SECRETARÍA DE EDUCACIÓN PÚBLICA

SEP

Instituto Tecnológico de Tuxtla Gutiérrez

Reporte Final

Residencia Profesional:

Desarrollo y Aplicación de Algoritmos de Cooperación en Robots Móviles

Manuel Antonio Castellanos Pérez
No. Control: 08270395
Ingeniería Electrónica
Alumno

Dr. Alejandro Medina Santiago
Ing. Álvaro Hernández Sol
L.F.M. Lester Acosta Maza
Ing. Vicente León Orozco
Revisores

Tuxtla Gutiérrez Chiapas,
Enero del 2015.

RESUMEN

En el siguiente trabajo veremos la cooperación entre dos robots móviles, ambos robots con tracción diferencial, al primer robot móvil se le denomino líder y al segundo seguidor, se les diseño dos algoritmos, uno de control lazo abierto y otro de lazo cerrado respectivamente. Se presentaran dos partes: La simulación y la aplicación real.

En la parte de la simulación en MATLAB, se diseñó los algoritmos a partir de los modelos cinemáticos de cada robot para programar su trayectoria y control. Para llevar a cabo la cooperación simulada fue fundamental usar las herramientas Simulink y Simulink 3D Animation contenidas en MATLAB.

En la aplicación real en ambos robots se usó la plataforma Arduino el cual está diseñado para soportar diferentes módulos electrónicos. Para cada robot se diseñó una placa para montar un módulo de antena XBee, el puente H, el sensor infrarrojo, conectores, cable de datos, baterías y componentes electrónicos. El robot móvil líder es controlado de forma inalámbrica por un celular que tiene la aplicación de TouchOSC Bridge, esto es posible a través del módulo XBee que permite la comunicación con la computadora e interacción con el celular usando la red de internet, la programación se realizó en el lenguaje de programación Processing, dicho robot líder en la parte de atrás tiene una tarjeta de color negro, para ser detectado por el robot móvil seguidor. Para la manipulación del robot móvil seguidor se le monto en la parte de enfrente-centro un sensor infrarrojo. El dato de salida del sensor infrarrojo da 40 milivolts cuando detecta la tarjeta de color negro (localizado en el robot móvil seguidor) cuando está a una distancia de 7 centímetros de un robot a otro, cuando el sensor envía al robot seguidor un dato diferente de 40 milivolts, significa que el robot líder está aumentando o disminuyendo su velocidad y como el robot líder está programado para mantenerse a una distancia de 7 centímetros respecto al robot líder, entonces también el robot seguidor aumentara o disminuirá su velocidad cual sea el caso.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS Y TABLAS

Contenido **páginas**

CAPÍTULO 1

INTRODUCCIÓN

La ciencia de los robots es un área muy completa ya que incluye varias disciplinas como por ejemplo las matemáticas, física, mecánica, electrónica, computación entre otras para el desarrollo de su aplicación ya sea en inteligencia artificial, algoritmos genéticos, algoritmo de cooperación etc., con la aplicación de estas herramientas nos permiten crear a "robots" mecánicos y diseñarlos con aspectos muy parecidos al del ser humano que inclusive pueden llegar a remplazarnos. Un aspecto importante es la aplicación que se le da a cada robot, por ejemplo [1]: exploración en el universo, fábricas mineras y terrenos marítimos, inspección y vigilancia, misiones de búsqueda, y rescate de personas, en la medicina robots que incluso realizan cirugías, ocio y entretenimiento, Investigación militar, industria metal-mecánica, industria química, agricultura, trasporte, entre otros. Las aplicaciones mencionadas nos da la oportunidad de hacer la vida más cómoda y menos peligrosa, pero para obtener esos beneficios es necesario programar a los robots para que tengan la capacidad de cooperar entre ellos, para ello se debe programar un algoritmo a cada robot para que estos actúen de forma autónoma mediante su propio proceso de razonamiento (esto es posible mediante el uso de sensores que le permite a cada robot percibir el entorno donde se encuentran), para la programación del algoritmo es muy común usar un control de lazo cerrado.

Para construir un robot móvil hay infinidad de diseños para su desplazamiento, por ejemplo: robots de ruedas, robots orugas, robots con patas. En los diseños de robots la mayoría se basan en robots de ruedas para su locomoción, ya que son más eficientes en energía en superficies lisas y firmes. Pero para que este se mueva de forma autónoma se necesita la programación del algoritmo de acuerdo al objetivo de la aplicación que se le quiera dar.

Actualmente se habla de software libre que vienen con sus respectivos tutoriales el cual permite ahorrar tiempo y dinero, como por ejemplo tenemos la plataforma Arduino en el que se puede realizar diversos trabajos ya que es una herramienta muy didáctica que se ajusta a las necesidades del proyecto, su construcción permite que se conecten mas placas o tablillas con componentes electrónicos sobre la plataforma Arduino eso permite hacer mas diminuto la estructura de los trabajos.

En este trabajo se presenta dos algoritmos para la cooperación con dos robots móviles (un robot móvil líder y otro robot móvil seguidor) ambos robots con tracción diferencial, dichos algoritmos están diseñados por medio de un control clásico de lazo abierto y un

control proporcional en lazo cerrado. En la primera parte se presenta la simulación de los algoritmos en MATLAB, utilizando la herramienta Simulink y Simulink 3D Animation para visualizar a los robots en 3D. Para realizar los algoritmos de los robots fue necesario el estudio de los modelos cinemáticos de cada robot, para programar la trayectoria y control de cada uno de ellos, esto es fundamental para simular la cooperación entre los dos robots.

Para el movimiento de cada robot se uso la plataforma Arduino, el cual está diseñado para soportar diferentes módulos electrónicos como, por ejemplo, el control de motores, comunicación inalámbrica, Ethernet, entre otros. El robot móvil líder se presenta equipado con un módulo de antena XBee para su control inalámbrico, esto permite la comunicación con la computadora e interacción con un celular, dicho celular tiene una aplicación para manipular al robot usando la red de internet. La programación se realizó en el lenguaje de programación Processing. Es importante recalcar que el robot líder tiene una tarjeta de color negro en la parte de atrás para que sea detectado por el robot seguidor a través del sensor infrarrojo. El sistema del robot móvil seguidor es manipulado por medio de un sensor infrarrojo que se encuentra en la parte frente-centro de dicho robot, a través de dicho sensor se estará monitoreando la distancia entre ambos robots. La salida del sensor cuando detecta a la tarjeta de color negro a una distancia de 7 centímetros nos proporciona 40 milivolts, ese dato es nuestra variable de control a emplear en la programación, para que el robot seguidor se mantenga detrás del robot líder en cada movimiento en línea recta que este realice.

1.1. ESTADO DEL ARTE

Se tienen referencias de artículos y proyectos donde se realizan diferentes sistemas de control y cooperación en robots móviles, a continuación presentamos algunos ejemplos.

Robot Móvil de Tracción Diferencial con Plataforma de Control Modular para Investigación y Desarrollo Ágil de Proyectos [2]. En este trabajo se presenta una plataforma de control modular que tiene la capacidad para que ser utilizado en proyectos que involucren temas como inteligencia artificial, control y robótica. Para realizar el proyecto se uso un auto con tracción diferencial (ver fig. 1.1). Para el control del robot se hace el uso de la plataforma Arduino, dicha plataforma es ideal para montar módulos que contengan los componentes y actuadores para realizar el proyecto, también tiene la capacidad de comunicación inalámbrica entre otros. Para el control inalámbrico del robot, el sistema se presenta con un módulo de XBee. Se le monto un a iPod Como sistema de

visión, para la transmisión inalámbrica de video por medio de la aplicación FaceTime. Para el control manual tanto para la realización de trayectorias predefinidas se presenta una interface grafica. La calidad con que las trayectorias son realizadas en relación a las trayectorias ideales es puesta a prueba.

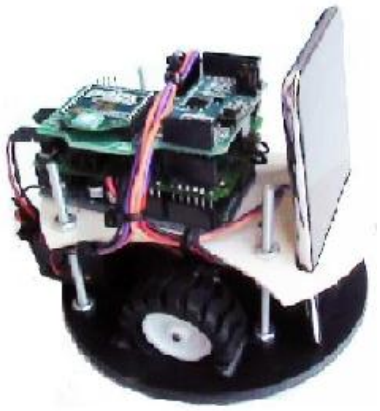


Fig. 1.1 Robot Móvil con Arduino y un IPod para la transmisión de video inalámbrico por WiFi[2].

MODELADO Y CONTROL DE UN ROBOT MÓVIL TIPO NEWT EN LA TAREA DE SEGUIMIENTO DE TRAYECTORIA [3]. En la realización de este trabajo se uso un robot móvil de ruedas tipo Newt, este tipo de robot permite presentar una derivación del modelo cinemático de dicho robot, para ello se ocupó la teoría Lagrangiana de la mecánica clásica. Se propuso un control que se basa en linealización de entrada-salida esto permite llevar a las variables de estado (x, y, ϕ) para que sigan la trayectoria nominal esto ($\dot{x}, \dot{y}, \dot{\phi}$) se da si el móvil cuando inicia su trayectoria se encuentra sobre un punto de dicha trayectoria ($\dot{x}, \dot{y}, \dot{\phi}$), para no generar más controversia se eligió el punto de inicio (0, 0, 0), este control es posible si se considera el modelo cinemático del móvil obtenido. Usando el modelo cinemático del robot y el control antes mencionado es como se lleva a cabo la tarea de control de seguimiento de trayectoria del móvil. Para verificar el desempeño de la estrategia de control propuesta para el robot móvil tipo Newt se verifica mediante simulaciones computacionales.

UNA ARQUITECTURA DISTRIBUIDA PARA EL CONTROL DE ROBOTS AUTÓNOMOS MÓVILES (2001) [4]. En este proyecto el autor propone la implementación de una arquitectura distribuida para robots esto permite que el robot bajo esa arquitectura realice tareas de forma autónoma en un entorno no conocido para dicho robot y además

con incertidumbre, esto se lleva a cabo en un entorno web para que el robot móvil realice la tele operación en tareas de exploración.

La realización de este proyecto se baso en el uso de algoritmos genéticos, lógica difusa y también redes neuronales estás disciplinas permiten que el robot realice tareas como: aprendizaje, interpretación del entorno, realización de mapas, planificación de trayectorias y localización. La implementación de la arquitectura, los algoritmos y el uso de una plataforma móvil tipo diferencial (usando un soporte de ruedas tipo castor), se pudo obtener el robot el robot Quaky-Ant (ver fig. 1.2), si observamos la figura también se hace el uso de ultrasonido, así como de una cámara RGB para retroalimentación visual. Este robot tiene aplicaciones útiles para fines de orientación, que otros robots comunes no tienen, estas aplicaciones son: Un compas electrónico que mide la orientación del robot con respecto al polo magnético de la tierra para compensar errores de edometría en interiores y un receptor GPS para posicionamiento absoluto, que se utiliza para compensar los errores de edometría en entornos exteriores.

Fig. 1.2 Robot Quaky-Ant[4].

CONTROL DE SEGUIMIENTO DE TRAYECTORIA Y FORMACIÓN DE ROBOTS MÓVILES [5]. Este artículo presenta el control de seguimiento de trayectoria y la formación de sistemas multi-robot, utilizando MATLAB Simulink 3D Animation. Se utilizó un control o lineal basado en las técnicas de linealización de entrada-salida para controlar el seguimiento de la trayectoria y la formación de robots. Los resultados muestran tres caminos diferentes, una línea recta, curva hipérbola y trayectoria de un semicírculo, para ello se mantiene la formación triangular simple de múltiples robots (como se muestra en la figura 1.3). En este caso, el control se hace para tres robots móviles, pero este tipo de leyes de control se puede utilizar para gran número arbitrario de robots y generar diferentes formaciones.

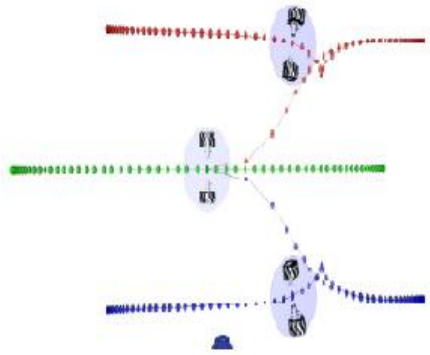


Fig. 1.3 Formación de Robots [5].

LIGA DE ROBOTS [6].

Esta área de la robótica, se ha hecho popular a tal grado que existen centros de investigación que realizan ligas de robots que tengan la capacidad de cooperar entre ellos para vencer al oponente. Las ligas más importantes en esta competencia son la F180 y la Middle SIze Robot League. La liga F180, se le da ese nombre ya que cada robot que esta compitiendo debe caber y permanecer en un área de 180 mm para poder competir. Las ligas están basados en una cámara global que cada jugador tiene acceso a ella, dicha cámara entrega información al sistema de la posición de su equipo (ver figura 1.4), del oponente y la pelota, de esa manera un agente central toma las decisiones sobre el control. La Middle Size Robot Legue cuenta con robots mas grande, completamente autónomos. La diferencia con las ligas anteriores es que estos tienen su propia visión y no es necesario de un agente externo para la coordinación entre los robots.

Fig. 1.4 Robots Cooperativos: Liga de robots [6].

Lo que aquí se presenta como proyecto es la cooperación con dos robots móviles tipo diferencial, dicho proyecto consiste: el robot móvil seguidor, como su nombre lo indica seguirá al robot móvil líder. Para que el robot seguidor haga su tarea se programo un algoritmo de control proporcional de lazo cerrado por medio de la plataforma Arduino, en el que se le monto un sensor infrarrojo, el cual los datos obtenidos por el sensor, es la variable de control retroalimentado (dicha plataforma Arduino y sensor están montados sobre el robot). Los datos que envía el sensor (datos analógicos) es obtenido al detectar una tarjeta de color negro que esta detrás del robot móvil líder, el Arduino recibirá el valor censado del color negro, dato que en la programación permitirá que el robot seguidor se mantenga detrás del robot líder para seguirlo en cada movimiento que este realice. Al robot líder se le programo un algoritmo de control de lazo abierto, por medio de la plataforma Arduino, el cual se le monto un módulo de XBee para la comunicación inalámbrica. Para manipular dicho robot se uso un celular el cual se le instalo la aplicación TouchoOSC bridge, para que dicho celular interactúe con la computadora y la red de internet inalámbrica, para la programación del algoritmo se uso el lenguaje de programación Processing ya que en dicha programación nos da la opción de programar graficas para visualizar en tiempo real la interacción celular-computadora-robot.

La simulación se hizo en el programa MATLAB utilizando la herramienta Simulink y Simulink 3D Animation para visualizar a los robots en 3D. Para realizar los algoritmos de los robots fue necesario el estudio de los modelos cinemáticos de cada robot, para programar la trayectoria y control de cada uno de ellos, esto es fundamental para simular la cooperación entre los dos robots.

El proyecto se presenta con principios básicos de control, sensores sencillos fáciles de manipular, plataforma Arduino con software libre, un módulo XBee, puente H y demás circuitos para realizar la placa. Respecto a los algoritmos empleados en los dos robots físicos no se hace el uso de las ecuaciones cinemáticas de cada robot, simplemente un algoritmo de control clásico de lazo cerrado (para el robot seguidor) por medio de la entrada de los datos emitidos por el sensor infrarrojo y un algoritmo de control de lazo abierto (para el robot líder), es una forma didáctica para empezar a conocer el área de la robótica.

1.2. JUSTIFICACIÓN

El presente proyecto se realizo con la finalidad de aplicar algoritmos que nos permitiera realizar la cooperación entre dos robots usando control clásico. Las partes que conforman el sistema, son fáciles de conseguir aun costo no elevado, el software usado es libre y la programación fácil de aprender para los que están iniciando en el ámbito de la programación, el control clásico usado esta basado en lo elemental para empezar a aprender los principios de control. Este proyecto es el principio no tan complejo en robots cooperativos, es una forma didáctica de entrar en el mundo de la robótica: control, electrónica, software, algoritmos, control difuso, redes neuronales modelos dinámicos y cinemáticos etc., para entender cada concepto antes mencionado es importante empezar desde lo elemental para volverse experto en dicha área.

Sin olvidar que los robots desde su idealización y puesta en prácticas están diseñados para el servicio del hombre y de sustituirlo en tareas comunes y peligrosas tales como:

- La desactivación de bombas
- Realización de traslados de cargas pesadas que representan trabajo rudo
- Exploración de espacios hostiles para el hombre
- Expediciones espaciales, marinas o desconocidas

Por ahora los resultados de este trabajo no serán llevados a la práctica con robots que resuelvan las tareas antes mencionadas, ya que se necesitan materiales más robustos, pero si es el principio para una visión hacia futuro.

1.3. OBJETIVOS

1.3.1. OBJETIVO GENERAL

Diseñar e implementar dos algoritmos para la cooperación entre dos robots móviles de tipo diferencial para que un robot sigua la trayectoria en línea recta del otro robot.

1.3.2. OBJETIVOS ESPECÍFICOS

- Realizar la programación de un algoritmo en Arduino de un control clásico proporcional de lazo cerrado para que siga una tarjeta de color negro por medio de un sensor infrarrojo

- Diseñar una placa para tener acceso a las entradas y salidas de la tarjeta Arduino y poder montar el puente H, XBee y alimentar el sensor infrarrojo y motores.

- Realizar la programación de un algoritmo en Processing del control clásico de lazo abierto usando el módulo XBee, para controlar un robot de forma inalámbrica por medio de un celular.

- Acondicionar y diseñar un circuito electrónico para el uso de los sensores infrarrojos.

- Llevar a cabo la simulación en MATLAB de dos algoritmos uno para cada robot para realizar la cooperación de los dos robots móviles, para que un robot siga al otro robot, usando la herramienta Simulink.

1.4. CARACTERIZACIÓN DEL ÁREA EN QUE TRABAJO

El proyecto se llevó a cabo en las instalaciones de la Universidad Politécnica de Pachuca. Carretera Pachuca-Cd. Sahagún, km 20, Ex-Hacienda de Santa Bárbara, C.P. 43830 Zempoala, Hidalgo. Teléfono 01 771 547 7517

El lugar principal de trabajo fue en el área del Laboratorio de Investigación, llamado Laboratorio de Investigación en Robótica y Electrónica Avanzada (LIREA), donde se me asigno un área de trabajo, en uno de los cubículos de dicho Laboratorio de la Universidad.

Misión: La UPP es una institución educativa que proporciona educación superior y de posgrado, realiza actividades de investigación y extensión, así como servicios de calidad en áreas tecnológicas; que forma integralmente profesionistas emprendedores e investigadores altamente calificados y con valores universales a través de programas educativos pertinentes, basados en competencias; con programas de investigación, desarrollo tecnológico, vinculación y extensión que contribuyen al desarrollo social y económico sustentable del país.

Visión: La UPP es una institución con programas educativos acreditados, pertinentes para el desarrollo del Estado y del País; con laboratorios y talleres certificados; que proporciona una sólida formación tecnológica, científica, humanista y bilingüe; con valores universales, éticos, de respeto al medio ambiente y a la diversidad cultural; que realiza investigación y desarrollo tecnológico en áreas de conocimiento que son estratégicas para el desarrollo de los sectores social, público y privado; con programas de extensión orientados a elevar, proteger y promover tanto la cultura como la calidad de vida de la sociedad.

Fig. 1.5 Ubicación geográfica de la Universidad Politécnica de Pachuca. [7]

1.5. PROBLEMAS A RESOLVER

Diseñar un algoritmo en programación Processing, para un robot móvil diferencial (robot líder) que nos permita controlar a dicho robot a través de un celular de forma inalámbrica con la ayuda de una PC y a través de la red de internet inalámbrica, para hacer que el robot se mueva hacia adelante, atrás, derecha, izquierda y se detenga totalmente.

Por otro lado también se tiene un segundo robot móvil diferencial (robot seguidor) que se le programará un algoritmo que tenga la capacidad de seguir al robot líder en los movimientos que este haga. Para que este siga al robot líder se usara un sensor infrarrojo.

Para ambos robots se usará un control clásico, se elegirá el adecuado para que cada robot realice la tarea particular antes mencionado.

Otra tarea a realizar es la simulación por medio de la herramienta MATLAB, en el que es necesario el estudio de sistemas de control, modelo cinemático y demás para simular la cooperación entre los dos robots.

1.6. ALCANCES Y LIMITACIONES

El algoritmo diseñado para el robot móvil líder tiene la capacidad para interactuar con un celular usando la red de internet inalámbrica y eso nos ayudara a controlar el robot realizando el desplazamiento que el operador decida ya sea adelante, atrás, izquierda, derecha o que se pare totalmente. La comunicación del celular con la computadora nos permite ver gráficamente el voltaje aplicado al robot móvil líder.

El algoritmo programado al robot móvil seguidor tiene la capacidad de seguir al robot móvil líder cuando este se mueva en línea recta, hacia delante o hacia atrás, por medio de un sensor infrarrojo localizado en el centro-frente de dicho robot.

La manipulación del robot móvil líder solo tiene un alcance máximo posible: 120 metros en exteriores (línea de vista) y 30 metros en interiores. El algoritmo programado al robot móvil seguidor mantendrá a dicho robot a una distancia de 7 cm con respecto al robot líder y puede programarse hasta una distancia máxima de 12 cm ya que el sensor infrarrojo (modelo QRD 1114) es muy sensible a la luz del ambiente perjudicando las medidas, dando lugar a errores y por tal motivo el robot no tiene la capacidad de seguir al robot líder cuando este se mueva hacia la derecha o izquierda. Se hizo la prueba de colocar 2 sensores más en la parte de enfrente del robot, uno a la derecha y otro a la izquierda, estos perdían sus referencias por los motivos antes mencionados, por eso se limito a que el robot seguidor solo siguiera al robot líder en una trayectoria de línea recta. Existen sensores infrarrojos más eficientes que hacen caso omiso a la luz del ambiente ya que tienen integrado un circuito modulador de luz, pero son caros con precios alrededor de $400 pesos cada uno.

Los algoritmos empleados para la simulación en MATLAB, se baso en las ecuaciones dinámicas de los robots, modelos de control y el estudio de sus trayectorias, estos algoritmos no se programaron directamente hacia la plataforma Arduino ya que es necesario el uso de encoders y aplicación de conocimientos reales en mecánica, por eso se opto en asimilar dicha simulación a lo real por medio de lo explicado en los dos primeros párrafos de este capitulo.

CAPÍTULO 2

FUNDAMENTO TEÓRICO

2.1. ROBOT MÓVILES

DEFINICIÓN

Un robot móvil es un dispositivo que se forma con componentes físicos y computacionales el cual se dividen en 4 subsistemas:

-Locomoción - Razonamiento
-Percepción - Comunicación

Robot es una palabra checa (robota) que viene de la traducción al inglés que significa: servidumbre, trabajo forzado. La palabra robots se uso por primera vez en una obra del escritor dramaturgo checo Karel Capek en 1921, en su obra que se tituló R.U.R (Rossumoví Univerzál ní Roboti), que si se traduce al español es: Robots Universal de Rossum. Después se unió Isaac Asimov escritor de origen ruso quien también se intereso por el género de la ciencia-ficción y propone la palabra robótica y la definió como la ciencia que estudia a los robots. El establece tres leyes, que los robots creados en su obra deben seguir:

- Un robot debe ser capaz de cuidar al ser humano y aun en su inacción no debe dejar que sufra daño
- Un robot debe estar capacitado para obedecer las órdenes de un ser humano, pero si en las ordenes atentan contra la primera ley no debe actuar.
- Un robot siempre debe cumplir las dos primeras leyes y en la protección de su existencia siempre y cuando no entre en conflicto con las dos leyes anteriores.

2.1.1. Arreglos de Ruedas

Existen varias formas de diseñar las ruedas puestas en los robots, las más comunes son: Diferencial, Síncrono, Tipo triciclo y Tipo carro.

DIFERENCIAL

El diseño de este tipo de robot consiste de dos ruedas unidas por un único eje, en este diseño cada rueda se controla de forma independiente, los movimientos que estos realizan son: Línea Recta, En arco y Vuelta sobre su propio eje.

Para que el robot se mantenga estable se usan una o dos ruedas adicionales llamadas caster. Si el robot tiene 3 ruedas se el conoce con el nombre de triangulo, el problema de este tipo de esquema es la posible inestabilidad, y si el robot tiene 4 llantas se le conoce como diamante pero como existe perdida de contacto de las ruedas de tracción es necesario el uso de un sistema de suspensión, en la figura 2.1 vemos el esquema del robot.

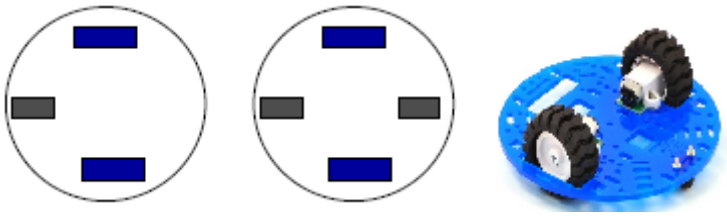


Fig. 2.1 Robot Móvil tipo Diferencial [8].

SÍNCRONO

El diseño de este tipo de robot, sus ruedas se mueven de forma síncrona, es decir, al mismo tiempo. La posición de las ruedas tiene dicha estructura para que siempre apunten a la misma dirección y para que al dar vuelta giren sobre el eje vertical, eso permite que la dirección de la estructura se mantenga y para eso se requiere de un mecanismo adicional para mantener el frente del chasis (estructura del robot) en la dirección de las ruedas.

En la figura 2.2 se aprecia el movimiento del robot por medio del eje de rotación de las dos ruedas delanteras. La ventaja de usar el sistema síncrono es que evitan los problemas de inestabilidad, pérdida de contacto del diferencial pero tienen mayor complejidad mecánica (eje de giro).

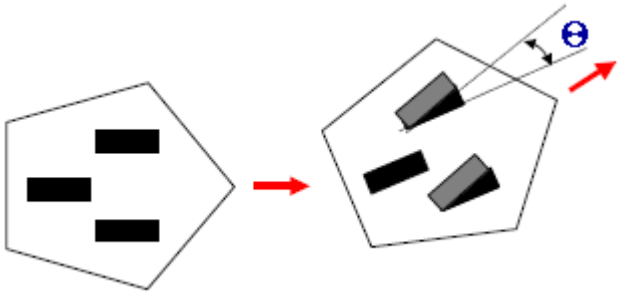


Fig. 2.2 Esquema del Robot Móvil tipo Síncrono [8].

TRICICLO

Los triciclos tienes dos ruedas fijas (ver figura 2.3) que permiten la tracción, también cuentan con una rueda delantera que da la dirección al robot, esta por lo regular no tiene tracción. La ventaja de este sistema es que tienen buena estabilidad y simplicidad mecánica, tiene facilidad para ir recto pero su cinemática es más compleja.

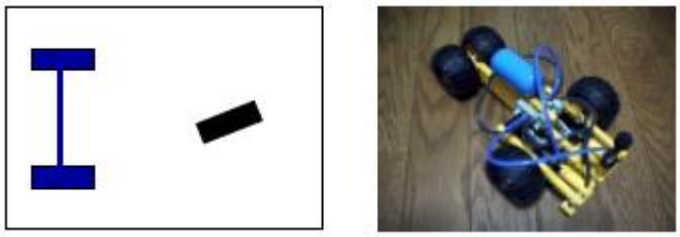


Fig. 2.3 Robot móvil tipo triciclo [8].

CARRO

Este sistema es muy parecido al del triciclo con la diferencia que esta cuenta con dos ruedas para tracción y dos ruedas para la dirección del robot, dicho sistema se observa en la figura 2.4. Respecto a las desventajas es que tiene una mayor complejidad mecánica que el triciclo, por que existe un acoplamiento entre las 2 ruedas de dirección otra desventaja es su complejidad cinemática. Las ventajas que posee es que tiene buena estabilidad y facilidad de ir derecho.

Fig. 2.4 Robot móvil tipo carro [8].

OMNIDIRECCIONAL

Este sistema cuenta con 3 ruedas posicionadas a 120° tal como se muestra en la figura 2.5. Dichas ruedas tienen la capacidad de girar en ambos lados eso permite un control lineal mas simplificado que en la estructura del robot diferencial. En las ligas de futbol de robots F-180 los campeones usan este tipo de ruedas.

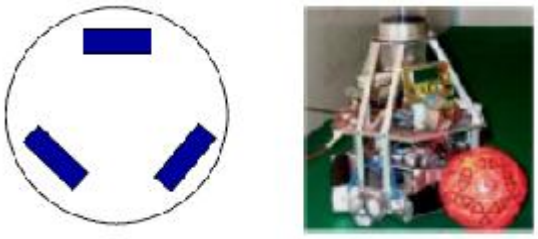


Fig. 2.5 Robot móvil tipo omnidireccional [8].

2.2 SISTEMAS DE ROBOTS COOPERATIVOS

El área de los robots cooperativos es aun nueva en el estudio de la robótica moderna. A pesar de que su estudio apenas comienza a desarrollarse, ya existen grandes cantidades de artículos publicados, en el cual se tratan estos sistemas desde varias perspectivas, en el cual los relacionan desdela electrónica hasta la biología y la psicología.

Si ocupáramos un solo robot para realizar una tarea se tardaría más que si la tarea lo hiciera con un sistema distribuido de robots [6]:

- Existen tareas que por más que las queramos simplificar para que un robot lo realice, siempre es necesario la intervención de dos o más robots, debido a la complejidad o a las limitaciones espaciales.
- Si necesitamos resultados de eficiencia y efectividad, cuando existe la necesidad de explorar y recabar información, es posible cuando se ocupa un sistema de múltiples robots, los cuales pueden abarcar un área más extensa.
- Las actividades tales como el movimiento de cargas, es evidente que un solo robot puede realizarlo pero mas lentamente, pero el ocupar un conjunto de robots puede efectuarse de forma más flexible según la magnitud de la carga.
- El uso de múltiples agentes es una ventaja, porque si alguno de ellos tiene problemas técnicos o de programación cuando están realizando alguna tarea, el objetivo final aun se puede lograr con los robots que queden. La desventaja de un robot único, su falla significa la imposibilidad de continuar y finalizar la tarea.

Parece todo fácil de realizar si queremos tener las ventajas al usar múltiples agentes, pero no es cierto, esto tiene alto grado de complejidad, tales como: la complejidad derivado de los sistemas distribuidos (ya sea redes o bases de datos) estos recursos son limitaciones ya que deben ser manejados para optimizar el resultado final, otra limitación es el diseño de la arquitectura útil para resolver problemas, sin que estos pierdan flexibilidad. Otra complejidad consiste cuando el agente autónomo sale al ambiente dinámico donde las condiciones pueden cambiar, es allí donde el algoritmo programado debe ser capaz de adaptarse a las condiciones del ambiente. La complejidad antes mencionada nos lleva a otros problemas adicionales como son: evitar colisiones en la trayectoria del robot y problemas referidos a la relación geométrica.

El objetivo de un sistema de cooperación entre robots es enfocarse a realizar una o varias tareas el cual persigue un objetivo final, eso permite la eficiencia y utilidad en comparación si solo uno lo hiciera o no hubiese cooperación entre ellos.

Realizar un sistema de cooperación entre robots genera costo, dificultad de construir y mantener un gran número de robots autónomos, eso permitió que en un principio el estudio de estos sistemas se limitaran a la teoría y simulaciones. Pero ahora que la tecnología ha ido avanzando ya se puede encontrar sistemas de cooperación aplicado para tareas reales.

2.3 SISTEMAS DE CONTROL

Los sistemas de control se dividen en dos tipos:

2.3.1 Sistema de control en lazo abierto

Cuando en un sistema su salida no afecta a la acción de control recibe el nombre de control de lazo abierto, dicho de otra forma la salida de este sistema no se mide y ni hay necesidad de realimentarla para compararla con la entrada. Veamos un ejemplo para entender dicho concepto, en una lavadora, el remojo, el lavado y el centrifugado en la lavadora operan con una base de tiempo. La salida del sistema es la ropa y esta no mide si la ropa esta limpia o sucia.

Los sistemas de control de lazo abierto sus salidas no necesitan compararse con la entrada de referencia, eso significa que a cada entrada de referencia le corresponde una condición de operación fija; si queremos buenos resultados necesitamos precisión en el sistema y para ello es necesario una buena calibración del sistema. La desventaja de este sistema es que si se presenta una perturbación la tarea deseada no se realiza; si queremos llevar a la práctica este sistema los requisitos son: conocer la relación entre la entrada y la salida y que no haya perturbaciones internas ni externas. La figura 2.6 nos muestra el diagrama a bloques del sistema.

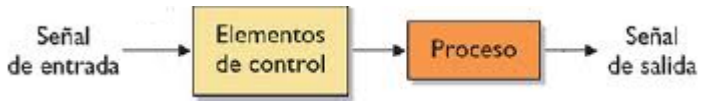


Fig. 2.6 Diagrama a bloques de un control de lazo abierto [9].

2.3.2 Sistema de control en lazo cerrado

Si un sistema tiene una relación entre la salida y la entrada de referencia y si estas se comparan y el resultado de la diferencia se usa como medio de control, entonces estamos hablando de un sistema de control de lazo cerrado (ver figura 2.7). Un ejemplo seria el sistema de control de temperatura de una incubadora de huevos, el proceso es medir la temperatura real y compararla con la referencia (temperatura deseada), entonces el termostato activa o desactiva el equipo de calefacción o enfriamiento para asegurar que la temperatura de la incubadora se mantiene en un nivel adecuado para la incubación de los huevos, independientemente de las condiciones externas.

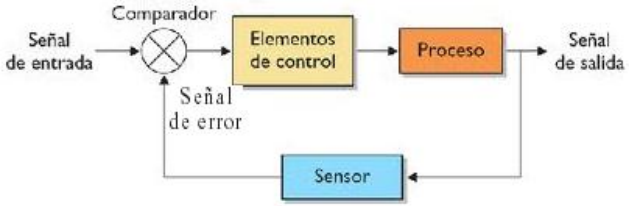


Fig. 2.7 Diagrama a bloques de un control de lazo cerrado [9].

Control Proporcional

El control proporcional consiste en que la salida del controlador es proporcional a la magnitud de error.

Este tipo de sistema se basa en un algoritmo lineal y proporcional, su principal objetivo es reducir la magnitud de error para dar estabilidad al proceso que se esta realizando.

Su característica principal es la presencia del offset, ya que solo reduce el error (no lo elimina).

Para aplicar este tipo de control tenemos una ecuación para su aplicación, si no se puede optar por diseñar un control adecuado a sus necesidades.

Ecuación: $y = Kc * e + y_0$

Donde:

y = salida del controlador

Kc = ganancia proporcional

e = error

y_0 = condiciones iniciales

2.4 MODELO CINEMÁTICO DE UN ROBOT MÓVIL TIPO DIFERENCIAL

En la figura 2.8 se observa la posición y orientación de un robot tipo diferencial. La posición puede describirse por las coordenadas (x, y) con respecto a un sistema de referencia fijo y la orientación por medio del ángulo φ que el robot forma con respecto del eje X_0.

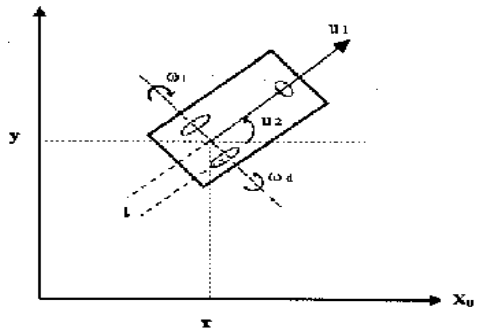


Fig. 2.8 Localización en el plano cartesiano de un robot móvil tipo diferencial [10].

Observemos nuevamente la figura 2.8, donde se encuentra u_1 el cual es la velocidad lineal del robot en la dirección perpendicular al eje de sus ruedas y también encontramos u_2 el cual es su velocidad angular. Entonces a partir de la figura 2.8, es posible encontrar las siguientes relaciones

$$\dot{x} = u_1 cos\varphi$$

$$\dot{y} = u_1 sin\varphi \qquad (1)$$

$$\dot{\varphi} = u_2$$

Donde x representa la posición a lo largo del eje X_0, y representa la posición a lo largo del eje Y_0 y φ representa la orientación del eje longitudinal del vehículo con respecto al eje X_0. Si consideramos que el sistema es ideal entonces la velocidad lineal u_1 y la velocidad angular u_2 pueden ser consideradas como variables de control (1). Pero si el análisis lo vemos desde el punto de vista real es necesario realizar la siguiente consideración: que dichas variables son funciones de las velocidades angulares de las ruedas del robot.

Consideremos las velocidades angulares de las ruedas derecha e izquierda del vehículo móvil y les daremos el nombre de w_d y w_i respectivamente. Entonces, es posible decir que u_1 y u_2 están relacionadas con w_d y w_i por medio de la siguiente matriz de trasformación.

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = T \begin{bmatrix} w_d \\ w_i \end{bmatrix}, T = \begin{bmatrix} \frac{r}{2} & \frac{r}{2} \\ \frac{r}{2l} & -\frac{r}{2l} \end{bmatrix} \qquad (2)$$

Donde r es el radio de las ruedas y 2l es la distancia entre ellas (como se muestra en la figura 2.8). Si nos damos cuenta la transformación T es no singular, eso significa que cualesquiera valores de las velocidades u_1 y u_2 pueden ser obtenidas mediante una adecuada selección de las variables w_d y w_i. Esto nos lleva a la representación alterna para el sistema (1) dada por.

$$\dot{x} = \frac{r}{2}(w_d + w_i)cos\varphi$$

$$\dot{y} = \frac{r}{2}(w_d + w_i)sin\varphi$$

$$\dot{\varphi} = \frac{r}{2l}(w_d - w_i)$$

2.5 PLATAFORMA ARDUINO [10]

Arduino es una plataforma de hardware libre, su construcción contiene una placa con un microcontrolador y un entorno de desarrollo, esta placa esta diseñada para facilitar el uso de la electrónica en múltiples proyectos. Su diseño es de libre distribución y utilización, que incluso podemos construirlo nosotros mismos.

Para programar el microcontrolador que contiene la placa Arduino, se hace mediante el lenguaje de programación Arduino basado en Wiring, el entorno de desarrollo Arduino que se basa en Processing y el cargador de arranque (bootloader) que corre en la placa. En la figura 2.9 puede se observa como es físicamente la placa Arduino UNO.

Partes principales que componen la tarjeta Arduino UNO.

1.- Conexión USB: Utilizada para la comunicación con computador.

2.- Alimentación: Plug para alimentar al Arduino cuando no este conectado al ordenador.

3.- Chip de comunicación: Chip de comunicación entre la computadora y el Arduino.

4.- Cristal de 16MHz: componente que permite el funcionamiento del microcontrolador.

5.- Conexiones Digitales: Funcionan tanto como entradas o salidas, aquellos con este símbolo "~" en frente son salidas PWM.

6.- Led: Esta conectado al pin 13, sirve para hacer pequeñas pruebas sin necesidad de conectar nada al Arduino

7.- Leds TX/RX: Indican que el Arduino se comunica con el ordenador

8.- Microcontrolador ATMEGA328: Es el cerebro del Arduino

9.- Barra de energía: Proporciona una fuente de energía para la alimentación de pequeños dispositivos externos u otra circuitería.

10.- Pines (TX/RX): para la comunicación en serie con dispositivos externos
11 – Indicador LED encendido - Indica cuando el Arduino está conectada a una fuente de alimentación
12 Botón RESET - Reiniciar el Arduino, comenzando su programa desde el principio
13 – Entradas analógicas - Entradas que podemos conectar potenciómetros u otros componentes analógicos

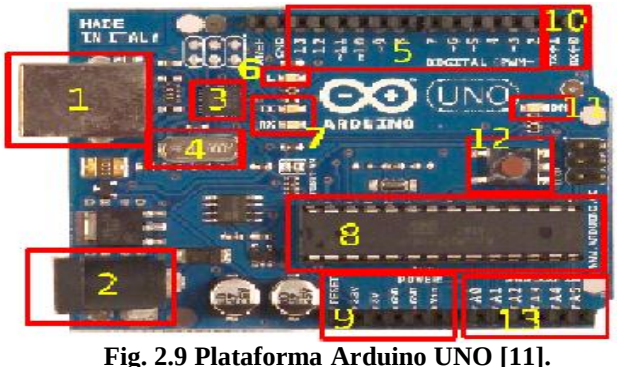


Fig. 2.9 Plataforma Arduino UNO [11].

Especificaciones de Entradas y Salidas de la Placa Arduino UNO.
La estructura del Arduino consta de 6 entradas analógicas que proporcionan una resolución de 10 bits y 14 entradas digitales, que operan a 5V y pueden proporcionar o recibir como máximo 40mA; dichas entradas se pueden configurar ya sea como entrada o salida. Alguno de los 14 pines pueden proporcionar una salida PWM (modulación por ancho de pulso), esos pines son 3, 5, 6, 8, 10 y 11. Si se conecta cualquier dispositivo a los pines 0 y 1, eso interferirá con la comunicación USB.

Entorno de Programación en Arduino.
Como la plataforma es de software libre, si queremos programar la placa es necesario descargarse de la página web de Arduino el entorno de desarrollo (IDE). En dicha página encontraras versiones para Windows y para MAC, y también las fuentes para compilarlas en LINUX.
Para escribir el código de la programación el entorno de desarrollo Arduino cuenta con un editor de texto, un área de mensajes: aquí se muestra información mientras se corren los programas y también muestra errores, una consola de texto: muestra la salida para el entorno Arduino y también mensajes error del código de la programación y otras informaciones, una barra de herramientas: que permite verificar el proceso de carga, creación, apertura y guardado de programas, y la monitorización serie con botones y por ultimo una serie de menús (como se muestra en la figura 2.10). Cuando se abre un nuevo programa, Arduino utiliza para escribir el software lo que denomina “Sketch (programa)”.

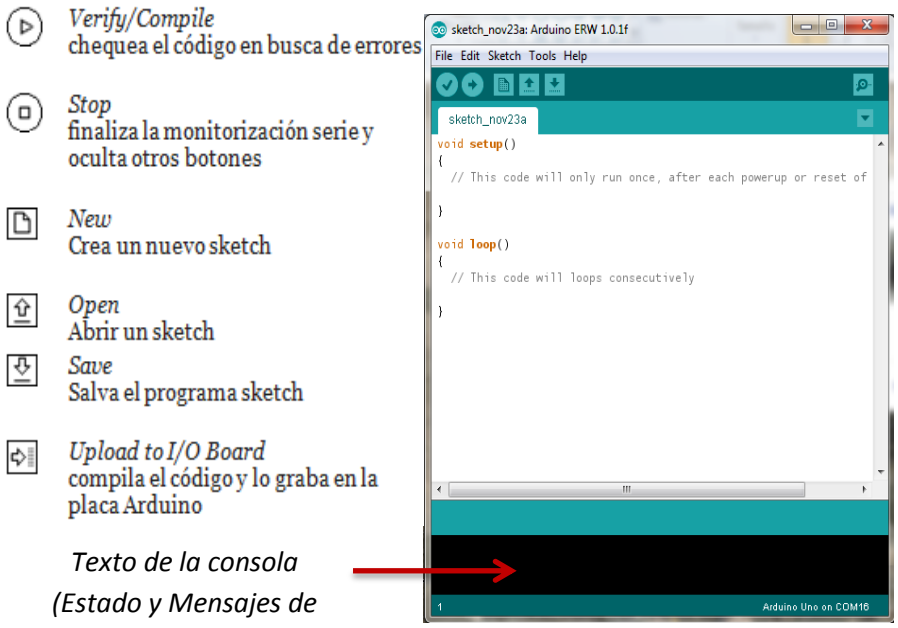


Fig. 7.10 Entorno Programación en Arduino [11].

Lenguaje de programación Arduino.

El lenguaje de programación en Arduino se basa en C/C++ y es posible cargar las construcciones de C estándar y algunas funcionalidades de C++. Permite relaciones con la librería AVR Libc y permite el uso de todas sus funciones, eso permite un entorno de programación multimedia.

La estructura básica de programación de Arduino divide la ejecución en dos partes: la función Setup() el cual constituye la preparación del programa esta función se ejecuta una única vez y es empleada para configurar el pinMode (por ejemplo si queremos que un pin digital funcione como entrada o salida) y permite inicializar la comunicación serie, la otra parte de la estructura es la función Loop() allí se da la ejecución del programa y se escribe el código a ser ejecutado continuamente (leyendo las entradas de la placa, salidas, etc.).

A continuación se presenta un ejemplo básico para entender la estructura y lenguaje de programación en Arduino, que consiste en apagar y encender un led con retardos de un segundo.

```
const int led = 13                    // etiqueta del pin 13, con el nombre de led
void setup() {
  pinMode(led, OUTPUT);               // Establece al pin 13 = led como salida
}
```

```
void loop() {
  digitalWrite(led, HIGH);          // Activa o pone en alto al pin 13=led
  delay(1000);                      // Pausa un segundo (1000 milisegundos)
  digitalWrite(led, LOW);           // Desactiva o pone en alto al pin 13=led
  delay(1000);                      // Pausa un segundo (1000 milisegundos)
}
```

Si observamos el código ejemplo cada instrucción acaba con (;) y los comentarios se indican con //. Así como en C se introducen bloques de comentarios también en la programación Arduino es posible por medio de: /* ... */.

2.6 SENSOR INFRARROJO

La característica de un sensor es que es un dispositivo eléctrico, mecánico, químico, que recibe información del ambiente en una medida cuantizada por lo regular lo refleja por medio de un nivel de tensión. El sensor infrarrojo es un dispositivo electrónico que en su campo de visión tiene la capacidad de medir la radiación electromagnética infrarroja de los cuerpos. Los cuerpos tienen un espectro de luz que refleja una cierta cantidad de radiación, los sensores infrarrojos son capaces de ver la radiación que se encuentran por debajo de la luz visible que para nosotros los humanos no es posible.

Principio de funcionamiento:

El funcionamiento del sensor consiste en: los rayos infrarrojos (IR) inciden dentro del fototransistor (este tiene un material piro eléctrico normalmente formando una lámina delgada dentro del nitrato de galio, nitrato de Cesio y ftalocianina de cobalto. Los transistores normalmente se encuentran formados en diversas configuraciones tales como: 1,2,4 píxeles de material piro eléctrico.

Sensores pasivos: su construcción solo tiene un fototransistor que se encarga de medir las radiaciones emitidas por los objetos.

Sensores activos: Su construcción esta basado en un circuito integrado que contiene la combinación de un emisor diodo LED infrarrojos (IRED) y un receptor fototransistor, en el que la posición entre ellos es en paralelo y muy cercanos (como se muestra en la figura 2.11).

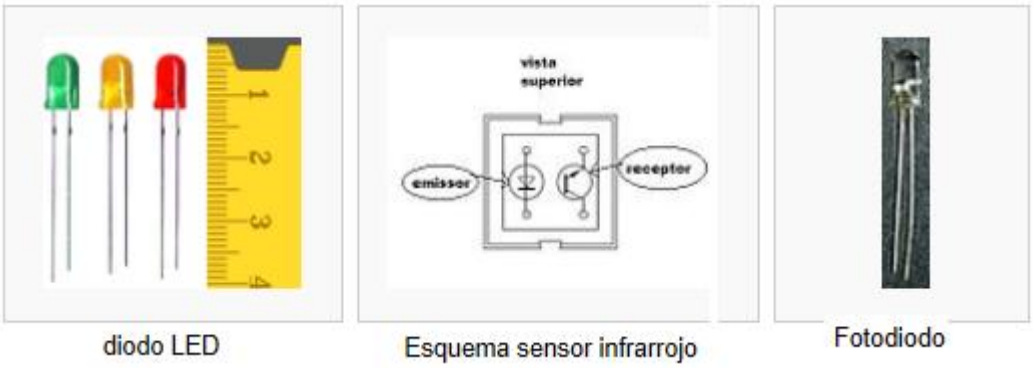


Fig. 2.11 Componentes de un sensor activo [12]

Clasificación según el tipo de señal emitida.

Sensores reflexivos: el diseño de este tipo de sensor se basa en una cara frontal en el que se encuentra tanto al LED como al fototransistor, su configuración permite medir la radiación proveniente del reflejo de la luz emitida por el LED. La desventaja de este tipo de sensor es que su configuración lo hace muy sensible a la luz del ambiente y eso permite obtener medidas erróneas, también el coeficiente de reflectividad del objeto se ve afectado dando lugar a que el sensor funcione diferente según el tipo de superficie

Sensores de ranura (Sensor Break-Beam): el principio de funcionamiento de este sensor es el mismo que el anterior, lo que hace la diferencia es la configuración de los componentes, estos se encuentran a la misma dirección una frente a la otra, de una ranura que por lo regular es estrecha, aunque existen también con ranuras grandes. Este tipo se sensor por lo regular se usa en control industrial y puede aplicarse en el control de las vueltas de un volante.

Sensores modulados: el funcionamiento de este sensor se basa también en el sensor de reflexión, solo que este contiene un circuito modulador en la emisión de la señal, eso permite la ventaja de reducir a un alto grado la influencia de la iluminación ambiental. El uso de estos sensores permite mejores resultados y más eficientes solo que su costo es mas elevado. Dichos sensores están orientados para ser aplicados a detección de presencia, medición de distancias, detección de obstáculos.

Sensores de barrido: Este sensor lo que lo hace único es que realiza el barrido horizontal de la superficie que esta reflectando y para ello usa señales moduladas para mejorar la independencia de la luz, el color o reflectividad de los objetos. Dicho sensor para que realice el barrido tiene un dispositivo de desplazamiento perpendicular al eje del objeto que se esta explorando, para que pueda obtener los datos de toda la superficie.

CAPÍTULO 3

PROCEDIMIENTOS Y DESCRIPCIÓN DE LAS ACTIVIDADES REALIZADAS

El desarrollo del proyecto empleado se baso en lo básico en control clásico, electrónica, parte de la mecánica y programación en Arduino. Para el desarrollo de los algoritmos empleados en los robots se hizo uso del control de lazo abierto y control proporcional de lazo cerrado.

3.1 DIAGRAMA A BLOQUES DEL FUNCIONAMIENTO DEL SISTEMA

En la figura 3.1 podemos observar el diagrama a bloques de los dos robots usados para el desarrollo del proyecto, donde se encuentran conectados a nuestra placa Arduino el sensor infrarrojo, el módulo de XBee, los puentes H, motores y su alimentación por medio de las baterías recargables. Y la interacción de la PC con el celular y los módulos de XBee.

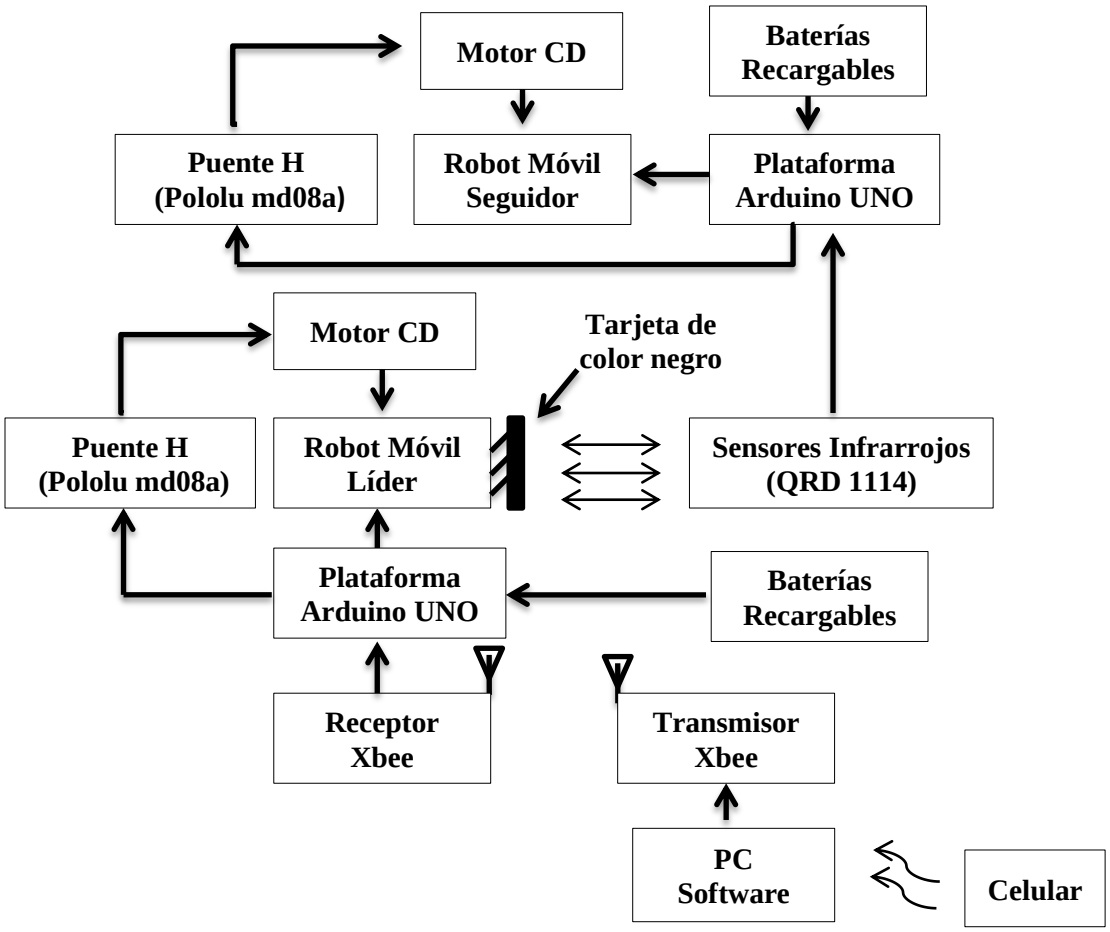


Fig. 3.1 Diagrama a Bloques de los Robots móviles cooperativos.

Plataforma Arduino UNO: Es el cerebro del robot ya que contiene un microcontrolador Atmega 328 P, el cual se programo para recibir y enviar datos para la acción de los robots. Alimenta al puente H, el sensor infrarrojo y los motores y la comunicación inalámbrica.

Motor CD: Estos motores permiten el desplazamiento de los robots. Son ideales para trabajar con robótica ya que es de escobillas que contiene imanes permanentes en el exterior y bobinas electromagnéticas montadas en el eje del motor. Su engranaje es de 50:1, trabaja con 6 V, a una velocidad de 700 rpm.

Modulo X-BEE-PRO S2: Es un protocolo de comunicaciones inalámbrico basado en el estándar de comunicaciones para redes inalámbricas IEEE. Las comunicaciones se realizan en la banda libre de 2.4 GHz, su alcance es de 100 metros en línea de vista y 30 metros en interiores. El módulo requiere una alimentación desde 2.8 a 3.4 V, la conexión a tierra y las líneas de transmisión de datos por medio del UART (TXD y RXD) para comunicarse con la tarjeta Arduino.

PC software: permite visualizar la interacción del celular y la programación Processing y analizar los datos que se esta enviando a través del celular.

Sensor Infrarrojo (QRD 1114): Sensor reflexivo optoacoplado, está diseñado para recibir datos de colores específicos, los datos obtenidos son analógicos que están en el orden de los milivolts. En iluminación interior, cerca de bombillas de incandescencia o halógenas los sensores pueden emitir lecturas erróneas debido a la emisión de infrarrojos.

Puente H (Pololu md08a): Permite el control de adelante y reversa de los motores. Se recomienda alimentar a motores de (VMOT): 4,5 V a 13,5 V. Requiere una alimentación de 2.7 a 5.5 Volts. Corriente máxima de salida: 3 A por canal, corriente de salida continua: 1 A por canal (puede conectar en paralelo para ofrecer 2 A continuos), Máxima frecuencia PWM: 100 kHz

Baterías Recargables: para la alimentación del Arduino, modelo 18650, se utilizan 4 de 3.6 volts a 3800 mAh cada una. Dos pilas para cada robot para soportar la demanda de voltaje y corriente de cada componente montado en la placa.

Celular: celular touch Samsung, el cual por medio de la aplicación TouchoOSC interactúa con la computadora y permite la manipulación del robot.

3.2 MATERIALES EMPLEADOS PARA EL DESARROLLO DEL PROYECTO

Para realizar la construcción de cada Robot Móvil Diferencial se utilizaron los componentes que se muestran en la Tabla 3.2.

Tabla 3.2 Lista de componentes.

Componentes	Cantidad
Plataforma Arduino	2
Programador de Xbee	1
Módulo Xbee	2
Puente H Pololu	2
Llantas de 42 mm de diámetro	4
Bola Castor de metal	2
Chasis o base para montar las piezas	2
Motor CD 6 V.	
Sensor Infrarrojo (QRD 1114)	3
Placa Fenólica 20X20	1
Baterías recargables de 3.6 volts	4
Computadora	1
Celular touch	1
Tarjeta de papel color negro 12X8 cm	1

Se usó el programa Fritzing para realizar el diseño de las pistas de cada robot el cual también nos da la opción de visualizar de manera física cada componente empleado. En la figura 3.3 se observa el diagrama de conexiones y diseño del robot móvil líder. Los materiales y componentes empleados para su construcción son: una placa Arduino, un puente h, dos módulos XBee (uno para el robot y el otro se encuentra en la tarjeta programadora para permitir la interacción inalámbrica entre la computadora, el robot y el celular), conectores hembra y macho, cables UTP, dos motores, un cable plug (para unir las baterías y alimentar a la tarjeta Arduino y los demás componentes y circuitos electrónicos), dos llantas, una bola castor, un chasis, dos baterías y el diseño de una placa para montar los componentes electrónicos.

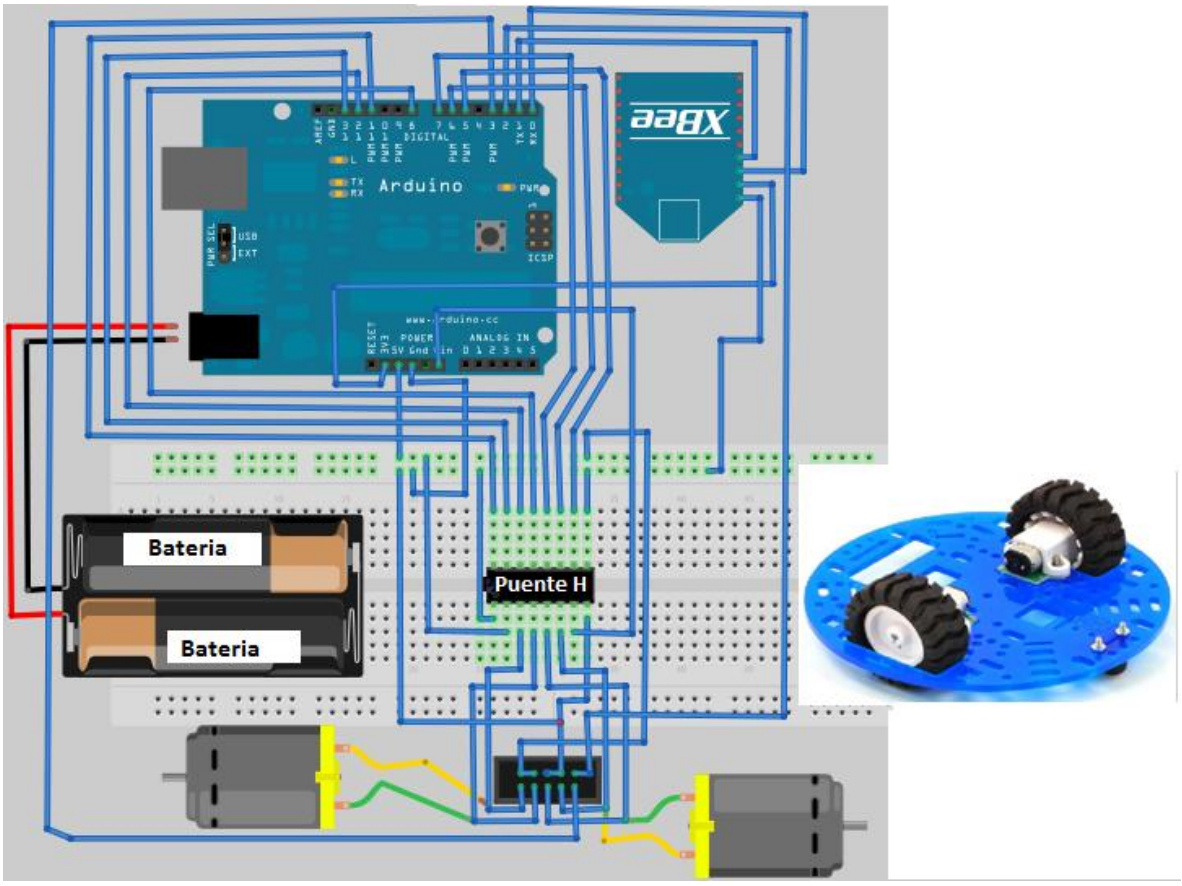


Fig. 3.3 Diagrama de conexionado y diseño del robot móvil líder

Para la construcción del robot móvil seguidor se emplearon los siguientes componentes: una placa Arduino, un puente h, un sensor infrarrojo, dos resistencias, conectores hembra y macho, cables UTP, dos motores, dos llantas, una bola castor, un chasis, un cable plug (para unir las baterías y alimentar a la tarjeta Arduino y por ende a los demás dispositivos y componentes electrónicos), dos baterías y también se diseñó una placa para montar los componentes electrónicos. La figura 3.4 nos muestra el diagrama de conexiones y el diseño del robot móvil seguidor.

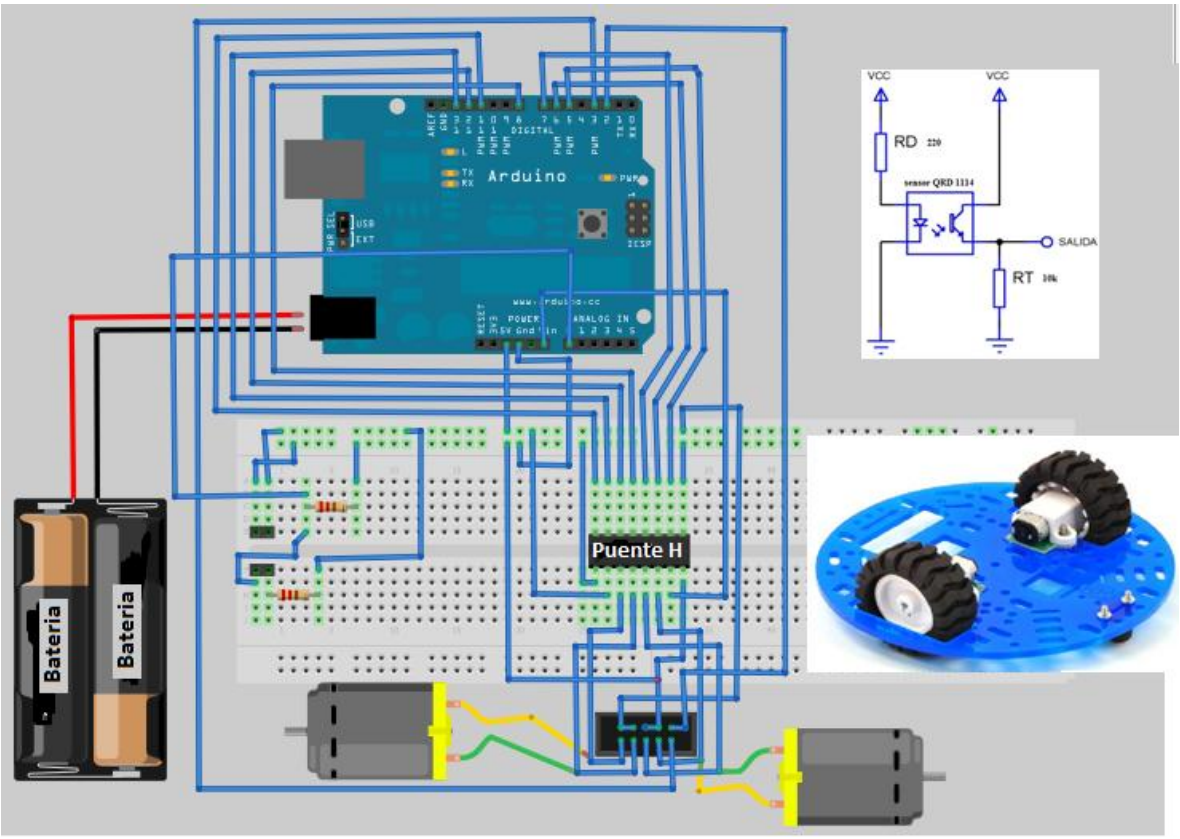


Fig. 3.4 Diagrama de conexionado y diseño del robot móvil seguidor

3.3 SIMULACIÓN EN MATLAB

En la figura 3.5 se presenta la programación de los algoritmos aplicados a los dos robots. Para llevar a cabo la simulación se hizo el uso de la herramienta Simulink y Simulink 3D Animation usando las librerías y bloques que estás contienen.

La estrategia de control que se propone se basa en: Un lazo basado en una ley de control basado en linealización por retroalimentación y su modelo del sistema, el cual incluye las distancias y orientación entre robots (bloque 3 de la figura 3.5) y el otro lazo esta asociado al modelo cinemático del vehículo (bloque 1 de la figura 3.5) que se encarga de generar los perfiles de velocidad para cada motor (a ser seguidos por el robot seguidor correspondientes al bloque 5) que corresponden a las trayectorias de posición deseadas.

En los siguientes párrafos veremos la programación que contiene cada bloque de la figura, los cuales cada línea se encuentra documentada para entender cada ecuación plasmada.

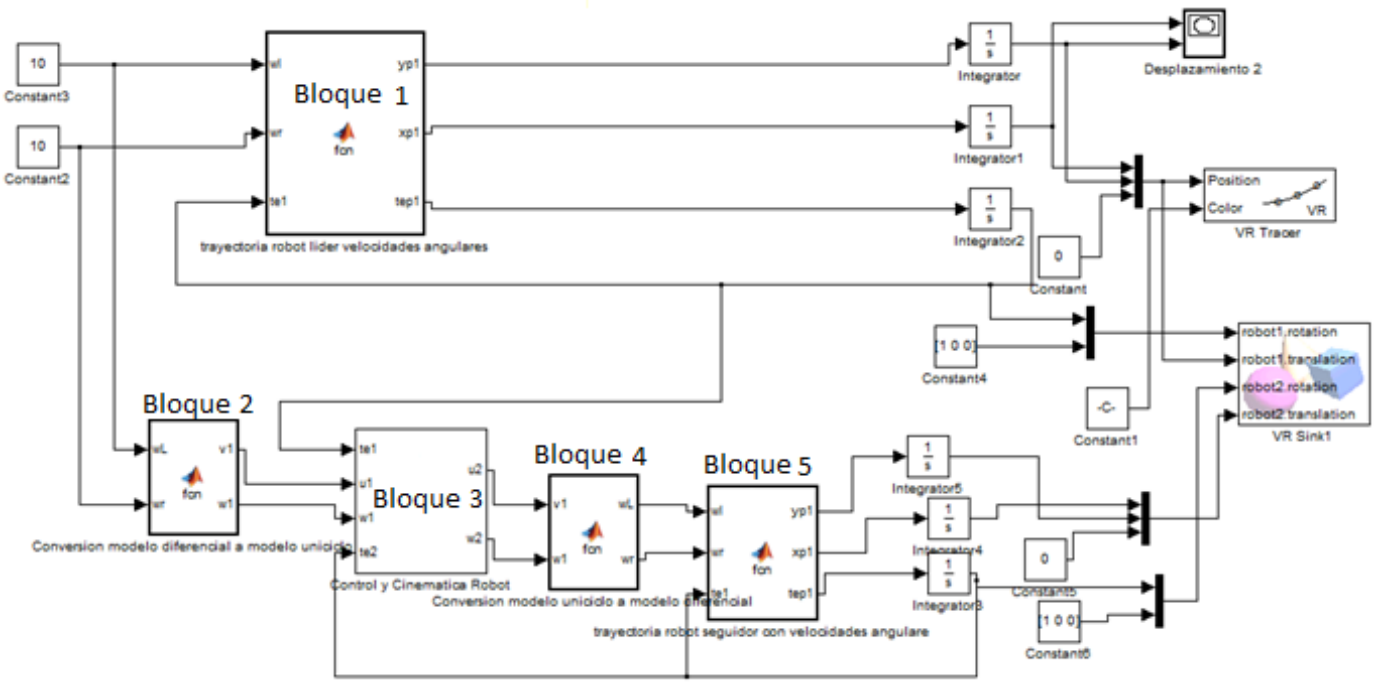


Fig. 3.5 Bloques de Programación en MATLAB

El bloque 1 tiene la programación del modelo cinemático que a su vez genera la trayectoria del robot líder en el que sus entradas son las velocidades angulares en cada rueda y sus salidas son la posición (x, y) y la orientación del robot con respecto al eje x. Las siguientes líneas son la programación de dicho bloque.

```
function [yp1, xp1, tep1] = fcn(wl,wr,te1)
%#codegen

%modelo cinematico del robot
r=.04; %radio de las llantas
l=.07; % distacia entre las llantas
xp1 = (r*(wl+wr)*cos(te1))/2;  % posicion en el eje x
yp1 = (r*(wl+wr)*sin(te1))/2;  % posicion en el eje y
tep1 = r*(wl-wr)/2*l;          % orientacion del eje longitudinal del
                               % del vehiculo con respecto al eje x
```

Si observamos la figura 3.5 el bloque 3 corresponde al control y cinemática del sistema, sus entradas en dicho bloque no son directamente velocidades angulares de las ruedas, por eso es necesario una transformación de velocidades angulares de las ruedas a velocidades lineales (la transformación corresponde del modelo diferencial al modelo uniciclo). Para ello es la función del bloque 2. La siguiente programación corresponde a dicho bloque.

```
function [v1,w1] = fcn(wL,wr)
%#codegen
r=.04; % radio de las ruedas
L=.07; % distancia entre las ruedas del robot

W=[wr;wL];   %matriz de velocidades angularesde las ruedas
T = [r/2 r/2;r/2*L -r/2*L]; % matriz de transformacion
U = T*W;    % multiplicacion de las dos matrices anteriores
v1 = U(1);  %elemento 1, velocidad lineal
w1 = U(2);  % elemento 2, velocidad angular
```

El bloque 3 tiene el nombre de control y cinemática del robot, este bloque es un subconjunto de dos bloques mas que se encuentran dentro de dicho bloque, donde se encuentra la ley de control basado en linealización por retroalimentación y su modelo del sistema [8]donde incluye las distancias y orientación entre robots. La figura 3.6 nos muestra los bloques que contiene dicho subconjunto.

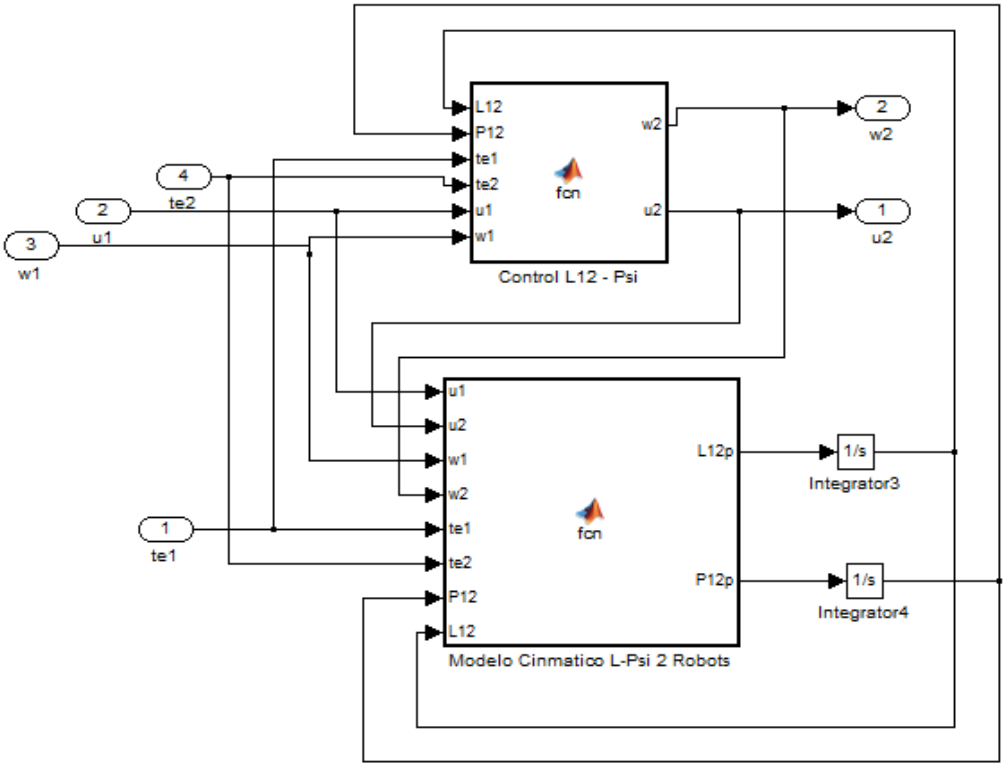


Fig. 3.6 Bloques del subconjunto Control y Cinemática del robot

El bloque llamado Control (de la figura 3.6), es la ley de control basado en linealización por retroalimentación, a continuación se presenta la programación de dicho bloque

```
function [w2,u2]= fcn(L12,P12,te1,te2,u1,w1)

a1  = 1; %% ganancias de control
a2  = 1; %% ganancias de control
d   = 0.06;  %% distancia del centro al castor
Ld12 = 2.4;   %% la distancia deseada entre los robots
Pd12 = 0*(2*pi/360);  %% angulo deseado entre el robot lider y el seguidor
%%%%%%%%%%%%%%%%%%%% Robot1 %%%%%%%%%%%%%%%%%%%%%%%
  %%ley de control basado en linealizacion por retroalimentacion
 Y1  = te1 + P12 -te2;   %%angulo del marco de referencia del sistema multiagente
 p12 = (a1*(Ld12 - L12) + u1*cos(P12))/cos(Y1);  %%variable usada en las ecuaciones de velocidad
 w2 = (cos(Y1)/d)*(a2*L12*(Pd12 - P12) - u1*sin(P12) + L12*w1 + p12*sin(Y1)); %%velocidad angular
 u2 = p12 - d*w2*tan(Y1); %% velocidad lineal
```

El segundo bloque de dicho subconjunto llamado Modelo Cinemático, se encuentra el modelo del sistema donde incluye las distancias y orientación entre robots. A continuación la programación de dicho bloque

```
function [L12p,P12p]= fcn(u1,u2,w1,w2,te1,te2,P12,L12)

%%modelo del sistema distancia y orientacion entre tobots
d   = 0.06;  %%distancia del castor hacia las llantas
Y1  = te1 + P12 -te2;  %% angulo del marco de referencia

L12p = u2*cos(Y1) - u1*cos(P12) + d*w2*sin(Y1); %distancia entre los robot
P12p = 1/L12*(u1*sin(P12) - u2*sin(Y1) + d*w2*cos(Y1) - L12*w1); %%distancia de orientacion
                                                                 %%entre  los robots

```

Observando nuevamente el bloque 3 de la figura 3.5 las salidas corresponden a velocidades lineales y es lógico ya que sus entradas fueron velocidades lineales. Y como el robot líder para su trayectoria esta basado en velocidades angulares de las ruedas del vehículo, debemos hacer de nuevo la trasformación inversa de lo que tiene programado el bloque 2. La siguiente programación corresponde al bloque 4.

```
function [wL,wr] = fcn(v1,w1)
%#codegen
r=.04; % radio de las ruedas
L=.07; % distancia entre las ruedas

U=[v1;w1]; %matriz de las velocidades angulares
T = [r/2 r/2;r/2*L -r/2*L]; % matriz de transformacion
W = (T^-1)*U; %multiplicacion de las matrices anteriores
wr = W(1); % elemento 1, velocidad angular de la rueda derecha
wL = W(2); % elemento 2, velocidad angular de la rueda izquierda
```

El bloque 5 corresponde a la trayectoria del robot seguidor en el que sus entradas son las velocidades angulares de la rueda (las mismas velocidades del robot líder) dando como salida la posición (x,y) y su orientación del robot con respecto al eje x. a continuación la programación de dicho bloque.

```
function [yp1, xp1, tep1] = fcn(wl,wr,te1)
%#codegen

%modelo cinematico del robot
r=.04; % radio de las ruedas del robot
l=.07; % distancia entre las ruedas del robot
xp1 = (r*(wl+wr)*cos(te1))/2; % posicion en el eje x
yp1 = (r*(wl+wr)*sin(te1))/2;  % posicion en el eje y
tep1 = r*(wl-wr)/2*l;   % orientacion del eje longitudinal del
                        % del vehiculo con respecto al eje x

```

Para obtener la simulación de los robot en forma 3D, fue necesario el diseño de las piezas del robot (llantas, motores, chasis, etc.) estas fueron diseñadas en el programa SolidWorks y dichas partes se ensamblaron en el mismo programa. En la figura 3.7 podemos observar las piezas diseñadas en SolidWorks y el ensamble de las mismas.

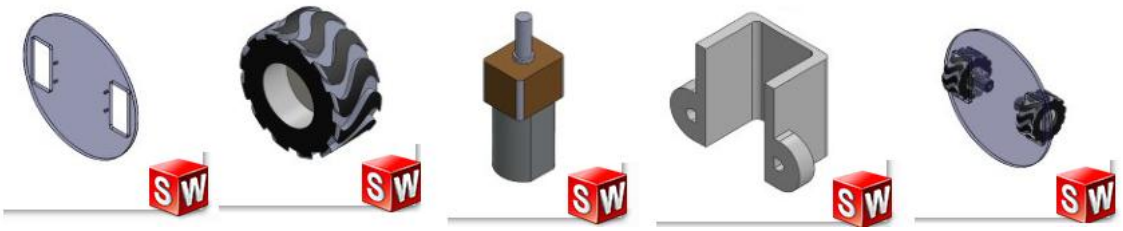


Fig. 3.7 Piezas de los robots diseñadas en SolidWorks y ensamble de las mismas

En la herramienta Simulink 3D Animation encontramos la librería VR Sink y VR Tracer. En la primera librería cargamos los robots ya ensamblados en SolidWorks y seleccionamos traslación y rotación que son las formas en las que se desplazaran los robots. Si observamos la figura 3.5 dicha librería tiene habilitadas cuatro entradas, dos para cada robot. Se usó la librería o herramienta Mux de Simulink el cual nos permite unir la posición de salida (x, y, z) del robot líder (en la posición z se puso una constante igual a 0 ya que en las salidas no tiene coordenada en z). Se usó un segundo componente Mux para unir la salida de orientación del robot con un bloque de números constantes (dicho bloque es para dar la orientación de inicio del robot). Los 2 datos de salida de la herramienta Mux lo conectamos en la figura de la librería VR Sink. El dato de posición del robot se conectó en traslación y el dato de orientación se conectó en rotación. El mismo procedimiento aplica para el robot seguidor.

La librería VR Tracer es para visualizar la posición del robot durante su trayectoria, esto permite trazar puntos en cada posición del recorrido del robot. En la entrada llamada Position de dicha librería se conecta las salidas (x, y) de los robots que son las misma que se conecto en la entrada de traslación de la librería VR Sink y en la entrada color se conecta una matriz que corresponde a un color especifico de los puntos que va trazando durante la trayectoria de los robots durante la simulación.

La Figura 3.8 nos muestra la simulación de la trayectoria de los robots empleando las librerías VR Sink y VR Tracer

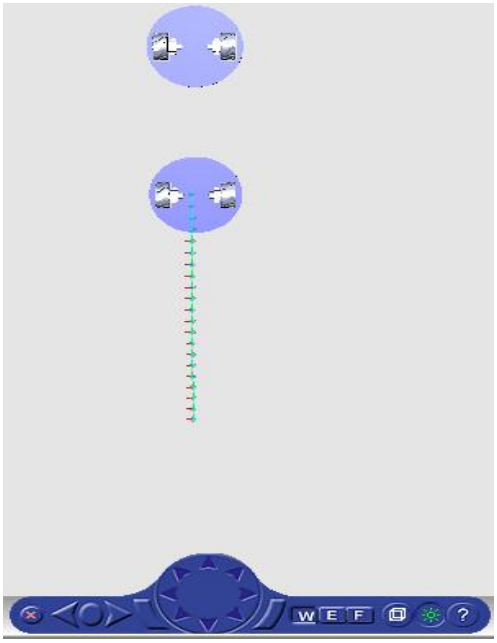


Fig. 3.8 Simulación en MATLAB de los dos Robot Móviles

3.4 PROGRAMACIÓN DE LOS ALGORITMOS EN LA PLATAFORMA ARDUINO.

Algoritmo del robot móvil seguidor.

Como se mencionó en el apartado de alcances y limitaciones que los algoritmos empleados en la simulación en MATLAB no seria la misma empleada para programar el Arduino.

Para el desarrollo del algoritmo en el robot seguidor se empleo un control proporcional de lazo cerrado, en el que nuestra variable de control retroalimentada es la señal analógica recibida por el sensor infrarrojo el cual nos proporciona datos en milivolts. El Arduino tiene una resolución ADC de 10 bits, el cual convierte el dato analógico de entrada del sensor infrarrojo de 0 a 1024.

Se puso una tarjeta negra detrás del robot líder, (dicho color es nuestro parámetro a muestrear) censamos el color a una distancia de 7 centimetros entre los dos robots y nos dio un voltaje de 40 milivolts, y al meterlo en el pin de entrada analógica A0 del Arduino este hizo la conversión dando como valor 10, es importante recalcar que el sensor infrarrojo es sensible a la luz del día y por ende los datos de lectura censados y la distancia entre los robots variara.

La teoría del control proporcional nos presenta la siguiente ecuación.

Ecuación: $y = Kc * e + y_0$

Donde:

y = salida del controlador

Kc = ganancia proporcional

e = error

y_0 = condiciones iniciales

Dicha ecuación fue empleada en la programación del algoritmo. Veamos un ejemplo, usando las variables empleadas en la programación Arduino

Dato: cuando el robot móvil líder se acerque al robot móvil seguidor, el sensor infrarrojo nos da un dato abajo de 40 milivolts y cuando se aleja del robot seguidor el dato censado es mayor de 40 milivolts. Entonces los 40 milivolts recibidos por el sensor infrarrojo es el parámetro que indica que el robot seguidor se encuentra a una distancia de 7 centímetros respecto al robot móvil líder; ya sea que estén parados o en movimiento.

Para determinar la ganancia del sistema (que en este caso es 2) se hicieron pruebas con ganancias igual a 1, 2, 3 y 4. Con la ganancia igual a 1, no había suficiente voltaje para que el robot móvil seguidor respondiera al movimiento del robot móvil líder. Cuando la ganancia se modificó a valores arriba de 2, el sistema se volvió muy sensible a la luz externa e inestable a la vez ya que el robot seguidor realizaba pequeños movimientos (hacia adelante y hacia atrás) cuando no debería ya que el robot móvil líder estaba estacionado. En cambio con la ganancia igual a 2, el sistema funciono de forma adecuada, ya que el robot móvil seguidor estuvo en sincronía con el robot móvil líder.

Supongamos que el valor censado por el sensor infrarrojo o valor de entrada es 8 (dato de la resolución de Arduino), eso significa que el robot líder está retrocediendo, ya que el dato censado esta debajo del valor de referencia. Veamos los cálculos

```
Refsc = 10;
k=2;
int sensorValueCen
ci=0;
Ec = Refsc – sensorValueCen;
wr = k * Ec + ci;
wl = k * Ec + ci;
```

El dato recibido por el sensor infrarrojo (a través de sensorValueCen) está debajo del valor de referencia, entonces al hacer los cálculos para determinar el error del sistema es: 10 – 8 (los datos son: el 10 es el valor de referencia menos dato de entrada: lectura del sensor infrarrojo) igual a 2. Para determinar la velocidad de las llantas aplicamos la fórmula: w= k*Ec + ci, donde k es la ganancia del sistema igual a 2, Ec= 2 (es el error calculado anteriormente) y ci es la condición inicial, para este caso es igual a 0; entonces w= 2*2+0= 4. La velocidad obtenida aplica a las dos llantas (derecha o izquierda) ya que es la misma fórmula.

Los resultados obtenidos caen en la condición wr > 0 y wl > 0 de la programación en Arduino y eso significa que las llantas del robot retroceden a una velocidad constante. Es bastante obvio que si el dato es mayor que 10 los resultados en las ecuaciones wr y wl serán negativas dando lugar a que las llantas avancen hacia adelante según el algoritmo programado en Arduino y si el valor censado es igual a 10 la salida pwm es cero para ambas velocidades aplicadas en la llantas del robot, eso significa que el robot líder esta parado totalmente. La velocidad pwm aplicada en las llantas dependerá del valor censado.

Las siguientes líneas corresponden al algoritmo programado en Arduino para el robot móvil seguidor.

```
const int AdelanteIzq = 13;                 // salida digital hacia el puente h
const int AdelanteDer = 7;                  // salida digital hacia el puente h
const int AtrasIzq = 12;                    // salida digital hacia el puente h
const int AtrasDer = 6;                     // salida digital hacia el puente h
                                            // varables globales
float a;
float b;
const int analogInPinI = A0;                // entrada de  la señal analógica del sensor infrarrojo
int sensorValueCen = 0;                     // inicializar la entrada

void setup()
{
                                            // pines hacia el puente h
    pinMode(AdelanteIzq, OUTPUT);           // motor izq ain2
    pinMode(AtrasIzq, OUTPUT);              // motor izq ain1
    pinMode(11, OUTPUT);                    // motor  pwma
    pinMode(8, OUTPUT);                     // motor  stby
    pinMode(AdelanteDer, OUTPUT);           // motor der bin2
    pinMode(AtrasDer, OUTPUT);              // motor der bin1
    pinMode(5, OUTPUT);                     // motor  pwmb
  Serial.begin(9600);                       // habilitar la comunicación serial
```

```
}

void loop()
{
  sensorValueCen = analogRead(analogInPin1);          // lectura del sensor  infrarrojo

  int Refsc = 10;                                     // valor de referencia de la entrada analógica del sensor para
                                                      // Mantenerlo a una distancia de 7 cm
  int ci = 0;
  float Ec = Refsc - sensorValueCen;                  // error que se retroalimenta del control proporcional
  int k = 2;                                          // ganancia del control proporcional
  float  wr = k * Ec + ci;                            // velocidad en la llanta derecha
  float  wl = k * Ec + ci;                            // velocidad en la llanta izquierda
  a = map(abs(wr),0,150,0,255);                       // velocidad de la llanta derecha, y datos de entrada del sensor
                                                      // reducido de 0 a 150, salida pwm de 0 a 255
  b = map(abs(wl),0,150,0,255);                       // velocidad de la llanta izquierda y datos de entrada del sensor
                                                      // reducido de 0 a 150 y la salida pwm de 0 a 255

  analogWrite(11,a);                                  // salida pwm dependiendo del valor wr
  analogWrite(5,b);                                   // salida pwm dependiendo del valor wl
  Serial.print("Pwm0: ");                             // imprimir el valor de pwm en wr
  Serial.print(a);
   Serial.print(" Pwml: ");                           // imprimir el valor de pwm en wl
  Serial.print(b);
  Serial.println(" Sen_cent: ");                      // imprimir el valor del sensor infrarrojo

  Serial.print(sensorValueCen);
  delay(100);

if (wr < 0){                                          // condiciones de adelante o atrás del robot

    digitalWrite(AdelanteDer,HIGH);
    digitalWrite(AtrasDer,LOW);
    digitalWrite(8,HIGH);
}
 else{
    digitalWrite(AdelanteDer,LOW);
    digitalWrite(AtrasDer,HIGH);
    digitalWrite(8,HIGH);
 }
 if (wl < 0){
    digitalWrite(AdelanteIzq,HIGH);
    digitalWrite(AtrasIzq,LOW);
    digitalWrite(8,HIGH);
 }else {
```

```
    digitalWrite(AdelanteIzq,LOW);
    digitalWrite(AtrasIzq,HIGH);
    digitalWrite(8,HIGH);

 }
}
```

Algoritmo del robot móvil líder.

Para la programación del algoritmo en Arduino para el robot líder, se empleo el control de lazo abierto, que simplemente se manda la señal pwm al puente H para que mueva los motores a la velocidad deseada. Para manipular dicho robot se empleó un celular touch. En la figura 3.9 vemos el diagrama a bloques de la comunicación del celular con la computadora para la interacción con el robot líder.

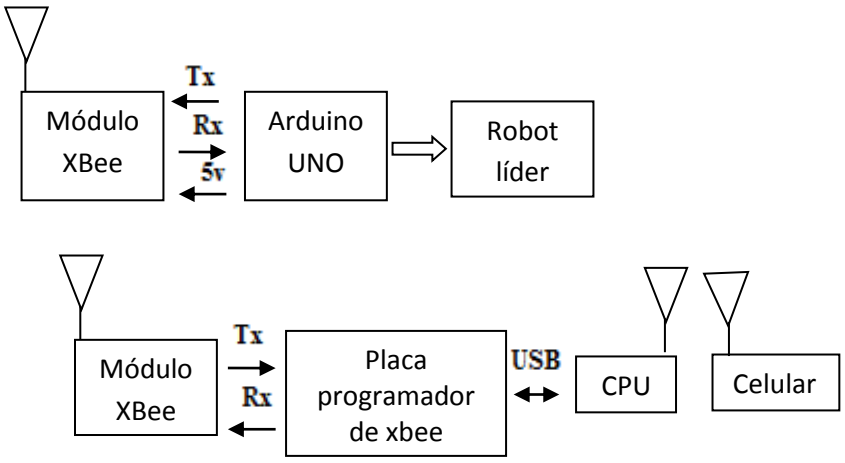


Fig. 3.9 Diagrama a bloques de la comunicación inalámbrica del robot líder

La programación del algoritmo se hizo en Processing, una herramienta que tiene un sinfín de librerías que hay que descargar para realizar la comunicación con el celular. Para manipular el robot por medio del celular es fundamental instalar en la computadora y en el celular el programa TouchOsc, el cual nos permite la interacción entre ambos dispositivos y también con el robot y para ello es importante una red de internet inalámbrica.

En la programación en Arduino Processing se configuro la comunicación de los módulos de XBee, se programó la imagen gráfica para visualizar la interacción de la computadora y el celular cuando este envía datos al robot, en el que se visualiza el voltaje enviado al robot.

La comunicación inalámbrica del robot móvil líder se logra con un módulo XBee .El sistema se utiliza para comunicarse con el Arduino como si fuera un puerto serial. A través del puerto se controla el robot por medio de la transferencia de caracteres. En comparación con otros sistemas de radiofrecuencia, el XBee permite la transmisión confiable debido a su manera de encriptación y codificación.

Para lograr que el robot pueda avanzar de manera lineal es necesario ajustar la velocidad de las llantas del robot. Por ejemplo, para que el robot pueda avanzar en línea recta es necesario que ambas ruedas giren a la misma velocidad. Para esto se utilizó un control de lazo abierto que controla la velocidad de cada llanta en forma independiente. Para poder ajustar las variables de control se utilizaron dos salidas de PWM (*Pulse Width Modulator*) del sistema Arduino.

La figura 3.10 nos muestra el control del robot por medio del celular. En la figura se observan 4 barras pero solo se habilitaron las barras 2 y 3 para controlar el robot. A la mitad de las barras es velocidad 0 o pwm igual a cero, al deslizar las barras hacia arriba el pwm desciende gradualmente hasta -125 y al deslizar las barras hacia abajo haciende el pwm hasta 125, eso permite controlar el robot hacia la izquierda o derecha alternando el deslizamiento de las barras o si se quiere una velocidad constante hay que deslizar las barras al mismo tiempo en la misma dirección. Se puso el rango de 125 a -125 para tener un buen manejo del robot y no haya problemas.

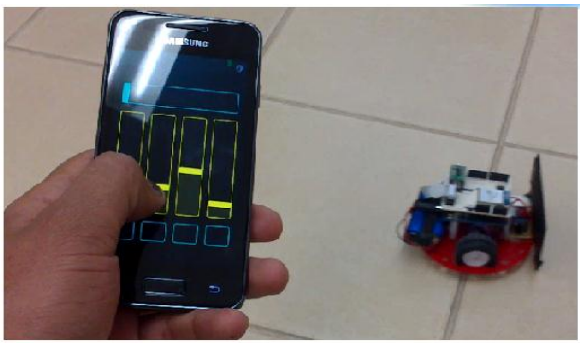


Fig. 3.10 Control del robot por medio de un celular

Los datos enviados por medio del celular se puede visualizar en la figura 3.11, que es la gráfica programada en Processing, eso nos ayuda a verificar que el XBee que está en el robot móvil líder si esta recibiendo la información y también para ver el rango de voltaje que se esta aplicando al robot.

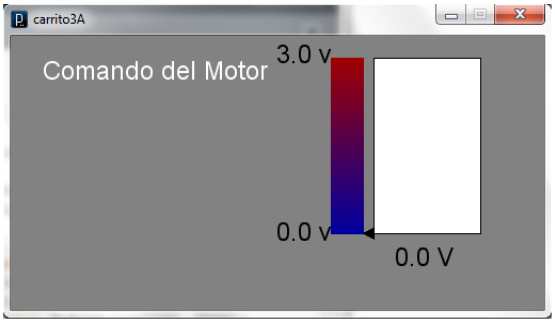


Fig. 3.11 Monitoreo de los datos al enviar información al robot

A continuación presentamos la programación en Arduino processing del control del robot móvil líder. Esta documentada es decir tiene comentarios para entender mejor la programación.

```
import processing.serial.*;
import cc.arduino.*;
Arduino arduino;
import oscP5.*;
import netP5.*;
OscP5 oscP5;

                                        // Variables de Celular //
float v_fader2 = 0.0f;
float v_fader4 = 0.0f;


PFont font;
int pin = -1;                           // la corriente de la luz encendido 'digital' LED //
int A=150;
int b=0;
int p=0;

                                        // Variables de Botones //
int ofsetx =50;
int cordx = 40;
int cordy = 50;
int ancho = 40;
int alto  = 40;
boolean pulsado1 = false;
boolean pulsado2 = false;
boolean pulsado3 = false;
float scale;
int a= 0;
```

```
                                                                    // Variables Velocidad //
int ofsetI;
float velI=0.0001;
float velD=0.0001;
float v;
int yDist;
PFont font12;
PFont font24;
float[] tempHistory = new float[1000];

void setup() {

   size( 500, 250);
   scale = width/40;
                                                                    // inicializar celular //
    frameRate(25);
 /* start oscP5, listening for incoming messages at port 8000 */
 oscP5 = new OscP5(this,8000);

      font = createFont( "Georgia ", 24 );
                                                                   // Inicializar Arduino //
 println(Arduino.list());
 arduino = new Arduino(this, Arduino.list()[0], 57600);
  for (int i = 0; i <= 13; i++)
  arduino.pinMode(i, Arduino.OUTPUT);
                                                                    // inicializar idex //
for(int index = 0; index<100; index++)
 tempHistory[index] = 0;
}

                                                                    // asignar un valor al evento //
void oscEvent(OscMessage theOscMessage) {

   String addr = theOscMessage.addrPattern();
   float  val  = theOscMessage.get(0).floatValue();

   if(addr.equals("/1/fader2"))       { v_fader2 = val; }
   else if(addr.equals("/1/fader4"))  { v_fader4 = val; }
}

                                                                     // Dibujar los Botones //
void draw() {

int stby =0;
 arduino.digitalWrite(8,Arduino.HIGH);
                                                                   // Los botones de activación adelante-reversa-freno //
```

```
  background(130);
  int[] digits = new int[13];
  stroke( 132);
  fill(255);
  text( "Comando del Motor", 30, 40 );

                                                    // conversión de barras a velocidades //
  velI =(int) map(v_fader2,0,1,-125,125);           // velocidad izquierda primera barra hacia delante //
  velD =(int) map(v_fader4,0,1,-125,125);           // velocidad derecha  primera barra hacia delante//

  a= (int) map(v_fader2,0,1,0,160);                 // para la grafica
  int pwmI=abs(int(velI));
  int pwmD=abs(int(velD));
  //int pwmB=pwmA;
  arduino.analogWrite(11,pwmI);
  arduino.analogWrite(5,pwmD);

  print(velI);
  print(' ');
  println(velD);

                                                     // barras cel //
  if(velD > 0){
  arduino.digitalWrite( 12, Arduino.LOW );
  arduino.digitalWrite( 13, Arduino.HIGH );
  }
  if(velD < 0){
  arduino.digitalWrite( 12, Arduino.HIGH );
  arduino.digitalWrite( 13, Arduino.LOW );
  }

  if(velI > 0){
  arduino.analogWrite( 6, 0);
  arduino.digitalWrite( 7, Arduino.HIGH );
  }
  if(velI < 0){
  arduino.analogWrite( 6, 255 );
  arduino.digitalWrite( 7, Arduino.LOW );
  }

                                                    // Graficar y variar velocidad //
stroke (0);

                                                    // rectángulo de colores//
```

```
for (int colorIndex = 0; colorIndex <= 160; colorIndex++)
{
stroke(160 - colorIndex, 0, colorIndex);
line(300, colorIndex + 20, 330, colorIndex + 20);
}

                                                        // dibujar grafica
stroke(0);
fill(255,255,255);
rect(340,20,100,160);
for (int index = 0; index<100; index++)
{
if(index == 99)
tempHistory[index] = a;
else
tempHistory[index] = tempHistory[index + 1];
point(340 + index, 180 - tempHistory[index]);
}

                                                        // escribir los valores de referencia
fill(0,0,0);
textFont(font);
textAlign(RIGHT);
text("3.0 v", 300, 25);
text("0.0 v", 300, 187);

                                                        // Dibujar los puntos del rectangulo
yDist =160-a;///int(160 - (160 * (a * 0.01)));
stroke(0);
triangle(330, yDist + 20, 340, yDist + 15, 340, yDist + 25);

                                                        // Escribe el voltaje
fill(0,0,0);
textFont(font);
textAlign(LEFT);
v= (float )map(a,0,160,0,3);

text(str((v)) + " V", 360, 210);

}
```

3.5 RESULTADOS

Usar la plataforma Arduino en el proyecto fue fundamental para obtener resultados satisfactorios ya que es una herramienta muy didáctica, amigable para cualquier proyecto, ya que su estructura permite la conexión de sensores, tanto analógicos como digitales, conexión del XBee, del puente H etc. Y también es autosuficiente ya que cuenta con salidas para alimentar tanto al sensor, al XBee y al puente H. la plataforma Arduino es accesible para poner sobre ella las placas que uno diseñe o placas de fabrica que se usan para montar los sensores, puente H y demás componente electrónico para realizar un proyecto. El la figura 3.12 vemos la plataforma Arduino allí podemos observar sus pines hembras para entradas analógicas, salidas digitales, alimentación (fuente de voltaje) y sus puertos de comunicación serial, esto permite que dicha placa sea accesible para montar sobre ella dispositivos para cualquier tipo de proyecto eso permite ahorrar espacio en la realización de futuros proyectos, tiempo e inclusive dinero.

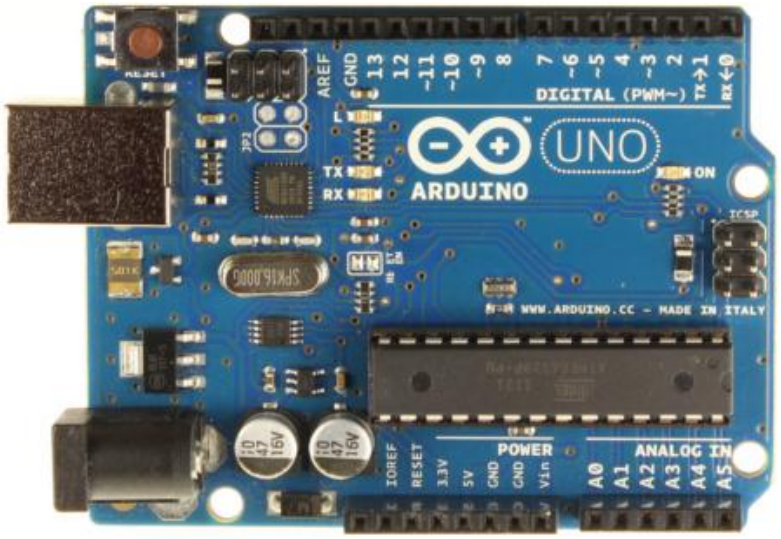


Fig. 3.12 Estructura del Arduino UNO

Como mencionamos en el párrafo anterior la estructura del Arduino es amigable para colocar placas sobre ella. La figura 3.13 corresponde al robot móvil seguidor se observa que la placa Arduino tiene montado una placa, este diseño permite el uso de cada uno de los componentes empleados para hacer funcionar el robot y poder realizar el proyecto.

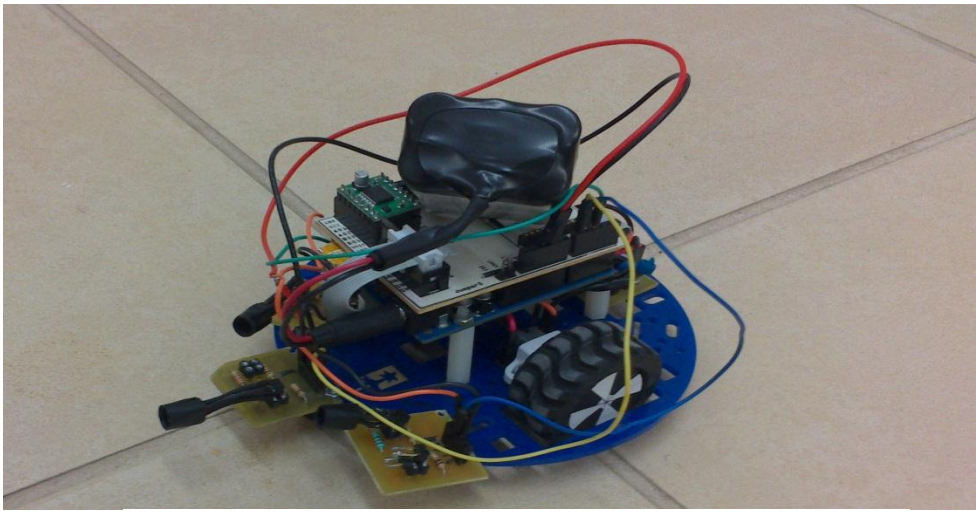


Fig. 3.13 Robot móvil seguidor

La figura 3.14 nos muestra el robot móvil líder, a diferencia del robot móvil seguidor este tiene montado la tarjeta XBee para la comunicación inalámbrica con el celular, también se observa la tarjeta negra, el cual es el dato o parámetro que el robot móvil seguidor toma para seguir al robot móvil líder.

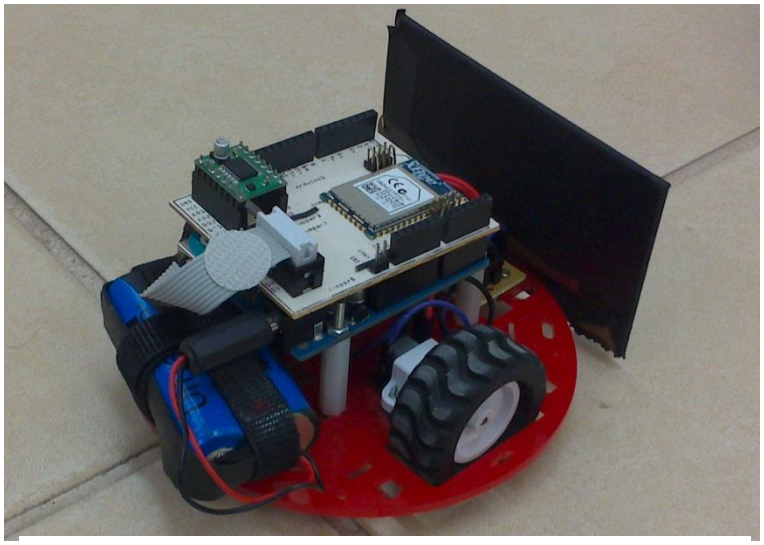


Fig. 3.14 Robot móvil líder

La manipulación del robot líder con el celular se realizo con éxito, también hubo buen resultado cuando el robot seguidor seguía al robot líder de esa manera se realizo la cooperación entre los dos robots. El seguimiento del robot seguidor respondía mejor cuando el robot líder retrocedía en línea recta ya que la concentración de luz del ambiente disminuía cuando la tarjeta de color negro se acercaba hacia el sensor infrarrojo, en cambio cuando el robot líder se desplazaba hacia adelante la tarjeta se alejaba del sensor infrarrojo abriendo paso a mayor reflexión de luz del ambiente y por ello en ocasiones el sensor perdía su referencia provocando que el seguimiento no se diera con éxito. En la figura 3.15 se observa la manipulación del robot líder por medio del celular y la posición de cada uno de los robots para que el robot seguidor sigua al robot líder permitiendo de esta forma la cooperación entre los dos robots móviles.

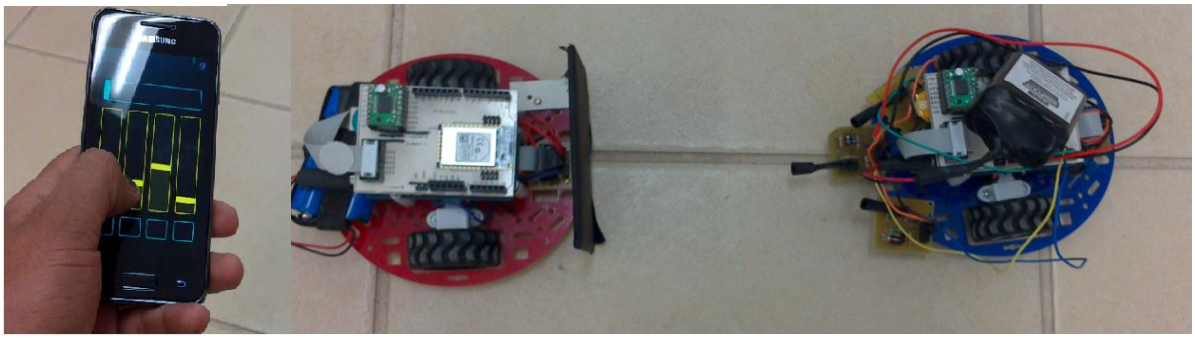


Fig. 3.15 cooperación de dos robots móviles tipo diferencial

La simulación en MATLAB se observa en la figura 3.16. , donde se le aplico una velocidad angular de 10 a cada una de las ruedas o puede ser otro valor de velocidad y su posición en $\pi/2$ también puede ser otro valor de orientación, allí podemos observar el desplazamiento de ambos robots manteniendo siempre una distancia durante su recorrido.

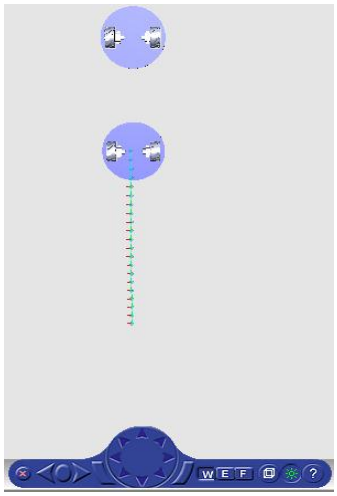


Fig. 3.16 Cooperación de robots móviles simulado en MATLAB

CONCLUSIÓN

En este proyecto se llevo a cabo la cooperación entre dos robots móviles de manera real utilizando un algoritmo para cada robot y también se presento la simulación en MATLAB empleando dos algoritmos diferentes a lo empleado en el sistema real, las tareas de simulación y la aplicación real de la cooperación entre los dos robots se realizaron satisfactoriamente.

El área de la robótica es muy amplia que no bastaría una vida para conocer a fondo cada concepto que lleva a un tema en particular. El presente trabajo me abrió un panorama más de lo que es la ciencia de la robótica, el aprender los conceptos mecánicos, ley de control, cinemática etc., es un conocimiento mas que tengo el privilegio de hacerlo parte del aprendizaje diario de la vida, es claro que dicho conocimiento esta plasmado en simulación en MATLAB, pero es el principio para llevarlo a la vida real, eso requiere de fuerza de voluntad para explorar, investigar, analizar y estudiar para lograr llevarlo a la practica y darle aplicaciones que ayuden a mejorar la vida humana o ayuda del planeta.

Durante la carrera te enseñan lo teórico de lo que se aplica en el ámbito laboral, en ciertas ocasiones te quedas con la incógnita de como hacerlo físicamente. Por medio de este proyecto he despejado dudas de lo teórico ya que aprendí a aplicar el control clásico, programando los algoritmos usados durante el proyecto en la plataforma Arduino, capaz de realizar cooperación entre dos robots, en el que aplique los principios de electrónica, control clásico y programación.

Durante el desarrollo del proyecto te das cuenta que nada es ideal, por mucho que se esfuerce la tecnología en crear cosas perfectas, es allí donde hay que investigar los factores que ocasionan los problemas y realizar ajustes para idealizar el sistema.

Poder aplicar conocimientos adquiridos es fenomenal, pero el adquirir más conocimiento durante la práctica es a un mejor. La realización del proyecto me ayudo a visualizar el mundo real y a comprender que necesito conocer e investigar las cosas que aun no se encuentran en mi lista de conocimientos.

BIBLIOGRAFÍA.

[1] http://www.urbe.edu/publicaciones/telematica/indice/html-vol7-2/articulo8.html

[2] Pérez Arreguín Jorge Israel, Tovar Arriaga Saúl, Ubaldo Giovanni Villaseñor Carrillo. Universidad Autónoma de Querétaro, Facultad de Informática.
Campus Juriquilla, Avenida de las Ciencias s/n, Juriquilla Querétaro Qro. CP. 76230
israel_1985@live.com.mx

[3] R. Silva Ortigoza1, M. A. Molina Vilchis1, V. M. Hernández Guzmán2.
1 CIDETEC-IPN. Departamento de Postgrado. Área de Mecatrónica. Unidad Profesional Adolfo López Mateos. C.P. 07700, México, D.F., MÉXICO. e-mail: rsilvao@ipn.mx, mamolinav@ipn.mx, mmarciano@ipn.mx, aportilla@ipn.mx
2 Universidad Autónoma de Querétaro, Facultad de Ingeniería. Ap. Postal 3-24. C.P. 76150, Querétaro, Qro., MÉXICO. e-mail: vmhg@uaq.mx

[4] Universidad de Murcia España. Depto. Ingeniería de Información y las Comunicaciones. Enero de 2001. Humberto Martínez Barberá.

[5] Cristian G. Pérez T.1, Eduardo S. Espinoza Q.1, Abel García B.1, Jesús M. Muñoz P.2
1 LIREA, Departamento de Mecatrónica, Universidad Politécnica de Pachuca, Zempoala, Hidalgo, México. Email: [cristian, steed, abel]@upp.edu.mx
2 Ingeniería de Electrónica y Telecomunicaciones, Universidad Politécnica de Puebla, San Mateo Coanala, Puebla, México.

[6] CARRASCO, R. y Cipriano, A. (2004) Fuzzy Logic Based Nonlinear Kalman Filter Applied to Mobile Robots Modelling.

[7] https://maps.google.com.mx/

[8] UNIVERSIDAD DE CHILE FACULTAD DE CS. FS. Y MATEMÁTICAS DEPARTAMENTO DE INGENIERÍA ELÉCTRICA. “ROBOTS MOVILES”

[9] Instituto Politécnico Nacional. Centro de investigación en computación laboratorio de reconocimiento de patrones “control visual de un robot móvil Khepera II”
http://wdvillegas.wordpress.com/2012/03/04/actividad-9-propiedades-de-los-sistemas/

[10] Eduardo Aranda Bricaire, Tomas Salgado Jiménez y Martin Velasco Villa. Centro de Investigación y Estudios Avanzados-IPN. Departamento de Ingeniería Eléctrica, Sección Mecatrónica. “Control no Lineal Discontinuo de un Robot Móvil”. Av. IPN 2508, Sn. Pedro Zacatenco, México D.F.

[11] http://dfists.ua.es/~jpomares/arduino/page_13.htm
http://alexmartins.net/blog/category/eletronica/arduino/
http://ro-botica.com/Arduino.asp
http://arduino.cc/es/Main/ArduinoBoardDuemilanove

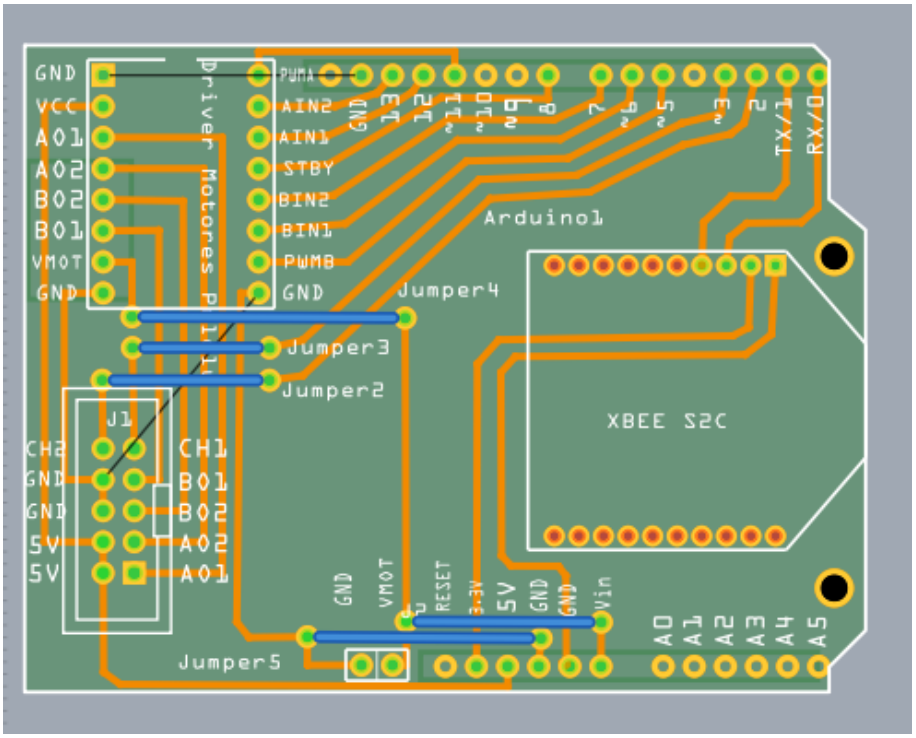
[12] http://es.wikipedia.org/wiki/Sensor_infrarrojo

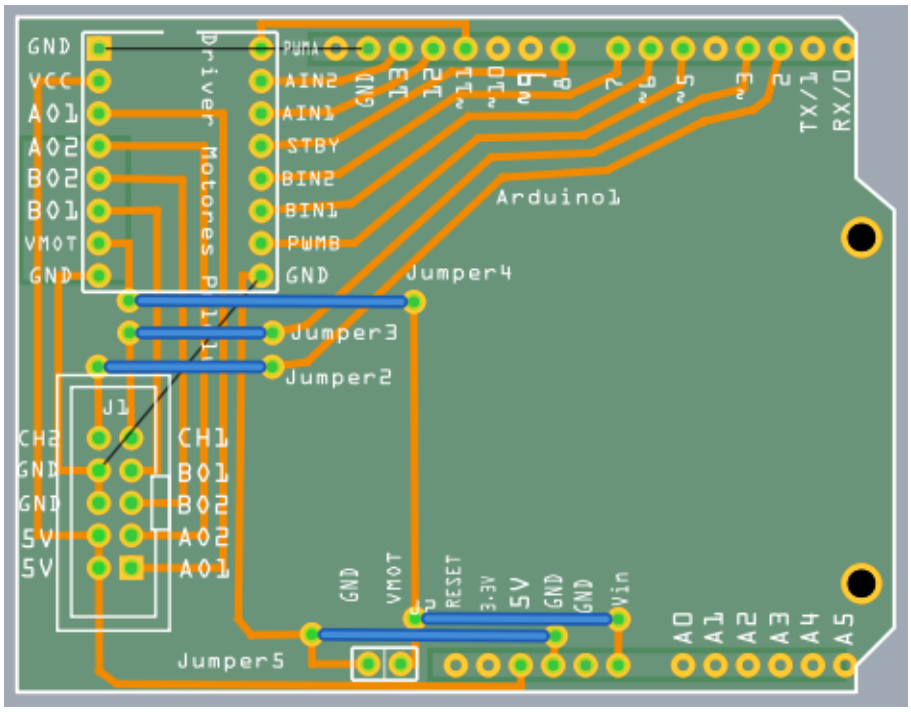
[13] Jaydey P. Desai, Jim Ostrowski, Vijay Kumar. General Robotics and Active Sensory Perception (GRASP) Laboratory, University of Pennsylvania. “Controlling formations of multiple mobile robots”

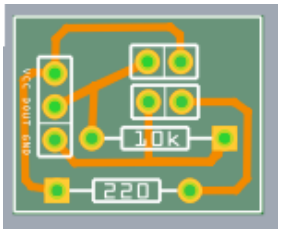
ANEXO A.- Diseño de las Tarjetas del Circuito impreso

Pistas usadas en el proyecto.

Pista para el robot líder, robot seguidor y para el sensor QRD 1114





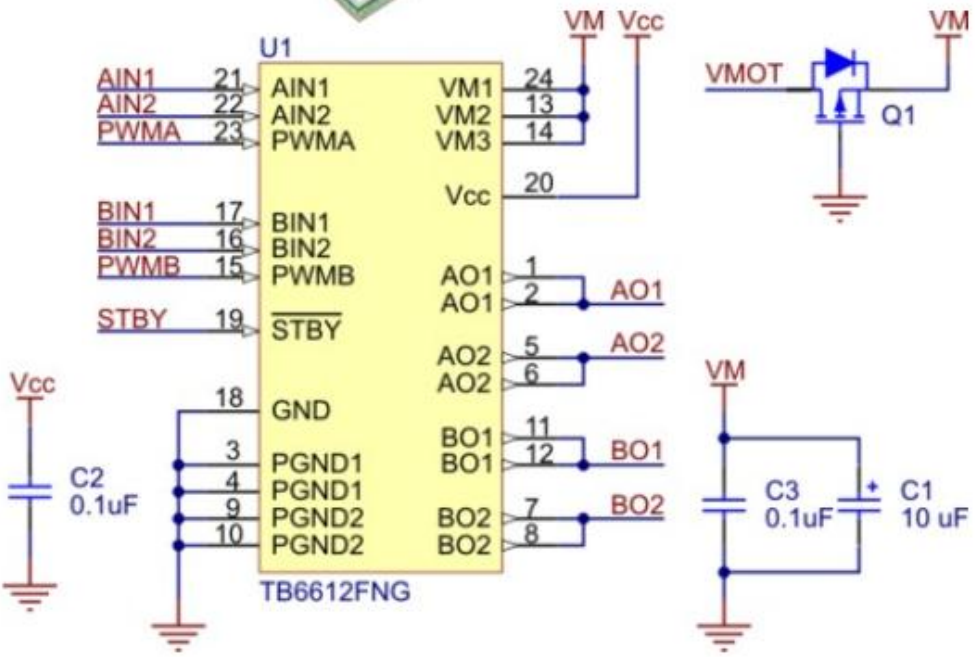


ANEXO B.- DATASHEET

DATASHEET DEL PUENTE H POLOLU

TB6612FNG Dual Motor Driver Carrier

Schematic Diagram

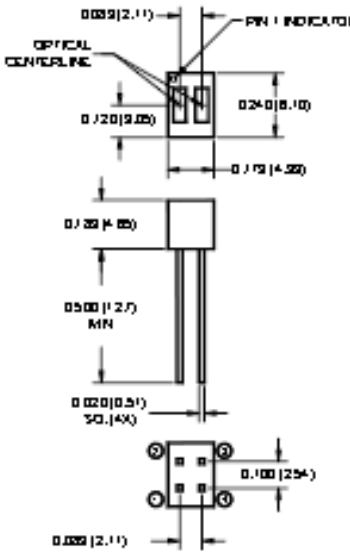


DATASHEET SENSOR INFRARROJO

FAIRCHILD
SEMICONDUCTOR

QRD1113/1114
REFLECTIVE OBJECT SENSOR

PACKAGE DIMENSIONS



PIN 1 COLLECTOR PIN 3 ANODE
PIN 2 EMITTER PIN 4 CATHODE

NOTES:

1. Dimensions for all drawings are in inches (millimeters).
2. Tolerance of ± .010 (.25) on all non-nominal dimensions unless otherwise specified.
3. Pins 2 and 4 typically .050" shorter than pins 1 and 3.
4. Dimensions controlled at housing surface.

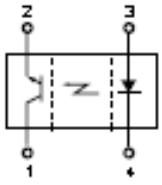
FEATURES

- Phototransistor Output
- No contact surface sensing
- Unfocused for sensing diffused surfaces
- Compact Package
- Daylight filter on sensor

NOTES (Applies to Max. Ratings and Characteristics Tables.)

1. Derate power dissipation linearly 1.33 mW/°C above 25°C.
2. RMA flux is recommended.
3. Methanol or isopropyl alcohols are recommended as cleaning agents.
4. Soldering iron tip 1/16" (1.6mm) from housing.
5. As long as leads are not under any spring tension.
6. D is the distance from the sensor face to the reflective surface.
7. Cross talk (I_{CX}) is the collector current measured with the indicator current on the input diode and with no reflective surface.
8. Measured using an Eastman Kodak neutral white test card with 90% diffused reflecting as a reflective surface.

SCHEMATIC



ABSOLUTE MAXIMUM RATINGS (T_A = 25°C unless otherwise specified)

Parameter	Symbol	Rating	Units
Operating Temperature	T_{OPR}	-40 to +85	°C
Storage Temperature	T_{STG}	-40 to +85	°C
Lead Temperature (Solder Iron)[2,3]	T_{SOL-I}	240 for 5 sec	°C
Lead Temperature (Solder Flow)[2,3]	T_{SOL-F}	260 for 10 sec	°C
EMITTER Continuous Forward Current	I_F	50	mA
Reverse Voltage	V_R	5	V
Power Dissipation[4]	P_D	100	mW
SENSOR Collector-Emitter Voltage	V_{CEO}	30	V
Emitter-Collector Voltage	V_{ECO}		V
Power Dissipation[4]	P_D	100	mW

1 of 4 100030A

FAIRCHILD
SEMICONDUCTOR

QRD1113/1114
REFLECTIVE OBJECT SENSOR

ELECTRICAL / OPTICAL CHARACTERISTICS (T_A = 25°C)

PARAMETER	TEST CONDITIONS	SYMBOL	MIN	TYP	MAX	UNITS
EMITTER Forward Voltage	I_F = 20 mA	V_F	—	—	1.7	V
Reverse Current	V_R = 5 V	I_R	—	—	100	µA
Peak Emission Wavelength	I_F = 20 mA	λ_{PE}	—	940	—	nm
SENSOR Collector-Emitter Breakdown	I_C = 1 mA	BV_{CEO}	30	—	—	V
Emitter-Collector Breakdown	I_E = 0.1 mA	BV_{ECO}	5	—	—	V
Dark Current	V_{CE} = 10 V, I_F = 0 mA	I_D	—	—	100	nA
COUPLED QRD1113 Collector Current	I_F = 20 mA, V_{CE} = 5 V D = .050" (4)	$I_{C(ON)}$	0.300	—	—	mA
QRD1114 Collector Current	I_F = 20 mA, V_{CE} = 5 V D = .050" (4)	$I_{C(ON)}$	1	—	—	mA
Collector Emitter Saturation Voltage	IF = 40 mA, Ic = 100 µA D = .050" (4)	$V_{CE(SAT)}$	—	—	0.4	V
Cross Talk	I_F = 20 mA, V_{CE} = 5 V, Bs = 0 [7]	I_{CX}	—	200	10	µA
Rise Time	V_{CE} = 5 V, RL = 100 Ω	t_r	—	10	—	µs
Fall Time	$I_{C(ON)}$ = 5 mA	t_f	—	50	—	µs

FAIRCHILD
SEMICONDUCTOR

QRD1113/1114
REFLECTIVE OBJECT SENSOR

TYPICAL PERFORMANCE CURVES

Fig. 1 Forward Voltage vs. Forward Current

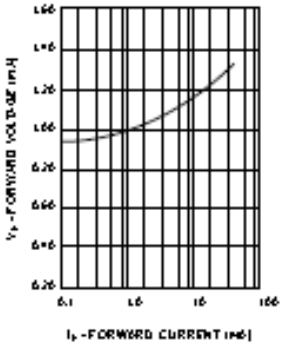


Fig. 2 Normalized Collector Current vs. Forward Current

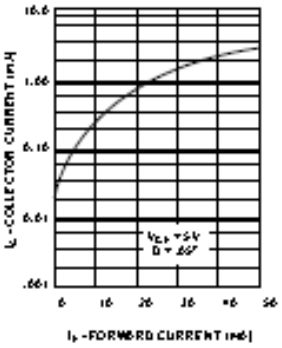


Fig. 3 Normalized Collector Current vs. Temperature

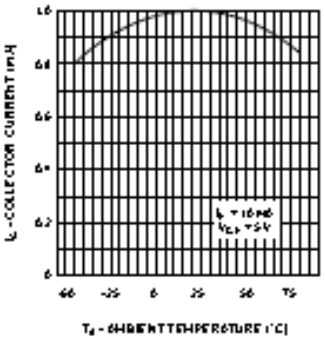


Fig. 4 Normalized Collector Dark Current vs. Temperature

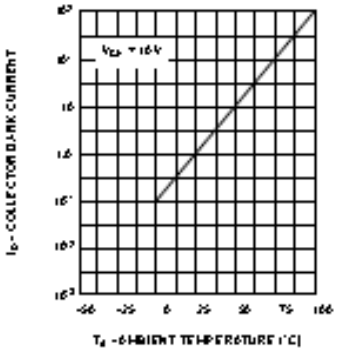
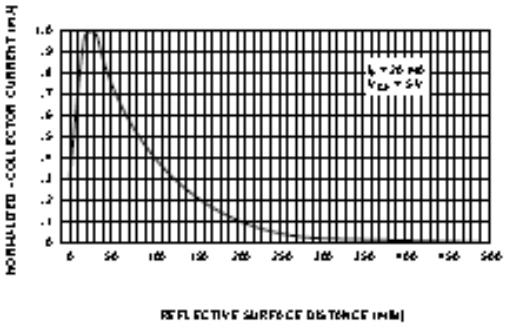


Fig. 6 Normalized Collector Current vs. Distance



DATASHEET MODULO XBEE

XBee® and XBee-PRO® ZB

ZigBee® Embedded SMT RF Modules for OEMs

Embedded SMT RF modules provide low-cost, low-power wireless connectivity and feature Surface Mount Technology and a Serial Peripheral Interface.

Overview

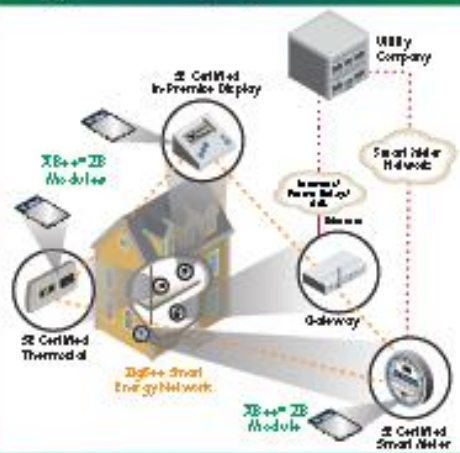
XBee and XBee-PRO ZB embedded SMT RF modules provide cost-effective wireless connectivity to electronic devices. They are interoperable with other ZigBee PRO feature set devices, including devices from other vendors*. They share a common hardware footprint and are available in a variety of protocols and frequencies.

Utilizing Surface Mount Technology (SMT), XBee and XBee-PRO ZB modules are ideal for applications in the energy and controls markets where manufacturing efficiencies are critical. The Serial Peripheral Interface (SPI) provides a high-speed interface and optimizes integration with embedded microcontrollers, lowering development costs and reducing time to market.

Products in the XBee family require no configuration or additional development. Programmable versions of the XBee-PRO ZB module make customizing applications easy. Programming directly on the module eliminates the need for a separate processor. Because the wireless software is isolated, applications can be developed with no risk to RF performance or security.

*Interoperability requires the ZigBee Feature Set or ZigBee PRO Feature Set to be deployed on all devices. Contact Digi Support for details.

Application Highlight



Features/Benefits

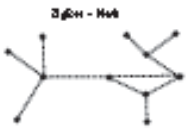
- Interoperable with other ZigBee compliant devices*
- Programmable versions of the XBee-PRO ZB enable custom ZigBee application development
- Advanced configuration options available via simple AT or API commands
- SMT form factor with side castellations for easy soldering
- 15 general-purpose I/O lines
- Link budgets of 110 dB for XBee and 119 dB for XBee-PRO ZB
- Industry leading sleep current of sub 1uA
- Firmware upgrades via UART, SPI or over the air

Related Products

Digi

www.digi.com

Platform	XBee® ZB SMT		XBee-PRO® ZB SMT	
	Standard	Programmable	Standard	Programmable
Performance				
Data Rate	RF 250 Kbps, Serial up to 1 Mbps			
Indoor/Urban Range	200 ft (60 m)		200 ft (60 m)	
Outdoor/RF Line-of-Sight Range	4000 ft (1200 m)		2 mile (3200 m)	
Transmit Power	2.1 mW (+3 dBm) / 6.3 mW (+8 dBm) boost mode		63 mW (+18 dBm) / 100 mW (+20 dBm)	
Receiver Sensitivity (1% PER)	-100 dBm / -102 dBm boost mode		-102 dBm	
Features				
Serial Data Interface	UART, SPI			
Configuration Method	API or AT commands, Local or over the air			
Frequency Band	ISM 2.4 GHz			
Interference Immunity	DSSS (Direct Sequence Spread Spectrum)			
ADC Inputs	(4) 10-bit ADC inputs			
Digital I/O	11			
Antenna Options	PCB (embedded), U.FL and RP pad			
Operating Temperature	-40° C to +85° C			
Dimensions (L x W x H) and Weight	0.87 in x 1.22 in x 0.12 in (2.20 cm x 3.60 cm x 0.30 cm); 1.60 oz (10.00 g)			
Programmability				
Memory	N/A	32 KB Flash / 2 KB RAM	N/A	32 KB Flash / 2 KB RAM
CPU Clock Speed	N/A	HC908 / up to 50.33 MHz	N/A	HC908 / up to 50.33 MHz
Networking and Security				
Encryption	128-bit AES			
Reliable Packet Delivery	Retries/Acknowledgements			
IDs	PAN ID and addresses, cluster IDs and endpoints (optional)			
Channels	16 channels	15 channels	16 channels	15 channels
Power Requirements				
Supply Voltage	2.1 to 3.6 V		2.7 to 3.6 V	
Transmit Current	33 mA @ 3.3 VDC / 45 mA boost mode	47 mA @ 3.3 VDC / 59 mA boost mode	300 mA @ 3.3 VDC	116 mA @ 3.3 VDC
Receive Current	29 mA @ 3.3 VDC / 31 mA boost mode	42 mA @ 3.3 VDC / 45 mA boost mode	31 mA @ 3.3 VDC	45 mA @ 3.3 VDC
Power-Down Current	<1 uA @ 25° C	3.5 uA @ 25° C	<1 uA @ 25° C	3.5 uA @ 25° C
Regulatory Approvals				
FCC, IC (North America)	Yes			
ETSI (Europe)	Yes		No	
C-TICK (Australia)	Pending			
TELEC (Japan)	Pending		No	



Visit www.digi.com for part numbers.

DIGI SERVICE AND SUPPORT - You can purchase with confidence knowing that Digi is here to support you with expert technical support and a one-year warranty. www.digi.com/support

Digi International	Digi International France	Digi International KK	Digi International (HK) Limited
877 912-3444	+33 1 55 61 98 98	+81 3 5428 0261	+852 2833 1008
952 912-3444	www.digi.fr	www.digi-intl.co.jp	www.digi.cn
info@digi.com			

Wireless M2M

91001620 C1/1110

BUY ONLINE • www.digi.com

Digi