DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

TESIS PREVIA A LA OBTENCIÓN DEL TITULO DE INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

TEMA: CONTROL DEL ROBOT PIONNER 3D UTILIZANDO UNA FPGA RIO

AUTOR: PARRA ANDRADE GEN EDUARDO

DIRECTOR: ING. ALEJANDRO CHACÓN, MSc.

CODIRECTOR: ING. ALEXANDER IBARRA, MSc

SANGOLQUÍ, SEPTIEMBRE 2015

UNIVERSIDAD DE LAS FUERZAS ARMADAS - ESPE
INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

CERTIFICADO

Ing. Alejandro Chacón MSc.

Ing. Alexander Ibarra MSc.

CERTIFICAN

Que el trabajo "CONTROL DEL ROBOT PIONNER 3D UTILIZANDO UNA FPGA RIO", realizado por el Sr. Gen Eduardo Parra Andrade, ha sido guiado y revisado periódicamente y cumple con normas estatutarias establecidas por la institución, en el Reglamento de Estudiantes de la Universidad de las Fuerzas Armadas - ESPE.

Debido a que se trata de un trabajo de investigación recomiendan su publicación. El mencionado trabajo consta de un documento empastado y un disco compacto el cual contiene los archivos en formato portátil de (pdf). Autorizan al Sr. Gen Eduardo Parra Andrade que lo entreguen al Ingeniero Luis Orozco, en su calidad de Coordinador de la Carrera.

Sangolquí, 03 de Septiembre 2015.

Ing. Alejandro Chacón MSc.
DIRECTOR

Ing. Alexander Ibarra MSc.
CO-DIRECTOR

UNIVERSIDAD DE LAS FUERZAS ARMADAS – ESPE
INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

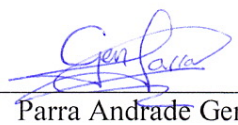
DECLARACIÓN DE RESPONSABILIDAD

Yo, PARRA ANDRADE GEN EDUARDO.

DECLARO QUE:

El proyecto de grado denominado "CONTROL DEL ROBOT PIONNER 3D UTILIZANDO UNA FPGA RIO", ha sido desarrollado en base a una investigación exhaustiva, respetando derechos intelectuales de terceros, conforme las citas que constan al pie de las páginas correspondientes, cuyas fuentes se incluyen en la bibliografía. Consecuentemente este trabajo es de nuestra autoría. En virtud de esta declaración, nos responsabilizamos del contenido, veracidad y alcance científico del proyecto de grado en mención.

Sangolquí, 03 de Septiembre 2015.



Parra Andrade Gen Eduardo

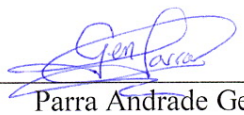
UNIVERSIDAD DE LAS FUERZAS ARMADAS – ESPE
INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

AUTORIZACIÓN

Yo, Parra Andrade Gen Eduardo.

Autorizo a la Universidad de las Fuerzas Armadas – ESPE la publicación, en la biblioteca virtual de la institución del trabajo "CONTROL DEL ROBOT PIONNER 3D UTILIZANDO UNA FPGA RIO", cuyo contenido, ideas y criterios son de nuestra exclusiva responsabilidad y autoría.

Sangolquí, 03 de Septiembre de 2015



Parra Andrade Gen Eduardo

DEDICATORIA

Todo el esfuerzo y dedicación en el desarrollo de este proyecto lo dedico a mi familia, en especial a mis padres y hermana por su apoyo incondicional en todo momento; y por ser el incentivo para seguir adelante con este objetivo.

Gen E. Parra A.

AGRADECIMIENTOS

Agradezco a mis padres, Angel Parra, y Maria Luisa Andrade siendo mis guías durante esta etapa de desarrollo profesional y el apoyo y amor incondicional que me brindado durante toda mi vida.

A mi hermana Karen Parra, por estar siempre presente brindándome su ayuda; a mis amigos y compañeros por todas las experiencias y recuerdos compartidos durante nuestra preparación profesional.

De manera especial un agradecimiento al Ing Alejandro Chacón y Alexander Ibarra quienes aportaron con sus consejos, conocimiento y tiempo para la elaboración del proyecto.

Finalmente a todas las personas que han estado presentes durante el desarrollo del proyecto, y momentos de ocio; quienes no podría terminar de mencionar pero son de gran importancia en mi vida

Gen E. Parra A.

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

El presente proyecto de fin de carrera presenta el desarrollo de un controlador basado en la FPGA RIO para la plataforma robótica móvil Pionner 3D. El controlador cuenta con la capacidad de procesamiento en Tiempo Real; comunicación inalámbrica con un computador de propósito general para el monitoreo, y control de alto nivel de la plataforma robótica. En el desarrollo del proyecto se determinan las características del controlador existente en el robot, y las capacidades del sistema embebido basado en FPGA para el correcto reemplazo de los controladores; el nuevo controlador cuenta con las mismas características operativas de la plataforma original, comprobando su funcionamiento mediante pruebas de precisión, velocidad y auto localización del sistema robótico en un entorno de trabajo. Con los sistemas integrados y en funcionamiento; utilizando las herramientas de LabVIEW y LabVIEW Robotics, se programan algoritmos básicos de navegación para el robot, y se los integra con algoritmos para la planificación de trayectorias dentro de ambientes estructurados que permitirán al robot ejecutar la tarea de navegación.

PALABRAS CLAVE:

- **ROBOT**
- **CONTROLADOR**
- **SISTEMA EMBEBIDO**
- **ALGORITMO**
- **FPGA**
- **TIEMPO REAL**

ABSTRACT

This project present the developing of a controller based on FPGA RIO for the robotics platform Pionner 3D. The controller has Real Time processing capacities; wireless communication with a general purpose computer for monitoring and high level controlling of the robotics platform. In the project development the characteristics of the existing controller in the robot are determined, and the capabilities of embedded system based on FPGA for the correct replacement of the controller; the new controller has the same operating characteristics of the original platform, checking the operation by testing accuracy, speed and auto location of the robotic system in a working environment. Once the systems are integrated and operational using the tools in LabVIEW and LabVIEW Robotics, basic navigation algorithms for robot are programmed, and integrated with algorithms for path planning in structured environments that enable the robot to perform the task of navigation.

KEY WORDS:

- **ROBOT**
- **CONTROLLER**
- **EMBEBBED SYSTEM**
- **ALGORITHMS**
- **FPGA**
- **REAL TIME**

CAPÍTULO I
INTRODUCCIÓN

1.1. Antecedentes

En los laboratorios de Automatización del Departamento de Eléctrica y Electrónica de la Universidad de las Fuerzas Armadas - ESPE, se encuentran, diferentes tipos de robots, entre ellos tres robots móviles de la marca Pioneer de la serie 3D. Desde el 2010 se ha intentado desarrollar un proyecto de tele-operación, cuyo avance se ha limitado por las capacidades de procesamiento del controlador propio del robot, para el desarrollo de este tipo de aplicaciones se requiere de un PC a bordo para su adecuado funcionamiento.

Conociendo las características del controlador existente, es necesario mejorar las limitaciones que tiene; se han revisado algunas alternativas, como el uso de la mainboard Mamba propia de la firma del robot, el uso de un computador a bordo con aplicaciones móviles, y el uso de sistemas embebidos, de tal manera que se mejoren las limitaciones de procesamiento sin perder las características funcionales del robot.

En los últimos años, se han desarrollado sistemas embebidos basados en FPGA que unen en un sistema la potencialidad de un procesador de tiempo real, y las capacidades de los FPGAs, estos sistemas permiten realizar controladores avanzados con herramientas de desarrollo de fácil aplicación, actualmente para el avance de este proyecto se cuenta con un sistema de desarrollo tipo compact rio, el cual se puede emplear como controlador del sistema robótico PIONNER 3D.

1.2.Justificación e Importancia

Para realizar la tele-operación de un robot, se requiere de un sistema en tiempo real para el procesamiento de la información propio-receptiva del robot, capacidades de procesamiento de imágenes, comunicación inalámbrica, y programación de rutas y planes de ruta.

El sistema robótico PIONNER 3D, ha sido estudiado y probado encontrándose que el procesador inicial no cumple con los requisitos necesarios para poder realizar las operaciones de control requeridas, por lo que es necesario incorporar un procesador de características más avanzadas. En la actualidad el DEEE de la Universidad de las Fuerzas Armadas - ESPE cuenta con un sistema de desarrollo FPGA-RIO de marca National Instruments, el cual tiene características de: Comunicación Real Time (RT), sistema FPGA, comunicación Ethernet y su herramienta de programación se la realiza a través de programación gráfica.

El cambio de controlador también obliga a tener que desarrollar un sistema de programación propio del robot, para la definición de su operación, control y los algoritmos de navegación.

Con los cambios realizados se podrá tener una plataforma adecuada para desarrollar en proyectos posteriores la tele-operación del sistema robótico.

1.3.Alcance del Proyecto

La plataforma robótica Pioneer 3D tendrá un nuevo controlador tipo COMPACT-RIO con capacidad de procesamiento en tiempo real, un sistema de control basado en FPGA, comunicación inalámbrica Wi-Fi, programación grafica basada en LabVIEW, con lo cual primero mantendrá las características funcionales que posee con el actual controlador; algoritmos de navegación basados en el uso de LabView Robotics Kit y

adicionalmente características avanzadas para procesamiento de imágenes, que permitirán el desarrollo de proyectos de tele-operación

1.4.Objetivos

1.4.1. General

Desarrollar el sistema de control y de programación de los algoritmos de navegación del Robot móvil Pioneer 3D utilizando un sistema embebido FPGA RIO y una programación gráfica, para aplicaciones de tele-operación.

1.4.2. Específicos

- Determinar las características iniciales del robot, en hardware, software y operación.
- Analizar los cambios físicos necesarios a efectuarse físicamente sobre el robot Pioneer para poder reemplazar el controlador de manera apropiada.
- Realizar las adaptaciones necesarias para emplear el sistema embebido COMPACT-RIO
- Programar los algoritmos de control básicos del sistema para su funcionamiento de acuerdo a las características iniciales del robot
- Implementar los algoritmos de navegación basados en las herramientas de programación del LabVIEW Robotics

CAPÍTULO II
MARCO TEÓRICO

En este capítulo analizaremos los elementos, tanto de hardware y software que se han utilizado en la elaboración de este proyecto; se describirá al robot Pioneer P3DX, detallando sus características operativas y funcionamiento. En cuanto a la FPGA RIO de National Instruments, se describirá sus funciones además del software de programación utilizado LabVIEW, resaltando sobre todo los módulos LabVIEW FPGA y LabVIEW Robotics.

Se explicará además los conceptos básicos de navegación, los tipos de ambientes a los que se expondrá el robot, y los algoritmos que permitirán al robot desplazarse de un punto a otro.

2.1.Hardware

Dentro de los elementos de hardware que utilizaremos en este proyecto tenemos dos componentes principales que son; la plataforma robótica Pioneer P3-DX la cuál será la encargada de ejecutar la tarea navegación; y el controlador, que en este proyecto utilizaremos una FPGA sbRIO 9636 como sistema embebido para la plataforma robótica. A continuación presentamos las características, y funcionamiento de cada uno de estos elementos:

2.1.1. El sistema robótico PIONNER 3D

La plataforma robótica Pioneer DX es una familia de robots móviles enfocada a proyectos de educación, investigación, exhibiciones y desarrollo de prototipos; estos robots son ensamblados de tal manera que son resistentes y con el tamaño adecuado

para ser utilizados dentro de aulas de estudio y laboratorios; esta plataforma contiene los elementos básicos para el sensado y la navegación, por estas características, además de su durabilidad y confiablidad la plataforma Pioneer DX es adecuada para el desarrollo de proyectos investigativos.

Figura 1.- Plataforma robótica Pionner P3-DX
Fuente: [6]

El modelo utilizado en este proyecto, el robot Pioneer P3-DX (referido de aquí en adelante com o P3-DX) tiene dimensiones externas de 44,5 x 40 x 24,5 cm con un peso total de 9kg y puede soportar una carga de hasta 23kg.

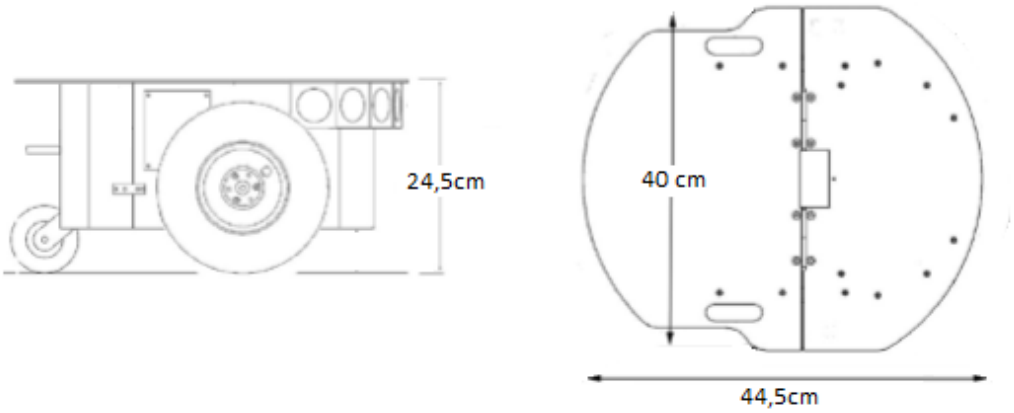


Figura 2.- Dimensiones del robot P3-DX
Fuente: [6]

A continuación se indica una tabla con las características detalladas para el robot Pioneer P3-DX:

Tabla 1
Características del robot P3-DX

Robot Pioneer3-DX	
Dimensiones	44,5 x 40 x 24,5 (largo - ancho - altura)
Peso	9 Kg (con una sola batería)
Carga máxima	23 kg
Batería	
Capacidad	12 V
Carga	252 W/h
Tiempo de ejecución	3-4 h
Tiempo de carga	12 h
Tracción	Diferencial, 2 ruedas motrices y 1 castor
Anillo frontal de US Incluido	8 sonares: 1 a cada lado, 6 delante, 1 cada 20°
Conexión	Puerto serie
Ruedas/Encoder	
encoder pulso/vuelta	500
relación motor/rueda	38,3:1
encoder pulso/milímetro	132
Diámetro de la llanta	19,53cm
Grosor de la llanta	4,74cm
Movilidad	
Dirección	Diferencial
Radio de giro	0cm
Velocidad lineal máxima	1,4 m/s
Velocidad de giro	300°/s

Para entender el funcionamiento del robot, analizaremos sus componentes dividiéndolos en cinco partes principales; cada una de estas partes del robot cuenta con su propia placa de control y conexiones eléctricas independientes; estas partes son:

- Baterías
- Panel de Control de usuario
- Sonares

- Motores y encoders
- Controlador

- **Baterías.-** La alimentación eléctrica para el funcionamiento del robot es realizada mediante un banco de hasta 3 baterías de 12V DC, 7 Amperios/horas; El voltaje suministrado por las baterías pasa a una placa de control que cuenta con un fusible para protección contra sobre corrientes, un relé que controla la alimentación de los demás componentes del robot, y reguladores de 5 y 12 Voltios DC todos ellos incluidos en una placa de control ELC1890 ubicada en la parte posterior del robot.
- **Panel de Control de Usuario.-** Ubicado en el lado izquierdo del robot, cuenta con indicadores y pulsadores para la interacción básica con el robot, además de un puerto RS-232 para la comunicación con un computador

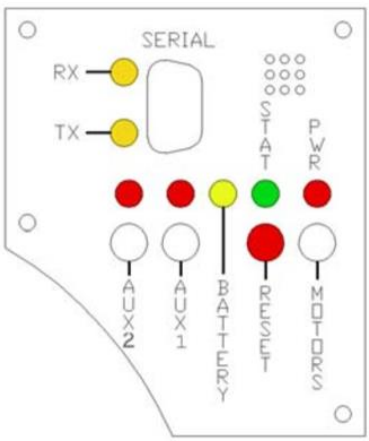


Figura 3.- Panel de Control de Usuario
Fuente: [6]

- **Sonares.-** El robot posee un arreglo de 8 sonares que permiten la detección de objetos para la evasión de obstáculos, reconocimiento de terreno, navegación y localización; cada uno de estos sensores tiene un alcance de

entre 0,1 a 4 metros. Su disposición en el robot permite tener una lectura de los obstáculos localizados en un rango de -90° a 90° relativos a la orientación del robot

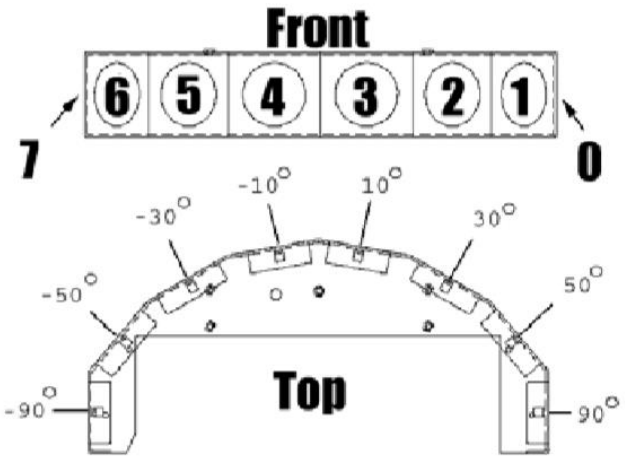


Figura 4.- Arreglo de sonares
Fuente: [6]

El arreglo de sonares realiza la lectura de cada uno de ellos mediante la multiplexación, controlada por la placa de control ELC1003 ubicada en la parte frontal del robot; esta placa además se encarga de proveer la alimentación eléctrica correspondiente a los sensores y una señal digital temporizada indicando la distancia a la que cada sensor ha detectado una obstrucción.

- **Motores y encoders.-** El robot Pioneer P3-DX cuenta con 2 motores DC de alta velocidad y torque, cada motor cuenta con su propio encoder de alta resolución para poder sensar la ubicación y velocidad de las ruedas; además cuenta con una caja reductora que acopla los motores con las ruedas del robot. Los motores funcionan con una alimentación de voltaje de 12V, la cual es controlada a través de un puente H para cada motor; de esta manera mediante pulsos eléctricos es posible controlar el sentido de giro de los motores y ajustar la velocidad haciendo uso de una señal eléctrica modulada por ancho de pulso (PWM).

Tanto los motores como los encoders están conectados a una placa ELC166 el cual contiene el acondicionamiento de voltajes necesarios y la comunicación con el controlador del robot

- **Controlador.-** El controlador es el cerebro del robot, se encarga de adquirir las señales medidas de los sensores, y de controlar a los motores para su navegación a nivel local; A su vez mediante conexión serial a un computador adicional, se completa la arquitectura cliente-servidor, en donde el programa cliente en el computador es encargado de la navegación global del robot; es aquí donde los algoritmos de alto nivel como evasión de obstáculos y planificación de trayectorias son ejecutados.

La plataforma P3-DX cuenta con un controlador ELC1814 Assy#1815 fabricado por MobileRobots el cual mediante buses de datos se comunica con las demás placas internas del robot; la conexión del controlador con las demás placas se indica en la Figura 5 en la que se resumen además las funciones básicas de cada una de ellas:

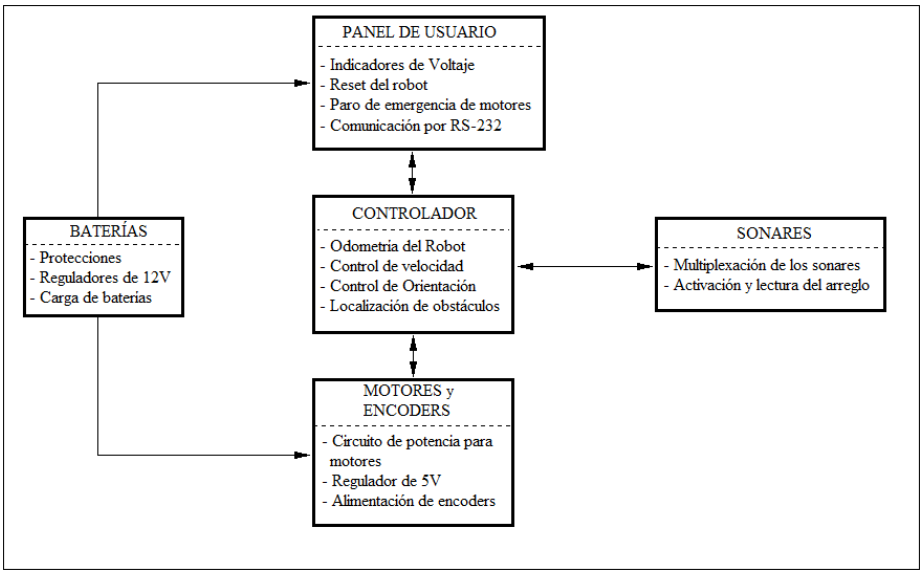


Figura 5.- Interacción componentes del robot

Según sea la aplicación para la cual estemos orientando a nuestro robot, podemos ejecutarlo en uno de los 4 modos permitidos que se explican a continuación:

- Servidor
- Joydrive
- Independiente
- Mantenimiento

En el modo servidor, utilizando el firmware existente del robot llamado ARCOS, podemos crear un software cliente en ARIA para el control de alto nivel del robot, la mayoría de aplicaciones utilizan al robot P3-DX en modo servidor debido a la facilidad y sencillez para acceder a las funciones del robot desde un computador; en este modo el programa cliente se encargará de la lógica de alto nivel del controlador, mientras que el servidor se encargará de la navegación del robot, corrigiendo su velocidad, orientación y posición de acuerdo a la lectura de los sensores correspondientes.

En el modo Joydrive, el robot es controlado de manera manual mediante un joystick, de manera predeterminada, cualquier comando enviado desde el joystick (acelerar, girar, frenar) tiene prioridad ante los comandos enviados por un programa cliente. Si existe algún programa cliente ejecutándose cuando se inicia el modo Joydrive, este solo será reanudado una vez se desconecte el joystick.

Además es posible operar al robot de manera independiente, es decir sin ningún software cliente; para lograrlo es necesario reprogramar la memoria flash del microcontrolador interno. Se lo puede realizar utilizando el software ARSHstub proporcionado por Pioneer. El modo mantenimiento nos permite revertir cualquier cambio realizado reestableciendo los parámetros del controlador a su estado inicial.

2.1.2. El sistema embebido FPGA RIO

Un sistema ON-BOARD o embebido es un sistema basado en microprocesador que es diseñado para el control de un rango de funciones no programables por el usuario final. Aunque brinda opciones al usuario respecto a su funcionalidad, no le permite añadir o modificar las mismas mediante cambio de la programación. Es diseñado para cumplir con tareas específicas de un sistema más amplio, frecuentemente en tiempo real.

Es la parte que se encuentra instalada en el interior de la máquina; interactúa de forma continua con el medio físico a través de sensores y actuadores, encargándose de la adquisición y procesamiento de datos relacionados directamente con el medio.

En un sistema embebido, la mayoría de sus componentes se encuentran dentro de la placa o tarjeta base, de donde viene su nombre "sistema ON-BOARD", entre sus componentes principales se encuentran el procesador, memorias, periféricos, software y algoritmos.

Los programas escritos para los sistemas embebidos son llamados firmware, y son guardados en memoria de solo lectura o memorias flash. Estos programas corren con recursos limitados, poca memoria y con pantallas y teclados pequeños o no existentes; se los programa en lenguajes compatibles con el microprocesador

- **La FPGA RIO de National Instruments**

National Instruments ha desarrollado una tarjeta embebida para control industrial combinando un sistema FPGA (del inglés Field Programmable Gate Array) con módulos de Tiempo Real, de comunicación Ethernet y entradas/salidas analógicas y digitales reconfigurables.

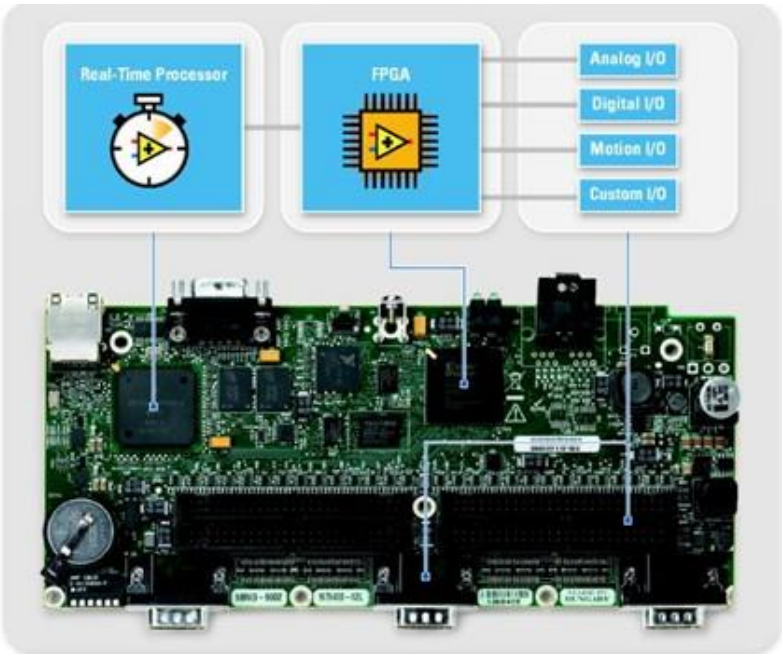


Figura 6.- Componentes de las tarjetas embebidas de National Instruments
Fuente: [8]

El módulo de Tiempo Real incluye un microprocesador para la ejecución de algoritmos de control, y permite operar en rangos determinados de frecuencia de reloj.

El módulo FPGA puede ser utilizado para procesamiento de datos obtenidos a través de los módulos de entradas/salidas y puede operar de manera independiente del módulo Real Time

El módulo Ethernet tiene un puerto de comunicación 8P8C el cual permite conectar la FPGA RIO a un PC para su programación, configuración y/o monitoreo

- **Características**

La FPGA utilizada en este proyecto es una sbRIO 9636 (single board RIO) compatible con la serie C (CompactRIO); su principal diferencia con la serie C es que no posee un conector RMC para su expansión de pines de entradas y salidas. Integra un procesador de Tiempo Real, un chip FPGA reconfigurable y sus entradas/salidas se encuentran disponibles en un solo circuito impreso. Tiene un total de 28 pines digitales disponibles al usuario, 4 salidas analógicas de 16 bits y 16 entradas analógicas de un

canal (8 usadas de manera diferencial) de 16 bits. Opera con un voltaje de 9-30V DC en un rango de temperatura de -40 a 85 °C.

La sbRIO 9636 tiene un puerto Ethernet de 10/100 Mbits/s que puede ser utilizado para su comunicación y programación mediante un computador; además posee puertos RS-232, R-485, CAN, USB y SDHC para el control de dispositivos periféricos.

Tabla 2
Características de la FPGA sbRIO 9636

	sbRIO-9636
Voltaje de operación	9 a 30V
Temperatura de operación	-40 a 85 °C
Procesador	400 MHz
Reloj	40 MHz
Memoria	512 MB
RAM	256 MB
RS-232	2
RS-485	1
CAN	1
USB	1
SDC	1
FPGA	LX45
Entradas/Salidas Digitales	28
Entradas Analógicas	16ch-16bit multiplexados
Salidas Analógicas	4ch-16bit
Conector RMC	0

Fuente: [9]

2.2.Software

En este apartado se tratarán las herramientas de software utilizadas en la programación de los algoritmos de navegación del robot; se explicará de forma breve las características de la programación gráfica, y los módulos de LabVIEW que nos permitirán desarrollar el proyecto.

2.2.1. Programación Gráfica

En los lenguajes de programación gráfica, la interacción se la realiza a través de un cursor (en la mayoría de los casos un mouse), el mismo debe ser capaz cumplir diferentes tareas, como seleccionar elementos, cablear, resaltar texto, etc.

La principal diferencia con la programación basada en texto es el orden en la ejecución de las instrucciones; mientras que los lenguajes basados en texto suelen ser secuenciales, la programación gráfica ejecuta sus instrucciones en paralelo. Entre los programas más conocidos cuya programación se la realiza de manera gráfica tenemos a LabVIEW de National Instruments y Simulink de MatLab.

LabVIEW sigue un modelo de flujo de datos para ejecutar Vis (Instrumentos Virtuales). Un nodo de diagrama de bloques se ejecuta cuando recibe todas las entradas requeridas. Cuando el nodo se ejecuta, produce datos de salida y pasa los datos al siguiente nodo en la trayectoria del flujo de datos. El movimiento de datos a través de los nodos determina el orden de ejecución de los VIs y las funciones en el diagrama de bloques.

En la Figura 7 podemos observar un ejemplo de código para una operación aritmética sencilla; en el caso de programación gráfica se ejecutará primero la suma, ya que dispone de las entradas necesarias y posteriormente la resta, utilizando la salida del bloque suma y la entrada constante 50; en el caso de la programación por texto las operaciones se ejecutan de manera secuencial ejecutando primero la suma debido a

que se encuentra entre paréntesis y restando el valor de 50 para obtener el resultado almacenado en Result.

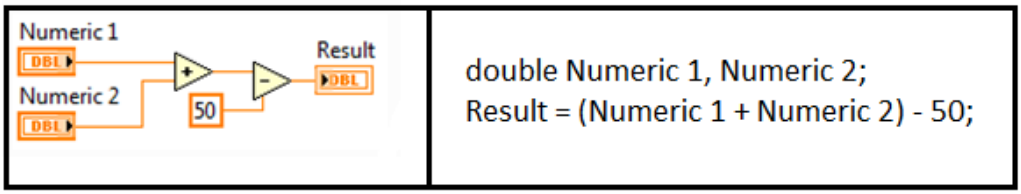


Figura 7.- Comparación de flujo de datos en programación gráfica Vs. Programación basada en texto

En un ejemplo más complejo podemos observar el paralelismo que existe en la ejecución de código de la programación gráfica; en la Figura 8 tanto los bloques de suma y división disponen de las entradas necesarias para su funcionamiento por lo cual se ejecutan en paralelo, la función random no necesita de entradas por lo cual es ejecutada en el primer ciclo de trabajo junto a la suma y división; en un segundo ciclo de trabajo, la función seno (sin(x)) y la función resta reciben las entradas que necesitan y se ejecutan entregando el resultado final a Result y sin(x).

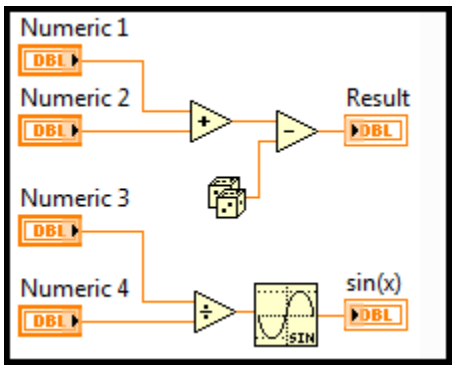


Figura 8.- Ejemplo de paralelismo

Si es necesario que se lleven a cabo acciones en un orden determinado, LabVIEW nos ofrece herramientas como estructuras secuenciales que forzan el orden de ejecución de un código.

La programación de los códigos se la realiza conectando terminales de diferentes bloques. Cada cable puede tener una sola fuente de datos, pero puede ser conectado a

varios terminales de entrada; para su fácil interpretación el cableado maneja un código de colores y grosor que indican el tipo de dato que se está transfiriendo.

Tabla 3
Código de cableado de LabVIEW para distintos tipos de datos

Tipo de dato	Escalar	Arreglo 1D	Arreglo 2D	Color
Número Doble				Naranja
Número Entero				Azul
Booleano				Verde
String				Rosado

Fuente: [8]

Además para el desarrollo de aplicaciones especializadas LabVIEW nos facilita módulos de desarrollo como LabVIEW FPGA, Robotics, RealTime, PID y otros más que incluyen sus propios bloques e instrucciones que facilitan la programación de algoritmos de control. Para este proyecto utilizaremos tanto el módulo FPGA y Robotics que se detallan a continuación:

2.2.2. Herramienta de desarrollo LabVIEW FPGA

El módulo de LabVIEW FPGA de National Instruments extiende las capacidades de desarrollo gráfico de LabVIEW a FPGAs. Con este módulo, se puede crear sistemas de medición y control personalizados en hardware sin tener que diseñar en lenguaje descriptor de hardware o a nivel circuito.

El desarrollo tradicional para sistemas basados en FPGA requiere el uso de herramientas de software de bajo nivel y lenguajes de descripción de hardware (HDLs). LabVIEW FPGA ofrece un enfoque de programación gráfica que simplifica la tarea de conectar a E/S y comunicar datos, mejorando enormemente la productividad del diseño y reduciendo el tiempo al mercado.

Tiene integradas herramientas de simulación y herramientas de depuración, para detectar cualquier error de implementación antes de la compilación. El código puede ser simulado y depurado con funciones de depuración de LabVIEW básicas como ejecución resaltada, puntos de interrupción y puntas de prueba.

Debido a que las FPGA de National Instrumentes operan con un procesador de tiempo real, LabVIEW FPGA cuenta con instrucciones compatibles con el procesador, incluidos bucles temporizados, pausas sincronizadas mediante pulsos de reloj y operaciones básicas ejecutables en un solo ciclo de procesador.

Además incluye las herramientas necesarias para establecer la comunicación con un computador, desde donde se pueden configurar las características de red de la FPGA, configurar los parámetros de ejecución de código y compartir datos para el procesamiento utilizando módulos adicionales de LabVIEW o simplemente para realizar un monitoreo de las variables de la FPGA utilizándolo como interfaz gráfica.

Entre las herramientas para la configuración de red y conexión con la FPGA se cuenta con el software NI LabVIEW RIO Evaluation Setup, que establece una primera conexión con el ordenador y configura los parámetros necesarios para su posterior programación de manera automática.

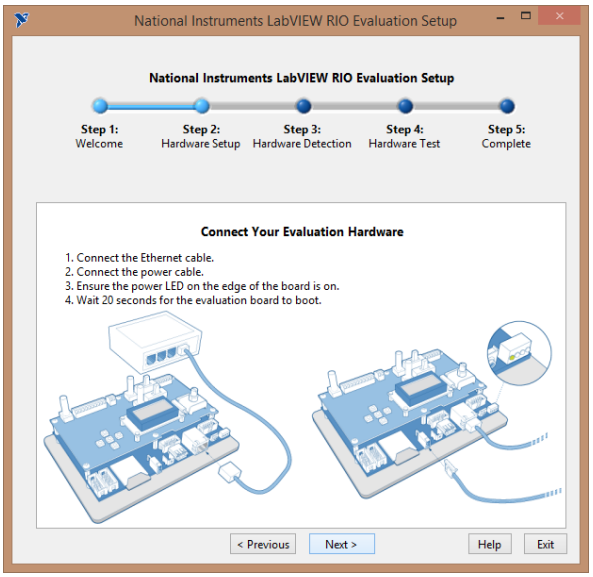


Figura 9.- Captura de NI LabVIEW RIO Evaluation Setup

Cuenta también con herramientas especializadas para la configuración manual de la tarjeta como el NI MAX que nos permite detectar todos los dispositivos de National Instruments existentes en la red, así como probar los componentes de cada dispositivo y cambiar los parámetros de conexión como su IP, máscara de red y Gateway.

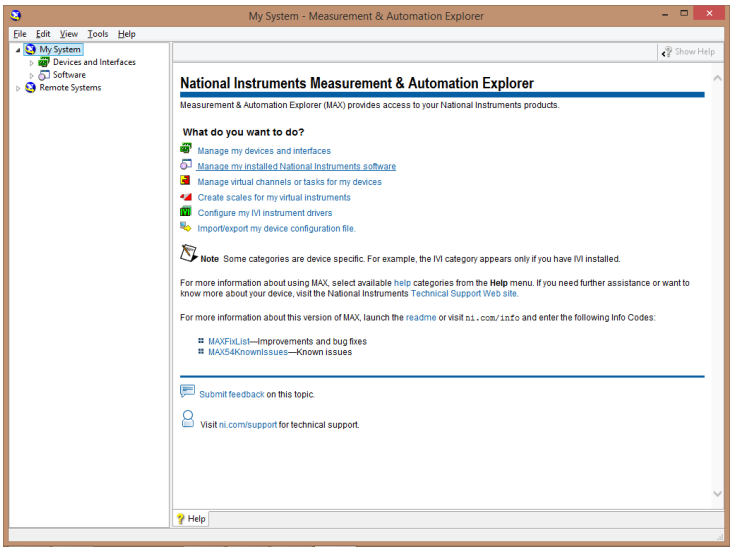


Figura 10.- Captura del programa NI MAX

2.2.3. LabVIEW Robotics

El módulo de robótica, LabVIEW Robotics ofrece herramientas de diseño y programación de todo tipo de robots; con su lenguaje de programación gráfico facilita la programación de sensores y actuadores; proporcionando un alto nivel de abstracción en cuanto a comunicación con los sensores, evasión de obstáculos, planificación de trayectorias, cinemática y más.

El módulo de robótica se conecta con el entorno de desarrollo de LabVIEW ampliando su biblioteca con algoritmos básicos para la operación inteligente, navegación y percepción del robot; por lo cual es una herramienta ideal para la generación de prototipos de tipo:

- Vehículos autónomos y semi-autónomos incluyendo sistemas agrícolas y militares
- Plataformas de robots de rescate
- Vehículos submarinos y aéreos
- Robots personales y de servicio
- Dispositivos médicos de robótica
- Robots académicos y de investigación

Además, esta plataforma es capaz de ejecutarse en una variedad de procesadores incluyendo sistemas comerciales en tiempo real, y basados en FPGA así como microprocesadores y microcontroladores personalizados. Provee además herramientas de simulación para el testeo de prototipos, algoritmos y códigos en un ambiente controlado incluso si no se dispone de un robot real.

Es con estas herramientas de software con las que se desarrolla el presente proyecto, utilizando los algoritmos existentes en LabVIEW Robotics y FPGA para controlar la navegación del robot Pioneer P3-DX utilizando como controlador a la FPGA sbRIO 9636

2.3.Navegación

Para un robot móvil, el proceso de navegación incluye la planificación de una trayectoria libre de obstáculos dentro de un área de trabajo para poder desplazarse desde una posición a otra. Para ello el robot debe ser capaz de cumplir tres tareas principales. Primero el robot debe conocer su localización en todo momento; para ello recurrimos a sensores que le permitirán emplear sistemas de posicionamiento para determinar su posición en el ambiente de trabajo. Su segunda tarea a cumplir es identificar el objetivo al que debe llegar, el mismo que será limitado por los grados de libertad del robot y el ambiente de trabajo. Finalmente el robot debe planificar una

trayectoria desde su ubicación inicial hacia la posición del objetivo de tal manera que se eviten los obstáculos existentes en el entorno de trabajo.

Los ambientes de trabajo que existen pueden ser de dos tipos, estructurados o no estructurados. Los ambientes o entornos estructurados que permanecen estáticos y no sufren cambios con el transcurso del tiempo, y los entornos no estructurados que son dinámicos y pueden presentar cambios inesperados en los elementos que lo conforman. Para cada tipo de entorno existen distintos enfoques en la tarea de navegación sin embargo deben cumplir las mismas etapas:

2.3.1. Etapas de la navegación

Para que el robot pueda desplazarse en un entorno de trabajo sin que se produzcan colisiones con obstáculos se deben cumplir las siguientes etapas:

- **Percepción del entorno:** El robot reconoce el entorno donde se encuentra; ya sea utilizando un mapa del ambiente de trabajo, o sensores, el robot es capaz de percibir los obstáculos que se encuentran a su alrededor.
- **Planificación de la trayectoria:** El robot genera una sucesión de posiciones intermedias entre su ubicación y la meta de manera que se evadan los obstáculos presentes en el entorno de trabajo.
- **Seguimiento de la trayectoria:** Mediante la activación de sus actuadores, el robot se desplaza sobre la trayectoria controlando su velocidad y orientación desplazándose sobre la trayectoria planificada.

2.3.2. Enfoques de la navegación:

Para la navegación del robot, podemos utilizar uno de los siguientes enfoques, ya sea que nos estemos basando en la percepción del robot a través de sus sensores. O que tengamos disponible un mapa del entorno de trabajo

- **Navegación basada en la percepción**.- El robot conoce su ubicación y la localización de obstáculos que estén interactuando con sus sensores; por ello

la planificación de trayectoria debe ejecutarse siempre que un nuevo obstáculo sea identificado. Su tiempo de respuesta debe ser rápido para evitar colisiones. El esquema de esta navegación se indica en la Figura 11

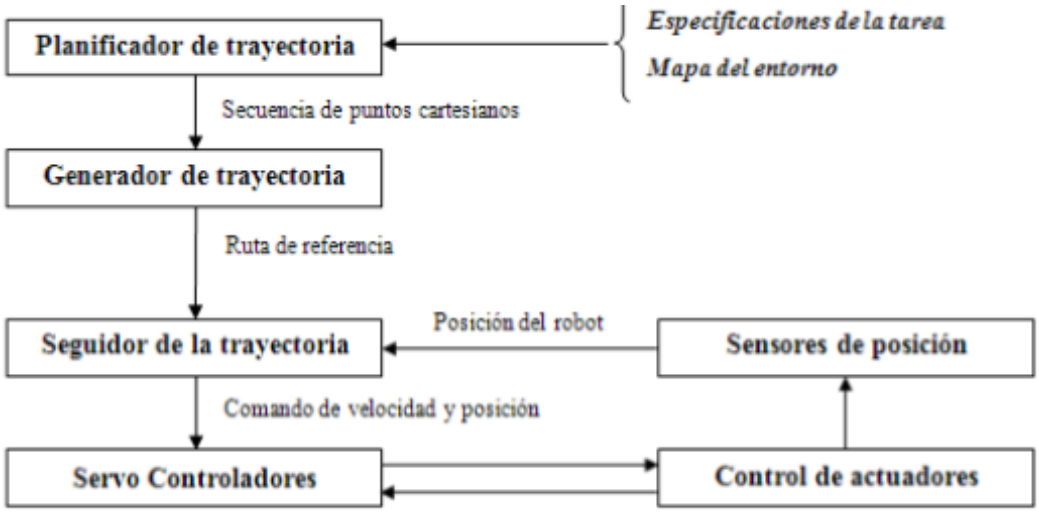


Figura 11.- Navegación libre de mapa
Fuente: [15]

- **Navegación basada en mapa**.- El mapa puede ser conocido por el robot de manera previa, o construirse en base a la percepción del robot; la trayectoria es planificada en base al mapa desde el punto inicial hasta la meta; sin embargo no considera los cambios que se puedan generar en el entorno de trabajo. Debido a esto suelen implementarse subrutinas de evasión que eviten colisiones con obstáculos no considerados de manera previa. En la Figura 12 se indica el esquema de navegación basada en mapa.

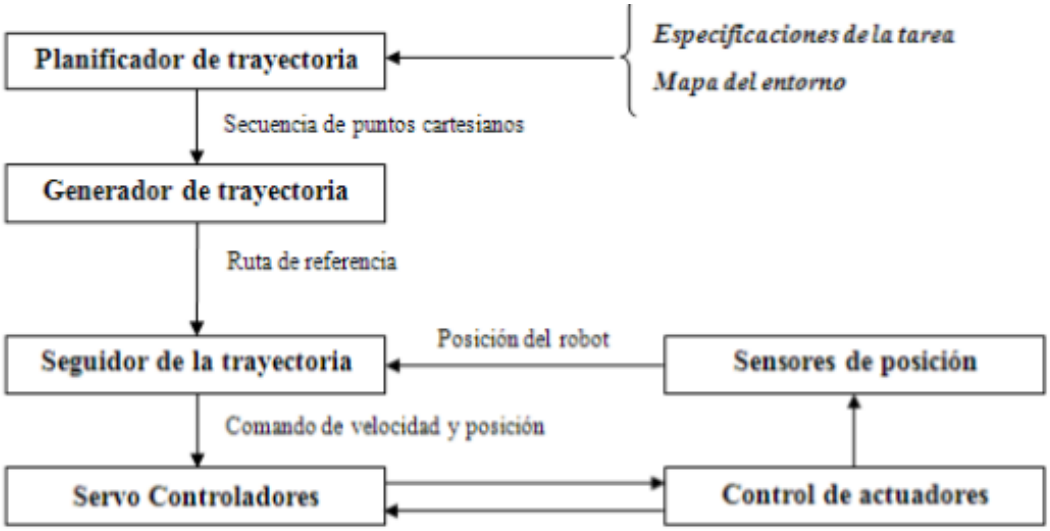


Figura 12.- Navegación basada en mapa
Fuente: [15]

2.3.3. Algoritmos de navegación

Existen una gran cantidad de algoritmos para robots móviles ya sean basados en mapa o en la percepción del robot; a continuación describiremos tres de los algoritmos comúnmente utilizados para la navegación de robots móviles:

- **Algoritmo A-Star**

Dentro de los algoritmos basados en mapa, el más sencillo de los algoritmos para la planificación de trayectorias es el A-Star.

La primera consideración de este algoritmo es dividir al entorno de trabajo en celdas de dimensión constante; estas celdas podrán ser definidas como libres u ocupadas (con obstáculo), además de identificar una celda inicial o de partida, y una celda final o de meta para la ejecución del algoritmo; en la Figura 13 se presentan las celdas ocupadas con color rojo, la celda de partida como A, y la celda de meta como B.

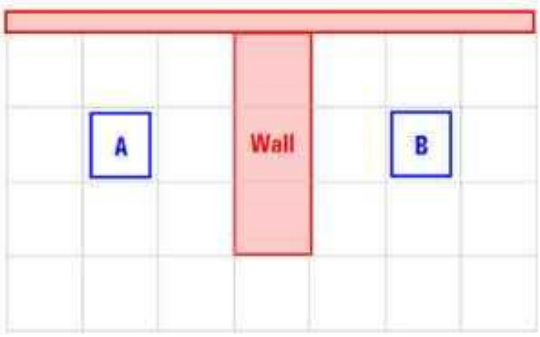


Figura 13.- Entorno de trabajo para el algoritmo A-Star
Fuente: [10]

Las celdas blancas en la figura indican todos los espacios por los cuales el robot puede navegar; el objetivo del algoritmo es encontrar la ruta (conjunto de celdas) que permitan al robot ir desde la celda de partida hasta la meta, con el menor esfuerzo posible.

Para escoger que movimiento realizar, el algoritmo asigna un costo a cada celda adyacente, según su costo movimiento (G), y la distancia mediada hasta la celda de meta (H), se escogerá con ello la celda o celdas que presenten el menor costo total (F) que se obtiene sumando los costos G y H. Para la primera iteración del algoritmo los costos asignados al mapa de la Figura 14 quedan de la siguiente manera:

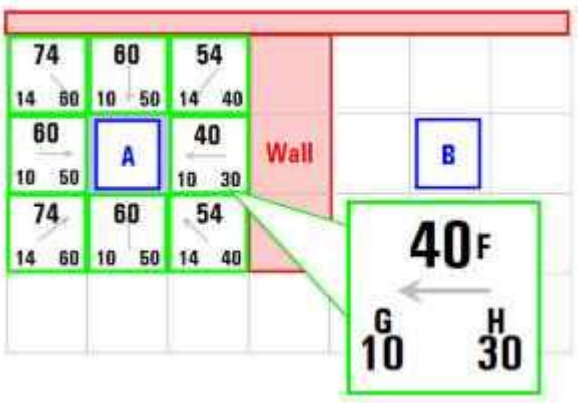


Figura 14.- A-Star; Asignación de costos a las celdas
Fuente: [10]

El algoritmo escoge la celda con el menor peso, y repite la asignación de costos a las nuevas celdas adyacentes, ignorando siempre aquellas celdas que se encuentren ocupadas por un obstáculo.

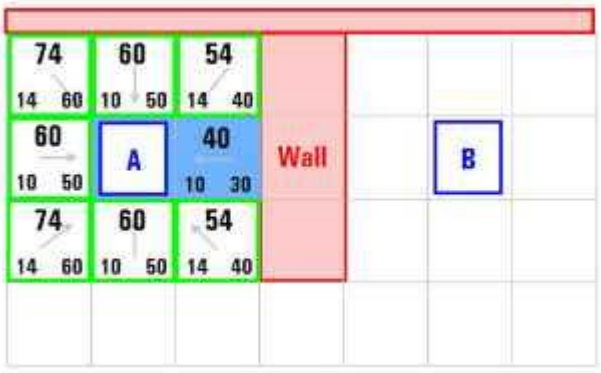


Figura 15.- A-Star; primera iteración
Fuente: [10]

El proceso se repite, y a medida que la ruta se amplia, más celdas con un mismo peso serán encontradas, para ello el algoritmo se ejecuta para cada una de ellas hasta que el algoritmo llegué a la celda de meta

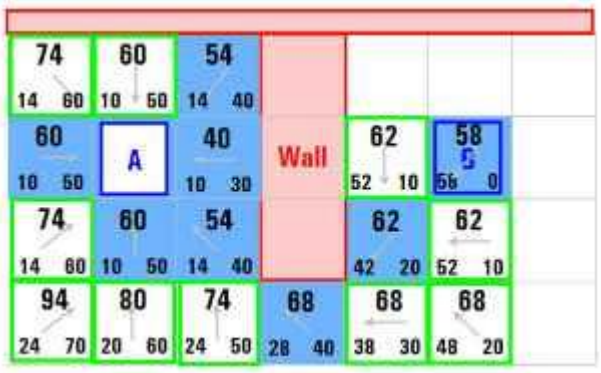


Figura 16.- A-Star; ejecución del algoritmo
Fuente: [10]

Una vez que la meta ha sido alcanzada se realiza la regresión desde la meta a la partida, obteniendo todas las celdas que pertenecen a la trayectoria del menor costo.

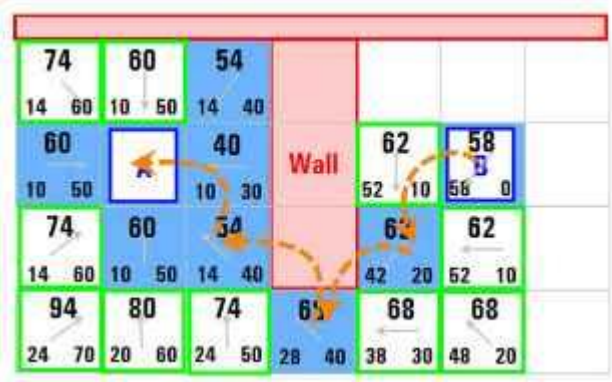


Figura 17.- A-Star; regresión desde la meta
Fuente: [10]

- **El algoritmo de Dijkstra**

El algoritmo de Dijkstra consiste en encontrar el camino más corto, desde un punto de referencia local o inicial hasta una meta definida, A diferencia del algoritmo A-Star genera puntos de referencia (nodos) y vértices aleatorios desde la partida hasta la meta y busca la trayectoria más corta basado únicamente en la distancia recorrida entre cada uno de los nodos.

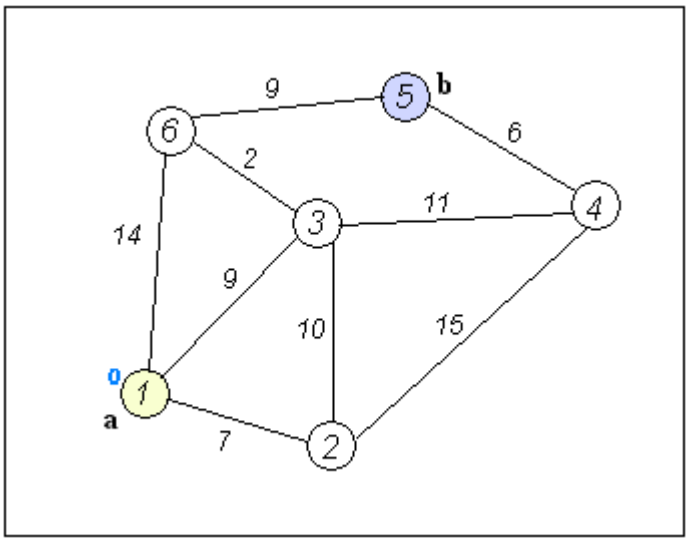


Figura 18.- Asignación de nodos para el algoritmo de Dijkstra
Fuente: [12]

El cálculo de la trayectoria óptima se la realiza siguiendo la misma lógica que el algoritmo A-Star, considerando que en lugar de celdas adyacentes, en este algoritmo tenemos conexiones entre nodos, y la distancia entre ellos será considerada como el costo total de movimiento.

- **Diagramas de Voronoi**

Para evitar que el algoritmo de Dijkstra genere nodos sobre posiciones inalcanzables en el entorno de trabajo del robot, podemos utilizar diagramas de Voronoi para la generación de nodos y vértices, los mismos que asegurarán que ninguna de las trayectorias que se genere a partir del algoritmo pase sobre un obstáculo existente conocido.

Los diagramas de Voronoi son una de las estructuras fundamentales dentro de la Geometría Computacional, de alguna forma ellos almacenan toda la información referente a la proximidad entre puntos.

Suponemos dado un conjunto finito de puntos en el plano P = {p1,..., pn} (con n mayor o igual que dos) y a cada pj le asociamos aquellos puntos del plano que están más cerca o igual suya que de cualquier otro de los pi con i distinto de j. Todo punto del plano queda así asociado a algún p_i, formándose conjuntos que recubren a éste. Existirán puntos que disten lo mismo de dos elementos de P y que formarán la frontera de cada región. Los conjuntos resultantes forman una teselación del plano, en el sentido de que son exhaustivos (todo punto del plano pertenece a alguno de ellos) y mutuamente excluyentes salvo en su frontera. Llamamos a esta teselación Diagrama de Voronoi plano (denotado Vor(P)). A cada una de las regiones resultantes las llamaremos regiones de Voronoi. Los puntos del conjunto reciben el nombre de generadores del diagrama. En la Figura 19 se muestra el diagrama de Voronoi de una nube puntos en el plano.

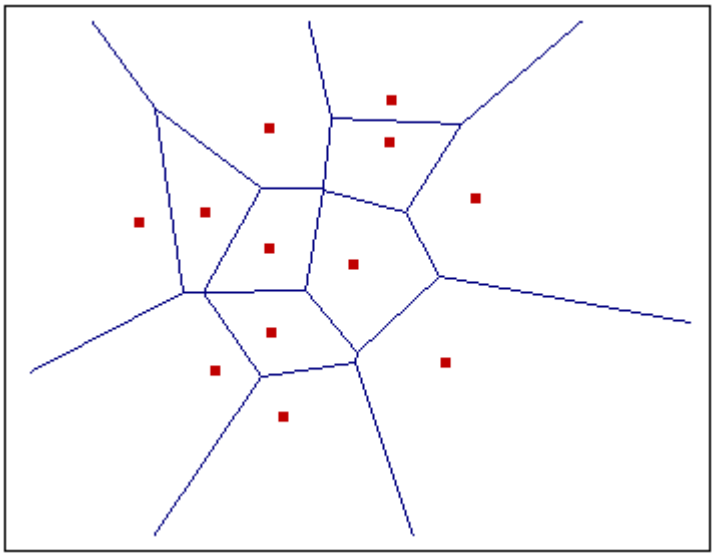


Figura 19.- Diagrama de Voronoi

CAPÍTULO III
SISTEMA DE CONTROL ON-BOARD

Como se mencionó en el capítulo anterior, un sistema ON-BOARD o embebido es diseñado para cumplir con tareas específicas de un sistema más amplio. En este capítulo definiremos las tareas que nuestro sistema embebido basado en FPGA ejecutará para que el robot P3-DX pueda realizar tareas de navegación; con las tareas del sistema embebido definidas, determinaremos las modificaciones necesarias a realizar para la adaptación del sistema embebido al sistema robótico disponible.

3.1.Definición de características

El sistema embebido que implementaremos en este proyecto, debe ser capaz de ejecutar las mismas tareas del controlador original del robot P3-DX; es decir debe ser capaz de ejecutar el control de bajo nivel del robot que incluye:

- Control de velocidad
- Control de orientación
- Odometría del robot
- Detección de obstáculos

Además incluirá un puerto de comunicación con un computador de propósito general externo que se dedicará a ejecutar algoritmos de alto nivel.

El sistema una vez instalado deberá interactuar con los sensores y actuadores disponibles en el robot para poder ejecutar sus tareas dedicadas a la vez que proporciona la información sobre la ubicación y velocidad del robot al computador de propósito general; un esquema de la arquitectura planteada se indica en la Figura 20.

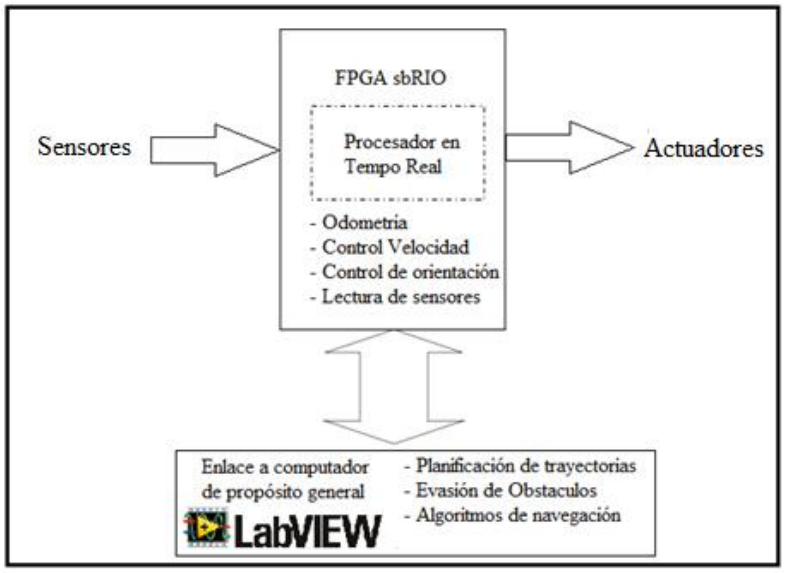


Figura 20.- Arquitectura del sistema de control

3.2.Programación del sistema embebido

Antes de definir los algoritmos de las tareas del sistema embebido es necesario que sea capaz de controlar los sensores y actuadores del robot;

3.2.1. Control de los componentes del robot

El robot posee sus propios sensores que le permiten realizar la detección de obstáculos, y conocer su ubicación y velocidad, el funcionamiento, diagramas de flujo y consideraciones necesarias para la programación de estos componentes y sus motores se indican a continuación; el código programado en LabVIEW se encuetra expuesto en el Anexo VI.- Programación del controlador:

- **Arreglo de sonares.-** Cada uno de los sensores tiene su propia dirección binaria de acceso, y realiza la medición disparando una onda de sonido ultrasónica, y midiendo el tiempo que demora la onda en retornar al sensor a través del eco producido. La tarjeta de control existente en el robot multiplexa los sonares e inhibe las señales de activación para evitar lecturas con distancia nula. El diagrama de flujo para la lectura de los señores se indica a continuación.

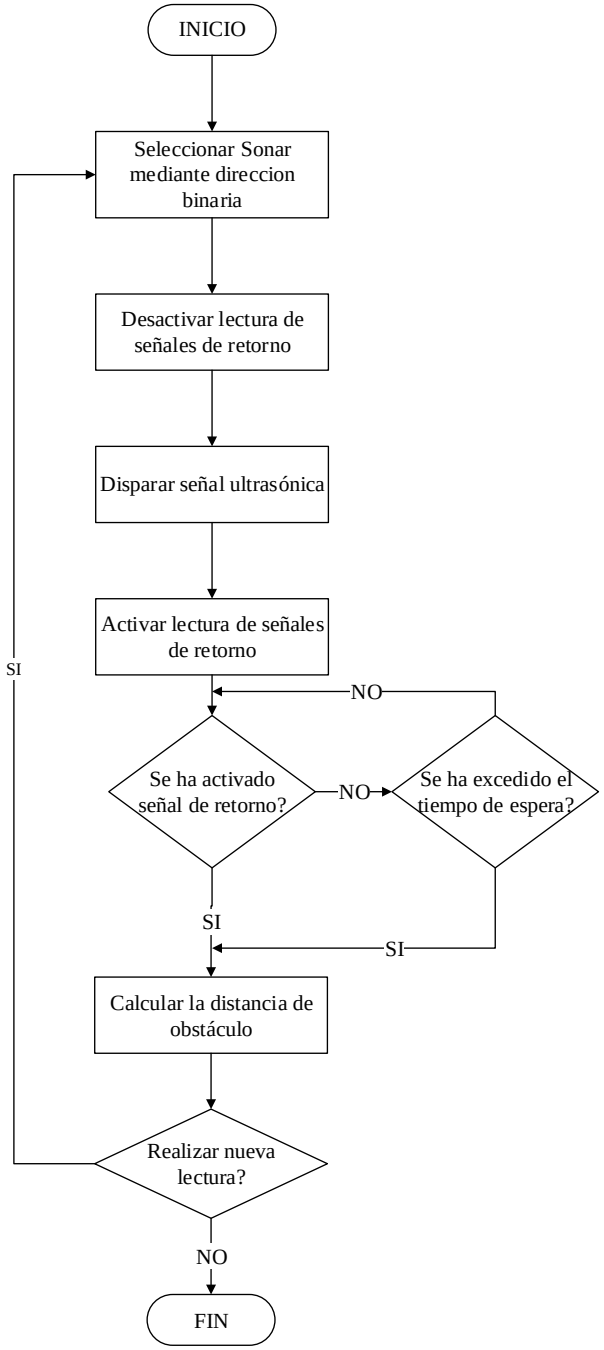


Figura 21.- Diagrama de flujo del arreglo de sonares

- **Encoders.-** Los motores del robot P3-DX tienen acoplados a su eje encoders de alta precisión para el control de velocidad y posición; Estos encoders operan mediante la lectura de dos canales digitales como se indica en la Figura 22:

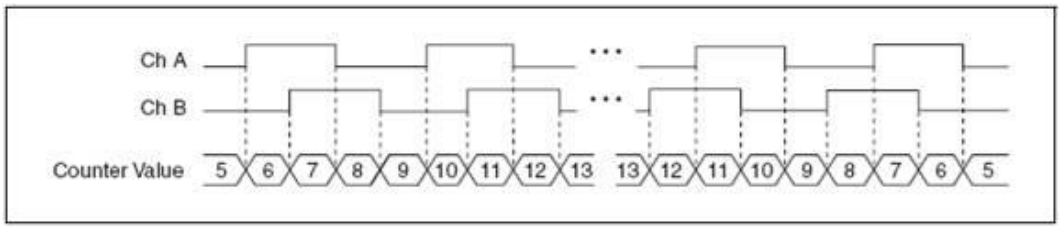


Figura 22.- Comportamiento esperado del contador programado
Fuente: [7]

Con los estados presentados por los Canales A y B definimos las funciones lógicas (1) y (2) pata conocer si el encoder realiza una lectura de movimiento de los motores, y en qué sentido de giro detecta este movimiento.

$$Movimiento = B \oplus B_{siguiente} + A \oplus A_{siguiente} \quad \textbf{(1)}$$

$$Sentido = A \oplus B_{siguiente} \quad \textbf{(2)}$$

La programación del contador será tal que aumente si se detecta un giro en sentido horario, y que se reduzca si el sentido es anti horario; este comportamiento se define por la máquina de estados indicada en la Figura 23:

- **Motores.-** Los motores del robot P3-DX funcionan mediante la activación de una señal PWM, y utiliza los encoders para sensar la velocidad de giro; mediante estas dos señales podemos implementar un control de velocidad digital que nos permita manipular los motores del robot.

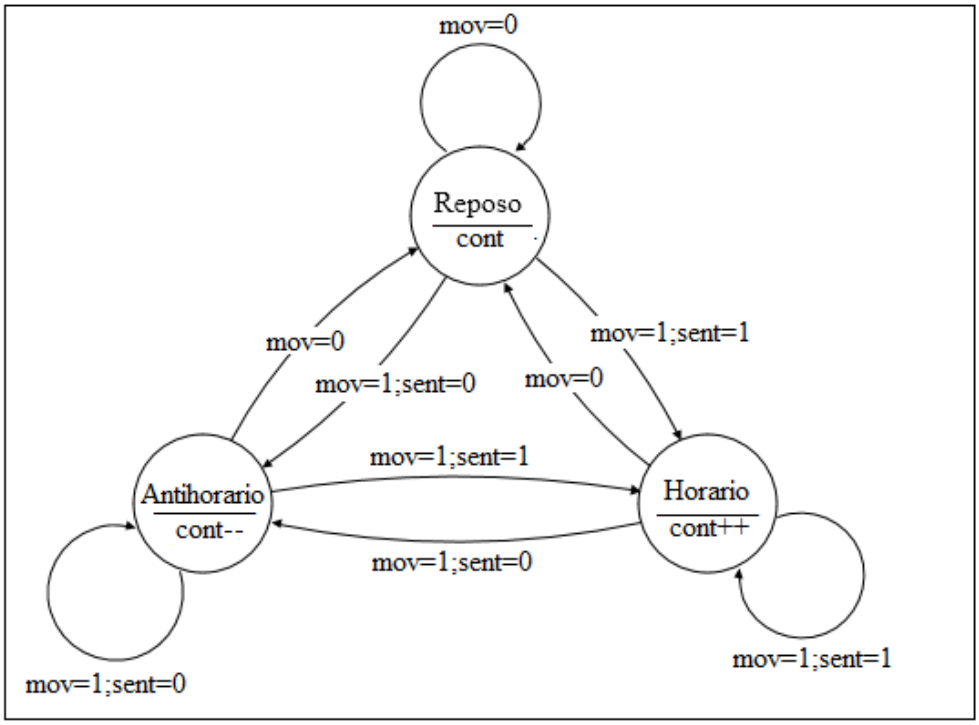


Figura 23.- Máquina de estados para el encoder

Los motores del robot pueden ser modelados mediante una función de transferencia de primer orden, definida por:

$$Motor(s) = \frac{velocidad(s)}{PWM(s)} = \frac{0.0917}{1 + 0.0015s} \quad \textbf{(3)}$$

En donde la velocidad es medida por los encoder en ticks/ms, y el PWM se define en ciclos de trabajo activos (entre 0 y 2000)

El control implementado en este proyecto es un PID (diseñado en el ANEXO I) que gracias a sus características nos permite tener un error en estado estacionario nulo, y rápidos tiempo de respuesta frente a cambios de velocidad debido a su acción integral.

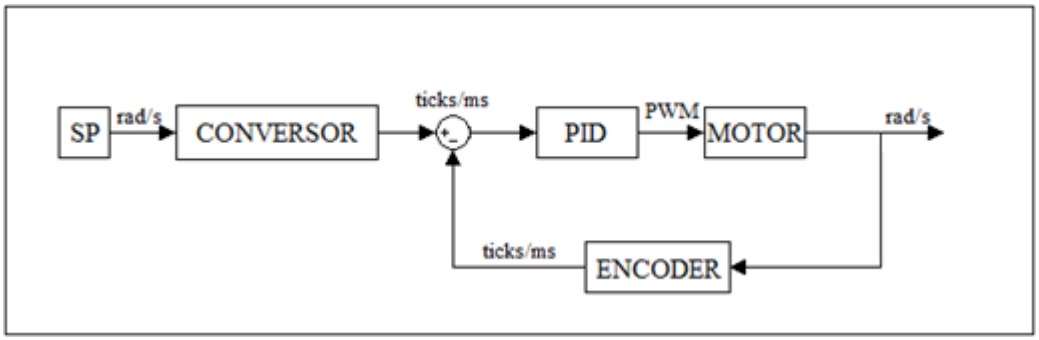


Figura 24.- Diagrama de control para los motores

En la Figura 24 presentamos el diagrama de control implementado, para los motores, en donde las ganancias del controlador PID (calculadas en el ANEXO I) son:

Tabla 4
Ganancias para el control PID de los motores

Kp	0.25
Ki	800
Kd	0.03

3.2.2. Control de velocidad y orientación

Para el control de la velocidad y orientación del robot utilizaremos el modelo cinemático del robot, definido en la ecuación (4) (validación del modelo en el Anexo II, utilizando el esquema de la Figura 25) el cual nos permite manipular la velocidad lineal y angular del robot mediante las velocidades independientes de los motores como se muestra en las ecuaciones (5) y (6)

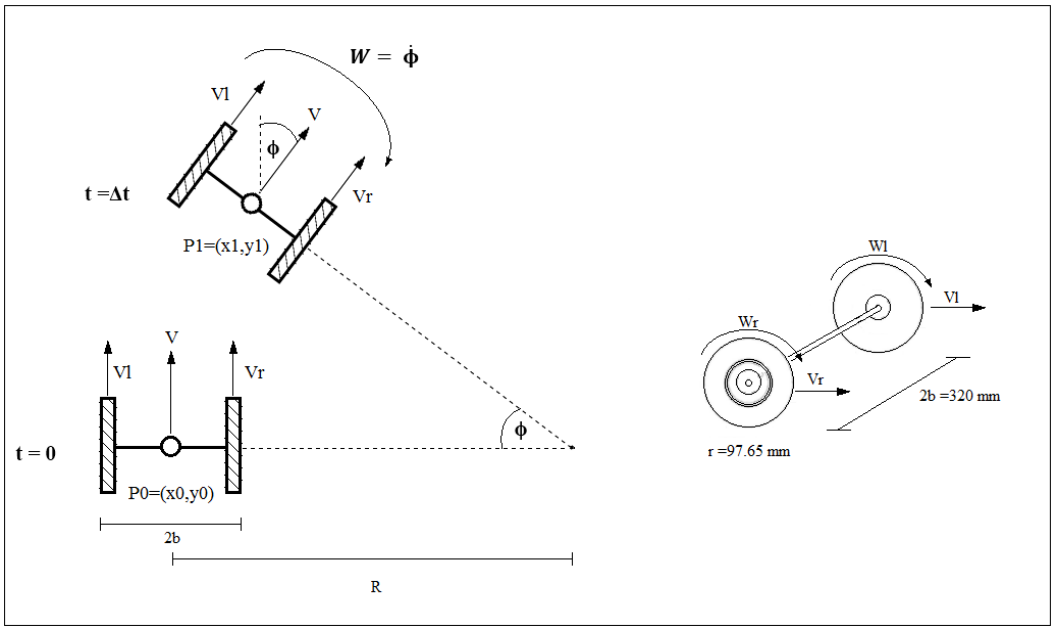


Figura 25.- Esquema para el modelamiento matemático del robot Pionner P3-DX

$$\begin{cases} \dot{x} = V cos(\phi) \\ \dot{y} = V sen(\phi) \\ \dot{\phi} = W \end{cases} \quad \textbf{(4)}$$

Donde:

$\dot{x}$ es la componente de velocidad del robot en el eje de coordenadas X

$\dot{y}$ es la componente de velocidad del robot en el eje de coordenadas Y

V es la velocidad lineal del robot medida en milímetros por segundo

ϕ indica la orientación del robot dentro del plano de trabajo

W es la velocidad angular del robot

$$W_l = \frac{V + bW}{r} \tag{5}$$

Donde:

W_l es la velocidad angular de la llanta izquierda del robot

b representa un medio de la separación existente entre las llantas del robot

r es el radio de las llantas del robot

$$W_r = \frac{V - bW}{r} \tag{6}$$

Donde:

W_r es la velocidad angular de la llanta derecha del robot

De esta manera para cualquier velocidad lineal deseada basta con calcular las velocidades W_l y W_r requeridas, y utilizarlas como SetPoint del controlador de velocidad de los motores.

En el caso de la orientación necesitamos definir la velocidad angular que nos permitirá alcanzar la orientación deseada; para ello debemos definir el error que existe respecto a la orientación actual del robot. Este error nos permitirá utilizar un controlador para calcular la velocidad angular necesaria.

3.2.3. Odometría del robot

Para que el robot conozca su ubicación y orientación, es necesario definir un modelo discreto que nos permita determinar los desplazamientos que realice el robot con ayuda de la lectura de los encoders.

Para ello discretizamos las distancias recorridas por cada motor utilizando la lectura realizada por el encoder.

$$d = 2\pi r \frac{\Delta tick}{76600} \quad \textbf{(7)}$$

Donde:

d es la distancia recorrida calculada por el encoder

$\Delta tick$ es el número de pulsos detectados por el encoder

Las distancias discretas de las llantas, nos permitirán hallar los cambios en posición y orientación del robot, utilizando el siguiente modelo (Validado en A II):

$$\begin{cases} x' = x + \dfrac{dl + dr}{2}\cos\phi \\ y' = y + \dfrac{dl + dr}{2}\operatorname{sen}\phi \\ \phi' = \phi + \dfrac{dl - dr}{2b} \end{cases} \quad \textbf{(8)}$$

Donde:

dl y dr son las distancias calculadas por los encoders izquierdo y derecho respectivamente

(x,y, ϕ) es la posición inicial del robot

(x', y', ϕ') es la posición actualizada del robot

3.2.4. Detección y localización de obstáculos

La lectura realizada sobre el arreglo de sonares nos permite conocer la distancia que existe entre un obstáculo detectado hacia el robot; esta distancia es relativa, sin embargo puede ser localizada en el ambiente de trabajo utilizando herramientas

geométricas básicas; de tal manera la ubicación de un obstáculo en el ambiente de trabajo será:

$$X_{obstáculo} = X_{robot} + d_{sensor}\cos(\phi_{sensor} + \phi_{robot})$$

$$Y_{obstáculo} = Y_{robot} + di_{sensor}\text{sen}(\phi i_{sensor} + \phi_{robot})$$

Donde:

$(X,Y)_{obstáculo} = Posición\ del\ obstáculo\ en\ el\ entonrno\ de\ trabajo$

$(X,Y)_{robot} = Posición\ del\ robot\ en\ el\ entonrno\ de\ trabajo$

$di_{sensor} = Distancia\ medida\ por\ el\ sensor\ i$

$\phi i_{sensor} = Orientación\ del\ sensor\ repecto\ al\ robot$

$\phi_{robot} = Orientación\ del\ robot\ respecto\ al\ entorno\ de\ trabajo$

3.3.Adaptación del sistema

En esta sección hablaremos de los cambios necesarios a realizar para la implementación de la FPGA sbRIO como controlador embebido del robot Pioneer P3-DX, para ello debemos determinar si los recursos que posee la FPGA son suficientes para cumplir las funciones básicas de control, y de no ser posible, determinar los cambios, ya sea en la arquitectura del sistema o en estructura física de los controladores para su correcto funcionamiento como controlador del robot.

3.3.1. Determinación de modificaciones

La FPGA sbRIO 9639 tiene las características mostradas en la Tabla 2, en la que se indica sus rangos de operación y capacidades de procesamiento y memoria.

En primer lugar necesitamos determinar el número de entradas/salidas análogas y digitales necesarias para el funcionamiento de los sensores y actuadores del robot, además necesitamos conocer la frecuencia a la que operaran para conocer si el reloj de la FPGA puede cumplir con los requisitos del sistema.

El controlador original del robot posee conectores independientes para cada sección del robot, y cada uno de estos conectores posee pines para señales de alimentación, lectura y control de los distintos sensores y actuadores. En la Figura 26 se indican los conectores que posee el controlador original del robot.

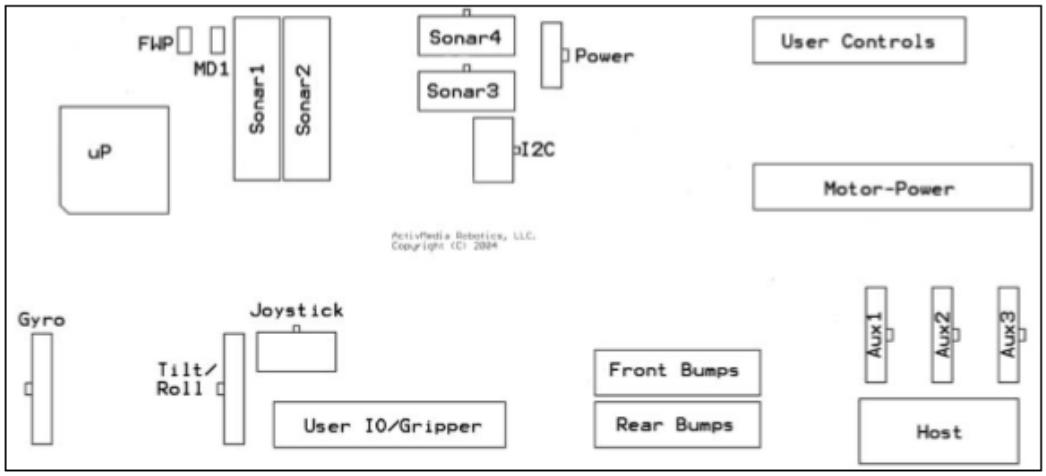


Figura 26.- Conectores del controlador SH2 del robot Pioneer P3-DX

Fuente: [6]

En el proyecto podemos utilizar las placas existentes de control de motores, sonares y baterías, por lo cual es necesario conocer los cables necesarios para la conexión desde la FPGA a las distintas placas, y el funcionamiento de cada uno de los pines que utilizaremos. . Con la información obtenida del manual, tenemos los pines necesarios de cada conector que vamos a utilizar resumidos en las Tabla 5 y Tabla 6

Tabla 5
Pines del Conector SONAR1 para controlar el arreglo de sonares

PIN	Señal	Descripción
1	A0	Selector de sensor
2	A1	Selector de sensor
3	A2	Selector de sensor
4	BINH	Desactiva la señal de retorno
5	INIT	Inicia el ping del sonar
6	VCC	Alimentación de 5V
7	VCC	Alimentación de 5V
8	GND	Común a tierra
9	GND	Común a tierra
10	ECHO	Señal de retorno

Fuente: [6]

Tabla 6
Pines del conector Power-Motor para el control de motores

PIN	Señal	Descripción
1	LPWM	PWM del motor izquierdo
2	LDIR	Sentido de giro motor izquierdo
3	RPWM	PWM del motor derecho
4	RDIR	Sentido de giro motor derecho
5	MEN	Habilitador de los motores
6	LEA	Canal A encoder izquierdo
8	REA	Canal A encoder derecho
10	REB	Canal B encoder derecho
12	LEB	Canal B encoder izquierdo
23	GND	Común a tierra
25	GND	Común a tierra

Fuente: [6]

En total necesitamos 15 señales de tipo digital para poder controlar los sensores y motores del robot, ya que la FPGA a utilizar en este proyecto cuenta con 28 pines digitales en total, disponemos de la cantidad suficiente para utilizarlos de forma directa sin ningún hardware adicional externo; en cuanto a la frecuencia con la que se debe generar la señal de estos pines los analizamos a continuación.

El diagrama de conexión de las FPGA con los componentes originales del robot se indica en la Figura 27:

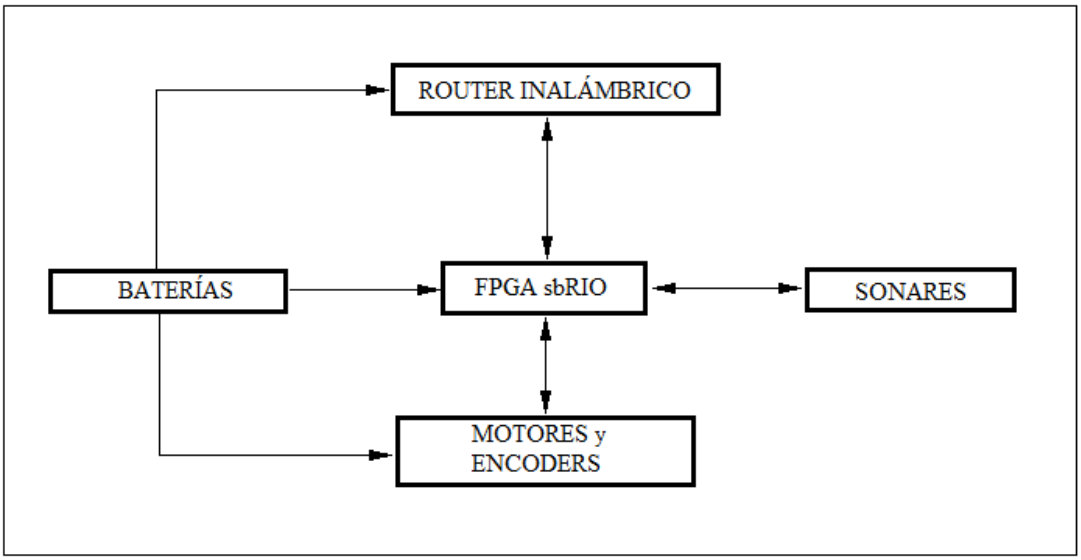


Figura 27.- Conexión FPGA en el sistema robótico

Consideramos además que los componentes originales del robot, operan con señales de control de 5V y alimentación de los componentes de 12V. La FPGA por su lado trabaja en rangos de operación de voltaje, estos son:

- Alimentación de 9 a 30 V
- Lectura de señales digitales con 0 lógico de 0-2.5V y 1 lógico de 3.3 a 5V
- Generación de señales digitales de 0 lógico de 0V y 1 lógico de 3.3 V

Con esta información se decide: utilizar la fuente de alimentación de 12V DC para alimentar a la FPGA sbRIO, conectar de manera directa las señales de lectura digitales, y elevar el voltaje de 3.3 V a 5 V de las señales de escritura de la FPGA.

3.3.2. Acondicionamiento de señales internas.

Se mencionó previamente que las señales digitales generadas por la FPGA tienen un nivel alto de 3.3 V, que no son suficientes para el robot que opera con señales de 5V; por lo que es necesario realizar un circuito que eleve el voltaje sin perder velocidad de operación; las opciones para integrar circuitos que operan a diferentes voltajes son amplias; sin embargo, debemos recordar que el espacio que disponemos en nuestra placa de control es limitado, además el circuito de acondicionamiento debe operar a la misma frecuencia de las señales generadas por la FPGA, es decir con una frecuencia máxima de operación de 40Mhz (ciclos de trabajo de 25 ns).

En el caso de este proyecto se ha decidido utilizar integrados tipo buffer para el control PWM de motores y transistores en configuración corte-saturación para el control de los sonares. Los buffers al encontrarse disponibles en circuitos integrados, tienen un tamaño reducido, sin embargo su tiempo de respuesta es menor al de transistores y además en algunos casos llegan a presentar pérdidas de voltaje mayores según la especificación del fabricante.

En el caso de los buffers, su funcionamiento es el de repetir los datos recibidos a un voltaje determinado, y enviar la señal a la salida del integrado; de esta manera cambiamos el voltaje de la señal de entrada de 3.3 V DC a los 5 V DC que necesitamos. Tiene la gran ventaja de que al ser circuitos integrados su tamaño es reducido. Se optó por utilizar esta solución con las señales de control de PWM, debido a que las señales controlan el funcionamiento de puentes H, lo que indica que se requieren bajos voltajes y corrientes para su activación, por lo que si se producen pérdidas de voltaje afectaran poco o nada en el funcionamiento de los motores.

Para el arreglo de sonares se utilizan transistores tipo NPN en configuración corte saturación; los transistores presentan un tiempo de respuesta menor (entre 18-25 ns según el fabricante) a los buffers, y caídas de voltaje mínimas, por lo que son ideales para el acondicionamiento de las señales que operaran en los períodos de oscilación esperados (40 ms para los sonares); la desventaja que presentan es el tamaño, por lo

que su uso lo limitamos a únicamente a las señales de lectura de la FPGA que provienen del arreglo de sonares.

De esta manera, la conexión de los elementos se resume en la Figura 28, en donde se indican todos los pines digitales utilizados que se detallaron en las Tablas 4 y 5. La configuración y conexión de los elementos puede observarse de mejor manera en el Anexo III de este documento.

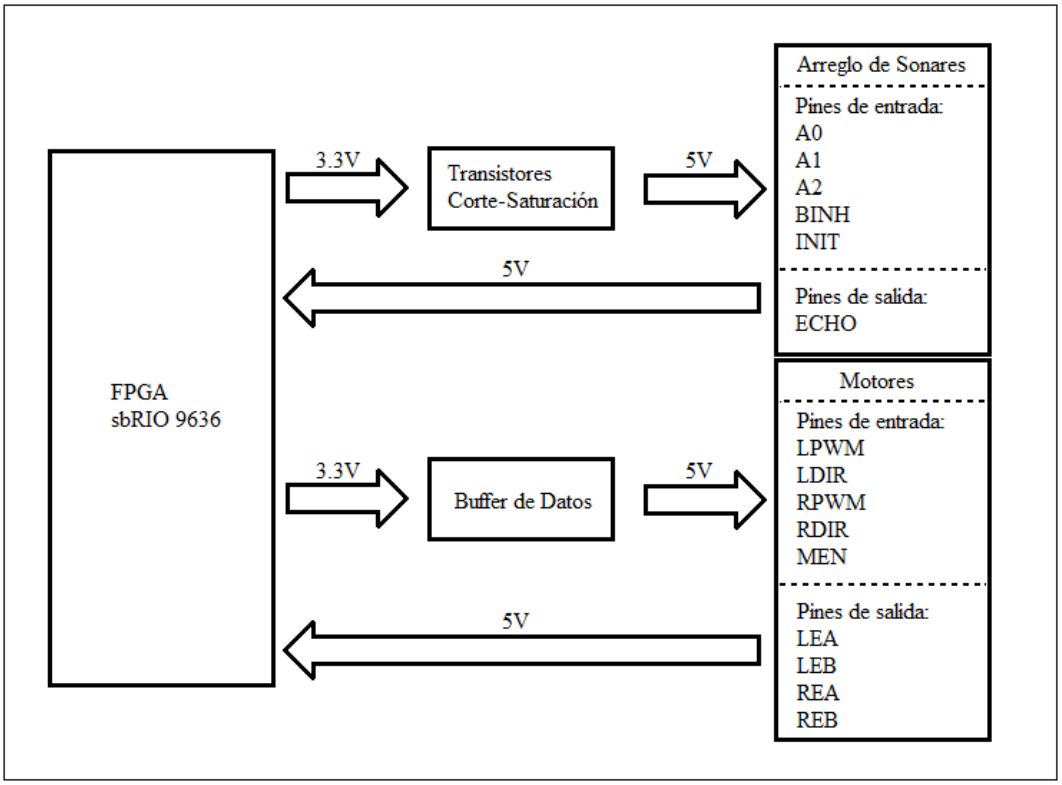


Figura 28.- Diagrama de conexión FPGA-P3DX

3.3.3. Estructura mecánica

Antes de realizar el cambio de controlador, debemos considerar el tamaño que ocupará la FPGA dentro del robot, además de cómo se realizaran las conexiones con los demás componentes internos, necesitamos definir los conectores compatibles entre los componentes existentes, y los soportes necesarios para que al momento de ubicarlos no generen inconvenientes con la estructura ya existente.

En la Tabla 7 se indican los conectores existentes en el proyecto, tanto en la FPGA sbRIO como dentro del robot Pioneer P3-DX.

Tabla 7
Conectores utilizados en el Sistema Embebido

NOMBRE	UBICACION	TIPO
SONAR1	Placa de sensores original del robot, ubicada en la parte frontal tras los sonares,	IDC de 10 pines macho con 2,5 mm de separación
MOTOR-POWER	Placa de potencia original del robot, ubicada tras el controlador original.	IDC de 26 pines macho con 2,5 mm de separación
J502 DIO	Parte frontal de la FPGA sbRIO	IDC de 50 pines con 2 mm de separación
Power FPGA	Parte frontal de la FPGA sbRIO	Mini-fit JR de 2 posiciones

De manera general los conectores existentes en la FPGA no son compatibles con los existentes en el robot; tampoco existen en el mercado adaptadores compatibles, por lo que es necesario diseñar una placa en la que se puedan realizar la conexión de los

distintos puertos utilizando pistas impresas. De esta manera, se tendrá un circuito impreso sobre el cual se podrán conectar los buses de datos compatibles con los conectores mostrados en la tabla anterior.

La placa impresa para la conexión del controlador deberá incluir además la alimentación de voltaje para la FPGA y demás componentes del sistema de control; el acondicionamiento de las señales utilizadas, y la interconexión de los distintos conectores permitiendo la correcta comunicación entre la FPGA y el robot y además incluir un puerto de comunicación para un computador de propósito general externo.

En cuanto al tamaño, de ser suficiente para abarcar todos los elementos de acondicionamiento y conectores, incluyendo soportes para colocar la FPGA, no debe exceder los 25 cm de largo ni 15 cm de alto, para que quepa dentro del espacio disponible en el interior robot. Debemos considerar además que la FPGA mide 10.2 x 15.3 cm con 6 postes de soporte ubicados en 2 filas de 3 postes cada una, con separación de 9.6 cm entre filas y de 7.36 cm entre cada poste. En la Figura 29 obtenida de la hoja de especificaciones se indican las medidas principales de la FPGA sbRIO

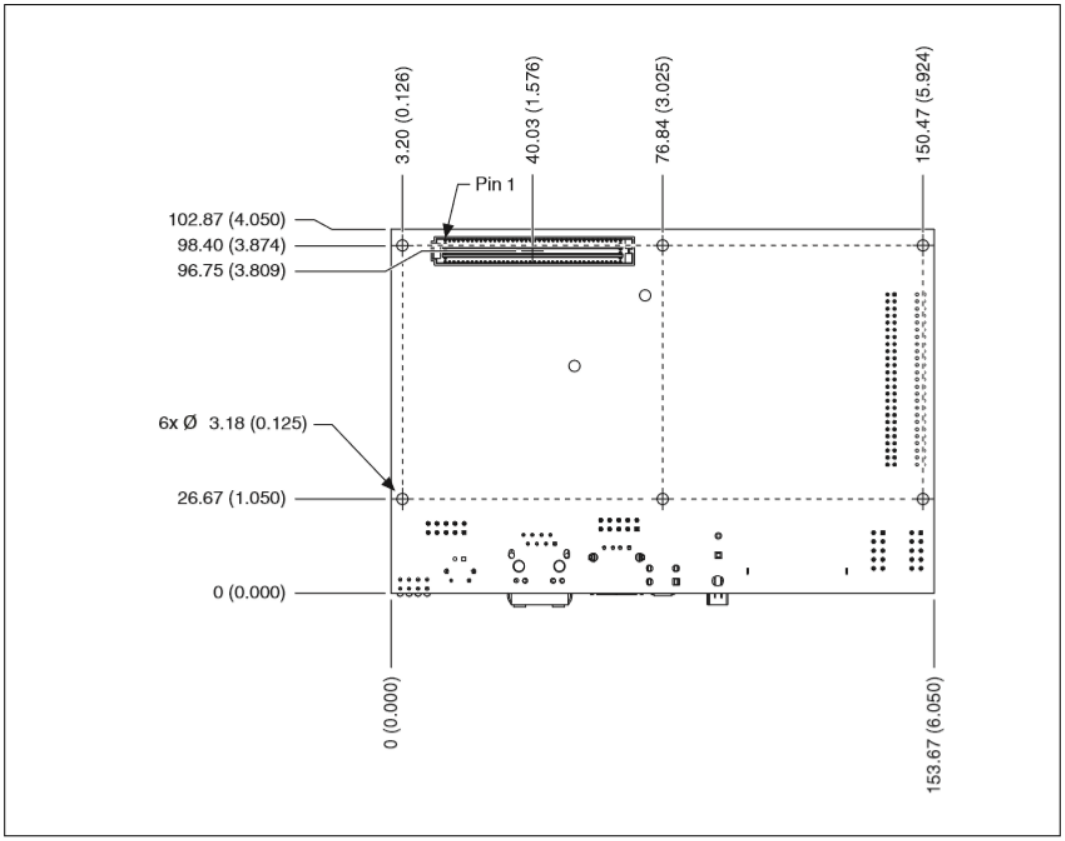


Figura 29.- Dimensiones FPGA sbRIO 9636 en milímetros y (pulgadas)
Fuente: [9]

Para realizar la placa de base se escogió una placa comercial de 20 x 10 cm en la cual colocaremos los siguientes conectores:

- Un conector IDC de 10 pines de 2,5mm para la conexión con la placa de los sonares existente en el robot
- Un conector IDC de 26 pines de 2.5mm para la conexión con la placa de motores y potencia del robot
- Un conector IDC de 50 pines de 2mm para conexión con los pines digitales de la FPGA
- Salida por cable con terminal Mini-fit macho de 2 posiciones para alimentación de 12V a la FPGA

- Conector micro-fit de 3 posiciones para conexión con las fuentes reguladas de 5 y 12 V existentes en el robot
- 6 postes para soporte de la FPGA de 3,18mm de diámetro

Además se incluirán elementos para el acondicionamiento de señales y conectores adicionales para los pines de la FPGA que queden sin ser utilizados.

La placa una vez construida y acoplada con la FPGA se colocará sobre los deslizadores existentes en el robot para el controlador original, cuya separación es de 20cm y se conectará con las placas de sonares y potencia del robot utilizando los mismos buses de datos existentes, compatibles con los conectores instalados.

La conexión de los componentes se indica en la Figura 30:

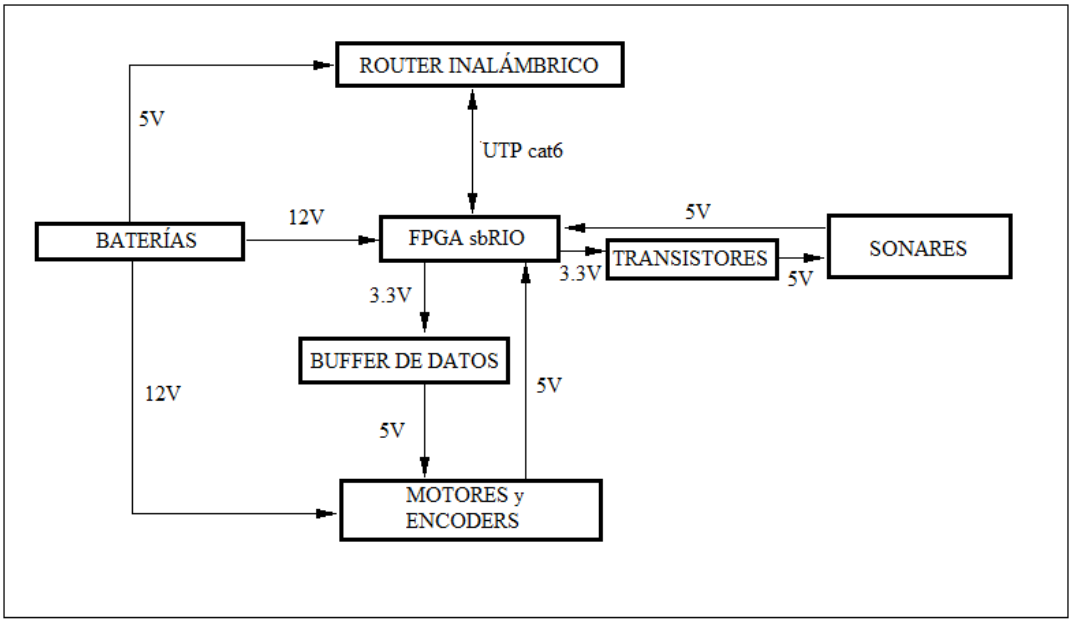


Figura 30.- Conexión de la FPGA con señales acondicionadas

3.3.4. Señales de comunicación inalámbrica

Un aspecto que ha sido mencionado, pero no tratado, es la comunicación del sistema embebido con el computador de propósito general; es el computador el que se

encargará del control de alto nivel del robot, por lo que una buena comunicación con la FPGA es importante en el desarrollo del proyecto.

Entre los varios puertos de comunicación que posee la FPGA sbRIO, su principal puerto para configuración inicial, programación y monitoreo, es un puerto Ethernet con conector RJ-45 categoría 5 compatible con la norma IEEE 802.3.

La configuración de los puertos Ethernet nos permite enlazarnos a una red de computadores a través de un router, aprovechando esta característica conectamos la FPGA a un router capaz de configurar una red inalámbrica a la cual estará conectada el computador. De esta manera establecemos la comunicación del computador con el sistema embebido.

Se ha escogido para este proyecto un router de la marca TP-Link modelo WR703N, el cual tiene las siguientes características:

- Un puerto Micro-USB para alimentación de 5V
- 1 Puerto Ethernet para configuración y/o conexión de un dispositivo
- Soporta modos cliente, repetidor, punto de acceso y como puente
- Compatible con IEEE 802.11b/g/n
- Dimensiones 5.7 x 5.7 x1.8 cm

Por sus características físicas, el router seleccionado puede ser colocado dentro del robot ya que el espacio que ocupa es reducido, y la alimentación que necesita de 5V la podemos obtener de la fuente regulada existente en el robot. Aprovechamos la placa de conexiones de la FPGA con el robot para añadir un puerto USB que nos servirá para suministrar el voltaje requerido por el router para su funcionamiento.

El modo de funcionamiento en el que se configura al router es como Acces Point, el cual nos permite conectar, a la FPGA de manera alámbrica a través de su puerto Ethernet, y al computador a través de la red inalámbrica utilizando tecnología Wi-Fi.

El esquema de los elementos conectados a la red se muestra en la Figura 31 a continuación:

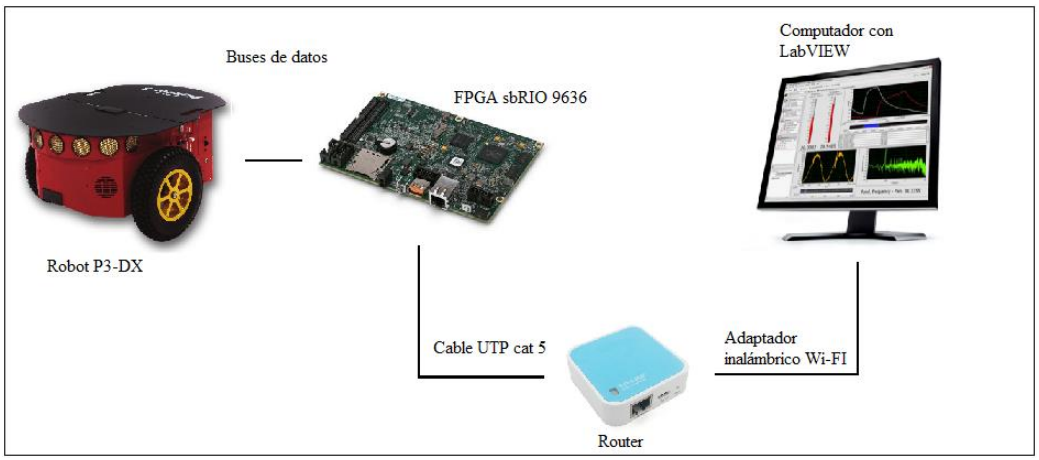


Figura 31.- Esquema de conexión de la red

CAPÍTULO IV
PROGRAMACIÓN DEL SISTEMA ROBÓTICO

En este capítulo trataremos los algoritmos de control que permitirán al robot ejecutar tareas de navegación; en el capítulo anterior se explicaron las tareas dedicadas que realiza la FPGA para controlar la navegación local del robot; en este capítulo explicaremos los algoritmos de alto nivel programados en un computador de propósito general, encargadas de la planificación de la trayectoria dentro de ambientes estructurados.

4.1.Algoritmos de control

A diferencia de los algoritmos de control programados en la FPGA, que permiten controlar la velocidad lineal y angular del robot, los algoritmos de control en el computador de propósito general, nos permitirán definir el comportamiento del robot para que pueda desplazarse de un punto a otro dentro de un entorno de trabajo.

4.1.1. Definición del algoritmo básico

En el marco teórico se trataron las etapas que debe cumplir el robot para poder cumplir con las tareas de navegación, en este apartado integraremos esas tareas en un algoritmo que nos permita posteriormente incorporar las tareas de la FPGA con los planificadores de trayectoria disponibles en las librerías de LabVIEW Robotics:

Para definir un algoritmo básico debemos conocer la información que es capaz de manejar la FPGA, y la que es necesaria para la ejecución de los de los algoritmos de navegación

Un esquema en el que se indica el funcionamiento del controlador lo indicamos en la Figura 32 a continuación:

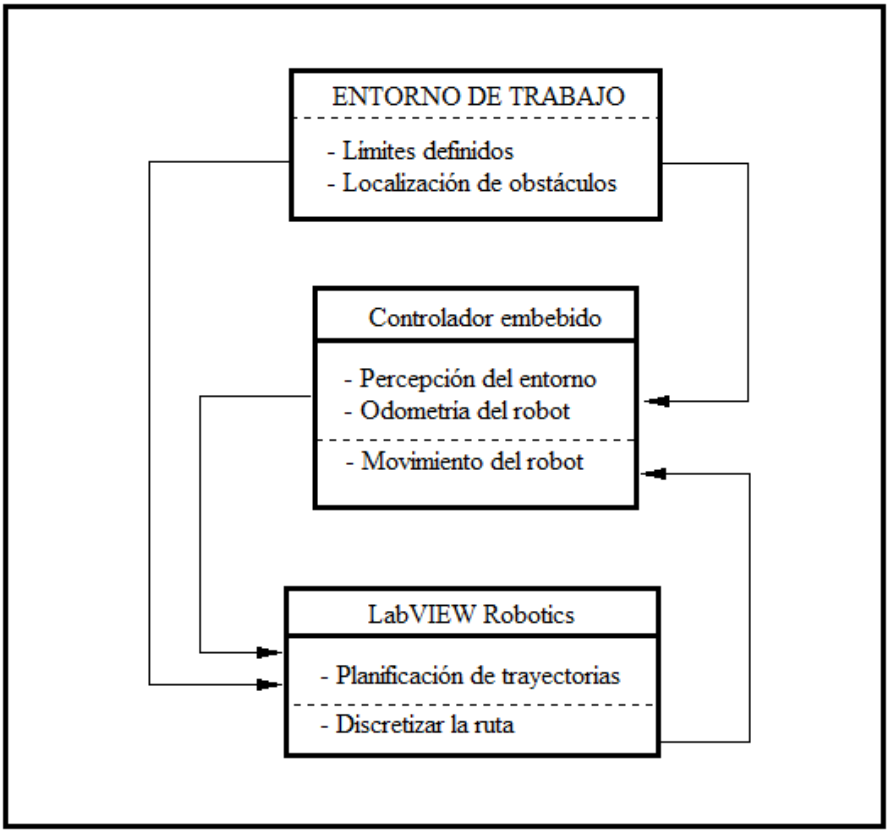


Figura 32.- Estructura del controlador

Con ello el algoritmo programado en el controlador con el objetivo de cumplir la tarea de navegación cumplirá con:

- Percepción del entorno a través de los sonares
- Auto localización del robot dentro del ambiente de trabajo
- Seguimiento de la ruta discretizada proporcionada por LabVIEW Robotics
- Actualización de la información requerida por LabVIEW Robotics

Con lo que el diagrama de flujo básico del robot queda definido en la Figura 33

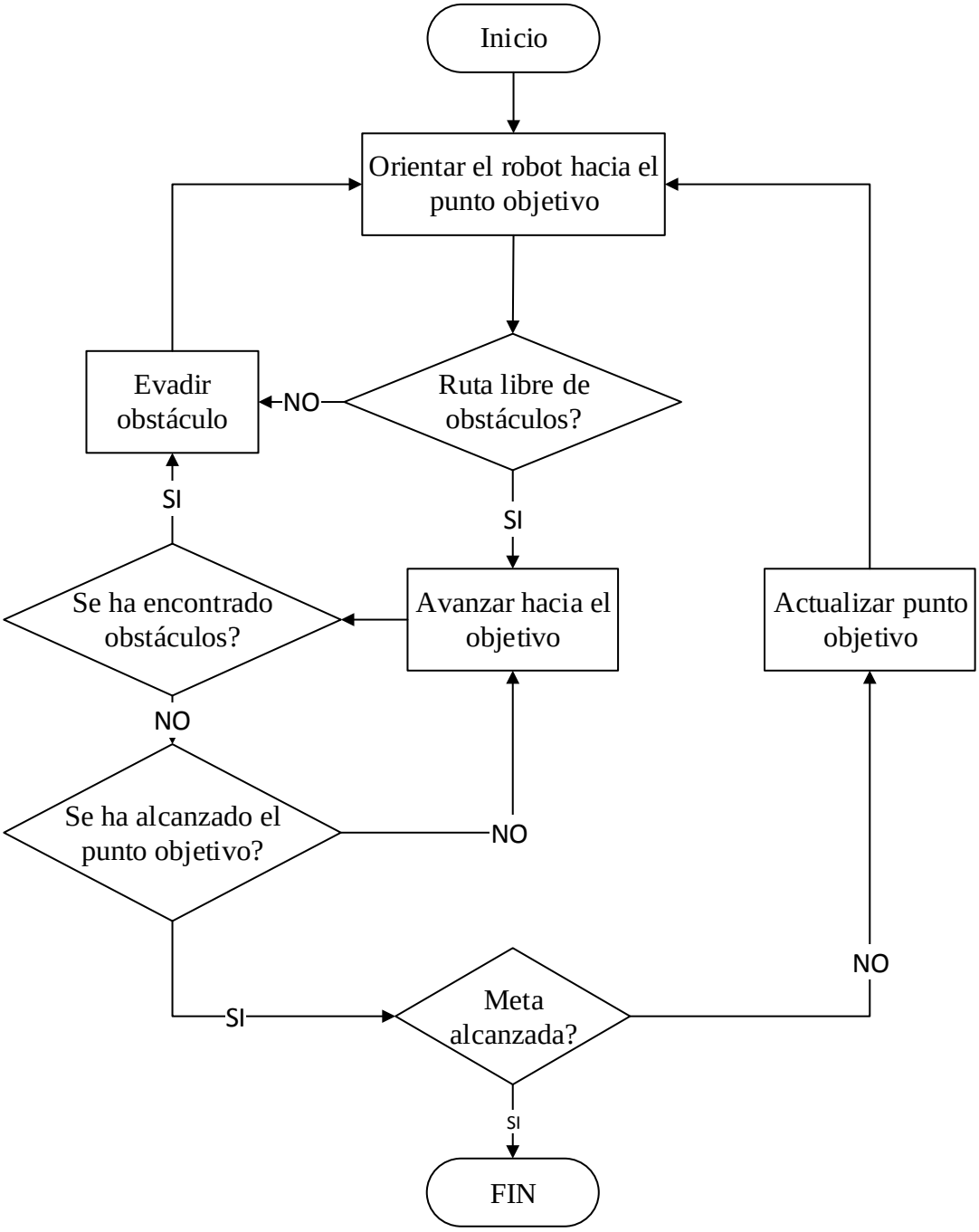


Figura 33.- Algoritmo básico para el control del robot

4.2.Algoritmos de navegación

Como mencionamos anteriormente en el marco teórico, el proceso de navegación incluye la planificación de una ruta libre de obstáculos para el desplazamiento del robot dentro de un ambiente de trabajo; en este capítulo explicaremos los algoritmos de LabVIEW Robotics para la planificación de rutas de robots móviles.

4.2.1. Algoritmos del LabVIEW Robotics

Dentro de las opciones de planificación de trayectorias, la librería *Robotics*, cuenta con la opción de planificación mediante el algoritmo de A-star; además cuenta con opciones para la creación de ambientes de trabajo basados en matriz de ocupancia, y mapas de nodos.

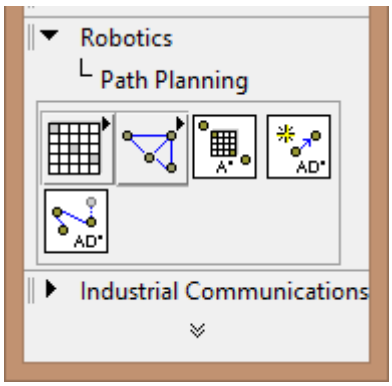


Figura 34.- Funciones de LabVIEW para planificación de rutas

A continuación explicaremos los algoritmos programados dentro de LabVIEW empleando las opciones existentes dentro de la librería.

- **A-star sobre matriz de impedancias**

Una matriz de impedancias, es una representación del ambiente de trabajo, en la que se indica la dificultad que tendría el robot para desplazarse sobre un área definida dentro del ambiente; cada elemento de la matriz tiene correspondencia con un área del ambiente de trabajo representado por coordenadas (X,Y)

Los elementos de la matriz pueden tomar valores entre 0 (el robot circula con el menor esfuerzo) y +∞ (El robot no podrá circular) que permitirán al algoritmo calcular la trayectoria óptima para la navegación del robot.

Una vez definido el espació de trabajo, es necesario definir el punto de origen, y el punto de meta, para que el algoritmo identifique la trayectoria óptima; con estos valores se ejecuta la planificación A-star explicada en el marco teórico, y la ruta calculada será enviada como puntos (x,y) al algoritmo de control del robot.

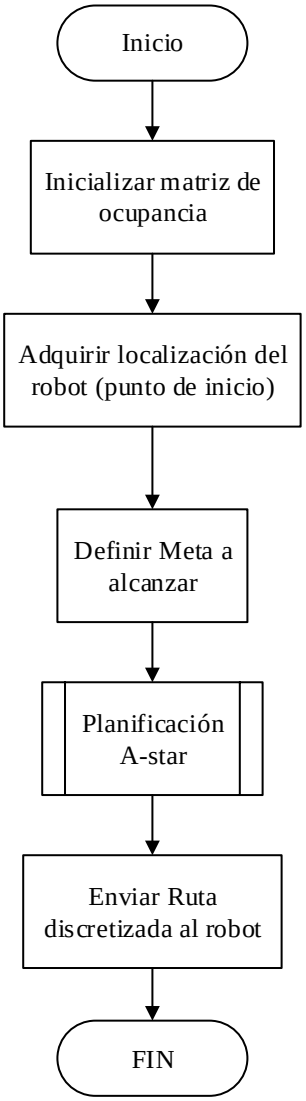


Figura 35.- Algoritmo A-star sobre matriz de ocupancia

- **A-star sobre mapa de nodos (Voronoi)**

A diferencia de la matriz de ocupancia, un mapa de nodos indica dentro del ambiente de trabajo, las trayectorias que el robot puede realizar, si el terreno es uniforme, la discriminación entre una y otra ruta, se la realiza mediante la distancia que el robot deberá recorrer entre un nodo y otro: La principal diferencia con el algoritmo de planificación utilizado sobre la matriz de impedancia, es la inicialización del ambiente de trabajo.

La ubicación de los nodos en el ambiente de trabajo se la puede realizar de forma manual, aleatoria, o siguiendo algoritmos avanzados como Voronoi, para el desarrollo de este proyecto utilizaremos la localización de Nodos mediante diagramas de Voronoi utilizando las herramientas existentes en las librerías de LabVIEW.

Los diagramas de Voronoi generan vértices alrededor de un punto en el plano XY, de tal manera que exista la mayor distancia posible entre cada vértice con cada punto del plano, como se muestra la Figura 36.

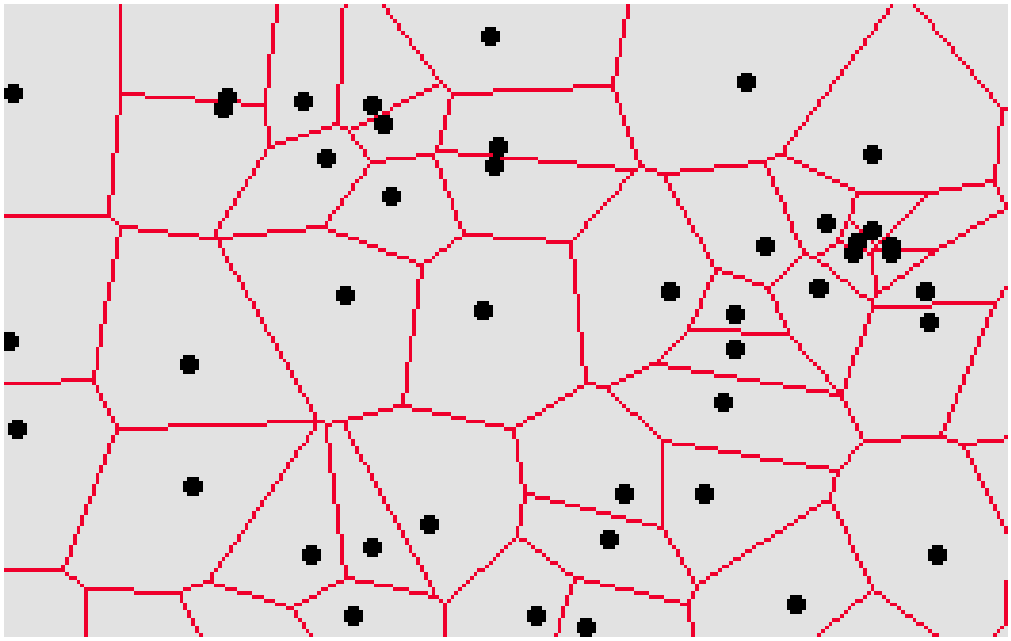


Figura 36.- Diagramas de Voronoi creado en LabVIEW

La intersección de dos o más vértices en el plano pasarán a formar los nodos de nuestro ambiente de trabajo, y cada uno de los vértices tendrá un peso asignado basado en su longitud, y la proximidad a los obstáculos que lo rodean; de tal manera que el algoritmo para la inicialización del ambiente de trabajo se indica en la Figura 37:

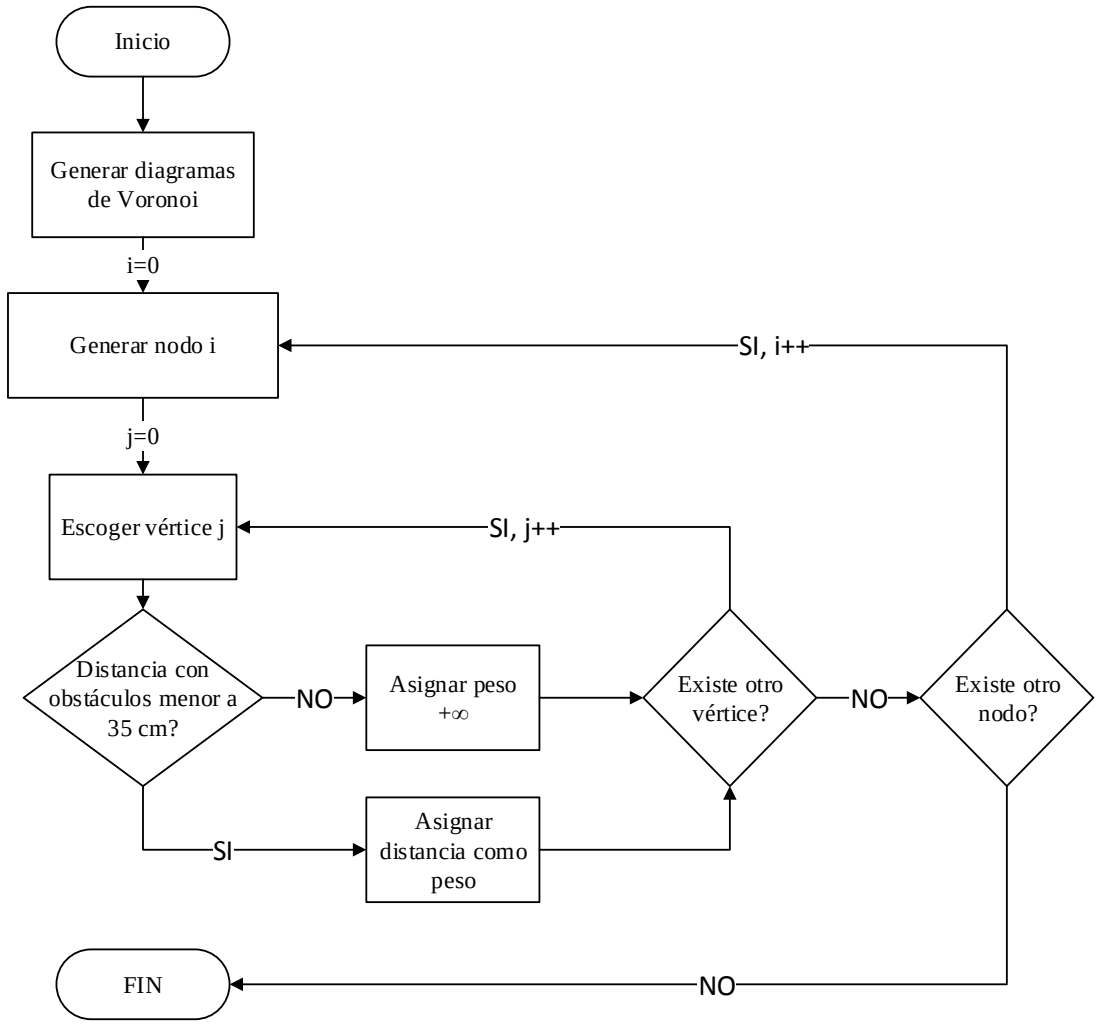


Figura 37.- Inicialización del ambiente de trabajo mediante Voronoi

Con lo que el algoritmo de planificación utilizando mapas de nodos generados por Voronoi se define como:

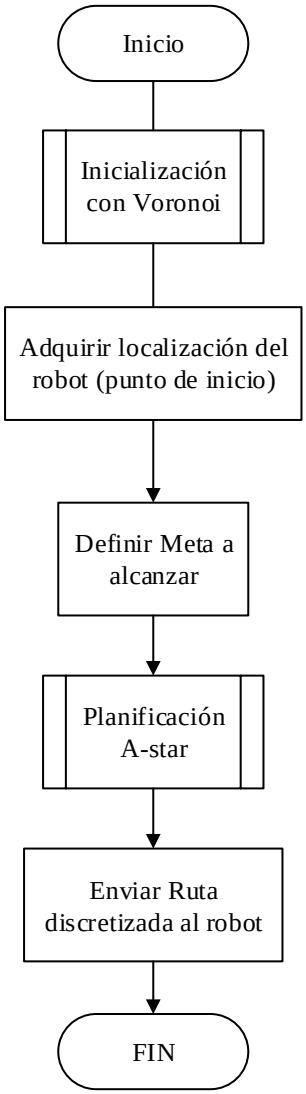


Figura 38.- Algoritmo A-star sobre Voronoi

CAPÍTULO V
EVALUACIÓN DEL SISTEMA

En este capítulo mostraremos los resultados de los algoritmos de control y navegación programados en el robot; comprobaremos los resultados obtenidos con el sistema embebido basado en FPGA contra los obtenidos utilizando el controlador original de la plataforma robótica Pioneer P3-DX. Se desarrollarán pruebas para comparar la velocidad de respuesta, auto localización, y detección de obstáculos utilizando el sistema de control basado en FPGA.

5.1.Pruebas de precisión y velocidad

Un correcto algoritmo de control del robot, permite desplazarse sin desviarse de su trayectoria, conocer la ubicación y orientación del mismo en cualquier momento durante la navegación; para comprobar el desempeño de los algoritmos de control programados en la FPGA se pedirá al robot mediante comandos que realice un desplazamiento lineal sobre un entorno de trabajo libre de obstáculos en donde se tomarán medidas de la ubicación final del robot y el error existente respecto a la meta programada, y el tiempo de ejecución del desplazamiento; estos valores serán comparados con los medidos sobre un robot P3-DX utilizando el controlador original ejecutando los mismos desplazamientos.

5.1.1. Desarrollo de la pruebas #1

El proceso consistirá en desplazar el robot por una distancia constante de un metro sin obstáculos; una vez que el robot se haya detenido completamente se realizarán mediciones de su posición final, respecto a su posición objetivo, y el tiempo de duración del desplazamiento; estos datos nos permitirán calcular la precisión y velocidad media del robot.

Este proceso se repetirá cinco veces para el controlador original del robot, y cinco veces con el controlador basado en FPGA, permitiendo comparar los resultados obtenidos.

- El robot es localizado en la posición de partida, junto a una línea guía como se muestra en la Figura 39
- Mediante comandos el robot se desplaza 1 metro sobre la línea guía
- Una vez que el robot se detenga por completo se mide la posición final del robot respecto al origen
- Se cronometra además el tiempo desde que el robot inició su movimiento hasta que se ha detenido por completo

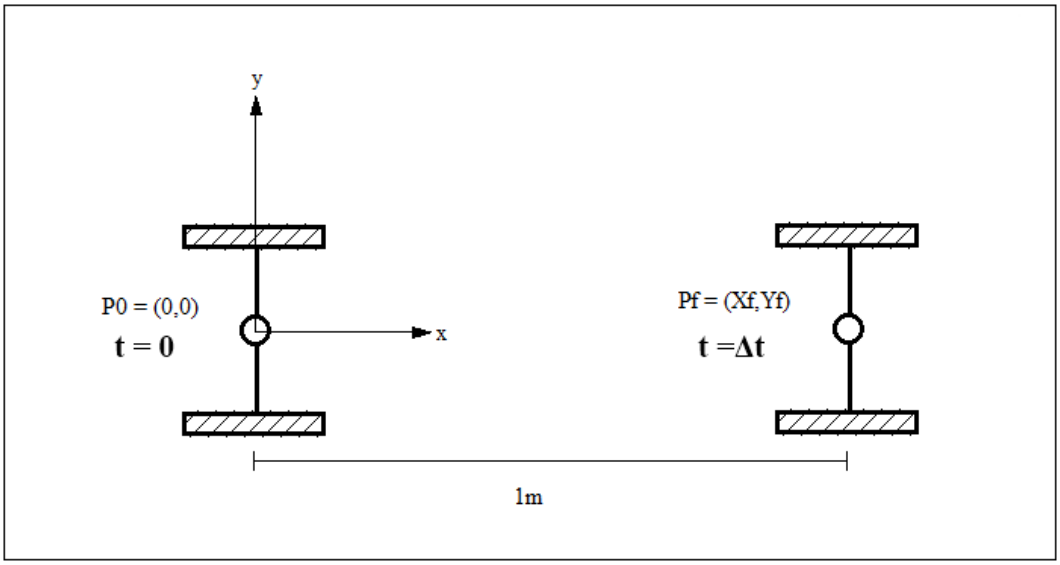


Figura 39.- Escenario de pruebas #1

Bajo este escenario las mediciones realizadas se resumen las Tabla 8 y Tabla 9

Tabla 8
Prueba #1, Mediciones realizadas con el controlador original

Prueba	Xf (cm)	Yf (cm)	Δt (s)
1	100	101	4,32
2	100	100	4,06
3	100	99	4,11
4	101	101	4,35
5	101	98	4,54

Tabla 9
Prueba #1, Mediciones realizadas con el controlador basado en FPGA

Prueba	Xf (cm)	Yf (cm)	Δt (s)
1	100	100	4,83
2	100	101	4,53
3	100	99	4,92
4	100	100	4,53
5	100	100	4,21

5.1.2. Análisis de resultados de la prueba #1

Para hallar la precisión que tienen los distintos controladores del robot, calcularemos la distancia promedio existente entre la posición alcanzada por el robot (Xf,Yf) con la meta programada (Xr,Yr) cm, utilizando la fórmula:

$$Desviación = \frac{1}{n}\sum_{i=0}^{n}\sqrt{(Xr_i - Xf_i)^2 + (Yr_i - Yf_i)^2}\,[cm] \quad \textbf{(9)}$$

Donde:

(Xr_i, Yr_i) = (100 , 100)

(Xf_i, Yf_i) es la posición final del robot en el intento *i* según la Tabla 8 y Tabla 9

n es el número de mediciones realizadas

La velocidad la calcularemos mediante la relación entre la ubicación final del robot y el tiempo transcurrido hasta que se detenga por completo.

$$Velocidad = \frac{1}{n}\sum_{i=0}^{n}\frac{10\sqrt{Xf_i^2 + Yf_i^2}}{\Delta t_i}\left[\frac{mm}{s}\right] \quad \textbf{(10)}$$

Donde:

(Xf_i, Yf_i) es la posición final del robot en el intento *i* según la Tabla 8 y Tabla 9

Δt_i es el tiempo que el robot demoró en realizar el recorrido en el intento *i*

n es el número de mediciones realizadas

Aplicando estas fórmulas a los datos adquiridos en la Tabla 8 y Tabla 9 obtenemos los resultados de la Tabla 10

Tabla 10
Resumen de los resultados obtenidos en la prueba #1

	Desviación (cm)	*Velocidad media (mm/s)*
Controlador Pioneer	1,13	233,64
Controlador FPGA	0,4	217,39

Con los valores de la desviación podemos calcular el la precisión que tiene el robot con cada uno de sus controladores indicados en la Tabla 11.

$$Precisión(\%) = \frac{Desviación}{Distancia\ Programada}x100 \quad \textbf{(11)}$$

Tabla 11
Precisión calculada de los controladores

	Precisión (%)
Controlador Pioneer	1,13
Controlador FPGA	0,4

Además la velocidad media del robot ha reducido de 233,64 mm/s a 217,39 mm/s con la implementación del controlador basado en FPGA, es decir, el controlador original alcanza una velocidad media del 93,04% del controlador original.

5.2.Pruebas de auto localización

El mayor inconveniente de operar con un sistema de odometría basado en encoders es la acumulación de errores en la posición del robot; estos errores tienden a aumentar con el tiempo de funcionamiento del robot, la cantidad de giros que realice y los derrapes producidos en los desplazamientos del robot; para medir el efecto que estos eventos tienen en el sistema de auto localización del robot desarrollaremos la siguiente prueba:

5.2.1. Desarrollo de la prueba #2

Esta prueba la desarrollaremos en un ambiente de trabajo cerrado, limitado por el área mostrada en la Figura 40, El robot se desplazará libremente hasta encontrar un obstáculo que limite sus movimientos, El obstáculo deberá ser detectado por los sonares y el robot girará para evadirlo.

El robot continuará desplazándose por un tiempo total de 100 segundos mientras evita colisiones mediante giros en el ambiente de trabajo; una vez transcurrido el tiempo de ejecución se medirá la posición real en la que se encuentra el robot, en coordenadas (Xf,Yf) y la posición en la que el sistema de odometría del robot indica (Xr,Yr).

En caso de que se produzca una colisión del robot, el intento se contará como fallido , será anulado y se reiniciará el intento; la prueba se repetirá hasta que se alcancen un total de 4 intentos exitosos, es decir sin colisiones durante el movimiento del robot.

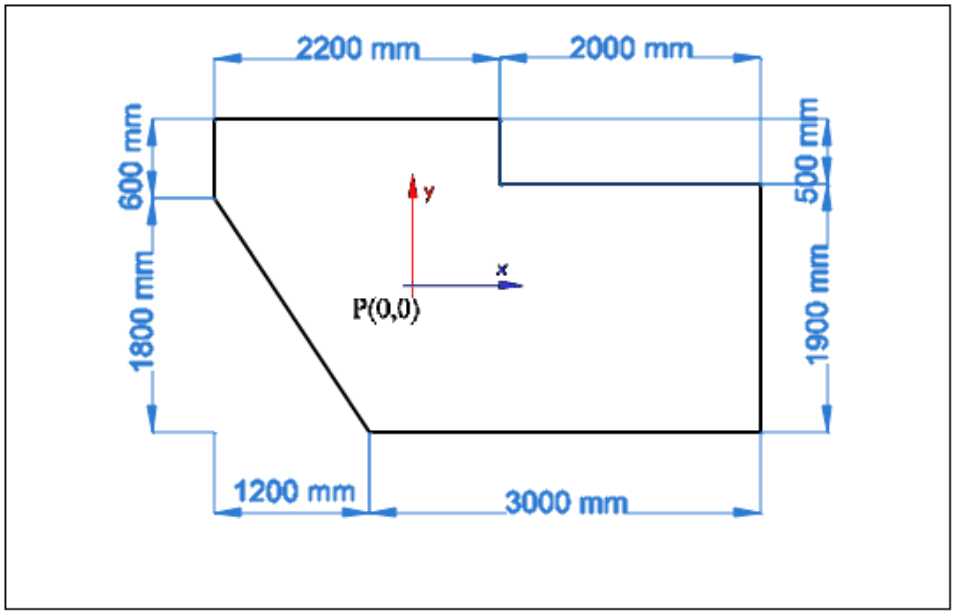


Figura 40.- Escenario de pruebas #2

Los resultados medidos en esta prueba se resumen en las tablas a continuación:

Tabla 12
Prueba #2, Mediciones realizadas con el controlador original

Intento	Tiempo recorrido (s)	Xr (mm)	Xf (mm)	Yr (mm)	Yf (mm)
1	100	2554	2430	-119	-930
2	100	269	-54	-1190	-960
3	100	1401	1480	236	-410
4	100	-501	-540	540	-30

Número de intentos anulados: 2

Tabla 13
Prueba #2, Mediciones realizadas con el controlador basado en FPGA

Intento	Tiempo recorrido (s)	Xr (mm)	Xf (mm)	Yr (mm)	Yf (mm)
1	100	2200	2310	-1121	-980
2	100	1988	1900	-1120	-1074
3	100	-864	-870	80	56
4	100	1248	1110	270	200

Número de intentos anulados: 1

5.2.1. Análisis de resultados de la prueba #2

Esta prueba se enfoca en la exactitud que tiene el sistema de auto localización del robot; para ello utilizamos la ecuación (9) sobre los datos de la Tabla 12 y Tabla 13 obteniendo los resultados mostrados en la Tabla 14

Tabla 14
Resultados obtenidos de la prueba #2

	Desviación promedio (mm)
Controlador Pioneer	609,77
Controlador FPGA	114,40

Con estos datos, podemos observar que el error en la exactitud del controlador se ha reducido a un 18,76% del error existente en el controlador original presentando una mejora significativa en el sistema de auto localización del robot.

5.3.Prueba para detección de obstáculos

Con el objetivo de comprobar el correcto funcionamiento de los sonares, se desarrolló la siguiente prueba:

- Se colocó un obstáculo plano de madera de 1000x1200 mm a distancias fijas de 250, 500, 750, 1000, 2000, 3000 y 4000 mm de los sonares.

- Se mide el error entre la distancia del obstáculo al sonar, y la distancia medida por el controlador.
- Este procedimiento se repite para los sonares frontales del robot.

Los resultados obtenidos se muestran en la siguiente Tabla 15:

Tabla 15
Prueba #3, Distancias medidas por el sonar

Controlador	*Distancia (mm)*						
	Sonar	250	500	750	1000	2000	3000	4000
PIONNER	3	244	495	749	1000	1998	3053	4015
	3	244	503	74	996	1997	3050	4079
	4	251	495	752	1001	1996	3026	4025
	4	240	498	748	996	1998	3041	4033
FPGA	3	245	504	745	999	2001	3066	4035
	3	239	499	748	997	2000	3067	4048
	4	249	498	746	995	1996	3056	4005
	4	243	502	751	995	1996	3022	4039

5.3.1. Análisis de resultados de la prueba #3

Para calcular la exactitud en la lectura de los sensores calculamos el error entre las lecturas de los sensores y las distancias reales de los objetivos utilizando la fórmula:

$$Error\ cuadrático = \frac{1}{n}\sum_{i=0}^{n}\left|\frac{Dr_i - Dm_i}{Dr_i}\right| x100\% \quad \textbf{(12)}$$

Donde:

Dr es la distancia real del obstáculo, y

Dm es la distancia medida por los sonares

Tabla 16
Resultados obtenidos de la prueba #3

	Error sensores (%)
Controlador Pioneer	0,95
Controlador FPGA	0,79

Para los dos controladores utilizados en el robot, observamos que el error en las lecturas es menor al 1%, permitiendo una localización exacta de los sensores dentro de su rango de funcionamiento (entre 100 y 4000 mm)

CAPÍTULO VI
CONCLUSIONES Y RECOMENDACIONES

6.1.Conclusiones

Al finalizar este proyecto, se demostró que el controlador implementado para la plataforma Pioneer P3-DX es capaz de cumplir las tareas básicas del controlador original; con los algoritmos programados en el sistema embebido se logró manipular los sensores y actuadores del robot involucrados de manera directa en la tarea de navegación. Estos algoritmos no solo son capaces de controlar los componentes del robot; sino que cuentan con parámetros de configuración para la calibración del robot en distintos medios (escalamiento de datos, sintonización de controladores, activación y desactivación de los componentes)

Se construyó una tarjeta de acondicionamiento que permite la integración del sistema embebido basado en FPGA con los componentes del robot, esta tarjeta electrónica, nos permite la conexión y control de los motores, encoders y sonares del robot (todos ellos funcionando mediante lógica digital), y se excluyeron los indicadores de voltaje de batería, buzzers y leds indicadores del panel de usuario, debido a la inaccesibilidad física de los canales analógicos de la FPGA una vez integrada con los componentes digitales del robot.

Mediante las pruebas desarrolladas se comprobó que la velocidad del controlador implementado es ligeramente menor al controlador original; sin embargo se ha mejorado el sistema de auto localización que le permite al robot conocer su ubicación y orientación dentro del ambiente de trabajo por un recorrido mayor al del controlador original. El alcance y precisión de los sensores se mantuvo igual a la del controlador

original, junto con funciones que permiten su activación o desactivación de manera independiente como se tenía antes del cambio de controlador.

Además se desarrollaron programas para la planificación de trayectorias utilizando los algoritmos disponibles en las librerías de LabVIEW Robotics, Estos algoritmos, permitieron la simulación de un ambiente estructurado para el robot, permitiendo comprobar la comunicación con el sistema de control embebido del robot a través de la red Wi-Fi; Permitiendo controlar los movimientos del robot para el seguimiento de la trayectoria planificada utilizando LabVIEW.

6.2.Recomendaciones

Se recomienda que la velocidad de rotación de las llantas no supere los 2,5 rad/s; esto implica una velocidad lineal máxima de 220 mm/s y una velocidad angular del robot no superior a los 2 rad/s; Para reducir los errores debidos al deslizamiento de las llantas sobre la superficie de trabajo.

El controlador del robot P3-DX ha sido configurado para operar en una superficie de deslizamiento lisa de tipo baldosa, si se requiere utilizar una superficie de deslizamiento diferente, será necesario configurar los parámetros del controlador del robot; es decir, realizar una sintonización de los controladores de los motores, y ajuste de los parámetros de escalamiento del sistema de odometría.

Se recomienda la construcción de una segunda placa de acondicionamiento, que permita el acceso a los canales analógicos de la FPGA, con el propósito de habilitar el panel de usuario del robot, permitiendo de esta manera al controlador, conocer el estado de las baterías e indicadores existentes en el robot.

REFERENCIAS

[1] CHOSET, H. (2005). PRINCIPLES OF ROBOT MOTION. CAMBRIDGE: THE MIT PRESS.

[2] V.F. MUÑOZ, A. OLLERO. (1996). SMOOTH TRAJECTORY PLANNING FOR MOBILE ROBOTS. COMPUTATIONAL ENGINEERING IN SYSTEM APPLICATIONS (CESA'96). LILLE (FRANCIA).

[3] V.F. MUÑOZ (1997). PLANIFICACIÓN DE TRAYECTORIAS PARA ROBOTS MÓVILES. TESIS DOCTORAL. DPTO. INGENIERÍA DE SISTEMAS Y AUTOMÁTICA. ISBN: 84-8498-970-4.

[4] BARRIENTOS, A (1996). FUNDAMENTOS DE ROBOTICA, MADRID (ESPAÑA)

[5] HEATH, S (2003). EMBEDDED SYSTEMS DESIGN, (REINO UNIDO) , ISBN: 0-7506-5546-1.

[6] PIONEER P3-DX. (2014). OBTENIDO DE MOBILE ROBOTS: HTTP://WWW.MOBILEROBOTS.COM/RESEARCHROBOTS/PIONEERP3DX.ASPX

[7] APLICACIONES ROBÓTICA: NATIONAL INSTRUMENTS. (2014). OBTENIDO DE NATIONAL INSTRUMENTS CORPORATION: HTTP://WWW.NI.COM/ROBOTICS/ESA/

[8] GETTING STARTED WITH COMPACTRIO : NATIONAL INSTRUMENTS. (2014). OBTENIDO DE NATIONAL INSTRUMENTS CORPORATION: HTTP://WWW.NI.COM/PDF/MANUALS/372596B.PDF

[9] MULTIFUNCTION RIO : NATIONAL INSTRUMENTS. (2014). OBTENIDO DE NATIONAL INSTRUMENTS CORPORATION: HTTP://WWW.NI.COM/PDF/MANUALS/370489G.PDF

[10] A* PATHFINDING FOR BEGINNERS. (2005). OBTENIDO DE HTTP://WWW.POLICYALMANAC.ORG/GAMES/ASTARTUTORIAL.HTM

[11] COMMUNITY: AN INTRODUCTION TO A* PATH PLANNING (USING LABVIEW) - NATIONAL INSTRUMENTS. (2010, ENERO 13). OBTENIDO JULIO 19, 2015, FROM HTTPS://DECIBEL.NI.COM/CONTENT/DOCS/DOC-8983

[12] GUTIERREZ, R. S. (N.D.). ROBOTICA2010 - 7 ALGORITMOS DE NAVEGACIÓN. RETRIEVED FROM HTTP://ROBOTICA2010 /7+ALGORITMOS+DE+NAVEGACIÓN

[13] MORALES BALLADARES; J.J., JARAMILLO FABARA; D. F. (2010). *DESARROLLO DE APLICACIONES Y DOCUMENTACIÓN DE LAS PLATAFORMAS ROBÓTICAS PIONEER P3-DX Y PIONEER3 P3-AT*. (TÉSIS). UNIVERSIDAD DE LAS FUERZAS ARMADAS - ESPE. SANGOLQUÍ.

[14] GUFFANTI MARTÍNEZ; D.A., (2013). *CONTROL REMOTO POR VOZ DEL ROBOT MÓVIL PIONNER P3-DX*. (TÉSIS). UNIVERSIDAD DE LAS FUERZAS ARMADAS - ESPE. SANGOLQUÍ.

[15] ALVAREZ ROMERO; L. M., FIGUEROA MONTENEGRO, J. E., (2011). *IMPLEMENTACIÓN DE ALGORITMOS DE NAVEGACIÓN UTILIZANDO LA PLATAFORMA DEL IROBOT CREATE Y MÓDULOS DE*

COMUNICACIÓN INALÁMBRICA XBEE. (TÉSIS). UNIVERSIDAD DE LAS FUERZAS ARMADAS - ESPE. SANGOLQUÍ.

[16] PETROV, P., DIMITROV, L. (2009). *NONLINEAR PATH CONTROL FOR A DIFFERENTIAL DRIVE MOBILE ROBOT* (TÉSIS). TECHNICAL UNIVERSITY OF SOFIA, BULGARIA

[17] ARANDA BRICAIRE; E., SALGADO JIMENEZ; T., VELAZSO VILLA; M., (2002). *CONTROL NO LINEAL DISCONTÍNUO DE UN ROBOT MÓVIL* [ABSTRACT]. COMPUTACIÓN Y SISTEMAS , NÚMERO ESPECIAL, 042-049

[18] RÍOS G., LUÍS HERNANDO; BUENO L., MAXIMILIANO, *MODELO MATEMÁTICO PARA UN ROBOT MÓVIL*, SCIENTIA ET TECHNICA, VOL. XIV, NÚM. 38, JUNIO, 2008, PP. 13-18, UNIVERSIDAD TECNOLÓGICA DE PEREIRA, PEREIRA, COLOMBIA

ACTA DE ENTREGA

El proyecto fue entregado en la Dirección de la Carrera de Ingeniería en Electrónica, Automatización y Control y reposa en la Universidad de la Fuerzas Armadas-ESPE, desde:

Sangolquí, 04 DE SEPTIEMBRE de 2015

ELABORADO POR:

PARRA ANDRADE GEN EDUARDO

100293903

Ing. Luis Orozco MSc.

DIRECTOR DE LA CARRERA DE INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL