Diseñar e implementar una red de comunicación inalámbrica para la experimentación en robótica de enjambre

Marlon Josué Castillo Martínez

UVG UNIVERSIDAD DEL VALLE DE GUATEMALA

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

Diseñar e implementar una red de comunicación inalámbrica para la experimentación en robótica de enjambre

Trabajo de graduación en modalidad de Tésis presentado por
Marlon Josué Castillo Martínez
Para optar al grado académico de Licenciado en Ingeniería Electrónica

Guatemala, enero del 2019

Vo.Bo.:

(f) _____

Ing. Pablo Daniel Mazariegos

Tribunal Examinador:

(f) _____

Ing. Pablo Daniel Mazariegos

(f) _____

MSc. Carlos Alberto Esquit

(f) _____

PhD. Luis Alberto Rivera

Fecha de aprobación: Guatemala, 4 de diciembre de 2018.

Agradecimientos

A Dios, por darme la fuerza necesaria en los momentos más difíciles de mi carrera y por haberme rodeado de amigos y familia que me alentaron a seguir adelante en todo momento. Gracias porque me diste la oportunidad de seguir mis sueños.

A mis padres, por alentarme y apoyarme a que luchara por mis sueños. Gracias porque todos los días trabajan arduamente para poder brindarme todo lo necesario, son mi mayor ejemplo y a ustedes dedico todo los logros durante mi carrera.

A la Fundación Juan Bautista Gutiérrez, pero en especial a Doña Isabel Gutiérrez de Bosch. Gracias por creer en los sueños y el talento de un joven que desea un mejor futuro para su familia y país. Sin su apoyo nada de esto hubiera sido posible.

A Cesar Azurdia, por brindarme su apoyo en la realización de este trabajo. Gracias porque aún sin conocerme me brindó un espacio de su tiempo para guiarme correctamente durante el diseño de la red de comunicación.

A Miguel Zea, por creer en mi talento y capacidad para trabajar en este proyecto. Gracias por el constante apoyo, sin eso este trabajo no hubiera llegado tan lejos.

A Pablo Mazariegos, por exigirme a trabajar con calidad y excelencia. Gracias por enseñarme a que todo se debe hacer de la mejor manera posible y que siempre se puede mejorar.

Índice

Lista de Figuras

Lista de Cuadros

Resumen

De forma general, el proyecto consiste en la construcción de una mesa de pruebas para la experimentación en robótica de enjambre, se busca que esta plataforma sea lo más económica posible y fácilmente escalable. El proyecto se estructura en dos partes principales, la primera es dedicada al procesamiento de imágenes para diseñar algoritmos de posicionamiento y la segunda parte será el enfoque principal de este trabajo en el cual se diseñará e implementará una red de comunicación inalámbrica para el envío de información entre un servidor y los módulos de comunicación de los robots en la mesa de pruebas.

Antes del diseño e implementación de la red, se evaluará distintas tecnologías inalámbricas y se seleccionará la que presente mejores características, dentro de las cuales se estarán evaluando principalmente el costo de implementación y velocidad de transmisión. Luego se seleccionará un módulo de comunicación de bajo costo y de fácil implementación para que pueda ser fácilmente escalable en número la red de módulos y puedan ser utilizados sin problemas en futuros diseños o mejoras del proyecto.

Luego de la implementación de la red de comunicación, se medirá la velocidad de transferencia máxima en la red. Por último, se medirá la velocidad de comunicación mínima que debe existir entre el microcontrolador y el módulo inalámbrico para que se pueda aprovechar al máximo el ancho de banda de la red.

CAPÍTULO 1

Introducción

La presente investigación se centra en el campo de la robótica de enjambre pero principalmente se enfoca en la construcción de una mesa de pruebas para investigación en dicho campo. Para llevar a cabo la construcción de la mesa de pruebas, este trabajo se dedicó al diseño e implementación de la red de comunicación inalámbrica. Se buscó que todo el proyecto fuera lo más económico posible para que pueda ser fácilmente replicable, pero también se buscó que sea lo suficientemente robusto para que pueda ser utilizado tanto en investigaciones de robótica de enjambre como para cursos universitarios de distintas áreas en electrónica, mecatrónica, computación, entre otras. Esta investigación se realizó por el interés de crear un espacio en el cual estudiantes e investigadores, tanto externos como internos, puedan poner a prueba algoritmos de robótica de enjambre dentro de un ambiente controlado, y por supuesto también permite pasar de modelos teóricos a modelos experimentales más aproximados a la realidad.

Para llevar a cabo dicha investigación se siguió cuatro pasos generales, investigación y evaluación, diseño, implementación, y medición. Primero se evaluó las tecnologías y módulos inalámbricos que mejor se adapten a las necesidades de la mesa de pruebas y en base a esto se realizó la selección. Luego se comenzó con el diseño de la red de comunicación, se evaluó nuevamente cada parámetro de diseño y se seleccionó los que permitían un mejor rendimiento en la red. Por último, se procedió a la implementación de la red de comunicación utilizando la tecnología, módulos y parámetros seleccionados en las etapas anteriores.

Por último, se procedió a la medición de la velocidad de transmisión máxima obtenida en la red y a determinar la velocidad de comunicación mínima entre el módulo inalámbrico y el microcontrolador que permita aprovechar al máximo el ancho de banda de la red.

CAPÍTULO 2

Objetivos

2.1. Objetivo General

Diseñar e implementar una red de comunicación inalámbrica en una mesa de pruebas para la experimentación en robótica de enjambre.

2.2. Objetivos Específicos

- Investigar, evaluar y seleccionar la tecnología de comunicación inalámbrica y el módulo de recepción/transmisión inalámbrica que mejor se adapten a las necesidades de la mesa de pruebas y que sean de bajo costo.
- Diseñar e implementar la red de comunicación en base a la tecnología inalámbrica seleccionada.
- Determinar la velocidad de transmisión máxima en la red de comunicación inalámbrica.
- Determinar la velocidad de comunicación mínima entre el módulo inalámbrico y el microcontrolador que permita aprovechar al máximo el ancho de banda de la red.

CAPÍTULO 3

Justificación

Lamentablemente en Guatemala y Latinoamérica las Universidades no cuentan con plataformas de experimentación en el campo de la robótica de enjambre. Para poder competir con Universidades extranjeras de alto prestigio en este campo es fundamental que se comience con la investigación y construcción de este tipo de ambientes de experimentación. Estos ambientes ya existen en otras universidades extranjeras, pero cabe destacar que puede llegar a ser muy costoso replicarlos o utilizarlos. Por lo cual, este proyecto busca el diseño e implementación de una red de comunicación inalámbrica de bajo costo y fácilmente escalable en una mesa de pruebas para la experimentación en robótica de enjambre, con la cual estudiantes de la Universidad del Valle de Guatemala y personal externo puedan realizar investigación en este campo dentro de un ambiente controlado. Es importante la investigación en el campo de la robótica de enjambre ya que con ello se pueden crear modelos de interacción entre robots, algoritmos de trabajo en equipo, entre otros. Un claro ejemplo de aplicación de la robótica de enjambre es la interacción entre robots para búsqueda y rescate en desastres naturales en lugares que sean peligrosos para los cuerpos de rescate humanos.

El campo de la robótica de enjambre lleva poco tiempo de investigación y en la actualidad hay muy pocas plataformas que permiten hacer investigación en este campo, por lo cual, esta investigación permitirá a la Universidad del Valle de Guatemala ser una de las primeras universidades no solo en Latinoamérica si no que también en el resto del mundo que cuenten con una plataforma de pruebas para robótica de enjambre diseñada y construida por estudiantes guatemaltecos.

Este trabajo se enfoca en el diseño e implementación de la red inalámbrica de comunicación, este módulo es fundamental debido a que una red de comunicación cableada presentaría grandes dificultades al momento de tener una gran cantidad de robots interactuando entre ellos. Por lo cual, una red inalámbrica presenta grandes ventajas como alta escalabilidad, fácil implementación, otorga maniobrabilidad a los robots, entre otros.

CAPÍTULO 4

Marco Teórico

4.1. CAPAS DEL MODELO OSI

El modelo OSI es una estructura utilizada para resolver el problema de las comunicaciones de datos y redes informáticas [8], este modelo se divide en siete capas o niveles. En un esquema de comunicación los usuarios finales como las PC, celulares, servidores, entre otros, implementan todas las capas mientras que los dispositivos intermedios como routers, switch y puntos de acceso solamente implementan cierta cantidad de capas de este modelo.

La figura 4.1 muestra un proceso de comunicación con el modelo OSI, cuando se envían datos estos pasan de forma descendente por el modelo OSI pero cuando se reciben datos estos pasan de forma ascendente. En el proceso de envío se van agregando bits al bloque original, este proceso se conoce como codificación, pero cuando se recibe se van eliminando bits al pasar por cada capa hasta obtener el mensaje original, este proceso se conoce como decodificación.

4.1.1. Capa Física

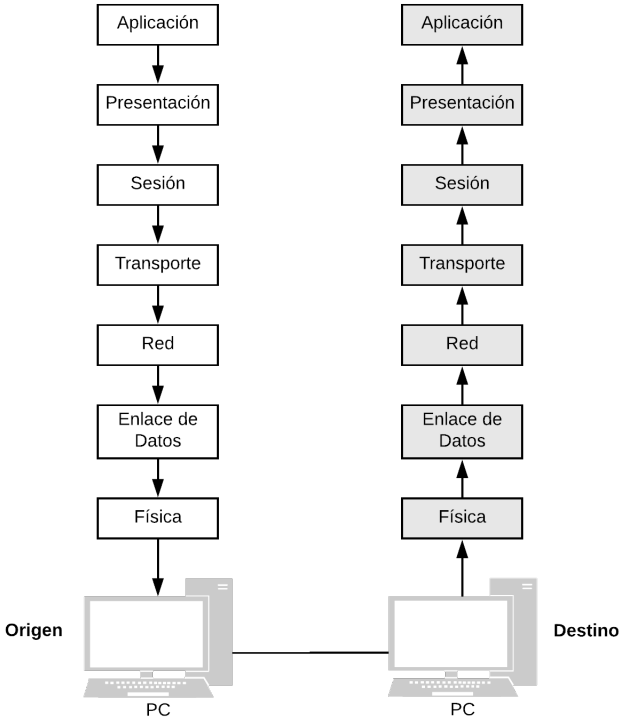
Esta se encarga de las tareas de transmisión física de señales eléctricas, ópticas, electromagnéticas, etc. Esta capa puede limitar la velocidad de transmisión y la interferencia o bits erróneos que llegan a su destino. La velocidad de transmisión se ve afectada principalmente por el medio que se encarga de transportar las señales físicas, ejemplo de estos medios son el aire, la fibra óptica o los cables de cobre [8].

El porcentaje de bits erróneos recibidos se debe principalmente a las interferencias en el medio, ejemplo de esto son la superposición de señales en el aire o la inducción electromagnética en los cables. Aunque este porcentaje de bits erróneos se debe a las interferencias en el medio físico es posible implementar algoritmos que disminuyan este porcentaje.

4.1.2. Capa de Enlace de Datos

Proporciona un servicio similar a la capa física permitiendo mejorar la fiabilidad de la transmisión. Esta capa añade bits adicionales al bloque que forma el mensaje, a esta combinación se le denomina

Figura 4.1: Modelo OSI.



[Elaboración propia]

trama [8]. Estos bits adicionales ayudan a detectar errores y también permiten que cuando se detectan dichos errores se haga una petición de retransmisión.

Otra tarea de estos bits adicionales es el control de flujo, esto es importante cuando el receptor no puede procesar las tramas tan rápido como las recibe, entonces el receptor utiliza los bits en la trama para indicarle al transmisor que se detenga el tiempo necesario para que pueda procesar los datos [8]. Un dispositivo de red que trabaja principalmente en esta capa es el switch, que también hace uso de direcciones MAC para identificar a los equipos participantes en la comunicación.

4.1.3. Capa de Red

Un dispositivo de red que trabaja principalmente en esta capa es el router. Esta capa añade bits a la trama y a esta combinación se le denomina paquete, su función principal es permitir que los datos lleguen de su origen al destino [8]. Para cumplir este objetivo la capa de red puede trabajar en dos modos:

- Datagramas: los paquetes son enviados sin que el origen y el destino hayan establecido una comunicación previa, esto puede causar que los paquetes lleguen fuera de orden, duplicados o incluso que no lleguen [8].
- Circuitos virtuales: garantizan la entrega de los paquetes y en el orden correcto, esto se debe a que el origen y destino establecen una comunicación previa en la cual un dispositivo intermedio

como un router establece un canal virtual por el cual se transmiten los paquetes [8].

En esta capa se hace uso del direccionamiento IP para identificar a los dispositivos que participan en la comunicación, esto permite que los routers identifiquen la interfaz por la cual deben enviar los paquetes.

4.1.4. Capa de Transporte

Se encarga de obtener una conexión fiable en cualquier tipo de red, en otras palabras, es el responsable de controlar las deficiencias de las transmisiones. Para cumplir este objetivo la capa de transporte puede hacer uso de uno de los siguientes protocolos:

- Protocolo de datagramas de usuario (UDP): no orientado a la conexión, se encarga de enviar paquetes IP sin que el origen y destino hayan establecido una conexión previa. Puede ser rápido por su simplicidad, pero inseguro debido a que no garantiza la entrega y tampoco tiene control de flujo de los paquetes [8].
- Protocolo de control de transmisión (TCP): orientado a la conexión, se encarga de enviar paquetes IP por una conexión previa entre el origen y destino. Garantiza la entrega de los paquetes y también implementa control de flujo, debido a esto es uno de los protocolos más utilizados en las redes de comunicación [8].

4.1.5. Capa de Sesión

Se encarga de gestionar y organizar las conexiones o el intercambio de datos, también se encarga del uso de los protocolos de sincronía de las aplicaciones [8].

4.1.6. Capa de Presentación

Se encarga de trabajar como traductor entre las diferentes plataformas de la red, en otras palabras, se encarga de establecer una forma universal de codificar la información. En esta capa se utilizan algoritmos de compresión para reducir el tamaño de los mensajes y algoritmos de cifrado para proteger la información que se envía por la red [8].

4.1.7. Capa de Aplicación

Se encarga de permitir el acceso a los servicios del resto de capas. En esta capa trabajan los servicios utilizados por las aplicaciones, es decir, el usuario no tiene interacción directa con la capa de aplicación si no que los programas utilizados por el usuario son los que interactúan con esta capa para luego acceder al resto de capas en el modelo OSI [8]. Por ejemplo, un usuario hace uso de un navegador para acceder a páginas web pero el navegador hace uso del protocolo HTTP de la capa de aplicación para ofrecer el servicio de páginas web.

4.2. TECNOLOGÍAS INALÁMBRICAS

Las tecnologías inalámbricas han surgido por la necesidad de eliminar las comunicaciones cableadas y así reducir costos. Estas tecnologías utilizan la modulación de ondas electromagnéticas

en el espacio. Con los años han surgido diferentes estándares y tecnologías para las comunicaciones inalámbricas, las tecnologías de más uso en la actualidad son Bluetooth, WiFi y LTE.

4.2.1. Bluetooth

Bluetooth es una tecnología de corto alcance y bajo consumo que permite la conexión inalámbrica entre dispositivos. Está definida por el estándar IEEE 802.15.1, utiliza la banda de 2.4 GHz y puede operar a velocidades de 1 a 24 Mbps. Al igual WiFi, divide el espectro en 79 canales de 1 MHz de ancho cada uno. Cuenta con un modo jerárquico de operación Master-Slave en el cual un Master puede conectarse directamente con hasta 7 esclavos, esta configuración se denomina piconet. Bluetooth permite la conexión entre piconets utilizando dispositivos que actúen como puente, a la red formada por piconets se le denomina scatternet [7]. Sus características son las siguientes:

- Bajo consumo de energía.
- Tiempo de conexión corto.
- Fácil implementación.
- Corto alcance.
- Baja velocidad de transmisión.
- Número reducido de dispositivos por piconet.

Debido a las características anteriores Bluetooth es utilizado comúnmente en aplicaciones donde no se requiere una velocidad de transmisión alta y se requiere un bajo consumo de energía, por ejemplo, la conexión de periféricos como teclados, mouse, auriculares, bocinas, entre otras.

4.2.2. WiFi

Al igual que Bluetooth utiliza la banda de 2.4 GHz, pero también permite el uso de 5 GHz. Está definida por el protocolo IEEE 802.11 que actúa en la capa física y de enlace de datos del modelo OSI. Según la versión de este protocolo divide el espectro en diferente cantidad de canales. Actualmente permite velocidades de 11 a 600 Mbps [5]. Sus características son las siguientes:

- Fácil implementación.
- Largo alcance.
- Alta velocidad de transmisión.
- Gran cantidad de dispositivos por red.

Debido a estas características WiFi es utilizado comúnmente en las redes de computadores conocidas como WLAN.

4.2.3. LTE

Esta tecnología permite satisfacer la creciente demanda de los usuarios y redes. Utiliza bandas de frecuencia licenciadas, esto quiere decir que una entidad compra la banda de frecuencia y es la única que puede permitir la transmisión de datos. LTE cuenta con canales de carga y descarga separados, en la descarga utiliza modulación OFDMA y en la carga SC-FDMA [15]. Sus características son las siguientes:

- Largo alcance.
- Alta velocidad de transmisión.
- Gran cantidad de dispositivos interconectados.
- Uso de bandas de frecuencia adaptables.
- Costo alto debido al uso de bandas licenciadas.

Debido a estas características LTE es utilizado en las redes de telefonía móvil.

4.3. WLAN

En [13] se define que una WLAN (Wireless Local Area Network) es una red local que utiliza como canal de transmisión las emisiones de radiofrecuencia, este tipo de redes son muy utilizadas como alternativa a las redes LAN cableadas o como extensión de ésta. Dentro de las áreas de cobertura local se pueden considerar: oficinas, universidades, hoteles, aeropuertos, hogares, entre otros. Este tipo de red se encuentra bajo el estándar IEEE 802.11 y dentro de sus ventajas se encuentran las siguientes:

- Movilidad y conectividad.
- Rapidez y flexibilidad.
- Bajo coste e instalación sencilla.

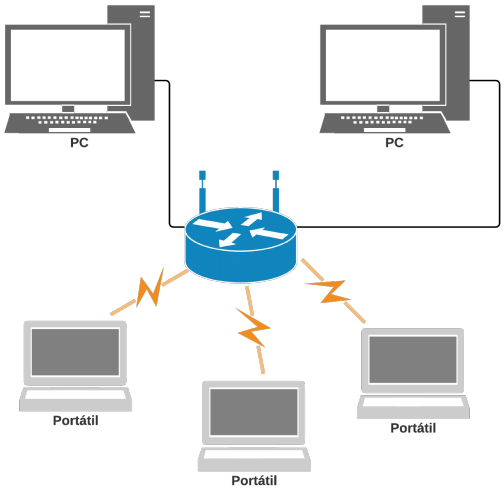
En la figura 4.2 se observa una topología común de una red WLAN, vemos que el punto de acceso funciona como un bridge entre las redes cableadas y las inalámbricas. Un bridge opera en el nivel de enlace del modelo OSI, este dispositivo emplea direcciones MAC para llevar a cabo acciones de filtrado, envío de tráfico e inundación [13].

4.4. PROTOCOLO IEEE 802.11

La IEEE creó el estándar 802.11 para redes inalámbricas con el fin de que fabricantes de hardware inalámbrico, empresas de telecomunicaciones y toda entidad relacionada en este campo pudiera tener compatibilidad con los productos de cualquier otra. Este estándar define las normas de la capa física y capa de enlace de datos del modelo OSI [12], con los años se ha desarrollado una gran cantidad de estándares, pero los más conocidos y utilizados son 802.11a, b, g y n.

- **802.11a:** ratificado en 199, utiliza modulación OFDM y gracias a esto puede llegar a alcanzar velocidades de transmisión de hasta 54 Mbps, aunque experimentalmente se logra llegar a 22

Figura 4.2: Topología básica de una WLAN.



[Elaboración propia]

Mbps. Utiliza la banda de 5 GHz, esta banda se encuentra menos saturada que la de 2.4 GHz lo cual reduce la interferencia en la transmisión de datos. Pero con esto se reduce la distancia a la cual se puede transmitir ya que la longitud de onda es más pequeña y estas señales son absorbidas más fácilmente por paredes. Divide el espectro en 12 canales sin solapamiento, no es compatible con el 802.11b a menos que el equipo esté diseñado para trabajar en ambos protocolos [12].

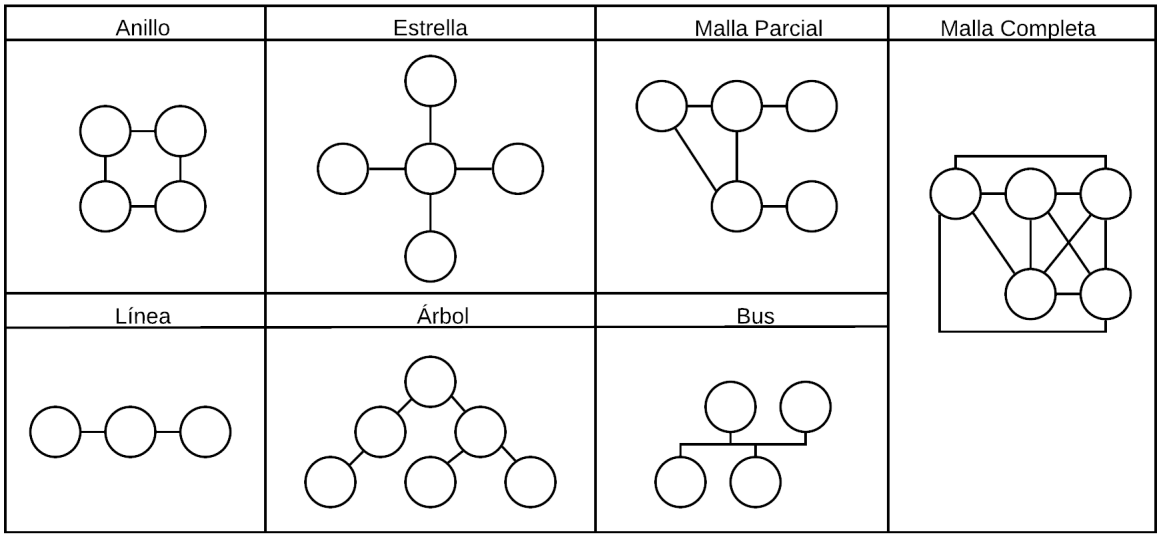
- **802.11b:** ratificado en 1999, su velocidad teórica es de 11 Mbps, aunque experimentalmente llega a 6 Mbps en conexiones TCP y 7 Mbps en UDP. Utiliza la banda de 2.4GHz y CSMA/CA como método de acceso al medio, utiliza modulación CCK en la capa física. Divide el espectro en 14 canales, cada uno con un ancho de banda de 22 MHz, lo cual provoca que cada canal pueda interferir con los más cercanos de cada lado. Por este motivo se recomienda utilizar los canales 1, 6 y 11 en redes que funcionen bajo este protocolo [12].
- **802.11g:** ratificado en 2003, utiliza modulación OFDM y puede alcanzar velocidades de hasta 54 Mbps, aunque al igual que el 802.11a en un escenario real puede llegar a alcanzar 22 Mbps. Utiliza la banda de 2.4 GHz y es compatible con los dispositivos 802.11b, lo cual hace que tenga las mismas características en cuanto a la división del espectro [12].
- **802.11n:** ratificado en 2009, compatible con tecnología MIMO (Multiple Input-Multiple Output) que permite utilizar varios canales a la vez para enviar y recibir datos con la incorporación de varias antenas. Esto permite alcanzar una velocidad teórica de hasta 600 Mbps, aunque en un escenario real se alcanza 100 Mbps. Puede trabajar en las bandas de 2.4 GHz y 5 GHz lo cual permite que sea compatible con los estándares 802.11a/b/g, utiliza modulación MCS [12].

4.5. TOPOLOGÍAS DE RED

La topología de una red representa la disposición de las conexiones en los nodos de una red. A continuación, se presentan las topologías utilizadas en redes según [6]. Una línea en la figura representa una conexión y no el medio físico que los conecta, ya que en redes inalámbricas todos los nodos comparten el mismo medio físico, el espacio [6].

- **Bus:** Todos los nodos comparten una misma conexión.
- **Estrella:** Cada nodo se conecta directamente a un nodo central, también conocido como concentrador. En una topología de estrella todos los datos pasan a través del nodo central antes de alcanzar su destino.
- **Línea:** Un conjunto de nodos conectados en una línea. Cada nodo se conecta a sus dos nodos vecinos excepto el nodo final e inicial.
- **Árbol:** Redes en estrella interconectadas entre sí a través de un nodo de enlace, el fallo de un enlace no significa el fallo total de la red.
- **Anillo:** Todos los nodos se conectan entre sí formando un lazo cerrado, a diferencia de la topología línea en esta configuración todos los nodos tienen conexión a sus dos nodos vecinos.
- **Malla Completa:** Existe enlace directo entre todos los nodos de la red. Una malla completa con n nodos requiere de n(n-1)/2 enlaces directos.Es la más costosa pero también la más confiable.
- **Malla Parcial:** Algunos nodos están conectados en una malla completa, mientras otros se conectan solamente a uno o dos nodos de la malla.

Figura 4.3: Topologías de red.



[Elaboración propia]

4.6. MODOS DE OPERACIÓN EN REDES INALÁMBRICAS

El estándar IEEE 802.11 establece dos modos para redes inalámbricas:

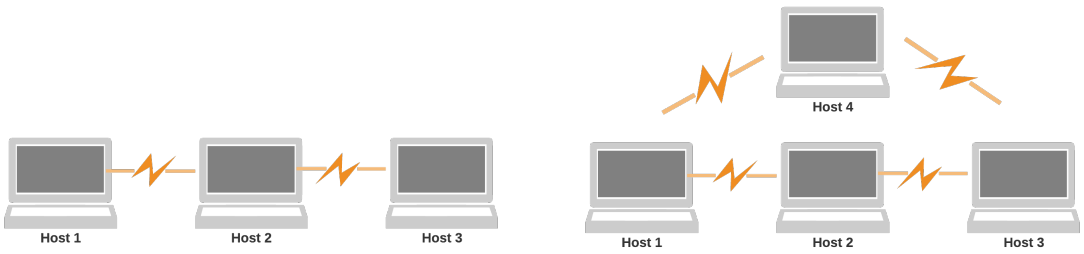
- Ad hoc
- Infraestructura

Los modos son independientes de la topología. Por ejemplo, una red en estrella puede ser implementada en modo ad-hoc o infraestructura. El modo puede ser visto como la configuración individual de la tarjeta inalámbrica de un nodo, más que como una característica de toda la red [6].

4.6.1. Ad-hoc

También conocido como punto a punto, los nodos inalámbricos pueden establecer una comunicación directa entre sí sin la necesidad de involucrar un punto de acceso central. Un inconveniente principal en esta red es que su rendimiento disminuye a medida que el número de nodos crece [6].

Figura 4.4: Configuración de red ad-hoc.



(a) Topología línea.

(b) Topología anillo.

[Elaboración propia]

En la figura 4.4 se puede observar distintas topologías en modo ad-hoc, se observa que no se necesita un nodo central para enviar mensajes de un host a otro. Por ejemplo, para enviar un mensaje del host 1 al 3 en 4.4a este debe pasar por el host 2, pero el host 2 puede enviar mensajes directamente al host 1 y 3. Con la topología en 4.4b al enviar un mensaje del host 1 al 3 este puede pasar por el host 4 o el host 2.

4.6.2. Infraestructura

En este modo hay un elemento central conocido como punto de acceso y los nodos son conocidos como clientes. Si el punto de acceso se conecta a una red Ethernet cableada, los clientes inalámbricos pueden acceder a la red cableada a través del punto de acceso. La figura 4.2 es un ejemplo de una red en modo infraestructura, en la cual el punto de acceso es el router que conecta los clientes inalámbricos con la red cableada y todos los paquetes enviados desde un nodo a otro pasan por el router [6].

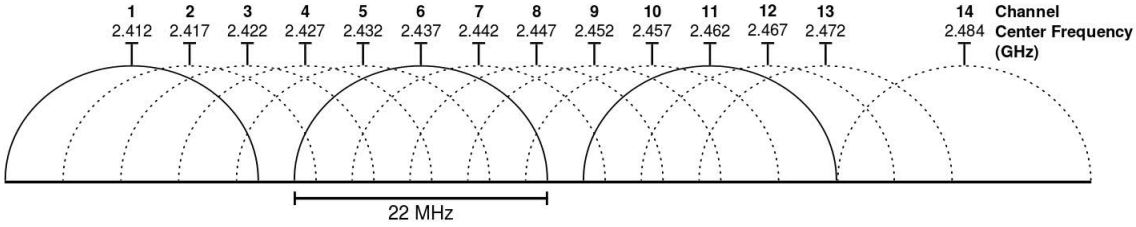
4.7. MODULACIÓN Y CANALES EN PROTOCOLO 802.11

WiFi opera principalmente en la banda de 2.4 a 2.4835 GHz, esto nos da un ancho de banda de 83.5 MHz para transmitir datos [10]. Para aprovechar al máximo este ancho de banda se utilizan diferentes tipos de modulación y se divide el espectro en canales.

4.7.1. Canales

En el 802.11b/g/n la banda de 2.4 GHz se divide en 14 canales con un ancho de banda de 22 MHz por canal. Cada canal está desplazado 5 MHz con respecto al anterior, por lo que los canales contiguos se solapan. Por lo general se permite el uso de 13 canales del espectro, aunque esto es regulado por cada país. En la figura 4.5 se observa que hay tres canales que no interfieren entre sí, estos canales son el 1, 6 y 11. Debido a esto, en determinadas áreas con varias redes se suele hacer uso de estos canales para evitar el solapamiento [10].

Figura 4.5: Canales de comunicación en WiFi.



[1]

4.7.2. Modulación

El principio de cualquier modulación es trasladar la información de una banda de frecuencias a otra. La modulación marca la velocidad máxima a la que podemos aspirar dentro de un estándar, todas las modulaciones usadas por los estándares de la familia IEEE 802.11 son digitales. Las modulaciones digitales tienen ciertas ventajas sobre las modulaciones analógicas, como por ejemplo, el aumento en la inmunidad al ruido, ahorro del ancho de banda con la multiplexación y compresión de datos, entre otras [10]. Las modulaciones están determinadas por las distintas variaciones del protocolo 802.11, es un factor sobre el cuál no se tiene control si no que simplemente viene por defecto en cada protocolo. Por tal motivo, no es necesario conocer a profundidad el funcionamiento de las modulaciones utilizadas por cada protocolo, pero si debe conocerse las características ofrecidas en cada variación del 802.11.

4.8. SEGURIDAD EN REDES INALÁMBRICAS

Un problema común en redes WiFi es la seguridad, el aire es un medio de propagación de libre acceso, y por tanto una red inalámbrica puede verse sometida a interceptación de paquetes, acceso no autorizado, usurpación y suplantación de identidad, interferencias aleatorias, denegación de servicio, etc. Para resolver este problema se necesita de mecanismos que garanticen la seguridad de la red, un mecanismo utilizado en la seguridad es la autenticación de las estaciones. Dentro de los métodos de autenticación y encriptación se encuentra WEP, WPA y WPA2 [10].

4.8.1. WEP

Un punto de acceso debe de autenticar a una estación antes de que ésta se asocie al mismo. Lo que se autentica con WEP son las estaciones, y no los usuarios. Usa un mecanismo de desafío/respuesta

con una clave secreta (de 64 o 128 bits) compartida por la estación y el punto de acceso, de manera que se niega el acceso a todo aquel que no tenga la clave asignada [10].

La estación envía una solicitud de autenticación al punto de acceso, y éste le responde con un texto desafío de 128 bits generado con un generador de números pseudo-aleatorios (PRGN), la clave secreta y el vector de inicialización (IV). La estación debe encriptar el texto de desafio con ayuda de su clave WEP y el IV, y enviarlo al punto de acceso. El punto de acceso lo desencriptará y lo comparará con el texto de desafío original. Si coinciden autenticará a la estación, en caso contrario fallará la autenticación [10].

La principal debilidad de WEP es que la clave compartida que se usa para encriptar y desencriptar las tramas de datos es la misma que la que se usa para la autenticación. Otra debilidad importante es que siempre se utiliza la misma clave de encriptación en la red.

4.8.2. WPA

Utiliza el método PSK (Pre-Shared Key). Solamente se necesita introducir una clave maestra o PSK en cada uno de los puntos de acceso y de las estaciones que conforman nuestra WLAN. Solamente podrán acceder al punto de acceso los dispositivos móviles cuya contraseña coincida con la del punto de acceso. Esta PSK nunca se transmite por el aire ni se utiliza para encriptar el flujo de datos, solamente se utiliza para iniciar el proceso de claves dinámicas TKIP (Temporal Key Encryption Protocol), por lo que es mucho más seguro que WEP. Se recomienda que las contraseñas empleadas sean largas (20 o más caracteres), porque se ha comprobado que WPA es vulnerable a ataques de diccionario si se utiliza una contraseña corta [10].

La principal ventaja de WPA sobre WEP es que la clave de encriptación y autenticación son independientes entre sí. Otra diferencia importante con WEP es que la clave de encriptación cambia constantemente, pero esta clave cambia de forma sincronizada entre el punto de acceso y la estación de tal forma que solo ellos conocen la clave de encriptación. Esto permite tener mayor seguridad en la encriptación de datos [10].

4.8.3. WPA2

Muy similar a WPA pero con algunas mejoras. La principal mejora con respecto a WPA es el método de encriptación, cambia TKIP por AES (Advanced Encryption Standard). La principal debilidad de WPA2 es que es susceptible a ataques DoS (Denial of Service) mediante el envío de paquetes basura (encriptados erróneamente) hacia la red. Si el punto de acceso recibe dos paquetes que no superan el código de integridad de mensaje (MIC) en un intervalo de 60 segundos, significará que la red está bajo un ataque activo y como resultado el punto de acceso tomará medidas drásticas como la disociación de cada estación. Esto previene los ataques DoS pero también causa una pérdida de conexión temporal en la red [10].

4.9. SOCKETS EN C++

Según [17] un socket es utilizado para crear y administrar puntos finales de comunicación, cada socket puede estar vinculado a una dirección local y remota. Estas direcciones definen la asociación entre dos o más pares que se comunican a través del socket. La API Socket contiene aproximadamente dos docenas de funciones del sistema que se pueden clasificar en las siguientes cinco categorías según [17]:

1. **Gestión local:** funciones que administran la información que normalmente se almacena dentro del núcleo del sistema operativo o en las bibliotecas del sistema.
2. **Conexión:** funciones encargadas de establecer o finalizar las conexiones.
3. **Transferencia de datos:** funciones que permiten el envío y recepción de datos a través de sockets.
4. **Gestión de opciones:** funciones que permiten a los programadores alterar el comportamiento del socket predeterminado para permitir multicasting, broadcasting, y la modificación del tamaño de los buffers de transporte.
5. **Direccionamiento de red:** funciones que permiten resolver nombres de dominio, como www.socket.com, a direcciones de red de bajo nivel, como 192.168.1.1.

Para la configuración de un socket se le debe indicar tanto la familia de protocolos que utilizará así como también la familia de direcciones en la cual podrá trabajar. Las funciones y sus parámetros se encuentran de forma más detallada en el API de sockets para C++.

4.10. ROBÓTICA DE ENJAMBRE

En [18] se describe que la robótica de enjambre(Swarm robotics) consiste en la coordinación de sistemas con un gran número de robots móviles que por lo general suelen ser simples individualmente. El comportamiento colectivo de estos sistemas surge de las interacciones entre los robots y su entorno. Este campo de estudió surgió gracias a la necesidad de modelar la interacción en los enjambres de abejas, bancos de peces, manadas, y cualquier otro tipo de interacción colectiva. Una característica especial de la robótica de enjambre es que cada individuo por si solo es simple, sin mayor utilidad y que tampoco puede realizar tareas complejas, pero a medida que se incrementa el número de individuos las tareas que pueden realizar son más complejas. Por ejemplo, imaginemos grandes cantidades de pequeños robots que interactúan entre sí para mover bloques pesados, un solo robot por si solo no es capaz de moverlo pero al interactuar con el resto de compañeros estos se unen y son capaces de realizar dicha tarea.

La robótica de enjambre se basa en el diseño de robots simples, pequeños y de bajo costo para que la población pueda ser fácilmente escalable. Las aplicaciones de este campo según [18] pueden ser:

- Búsqueda y rescate.
- Mapeo de terrenos peligrosos.
- Simulación de modelos de interacción colectiva.
- Inteligencia colectiva.

Algunas características importantes de estos sistemas son:

- Gran cantidad de individuos.
- Control centralizado o descentralizado.
- Flexibilidad.
- Alta escalabilidad.
- Tolerante a la eliminación o adición de individuos al sistema.

4.11. MÓDULO INALÁMBRICO ESP8266 v1

El ESP8266 es un módulo WiFi utilizado comúnmente para aplicaciones de IoT(Internet of Things) debido a su bajo costo y fácil implementación. En la actualidad existen muchas versiones de este módulo, pero la versión 1 es la más utilizada debido a la gran documentación que existe por parte de distintos usuarios y a su muy bajo costo. A pesar de ser la primera versión, está ofrece una gran cantidad de prestaciones ya que permite configurar muchos parámetros en el diseño de la red WiFi. Este módulo se comunica con comandos AT a través de un puerto serial.

El módulo ESP8266 funciona gracias a un microprocesador de 32 bits de la serie L106 Diamond de Tensilica con una velocidad de reloj de 160 MHz, permite la comunicación con otros dispositivos a través de sus GPIOs y un puerto UART. Tiene integrada una antena, amplificador de potencia, amplificador de recepción de bajo ruido, filtros y módulos de administración de energía [4]. Características principales:

- Soporta los protocolos IEEE 802.11 b/g/n y los 14 canales en la banda de 2.4 GHz.
- Permite trabajar con los modos Station, P2P y SoftAp.
- Soporta MIMO 1x1 y 2x1.
- La potencia en Tx es de +20, +17, +14 dBm en 802.11 b, g y n, respectivamente.
- La sensibilidad en Rx es de -91, -75 y -72 dBm en 802.11 b, g y n, respectivamente.
- Puede alcanzar una velocidad de transmisión máxima de 54 Mbps.
- Tiene un modo sleep para ahorro de energía.
- Opera de 2.5-3.6 V y con un consumo máximo de 170, 140 y 120 mA utilizando el 802.11 b, g y n, respectivamente.
- Trabaja con seguridad WPA, WPA2 y WEP.
- Acepta IPv4 y comunicación con TCP y UDP.

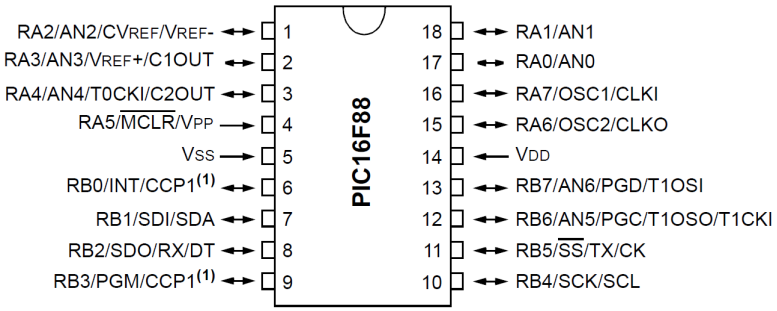
4.12. PIC16F88

El PIC16F88 es un microcontrolador fabricado por Microchip, de muy bajo costo y con muy buenas características. Dentro de sus características principales se puede mencionar:

- Oscilador interno de 8 MHz y externo de 20 MHz.
- PWM de 10 bits.
- Comunicación por puerto UART, SPI o I2C.
- Conversor Analógico/Digital de 7 canales y 10 bits.
- Timer de 8 bits.
- 16 pines I/0.

Tiene un arquitectura de 8 bits, es de bajo consumo, en total tiene 18 pines y utiliza un set de instrucciones RISC.

Figura 4.6: Diagrama de pines.



[3]

CAPÍTULO 5

Antecedentes

5.1. EL ROBOTARIUM DE GEORGIA TECH

En septiembre de 2016 se publicó el artículo *Robotarium: A remotely accessible swarm robotics research testbed* [14], un proyecto desarrollado por Daniel Pickem, Paul Glotfelter, Li Wang, Mark Mote, Aaron Ames, Eric Feron, y Magnus Egerstedt pertenecientes a Georgia Institute of Technology.

El documento anteriormente mencionado describe al Robotarium, el objetivo principal del proyecto es el diseño y construcción de una plataforma de pruebas de bajo costo que permita el acceso remoto a grupos de investigación y estudiantes. Esta plataforma permite pasar fácilmente de la teoría a la simulación en las investigaciones de robótica de enjambre. A grandes rasgos el proyecto está formado por tres sistemas:

- Reconocimiento de imágenes.
- Comunicación entre el servidor y los robots.
- Comunicación entre el servidor a la nube para el acceso remoto.

Con estos tres sistemas un usuario puede conectarse de forma remota hacia el servidor para ejecutar un algoritmo, el servidor envía los comandos a los robots y la cámara de reconocimiento de imágenes se encarga de actualizar el estado y/o posición de los robots en la plataforma a medida que se ejecuta el algoritmo. Además de la plataforma, los investigadores del Robotarium diseñaron y construyeron sus propios robots móviles. Las características de estos robots son:

- Utiliza el módulo WiFi ESP8266 para la comunicación inalámbrica con el servidor.
- Está equipado con una batería LiPo de 400 mAh que permiten un tiempo de funcionamiento continuo de 40 minutos.
- Utiliza un microcontrolador Atmega 168.
- Tiene sensores de corriente para los motores y sensores de temperatura.
- Posee dos motores stepper, el robot puede moverse a una velocidad de máxima de 10 cm/s.

- Tiene un tamaño aproximado de 3 cm por lado.

El Robotarium respalda la viabilidad de la investigación realizada en este trabajo ya que la plataforma sigue funcionando en la actualidad, incluso algunos de los componentes utilizados en el Robotarium fueron utilizados para esta investigación como lo es el caso del módulo WiFi ESP8266.

5.2. ZOOIDS DE STANFORD

En octubre de 2016 se publicó el artículo *Zooids: Building Blocks for Swarm User Interfaces* [11], un proyecto desarrollado por Mathieu Le Goc, Lawrence H. Kim, Ali Parsaei, Jean-Daniel Fekete, Pierre Dragicevic y Sean Follmer pertenecientes a Inria, Stanford University, Université Paris-Sud y Université Paris-Saclay

El documento anteriormente mencionado describe a los Zooids, el objetivo principal del proyecto es el diseño y construcción de pequeños robots móviles que puedan ser utilizados en una interfaz Swarm que ellos también diseñaron y construyeron. Los robots tienen las siguientes características:

- 26 mm de diámetro y 21 mm de altura.
- Pesan alrededor de 12 g.
- Posee una batería LiPo de 100 mAh que permite un tiempo de trabajo de 1 hora.
- Tiene una velocidad máxima de 74 cm/s aunque por motivos de estabilidad se limita a 44 cm/s.
- Posee un sensor Touch para detectar al usuario.
- Utiliza un microcontrolador ARM TM32F051C8 de 48 MHz.
- Para la comunicación inalámbrica utiliza un módulo RF Nordic nRF24L01+.

La posición de los Zooids en la plataforma es comandado por un servidor, pero la detección de la posición de cada robot es determinado por un proyector que envía la información al servidor. Debido a que este sistema de detección no está basado en una cámara, no hay una latencia grande para el control de retroalimentación, lo cual permite que el control de posición sea más estable. El proyector utilizado es un DLP LightCrafter de Texas Instruments Inc. con una velocidad de cuadros por segundo de 3000 Hz. Se proyecta una secuencia de patrones codificados en gris sobre una superficie plana, luego los fotodiodos en el robot decodifican independientemente el código gris en una ubicación dentro del área proyectada, y envía su posición y orientación a la computadora maestra. Debido a la cantidad de patrones, la frecuencia de actualización de la posición es de aproximadamente 73 Hz.

Este proyecto respalda la viabilidad de la investigación realizada en este trabajo y demuestra que otros tipos de comunicación inalámbrica pueden ser funcionales para la comunicación en un enjambre de robots.

5.3. KILOBOTS DE HARVARD

En mayo de 2012 se publicó el artículo *Kilobot: A Low Cost Scalable Robot System for Collective Behaviors* [16], un proyecto desarrollado por Michael Rubenstein, Christian Ahler, y Radhika Nagpal pertenecientes a School of Engineering and Applied Sciences en Harvard University,

El documento anteriormente mencionado describe a los Kilobots, el objetivo principal del proyecto es el diseño y construcción de pequeños robots móviles que puedan interactuar entre sí para poder cumplir una tarea determinada en conjunto. Los robots tienen las siguientes características:

- Se mueve haciendo uso de dos motores de vibración y tres piernas de soporte.
- Utiliza una batería de ion de litio de 3.4 V y 160 mAh, esto permite que cada robot funcione de 3 a 24 horas dependiendo del nivel de actividad.
- Posee un infrarrojo transmisor/receptor que permite la carga de nuevos programas, la interacción entre los robots y la medición de distancia entre robots cercanos.
- Posee un sensor de luz para medir la intensidad de luz en el robot.
- Tiene un tamaño aproximado de 33 mm de diámetro y 34 mm de alto.

Otra característica muy importante en estos robots es que tiene un precio de $14, lo cual es muy económico en comparación con otras plataformas móviles para robótica de enjambre. Aunque el proyecto se centra principalmente en los robots, también se diseñó e implementó una plataforma de pruebas. En esta plataforma se cuenta con un equipo que emite la luz infrarroja a todos los robots al mismo tiempo cuando se necesita cargar una nueva configuración en los robots. Los robots reciben la nueva configuración y comienzan a interactuar entre ellos para cumplir con las especificaciones de la tarea. La plataforma también cuenta con una estación de carga, en la cual los robots se acercan y con un pin metálico que poseen comienzan a cargarse.

Este proyecto respalda la viabilidad de la investigación realizada en este trabajo y demuestra que una comunicación por medio de luz puede ser funcional para la comunicación en un enjambre de robots y que el procesamiento descentralizado puede funcionar en la manipulación de los robots.

5.4. REINGENIERÍA DE MEGAPROYECTOS FASE I

En octubre de 2017 se presentó la tésis de licenciatura *Reingeniería de Megaproyectos Fase I-Módulo de Swarm Robotics* [9], un proyecto desarrollado por los estudiantes de la Universidad del Valle de Guatemala, Erick Hernández, Johnny Guzmán, José Mérida, Otto Wantland, Vidal Villegas, William Orozco y Rony Ajtún.

El documento anteriormente mencionado describe el diseño y construcción de un robot modular destinado a trabajar en enjambre. Los robots tienen las siguientes características:

- Comunicación inalámbrica WiFi a través del módulo ESP8266.
- Uso del protocolo de comunicación TCP.
- Utiliza el microcontrolador Teensy LC para controlar todos los sensores y módulos en el robot.
- Utiliza dos motores DC con encoders y un driver DRV8833.
- Equipado con 6 sensores ultrasónicos y dos baterías LiPo de 2500 mAh.

Además del diseño y construcción de los robots, el trabajo también realiza la implementación de la comunicación WiFi. Para ello crearon una librería en Matlab que permitiera un uso fácil de la comunicación para cualquier persona con conocimientos básicos de programación en Matlab. Esta librería también está hecha para funcionar con varios robots en la red y que así puedan ejecutar algoritmos de control o de enjambre.

Este proyecto valida el uso del módulo ESP8266 como una interfaz de comunicación inalámbrica entre la computadora y los robots de la red, también demuestra que la comunicación por TCP es una mejor opción que UDP.

CAPÍTULO 6

Alcance

Este trabajo solamente se centra en el diseño e implementación de la red de comunicación inalámbrica para experimentación en robótica de enjambre, el cual puede complementarse con la investigación del estudiante André Rodas de la Universidad del Valle de Guatemala quién se encarga de hacer el procesamiento de imágenes para obtener la posición de los robots en la mesa de pruebas, pero este trabajo se encarga únicamente de la comunicación inalámbrica y no del procesamiento de imágenes.

El diseño y construcción de los robots no forma parte de este trabajo, pero si la configuración y programación del microcontrolador y del módulo inalámbrico ESP8266 para la comunicación WiFi. La comunicación WiFi entre los módulos y el servidor es de una sola vía, esto quiere decir que los módulos reciben los comandos del servidor pero los módulos no envían información del estado del robot al servidor, esto por que se busca que el microcontrolador no realice tanto procesamiento y que de esta forma los diseños futuros de robots sean simples. Si forma parte la medición de la velocidad máxima de transmisión y la selección de la velocidad mínima entre el módulo y el microcontrolador. Este trabajo tampoco se enfoca en el diseño de algoritmos anti-colisión de los robots y tampoco del entrelazamiento entre la IP y el ID del robot determinado por el control de posición.

CAPÍTULO 7

Metodología

El trabajo se realizó durante un período aproximado de ocho meses, la metodología utilizada en este trabajo se divide en cuatro partes de orden consecutivo, investigar y evaluar, diseñar, implementar y medir.

7.1. INVESTIGAR Y EVALUAR

Se busca que la mesa de pruebas sea de bajo costo y fácilmente escalable, por tal motivo estas dos características serán principalmente evaluadas en cada selección que se realice durante la investigación. Se utilizará una tabla comparativa que permita visualizar la mejor opción, este método se usa para seleccionar la tecnología y el módulo inalámbrico.

7.2. DISEÑAR

Luego de tener seleccionada la tecnología y el módulo inalámbrico se procederá a determinar los parámetros de diseño de la red de comunicación tomando en cuenta que se busca aprovechar al máximo la velocidad de transmisión permitida por la tecnología y el módulo inalámbrico. Para esto se investigará sobre cada uno de los parámetros que influyen en la velocidad de transmisión, algunos de los más importantes a tomar en cuenta son el tipo de modulación, el protocolo de transporte, la topología de la red, el modo de operación de los módulos inalámbricos, la seguridad de la red y el direccionamiento. Para cada una de estas características se seleccionará la que mejor se adapte a las necesidades de la mesa de pruebas.

7.3. IMPLEMENTAR

Teniendo los parámetros de diseño seleccionados se procederá a implementar la red de comunicación. Para esto se debe de seleccionar los equipos que participarán en el intercambio de información, así como también la cantidad de cada uno de ellos, estos equipos deben tomar el rol de cliente, servidor e intermediario. Estos serán seleccionados en base a la topología de la red y a los equipos disponibles en la Universidad del Valle de Guatemala. Siempre se debe tomar en cuenta que se busca que los equipos sean los más económicos posibles y fácilmente implementables.

7.4. MEDIR

Luego de seleccionar los equipos se procederá a medir la velocidad de transmisión de la red, para esto se realizará un enlace punto a punto entre dos computadoras de tal forma que todo el ancho de banda de la red sea dedicado solamente a esta comunicación. Se enviarán paquetes de distintos tamaños de un punto a otro durante cierto tiempo y con la ayuda de algún software de computadora se podrá medir la velocidad de transmisión máxima en la red.

Por último, se procede a determinar la velocidad de comunicación mínima entre el microcontrolador y el módulo inalámbrico. Para esto se realizará una conexión punto a punto entre el servidor y el módulo inalámbrico, luego se enviarán paquetes desde el servidor al módulo con el fin de que el receptor pueda emitir una señal de HIGH o LOW cada vez que reciba un paquete de información, con esto se espera obtener una señal cuadrada la cual será medida con un osciloscopio para obtener la frecuencia de la misma. Con la frecuencia de la señal se podrá determinar la velocidad de transmisión de paquetes en la red según la velocidad de comunicación entre el microcontrolador y el módulo. Con esta medición también se podrá determinar que tan cercano es la velocidad de transmisión con respecto al la velocidad de comunicación configurada, en base a esto también se podrá obtener la frecuencia de muestreo de cada robot según la velocidad de comunicación configurada y la cantidad de robots en la red.

Tabla 7.1: Cronograma de actividades.

Mes	Día	Descripción
Febrero	9	Se realizará una investigación sobre las distintas tecnologías de comunicación inalámbrica y las características de cada una.
	16	
	23	Evaluar y seleccionar la tecnología de comunicación a utilizar en la mesa de pruebas.
Marzo	2	Se realizará una investigación sobre los distintos módulos inalámbricos y las características de cada uno.
	9	
	16	Evaluar y seleccionar el módulo de comunicación a utilizar en la mesa de pruebas.
	23	Investigar los parámetros de diseño de la red de comunicación.
	30	
Abril	6	Seleccionar los parámetros de diseño de la red buscando obtener la mayor velocidad de transmisión posible.
	13	
	20	Diseñar la red de comunicación, seleccionando los equipos a utilizar, la configuración de cada uno, los protocolos, etc.
	27	
Mayo	4	Configuración del módulo inalámbrico.
	11	
	18	Diseño del circuito para la comunicación entre el módulo y el microcontrolador.
	25	
Junio	1	Programación en C para la comunicación entre el módulo y el microcontrolador.
	8	
	15	
	22	Implementación de la red de comunicación inalámbrica.
	29	
Julio	6	
	13	Programación en C++ de los protocolos de comunicación entre el servidor y los módulos inalámbricos.
	20	
	27	
Agosto	3	
	10	
	17	Medición de la velocidad de transmisión obtenida en la red.
	24	
	31	Determinar la velocidad máxima de transmisión en la red.
Septiembre	7	Determinar la velocidad de comunicación mínima entre el módulo y el microcontrolador buscando que se aproveche al máximo el ancho de banda de la red.
	14	
	21	

CAPÍTULO 8

Elección de tecnología y módulo inalámbrico

8.1. Comparación de tecnologías

Tabla 8.1: Tabla comparativa de tecnologías inalámbricas

Característica	**Bluetooth**	**WiFi**	**LTE**
Cantidad de nodos por red	7	255 con direcciones clase C	>200 con celdas de 5 MHz
Velocidad de transmisión	1-24 Mbps	11-100 Mbps	20-100 Mbps
Costo de implementación	Bajo	Bajo	Alto
Ancho de banda	2.4 GHz	2.4 GHz o 5 GHz	Bandas licenciadas
Uso	Mouse, teclados auriculares	Redes de computadora	Telefonía móvil

[Elaboración propia]

En la tabla 8.1 se observa que LTE no es posible implementarla en la red de la mesa de pruebas debido a que representa un gasto elevado por la compra de datos a las compañías de telefonía en el país. Bluetooth tampoco es posible implementarla debido a su limitado números de usuarios y su lenta velocidad de transmisión. Por lo tanto, la tecnología más óptima es WiFi la cual me permite tener una gran cantidad de usuarios en la red, alta velocidad de transmisión y por ende baja latencia para evitar las colisiones de los robots.

8.2. Módulo inalámbrico

El módulo inalámbrico fue seleccionado de la misma forma que la tecnología inalámbrica, en la tabla 8.2 se observan las características de tres módulos distintos. Se observa que tienen características similares, pero las características más importantes de la red de comunicación son el bajo costo y la alta velocidad de transmisión, centrandose en estas dos características se hace evidente que el módulo ESP8266 es muy superior en cuanto a costo y velocidad de transmisión. Por lo cual, el módulo inalámbrico a utilizar es el ESP8266. Se seleccionó no solo por las características descritas en la tabla 8.2 sino que también porque es utilizado en el trabajo realizado por [14] y [9], además, es uno de los módulos más utilizados por IoT y económicos en la actualidad.

Tabla 8.2: Módulos WiFi

Característica	**ESP8266**	**RN-XV WiFly**	**XBee WiFi S6B**
Ancho de banda	2.4 GHz	2.4 GHz	2.4 GHz
Costo	$3.25	$31.73	$29.00
Cantidad de nodos	Depende de la IP	Depende de la IP	Depende de la IP
Dimensiones	2.48x1.43 cm	3.25x2.5 cm	4.5x2.5 cm
Protocolo 802.11	b/g/n	b/g	b/g/n
Consumo	145 mA	38 mA	309 mA
Velocidad de transmisión máxima	54 Mbps	464 kbps	72 Mbps

[Elaboración propia]

CAPÍTULO 9

Diseño de red WiFi

9.1. Actualización de firmware ESP8266

Debido a que el módulo ESP8266 viene configurado con cierta versión de firmware es necesario realizar una actualización de este para tener un mayor número de funciones disponibles, esto nos permite tener mayor control sobre la configuración de los módulos WiFi. Para realizar esta actualización se deben seguir los siguientes pasos, los enlaces a los programas se encuentran en el anexo A.

1. **Descargar NONOS SDK:** este es el firmware que se carga al módulo, en la fecha que se realizó esta investigación la versión más reciente es la 2.2.0.
2. **Descargar FLASH DOWNLOAD TOOLS:** es el programa oficial de espressif utilizado para descargar el firmware a las plataformas ESP.
3. **Memoria Flash:** las plataformas ESP utilizan diferentes modelos de memorias flash, por tal motivo antes de realizar la actualización del firmware se debe obtener el modelo de la memoria flash, buscar el datasheet y observar las especificaciones de capacidad, velocidad y modos de SPI. El modelo de la memoria flash se puede obtener directamente de los números escritos en el chip del PCB. En este caso la memoria flash del ESP8266 es la PN25F08, la cual es de 8 Mbits, trabaja con Standard, Dual o Quad SPI y puede trabajar con frecuencia máxima en SPI de 108 MHz.
4. **Direcciones de memoria:** dentro de la carpeta NONOS SDK abrir el archivo README.md (abrir con bloc de notas o algún otro editor de texto) ubicado en \bin\at. Este archivo proporciona la información respectiva sobre las direcciones de memoria en la que se debe guardar cada uno de los archivos necesarios para actualizar el firmware según la capacidad de la memoria flash. La información de las direcciones para una memoria flash de 8 Mbits con una configuración 512+512 en README.md se puede ver de la siguiente forma:

```
# BOOT MODE
## download
```

```
### Flash size 8Mbit: 512KB+512KB
    boot_v1.2+.bin              0x00000
    user1.1024.new.2.bin        0x01000
    esp_init_data_default.bin   0xfc000 (optional)
    blank.bin                   0x7e000 & 0xfe000
```

5. **Cargar archivos:** abrir el programa del paso 2 y cargar los 4 archivos siguientes, en donde "x" indica la versión del SDK o la versión del archivo correspondiente y de preferencia siempre utilizar el más reciente:

   - boot_vx.bin ubicado en \bin.
   - user1.1024.new.2.bin ubicado en \bin\at\512+512 o \bin\at\1024+1024, según la configuración de la memoria. Para conocer la configuración de la memoria se puede utilizar el comando AT+GMR. En este caso la configuración es 512+512 como se observa en la figura 9.1.
   - esp_init_data_default_vx.bin ubicado en \bin.
   - blank.bin ubicado en \bin.

Figura 9.1: Resultado del comando AT+GMR para conocer la configuración de la memoria.

```
 ets Jan  8 2013,rst cause:2, boot mode:(3,6)

load 0x40100000, len 1856, room 16
tail 0
chksum 0x63
load 0x3ffe8000, len 776, room 8
tail 0
chksum 0x02
load 0x3ffe8310, len 552, room 8
tail 0
chksum 0x79
csum 0x79

2nd boot version : 1.5
  SPI Speed      : 40MHz
  SPI Mode       : DIO
  SPI Flash Size & Map: 8Mbit(512KB+512KB)
jump to run user1 @ 1000
```

[Elaboración propia]

Después de la carga de archivos se debe colocar en el lado derecho las direcciones de memoria a cada archivo según README.md y luego habilitar las casillas de la izquierda.

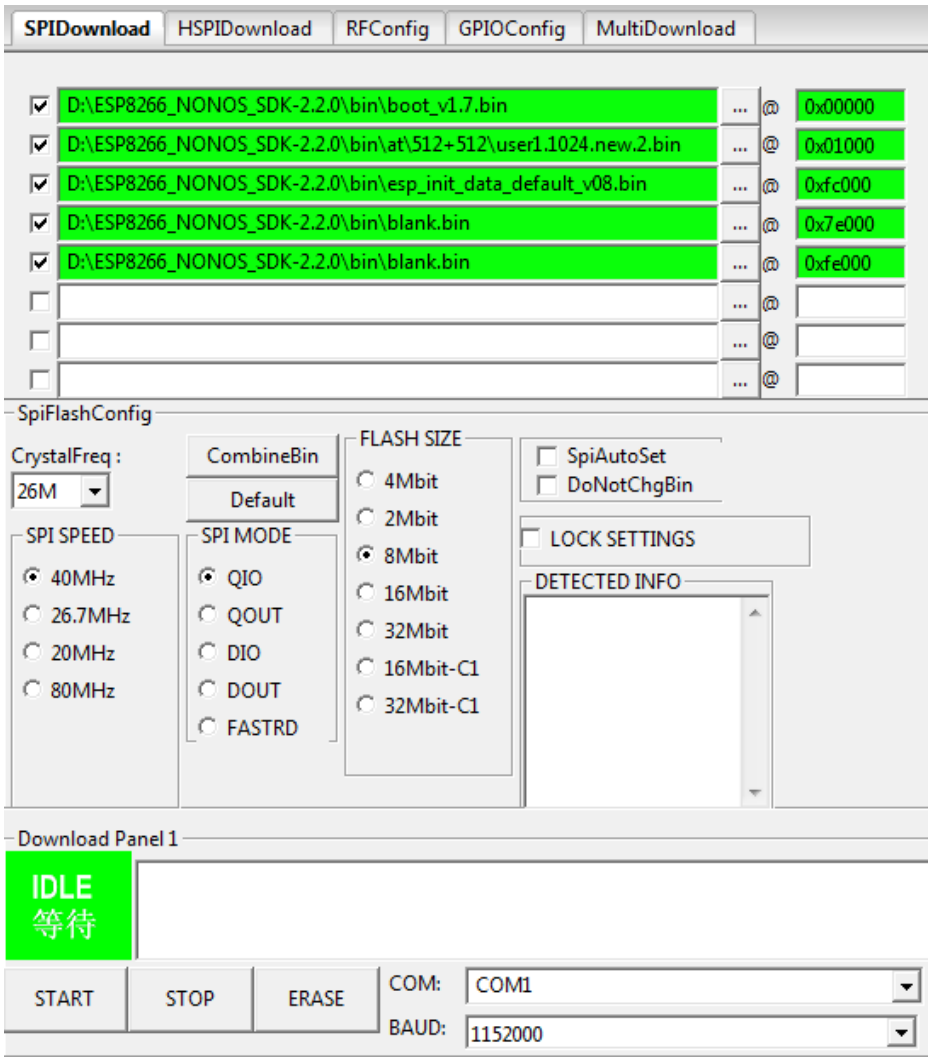
6. **Colocar información del módulo:** en la parte inferior del programa se debe colocar la frecuencia del cristal del módulo que tengamos (esto se observa en la placa), el modo SPI, la velocidad y la capacidad de la memoria Flash. En este caso el módulo tiene un cristal de 26 MHz, la memoria es de 8 Mbits, el SPI puede trabajar a 40 MHz y en modo QIO.

7. **Descarga de firmware:** cabe destacar que para hacer este proceso se debe conectar el módulo ESP8266 con la computadora a través de un FTDI, esta conexión se realiza por el puerto serial. Luego en el programa para descargar el firmware se debe selecciona el puerto COM y la velocidad de transmisión, que por defecto es 115200. Se coloca el módulo en modo escritura, esto se hace colocando GPIO0 a LOW y luego aplicar un pequeño pulso de LOW al pin RST, el pin RST se puede quedar al aire luego de esta operación. Por último, en el programa se debe presionar START y esperar a que termine la actualización.

8. **Comprobación de firmware:** luego de terminar la actualización se debe desconectar el GPIO0 de LOW y aplicar otro pulso de LOW a RST, estos dos pines pueden quedarse al aire luego de realizar la operación. Se puede hacer uso de los comandos AT+RST y AT+GMR a través de una terminal en la computadora para verificar que el firmware se actualizó a la versión correcta.

Esta actualización de firmware permite desactivar el echo del puerto serial para evitar saturarlo, modificar el baudrate, configurar los mensajes de respuesta del sistema del módulo, establecer la potencia de transmisión, poner en modo sleep para ahorro de batería, entre otros.

Figura 9.2: Configuración en Flash Download Tools para descargar el nuevo firmware al ESP8266.



[Elaboración propia]

9.2. Parámetros de diseño

Para aprovechar al máximo la capacidad y velocidad de la red se seleccionaron ciertos parámetros de diseño, se tomaron en cuenta aspectos como la reducción de interferencia, el aumento de velocidad,

escalabilidad para agregar nuevos nodos, confiabilidad en la entrega de paquetes y la seguridad de la red.

9.2.1. Modo y topología de red

Debido a que se necesita que los módulos tengan comunicación directa con el servidor se seleccionó el modo infraestructura y la topología en estrella, en este modo tanto el servidor como los módulos funcionan como clientes y el dispositivo centralizador es un Router Linksys WRT54GL. Debido a esto también se seleccionó un direccionamiento de clase C, 192.168.1.0/24, con la cual se puede llegar a obtener 254 clientes conectados a la red WiFi.

No se utilizó un modo ad hoc debido a que el rendimiento de la red se vería afectado al aumentar la cantidad de nodos, también se descartó porque los equipos disponibles para trabajar solamente aceptaban redes ad hoc con el protocolo 802.11b y con el cual se obtiene una velocidad de transmisión menor que con 802.11g.

9.2.2. Protocolo 802.11

El módulo ESP8266 permite trabajar en los protocolos 802.11b/g/n, con la versión "n"de este protocolo se puede obtener una velocidad de hasta 65 Mbps, pero debido a que el router utilizado solo puede trabajar con la versión "g", se determinó que el protocolo utilizado sería el 802.11g obteniendo una velocidad teórica de 54 Mbps. Cabe destacar que según el datasheet el módulo puede alcanzar una velocidad máxima de transmisión de 54 Mbps, con lo cual logra deducirse que aunque se seleccione la versión "n"la velocidad está limitada por el fabricante del módulo, aunque esto no es algo que se pueda comprobar ya que no se tienen un router disponible que trabaje en la versión "n".

9.2.3. Canal

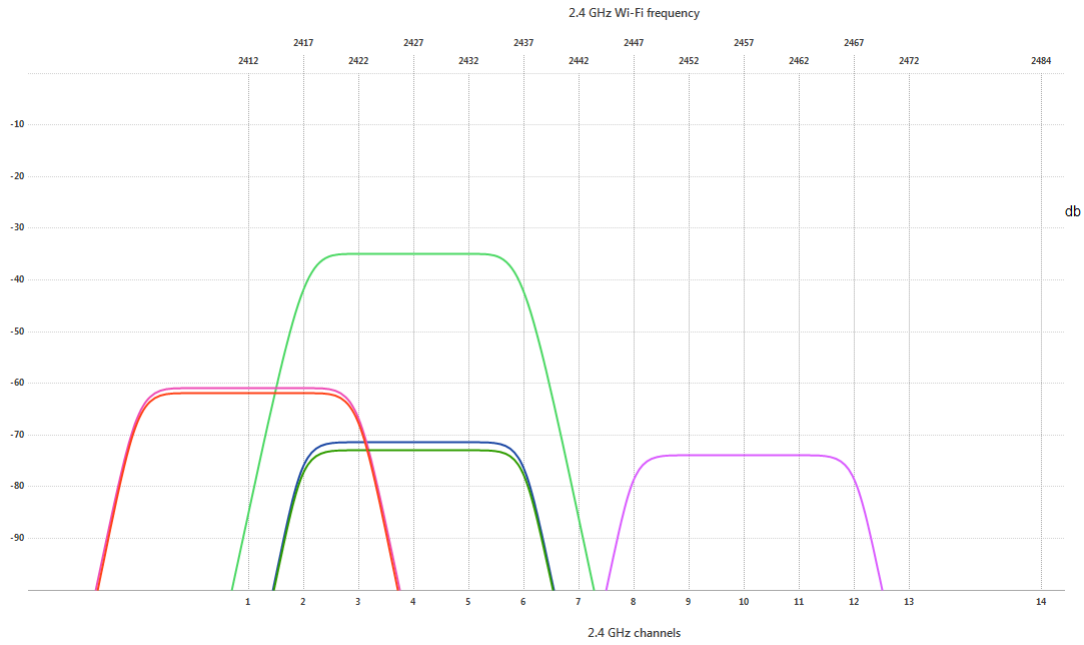
El ESP8266 puede trabajar en los 14 canales definidos por la IEEE 802.11b/g/n, pero en el caso de Guatemala se encuentran disponibles solamente los primeros 13 canales. Para seleccionar los canales de comunicación óptimos se tomó como referencia el salón J313 de la Universidad del Valle de Guatemala, se utilizó un analizador de red WiFi para obtener un mapa del espectro y los canales utilizados en el área. El analizador de red utilizado es NetSpot[1] con licencia gratuita, en la figura 9.3 se puede observar los canales y las redes WiFi presentes en el entorno.

Se observa que el canal 11 es el más óptimo a utilizar debido a que solamente una red del entorno interfiere con este canal, en el resto de canales hay por lo menos dos redes. También se observa que la red en el canal 11 tiene muy poca potencia, esto ayuda a reducir la interferencia en la red de la mesa de pruebas. Por lo tanto, el canal a utilizar es el 11. La configuración del canal a utilizar solamente se realiza en el router, pero los dispositivos a conectar deben poder trabajar en el canal deseado. Si en algún futuro se cambia de lugar la plataforma de pruebas o hay nuevas redes en el entorno, debe realizarse nuevamente este análisis para seleccionar el canal óptimo y así cambiar la configuración del router.

9.2.4. Seguridad de la red

La seguridad no es algo de gran importante en la red de la mesa de pruebas debido a que si alguien escucha los paquetes que envío no podrán entenderlos ya que estos solamente contienen

Figura 9.3: Análisis de redes activas en los distintos canales del entorno con NetSpot.



[Elaboración propia]

comandos para los robots. Pero si necesito un pequeño grado de seguridad ya que tampoco quiero que cualquier dispositivo se conecte y envíe paquetes causando demasiado tráfico y reduciendo el rendimiento de la red. Por estas características se seleccionó la seguridad WEP, la cual me ofrece una seguridad mínima y no congestiona mi red con claves PSK como lo hace WPA/WPA2.

9.2.5. Protocolo de transporte

En la tabla 9.1 se puede observar una comparación entre los protocolos de transporte TCP y UDP, vemos que TCP es el más seguro de los dos, pero UDP puede que llegue a obtener una pequeña mejora en la velocidad de transmisión debido a su simplicidad. Las características que necesito en la red son las siguientes:

- **Confiabilidad en la entrega de paquetes:** se necesita que los comandos sean entregados a los robots para responder anticipadamente a cualquier colisión que pueda producirse.
- **Control de flujo:** los datos se envían del servidor al módulo y luego del módulo al microcontrolador, este último se encarga de procesar los datos y tomar acciones. El ESP8266 y el microcontrolador se comunican por un puerto serial, esto limita la velocidad a la que se pueden procesar los datos ya que la velocidad de transmisión en la red es muy superior a la velocidad de comunicación serial. Por este motivo se necesita un control de flujo que pueda adaptar la velocidad de transmisión del servidor a la velocidad de recepción por puerto serial del microcontrolador.

Debido a las características anteriores se seleccionó el protocolo TCP, este me ofrece confiabilidad en la entrega de paquetes y control de flujo en la transmisión.

Tabla 9.1: Protocolos de transporte.

TCP	**UDP**
Asegura la recepción de datos	No asegura la recepción de datos
Establece conexión	No establece conexión
Tiene control de flujo y confirmación	No hay control de flujo ni confirmación

[Elaboración propia]

CAPÍTULO 10

Implementación de red WiFi

10.1. Selección de equipos

Antes de comenzar con la implementación de la red se deben seleccionar los equipos que deberán participar en la misma, los dispositivos seleccionados se observan en la Figura 10.1 pero a continuación se procede a explicar cada uno con más detalle. Ya se conoce que se utilizaran como clientes los módulos ESP8266, pero también se necesita de un dispositivo que procese los datos recibidos o enviados en la red, un servidor que se encargue de administrar la comunicación y un dispositivo intermediario que permite la conexión entre el servidor y los clientes.

10.1.1. Servidor

Como servidor se utiliza una computadora de uso común, el único requisito es que tenga instalado Visual Studio con C++ para poder ejecutar el algoritmo para la conexión TCP, en este caso se utilizó Visual Studio Community 2015. Este dispositivo es un servidor TCP, este se encarga de abrir el canal de comunicación y aceptar las solicitudes de comunicación de los módulos WiFi. Este dispositivo debe tener una dirección IP estática (en este caso es la 192.168.1.2) ya que el microcontrolador que se comunica con el módulo ya tendrá establecido que debe conectarse a dicha dirección, por otro lado, la dirección IP de cada módulo se asigna por DHCP. Si la dirección IP del servidor se establece por DHCP entonces existe la posibilidad de que esta cambie y que los módulos no puedan conectarse al servidor.

10.1.2. Dispositivo de procesamiento

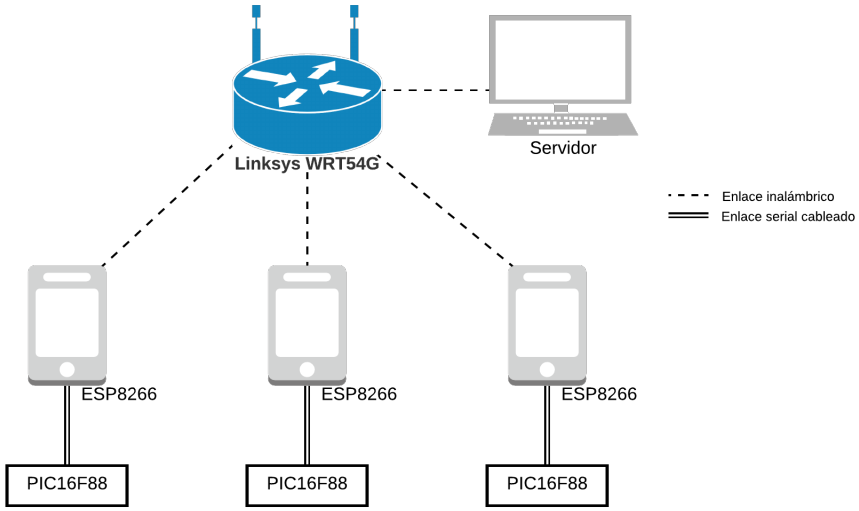
Los ESP8266 solamente se encargan de recibir los paquetes y enviar la información por el puerto serial, en el envío ocurre lo mismo solo que de forma contraria. Por este motivo, se necesita de un dispositivo que sea capaz de procesar estos paquetes, para tales fines se utilizó un PIC16F88. Los motivos por los cuales se seleccionó este dispositivo son los siguientes:

- Fácil de programar.
- Económico.
- Disponible de forma inmediata en la Universidad.
- No se necesita de muchas prestaciones para la comunicación con el módulo WiFi, el único requisito es que tenga puerto serial.

10.1.3. Dispositivo intermediario

Como dispositivo intermediario se utilizó un Router Linksys WRT54GL, se utilizó principalmente porque ya se tenía disponible en la Universidad y no era necesario recurrir a gastos en la compra de uno nuevo. Las características principales de este dispositivo son que permite una velocidad de transmisión máxima de 54 Mbps y trabaja bajo los protocolos IEEE 802.11 b/g. Estas características son suficientes para poder implementar la red de comunicación con los módulos ESP8266.

Figura 10.1: Red de comunicación WiFi para la mesa de pruebas.



[Elaboración propia]

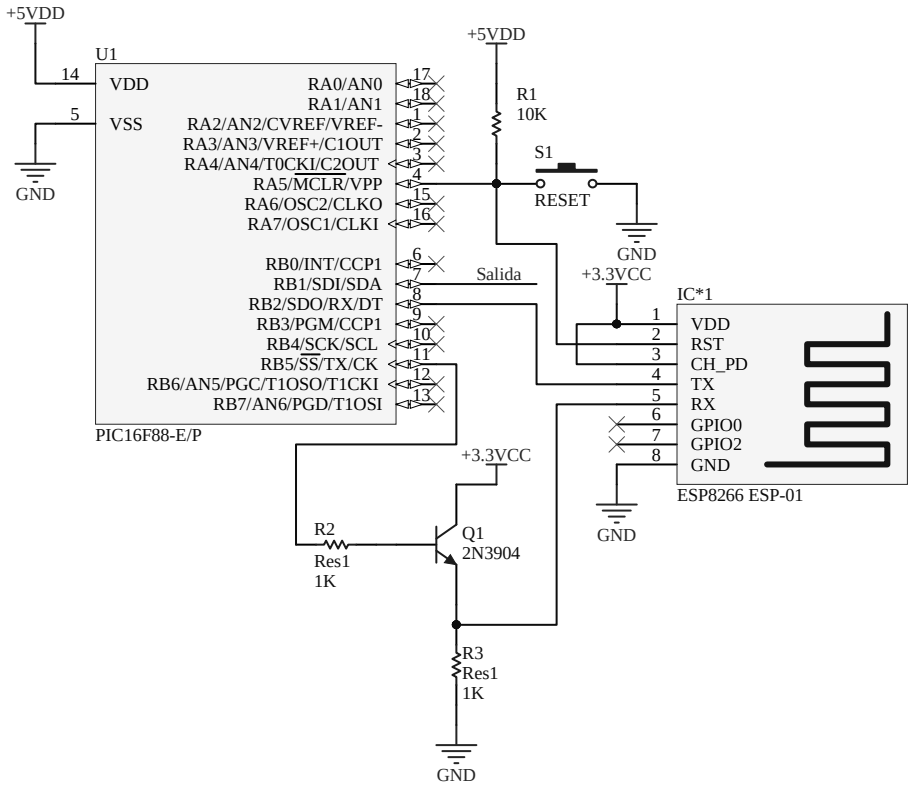
10.2. Programación de PIC16F88

El PIC16F88 es el encargado de procesar todos los paquetes recibidos o enviados a través del ESP8266, como se mencionó en la sección anterior se utilizó este PIC debido a su simplicidad de programación y disponibilidad. Este PIC puede trabajar con oscilador externo o interno, pero con fines de ahorro de espacio se utilizó un oscilador interno de 8 MHz. Se utilizó el puerto serial del PIC para comunicarse con el módulo, en este caso se definió un baudrate de 250K debido a que con la frecuencia de reloj seleccionada se lograba obtener un error de 0 %.

Uno de los mayores inconvenientes al momento de la comunicación entre el PIC y el módulo es que el primero utiliza 5 VDC en su alimentación mientras que el segundo trabaja a 3.3 VDC, entonces existe el riesgo que los pines de salida del PIC puedan dañar los pines de entrada del módulo

debido a que este no es capaz de soportar los 5 V en su pines de entrada. En el caso de los pines de salida del módulo no hay inconvenientes ya que el PIC es capaz de recibir los 3.3 V. Por esta razón, se realizó un acople con un transistor 2N3904, de esta forma se logra adaptar los 5 V del PIC a los 3.3 V que recibe el módulo. El boton de RESET se utiliza para el PIC y el módulo, el pin de Salida es utilizado para generar una señal cuadrada cuando se realizan las distintas pruebas de recepción de paquetes. También se debe tomar en cuenta que el módulo es sensible al ruido en su alimentación, para todas las pruebas realizadas en está investigación se utilizó como fuente una batería que fuera capaz de proporcionar los 5V y para la alimentación en el ESP8266 se utilizó un regulador LM1117 de 3.3V.

Figura 10.2: Diagrama de conexión entre PIC16F88 y módulo ESP8266.



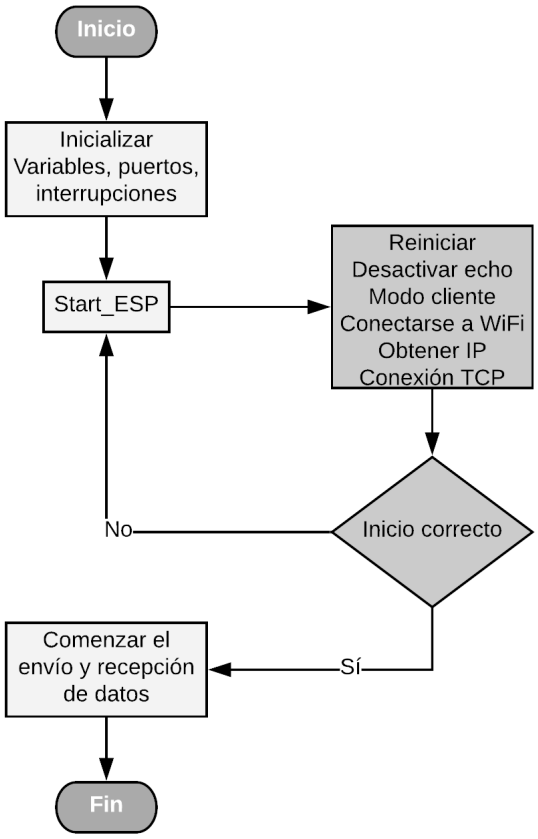
[Elaboración propia]

En la Figura 10.3 se logra observar el proceso utilizado para la inicialización de la comunicación, el primer paso es el mismo que en cualquier microcontrolador ya que consiste solamente en inicializar todo lo necesario en el PIC. En este caso se configura un oscilador interno a 8 MHz, un puerto digital de salida en RB1, el puerto de comunicación serial a 250K baud y la habilitación de la interrupción en la recepción de datos por puerto serial.

En el segundo paso se utiliza una función llamada Start_ESP en la cual se realiza todo el proceso necesario para que el módulo pueda conectarse a la red WiFi. Estos comandos se muestran en la tabla 10.1 y son enviados por puerto serial. El módulo siempre envía una respuesta para saber si el comando se realizó correctamente, por esta razón, en cada comando que se envía el pic espera la respuesta de que el comando fue recibido y realizado correctamente. Si por algún motivo la respuesta recibida no es la indicada en la tabla 10.1, entonces se aplica un RESET para que el PIC vuelva a enviar los comandos desde el principio.

Luego de realizar la conexión WiFi se procede a realizar la conexión por TCP. Si todos los pasos

Figura 10.3: Diagrama de flujo de la inicialización entre el PIC16F88 y el módulo ESP8266



[Elaboración propia]

se realizan correctamente, entonces comienza el proceso de recepción y envío de datos, pero si este no es el caso significa que hubo un problema ya sea en la conexión a la red WiFi o en la conexión TCP. El programa utilizado para el PIC puede descargarse del anexo A

Algo importante es que mientras se lleva a cabo la función Start_ESP están desactivadas las interrupciones, esto permite que después de cada comando enviado el pic pueda quedarse esperando la respuesta del módulo, de esta forma nos aseguramos que no se va a realizar ninguna otra tarea mientras se inicializa el módulo WiFi. Luego de que fue exitosa la inicialización se activan nuevamente las interrupciones para comenzar a recibir los paquetes en la red.

10.2.1. Recepción de información.

Para recibir la información del ESP8266 se configura en el PIC la interrupción de RX en el puerto serial, la bandera relacionada a esta interrupción es RCIF. esta se habilita cuando se termina de escribir en el registro RCREG una palabra de 8 bits proveniente del puerto RX y se deshabilita cuando se lee el registro.

En la función de interrupción se verifica si la información que se está recibiendo es de un comando válido, los comandos tienen un ancho de 4 bytes. Para esto se utiliza el carácter '-'como inicio del

comando, sigue el número de comando, el valor del comando y el carácter de final '\0'. Entonces un comando válido completo sería '-La\0', sin las comillas. Si se detecta el carácter de inicio de un comando válido entonces los siguientes caracteres recibidos se almacenan en un vector de chars hasta recibir el carácter de final, pero si el número de caracteres recibidos desde el inicio es mayor a 4 entonces se desecha todo lo recibido anteriormente porque indica que no es un comando válido para procesar. Cuando se termina de recibir un comando válido se habilita una bandera, en el ciclo principal se comprueba constantemente el valor de esta bandera para determinar si ya se puede procesar la información, si la bandera está habilitada entonces se procede a realizar una acción según el comando.

Tabla 10.1: Comandos AT del pic al módulo WiFi.

Comando	**Descripción**	**Respuesta**
AT+RST	Reinicia el módulo	ready
ATE0	Desactiva el echo del puerto serial	OK
AT+CWMODE_CUR=1	Activa el modo cliente	OK
AT+CWJAP_CUR="Megaproyecto", "ABC1234567"	Se conecta a la red WiFi	OK
AT+CIPSTART="TCP", "192.168.1.2",54000	Establece la conexión TCP	OK

[Elaboración propia]

10.3. Programación de Sockets en C++

En un proceso de comunicación TCP existen dos diferentes papeles que se pueden asumir, el primero es como servidor TCP y el segundo como cliente. En este caso los clientes TCP serán los módulos WiFi y el servidor TCP será creado en una computadora común desde C++. El proceso para que el servidor y el cliente puedan comunicarse consiste en tres simples pasos:

1. Iniciar servidor TCP.
2. El cliente hace la solicitud de conexión según la IP y el puerto habilitados por el servidor.
3. El servidor decide si acepta o no la solicitud de conexión del cliente.

El proceso para inicializar el servidor TCP se observa en la Figura 10.4. WinSock es la librería utilizada para el manejo de sockets en Windows dentro de la plataforma C++, esta contiene todas las funciones necesarias para crear y manejar los sockets. Luego de incluir esta librería se comienza a crear el socket, para crear el socket debe indicarse la familia de protocolos y de direcciones que serán utilizadas, en este caso se utilizo el protocolo TCP y la familia de direcciones IPv4. Luego de crear el socket se realiza un enlazamiento entre el socket y la dirección IP y el puerto, esto se utiliza para indicar que podrá accederse al socket de comunicación solamente por la dirección IP y el puerto seleccionados. En este caso se utilizó la IP 192.168.1.2 y el puerto 5400.

Por último, se procede a indicar al socket que comience a escuchar las solicitudes de los clientes, entonces cuando el servidor recibe la solicitud este evalúa si acepta o no la conexión. Si la conexión es aceptada entonces comienza la comunicación en ambas vías, si no, entonces puede ser que el cliente tenga la IP o puerto erróneos determinados por el servidor.

10.3.1. Sockets Asíncronos

Un problema encontrado al momento de programar los sockets en C++ era que la función accept(), encargada de aceptar las solicitudes de los clientes, era una función de bloqueo. Esto significa que al llamar esta función el programa queda detenido a la espera de una solicitud de conexión de algún cliente, esto evita que el sistema de comunicación inalámbrica sea flexible. Por ejemplo, supongamos que al inicio de un programa aceptamos todas las conexiones entrantes y que después de un tiempo un nuevo nodo necesita agregarse al sistema, ¿Cómo hacemos para aceptar la nueva conexión?. Existen dos posibilidades:

1. Si conocemos el momento exacto en el que un nuevo nodo se conectará al sistema podemos incluir la función accept() en nuestro programa, pero conocer el momento exacto es muy poco probable y conllevaría a que el programa no realice ninguna otra tarea hasta esperar la conexión.
2. Utilizar sockets asíncronos que permitan utilizar la función accept() solamente cuando se realice una solicitud de conexión.

Los sockets asíncronos funcionan de forma muy similar a las interrupciones en un microcontrolador. En este caso el encargado de estas interrupciones en Windows se conoce como manejador de ventanas, un manejador de ventanas se encarga de escuchar todos los eventos que ocurren en una ventana determinada. Cuando ocurre un evento en la ventana, el manejador llama a una función en la cual se realiza una comprobación de las banderas y se realiza una acción según la bandera activada. Una característica especial del manejador de ventanas es que acumula los eventos pero no los procesa hasta que el evento anterior sea procesado.

Dentro de la función que llama el manejador de ventanas se verifica si la bandera FD_ACCEPT fue activada, si es el caso entonces se utiliza la función accept() para que acepte la conexión entrante. Cuando se acepta un conexión se guardan tres valores distintos en tres vectores, el primer vector (arregloSockets[]) almacena los descriptores de tipo socket que retorna la función accept(), el segundo vector (direccionesIP[]) almacena la dirección IP de cada cliente y el tercer vector (clientes[]) almacena la estructura de información de cada cliente conectado. De esta forma podemos aceptar una conexión en cualquier momento y gracias al manejador de ventanas si muchas conexiones se solicitan al mismo tiempo este se encarga de ir procesando una por una. El uso de sockets asíncronos nos permite tener flexibilidad en la adición y eliminación de nuevos nodos al sistema en funcionamiento.

Cuando se envía paquetes a un cliente en específico la función sendto() recibe dos tipos de descriptores, el primero es el que crea el servidor (existe solamente 1 sin importar la cantidad de conexiones) para aceptar las conexiones y el segundo es el que se crea para cada cliente conectado (se crea uno para cada conexión establecida). Cuando se necesita enviar un paquete a un cliente se busca la dirección IP en el vector correspondiente, cuando encuentra la dirección IP se extrae la posición en el vector y con esta información modifica el descriptor del servidor para que la dirección IP y el puerto correspondan a los del cliente, por último, utiliza la función sendto() y le pasa la información del descriptor del cliente y del servidor ya con la dirección IP y el puerto modificados.

Para la implementación de los sockets en la red de comunicación de este proyecto se creó una librería propia en la cual se hace uso de las funciones del API de sockets para Windows de 32 bits en C++. El objetivo principal de esta librería es dar flexibilidad a los investigadores que hagan uso de esta red, de esta forma solamente necesitan incluir la librería en sus algoritmos y hacer uso de las funciones para comunicarse con los robots en la red. La librería se llama SocketTCP.h, sus funciones son las descritas en la tabla 10.2 y puede descargarse del enlace en el anexo A.

Tabla 10.2: Funciones de la librería SocketTCP.h.

Función	**Tipo**	**Descripción**
IniciarSocket(int puerto)	void	Inicializa el socket en el puerto determinado y la interfaz WiFi de la computadora, habilita los sockets asíncronos y crea el manejador de ventanas que permite procesar los eventos de los sockets asíncronos.
EnviarTCP(char *mensaje, int largo, int pos)	bool	Envía el mensaje a un determinado agente según la posición del arreglo en el que se encuentre, si el robot está conectado entonces envía el mensaje y devuelve TRUE, si no, entonces devuelve FALSE y por lo tanto no se envía el mensaje.
CerrarSocket(void)	void	Cierra de forma segura el socket abierto al principio de la conexión, este comando se utiliza cuando el algoritmo termina.

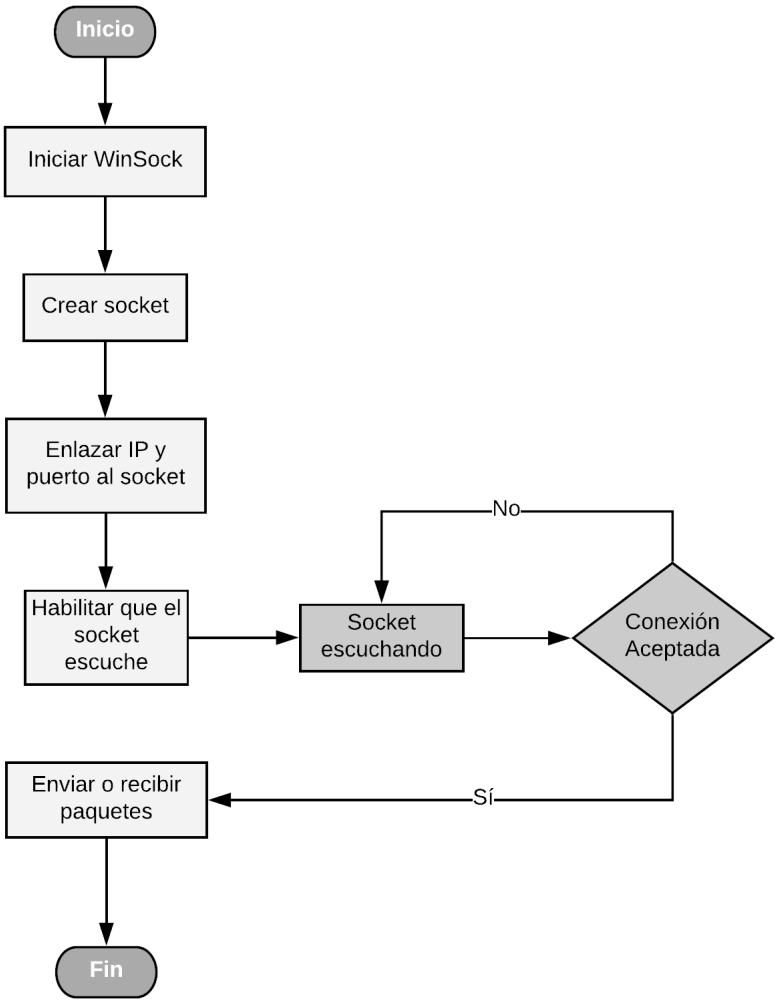
[Elaboración propia]

Tabla 10.3: Funciones para la comunicación entre el PIC y el módulo WiFi.

Función	**Tipo**	**Descripción**
Start_UART(void)	void	Se encarga de realizar toda la configuración para el puerto UART del PIC
SendChar_UART(char c)	void	Envía el valor de c por el puerto serial, sale de la función hasta que termina de enviar el dato de 8 bits.
SendString_UART(char *c)	void	Utiliza la función SendChar_UART() para enviar el contenido al que apunta c, sale de la función hasta que encuentra el valor \n que indica el final de la cadena.
ReceiveChar_UART(void)	char	Recibe un carácter por el puerto serial, se mantiene en esta función hasta que termine de recibir el valor completo de 8 bits. Luego retorna el dato recibido.
ReceiveString_UART(char *c)	void	Utiliza la función ReceiveChar_UART(void) para leer el dato por puerto serial y lo almacena en la dirección a la que apunta c. Deja de almacenar los datos cuando encuentra el valor \n que indica el final de una respuesta del módulo o si sobrepasa un valor de 8 caracteres que ya no es una respuesta valida del módulo. Al final agrega el valor \0 que es necesario para hacer comparaciones con cadenas en el programa principal.
Start_ESP(void)	void	Envía los comandos AT al módulo WiFi para realizar la configuración inicial y conectarse a la red y al servidor.
WaitResponse_ESP(char *c)	void	Espera una respuesta válida del módulo y lo compara con el dato al cual apunta c, el cuál corresponde a la respuesta que se espera recibir. Esta es una función de bloqueo, eso quiere decir que no sale de la función hasta que reciba una respuesta correcta. Cuando esto ocurre simplemente debe enviarse un reset al PIC para que vuelva a enviar los comandos desde el principio.

[Elaboración propia]

Figura 10.4: Diagrama de flujo de la inicialización del servidor TCP en C++



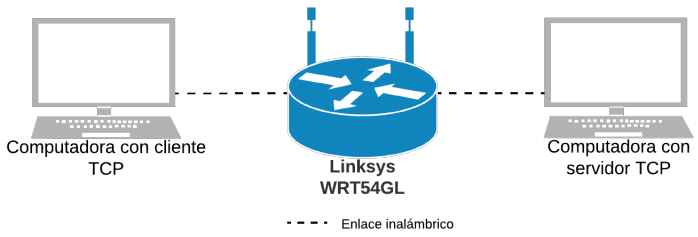
[Elaboración propia]

CAPÍTULO 11

Medición de la velocidad de transmisión y comunicación

11.1. Medición de la velocidad de transmisión máxima en la red

Figura 11.1: Diagrama de conexión para la medición de velocidad en la red.



[Elaboración propia]

Para la medición de la velocidad de transmisión en la red no se utilizó el módulo ESP8266 ni el PIC, esto porque estos dispositivos tienen un procesamiento de datos menor a la velocidad teórica de transmisión que se puede tener y lo cual evitaría obtener un resultado confiable. Para medir esta velocidad se utilizó el diagrama de la figura 11.1 y la herramienta PassMark PerformanceTest[2] 9.0 en su versión de prueba gratuita. Esta herramienta permite correr un benchmark en el cual se mide la velocidad de transmisión de datos neta en la red con TCP o UDP. Es necesario hacer énfasis en que está herramienta mide los datos y no todos los bits enviados, ya que como bien se sabe, cuando se envían datos por la red estos son encapsulados por cada una de las capas del modelo OSI, esto significa que se agregan más bits a los datos para poder ser transportados en la red.

La velocidad teórica de transmisión que se puede tener en la red está limita por el router, esto quiere decir que aunque se tengan dispositivos que transmitan a 100 Mbps la velocidad máxima sera

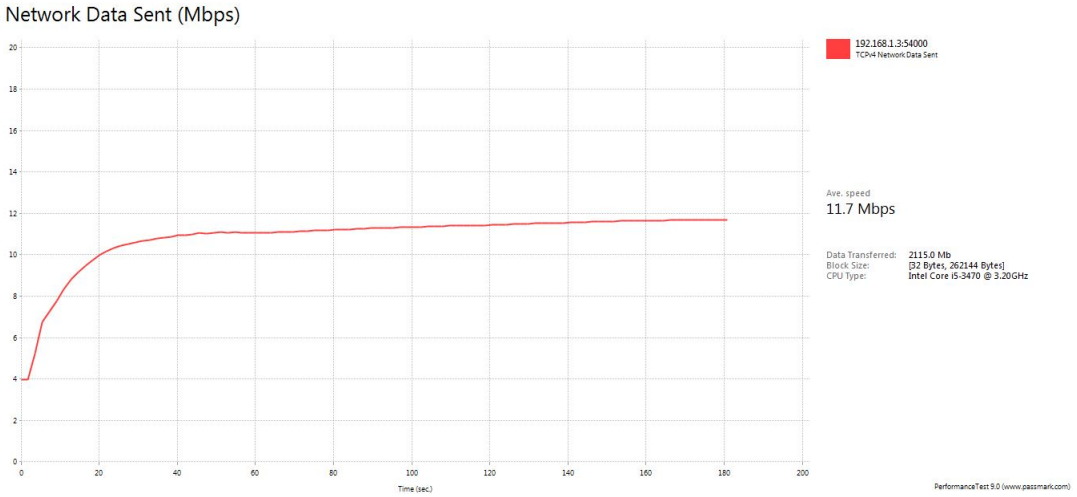
de 54 Mbps, la cual es la que se indica en las especificaciones del router Linksys WRT54GL. Esta velocidad indica la tasa a la que se transmiten bits de un punto a otro, pero como se menciona antes, la herramienta mide los bits de datos netos, esto causa que la velocidad de transmisión dada por la herramienta sea menor que la teórica ya que son bits de datos y no bits de todo el paquete y sus encabezados.

Tabla 11.1: Velocidad de transmisión obtenida con PassMark PerformanceTest utilizando el puerto 54000, un tiempo de 20 segundos y con paquetes de tamaño constante en cada prueba.

	32 B	10 KB	100 KB	200 KB	262 KB
Velocidad de transmisión (Mbps)	3.98	6.78	11.88	11.18	9.91
	3.94	8.87	11.43	10.47	10.11
	3.75	9.09	12.34	10.16	10.59
	3.98	6.82	12.22	11.09	9.56
	3.96	9.31	11.94	10.09	9.34
Promedio (Mbps)	3.92	8.17	11.96	10.6	9.90

[Elaboración propia]

Figura 11.2: Gráfica de velocidad de transmisión obtenida con PassMark PerformanceTest utilizando el puerto 54000, un tiempo de 3 minutos y con paquetes de tamaño variable (de 32 bytes a 262.144 Kbytes).



[Elaboración propia]

En la tabla 11.1 se observa que al aumentar el tamaño de los paquetes también incrementa la velocidad de transmisión, esto sucede porque con paquetes más grandes se puede transmitir más rápido la información ya que son menos bits de los encabezados los que se tienen que enviar por la red. Es decir, si enviamos 10 paquetes de 32 bytes entonces se tienen que enviar los bits de 10 encabezados, pero si enviamos un solo paquete de 320 bytes entonces solo debe transmitirse los bits de un solo encabezado. Esto disminuye la cantidad de bits que se tiene que transmitir y por lo tanto aumenta la tasa neta de datos que se recibe y envía. En la misma tabla se observa que después de 100 KB comienza a disminuir la velocidad de transmisión, esto se debe a que con paquetes muy grandes comienzan a surgir problemas de saturación de buffer, congestión y pérdida de paquetes. Para cada red existe un límite de tamaño de los paquetes antes que comience a generar deficiencias, en este caso se puede deducir que el tamaño límite se encuentra cercano a los 100 KB.

En la figura 11.2 se observa una prueba realizada con tamaño de paquetes variable para conocer

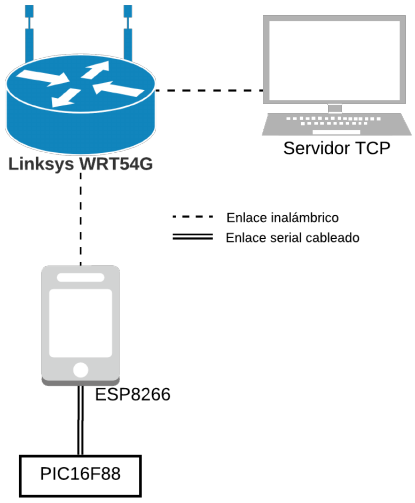
cual era el límite de la velocidad de transmisión, se observa claramente que existe un límite en 12 Mbps, dato que también puede corrobarse con los resultados en la tabla 11.1 ya que el promedio máximo no supera los 12 Mbps. Con esto se puede concluir que la velocidad máxima de transmisión por TCP es de 12 Mbps en la red de comunicación inalámbrica.

11.2. Medición de la velocidad de comunicación entre el PIC y el módulo WiFi

Es necesario saber la velocidad de comunicación mínima entre el PIC y el módulo que permitan aprovechar al máximo el ancho de banda de la red, esto permite determinar la frecuencia de actualización y la cantidad de robots que se pueden tener en la red. Para esto se realizó varias corridas con diferentes baudrate de comunicación entre el PIC y el módulo y se midió la velocidad de transmisión neta de datos que se obtenía, esto indica la tasa a la que se recibe cada comando válido en el PIC cuando el servidor se mantiene enviando paquetes constantemente. Para esto se generó una señal cuadrada en el PIC, el procedimiento es el siguiente:

1. Se inicia el servidor TCP en la computadora con la librería SocketTCP.h.
2. Se reinicia el PIC para que comience a enviar los comandos AT al módulo y pueda establecerse la conexión TCP con el servidor.
3. Cuando la conexión esté establecida, el servidor comenzará a enviar paquetes al módulo.
4. Si el PIC recibe un paquete correcto se niega el estado del puerto RB1, como se indica en la figura 10.2. Este paso se repite hasta que se detiene el servidor.

Figura 11.3: Diagrama de conexión para la medición de velocidad de comunicación con el módulo ESP8266 y el PIC16F88.



[Elaboración propia]

La señal de salida es medida con un osciloscopio para determinar su frecuencia, los resultados se observan en la tabla 11.2 y en la figura 11.4 se observa el resultado con 9.6 KBaud. Para el calculo

de la velocidad de datos neta se utiliza la siguiente ecuación en donde la frecuencia se multiplica por dos debido a que el cambio de estado se realiza en cada semiciclo y no en cada ciclo:

$$Vbps = (f * 2)(8 * n) \tag{11.1}$$

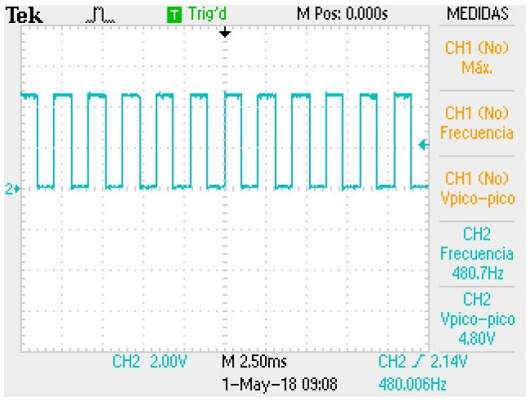
donde:

$Vbps$ es la velocidad de transmisión neta en bits por segundo.

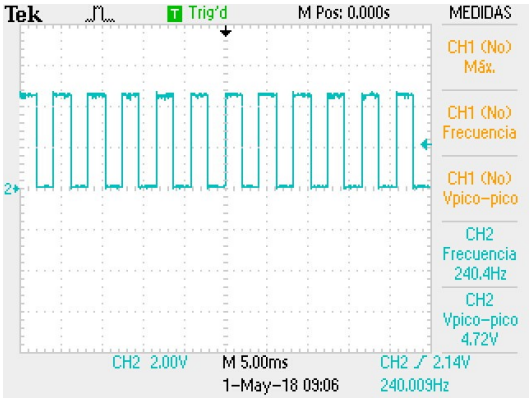
f es la frecuencia de la señal cuadrada.

n es el número de bytes enviados del servidor al cliente.

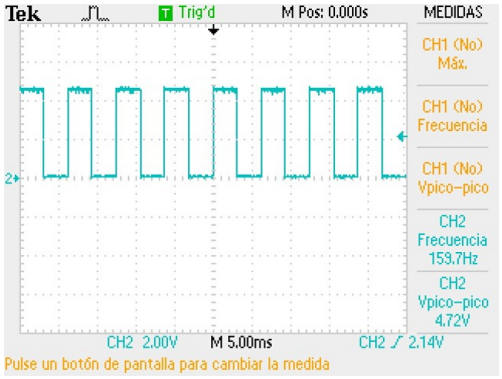
Figura 11.4: Señal obtenida al negar la salida RB1 en la recepción de paquetes con un baudrate de 9.6 KBaud.



(a) Paquetes de 1 Byte.



(b) Paquetes de 2 Bytes.



(c) Paquetes de 3 Bytes.

[Elaboración propia]

Al configurar el baudrate de comunicación entre el PIC y el módulo a 9.6 KBaud, también puede interpretarse en esta caso como 9.6 Kbps, se esperaría que la velocidad de transmisión neta fuera la misma, pero según la tabla 11.2 no es así ya que se obtiene una velocidad de 7.70 Kbps. Sin embargo, en el resto de resultados se observa que el porcentaje de error se mantiene aproximadamente igual, con esto se concluye que el error es constante y no se ve afectado al incrementar el baudrate. Este error es causado por tres factores:

1. En la figura 11.3 se observa que la comunicación entre el servidor y el módulo no es directa ya que el paquete tiene que pasar por el router, por lo cual, la fuente de error es causada por el

Tabla 11.2: Resultados de la velocidad de comunicación entre el PIC y el módulo y la velocidad de transmisión de datos neta con paquetes de 1 Byte entre el servidor y el cliente.

Baudrate (KBaud)	**Frecuencia (Hz)**	**Velocidad de transmisión (Kbps)**	**%Error**
9.6	481	7.70	19.83
19.2	960	15.36	20.00
38.4	1921	30.74	19.96
50	2501	40.02	19.97
250	12505	200.08	19.97

[Elaboración propia]

tiempo que le toma viajar al paquete desde el servidor al router y luego del router al módulo WiFi.

2. El tiempo que le toma al microprocesador en el módulo ESP8266 desencapsular el paquete para enviar solamente los datos hacía el PIC.

3. El tiempo que toma generar y enviar el ACK desde el módulo al servidor para confirmar que se recibió el paquete con éxito.

En la figura 11.4 se observa que al modificar el tamaño de los paquetes también cambia la frecuencia obtenida, aunque si colocamos estas frecuencias en la ecuación 11.1 se observa que se obtiene la misma velocidad de transmisión neta de 7.70 Kbps. Con esto podemos concluir que el tamaño de los paquetes no afecta en la velocidad de transmisión neta, esto aplica solamente cuando los paquetes son pequeños, ya que con paquetes de mayor tamaño la velocidad de transmisión si se ve afectada como se mencionó en la sección anterior.

La velocidad de transmisión y el porcentaje de error obtenidos permite determinar la frecuencia de actualización que debe tener cada robot en la red, la frecuencia de actualización obtenida es suponiendo que los paquetes se envían de forma consecutiva a cada robot, esto quiere decir que en un intervalo de tiempo primero se envía un paquete de 4 bytes al primer robot, luego un paquete al segundo robot y así sucesivamente hasta que se comienza nuevamente con el primero. La frecuencia de actualización indica un cambio de estado en el robot, el comando de 4 bytes recibido contiene la información suficiente para que el robot realice una acción y cambie su estado actual. La ecuación obtenida es la siguiente:

$$F_A = \frac{Baudrate - (Baudrate * 0.2)}{8n} \tag{11.2}$$

donde:

F_A es la frecuencia de actualización de cada robot.

$Baudrate$ es la velocidad de comunicación teórica entre el PIC y el módulo.

n es el numero de bytes enviados del servidor al ESP8266.

En la ecuación 11.2 se resta el 20 % del baudrate teórico debido al error constante que se obtuvo en la tabla 11.2. En la tabla 11.3 se observa que con 9.6 Kbaud se puede obtener una frecuencia de actualización de 240 Hz y para poder determinar la cantidad de robots que pueden mantener esa misma frecuencia en la red solamente se debe dividir la velocidad de transmisión máxima (12 Mbps) y la velocidad de transmisión que en este caso es de 7.7 Kbps, esto indica que 1558 agentes pueden conectarse a la red manteniendo una frecuencia de actualización de 240 Hz. También se observa que

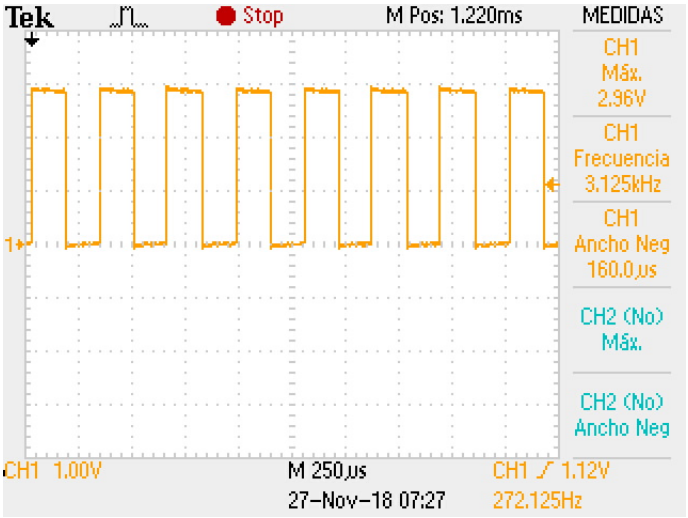
si se incrementa el baudrate aumenta la frecuencia de actualización pero disminuye la cantidad de agentes en la red, esto indica que debemos seleccionar el baudrate en base a la cantidad de agentes y frecuencia de actualización que necesitemos.

Tabla 11.3: Estimaciones de la frecuencia de muestreo y cantidad de robots para diferentes baud rate con paquetes de 4 bytes.

Baudrate (Kbaud)	**Velocidad de transmisión (Kbps)**	**Número de agentes**	**Frecuencia de actualización (Hz)**
9.6	7.7	1558	240
19.2	15.36	781	480
38.4	30.74	390	960
50	40.02	299	1250
250	200.08	60	6250

[Elaboración propia]

Figura 11.5: Frecuencia de actualización con 10 agentes conectados a la red y recibiendo paquetes de 4 bytes cada uno consecutivamente.



[Elaboración propia]

Se realizó la configuración del módulo WiFi y el PIC para que ambos pudieran comunicarse a 250 Kbaud, según la ecuación 11.2 se debe obtener una frecuencia de actualización de 6.25 KHz. El resultado se observa en la figura 11.5, pero se debe tomar en cuenta que en realidad el cambio de estado se realiza en cada semiciclo ya que la rutina programa es que niegue su salida al recibir un paquete de 4 bytes, por lo tanto, la frecuencia de actualización es de 6.25 KHz que corresponde a lo esperado. También se realizó la prueba de añadir más agentes a la red y comprobar que se sigue manteniendo la frecuencia de actualización en cada uno, en este caso solamente se logró conectar 10 agentes ya que no se tiene la capacidad de conectar a más de 60, se observó que con los 10 agentes conectados y recibiendo paquetes la frecuencia de actualización se sigue manteniendo sin cambios.

CAPÍTULO 12

Conclusiones

1. En comparación con Bluetooth y LTE, WiFi es la mejor opción para el diseño de una red de comunicación inalámbrica ya que es económica, veloz y fácilmente escalable.
2. El módulo ESP8266 es uno de los más económicos en el mercado y posee muy buenas prestaciones, pero el principal inconveniente es su elevado consumo de corriente que evita poder utilizar baterías de menor tamaño y por lo tanto no permite un diseño de robots tan pequeños.
3. Se debe alimentar el módulo ESP8266 con una batería ya que es muy sensible al ruido en su alimentación. Si se conecta a una fuente conmutada entonces se debe colocar capacitores de bypass entre las terminales de alimentación, un valor aceptable es de 1000 uF.
4. El PIC16F88 se utilizó para comunicarse con el módulo ESP8266 y así poder procesar los mensajes transmitidos por la red WiFi, este microcontrolador se seleccionó debido a su simplicidad de programación, bajo costo y disponibilidad.
5. En la fecha que se realizó esta investigación y tomando como referencia el laboratorio J313 de la Universidad del Valle de Guatemala, el canal óptimo para la red de comunicación es el 11, es el menos saturado en el ambiente y por lo tanto presenta menos interferencia. Sin embargo, si se detectan nuevas redes en el entorno o si se necesita cambiar de lugar la red, entonces debe realizarse nuevamente el análisis para determinar el canal óptimo.
6. La red de comunicación utiliza el protocolo 802.11g con una topología en estrella y en modo infraestructura, utiliza seguridad WEP y el protocolo de transporte TCP.
7. El servidor TCP se programó en C++ debido a que permite un mayor control en las conexiones y para que pueda ser compatible con el algoritmo de posicionamiento diseñado por el estudiante André Rodas de la Universidad del Valle de Guatemala.
8. La implementación de sockets asíncronos agrega flexibilidad y escalabilidad ya que se puedan adicionar o eliminar robots en la red de comunicación en cualquier momento y sin interrumpir durante mucho tiempo la ejecución del algoritmo.
9. Cuando los paquetes TCP son pequeños no se ve afectada significativamente la velocidad de transmisión en la red.
10. Cuando el tamaño de los paquetes TCP supera los 100 KB comienza a disminuir la velocidad de transmisión en la red, esto se debe a que comienzan a surgir problemas de saturación de buffer, congestión y pérdida de paquetes.

11. La velocidad de transferencia máxima en la red de comunicación inalámbrica por medio de TCP es de 12 Mbps.

12. Al realizar una comparación entre el baudrate de comunicación y la velocidad de transmisión obtenida se encontró que existe un 20 % de error. Esto indica que la velocidad de transmisión obtenida es menor a la velocidad de comunicación configurada.

13. Con un baudrate de comunicación mayor entre el microcontrolador y el módulo WiFi, se puede obtener mayores frecuencias de actualización en cada robot de la red.

14. No existe una velocidad de comunicación mínima, si no que esta depende de la frecuencia de actualización deseada y la cantidad de robots en la red.

CAPÍTULO 13

Recomendaciones

1. Se recomienda realizar el diseño e implementación de una nueva red de comunicación inalámbrica, pero con módulos RF, en especial con el módulo nRF24L01+ de Nordic. Con esto se puede tener mayores velocidades de transmisión y menor consumo de corriente, pero también incrementa la dificultad de implementación.

2. Se recomienda actualizar el firmware del ESP8266 siempre al más reciente ya que permite mayor control sobre distintos parámetros de la red y el módulo.

3. Se recomienda evaluar la posibilidad de utilizar versiones del ESP8266 que permitan programar el microprocesador de 32 bits utilizado en el módulo. Con esto se puede ahorrar espacio, tener mayor poder de procesamiento y poder alcanzar mayores frecuencias de actualización.

4. Si no se utiliza una versión del ESP8266 que permita programar su microprocesador, se recomienda utilizar un microcontrolador más sofisticado que permita alcanzar mayores velocidades de baudrate.

5. Se recomienda agregar el código para la recepción de datos desde el módulo al servidor en la librería SocketTCP.h para que futuros diseños de robots con sensores puedan enviar la información al servidor.

Bibliografía

[1] *Aumente su velocidad Wi-Fi con la ayuda de NetSpot - elija el mejor canal WiFi.* NetSpot. `https://www.netspotapp.com/es/wifi-channel-scanner.html`.

[2] *PassMark PerformanceTest - PC benchmark software.* PassMark Software. `https://www.passmark.com/products/pt.htm`.

[3] *PIC16F88 Datasheet.* Microhip, 2013. `https://www.microchip.com/wwwproducts/en/PIC16F88`.

[4] *ESP8266 Datasheet.* Espressif, 2018. `https://www.espressif.com/`.

[5] Andreu, J.: *Redes inalámbricas (Servicios en red):.* Editorial Editex, 2011.

[6] Buettrich, S. y A. Escudero.: *Unidad 04: Topología e Infraestructura Básica de Redes Inalámbricas.* Proyecto TRICALCAR, 2007. `https://unac.edu.pe/images/inventario/documentos/manuales/topologia-e-infraestructura_guia_v02.pdf`.

[7] Garin, D. y M. Hazard: *Bluetooth: Proyecto Elo322-Redes de Computadores I.* Departamento de Electrónica UTFSM, 2013. `http://profesores.elo.utfsm.cl/~agv/elo322/1s13/project/reports/Bluetooth.pdf`.

[8] Griera, J.Í. y J.M.B. Ordinas: *Estructura de redes de computadores.* Editorial UOC, S.L., 2009.

[9] Hernández, E., J. Guzmán, J. Mérida, O. Wantland, V. Villegas, W. Orozco y R. Ajtún: *Reingeniería de Megaproyectos Fase I-Módulo de Swarm Robotics.* Universidad del Valle de Guatemala, 2017.

[10] Horno, J. J. A.: *Redes de Área Local Inalámbricas: Diseño de la WLAN de Wheelers Lane Technology College.* Tesis de Doctorado, Universidad de Sevilla. España, 2008.

[11] Le Goc, Mathieu, Lawrence H Kim, Ali Parsaei, Jean Daniel Fekete, Pierre Dragicevic y Sean Follmer: *Zooids: Building blocks for swarm user interfaces.* En *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, páginas 97–109. ACM, 2016.

[12] Mauri, J.L., M.G. Pineda y F.B. Seguí: *IPTV: La televisión por internet.* Editorial Vértice, 2008.

[13] Pellejero, I., F. Andreu y A. Lesta: *Fundamentos y aplicaciones de seguridad en redes WLAN: de la teoría a la práctica.* Marcombo, 2006.

[14] Pickem, Daniel, Paul Glotfelter, Li Wang, Mark Mote, Aaron Ames, Eric Feron y Magnus Egerstedt: *The Robotarium: A remotely accessible swarm robotics research testbed.* En *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, páginas 1699–1706. IEEE, 2017.

[15] Pérez, F: *Redes Móviles Terrestres 4G.* Esc. Técnica Super. Ing. Univ. Pontif. Comillas, páginas 1–12, 2010.

[16] Rubenstein, Michael, Christian Ahler y Radhika Nagpal: *Kilobot: A low cost scalable robot system for collective behaviors.* En *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, páginas 3293–3298. IEEE, 2012.

[17] Schmidt, D. C., S. D. Huston y S. D Huston: *C++ Network programming, volume 1: mastering complexity with ACE and patterns.* Addison-Wesley Professional, 2001.

[18] Tan, Y.: *Handbook of Research on Design, Control, and Modeling of Swarm Robotics.* IGI Global, 2015.

ANEXO A

Enlaces a programas

1. NONOS SDK: `https://www.espressif.com/en/support/download/sdks-demos`
2. FLASH DOWNLOAD TOOLS: `https://www.espressif.com/en/support/download/other-tools`
3. GitHub: `https://github.com/MarlonCastillo/Red-Inal-mbrica-para-Swarm-Robotics.git`

Glosario

ACK Mensaje que se envía para confirmar la recepción de un paquete en la red. 46

benchmark Prueba utilizada para medir el rendimiento de un equipo o programa informático. 42

buffer Espacio de memoria en el que se almacenan datos. 43

DHCP Protocolo encargado de asignar direcciones IP de forma dinámica en una red. 33

IoT El internet de las cosas (en inglés, Internet of Things) es la interconexión de dispositivos en internet, por ejemplo, sensores, teléfonos, carros, etc. 26

latencia Tiempo que tarda un paquete de datos en llegar de un punto a otro. 25