invertido con el Quansar cube y la pantalla LED han sido diseñados para ser realizables utilizando un solo sistema de hardware.
Al tener la lista de todos los demos que se pueden integrar al sistema se consideró cuales eran los demos que son primordiales para el sistema y el uso que se le va a brindar y cuales demos no tan críticos pueden ser agregados más adelante una vez el prototipo haya sido probado.

Para esto se organizó una reunión representantes del departamento de Marketing. En dicha reunión se explicó los objetivos del sistema y las posibilidades de demos disponibles; para lo cual se recibió retroalimentación valiosa por parte de los ingenieros del equipo, entre sus aportes más valiosos se destacan:

- El myRIO debe poder ser desmontable del sistema para así poder mostrar su portabilidad.
- Los demos vistosos y excéntricos son más llamativos que los demos funcionales a la hora de mostrar el dispositivo.
- Se debe presentar la oportunidad al usuario para utilizar y desarrollar sus propias aplicaciones in situ.
- Considerar el limitado número de puertos para realizar conexiones que posee el myRIO.

Ahora bien, otras de las consideraciones a tomar en cuenta para decidir sobre la implementación final del sistema corresponden a su complejidad y tiempo disponible, por esta razones se determinó que debido a su complejidad y al hecho de que representa prácticamente todo el hardware necesario para todos los demos diseñados, se implementaría únicamente el sistema de seguimiento en tiempo real por reconocimiento de patrones de objetos. De esta manera habría una optimización del tiempo necesario para crear la aplicación y sus correspondientes partes: diseño, hardware, software, optimización.
Además, al implementar este demo, las modificaciones al sistema necesarias para implementar las demás aplicaciones son mínimas.

Finalmente basado tanto en las limitaciones de tiempo, complejidad y opiniones de los ingenieros de campo, se decidió que el sistema, aparte de sus requisitos previos debe cumplir con las siguientes características:

- myRIO desmontable.
- Solo un demo de alto nivel en el que se demuestran las cualidades de control automático, sistemas de visión y procesamiento del myRIO.
- El usuario debe ser capaz de realizar conexiones al myRIO.

Implementación de la solución:

Basado en estos requisitos la propuesta de diseño consiste en un sistema combinado de cámara de video y sensor ultrasónico cuyo movimiento está dado por un servomotor, donde este a su vez se encuentra sobre una base giratoria la cual debe su movimiento a un motor a pasos en su base. A su vez, se cuenta con un par de protoboards las cuales servirán para que los usuarios desarrollen pequeñas aplicaciones para probar el myRIO.
Debido a la limitada cantidad de puertos del myRIO, ambos sistemas son independientes (cámara y protoboards) los cuales por medio de relés entrarán en contacto con los terminales del myRIO según sea la necesidad del usuario.

Un modelo inicial del sistema se puede apreciar en las siguientes imágenes:

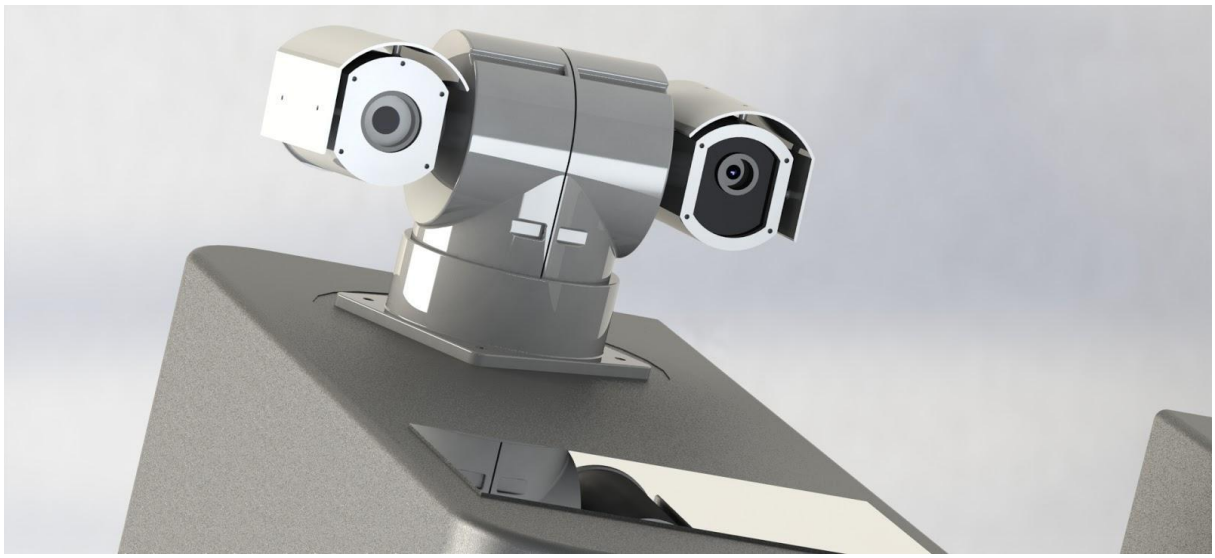


Figura 14. *Modelo de la base para la cámara del sistema.*

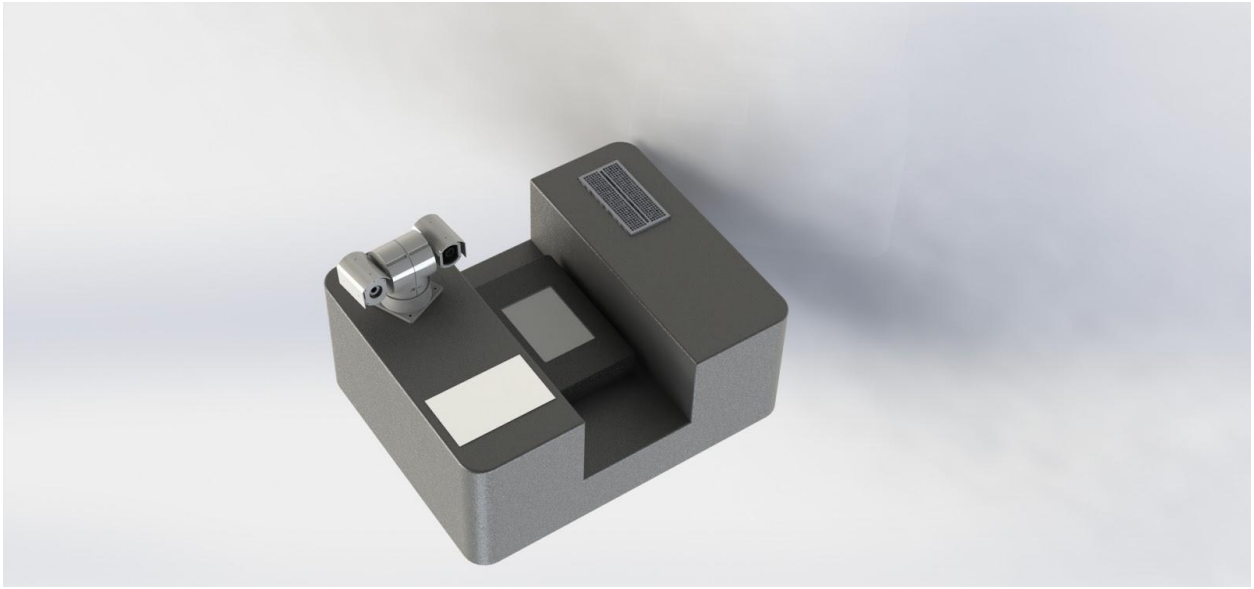


Figura 15. *Modelo de la base para el myRIO.*

Hardware del Prototipo:

Como se aprecia en las figuras 14 y 15 el sistema se encuentra sobre una base o plataforma en la cual van todos los componentes del myRIO demobox, tanto sobre como debajo de la misma.
El mismo consta de un sistema de rastreo en tiempo real por medio de un sistema de visión, un sistema similar al de las cámaras de vigilancia encargadas del monitoreo de intrusos y demás.

Este sistema funciona en conjunto con el software de sistema de visión, de manera que el programa analiza el desplazamiento de un objeto patrón a lo largo del espacio y transmite las mediciones de desplazamiento en los ejes X y Y respecto a una referencia establecida. Esta medición de distancia se realiza tomando como unidad de medida el número de píxeles de la imagen, luego a estos píxeles se les da un valor correspondiente en milímetros para así poder hacer una relación entre milímetros y grados rotacionales.

El objetivo de transformar el número de píxeles en la imagen a un valor de grados rotacionales es el hecho que el sistema obtiene su movimiento debido a un par de motores cuyas funciones son las siguientes:

Motor a pasos: La función de este motor, el cual se encuentra ubicado bajo la base del sistema de cámara es mover el disco en el cual esta se encuentra montada, de esta manera el sistema obtendría rotación en el plano XZ.

Servomotor: La función de este motor, el cual se encuentra ubicado dentro de la cabeza del sistema de cámara es la de rotar la base de la cámara, de esta manera la cámara obtiene rotación en el plano ZY.

Al combinar la acción de ambos motores, se le permite al sistema poder seguir al objeto patrón en el espacio independientemente del desplazamiento del mismo.

También se cuenta con una pantalla LCD para la visualización de datos tanto del sistema de visión como de posibles aplicaciones que se agreguen luego tales como los demos de menor nivel explicados anteriormente, esto con el objetivo de brindar la posibilidad de desarrollo de nuevas aplicaciones y que el sistema no depende únicamente de la pantalla de una computadora.

Ahora bien, del lado derecho de la base se encuentran un par de MXP Breadboard de la marca DIGILENT las cuales cuentan con las terminales previamente ensambladas para encajar con los puertos MXP del myRIO, de esta manera se le brinda al usuario común la posibilidad de conectar sensores y/o circuitos de distintos tipos.

Esto permitiría que los usuarios no solo vean las capacidades del myRIO sino que las exploren in situ directamente.

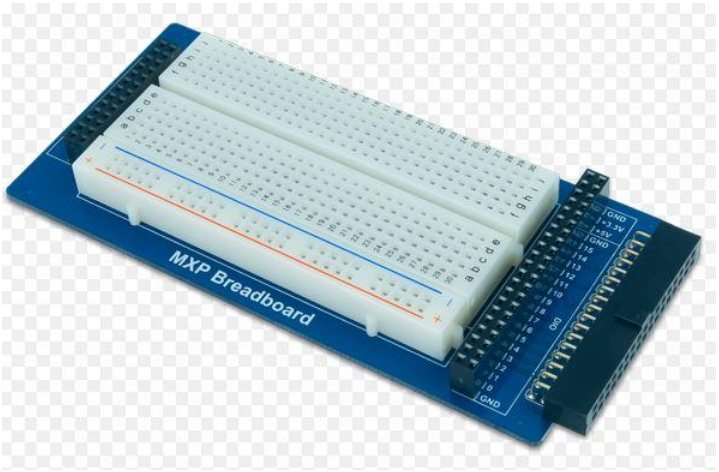


Figura 16. *MXP Breadboard.*

Ahora bien, este sistema irá dentro de un estuche protector de la marca CaseCruzer para así poder transportarlo de manera fácil y eficiente.

Sistema de Visión:

El funcionamiento del sistema de visión del myRIO demobox se basa en el reconocimiento de patrones de objetos, esto es, el software tiene precargado una forma de objeto predeterminada y se encarga de seguir dicho objeto en el cuadro de la imagen para así mantenerlo en el centro del cuadro.

Para realizar dicha acción el software consiste de varios pasos o módulos de sistema los cuales en conjunto permiten la implementación del programa, primeramente se obtiene una captura de imagen desde la cámara del dispositivo para así poder determinar el patrón del objeto a seguir.
Se implementa una función "*detect object"* la cual permite generar un cuadro sobre la imagen plantilla y así determinar el objeto que se desea seguir, sin embargo es necesario hacer un threshold al objeto y así filtrarlo de los demás objetos que pueden estar presentes en el cuadro.

Ahora bien, para mayor facilidad en el manejo de la imagen y a su vez para poder hacer uso de otras funciones que posee LabVIEW y que son necesarias para el sistema, se debe convertir la

imagen de un cuadro RGB a una imagen en escala de grises, lo cual se realiza luego del paso de detección de objetos por forma.

De igual manera se implementa una función de seguimiento en la cual se genera un cuadro que va a buscar y seguir el objeto anteriormente marcado en la función de detección de objetos para así poder tener un registro en tiempo real de su posición actual. Esto con el objetivo de que el objeto no quede fuera del campo de visión de la cámara.

Una vez se tengan estas funciones implementadas, se procede a realizar mediciones en la imagen, dichas mediciones tienen en cuenta el desplazamiento en píxeles del objeto desde una referencia fija tanto en el eje x como en el eje y, esto con el fin de poder determinar el desplazamiento del objeto en ambos ejes y saber cuánto debe desplazarse cada motor de la base de la cámara para realizar un seguimiento adecuado.

En la siguiente imagen se observa como el sistema identifica el objeto, establece un sistema de coordenadas y realiza las mediciones desde los bordes del mismo hasta la referencia fija (en este caso la referencia se encuentra únicamente para la medición en el eje x)

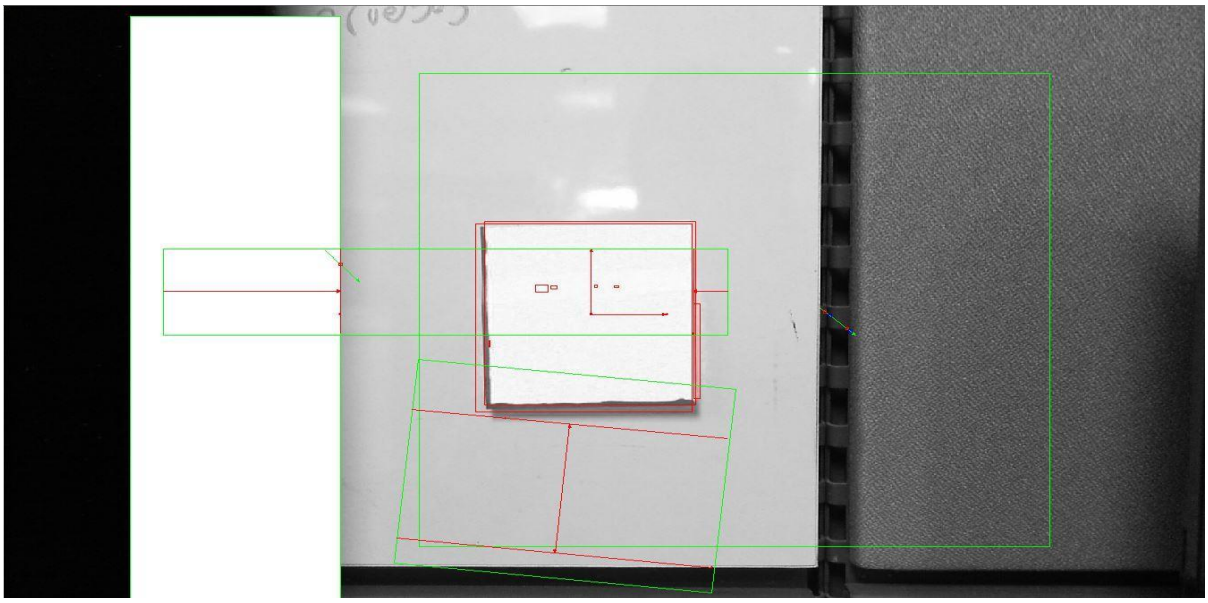


Figura 17. *Sistema de visión reconociendo el objeto patrón.*

Nuevamente se puede apreciar el funcionamiento del sistema pero dentro de su interfaz de LabVIEW en la siguiente imagen.

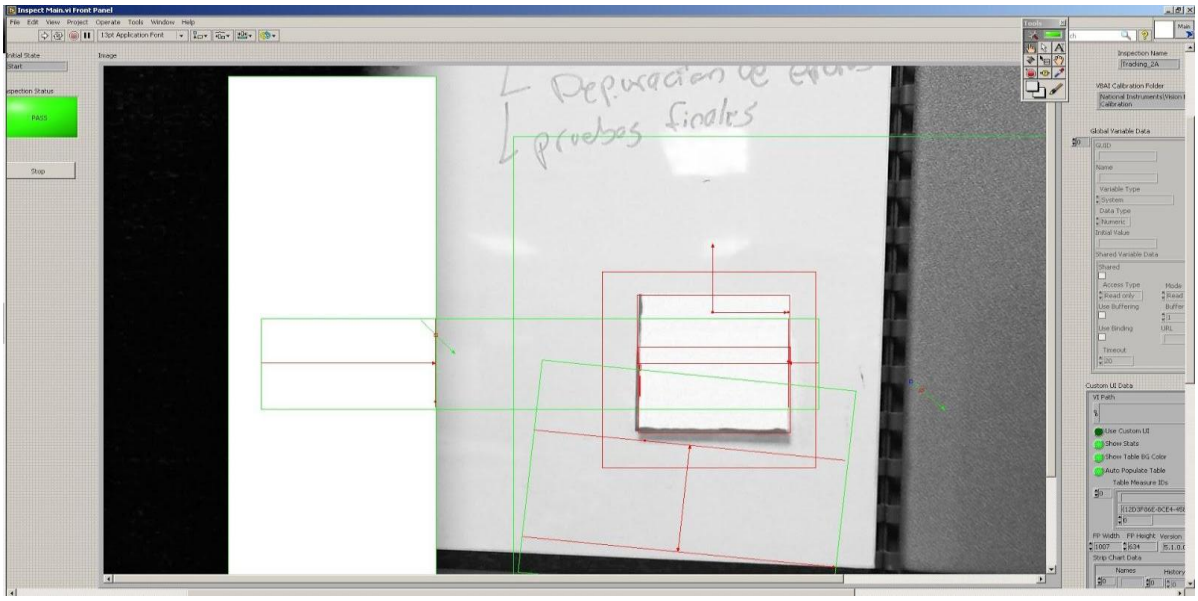


Figura 18. *Panel frontal del sistema de visión funcionando.*

Explicación del código del sistema de visión:

Para la implementación del sistema de visión se crearon varios VIs, en esta sección se explicará el funcionamiento de aquellos VIs más relevantes para la comprensión del sistema.

Main:

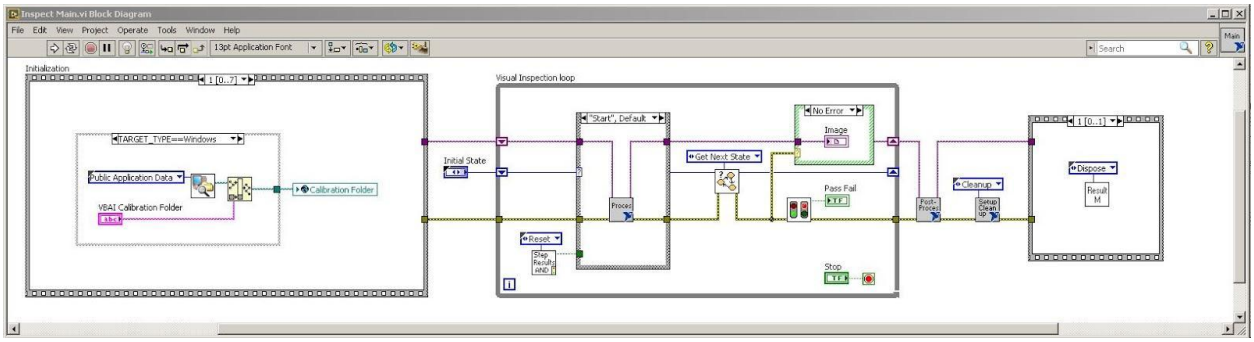


Figura 19. *Diagrama de bloques del VI principal.*

El VI principal consiste de tres partes:

1. Inicialización del sistema: Esta sección se encarga de preparar las rutas donde se almacenarán las imágenes, configurar la cámara, inicializar el sistema de coordenadas, la región de interés, el administrador de la máquina de estados, inicializa las variables globales y los componentes de inspección visual.

2. Ciclo de inspección visual: En esta sección se realiza todo el proceso de las imágenes, consiste en una máquina de estados donde se realiza el procesado de los datos adquiridos por las imágenes de la cámara. Este ciclo se ejecuta indefinidamente hasta que el usuario presione el botón de Stop.

3. Cierre: Esta última parte del código se encarga de limpiar los registros y de liberar la memoria utilizada.

Ciclo de Inspección Visual:

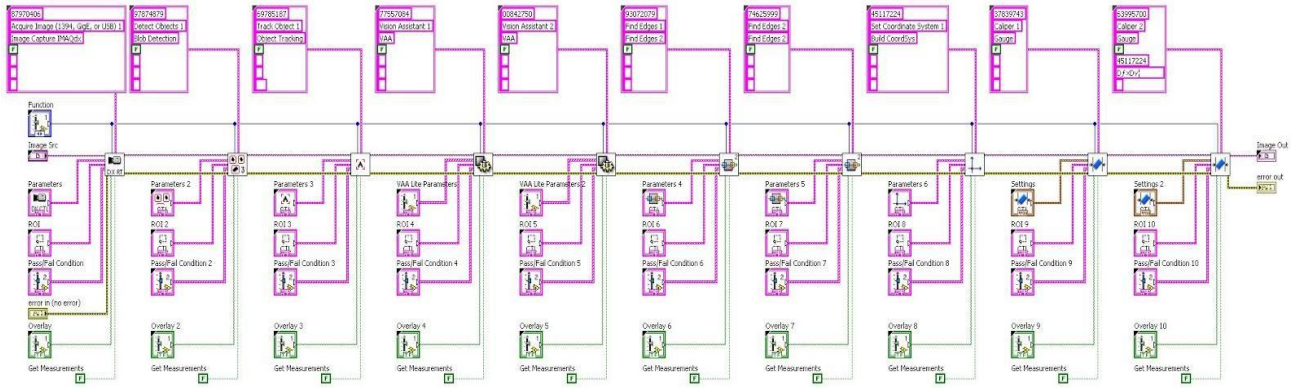


Figura 20. *Diagrama de bloques del ciclo de inspección visual.*

Este VI es el que se encarga de todo el proceso y análisis de las imágenes.

Como se había explicado anteriormente en el funcionamiento del sistema, consta de las siguientes partes en el orden respectivo

:

1. Adquisición de imágenes.
2. Detección de objetos.
3. Reconocimiento de patrones.
4. Extracción de rango de imagen HSV para convertir a escala de grises.
5. Erosión de la imagen y threshold para la creación de la referencia en el eje x.
6. Detección de bordes en ambos ejes.
7. Creación de sistema de coordenadas.
8. Medición de distancia entre bordes en ambos ejes.

Software de Control:

El control del sistema, en este caso de los motores, hace uso del módulo NI Softmotion de LabVIEW.

Este control hace uso de la señal de medición que toma el sistema de visión, dicha medición brinda un dato en medidas reales de milímetros (para lo cual primero debe establecerse un patrón de tamaño). Dicho dato se transforma matemáticamente desde una medición linear hacia una medición angular, esto con el objetivo de obtener la cantidad de grados que debe rotar cada uno de los motores.

Este dato angular se utiliza para que la función de Softmotion Linear Travel Express.VI envíe una señal de control al motor para que así este gire los grados necesarios hasta llegar a la posición requerida.

Ahora bien, debido a la naturaleza del sistema, el cual no cuenta con un sistema de coordenadas fijo y a la forma en que el mismo obtiene las mediciones angulares, es necesario que sea un control incremental y no absoluto, así si bien se pierde la referencia inicial 0, el sistema responderá de una mejor manera a las señales que se le envían, i.e cuando el mismo reciba un dato de ángulo, se moverá esa cantidad respecto a su posición actual, cosa que permitiría al sistema el uso de ángulos negativos y así determinar la dirección de rotación. En cambio un sistema de medición absoluto dificulta la lógica de programación ya que habría que hacer una conversión matemática para determinar el ángulo de la nueva posición del objeto patrón en la imagen respecto a un sistema coordenado que se desplaza. Esto implicaría el uso de matrices de rotación y traslación por lo cual la matemática se dificulta.

PID:

Para el diseño del PID se utilizó el método de Zieger-Nichols como primer aproximación a los valores de las ganancias del sistema.
Sin embargo es importante hacer la notación que la planta en sí es estable, el servomotor común y el de rotación continua, por lo cual el objetivo del PID en este caso es el de disminuir el tiempo de asentamiento y el tiempo de subida de la respuesta del sistema.

El modelo matemático de la función de transferencia de un servomotor es el siguiente:

$$\frac{C(s)}{R(s)} = \frac{\frac{K}{J}}{S^2 + (B/J)s + (K/J)}$$

Ecuación 1. *Función de transferencia de un servomotor en términos de variables.*

Donde J representa el momento de inercia de la carga y B el coeficiente de fricción viscosa del motor.

Se eligió como primera aproximación teórica darle al sistema una sobreenlongación máxima ante la respuesta escalón de 0.2 y un tiempo pico de 1 segundo. Esto para calcular inicialmente el valor de K y así poder despejar numéricamente la función de transferencia. El procedimiento matemático puede observarse en el apéndice A.

$$G(s) = \frac{12{,}45}{s^2 + \frac{5}{2} \cdot s + 12{,}45}$$

Ecuación 2. *Función de transferencia de un servomotor.*

El modelo básico de un controlador PID es el siguiente:

$$G_c(s) = K_p \cdot \left(1 + \frac{1}{T_{is}} + T_{ds}\right)$$

Ecuación 3. *Estructura de un PID.*

En este caso se determinó como punto inicial utilizar los siguientes valores para las constantes del PID y con estos comenzar el proceso de ajuste del controlador.

$$K_p = 8.67$$

$$T_i = 2.22$$

$$T_d = 0.555$$

Por lo tanto el controlador tiene la siguiente forma:

$$8.67 + 2.22 * \frac{1}{s} + 0.555s$$

Ecuación 4. *PID con Zieger-Nichols antes del ajuste.*

Ahora bien, debido a que este controlador es solo un punto de partida inicial con el cual se puede ajustar la respuesta en lazo cerrado del sistema, para ajustarlo de manera adecuada se eligió utilizar el módulo control system toolbox de Matlab y así ajustar la respuesta lo mejor posible para su uso en el myRIO demobox.

El PID ya ajustado con la ayuda de Matlab es el siguiente:

$$2.4707 + 4.4335 * \frac{1}{s} + 0.34421s$$

Ecuación 5. *PID final luego de ser ajustado con Matlab.*

Resultados:

Estuche:

El diseño se ajustó para que pudiera entrar en un estuche CaseCruzer, como se explicó anteriormente el mismo permite que todo el cableado se mantenga en la parte inferior del sistema donde este no sea visible.

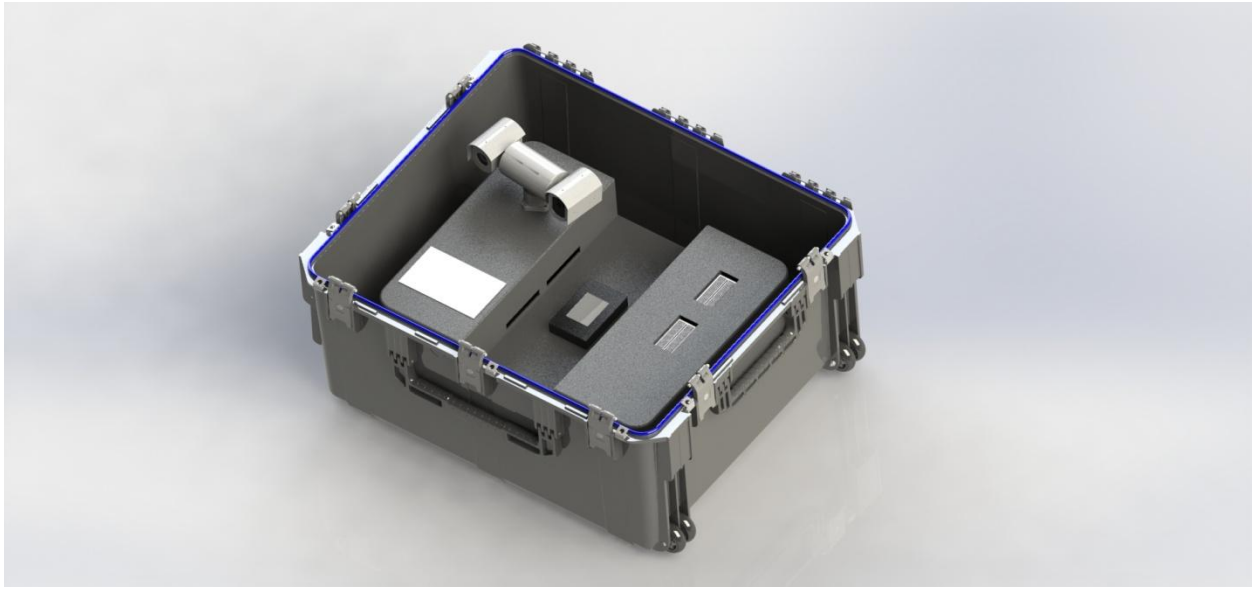


Figura 21. *myRIO demobox en el estuche sin cubierta superior.*

Como se puede apreciar en la imagen anterior, se ajustaron las dimensiones del myRIO demobox para que el mismo pudiera encajar en el estuche seleccionado, cabe mencionar que el estuche llevará una espuma protectora la cual hará un ajuste al demobox para que este no se mueva durante su transporte y amortiguará cualquier golpe al cual pueda ser sometido.

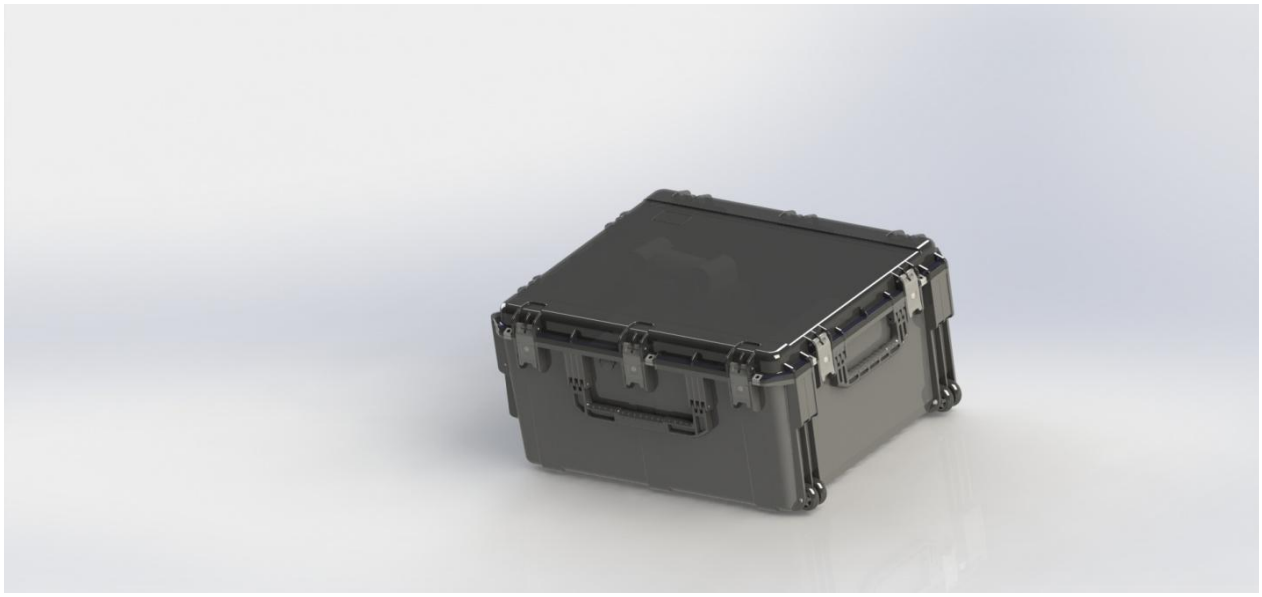


Figura 22. *myRIO demobox en el estuche con la cubierta superior cerrada.*

Se pueden apreciar los componentes principales del demobox en la Figura 21. Los cuales son, las protoboards, el sistema de cámara, el myRIO y la pantalla LCD. Además se ven los orificios desde los cuales se conectará el cableado hacia el myRIO.

Sistema de Visión:

Los resultados obtenidos con el sistema de visión revelaron una serie de factores relevantes tanto para consideraciones durante el desarrollo de aplicaciones similares como para rediseños del sistema del myRIO demobox.

Si bien el algoritmo del sistema funciona de la forma que se espera, hay factores externos que afectan su rendimiento y lo hacen ineficiente en ambientes no controlados.
Se pudo observar que la función de seguimiento del objeto funciona perfectamente siempre que este se mantenga dentro del cuadro de visión de la cámara, al igual que su caliper o función de medición de distancia entre objetos.

Sin embargo uno de los problemas que surgió durante el desarrollo del sistema fue la necesidad de implementar una referencia fija respecto al borde de la imagen para así poder tomar las mediciones desde el borde de la referencia hasta el borde del objeto detectado.

Esta situación se presenta por el hecho que el sistema no es capaz de detectar el límite del cuadro de la imagen en video ya que al moverse la cámara este mismo límite se mueve.

Otra de las condiciones que afectan el sistema es su uso en ambientes irregulares, debido a su naturaleza de utilizar bordes para fijar referencias entre dos objetos y así poder realizar las mediciones de distancia, se puede hacer una falsa detección de bordes en la imagen, esto tiene su razón de ser en que el sistema no diferencia si el borde pertenece al objeto patrón, otro objeto o incluso una grieta en el fondo de la imagen.

Ahora bien, otro de los resultados obtenidos durante la implementación del sistema tiene que ver con el hecho del tipo de rastreo a usar en el algoritmo; inicialmente se realizaba un filtrado de color, sin embargo por cuestiones de iluminación o bien presencia de objetos del mismo color al del patrón se pueden realizar falsas detecciones en la imagen. Por lo tanto se decidió implementar un sistema de rastreo basado en patrones geométricos, de esta manera el efecto de la iluminación ya no es tan relevante para la detección del objeto.

PID:

EL sistema para el cual se diseñó el PID, los servomotores, son en sí estables en su funcionamiento, el objetivo del diseño del controlador era el de disminuir el tiempo de respuesta del sistema para que así se pudiera adaptar al seguimiento en tiempo real del objeto patrón según las señales enviadas por el caliper del sistema de visión.

Utilizando el Control System Toolbox de Matlab se obtuvo la respuesta a una entrada escalón de la planta, la cual en este caso es el servomotor. En la siguiente figura puede observarse su tiempo de subida así como el tiempo de asentamiento sin los efectos del controlador PID.

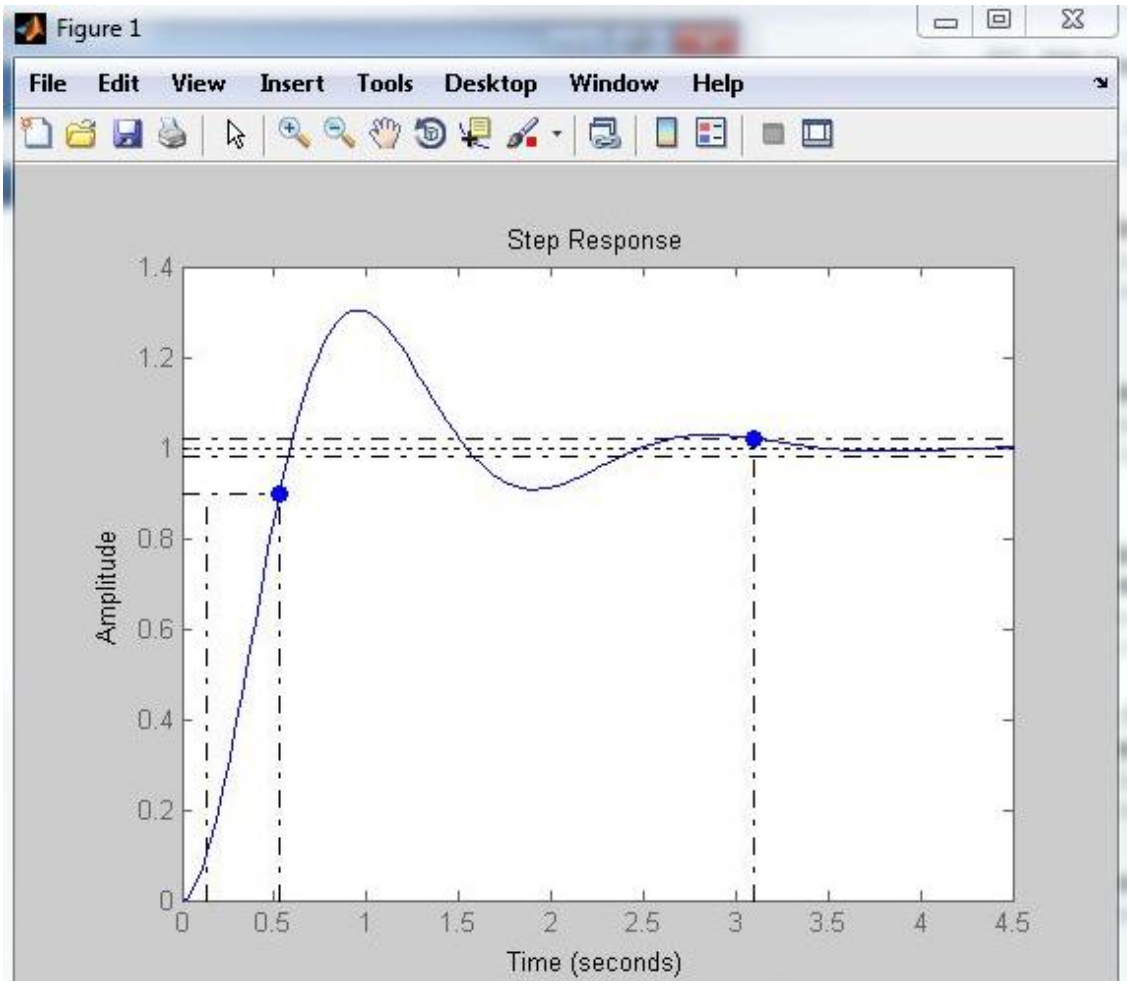


Figura 23. *Respuesta de la planta a la entrada escalón sin controlador PID.*

Luego al tomar como base los valores de las ganancias calculadas mediante el método de Zieger-Nichols se ajustó el PID de forma que se obtuviera una respuesta más rápida del sistema ante una entrada escalón. Esto se realizó utilizando el PID tuner de Matlab. En la Figura 24 se puede observar como el tiempo de subida ha disminuido al igual que el tiempo de asentamiento de la respuesta de la planta.

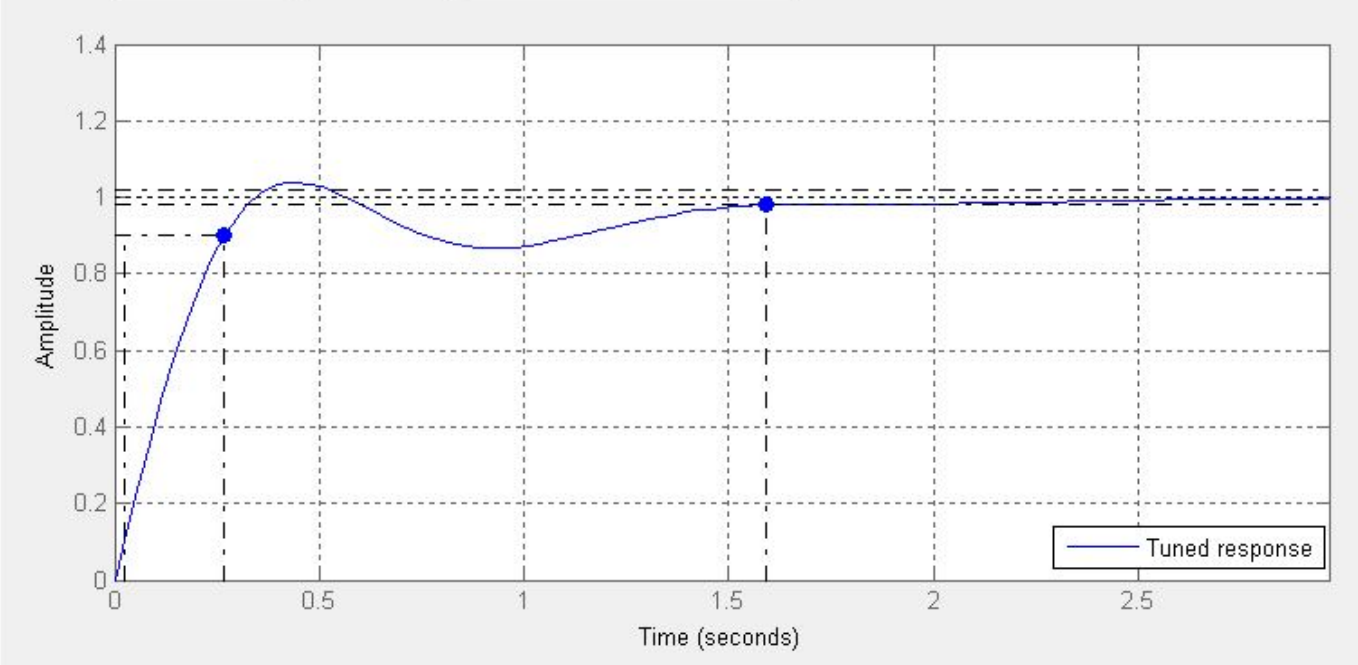


Figura 24. *Respuesta de la planta a la entrada escalón con el controlador PID.*

Por lo tanto se logra observar que el objetivo de la implementación del PID, el cual era disminuir el tiempo de respuesta de la planta se ha logrado cumplir satisfactoriamente.

Consideraciones para rediseño:

Estuche:

En el caso de la caja o estuche del myRIO demobox se pueden realizar varias recomendaciones de rediseño para futuras aplicaciones.
En este caso hay que tomar en cuenta que el sistema al ser un primer prototipo no cuenta con todas las características que el myRIO puede soportar, y es por esta razón que dichas características pueden ser consideradas para agregarse en futuros diseños del demobox. Algunas de estas son las siguientes:

- Terminales para conexión del Quanser cube.
- Pantalla LED RGB 32x32.
- Disminución del tamaño.
- Agregar otros demos de la lista previamente diseñada.
- Agregar un dispositivo USB para almacenar las rutinas de código y eliminar la necesidad de una PC.

Sistema de Visión:

Debido a las complicaciones inherentes de los sistemas de visión respecto a la iluminación presente en el campo de trabajo es necesario controlar o al menos disminuir el efecto de dichos factores al mínimo posible para obtener un funcionamiento óptimo.

Para esto primeramente debe diseñarse un objeto patrón para el sistema, esto es, un objeto con forma y color predeterminados para que el sistema lo pueda identificar de manera rápida y sencilla sin tener que hacer configuraciones extras. Dicho objeto debe desplazarse frente a un fondo patrón para así eliminar la posibilidad de detección de bordes distintos a los del objeto a detectar y que así no se realice ninguna confusión durante la medición de distancias.

Ahora bien, debido al hecho que el myRIO demobox se utilizará en el campo, no se puede controlar qué tipo de iluminación estará presente en cada una de las localidades donde se vaya a presentar, por esta razón es necesario maximizar el contraste entre el objeto a detectar y su

respectivo fondo. Dicho esto, una recomendación para este patrón sería un objeto negro sobre fondo blanco, de esta manera los bordes del objeto serán bastante definidos independientemente del tipo de iluminación presente en el ambiente.

Conclusiones:

Con la elaboración del diseño del primer prototipo del myRIO demobox se lograron sentar las bases para la construcción de un dispositivo capaz de mostrar las cualidades en distintas áreas de la ingeniería por parte del myRIO.

Debido a limitaciones con el tiempo disponible y el nivel de complejidad tanto en hardware como en software, se limitó el diseño a un único demo para así poder realizar pruebas de funcionamiento una vez construido el dispositivo y evaluar su desempeño en el campo.

Se cumplió el objetivo de desarrollar una plataforma que permita hacer simulaciones en el ambiente de la Mecatrónica puesto que el mismo combina sistemas de visión, control automático, control de motores e incluso sistemas embebidos, los cuales son parte de las áreas de estudio de la mecatrónica.

Se implementó software en el lenguaje de programación LabVIEW de manera que el mismo sea fácil de entender y permita ser modificado en posibles rediseños para así poder adaptarse a las características nuevas a ser agregadas al demobox.

El dispositivo es completamente portátil gracias al uso del estuche CaseCruzer el cual permite transportarlo tanto a mano o bien sobre las ruedas que este mismo posee.

Se elaboró un manual con los componentes mínimo necesarios así como los planos de construcción para poder manufacturar el prototipo del myRIO demobox.

Finalmente, el prototipo se ha visto que cumple con las necesidades requeridas por la empresa National Instruments, el mismo será utilizado como base principal para hacer uso de las herramientas del diseño concurrente característico de la mecatrónica y así poder obtener un demobox final listo para ser utilizado en toda la región latinoamericana.

Recomendaciones:

1. Elaborar una lista directa de todos los componentes a utilizar y realizar la compra de los mismos, de esta manera se vuelve más sencillo la realización de pruebas in situ y se ahorra tiempo durante el desarrollo del equipo.

2. Poseer experiencia previa de al menos unos seis meses en la programación con el lenguaje LabVIEW y sus módulos FPGA y Real Time, ya que ambos son esenciales en la utilización de equipos de la gama RIO de National Instruments.

Bibliografía:

[1] Bitter, R., Mohiuddin, T. & Nawrocki, M. (2007). *LabVIEW advanced programming techniques* (2° ed.) Boca Raton, Florida: Taylor & Francis Group.

[2] De Silva, C. (2005). *Mechatronics: An Integrated Approach* Boca Raton, Florida: CRC Press.

[3] Gonzales , A., Martínez de pisón, F., Pernía, A., Alba, F., Castejón, M., Ordieres, J. et al (2006). *Técnicas y algoritmos básicos de visión artificial* La Rioja: Universidad de La Rioja. Servicio de Publicaciones.

[4] Kehtarnavaz, N., & Kim, N. (2005). *Digital signal processing system-level design using LabVIEW*. Burlington, MA, USA: Newnes.

[5] Kuo, B. (1996). *Sistemas de Control Automático* México : Prentice Hall.

[6] Lajara, J. & Pelegrí, J. (2007). *LabVIEW Entorno Gráfico de Programación* (Primera ed.) Barcelona: Marcocombo.

[7] National Instruments (2011, 15 de Agosto). *The History of LabVIEW - National Instruments* Recuperado el 29 de Octubre del 2014, de http://www.ni.com/video/2416/en/

[8] National Instruments (2012, 03 de Mayo). *FPGA Fundamentals - National Instruments* Recuperado el 29 de Octubre del 2014, de http://www.ni.com/white-paper/6983/en/

[9] National Instruments (2013). *LabVIEW Core 1 Course Manual* (August 2013 ed.) Manuscrito no publicado.

[10] National Instruments (2013). *LabVIEW Core 2 Course Manual* (August 2013 ed.) Manuscrito no publicado.

[11] National Instruments (s. f.). *NI LabVIEW - Improving the Productivity of Engineers and Scientists - National Instruments* Recuperado el 29 de Octubre del 2014, de http://www.ni.com/labview/

[12] National Instruments (s. f.). *NI myRIO - National Instruments* Recuperado el 29 de Octubre del 2014, de http://www.ni.com/myrio/esa/

[13] University of Michigan (s. f.). *Control Tutorials for MATLAB and Simulink - Introduction: PID Controller Design* Recuperado el 12 de Noviembre del 2014, de http://ctms.engin.umich.edu/CTMS/index.php?example=Introduction§ion=ControlPID#1

[14] Ogata, K. (2010). *Ingeniería de Control Moderna* (5 ed.) Madrid: Pearson Educación.

[15] Whitley, K. N. (2000). *Empirical research of visual programming languages: An experiment testing the comprehensibility of LabVIEW.* (Order No. 9970079, Vanderbilt University). *ProQuest Dissertations and Theses,* , 321-321 p. Recuperado de http://search.proquest.com/docview/304632012?accountid=27651. (304632012).

[16] Zhong, J. (2006). *PID Controller Tuning: A Short Tutorial* Recuperado el 12 de Noviembre del 2014, de http://saba.kntu.ac.ir/eecd/pcl/download/PIDtutorial.pdf

Apéndice A: Diseño del PID

Diseño de los parámetros del PID por el método de Zieger-Nichols

Las constantes del motor son las siguientes:

$J = 0.4kg - m^2$
$B = \frac{1Nm}{rad}/seg$

Para despejar el valor de K de la ecuación de la función de transferencia del motor primero es necesario despejar la ecuación de la sobre-elongación máxima

$$M_p = e^{-(\zeta/\sqrt{1-\zeta^2})\pi} = 0.2 \quad (1)$$

Al despejar dicha ecuación se obtiene que $\zeta = 0.456$

Se resuelve la ecuación del tiempo pico para obtener el valor de frecuencia

$$t_p = \frac{\pi}{\omega_d} = 1 \quad (2)$$

De la ecuación (2) se obtiene que $\omega_d = 3.14$

Ahora es necesario despejar la frecuencia natural la cual viene dada por:

$$\omega_n = \frac{\omega_d}{\sqrt{1-\zeta^2}} \quad (3)$$

Esto implica que $\omega_n = 3.53$

Como la frecuencia natural es igual a $\sqrt{\frac{K}{J}}$ [11] entonces se despeja que:

$$K = J\omega_n{}^2 = 4.98Nm$$

Ahora bien, finalmente con el valor de K despejado se puede obtener la función de transferencia de la planta la cual es:

$$G_c(s) = \frac{12.45}{s^2 + \frac{5s}{2} + 12.45} \tag{4}$$

Para diseñar el PID se sabe que su ecuación general es la siguiente:

$$G_c(s) = K_p(1 + \frac{1}{T_{is}} + T_{ds}) \tag{5}$$

Con

$$T_i = 0 \wedge T_d = 0$$

La nueva función de transferencia entre la planta y el controlador viene dada por:

$$\frac{12.45k_p}{s^2 + \frac{5s}{2} + 12.45 + k_p} \tag{6}$$

Utilizando el Criterio de Routh-Hurwitz se obtiene que el valor de $k_p < -12.45$, por lo tanto se le asigna un valor de -14.45.

Para encontrar la frecuencia de oscilación sostenida, se sustituye $s = j\omega$ en la ecuación característica del modo siguiente:

$$j\omega^2 + \frac{5}{2}j\omega - 2 = 0 \qquad (7)$$

De esto se obtiene que $\omega = \sqrt{2}$

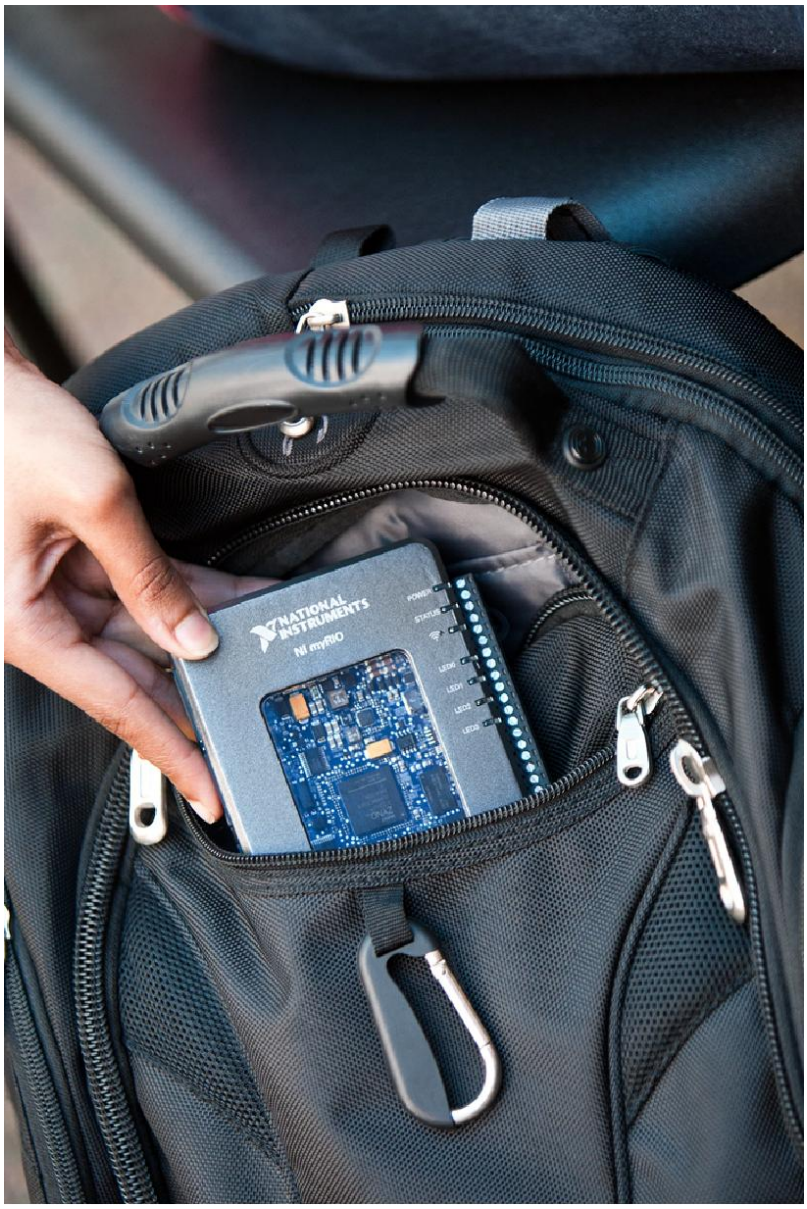
Ahora utilizando la tabla 8-1 de [11] se obtienen los siguientes valores para las constantes del PID:

K_p	8.67
T_i	2.22
T_d	0.55

Apéndice B: Características del myRIO

NI myRIO

Design Real Systems, Fast



El objetivo del desarrollo del NI myRIO es el de entregar a estudiantes universitarios un dispositivo embebido que los pueda ayudar a diseñar y desarrollar sistemas en ingeniería que sean complejos de una forma más rápida y económica.

Debido a sus características el NI myRIO puede ser utilizado para el diseño de sistemas de control, sistemas robóticos, e incluso mecatrónicos. Todo esto mediante el uso de una única herramienta reconfigurable gracias al FPGA que posee el dispositivo.

Explicación del Hardware:

.

El myRIO es un dispositivo que provee al usuario con múltiples entradas y salidas analógicas, digitales e incluso de audio. Además posee conectividad por medio de USB y WiFi bajo el estándar 802.11b,g,n.

Dispone de un procesador programable ARM Cortex-A9 dual-core de 667 MHz y E/S personalizadas de FPGA Xilinx las cuales pueden ser usadas por estudiantes para empezar el desarrollo de sistemas y resolver problemas de diseño más rápido. El dispositivo NI myRIO permite interactuar con el Zynq-7010 Sistema en un Chip completamente programable (SoC) para liberar el poder del software de diseño de sistemas NI LabVIEW en aplicaciones de tiempo-real (real-time RT) y en nivel FPGA.

Además dentro de su propio encapsulado cuenta con 4 LEDs, un acelerómetro y giroscopio y un botón programable. Junto a esto, en los puertos de conexión MXP se cuenta con entradas UART para comunicación entre dispositivos.

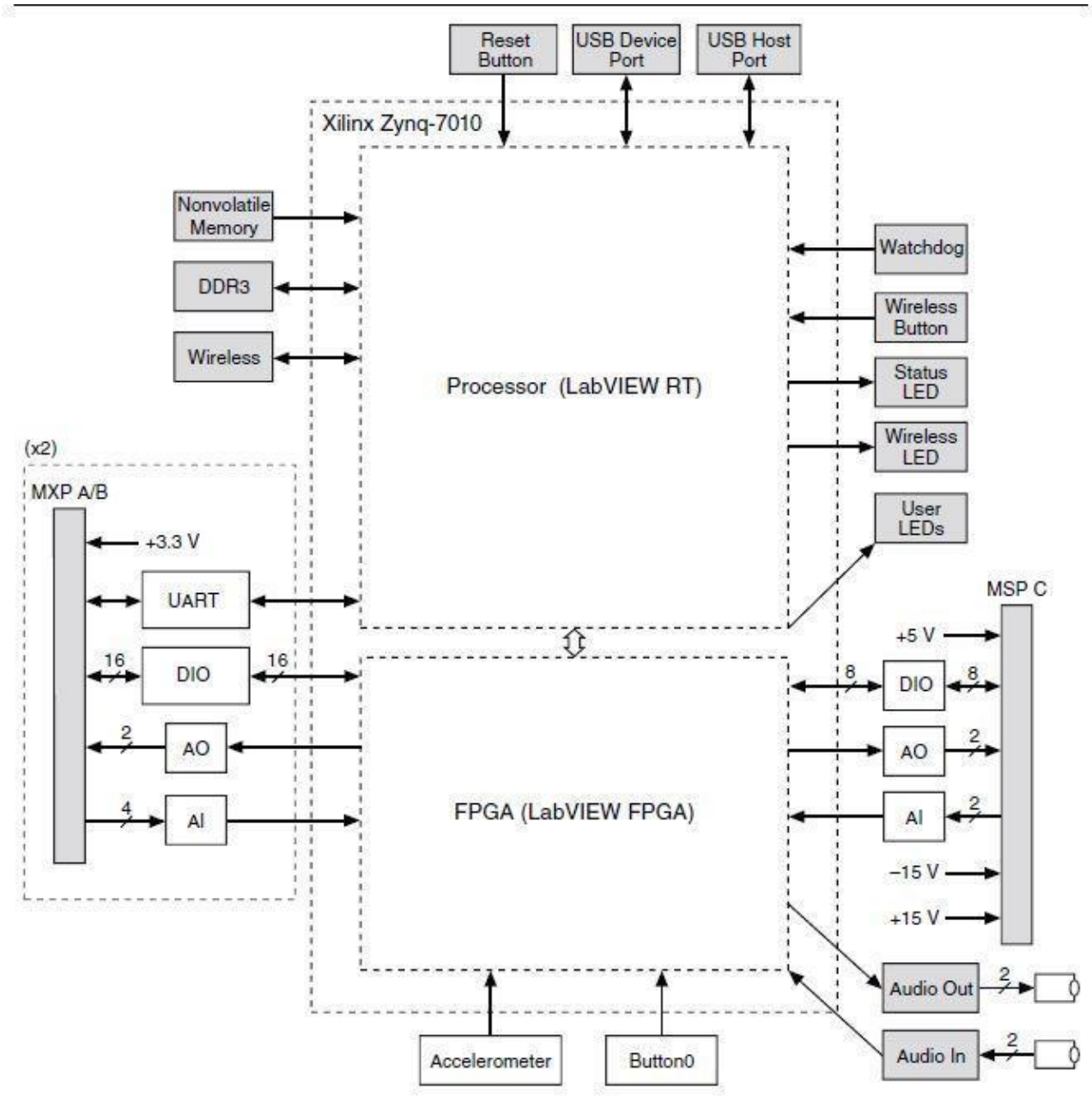


Figura B1. *Diagrama de bloques del NI myRIO.*

Ahora bien, los conectores MXP A y B son idénticos entre sí y las señales que llegan a ambos se diferencia a nivel de software gracias a los nombres de los conectores. Cada uno de los terminales MXP configurados por defecto se observan en la figura 2.

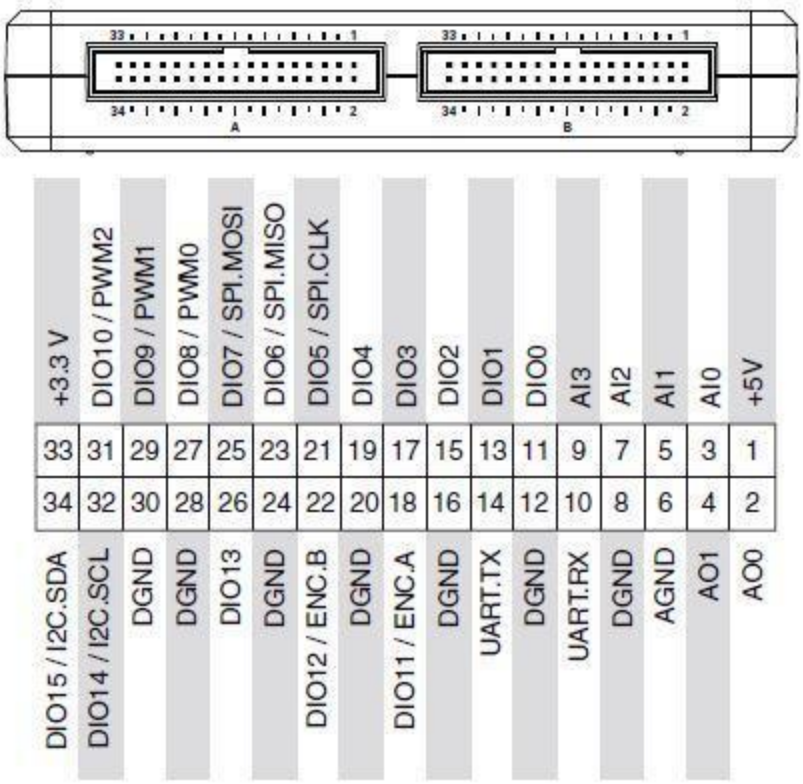


Figura B2. *Pines del puerto MXP*

Al igual que los dos puertos MXP, el myRIO posee entradas y salidas extra tanto analógicas como digitales por medio de su puerto MSP. Su diagrama de terminales se aprecia en la figura 3.

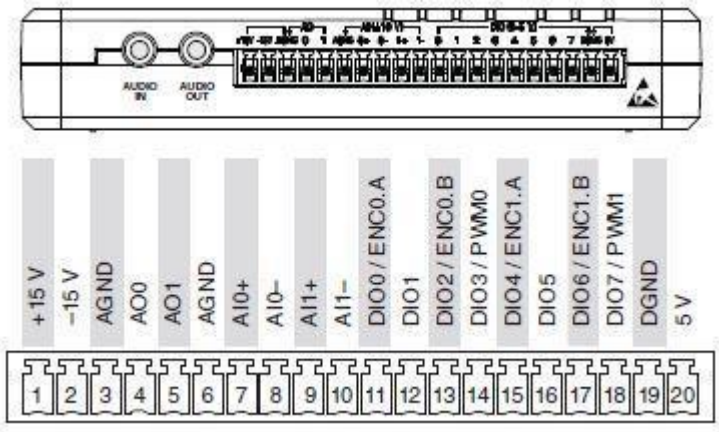


Figura B3. Pines del puerto MSP.

El NI myRIO posee un acelerómetro de tres ejes, el cual realiza un muestreo continuo de cada eje y actualiza el valor medido en el registro de lectura. Estos datos pueden ser utilizados ya sea en su forma *raw* o bien en términos de grados convencionales una vez se ha hecho la conversión correspondiente.

Especificaciones:

Procesador:

Tipo: Xilinx Z-7010

Velocidad de procesamiento: 667MHz

Núcleos: 2

Memoria:

No volátil: 256MB

Memoria DDR3: 512MB

FPGA:

Tipo: Xilinx Z-7010

WiFI:

Radio: IEEE 802.11 b,g,n

Banda de frecuencia: ISM 2.4GHz

Ancho de banda: 20MHz

Canales: USA 1-11, International 1-13

Rango: 150m

Seguridad: WPA,WAP2,WPA2-Enterprise

Puertos USB:

USB: USB 2.0 Hi-Speed

Entradas Analógicas:

Velocidad de muestreo: 500kS/s

Resolución: 12 bits

Protección sobretensión: ±16V

Salidas Analógicas:

Velocidad de actualización: 354kS/s

Resolución: 12bits

Protección sobretensión: ±16V

Tensión al encendido: 0V luego de la inicialización del FPGA

E/S Digitales:

Número de líneas:

Conectores MXP	2 puertos de 16 E/S digitales
Conector MSP	1 puerto de 8 E/S digitales

Niveles lógicos: Entrada compatible con LVTTL 5V, Salida LVTTL 3.3V

Frecuencias de funciones digitales secundarias:

SPI: 4MHz

PWM: 100kHz

Entrada de encoder en cuadratura: 100kHz

I^2C: 400 kHz

Acelerómetro:

Cantidad de ejes: 3

Rango: ±8 g

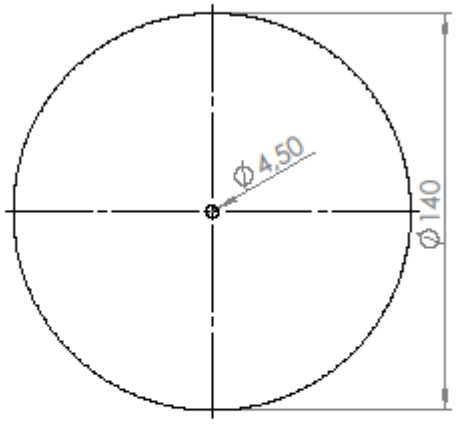
Resolución: 12 bits

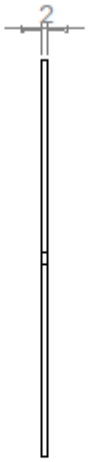
Velocidad de muestreo: 800S/s

Apéndice C: Planos de Construcción

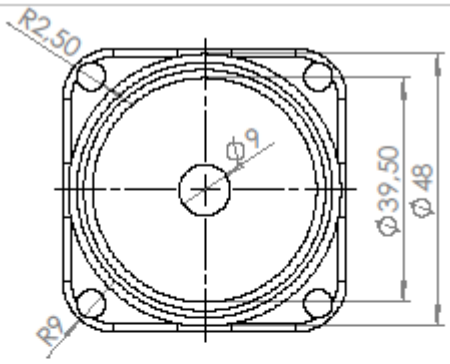
En esta sección se presentan los planos de construcción y la lista de componentes necesaria para la construcción del myRIO demobox.

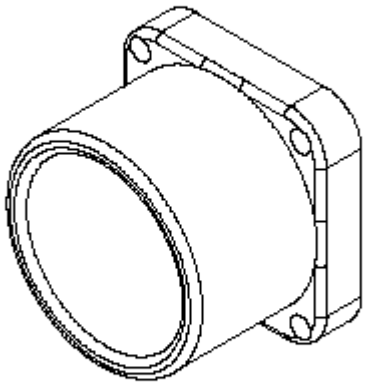
Componente	**Cantidad**
Conector MXP Hembra	2
Conector MXP Macho	2
Servomotor GWS s35+ XF	2
Cámara Microsoft 1080p HD	1
Digilent MXP Breadboard	2
NI Iluminated_PB_NEW	27
Power Source 5V	1
Velcro	1 [m]
Lámina de aluminio	2 [m] x 2[m] x 3[mm]
LED Display Digilent PmodCLS	1
H-Bridge Driver Digilent PmodHB5	2

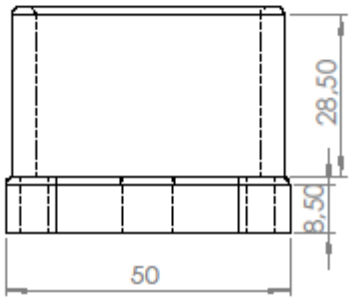




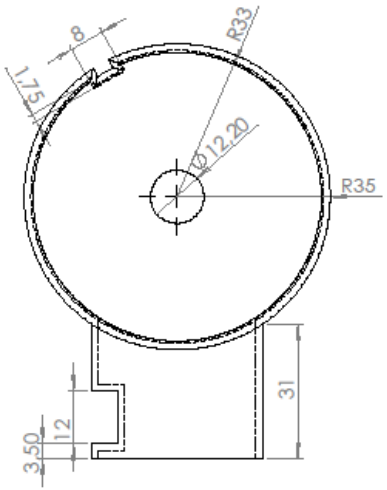
SI NO SE INDICA LO CONTRARIO: LAS COTAS SE EXPRESAN EN MM ACABADO SUPERFICIAL: TOLERANCIAS: LINEAL: ANGULAR:		ACABADO:			REBARBAR Y ROMPER ARISTAS VIVAS	NO CAMBIE LA ESCALA	REVISIÓN
	NOMBRE	FIRMA	FECHA			TÍTULO:	
DIBUJ.							
VERIF.							
APROB.							
FABR.							
CALID.				MATERIAL:		N.º DE DIBUJO	A4
						base circular	
				PESO:		ESCALA:1:2	HOJA 1 DE 1

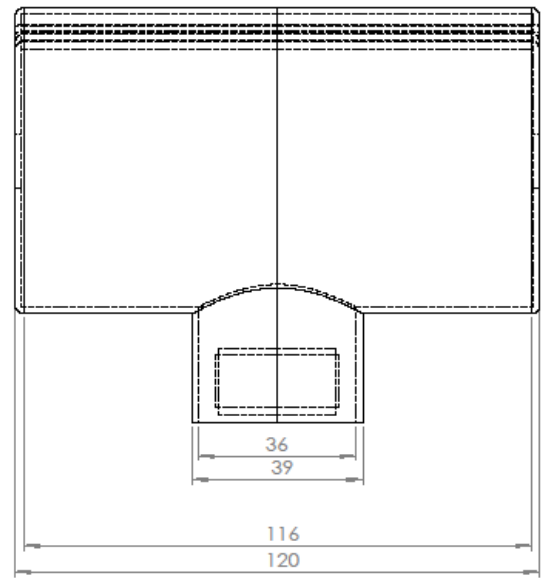


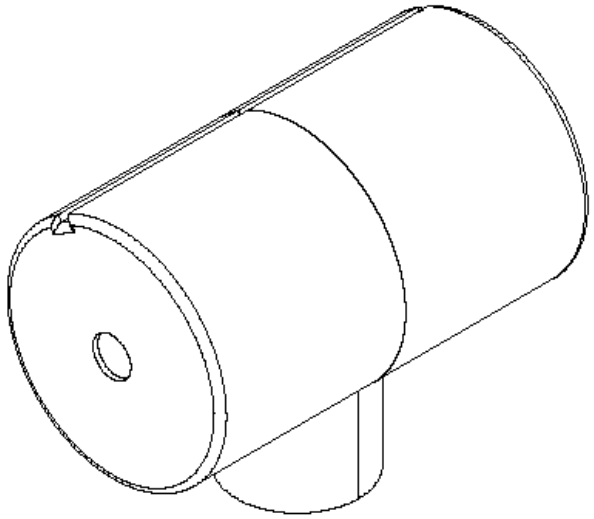




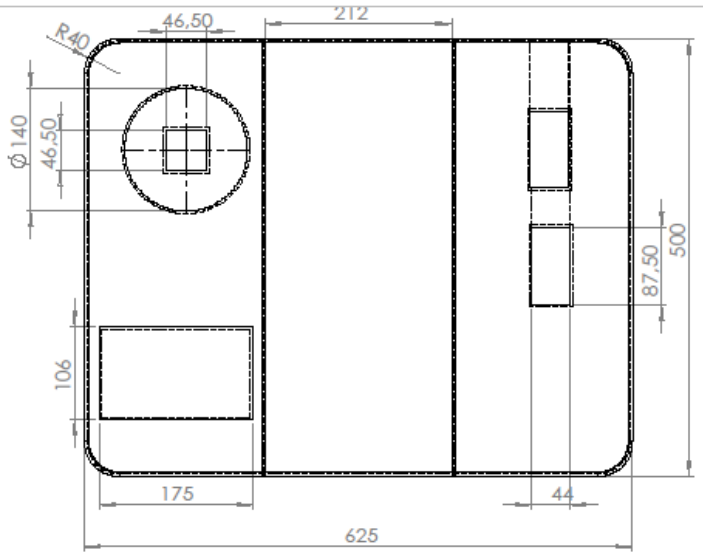
SI NO SE INDICA LO CONTRARIO: LAS COTAS SE EXPRESAN EN MM ACABADO SUPERFICIAL: TOLERANCIAS: LINEAL: ANGULAR:	ACABADO:			REBARBAR Y ROMPER ARISTAS VIVAS	NO CAMBIE LA ESCALA	REVISIÓN
	NOMBRE	FIRMA	FECHA		TÍTULO:	
DIBUJ.						
VERIF.						
APROB.						
FABR.						
CALID.				MATERIAL:	N.º DE DIBUJO: base	A4
				PESO:	ESCALA:1:1	HOJA 1 DE 1

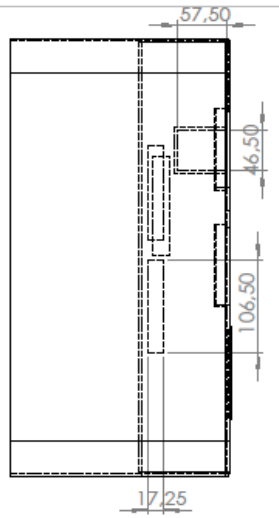


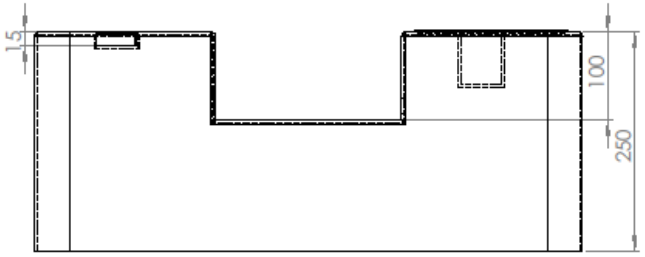




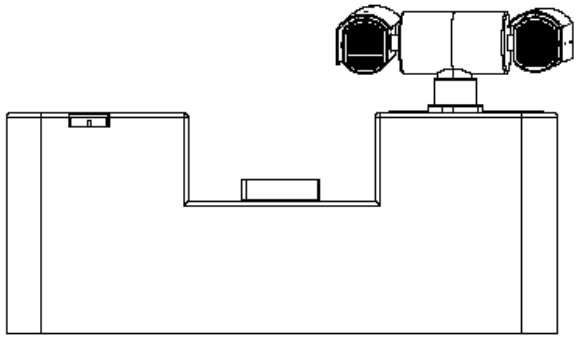
SI NO SE INDICA LO CONTRARIO: LAS COTAS SE EXPRESAN EN MM ACABADO SUPERFICIAL: TOLERANCIAS: LINEAL: ANGULAR:	ACABADO:				REBARBAR Y ROMPER ARISTAS VIVAS	NO CAMBIE LA ESCALA	REVISIÓN
	NOMBRE	FIRMA	FECHA			TÍTULO:	
DIBUJ.							
VERIF.							
APROB.							
FABR.							
CALID.				MATERIAL:		N.º DE DIBUJO	A3
						Main	
				PESO:		ESCALA:1:2	HOJA 1 DE 1

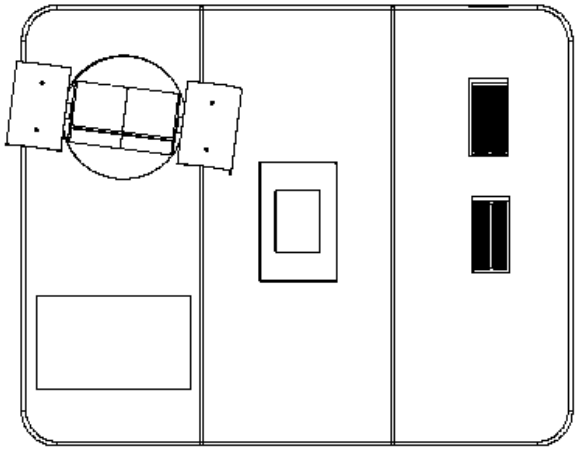


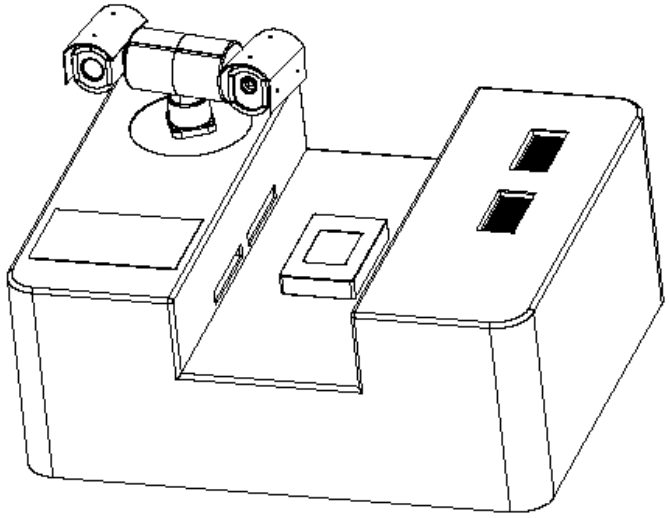




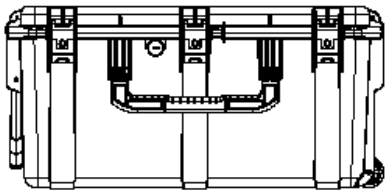
SI NO SE INDICA LO CONTRARIO: LAS COTAS SE EXPRESAN EN MM. ACABADO SUPERFICIAL: TOLERANCIAS: LINEAL: ANGULAR:			ACABADO:			REBARBAR Y ROMPER ARISTAS VIVAS	NO CAMBIE LA ESCALA	REVISIÓN
	NOMBRE	FIRMA	FECHA				TÍTULO:	
DIBUJ.								
VERIF.								
APROB.								
FABR.								
CALID.				MATERIAL:			N.º DE DIBUJO: caja	A3
				PESO:			ESCALA:1:5	HOJA 1 DE 1

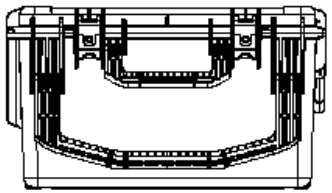


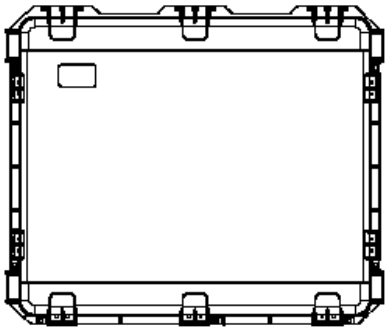


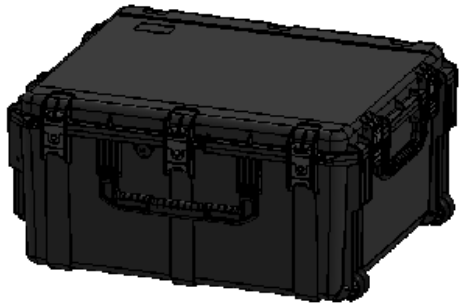


SI NO SE INDICA LO CONTRARIO: LAS COTAS SE EXPRESAN EN MM ACABADO SUPERFICIAL: TOLERANCIAS: LINEAL: ANGULAR:		ACABADO:			REBARBAR Y ROMPER ARISTAS VIVAS	NO CAMBIE LA ESCALA	REVISIÓN
	NOMBRE	FIRMA	FECHA			TÍTULO:	
DIBUJ.							
VERIF.							
APROB.							
FABR.							
CALID.				MATERIAL:		N.º DE DIBUJO	A3
						demobox	
				PESO:		ESCALA:1:10	HOJA 1 DE 1









SI NO SE INDICA LO CONTRARIO: LAS COTAS SE EXPRESAN EN MM ACABADO SUPERFICIAL: TOLERANCIAS: LINEAL: ANGULAR:		ACABADO:				REBARBAR Y ROMPER ARISTAS VIVAS	NO CAMBIE LA ESCALA	REVISIÓN
	NOMBRE	FIRMA	FECHA				TÍTULO:	
DIBUJ.								
VERIF.								
APROB.								
FABR.								
CALID.				MATERIAL:			N.º DE DIBUJO	A3
							estuche	
				PESO:			ESCALA:1:10	HOJA 1 DE 1

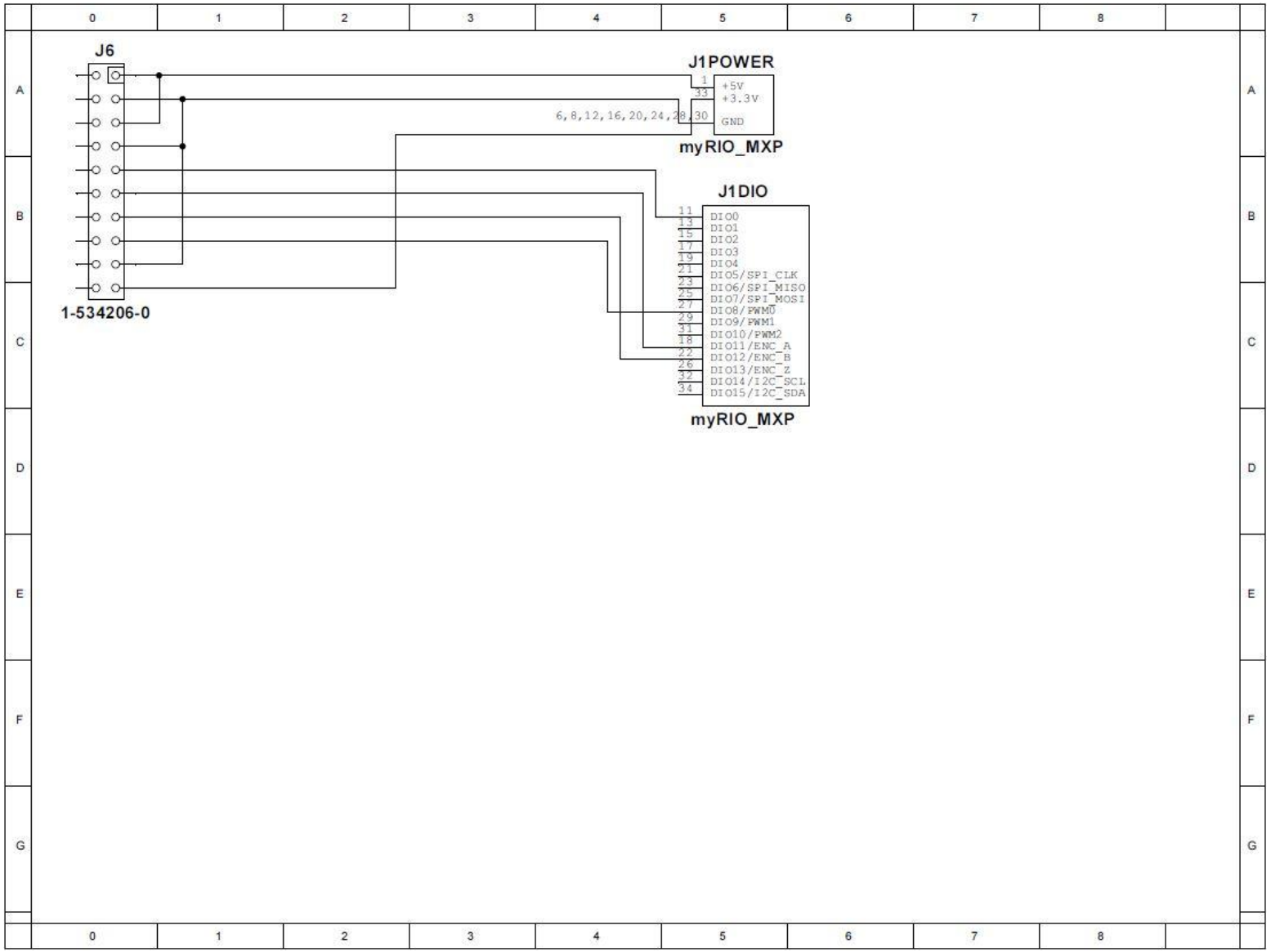


Figura C2: *Esquemático de conexión de la terminal de los motores al myRIO*

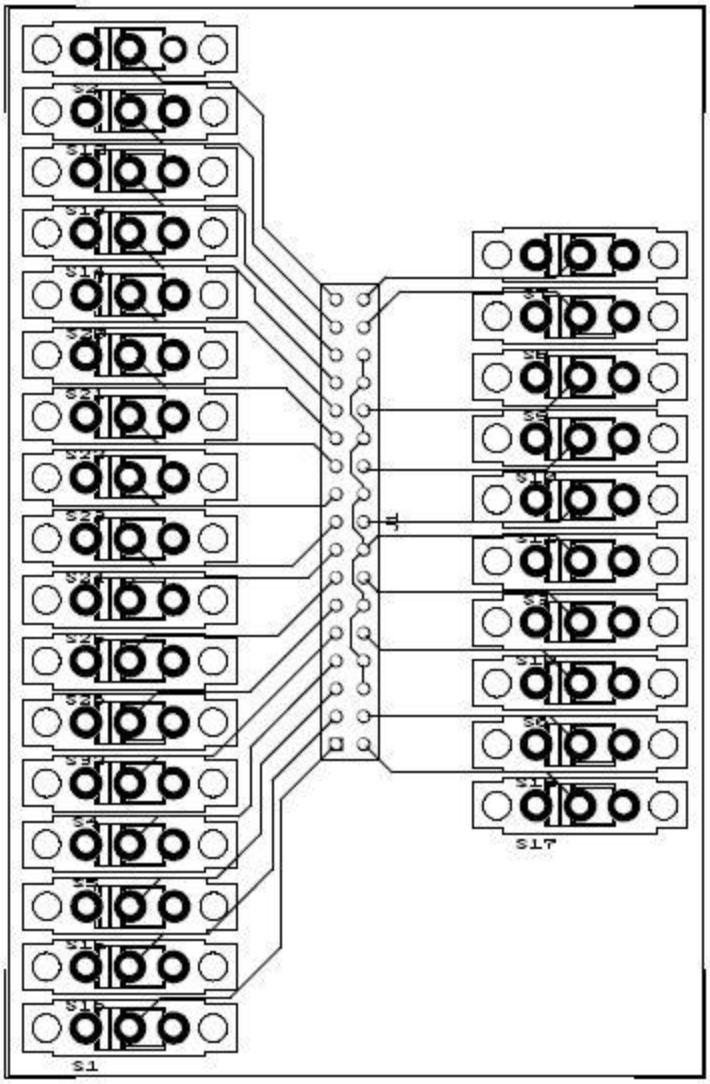


Figura C2: *Circuito PCB de Relays para la conexión y desconexión de los módulos del demobox.*

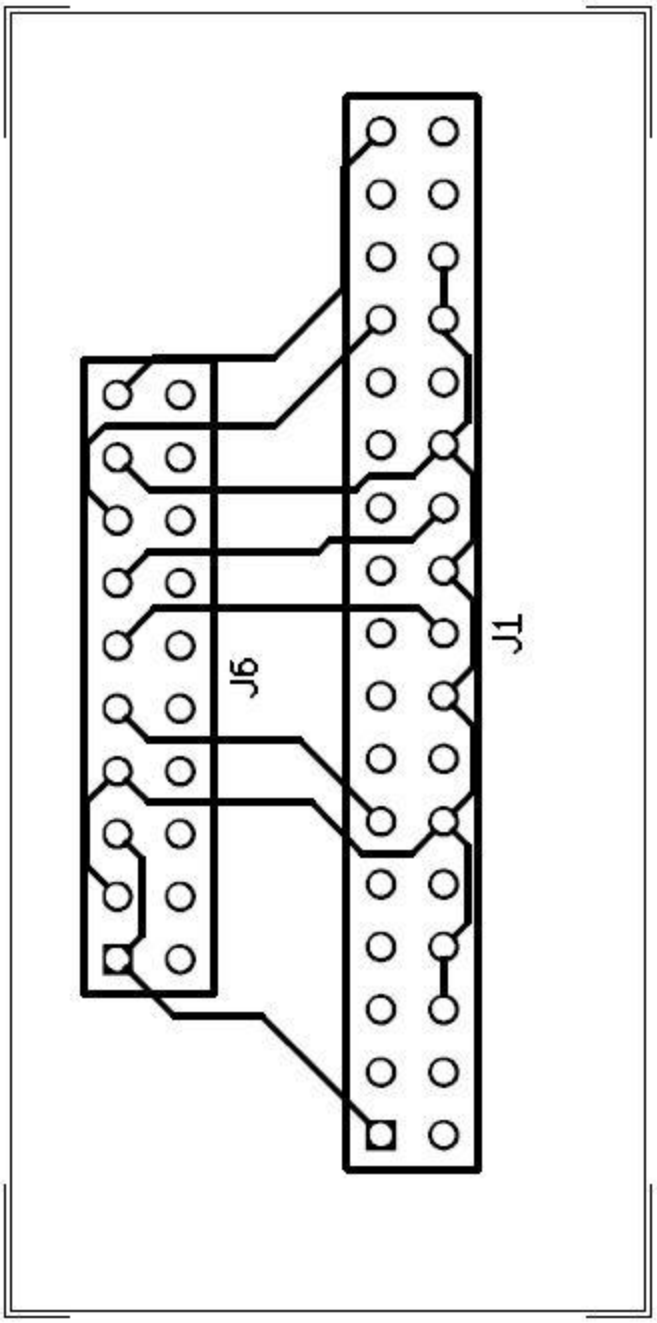


Figura C3: *Circuito PCB de conexión entre la terminal de motores y el MXP del myRIO.*