SISTEMA DE LOCALIZACIÓN EN UNA RED DE ÁREA LOCAL INALÁMBRICA (WLAN) 802.11b

AUTORES:
ELKIN FERNANDO RUIZ ÁLVAREZ
DANIEL SARAVIA QUIROGA

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERIAS FÍSICO-MECÁNICAS
ESCUELA DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES
BUCARAMANGA
2005

SISTEMA DE LOCALIZACIÓN EN UNA RED DE ÁREA LOCAL INALÁMBRICA (WLAN) 802.11b

AUTORES:

ELKIN FERNANDO RUIZ ÁLVAREZ
DANIEL SARAVIA QUIROGA

PROYECTO DE GRADO

DIRECTORES:

PhD. OSCAR GUALDRÓN GONZÁLEZ

MIE. SAMUEL GONZALO PINZÓN BARRIOS

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERIAS FÍSICO-MECÁNICAS
ESCUELA DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES
BUCARAMANGA

2005

DEDICATORIA

A mis papás, hermanos y abuelo Manuel.

Elkin Fernando Ruiz Álvarez

A mi mamá, a mi papá y mi abuela Erne.

Daniel Saravia Quiroga

AGRADECIMIENTOS

Gracias a Dios por las oportunidades y por el mundo.

Agradecemos de forma especial a nuestros directores PhD. Óscar Gualdrón González y MIE. Samuel Gonzalo Pinzón Barrios por su orientación, sus conceptos y sus grandes aportes a este trabajo.

A nuestras familias por su paciencia y su apoyo en todo momento.

A nuestros amigos y compañeros por acompañarnos en este camino y por darnos siempre su voz de aliento.

A la Ingeniera Shirley Paola Herrera por su ayuda incondicional en varias instancias del proyecto.

Elkin y Daniel

Quiero agradecer de manera especial a Erika y Adriana por su amistad incondicional, a mi ángel y voz de aliento Angie, a Hugo y Orfa por sus consejos, a Daniel Saravia por darme la oportunidad de trabajar a su lado y por supuesto a toda mi familia. Sin ustedes no habría sido posible la realización de este proyecto.

Elkin

A Elkin, mi compañero, por aceptar el reto de conocerme y trabajar conmigo.

A Shirley por todas las sonrisas.

Daniel

TÍTULO: SISTEMA DE LOCALIZACIÓN EN UNA RED DE ÁREA LOCAL INALÁMBRICA (WLAN) 802.11b

AUTORES: Elkin Fernando Ruiz Álvarez
Daniel Saravia Quiroga

PALABRAS CLAVES: Localización, *WLAN*, *outdoor*, puntos de acceso, tarjeta inalámbrica, red neuronal, algoritmo de búsqueda, interfaz.

En este trabajo, se diseña e implementa un sistema de localización dirigido especialmente a dispositivos portátiles. El sistema explora la posibilidad de determinar la ubicación de un dispositivo móvil a partir de la información de potencia de las señales emitidas por dos puntos de acceso (*access points*) y que son recibidas por una tarjeta inalámbrica, operando bajo el estándar 802.11b. Dicho sistema ha sido diseñado para trabajar en un ambiente exterior (*outdoor*), aplicando un modelo empírico y con área de cobertura de 50x50 m^2.

Las técnicas empleadas en este proyecto son un algoritmo de búsqueda y una red neuronal, escritos en *Matlab*, a los cuales se les dan como entradas dos potencias recibidas por un adaptador inalámbrico y leídas a través de la herramienta *Netstumbler*, y entregan como resultado ya sea un punto en coordenadas cartesianas o una posible zona de ubicación del dispositivo.

Para el desarrollo de las técnicas, se construyen bases de datos con información de intensidad de señal, correspondientes a una grilla de puntos separados 3 m entre si. El algoritmo de búsqueda emplea las bases de datos para buscar coincidencia con los valores de potencia que se le suministran y entregar una zona de ubicación. En el caso de la red neuronal se emplea un modelo de aprendizaje supervisado que utiliza la base de datos para su entrenamiento, donde busca un patrón en los valores de potencia que le permita calcular una posición representada en coordenadas X e Y.

Para la visualización de los resultados, se presenta una interfaz gráfica de fácil operación, donde el usuario escoge la técnica de su preferencia ya sea red neuronal o algoritmo de búsqueda.

TITLE: LOCATION SYSTEM IN A WIRELESS LOCAL AREA NETWORK (WLAN) 802.11b

AUTHORS: Elkin Fernando Ruiz Álvarez
Daniel Saravia Quiroga

KEY WORDS: Location, WLAN, outdoor, access points, wireless card, neural network, search algorithm, interface.

In this project, a portable devices-aimed location system is designed and implemented. The system explores the possibility to determine the mobile device position from information obtained from two access points' emitted signals which are received by a wireless card, working on 802.11b standard. Such a system has been designed to operate in outdoors, by applying an empirical model and covering a 50x50 m^2 area.

The developed techniques in this project are a search algorithm and a neural network, written in Matlab, which receive as inputs two signal strength values registered by a wireless adapter and read through Netstumbler, and calculate a point or a possible device location zone.

For the techniques development, databases are built with signal strength information, of three-meters-separated point grid. The search algorithm seeks for coincidences with the supplied strength values and shows a location zone. In neural network case, a supervised learning model is used which utilize the database for its training, where a pattern is searched so that it calculate an X-Y position.

For displaying the results, an easy-operation graphical interface is presented, where the user choose the technique, say neural network or search algorithm.

TABLA DE CONTENIDO

Pág.

INTRODUCCIÓN 12

1. FUNDAMENTOS TEÓRICOS 16

1.1 RED DE ÁREA LOCAL INALÁMBRICA (WLAN) 16
1.1.1 Modo de operación.. 17
1.1.2 Puntos de acceso (*AP*). 17
1.1.3 Configuraciones 18

1.2 Wi-Fi (WIRELESS FIDELITY) 20
1.2.1 Estándares de *Wi-Fi*. 21
1.2.2 Compatibilidad entre diferentes estándares. 21

1.3 LOCALIZACIÓN EN REDES INALÁMBRICAS 22
1.3.1 Modelo empírico 23
1.3.2 Modelo de propagación. 23
1.3.3 Comparación empírico vs. Propagación. 24

1.4 SISTEMAS DE LOCALIZACIÓN – GPS 25

1.5 PÉRDIDAS EN ESPACIO LIBRE 25

2. SISTEMA DE LOCALIZACIÓN 27

2.1 UBICACIÓN DE LOS PUNTOS DE ACCESO (AP) 27

2.2 SOLUCIÓN PROPUESTA 34
2.2.1 Solución basada en la ecuación de pérdidas en espacio libre 34
2.2.2 Solución mediante redes neuronales. 34
2.2.3 Resultados del entrenamiento. 44
2.2.4 Solución mediante algoritmo de búsqueda. 47

3. CAMPAÑAS DE MEDICIÓN 49

3.1 SELECCIÓN DEL ESCENARIO 49

3.2 TOMA DE MEDIDAS 51
3.2.1 Toma de medidas en un solo punto. 51
3.2.2 Construcción del mapa de potencias. 55

3.3 VERIFICACIÓN DEL DISEÑO REALIZADO EN LABORATORIO 58
3.3.1 Red neuronal. 58
3.3.2 Algoritmo de búsqueda. 60

4. PRUEBAS Y RESULTADOS 64

4.1 RED NEURONAL 66

4.2 ALTERNATIVAS DE SOLUCIÓN..... 70
4.2.1 Respecto al algoritmo de búsqueda.. 71
4.2.2 Respecto a la red neuronal. 77

5. CONCLUSIONES Y RECOMENDACIONES 81

5.1 CONCLUSIONES..... 81

5.2 RECOMENDACIONES..... 84

BIBLIOGRAFÍA 86

BIBLIOGRAFIA COMPLEMENTARIA 88

ANEXO A.....90

LISTA DE FIGURAS

Pág.

Figura 1. Configuración punto a punto 19
Figura 2. Configuración modo infraestructura..... 20
Figura 3. Arquitectura 802.11..... 21
Figura 4. Configuración extremo - centro 29
Figura 5. Configuración extremo - extremo diagonal 30
Figura 6. Configuración centro - centro 31
Figura 7. Configuración extremo - extremo..... 32
Figura 8. Configuración asimétrica 33
Figura 9. Diagrama de la red neuronal 40
Figura 10. Desviación en la predicción 46
Figura 11. Desviación absoluta 46
Figura 12. Diagrama de bloques del algoritmo de búsqueda..... 47
Figura 13. Resultado ejemplo del algoritmo de búsqueda 48
Figura 14. Estadio 1° de Marzo y sus alrededores..... 50
Figura 15. Comportamiento de las tarjetas D-LINK y ORINOCO..... 52
Figura 16. Comportamiento en el tiempo de la potencia recibida por la tarjeta……54
Figura 17. Puntos de medición y secuencia de barrido..... 56
Figura 18. Distribuciones de potencia según mediciones..... 57
Figura 19. Errores entre ecuación de *Friis* y mediciones 57
Figura 20. Soluciones del algoritmo de búsqueda 62
Figura 21. Presentación de la Interfaz 69
Figura 22. Opciones de la Interfaz 70
Figura 23. Primera solución alternativa del algoritmo de búsqueda. ... 72
Figura 24. Ubicación de tres puntos de acceso en la grilla..... 73
Figura 25. Segunda solución alternativa del algoritmo de búsqueda ... 74
Figura 26. Ubicación de cuatro puntos de acceso en la grilla 76
Figura 27. Tercera solución alternativa para el algoritmo de búsqueda 77
Figura 28. Regresión, ecuación de *Friis* y dispersión de los datos 80

LISTA DE TABLAS

Pág.

Tabla 1. Comparación entre estándares 21
Tabla 2. Comparación modelo empírico vs. propagación 24
Tabla 3. Cotas para definir el error de aprendizaje..... 38
Tabla 4. Comparación entre una red 2-16-16-16-16-1 y una 2-16-1 .. 41
Tabla 5. Comparación entre redes con diferente número de neuronas en la capa oculta 42
Tabla 6. Tabla de parámetros de las redes 44
Tabla 7. Resultados del entrenamiento datos teóricos 45
Tabla 8. Datos estadísticos de las tarjetas D-LINK y ORINOCO. 52
Tabla 9. Medidas estadísticas del experimento en el punto (24,24) 55
Tabla 10. Parámetros y resultados experimentales de la red. 60
Tabla 11. Resultados obtenidos del algoritmo de búsqueda 65
Tabla 12. Esquema consolidado de error para algoritmo de búsqueda66
Tabla 13. Resultados obtenidos de la red neuronal. 67
Tabla 14. Esquema consolidado de error para red neuronal 68
Tabla 15. Parámetros y resultados alternativos para la red neuronal. 78

LISTA DE ANEXOS

Pág.

ANEXO A *MATLAB* **.....90**

ANEXO B EQUIPO UTILIZADO **.....94**

ANEXO C NETWORK STUMBLER v.0.4.0**.....100**

ANEXO D CÓDIGOS **.....103**

ANEXO E REDES NEURONALES ARTIFICIALES **.....133**

INTRODUCCIÓN

El estándar *IEEE* 802.11, llamado también *Wireles LAN* (WLAN) es una de las recientes adiciones al campo de las comunicaciones inalámbricas. El estándar WLAN, aprobado por el grupo de trabajo *IEEE* 802.11 en Junio de 1997, fue desarrollado para mejorar los usos de las redes de computadores actuales. Éste habilita las redes de área local para que sean extendidas a través de las comunicaciones inalámbricas. La tecnología ha probado ser extremadamente útil en los casos cuando los computadores móviles necesiten conectarse a núcleos de LAN existentes.

La especificación de este estándar que está siendo más ampliamente usada y aceptada para configuración y operación de redes inalámbricas es la *IEEE* 802.11b; se trata de una extensión del estándar que define enlaces de 11 Mbps y transmisión en la banda de 2.4 GHz. Esta extensión es la más popular, por encima de la extensión 802.11a, debido principalmente a su gran cobertura en ambientes interiores, a pesar de su ancho de banda menor comparado con el 802.11a. En la operación a 2.4 GHz (a diferencia de los 5GHz para 802.11a), las señales transmitidas usando 802.11b pueden penetrar objetos con menos reducción de energía de la señal y pueden viajar distancias más largas, lo cual ofrece una señal de mayor cobertura. Debido a la inmensa popularidad y rápida adopción de 802.11b, fue introducida la extensión 802.11g para proveer una mayor tasa de transferencia y que además fuera compatible con dispositivos de 802.11b.

El trabajo propuesto pretende utilizar los equipos WLAN disponibles en la escuela y diseñar un sistema de localización que trabaje eficientemente en ambientes exteriores. Más específicamente, el proyecto pretende diseñar un sistema de localización basado en software dirigido especialmente a dispositivos portátiles. El sistema explorará la posibilidad de determinar la ubicación de un dispositivo móvil a partir de la información de potencia de las señales emitidas por varios puntos de acceso (*access points*) y que serán recibidas por una tarjeta inalámbrica, operando bajo el estándar 802.11b. Para tal efecto, se probarán varias tarjetas de diferentes fabricantes y referencias, con el objeto de determinar cual de ellas ofrece un mejor desempeño.

1. FUNDAMENTOS TEÓRICOS

1.1 RED DE ÁREA LOCAL INALÁMBRICA (WLAN)

Una WLAN (*Wireless Local Area Network*) es una red de comunicación de datos flexible que se tiene como alternativa o extensión de las conocidas redes cableadas LAN. Debido a su operación por medio de ondas electromagnéticas, facilita la movilidad de los usuarios conectados a la red. Este tipo de redes va en crecimiento, ya que están siendo usadas en diferentes campos gracias a su facilidad de instalación y gran cantidad de servicios.

Dentro de sus características más importantes están la movilidad, la facilidad de instalación y la flexibilidad. La primera de ellas, movilidad, consiste en tener acceso a la información en tiempo real y en cualquier lugar en que se encuentre el usuario, siempre dentro de la red. La segunda, facilidad de instalación, elimina el tener que construir redes cableadas y las incómodas obras que en ocasiones éstas acarrean, mejorando el aspecto físico de las instalaciones donde la red se encuentre y permitiendo un mayor y rápido acceso a usuarios temporales. Y finalmente, la tercera, flexibilidad, que sugiere que, en comparación con las redes cableadas, este tipo de redes pueden superar obstáculos, incluso paredes y se hacen muy útiles en lugares donde realizar una obra de cableado es poco posible y costosa.

Sin embargo, cabe mencionar algunas limitaciones que las WLANs poseen, entre ellas, un ancho de banda compartido entre los usuarios; esto hace que el ancho de banda se reduzca y que sea muy difícil la

transferencia de archivos de gran tamaño. Además, por ser una red abierta posee problemas de seguridad, puesto que cualquier usuario que posea un dispositivo inalámbrico compatible puede acceder a todos los datos que circulen por la red.

1.1.1 Modo de operación. Como ya se mencionó, las WLANs operan mediante ondas electromagnéticas que transportan datos llevando la energía a un receptor remoto. Los datos están superpuestos sobre las ondas lo cual les permite ser extraídos con facilidad por el receptor.

Si los datos son transmitidos a diferentes frecuencias, no habrá interferencia entre ellas, pues puede haber coexistencia. Para extraer los datos el receptor se sitúa en la frecuencia de la portadora y descarta las demás. En una WLAN, es importante contar con un punto de acceso (*AP*[1]). Éste se conecta a un punto fijo de la LAN cableada, recibe la información, la almacena y la envía entre la WLAN y la LAN. Los usuarios se conectan a la red por medio de adaptadores inalámbricos (tarjetas) que proporcionan una interfaz entre el servicio de operación de red y las ondas, a través de antenas. [Wikipedia/WLAN]

1.1.2 Puntos de acceso (*AP*). Estos son dispositivos usados para interconectar dispositivos de comunicación inalámbrica, con el fin de crear redes. Están generalmente conectados a redes cableadas y pueden enviar datos de un lado a otro de la red. Cuando se conectan varios de ellos entre sí, estas redes proporcionan lo que se denomina *roaming*[2].

Los *APs* determinan cuando puede transmitir un cliente. Sin embargo,

[1] Access Point
[2] Extensión del servicio de conectividad en una red que es diferente a la red con la cual la estación esta registrada.

para los sistemas que operan bajo los estándares 802.11[3], se emplea un algoritmo seudo-aleatorio de distribución para determinar si el cliente puede transmitir. Otro uso de los *APs* es como puente entre dos redes cableadas. [Wikipedia/Access_point]

Hay un número limitado de frecuencias legalmente disponibles para el uso de redes inalámbricas. Generalmente, los *APs* usan diferentes frecuencias para comunicarse con sus clientes con el fin de evitar interferencia entre dos sistemas vecinos. Los dispositivos inalámbricos son capaces de detectar el tráfico de datos en otras frecuencias y pueden rápidamente cambiarse a otra frecuencia para alcanzar mejor recepción en un *AP* distinto. [Wikipedia/Access_point]

Normalmente, un *AP* bajo el estándar 802.11 puede soportar hasta 30 clientes en la red dentro de un radio de aproximadamente 100 m, aunque esta característica podría modificarse debido a diferentes variables, como el ambiente en que se esté trabajando (interiores o exteriores), la consistencia del terreno, los obstáculos, el tipo de antena, el clima, la frecuencia de operación y la potencia de salida del dispositivo.

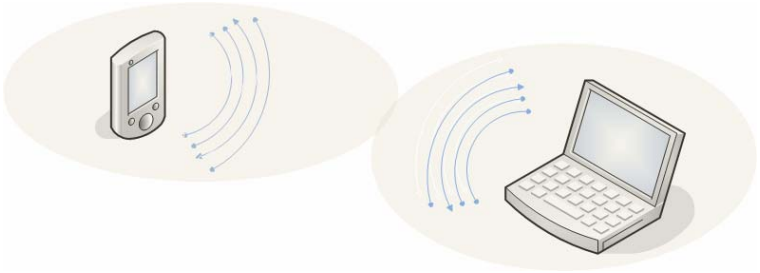
1.1.3 Configuraciones. Existen varios tipos de configuraciones de red que pueden ser simples, como la tipo punto a punto, o complejas, como la tipo infraestructura, dependiendo de la aplicación para la cual sea necesaria.

La conexión punto a punto (*peer-to-peer*), que se muestra en la figura 1, es una conexión entre dos equipos que tienen una tarjeta inalámbrica WLAN y cada uno de ellos tiene acceso al otro, pero no a un servidor

[3] Estándar IEEE para comunicaciones inalámbricas

central. Típicamente, se utiliza para que dos estaciones compartan la misma conexión a Internet. Este tipo de configuración no requiere configuración inicial.

Figura 1. Configuración punto a punto



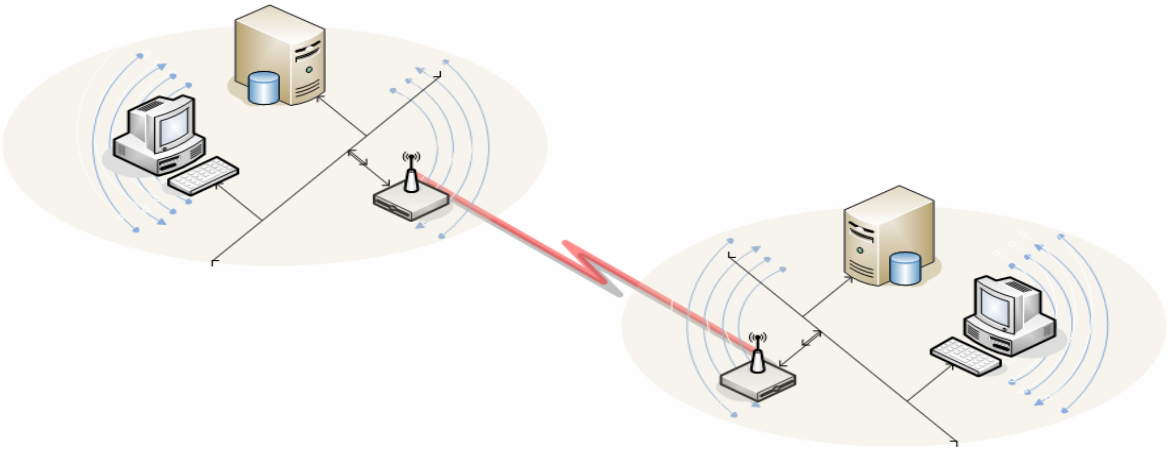
Fuente: Autores del proyecto

En el modo de infraestructura que se observa en la figura 2, un AP hace de puente entre una red inalámbrica y una red cableada *Ethernet*[4]. En este modo también, es necesario un punto de conexión central para los clientes de la WLAN, por lo cual, se requieren *APs* que cumplan con esta tarea. [Wikipedia/WLAN]

Finalmente, se puede concluir que las redes inalámbricas son una realidad y que aprovechando al máximo todas sus características se convertirán en sistemas comunes en hogares, oficinas e instituciones educativas, puesto que aportan la movilidad y la flexibilidad, que las redes cableadas no pueden proporcionar. El único inconveniente a resolver es que todos los dispositivos se comuniquen a través del mismo protocolo, para así, garantizar su expansión.

[4] Tecnología utilizada para acceder al medio, capa 2 Protocolos TCP/IP

Figura 2. Configuración modo infraestructura



Fuente: Autores del Proyecto

1.2 Wi-Fi (WIRELESS FIDELITY)

Es la tecnología de transmisión de datos o acceso a Internet inalámbrico que funciona basado en los estándares 802.11 a, 802.11b ó 802.11g y trabaja en la banda de 2.4 GHz que es de uso libre. Esta banda por ser de uso libre no tiene regulación del gobierno, por lo que está expuesta a soportar interferencias y tiene limitaciones en área de cobertura. Por consiguiente, es utilizada generalmente en espacios cerrados y grupos privados como cafés Internet, empresas, aeropuertos, centros comerciales y *hotspots*[5], entre otros.

El uso de *Wi-Fi*, está encaminado a lograr una mayor expansión de las WLANs que permitan el acceso a la red de cualquier dispositivo inalámbrico que posea una tarjeta *Wi-Fi*.

[5] Hotspot: serie de *APs* ubicados cubriendo un área, cada uno de ellos conectados a una red diferente.

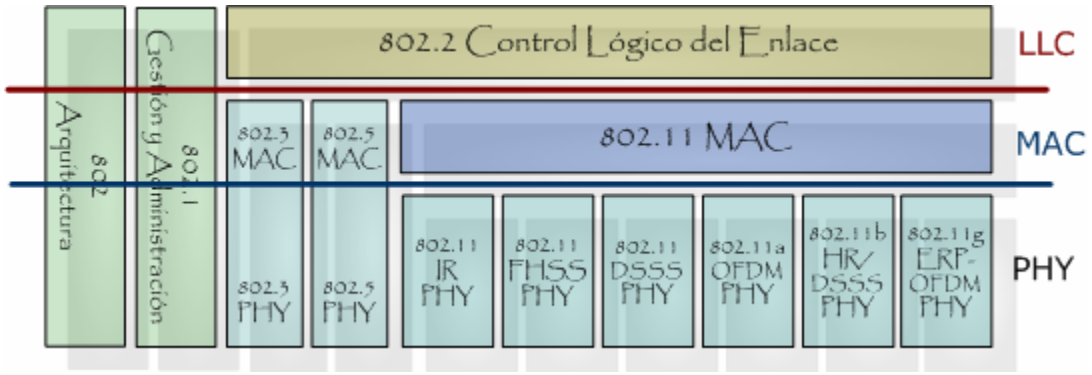
1.2.1 Estándares de *Wi-Fi*. Los estándares que actualmente se están usando en la tecnología *Wi-Fi* son los *IEEE* 802.11a, 802.11b y 802.11g. El primero de ellos, 802.11a, tiene una tasa de transmisión hasta de 54 Mbps en la banda de 5 GHz, 802.11b opera con una tasa de transmisión de datos hasta de 11 Mbps en la banda de 2.4 GHz, y finalmente 802.11g, con tasa de transmisión de datos hasta de 54 Mbps operando en la banda de 2.4 GHz. En la Tabla 1 se encuentra un cuadro comparativo de estos estándares. [Ministerio de Comunicaciones, 2003]

Tabla 1. Comparación entre estándares

Estándar de tecnología WLAN	**802.11a**	**802.11b**	**802.11g**
Máxima tasa de transferencia	54 Mbps	11 Mbps	54 Mbps
Frecuencia	5 GHz	2.4 GHz	2.4 GHz
Cobertura (Aprox.)	50 m	100 m	100 m
Compatibilidad	-	802.11g	802.11b
Problemas de interferencia	Teléfonos inalámbricos a 5 GHz	Hornos microondas, teléfonos inalámbricos a 2.4 GHz, *Bluetooth*	Hornos microondas, teléfonos inalámbricos a 2.4 GHz

1.2.2 Compatibilidad entre diferentes estándares. El estándar *IEEE* 802.11 define la operación compartida de todos los dispositivos que cumplen con él. Sin embargo, los dispositivos pueden ser construidos con diferentes esquemas de modulación y usando diferentes diseños que hacen que no sea posible la comunicación.

Figura 3. Arquitectura 802.11



Fuente: Adaptado de "*Wireless LAN location system*" pag. 6 [Shih, 2003]

IEEE 802.11 define los componentes del Control de Acceso al Medio (*MAC*[6] *Medium Access Control*) y la capa Física *(PHY*[7]) para la transmisión. La capa *MAC* es un método usado por todos los dispositivos 802.11 y 802.11x.

Los dispositivos que usan diferentes *PHY* no son capaces de comunicarse unos con otros. IEEE 802.11 también ha especificado dos técnicas de transmisión, a saber, Espectro Extendido por Salto de Frecuencia (*FHSS*, *Frecuency Hopping Spread Spectrum*) y Espectro Extendido por Secuencia Directa (*DSSS*, *Direct Sequence Spread Spectrum*). Ambas técnicas necesitan ser soportadas por 802.11 para que se pueda realizar la comunicación. *FHSS* y *DSSS* no poseen interoperabilidad, es decir, ninguno de los dos puede recibir información del otro. Los estándares IEEE 802.11b y 802.11g han añadido esquemas de modulación para mejorar las tasas de transmisión de datos usando *DSSS.* [Shih, 2003]

1.3 LOCALIZACIÓN EN REDES INALÁMBRICAS

Los valores de potencia de la señal desde el *AP* son medidos por el

[6] Capa de Acceso a la red Protocolos TCP/IP
[7] Physic layer (capa física) Protocolos TCP/IP

dispositivo de posicionamiento, y basados en estas medidas tales como calidad de la señal, potencia, relación señal a ruido, entre otras, se han implementado dos modelos: el modelo empírico y el modelo de propagación.

1.3.1 Modelo empírico. Éste es un modelo basado en medidas hechas previamente. Se construye un mapa que contiene marcas de determinados puntos. Las coordenadas de los puntos son conocidas y la información de la señal se recoge y se almacena en una base de datos. Cuando se tiene un dispositivo con una posición desconocida, la información de la señal desde el *AP* se envía a la base de datos para hacer la comparación. El modelo, entonces, encuentra con cual de los datos de los puntos conocidos es más compatible la nueva entrada.

Hay dos desventajas principales con este modelo:

- Se requiere un gran esfuerzo en la construcción tanto del mapa como en la base de datos.
- El modelo puede perder exactitud si las condiciones del medio cambian con respecto a cuando fue construido el mapa.

Sin embargo, hay soluciones para estas desventajas: para la primera, la construcción de un mapa basado en *software* especificado para recolectar los datos y registrarlos en la base; y para la segunda, construir varios mapas de potencia en diferentes condiciones del ambiente.

1.3.2 Modelo de propagación. Aprovecha la dependencia que hay entre de las pérdidas de potencia de una onda y el ambiente. Estas pérdidas pueden ser modeladas por medio de radio propagación conocida y teorías de pérdidas por medio (*pathloss*). Usando estas teorías, la posición de un dispositivo con respecto a un *AP* puede ser calculada por el

valor de las pérdidas de potencia que se recibe. Teniendo las distancias entre tres o más *APs* se puede usar el método de triangulación para determinar la posición del dispositivo. Este modelo es ideal para usarlo en ambientes interiores con redes de las características de WLAN.

Aunque este modelo es la mejor solución para la localización WLAN, se han encontrado ciertos inconvenientes con la implementación: con el fin de alcanzar una buena exactitud, las medidas de las pérdidas que el sistema recibe desde los *AP* deben ser precisas, es decir, el sistema debe tener un completo diseño en *hardware* y *software*. El *hardware* debe detectar cuidadosamente la señal con el fin de obtener las pérdidas correctas, y el *software* debe entender específicamente la comunicación desde el *hardware* para obtener los valores de señal correctos y hacer los cálculos de distancia. Los sistemas que implementan el modelo de propagación son mucho más complejos.

1.3.3 Comparación empírico vs. Propagación. La tabla 1.2 muestra los aspectos que difieren entre un modelo y otro**:**

Tabla 2. Comparación modelo empírico vs. propagación

Modelo / Características	Empírico	Propagación
Complejidad del Diseño	Bajo	Alto
Exactitud de Localización	Bueno	Bueno
Reusabilidad	Baja	Alta
Costo de Implementación	Alto	Bajo

Fuente: Adaptado de "Wireless LAN location system"

1.4 SISTEMAS DE LOCALIZACIÓN – GPS[8]

Los sistemas *GPS* funcionan con ayuda de satélites que proporcionan la ubicación de los mismos. El sistema entrega información de localización dando las coordenadas x, y e z. Estos sistemas han sido implementados en ambientes exteriores. La causa de que no hayan sido usados para ambientes interiores es que las distancias relativas entre los puntos de referencia y el dispositivo no se pueden calcular fácilmente ya que los canales de comunicación por los que viaja la señal son extensos y no siempre están vacíos, por lo cual la exactitud se ve altamente afectada.

La relación que existe entre este proyecto y los *GPSs*, haciendo una analogía y guardando las proporciones, está en que los *APs* dentro de la WLAN hacen las veces de un satélite y ciertos algoritmos a partir de las potencias recibidas, derivan las coordenadas x e y del dispositivo móvil.

1.5 PÉRDIDAS EN ESPACIO LIBRE

En sistemas de comunicación hay un tipo de pérdidas las cuales son sumamente importantes, puesto que en la operación de dichos sistemas se presentan cuando éstos se encuentran tanto en ambientes exteriores (*outdoor*) como en interiores (*indoor*). Estas pérdidas están caracterizadas por la ecuación de transmisión *Friis*

$$\frac{P_r}{P_t} = G_a G_b \left(\frac{\lambda}{4\pi r} \right)^2 \quad (1)$$

[8] Global Positioning System

Donde G_a y G_b son las ganancias de antena, r es la distancia entre ellas, y λ es la longitud de onda.

Esta ecuación puede ser arreglada para expresar las pérdidas en espacio libre como:

$$L_f = \left(\frac{4\pi r f}{c} \right)^2 \quad (2)$$

También se puede obtener la siguiente expresión en decibels con la frecuencia f en megahertz y la distancia R en kilómetros: [Saunders, 1999]

$$L_{F(dB)} = 32.4 + 20\log(R) + 20\log(f)\,. \quad (3)$$

2. SISTEMA DE LOCALIZACIÓN

La solución de cualquier problema de ingeniería debe estar soportada por un estudio teórico donde se analicen aspectos que puedan influir en los resultados. Por consiguiente, en este capítulo, se aborda lo concerniente a la solución teórica de la posición del dispositivo al interior de la red, tomando como punto de partida una óptima ubicación de los puntos de acceso, de tal forma que no existan puntos con igual combinación de potencia. Posteriormente, se proponen soluciones y se aplican dos técnicas computacionales que permiten hallar las coordenadas x e y de la ubicación del dispositivo en la zona.

2.1 UBICACIÓN DE LOS PUNTOS DE ACCESO (AP)

Una vez se ha determinado el escenario para realizar las campañas de medición es de vital importancia para la realización del proyecto hacer una adecuada selección del punto de ubicación de los puntos de Acceso dentro del área escogida. Para tal efecto hay que tener en cuenta los rangos de tolerancias y cobertura de los equipos. Adicionalmente debe evitar que dos puntos diferentes tengan la misma distribución de potencia. Este hecho se puede presentar debido a que sólo se cuenta con dos *APs* para nuestro propósito.

Las especificaciones de los *APs* dicen que tienen un rango de cobertura de 100m, mientras que el escenario escogido tiene un área de 50 x 50 m^2, por lo tanto la cobertura del escenario es completa.

Un mapa con la distribución de potencias fue creado basado en la

ecuación de pérdidas en espacio libre (Ec. 3).

Para determinar una posición adecuada de los puntos de acceso, donde no haya dos puntos con la misma distribución de potencia se elaboró un programa en *Matlab*[9] que permite establecer el porcentaje de puntos que presentan esta característica. Este programa fue probado emulando la ubicación de los puntos de acceso mediante la ecuación de pérdidas en espacio libre en diferentes puntos dentro del área. Las ubicaciones escogidas para la prueba son:

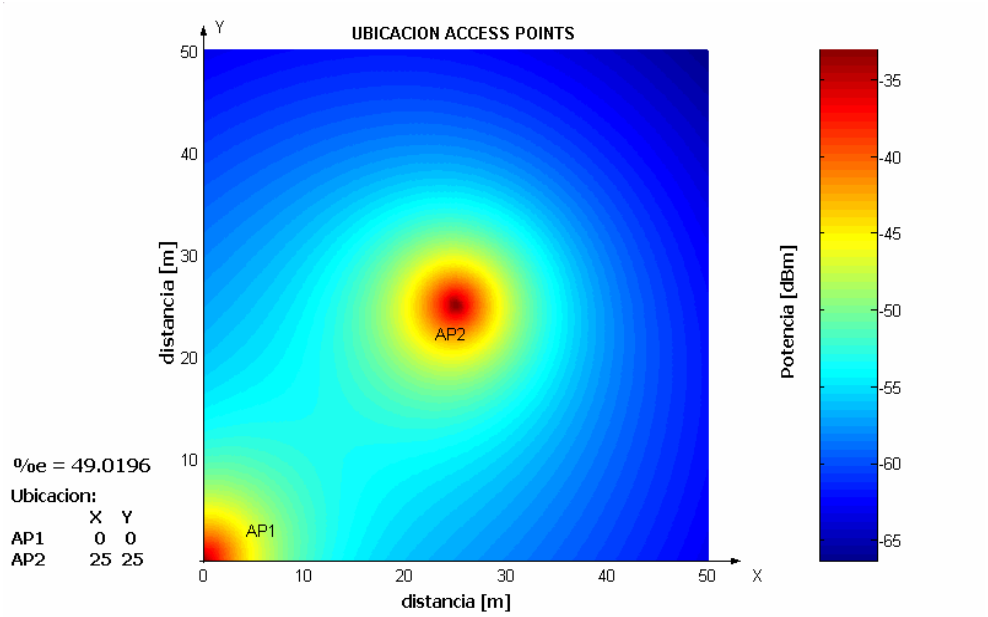
- AP1 en el punto (0,0) y AP2 en el punto (25,25)
- AP1 en (0,0) y AP2 en (50,50)
- AP1 en (25,12) y AP2 en (25,37)
- AP1 en (0,0) y AP2 en (0,50)
- AP1 en (0,0) y AP2 en (44,2)

Los resultados encontrados se muestran continuación.

La primera distribución escogida para los *APs* corresponde a uno en un extremo y el otro en el centro. Esta ubicación produjo un porcentaje de puntos con la misma combinación de potencia del 49%. En la figura 4 se puede apreciar una gráfica de la distribución de potencias.

[9] Lenguaje de alto desempeño que integra cálculos, visualización y programación en un ambiente de fácil uso donde los problemas soluciones están expresados en una notación matemática familiar al usuario.

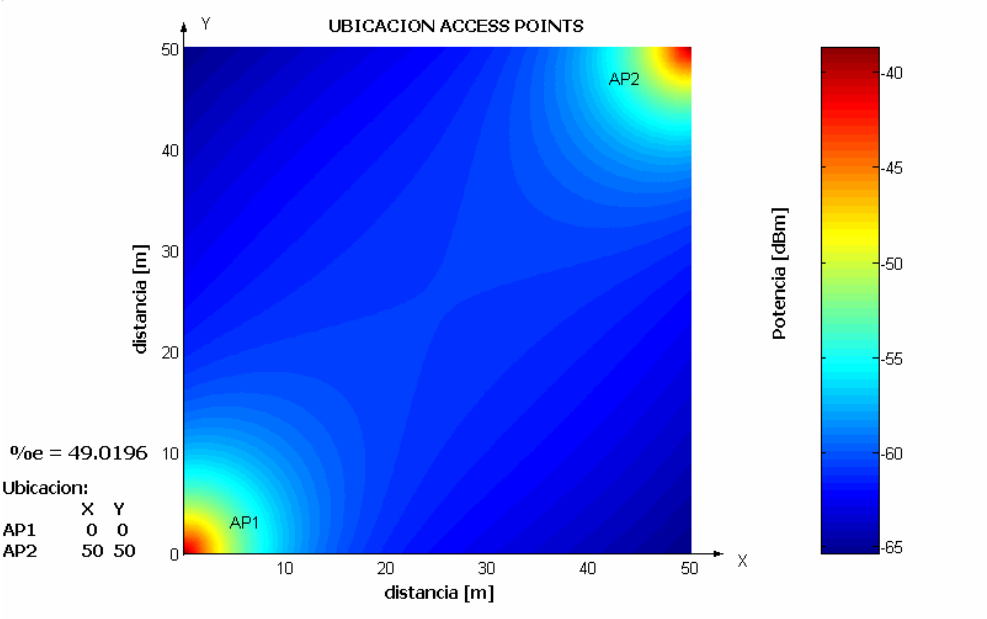
Figura 4. Configuración extremo - centro



Fuente: Autores del Proyecto

La segunda distribución, con un AP, fue uno en un extremo y otro en la diagonal, generó un porcentaje de puntos con igual potencia del 49%. En la figura 5 se puede ver la distribución de potencias.

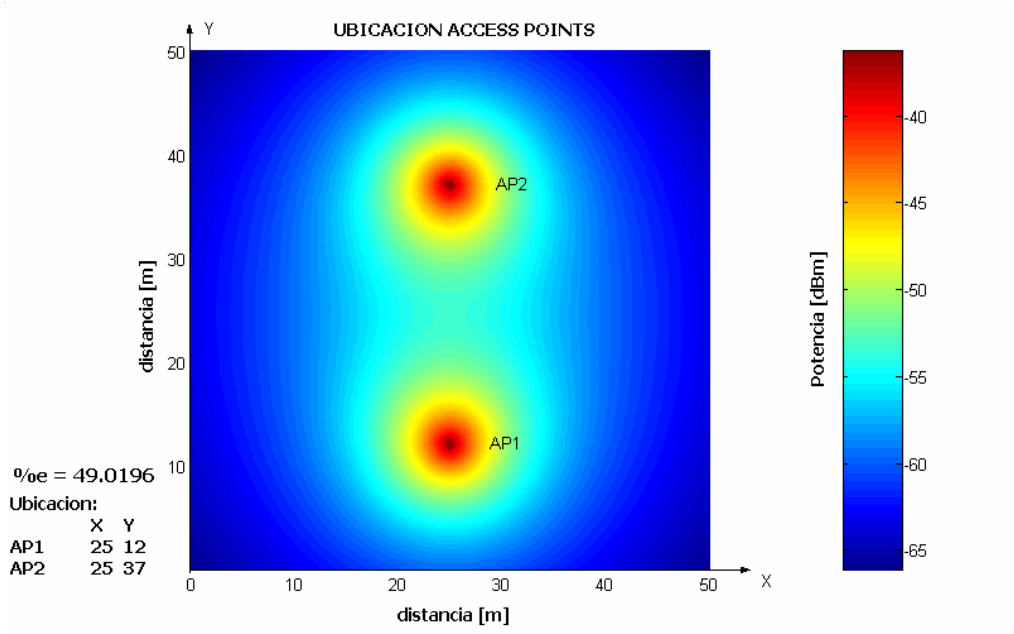
Figura 5. Configuración extremo - extremo diagonal



Fuente: Autores del Proyecto

Otra de las distribuciones se escogió con un AP frente al otro sobre la línea que divide la cuadrícula por la mitad, con un porcentaje de combinación de potencias iguales del 49%. La figura 6 muestra la distribución de potencias de los AP para esta ubicación.

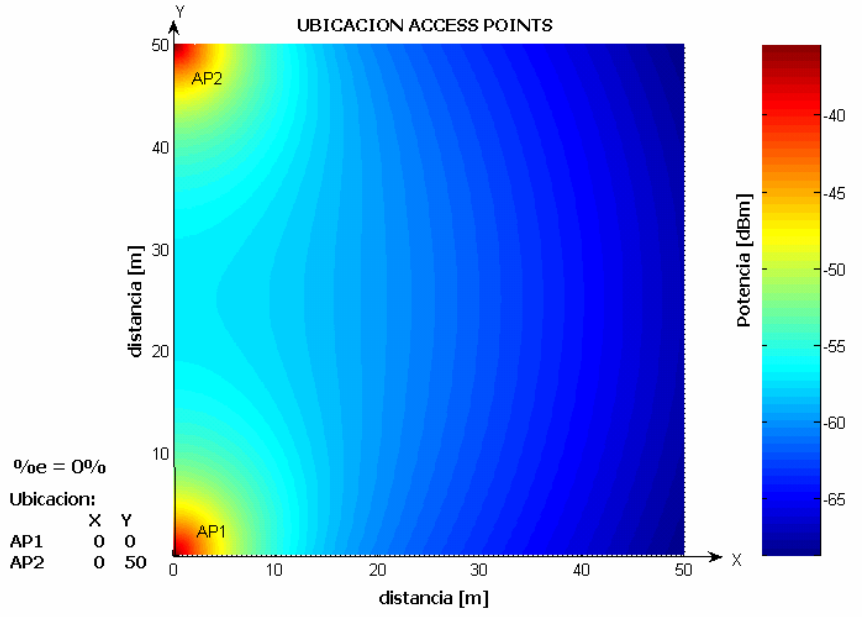
Figura 6. Configuración centro - centro



Fuente: Autores del Proyecto

La siguiente configuración consiste en uno frente al otro pero en los extremos de la cuadrícula. Esta configuración conduce a un porcentaje del 0%, es decir, no existen dos puntos dentro de la grilla con la misma combinación de potencias. La figura 7 muestra la distribución de potencias para este caso.

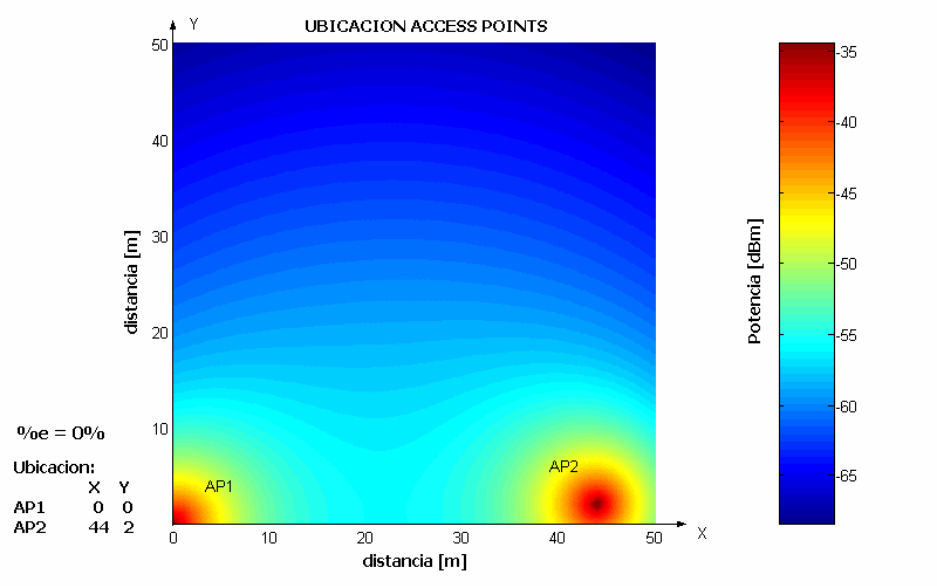
Figura 7. Configuración extremo - extremo



Fuente: Autores del Proyecto

Como última distribución se seleccionó una que no tuviera ningún tipo de simetría arrojando de igual manera un porcentaje del 0%. Esta distribución se puede apreciar en la figura 8.

Figura 8. Configuración asimétrica



Fuente: Autores del Proyecto

Con base en un análisis de simetría, se pude concluir que para efectos de lograr una adecuada ubicación de los *APs* es necesario evitar simetría con más de un eje en su ubicación respecto al mapa; esto permite tener un mapa de potencias que no presente alguna combinación de potencias igual para puntos diferentes dentro de la grilla. No obstante, esta observación se hizo teniendo como base las cinco configuraciones descritas. Para hacer más concreta esta afirmación, se requiere un análisis más detallado haciendo uso de otras configuraciones.

La configuración escogida para la distribución de potencias dentro del mapa fue la mostrada en la figura 7, ubicando un punto de acceso frente al otro en un extremo de la cuadrícula ya que ésta no presenta combinaciones de puntos con la misma distribución de potencias.

2.2 SOLUCIÓN PROPUESTA

Una vez escogida la ubicación de los puntos de acceso se procede a la construcción del mapa teórico de intensidades de señal para luego pasar a la elaboración de posibles métodos que permitan obtener la ubicación del dispositivo móvil al interior de la red.

Se consideraron tres métodos para dar solución teórica al problema planteado, que son:

- Solución basada en la ecuación de pérdidas en espacio libre
- Solución mediante redes neuronales
- Solución mediante algoritmo de búsqueda

2.2.1 Solución basada en la ecuación de pérdidas en espacio libre. Una de las metodologías utilizadas en la solución del problema de determinar la ubicación fue mediante un programa[10] en *Matlab* que resuelve la ecuación de pérdidas en espacio libre. La aplicación recibe la información de potencia de la señal y calcula el punto de ubicación dentro del escenario expresado en coordenadas X e Y.

2.2.2 Solución mediante redes neuronales. Un segundo método empleado en la determinación de la posición fue la utilización de redes neuronales artificiales, con las cuales se espera tener una incertidumbre no mayor a 10m del lugar donde se encuentra el dispositivo inalámbrico.

Esta meta se establece teniendo como referencia el trabajo realizado por Johnny Shih [Shih, 2003], donde con una infraestructura de seis APs se consiguió una incertidumbre de aproximadamente 4 m.

[10] Ver anexo D

- **Diseño de la red neuronal.** Como entrada a la red neuronal se tiene una combinación de potencias obtenidas a partir de la ecuación de pérdidas en espacio libre, que representarán las potencias recibidas por la tarjeta de cada uno de los puntos de acceso; como salida de la red se tendrán las coordenadas X e Y dentro del perímetro escogido. El diseño del modelo neuronal se describirá detalladamente a continuación.

Uno de los aspectos fundamentales de las redes neuronales artificiales es su capacidad de generalizar a partir de ejemplos, lo que constituye el problema de la memorización frente a la generalización, entendiéndose por generalización a la capacidad de la red a dar una respuesta correcta ante patrones que no han sido empleados en su entrenamiento.

Al iniciar el entrenamiento la red se adapta progresivamente al conjunto de aprendizaje, acomodándose al problema y mejorando la generalización. Sin embargo, en un momento dado el sistema se ajusta demasiado a las particularidades de los patrones empleados en el entrenamiento, aprendiendo incluso el ruido presente en ellos, por lo que crece el error que comete ante patrones diferentes a los empleados en el entrenamiento. En este momento la red no se ajusta correctamente al *mapping*, sino que simplemente está memorizando los patrones del conjunto de aprendizaje, esto técnicamente se denomina sobreaprendizaje o sobreajuste [Del Brío, Molina, 2002] . Idealmente, dada una arquitectura de red neuronal, esta debería entrenarse hasta un punto óptimo en el que el error de generalización sea mínimo.

En definitiva, la capacidad de generalización y aprendizaje de la red la determinan en buena medida las siguientes características:

- Normalización de los datos
- Error de entrenamiento
- El número de ejemplos de entrenamiento

- La complejidad del problema
- La arquitectura de la red

Estos parámetros están muy relacionados y definen en buena medida el diseño de la red neuronal; en términos generales, cuanto más complejo sea el problema a modelar, más grande deberá ser la red y, por lo tanto, más ejemplos se necesitarán para entrenarla. Una normalización adecuada de los datos de entrenamiento evita la saturación temprana en la capa de entrada. Existen varias formas de luchar contra el fenómeno de sobreaprendizaje; por ejemplo la parada temprana o limitar el tamaño de la arquitectura de la red.

- **Normalización de los datos.** Las variables de entrada y salida de la red son valores continuos de potencia y posición respectivamente. Para el entrenamiento de la red y como parte del procesamiento de los datos, las entradas y salidas han sido normalizadas entre [-1; 1] que son los rangos manejados por las funciones de activación, evitando así una saturación prematura en la capa de entrada. Debido a que los rangos de variación de las entradas y salidas son conocidos, y después de probar varios tipos de normalización se escogió la función de *Matlab* denominada *prestd* que, preprocesa los datos para que la media sea cero y su desviación estándar uno, llevando a que un alto porcentaje de los datos se encuentren en el intervalo [-1, 1] aplicando la ecuación (4):

$$Pn = \frac{P - mean(P)}{std(P)} \quad (4)$$

Pn: Conjunto de patrones de entrada, o salida, normalizado

P : Conjunto de patrones de entrada, o salida, a normalizar

$Mean$: Función de *Matlab* que devuelve el valor medio del conjunto de datos que se le suministre.

Std : Función de *Matlab* que devuelve la desviación estándar del conjunto de datos que se le suministre.

- **Error de entrenamiento.** Uno de los métodos más utilizados para una parada temprana en el proceso de entrenamiento es la definición del error de aprendizaje, que suele calcularse como el error cuadrático medio (MSE) de los resultados proporcionados por la red para el conjunto de patrones de aprendizaje.

La meta del error de entrenamiento se calcula como la sumatoria de la diferencia máxima al cuadrado entre los resultados obtenidos y los esperados, promediado sobre todas las salidas y todos los patrones de entrada para el aprendizaje.

$$MSE = \frac{1}{NQ} \sum_{n,q-1}^{N,Q} (T_n^q - Y_n^q)^2 \quad (5)$$

T_n^q : Valores deseados de la n-sima salida de la red cuando se presenta el patrón q-simo en la entrada.

Y_n^q : Valores obtenidos de la n-sima salida de la red cuando se presenta en la entrada el patrón q-simo.

Q : Número de patrones en la entrada.

N : Número de salidas.

Como el objeto de la red neuronal es el de predecir la posición del dispositivo inalámbrico a partir de las potencias emitidas por los puntos de acceso, podríamos esperar como diferencia máxima aceptable entre las salidas esperadas y las obtenidas, una incertidumbre de 10m en la predicción de la coordenada.

A partir de esto, y según un trabajo realizado previamente en el grupo de investigación CPS[11] [Guzmán, 2005], se pueden definir dos cotas para la meta del error de entrenamiento, la primera es la más exigente y una segunda que será más flexible.

La primera cota para el error corresponde a:

$$MSE \geq \frac{(T_n^q - Y_n^q)^2_{\max}}{NQ} \geq \frac{dif^{\,2}_{\max}}{NQ} \quad (6)$$

$dif_{\max}$: Diferencia máxima.

La segunda se puede definir como sigue:

$$MSE \leq (T_n^q - Y_n^q)^2_{\max} \leq dif^{\,2}_{\max} \quad (7)$$

En la tabla 3 se muestran diferentes cotas para el error de aprendizaje obtenidas a partir de errores que resultarían aceptables en la predicción de la coordenada.

Tabla 3. Cotas para definir el error de aprendizaje

Desviación en la Predicción (m)	**Error de aprendizaje para le red**	
	Cota inferior	**Cota superior**
10	0.1949	0.3897
5	0.0487	0.0974
2.5	0.01218	0.0243
1.25	0.0030	0.0061
0.5	4.8716e-4	9.7432e-4

Fuente: Autores del Proyecto

[11] Grupo de Investigación en Conectividad y Procesado de Señal

- **Número de ejemplos de entrenamiento.** Usualmente, de todo el conjunto de entrenamiento se emplea aproximadamente un 80% de los patrones para entrenar, reservándose un 20% como conjunto de prueba [Del Brío, Molina, 2002] .

Los datos obtenidos en la campaña de medición constituyen la base para generar el conjunto de patrones de entrenamiento y verificación que requiere una red neuronal para su puesta a punto.

En este caso se utilizó el 78% de los datos para el entrenamiento, dejando el 22% de estos para probar el desempeño de la red.

- **Complejidad del problema.** Debido al tipo de problema y para reducir la complejidad de la red neuronal y el número de patrones necesarios para entrenarla adecuadamente, se separará el problema en dos modelos neuronales que permitan predecir independientemente cada una de las coordenadas del punto de ubicación del dispositivo móvil al interior del escenario.

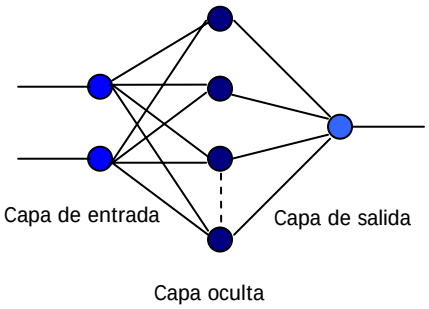
El tipo de red utilizado es el Perceptrón multicapa y el algoritmo de aprendizaje empleado es el denominado de retropropagación de errores (*backpropagation*) o *BP*.

Se escogió el Perceptrón Multicapa debido a que este tipo de red puede aprender tareas complejas y su alto grado de conectividad le permite extraer progresivamente las características que presentan los patrones de entrada, que para nuestro caso son los valores de intensidad de señal emitidas por los puntos de acceso. Además, son redes bastante robustas en aplicaciones de predicción, lo que las convierte en una herramienta adecuada para dar una posible solución al problema de localización del dispositivo al interior de la WLAN. Como regla de aprendizaje se utilizó el

algoritmo *Backpropagation* que es muy empleado en neuronas que poseen funciones de activación no lineales y diferenciables, como es el caso de la Perceptrón multicapa.

- **Arquitectura de la red.** Para este caso se escogió una arquitectura de tres capas; la capa de entrada formada por dos (2) neuronas, la capada salida con una (1) neurona y la capa oculta compuesta por dieciséis (16) neuronas. La selección de la arquitectura de la red se expondrá a continuación.

Figura 9. Diagrama de la red neuronal



Fuente: Autores del Proyecto

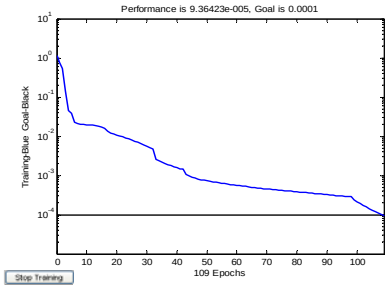
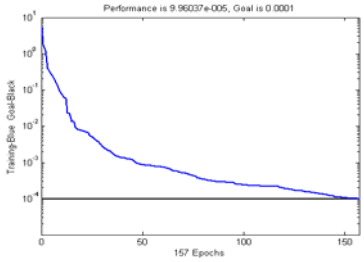
- **Parámetros seleccionados para el entrenamiento [Hurtado, Reyes, 2001]**

o **Número de capas**: Debido a la forma en que se auto-organiza la información a través de las redes multicapa, estas pueden incurrir en un problema de memorización a medida que aumenta el número de capas ocultas. Este problema se analiza entrenando dos redes con la misma salida y variando el número de capas ocultas.

Se puede observar que aumentar excesivamente el número de capas

ocultas puede generar un aumento exagerado en el tiempo empleado para el entrenamiento sin que haya una reducción apreciable en el error; por esto se escogió una arquitectura de una sola capa oculta.

Tabla 4. Comparación entre una red 2-16-16-16-16-1 y una 2-16-1

	Red A	**Red B**
Arquitectura	2-16-16-16-16-1	2-16-1
Tiempo de Entrenamiento	2 minutos	35 segundos
Desviación absoluta promedio de predicción	0.1707m	0.1754m
Desviación máxima de predicción	1.0452m	0.7625m
Curvas de entrenamiento		
Red A	**Red B**	
		
Parámetros comunes		
Número máximo de épocas	1000	
Error de entrenamiento	5e-4	
Rata de aprendizaje	0.05	
Mín. rendimiento del gradiente	1e-30	
Función de activación entrada y capa oculta	tansig	
Función de activación capa de salida	purelin	

- **Número de neuronas por capa**: Después de comprobar que establecer una red con una capa oculta permite predecir la posición sin incurrir en problemas de memorización, se debe establecer cuál es el número adecuado de neuronas para la capa oculta. Escoger una arquitectura con pocas neuronas en la capa oculta puede generar que la red no entrene adecuadamente, mientras que un aumento excesivo en el número de neuronas de esta capa puede aumentar enormemente el tiempo de entrenamiento y caer en problemas de memorización. Finalmente el número de neuronas escogido para la capa oculta se fijó en dieciséis (16), el cual permite una buena generalización del comportamiento de la salida ante las potencias de entrada a la red. En la tabla 5 se puede observar la repercusión que tiene la escogencia de pocas ó demasiadas neuronas en la capa oculta para entrenamiento de una red neuronal.

Tabla 5. Comparación entre redes con diferente número de neuronas en la capa oculta

	Red A	**Red B**	**Red c**
Arquitectura	2-4-1	2-16-1	2-30-1
Tiempo de Entrenamiento	30 segundos	35 segundos	30 segundos
Desviación absoluta promedio de predicción	0.2509m	0.1754m	0.5691m
Desviación máxima de predicción	1.0380m	0.7625m	6.9300m
Parámetros comunes			
Número máximo de épocas	1000		
Error de entrenamiento	5e-4		

Rata de aprendizaje	0.05
Mín. rendimiento del gradiente	1e-30
Función de activación entrada y capa oculta	Tansig
Función de activación capa de salida	Purelin

Fuente: Autores del Proyecto

- **Función de activación**: Este tipo de red requiere una función de activación continua, que defina la salida de las neuronas en función del nivel de actividad de sus entradas; principalmente existen dos funciones que pueden ser empleadas para tal efecto: sigmoidal y lineal. Para las capas de entrada y oculta se tomó como función de activación una sigmoidal; debido a que la derivada de esta función existe en todo el intervalo, permitiendo que las reglas de aprendizaje se puedan implementar satisfactoriamente. A la capa de salida se asignó una función lineal con el fin de que la salida tuviera un rango libre de variación.

Matlab permite implementar las funciones sigmoidal y lineal pura de la siguiente manera:

$$\tan sig(v) = \frac{2}{1+e^{-2v}} - 1 \quad (8)$$

El rango de variación de esta función está entre [-1,1], lo cual es adecuado para el tipo de normalización aplicada a los datos para el entrenamiento.

La lineal pura se representa como sigue:

$$purelin = v \quad (9)$$

En definitiva, la red neuronal se diseñó teniendo como parámetros de entrada los valores obtenidos de la ecuación de pérdidas en espacio libre que permite modelar el comportamiento de los puntos de acceso en ambientes exteriores. Los parámetros seleccionados para el entrenamiento del modelo neuronal con sus respectivos valores se enuncian en la tabla 6.

Tabla 6. Tabla de parámetros de las redes

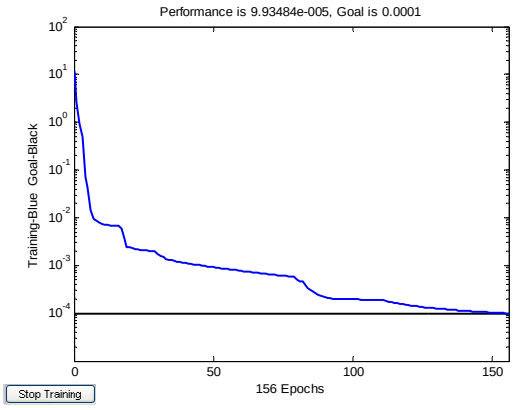
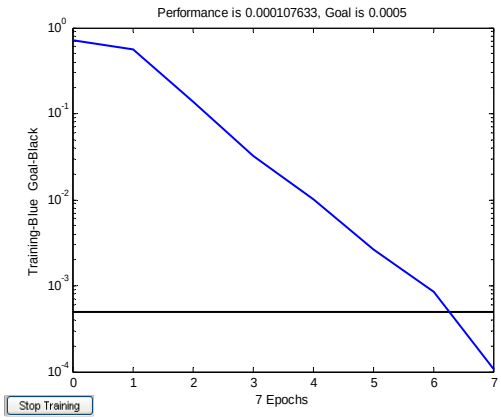
Parámetros para el entrenamiento	
Arquitectura	2-16-1
Tipo de red	Perceptron Multicapa
Algoritmo de entrenamiento	Levenberg-Marquardt (trainlm)
Función de activación entrada y capa oculta	Tansig
Función de activación capa de salida	Purelin
Error de entrenamiento	5e-4
Número máximo de épocas	1000
Rata de aprendizaje	0.05
Mín. rendimiento del gradiente	1e-30

Fuente: Autores del Proyecto

2.2.3 Resultados del entrenamiento. Como se mencionó anteriormente, la red se entrenó con un número específico de parámetros que fueron mostrados en la tabla 6. El la tabla 7 se presentan algunos resultados que pretenden mostrar en cierta forma la eficiencia del diseño de la red. Es importante mencionar que estos resultados están basados en patrones de entrada que fueron tomados de la ecuación de pérdidas en espacio libre; es decir, es un diseño teórico que habrá que ajustar de

acuerdo con los datos que se obtengan en las mediciones de campo. El diseño con los datos experimentales junto con sus resultados se presentarán en el siguiente capítulo.

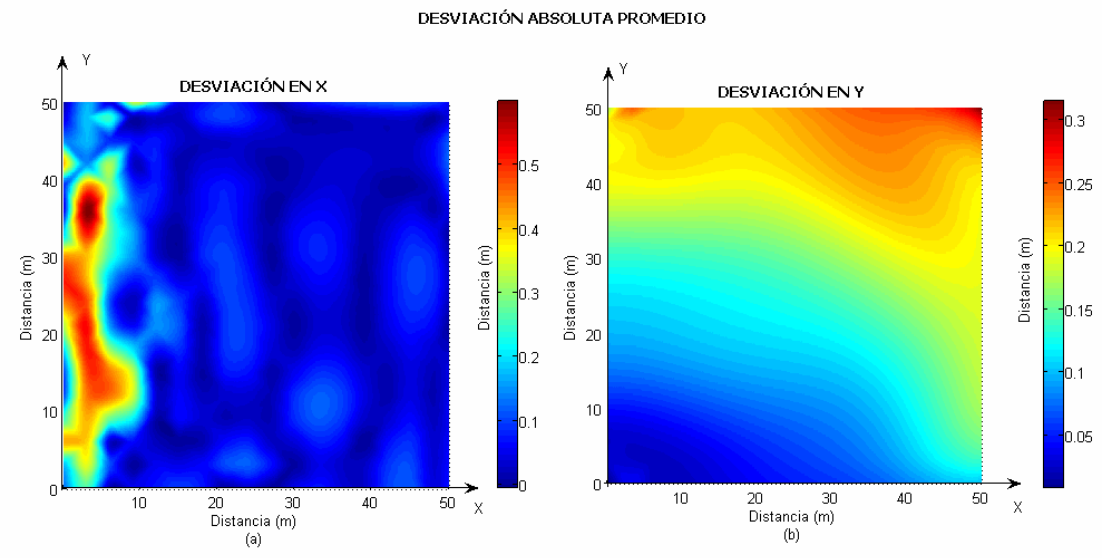
Tabla 7. Resultados del entrenamiento datos teóricos

Red Datos Teóricos					
Coordenada	**MSE alcanzado**	**Número de épocas**	**Desviación de Predicción (m.)**		**Desviación absoluta promedio**
			Mínimo	**Máximo**	
X	9.9897e-5	367	0.0047	0.3442	0.0822m
Y	10.7633e-5	7	0.0016	0.1631	0.0874m
Curvas de entrenamiento					
Coordenada X			**Coordenada Y**		
					

Fuente: Autores del Proyecto

La figura 10 ilustra cuales zonas de la grilla presentan una mayor incertidumbre en la determinación de la coordenada. En la figura 10 (a) y (b) se observa la desviación absoluta en la predicción de la coordenada X e Y respectivamente, dentro de la grilla.

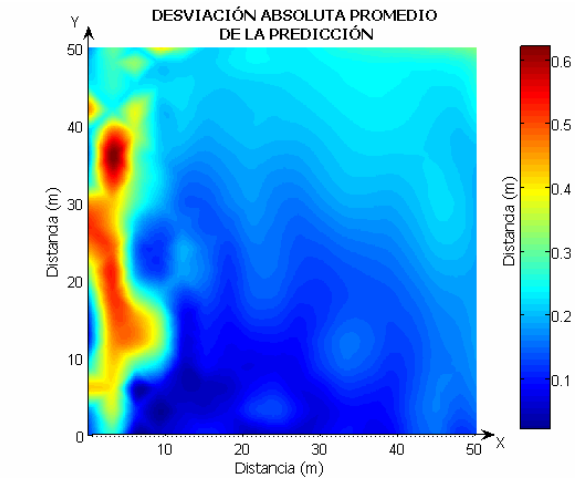
Figura 10. Desviación en la predicción



Fuente: Autores del Proyecto

La figura 11 muestra de igual manera la incertidumbre de la posición del dispositivo, representando qué tan lejos se encuentra el resultado presentado por la red del punto real.

Figura 11. Desviación absoluta



Fuente: Autores del Proyecto

2.2.4 Solución mediante algoritmo de búsqueda. Una alternativa adicional es dar una solución al problema mediante un algoritmo de búsqueda, que reciba un par de potencias y devuelva una zona donde esté ubicado el dispositivo que se quiere localizar.

La operación del algoritmo de búsqueda se muestra en el diagrama de bloques de la figura 12.

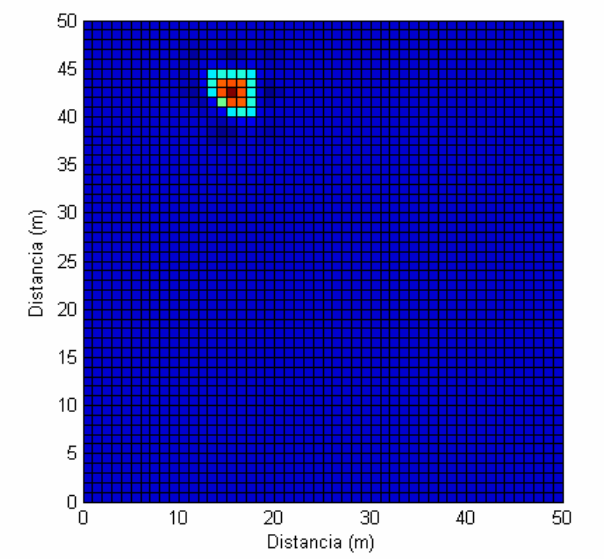
Figura 12. Diagrama de bloques del algoritmo de búsqueda

Fuente: Autores del Proyecto

Como está ilustrado en el anterior diagrama de flujo, se tienen las potencias de entrada y dos matrices que son con la cuales se hace la comparación para la búsqueda. A continuación, se hacen varios procedimientos de descarte insertando unos (1) en aquellos lugares de las matrices donde se encuentre coincidencia con el valor de potencias, y ceros donde haya valores diferentes, para luego superponer las matrices y encontrar una zona única de localización.

En la figura 13 se muestra el resultado que se obtiene al introducir las potencias -60 y -52, del AP1 y el AP2, respectivamente y que está ubicado en el punto (17,43) de la grilla.

Figura 13. Resultado ejemplo del algoritmo de búsqueda



Fuente: Autores del Proyecto

En el ejemplo mostrado, se observa que el algoritmo de búsqueda muestra una zona de forma irregular alrededor del punto de ubicación real del punto al que se está haciendo referencia. Es decir, que este algoritmo es preciso en su operación y entrega un resultado confiable.

3. CAMPAÑAS DE MEDICIÓN

La selección de un escenario propicio, que no introduzca una cantidad significativa de interferencia y el tener un método indicado para realizar las mediciones son aspectos fundamentales que permitirán una adecuada caracterización de la propagación de las señales, los cuales se expondrán en las dos primeras secciones de este capítulo. En la siguiente sección, se hará un análisis comparativo entre los mapas de potencias teórico y experimental. En las secciones posteriores se desarrollarán técnicas que pretenden dar una solución real al sistema de localización, que estará implementado en el mismo cliente.

3.1 SELECCIÓN DEL ESCENARIO

En el presente trabajo de grado, como se planteó en los objetivos, se pretende localizar un dispositivo móvil al interior de una red de área local inalámbrica en un ambiente exterior (*outdoor*).

Debido a esto, se ha seleccionado un escenario que presente ciertas características, las cuales permitan una adecuada propagación de las señales emitidas por los *APs*. Estas características son un espacio amplio, con un mínimo de elementos de interferencia tales como rejas, paredes, postes de alumbrado público, líneas de transmisión de potencia, entre otras; en otras de estas características se encuentran lo plano del terreno y, por supuesto, el ser un espacio libre.

El escenario designado es el estadio de fútbol 1º de Marzo de la UIS. Este escenario es escogido por reunir la mayor parte de las características antes mencionadas. El estadio se encuentra ubicado en la parte centro-oriental del campus universitario, a espaldas del auditorio Luis A. Calvo y del edificio de administración. A la parte oriente se encuentran los escenarios deportivos múltiples, al sur la cancha de fútbol sur (arena) y las canchas de tenis y al norte espacio descubierto.

Figura 14. Estadio 1° de Marzo y sus alrededores

Fuente: Autores del Proyecto

Así como características favorables, el escenario cuenta también con características que pueden afectar las mediciones y posteriores pruebas. Se encuentran entre éstas los arcos de fútbol, la irregularidad del terreno en algunas partes, el alambrado que está a su alrededor y la humedad. Puesto que es un escenario deportivo abierto, también es posible que las

personas que se encuentren realizando actividades deportivas allí, afecten la propagación.

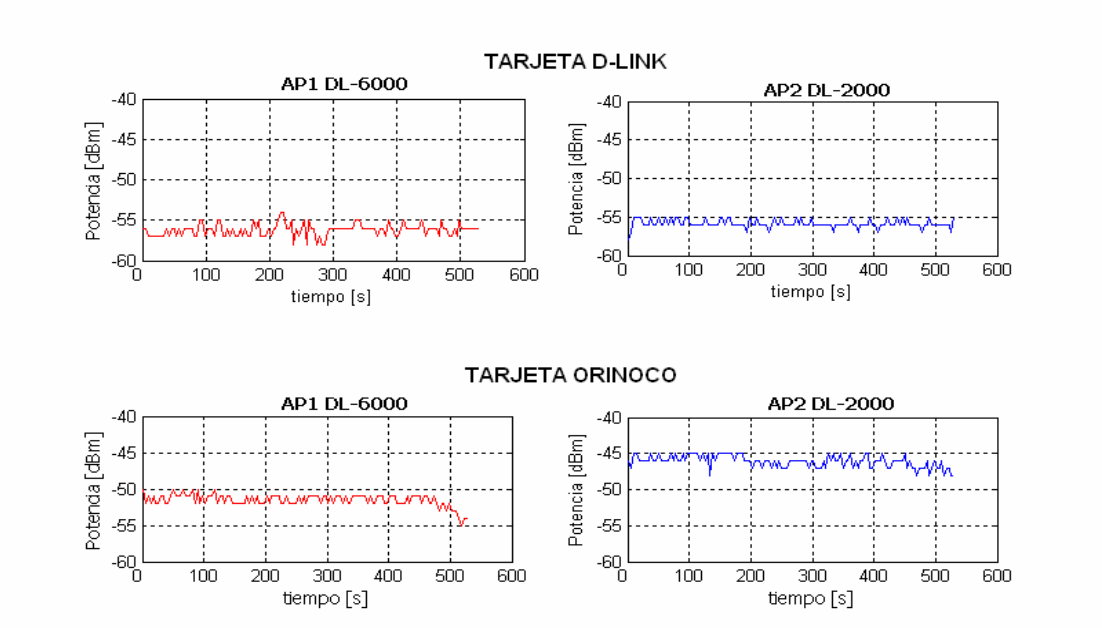
3.2 TOMA DE MEDIDAS

3.2.1 Toma de medidas en un solo punto. Para encontrar características de las tarjetas y los puntos de acceso, se realizaron una serie de experimentos que consistieron en la toma de mediciones de potencia en un solo punto con los cuales se hallaron medidas estadísticas y se hicieron observaciones de los cambios con respecto al tiempo.

- **Toma de medidas en un solo punto en el laboratorio.** El objetivo de este experimento es probar propiedades de diferentes tarjetas para hacer una adecuada selección del equipo con el cual se hará la campaña de medición. La escuela cuenta con tarjetas inalámbricas de diferentes fabricantes por lo cual se debe buscar cual de éstas presenta características óptimas para el desarrollo del proyecto, de acuerdo con ciertos parámetros como lo son estabilidad y nivel de sensibilidad. Este experimento fue realizado en el Laboratorio de Redes del edificio de Eléctrica antigua, salón 201. Las mediciones fueron tomadas durante diez (10) minutos con un lapso entre cada registro de aproximadamente dos (2) segundos.

La figura 15 muestra el comportamiento en el tiempo de las tarjetas para cada uno de los puntos de acceso. De esta gráfica se puede observar que el punto de acceso DWL-6000 presenta una mayor estabilidad con la tarjeta ORINOCO que con la D-LINK; mientras que, el punto de acceso DWL-2000 muestra un comportamiento semejante con las dos tarjetas.

Figura 15. Comportamiento de las tarjetas D-LINK y ORINOCO.



Fuente: Autores del Proyecto

La tabla 8 pretende mostrar el comportamiento estadístico de las tarjetas D-LINK y ORINOCO. En esta se observa que la tarjeta D-LINK muestra una menor desviación que la ORINOCO pero, la tarjeta ORINOCO registra mayor potencia que la D-LINK.

Tabla 8. Datos estadísticos de las tarjetas D-LINK y ORINOCO.

Access Points	Tarjeta	Media	Desviación Estándar	Coeficiente Variación	mín.	máx.
DWL-6000	Orinoco	-51.5564	0.8295	62.1530	-55	-50
	D-Link	-56.1955	0.7732	72.6779	-58	-54
DWL-2000	Orinoco	-46.0752	0.8130	56.6730	-48	-45
	D-Link	-55.8346	0.5530	100.9741	-58	-55

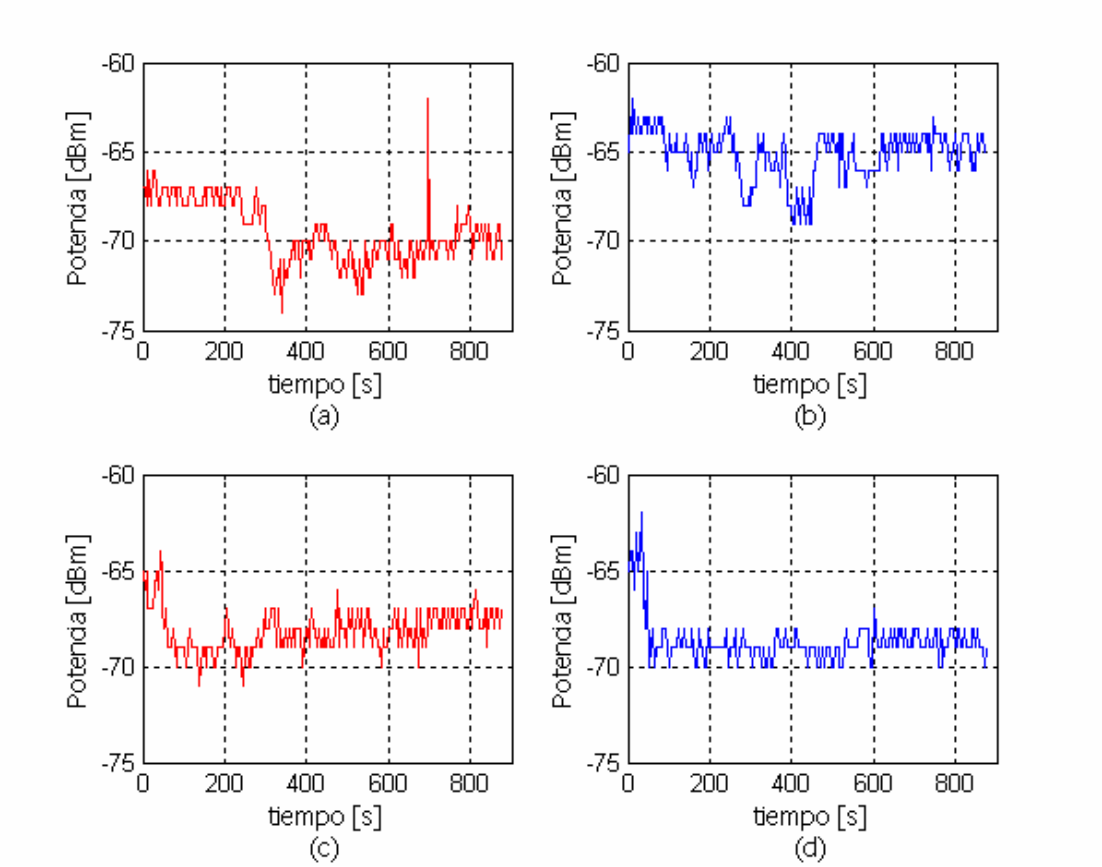
Fuente: Autores del Proyecto

Del análisis realizado a las gráficas de la figura 15 y los datos presentados en la tabla 8 se puede concluir que es más adecuado utilizar la tarjeta ORINOCO para la construcción del mapa de potencias, debido a que ésta recibe mayor potencia, y en cuanto a su desviación es comparable con la presentada por la D-LINK; además, la diferencia entre los valores de potencia máximo y mínimo recibidos por cada una de las tarjetas está en el mismo orden.

- **Toma de medidas en un solo punto en el escenario.** Las cuatro gráficas de la figura 16, muestran los comportamientos observados al realizar mediciones de potencia en el estadio de fútbol 1° de Marzo de la UIS. Estas mediciones registradas por la tarjeta inalámbrica ubicada en el punto central del área durante 15 minutos, haciendo registros cada 2 segundos, aproximadamente.

En las figuras 16 (a) y (b), se encuentran registradas las mediciones para cada uno de los *APs* cuando se encuentran operando solos; esto es, que uno de los dos se encuentra encendido y el otro no. Para las figuras 16 (c) y (d), se pusieron en operación los dos *APs* simultáneamente.

Figura 16. Comportamiento en el tiempo de la potencia recibida por la tarjeta



Fuente: Autores del Proyecto

Al contrario de lo esperado, las gráficas de operación simultánea muestran a partir del centésimo segundo, aproximadamente, una estabilidad de los posibles valores de potencia para ese punto de la grilla. Esto quiere decir, que los valores varían en un rango de seis valores más o menos. En cambio, para la operación de cada uno de los *APs* solos, la estabilidad es muy poca, y los valores varían entre un número mayor de posibilidades.

Las medidas estadísticas que se obtuvieron con este experimento se presentan en la tabla 9.

Tabla 9. Medidas estadísticas del experimento en el punto (24,24)

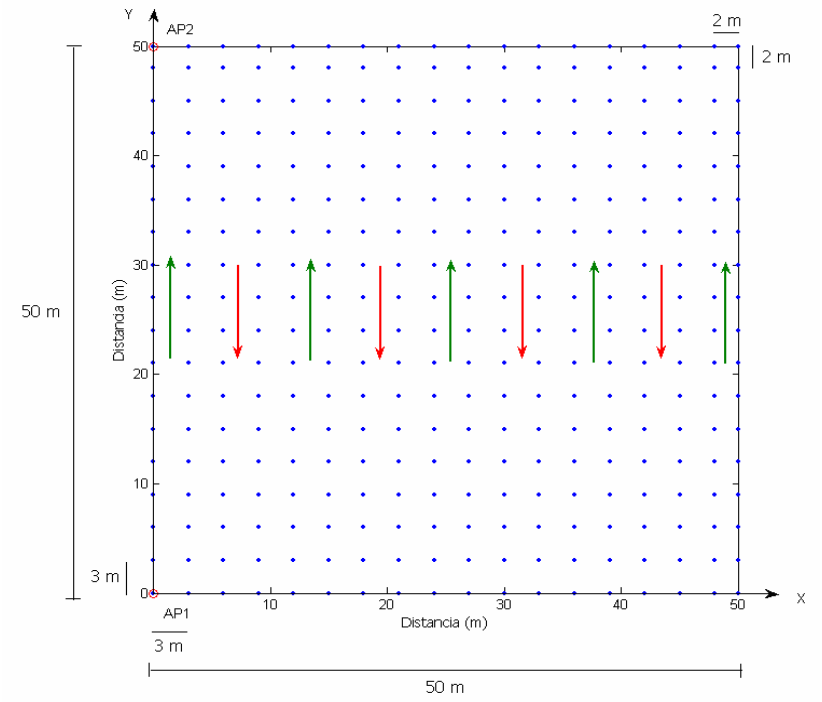
Medida estadística punto de acceso	Desviación estándar	Media	Coeficiente de variación
AP1	1.7706	69.4545	39.2273
AP2	1.4137	65.1227	46.0652
AP1*	1.0978	68.1364	62.0689
AP2*	1.2089	68.6500	58.7876

Fuente: Autores del Proyecto

3.2.2 Construcción del mapa de potencias. Las mediciones se realizaron con el fin de construir un mapa de potencias (grilla) con una separación de 3 metros entre cada punto. Debido a que la superficie es de 50 x 50 m^2, la medición correspondiente a la última línea de puntos solamente posee una separación de 2 metros con respecto a la línea de puntos que la precede. En la figura 17 se muestran los puntos a los cuales se miden las potencias y la secuencia de barrido con que fueron efectuadas las mediciones.

* Operación simultánea de los *APs*

Figura 17. Puntos de medición y secuencia de barrido

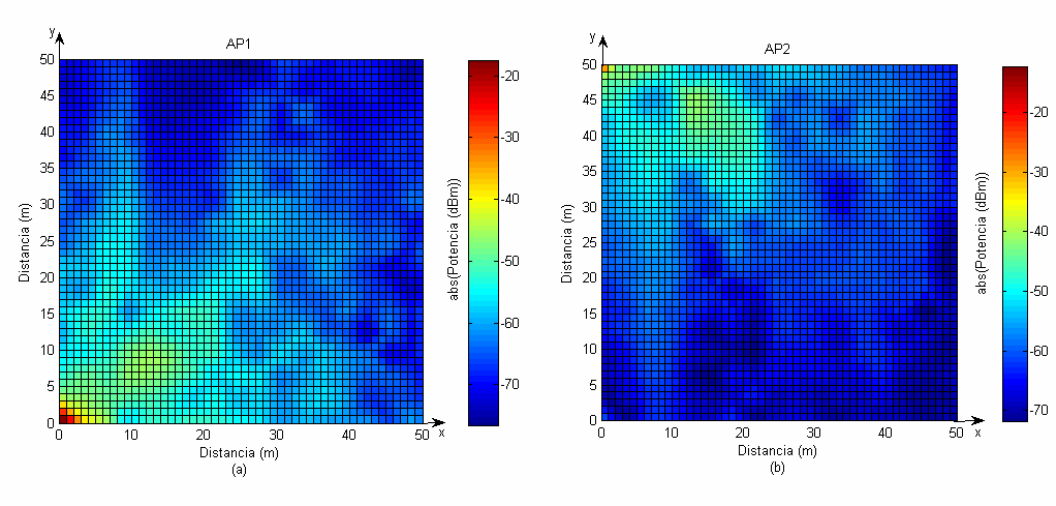


Fuente: Autores del Proyecto

Por cada punto, se hicieron 5 mediciones cada 10 segundos para luego encontrar la media entre ellos y caracterizar el punto con el valor obtenido.

Después de las mediciones realizadas se obtuvieron las distribuciones de potencia de cada uno de los AP que se muestran el las figuras 18 (a) y (b).

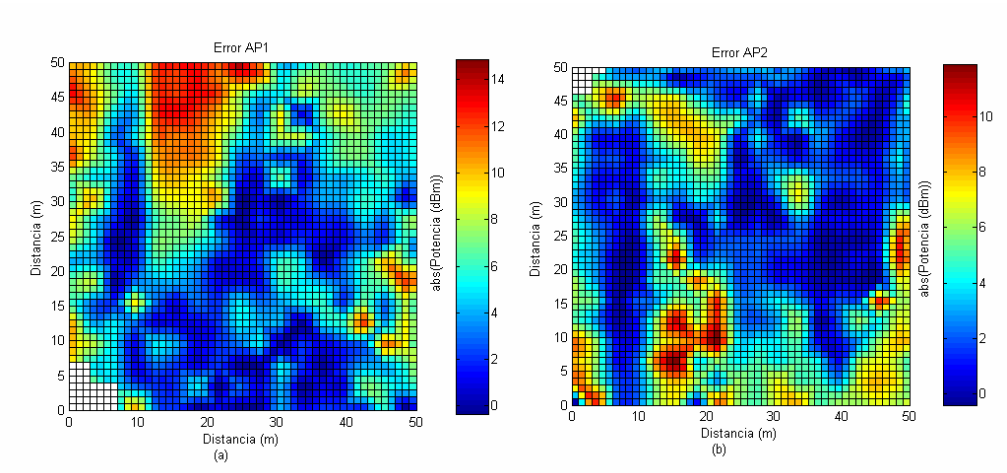
Figura 18. Distribuciones de potencia según mediciones



Fuente: Autores del Proyecto

Al hacer la comparación entre el mapa obtenido como producto de las mediciones y el mapa construido en el laboratorio a partir de la ecuación de *Friis*, se obtienen las gráficas correspondientes a los errores obtenidos y se muestran en las figuras 19 (a) y (b).

Figura 19. Errores entre ecuación de *Friis* y mediciones



Fuente: Autores del Proyecto

Las partes de las gráficas donde predomina el color rojo, son aquellas que tienen mayor diferencia con respecto a la ecuación de *Friis*, y las que tienen un color azul fuerte son aquellas con menor error. Las partes donde hay un manchón blanco, son aquellas partes donde algún argumento de los logaritmos de la ecuación de *Friis* (Ec. 3) es cero; por lo tanto, es un valor indeterminado.

3.3 VERIFICACIÓN DEL DISEÑO REALIZADO EN LABORATORIO

Para la verificación del diseño se han desarrollado dos técnicas a partir de los datos obtenidos en las campañas de medición para la búsqueda de una solución real al problema de la localización en la WLAN determinada por los objetivos. La primera técnica es una red neuronal que aproxime de buena manera la posición que se desea conocer del dispositivo móvil. La segunda de ellas, es un algoritmo de búsqueda que, dependiendo de algunos parámetros, resuelva dicha posición, ya sea mostrando una posible zona de ubicación o un punto de coordenadas

3.3.1 Red neuronal. La predicción de las coordenadas se hará por medio de dos redes neuronales, cuyos patrones de entrada serán las potencias emitidas por los puntos de acceso. Cada red neuronal determinará una coordenada. Por tanto, para cada red se tendrán dos entradas y una salida. No existe ninguna metodología estricta que asegure el éxito de un diseño de esta naturaleza, ni tampoco un único modelo que satisfaga los requerimientos del proyecto.

El método de predicción requiere que los datos sean presentados en forma de ejemplos que registren el comportamiento de las coordenadas ante los patrones de entrada. Esta información de patrones experimentales de entrada-salida para el entrenamiento de la red

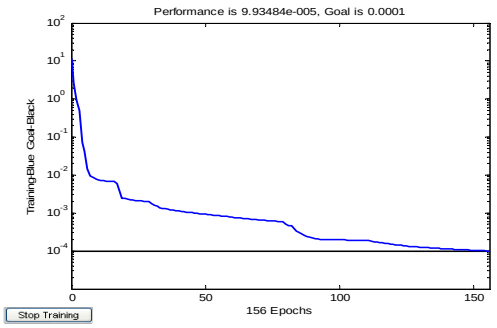
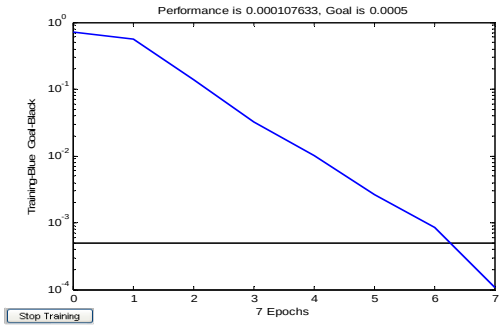
neuronal fueron recopilados de la campaña de medición, donde las entradas son los valores de potencia emitidos por los puntos de acceso que fueron registrados por medio de la herramienta *Network Stumbler*[12] y las salidas son las coordenadas X e Y del punto donde se tomó cada registro de potencia.

Como se describió en el capítulo dos (2) la capacidad de aprendizaje de un modelo neuronal está determinada en buena medida por la selección adecuada de los parámetros de entrenamiento. Los parámetros seleccionados como parte del diseño de la red, junto con los resultados obtenidos se muestran en la tabla 10. Este diseño fue realizado siguiendo la metodología descrita en el capítulo anterior.

En la tabla 10, se puede observar que la red neuronal tiene problemas para determinar con precisión la posición del dispositivo al interior de la grilla. A pesar de que la red converge rápidamente y en pocas iteraciones al error de aprendizaje determinado en el diseño, presenta una desviación absoluta en la predicción de la coordenada bastante elevada, por el orden de 19m. Es decir la fidelidad del modelo neuronal no es la esperada. Alrededor del 20% de los datos escogidos para la simulación de la red presentaron una desviación absoluta en la predicción superior a 10m.

[12] Ver anexo C

Tabla 10. Parámetros y resultados experimentales de la red.

Red Datos Experimentales						
Coord.	**Error de entrena-miento**	**MSE alcanzado**	**Número de épocas**	**Desviación de la predicción (m)**		
				Mín.	**Máx.**	**Absoluta Promedio**
X	0.3	0.298675	43	0.2385	17.7900	5.3330
Y	0.1949	0.164676	6	0.2781	18.4381	6.5009
Curvas de entrenamiento						
Coordenada X			**Coordenada Y**			
						
Parámetros comunes						
Número máximo de épocas			1000			
Rata de aprendizaje			0.05			
Mín. rendimiento del gradiente			1e-30			
Función de activación entrada y capa oculta			Tansig			
Función de activación capa de salida			Purelin			

Fuente: Autores del Proyecto

3.3.2 Algoritmo de búsqueda. Como alternativa para el desarrollo del sistema de localización, se han propuesto diferentes opciones para llegar a una solución lo suficientemente aceptable.

Una de estas alternativas es la determinación de un algoritmo de

búsqueda capaz de encontrar la posición del dispositivo dentro del área que delimita la grilla.

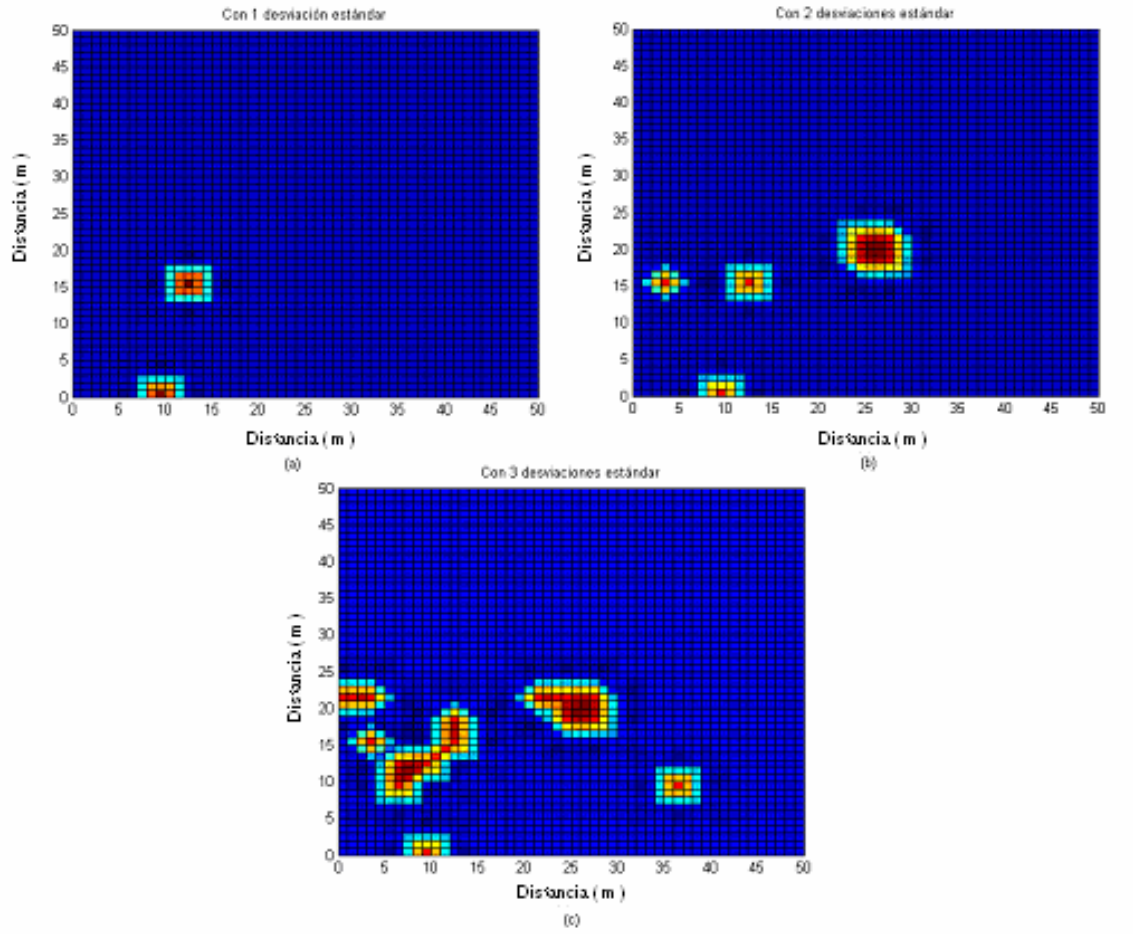
Partiendo del mapa de potencias construido con las mediciones realizadas en la sección 3.2.2, se ha dispuesto un algoritmo de búsqueda escrito en *Matlab*[13], que compare las potencias de entrada con las potencias consignadas en las matrices que describen los mapas de potencia experimentales.

Este algoritmo se ha diseñando para hacer la localización, mediante la comparación antes descrita. Para esto, se introducen las potencias transmitidas por los *APs* y el número de desviaciones estándar con que se desea que el algoritmo opere. La desviación estándar mencionada es la encontrada en la sección 3.2.1. Con estos parámetros, el algoritmo calcula una coordenada o una zona de posible posición del dispositivo. Para la comparación, el algoritmo contiene las matrices de potencia de los dos *APs* con los que se realizaron las mediciones, es decir, en el punto (0,0) de la grilla el DWL – 6000 (AP1) y en el punto (0,50) el DWL – 2000 (AP2).

En la figura 20 (a) se muestra un ejemplo de la solución con una desviación estándar, la figura 20 (b) con dos y en la figura 20 (c) con tres, para la combinación de potencias -56 dBm y -62 dBm medidas desde el AP1 y AP2, respectivamente, y que representa el punto ubicado en (12,15).

[13] Ver Anexo D

Figura 20. Soluciones del algoritmo de búsqueda



Fuente: Autores del Proyecto

En la figura 20, las zonas de color rojo intenso, son aquellas donde el algoritmo encontró potencias que pertenecen al rango delimitado por la desviación estándar.

Además, las figuras señalan la limitación que tiene el algoritmo para la infraestructura y la base de datos con que se cuenta, esto es, que presenta ambigüedad en la localización. Para cualquier valor de desviación estándar, el algoritmo entrega más de una solución, ya sea

varios puntos posibles, en el caso de una desviación estándar, como también varias zonas cuando se presenta más de una desviación estándar.

Para dar solución a estos inconvenientes, se ha probado con ciertas alternativas cambiando algunas de las condiciones del algoritmo, que se describen más adelante.

4. PRUEBAS Y RESULTADOS

Para realizar las pruebas del sistema de localización con sus dos alternativas, red neuronal y algoritmo de búsqueda, se seleccionaron 20 puntos ubicados dentro del área del experimento, al azar, pero teniendo en cuenta que los puntos se distribuyeran adecuadamente dentro del mismo, esto es, que se cubrieran todas las zonas del escenario escogido para el proyecto.

4.1 ALGORITMO DE BÚSQUEDA

En la tabla 11 se muestran los resultados de aplicar el algoritmo de búsqueda a los 20 puntos antes mencionados. Cada punto está descrito en la tabla por sus coordenadas reales, **Xr** e **Yr.**

Las desviaciones en X (**Err. X**) e Y (**Err. Y**) para una desviación estándar, se hallaron al hacer la diferencia entre las coordenadas respectivas, mientras que la incertidumbre hallada para el algoritmo con dos y tres desviaciones se encontró calculando la distancia del punto real con el lugar más cercano al perteneciente a la zona encontrada por el algoritmo.

La casilla de estado para el algoritmo con dos y tres desviaciones implica que las coordenadas reales están o no dentro de la zona que proporciona el algoritmo en su solución. El resultado 'En zona' indica que el punto se encuentra dentro de ella y 'No zona' que no lo está.

Tabla 11. Resultados obtenidos del algoritmo de búsqueda

Posición real (m)		Potencia (dBm)		Una desviación (m)				Dos desviaciones		Tres desviaciones	
Xr	Yr	AP1	AP2	X	Y	Err. X	Err. Y	Estado	Error (m)	Estado	Error (m)
0	0	-17	-61	0	0	0	0	En zona	0	En zona	0
0	50	-70	-12	0	50	0	0	En zona	0	En zona	0
50	0	-63	-73	No	No	-	-	En zona	0	En zona	0
50	50	-73	-63	No	No	-	-	En zona	0	En zona	0
27	0	-57	-68	24	3	3	3	No zona	1.4142	En zona	0
50	27	-69	-67	No	No	-	-	No zona	3.1623	No zona	2
27	50	-72	-52	27	50	0	0	En zona	0	En zona	0
0	27	-66	-57	24	39	24	12	En zona	0	No zona	1.4142
27	27	-57	-59	3	21	24	6	En zona	0	No zona	2
39	6	-62	-64	39	15	0	9	En zona	0	En zona	0
12	12	-51	-64	No	No	-	-	En zona	0	En zona	0
42	33	-67	-62	45	30	3	3	No zona	1	No zona	3
9	39	-64	-49	9	39	0	0	En zona	0	En zona	0
18	6	-52	-65	18	6	0	0	En zona	0	En zona	0
36	18	-63	-60	33	36	3	18	No zona	4.4721	En zona	0
24	45	-70	-57	30	42	6	3	No zona	1	No zona	1
15	24	-64	-65	No	No	-	-	No zona	11.4018	No zona	9.4868
9	6	-47	-61	9	6	0	0	En zona	0	En zona	0
45	45	-70	-61	45	42	0	3	No zona	1	En zona	0
12	33	-68	-59	12	33	0	0	En zona	0	En zona	0

Fuente: Autores del Proyecto

La tabla 11 muestra los resultados obtenidos al aplicar el algoritmo de búsqueda a una medición de potencias en espacio libre.

En la tabla 12 se presenta un resumen de los errores encontrados en las pruebas. Cabe anotar, que para los errores con una desviación estándar, no se tuvo en cuenta la información de los puntos que el algoritmo no encontró.

Tabla 12. Esquema consolidado de error para algoritmo de búsqueda

	Algoritmo de Búsqueda			
Desviación en la predicción	**1 Desviación**		**2 Desviaciones (m)**	**3 Desviaciones (m)**
	X (m)	**Y (m)**		
Absoluta promedio	4.2	3.8	1.1725	0.945
Máxima	24	18	11.4018	9.4868
Mínima	0	0	0	0
Desviación estándar	8.2393	5.3745	2.6902	2.2021

Fuente: Autores del Proyecto

Como ya se había acotado anteriormente [Sec. 3.3.2], el algoritmo de búsqueda es muy limitado en su solución. Algunas combinaciones de potencia no son encontradas por él y en otras ocasiones los errores son muy altos.

4.1 RED NEURONAL

En la tabla 13 se pueden apreciar los resultados que se obtienen al aplicar el modelo neuronal teniendo como entrada a los valores de potencia obtenidos en las mediciones de campo.

Tabla 13. Resultados obtenidos de la red neuronal.

Potencia (dBm)		Posición Real (m)		Predicción (m)		Desviación en la predicción (m)	
AP1	**AP2**	**X**	**Y**	**X**	**Y**	**Desviación X**	**Desviación Y**
-17	-61	0	0	0.9115	7.9013	0.9115	7.9013
-66	-57	0	27	22.2734	33.7064	22.2734	6.7064
-70	-12	0	50	1.4493	46.5425	1.4493	3.4575
-47	-61	9	6	3.8311	11.1477	5.1689	5.1477
-64	-49	9	39	6.2052	32.4206	2.7948	6.5794
-51	-64	12	12	10.2019	11.2085	1.7981	0.7915
-68	-59	12	33	35.0255	36.6034	23.0255	3.6034
-64	-65	15	24	39.9569	21.8046	24.9569	2.1954
-52	-65	18	6	12.9677	10.7221	5.0323	4.7221
-70	-57	24	45	32.8723	41.1358	8.8723	3.8642
-57	-68	27	0	20.9830	8.2621	6.0170	8.2621
-57	-59	27	27	14.1346	18.4482	12.8654	8.5518
-72	-52	27	50	13.9271	44.9777	13.0729	5.0223
-63	-60	36	18	22.6492	26.3739	13.3508	8.3739
-62	-64	39	6	29.5593	20.6354	9.4407	14.6354
-67	-62	42	33	41.2842	32.1453	0.7158	0.8547
-70	-61	45	45	41.6053	38.9025	3.3947	6.0975
-63	-73	50	0	49.2110	4.1295	0.7890	4.1295
-69	-67	50	27	49.5204	24.5263	0.4796	2.4737
-73	-63	50	50	44.2514	41.5984	5.7486	8.4016

Fuente: Autores del Proyecto

En la tabla 14 se muestran algunos resultados estadísticos obtenidos a partir de los resultados de las pruebas, expuestos en la tabla 13.

Tabla 14. Esquema consolidado de error para red neuronal

	Red neuronal	
Desviación en la predicción	**X (m)**	**Y (m)**
Absoluta promedio	8.1079	5.5886
Máxima	24.9569	14.6354
Mínima	0.4796	0.7915
Desviación estándar	7.8306	3.2668

Fuente: Autores del Proyecto

Como se mencionó en el capítulo anterior [Sec. 3.3.1], el modelo neuronal tiene problemas para determinar de manera correcta la posición de la tarjeta. Esto puede ser observado en la columna de desviación en la predicción, que representa en cuántos metros falló la red neuronal en la predicción de la coordenada.

Esta limitante se debe a la infraestructura de la red WLAN, ya que solo se cuenta con dos *APs* y las tarjetas no están diseñadas para mantener estabilidad, causando que el comportamiento de las señales de potencia en un mismo punto oscile dentro de un rango bastante alto, afectando de manera considerable la solución dada por el modelo neuronal.

4.3 Interfaz gráfica en *Matlab*

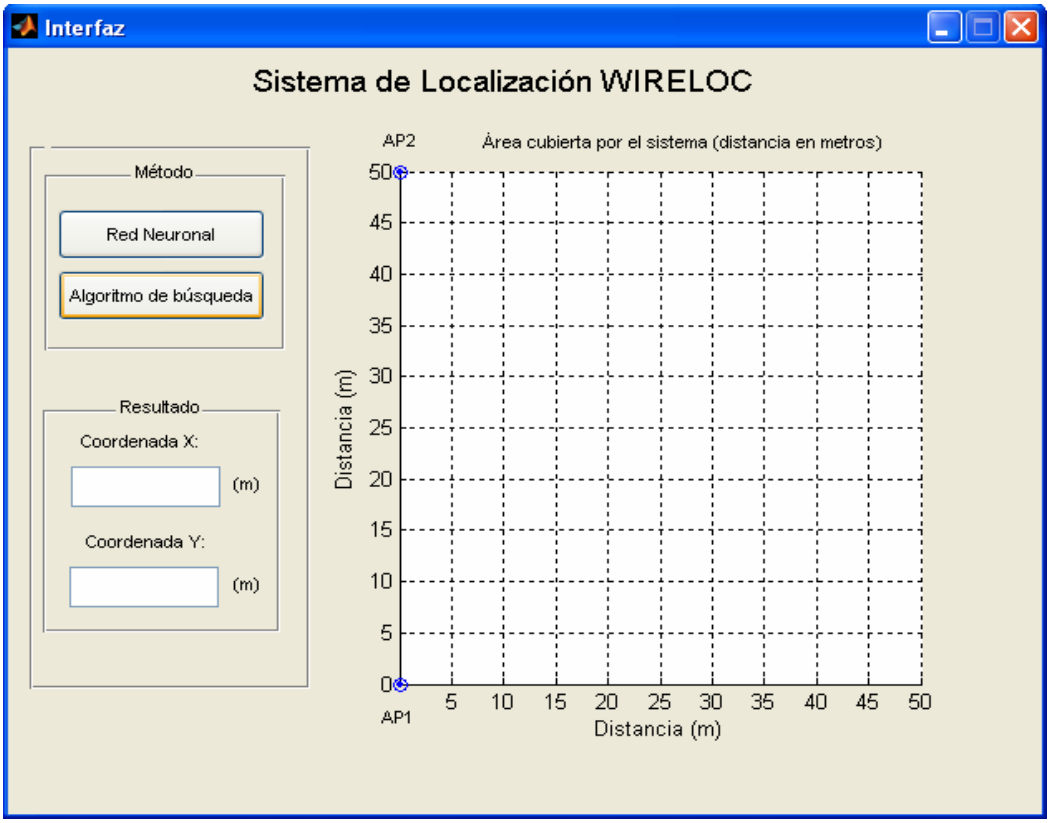
Para efectos de presentar al usuario un entorno agradable que le permita introducir los valores de potencia, escoger qué modelo a utilizar para determinar la ubicación de la tarjeta (Algoritmo de búsqueda o Red neuronal) y desplegar en un contexto gráfico la solución, se desarrolló una interfaz gráfica utilizando la *toolbox Guide* de *Matlab.*

La interfaz gráfica permite introducir los valores de potencia a través de una casilla de edición de texto. Para hallar la solución, el usuario tendrá la

posibilidad de escoger entre el algoritmo de búsqueda o la red neuronal; además, para el algoritmo de búsqueda habrá la posibilidad de escoger con cuantas desviaciones ejecutar el algoritmo, mediante un menú desplegable. La solución será mostrada en forma de coordenadas X-Y por medio de un cuadro de texto y, a través de una ventana que muestre el punto de ubicación ó la zona donde éste encuentra la tarjeta, según se haya escogido entre la red neuronal ó el algoritmo de búsqueda.

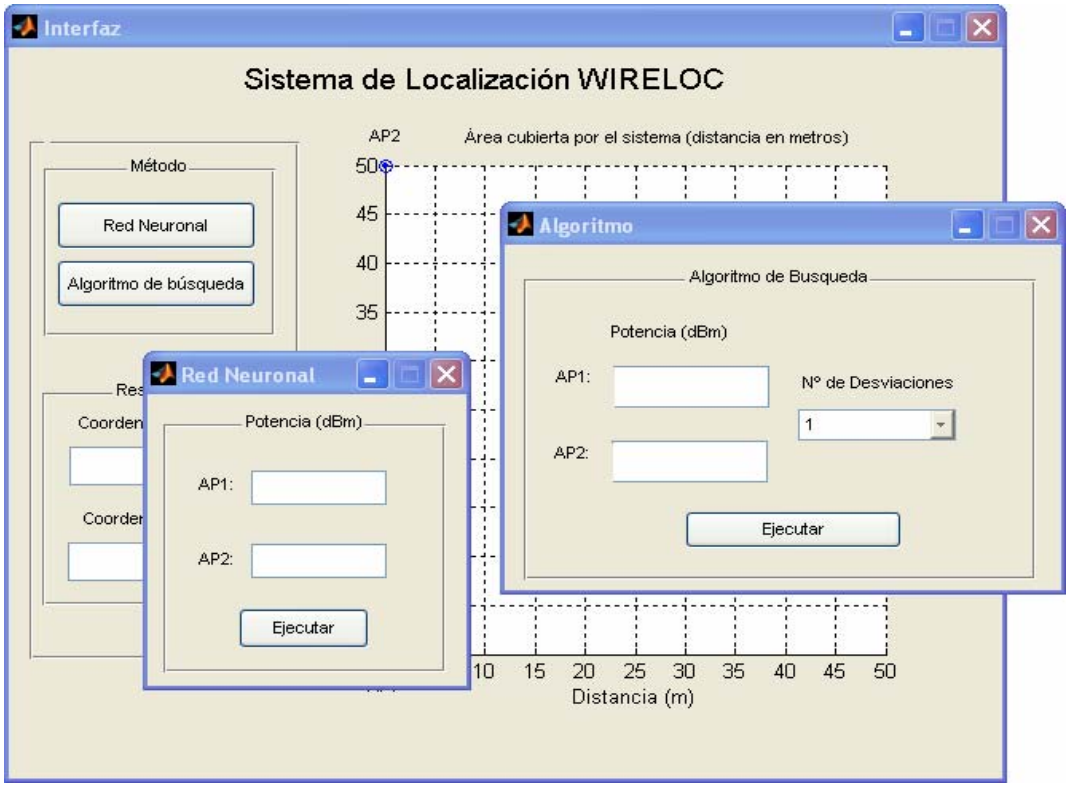
En las figuras 21 y 22 se muestran tanto el panel inicial de la interfaz, como los paneles que se despliegan al dar clic sobre los botones Red Neuronal ó Algoritmo de búsqueda.

Figura 21. Presentación de la Interfaz



Fuente: Autores del Proyecto

Figura 22. Opciones de la Interfaz



Fuente: Autores del Proyecto

4.2 ALTERNATIVAS DE SOLUCIÓN

Para posteriores estudios, con miras a mejorar el sistema en cuanto a su precisión, se ha encontrado, al realizar diferentes pruebas de laboratorio, que al aumentar el número de *APs* en el área se mejora en un gran porcentaje la precisión en las predicciones de la posición.

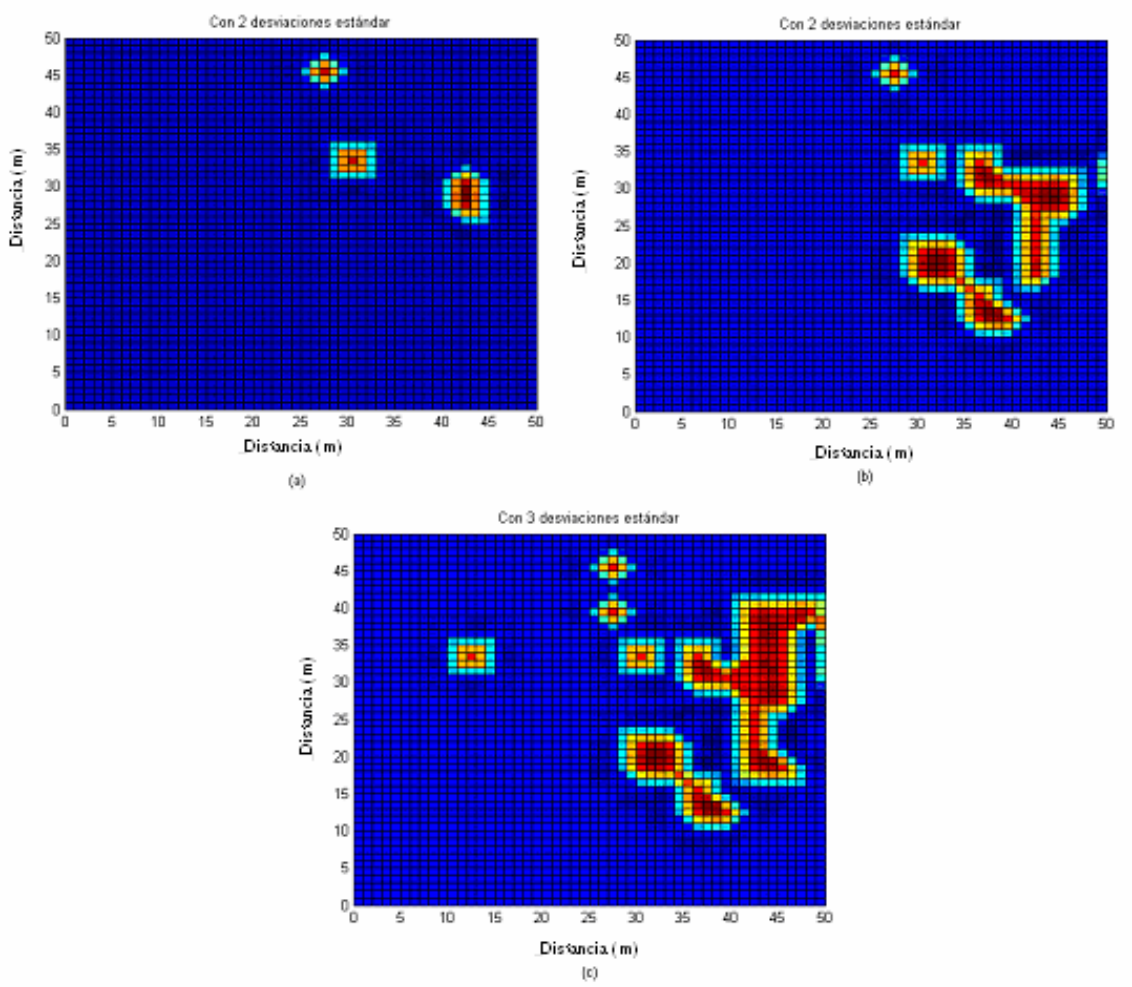
4.2.1 Respecto al algoritmo de búsqueda. Para dar solución a los inconvenientes que se presentaron en la sección 4.1, se ha probado con ciertas alternativas cambiando algunas de las condiciones del algoritmo, que se describen a continuación.

- **Con dos puntos de acceso de la misma referencia[14].** Esta alternativa se basa principalmente en el algoritmo de la sección 3.3.2, con la diferencia que las matrices de potencia con las que se realiza la comparación, pertenecen a las mediciones tomadas de las señales transmitidas por el punto de acceso DWL – 2000AP+ únicamente, es decir, la matriz es reflejada de izquierda a derecha con el comando fliplr[15] de *MatIab*, ubicándose nuevamente, de una manera virtual, dos puntos de acceso en los puntos de coordenadas (0,0) y (0,50).

En la figura 23 se muestran los resultados que se obtienen al ejecutar este algoritmo. En la figura 23 (a), con una desviación estándar, y en las figuras 23 (b) y (c), con dos y tres desviaciones estándar, respectivamente. La combinación de potencias es -66 y -61 dBm, que corresponde al punto (33,21).

[14] Ver anexo D
[15] Ver anexo A

Figura 23. Primera solución alternativa del algoritmo de búsqueda.



Fuente: Autores del Proyecto

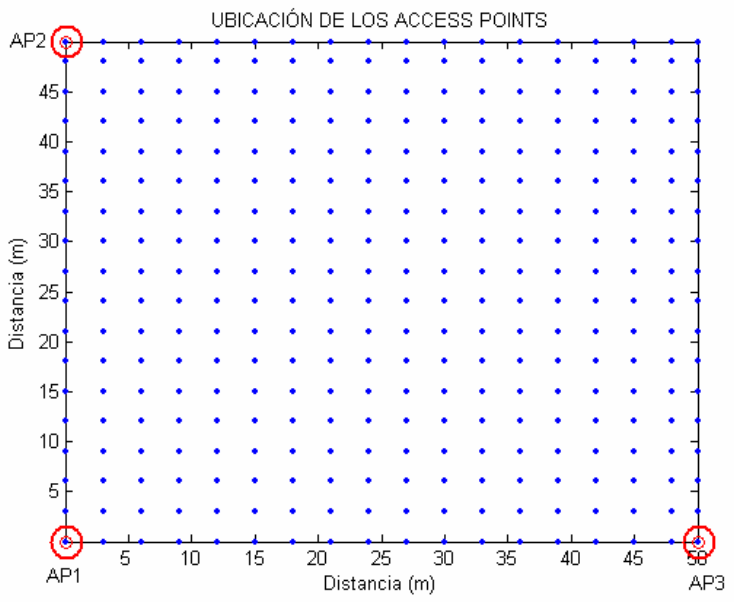
Nuevamente, como en el algoritmo de la sección 3.3.2, se presenta el inconveniente de divergencia de las posibles soluciones, esto es, existen varias posibles soluciones, y evidentemente esto implicaría una incertidumbre elevada en la localización.

- **Con tres puntos de acceso[16].** La propuesta es emplear tres puntos de acceso, en lugar de dos, para reducir la probabilidad de error.

[16] Ver anexo D

Se asume que teniendo esta infraestructura se eliminan los puntos con combinaciones de potencia igual, caso en el cual la divergencia antes observada desaparece. Los tres *APs* son, como en el caso anterior, la reflexión de la matriz de mediciones del AP2 tanto de derecha a izquierda, como de arriba hacia abajo, con los comandos de *fliplr* y *flipud*[17]; esto quiere decir que para esta solución tenemos un AP en el punto (0,0), otro en (0,50) y un tercero en el punto (50,0). En la figura 24 se ilustra la ubicación de éstos en la grilla.

Figura 24. Ubicación de tres puntos de acceso en la grilla



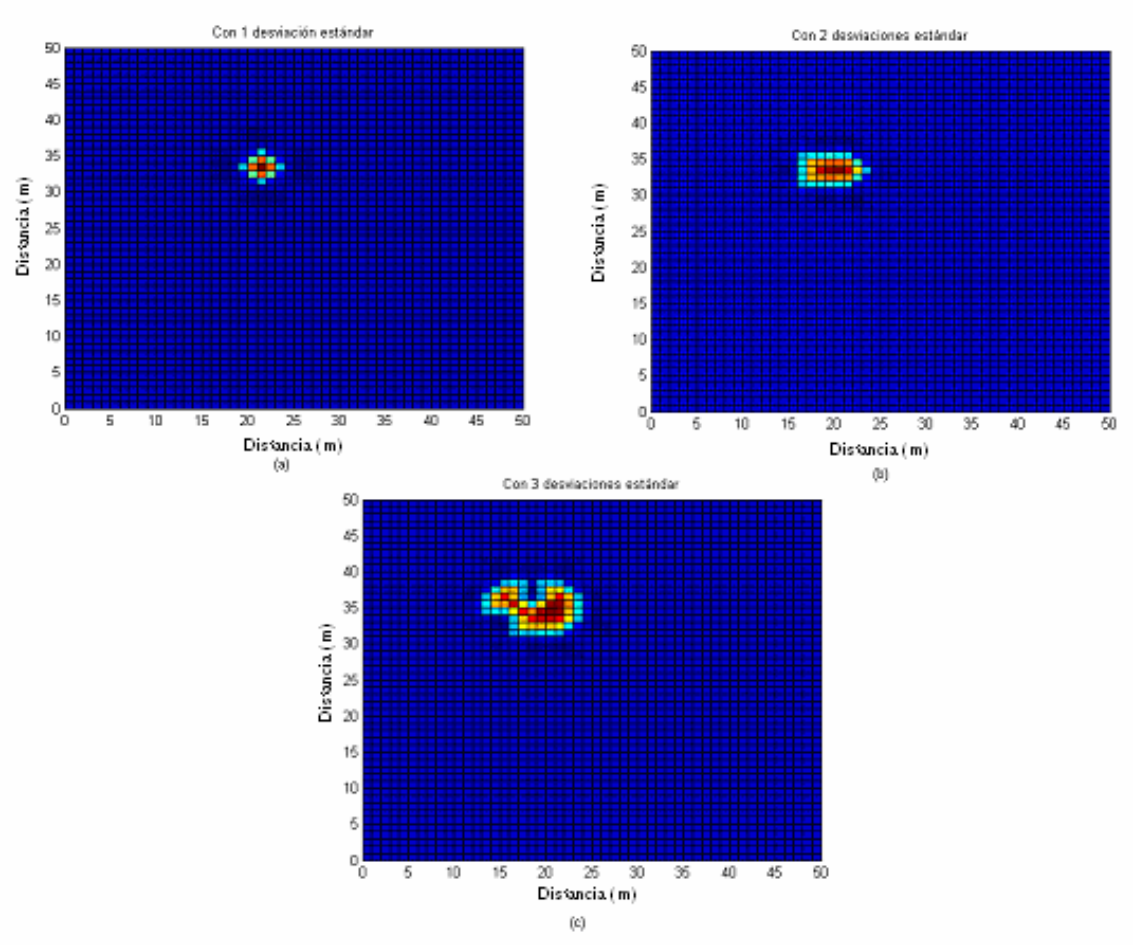
Fuente: Autores del Proyecto

Al igual que los dos algoritmos anteriormente descritos, éste solicita la introducción de las tres potencias leídas de las señales de los *APs* por medio de la herramienta *software Netstumbler*, así como el número de desviaciones estándar.

[17] Ver anexo A

En la figura 25 se presentan los resultados para una, dos y tres desviaciones estándar, en las figuras 25 (a), (b) y (c), respectivamente, para el punto (21,33) que tiene como combinación de potencias -65, -51 y -61 dBm correspondientes a cada uno de los *APs*.

Figura 25. Segunda solución alternativa del algoritmo de búsqueda



Fuente: Autores del Proyecto

Se advierte de las figuras, que el sistema de localización responde mejor cuando tiene tres potencias provenientes de tres *APs*. Para el punto de

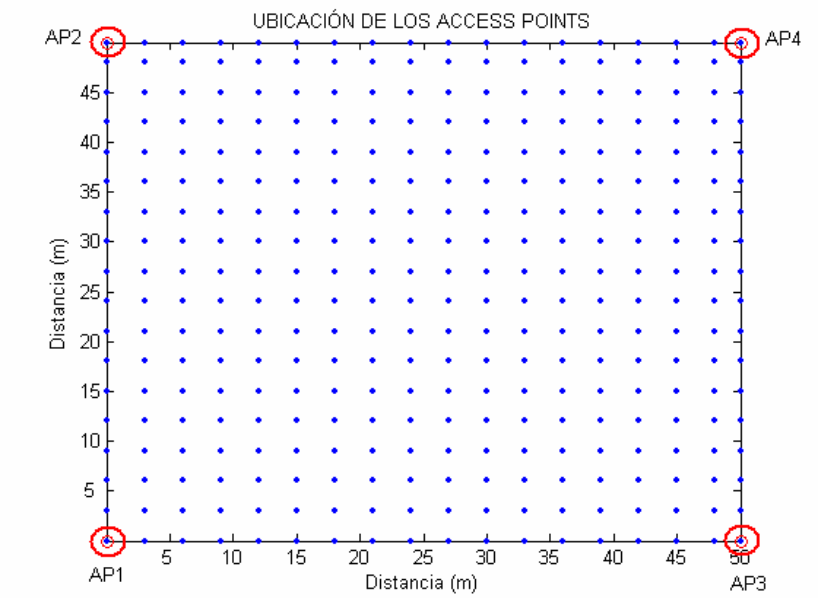
prueba, se puede decir que el algoritmo es válido pues en los tres casos, tres diferentes desviaciones estándar, arroja un resultado único, es decir, un solo punto o una zona única donde se encuentra el dispositivo. Sin embargo, para otros puntos el sistema puede no encontrar la combinación y tener una desviación, que no obstante, se encuentra dentro del límite aceptable para este sistema.

- **Con cuatro puntos de acceso[18].** Esta última propuesta consta del mismo principio de todos los anteriores algoritmos, con la diferencia de que se emplea una infraestructura conformada por cuatro *APs*. De la misma forma como en los algoritmos descritos previamente, en este se ha reflejado la matriz obtenida con las mediciones del AP2 para lograr ubicar los cuatro *APs* en las cuatro esquinas de la grilla, esto es, que existe un AP en el punto (50,50) además de los utilizados en algoritmo anterior. El propósito de agregar un *AP* es para intentar conseguir disminución en las posibles desviaciones y eliminar la ambigüedad en la solución. La figura 26 muestra la distribución de éstos en la grilla.

Aunque se esperaría que el resultado con esta alternativa fuese aún más precisa que con el algoritmo que emplea tres puntos de acceso, no lo es del todo. De las diferentes combinaciones que se pusieron a prueba, hay mayor número de combinaciones no encontradas que con el anterior. Pero por otra parte, cuando la combinación es encontrada, la exactitud en la localización es mayor.

[18] Ver anexo D

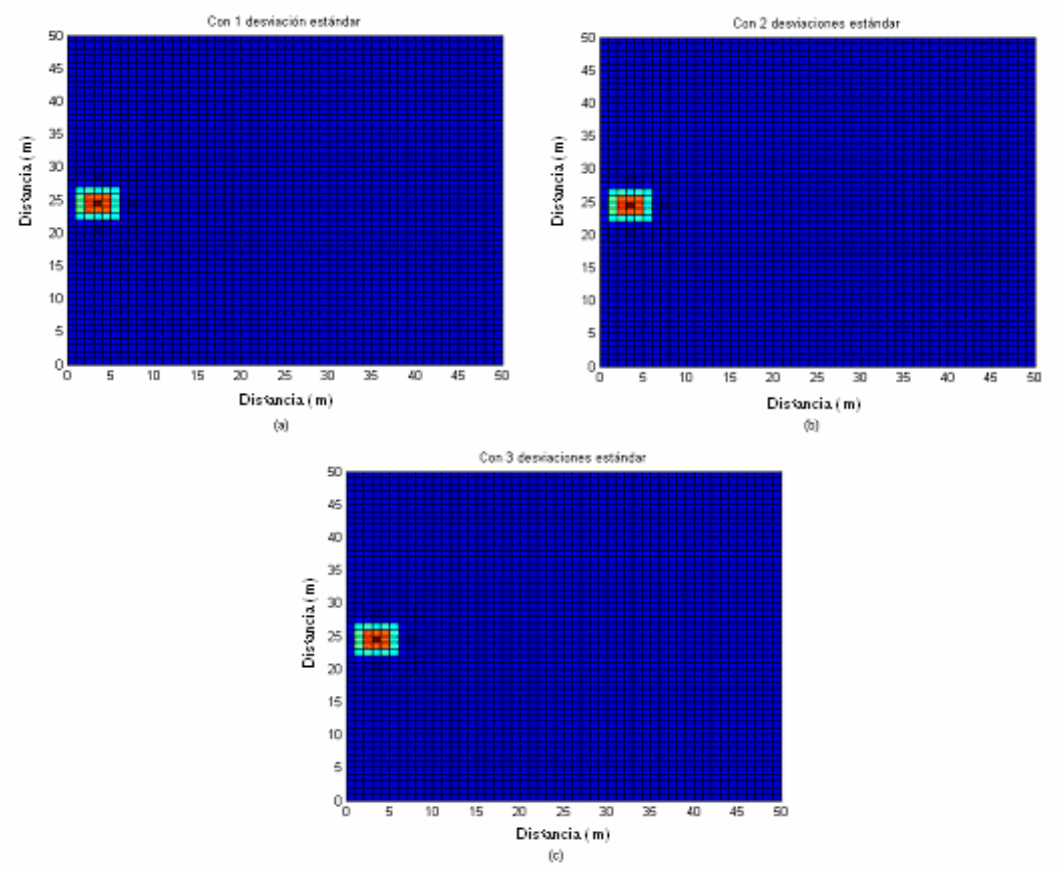
Figura 26. Ubicación de cuatro puntos de acceso en la grilla



Fuente: Autores del Proyecto

Como ejemplo se presentan en la figura 27 (a), (b) y (c), los resultados obtenidos para la combinación potencias -56, -58, -69 y -72 dBm del punto (3,24).

Figura 27. Tercera solución alternativa para el algoritmo de búsqueda



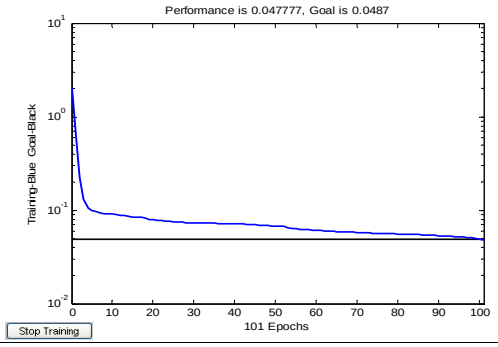
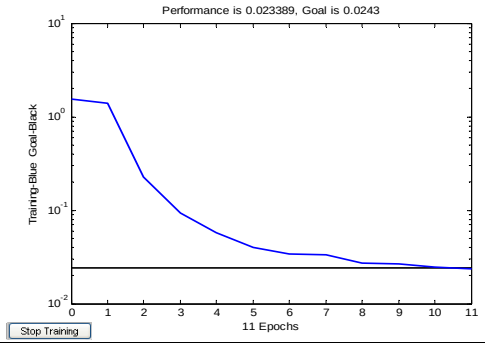
Fuente: Autores del Proyecto

Respecto a la red neuronal. El principal problema que presenta el modelo neuronal es que los datos de potencia obtenidos en las mediciones de campo y utilizados para el entrenamiento de la red oscilan a medida que el dispositivo móvil se aleja de la antena; es decir, no tienen una clara tendencia a disminuir con la distancia, que es el comportamiento que se espera de las señales que se propagan en espacio libre.

En la tabla 15 se presenta el resultado del entrenamiento de la red neuronal para predecir la posición del dispositivo móvil. Los patrones

utilizados para el entrenamiento del modelo neuronal son tomados a partir de los datos de potencia emitida por el punto de acceso, DWL-2000, obtenidos en la campaña de medición. Estos datos se reflejan a los extremos de la grilla para simular la presencia de más puntos de acceso (cuatro en total), con el fin de dar una solución alternativa para el sistema de localización.

Tabla 15. Parámetros y resultados alternativos para la red neuronal.

<table>
<tr><th colspan="6">Red Datos Experimentales</th></tr>
<tr><th rowspan="2">Coordenada</th><th rowspan="2">Error de entrena-miento</th><th rowspan="2">MSE alcanzado</th><th rowspan="2">Número de épocas</th><th colspan="3">Desviación de la predicción (m)</th></tr>
<tr><th>Mín.</th><th>Máx.</th><th>Absoluta promedio</th></tr>
<tr><td>X</td><td>0.0487</td><td>0.047777</td><td>101</td><td>0.0157</td><td>14.9158</td><td>5.7282</td></tr>
<tr><td>Y</td><td>0.0243</td><td>0.023389</td><td>11</td><td>0.0121</td><td>9.8619</td><td>2.9363</td></tr>
<tr><th colspan="6">Curvas de entrenamiento</th></tr>
<tr><th colspan="3">Coordenada X</th><th colspan="3">Coordenada Y</th></tr>
<tr><td colspan="3"> 
</td><td colspan="3"> 
</td></tr>
<tr><th colspan="6">Parámetros comunes</th></tr>
<tr><td colspan="3">Número máximo de épocas</td><td colspan="3">1000</td></tr>
<tr><td colspan="3">Rata de aprendizaje</td><td colspan="3">0.05</td></tr>
<tr><td colspan="3">Mín. rendimiento del gradiente</td><td colspan="3">1e-30</td></tr>
<tr><td colspan="3">Función de activación entrada y capa oculta</td><td colspan="3">Tansig</td></tr>
<tr><td colspan="3">Función de activación capa de salida</td><td colspan="3">Purelin</td></tr>
</table>

Fuente: Autores del Proyecto

El aumento en el número de *Access Points* contribuye a mejorar la precisión del sistema de localización, reduciendo la desviación en la predicción en 5 m aproximadamente. No obstante, la variabilidad en las señales de potencia genera un error acumulativo, por tanto al hacer el reflejo de este valor para simular la presencia de más *APs* no se genera una disminución en la incertidumbre de la predicción que justifique el aumento de la carga computacional. A menos que se pueda asegurar estabilidad en las señales obtenidas por la tarjeta y un comportamiento que se asemeje más al observado en la ecuación de perdidas en espacio libre, no es recomendable la utilización de un modelo neuronal para abordar un problema de localización de un dispositivo dentro de una red LAN inalámbrica.

En la figura 28 se presenta una gráfica que permite apreciar la variación de las señales de potencia emitidas por los puntos de acceso en función de la distancia, así como también la curva característica de la ecuación de *Friis*.

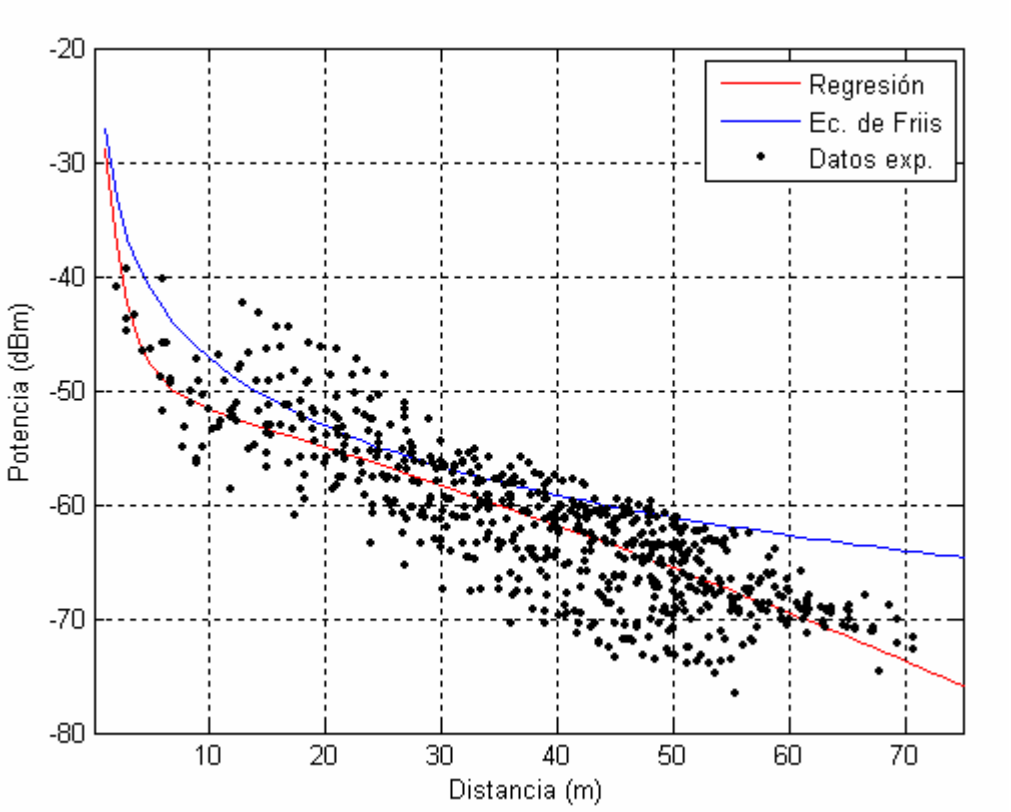
La figura 28 también muestra la gráfica que se obtiene al ajustar los datos de potencia vs. distancia a una ecuación exponencial. A pesar de la gran dispersión que presentan estos datos, la ecuación obtenida de la regresión permite observar un comportamiento de la señal de potencia similar al presentado por la ecuación de pérdidas en espacio libre.

La regresión está representada por la ecuación:

$$f(x) = -48.9e^{0.005861\,x} + 33.86e^{-0.5109\,x} \quad (10)$$

Los coeficientes presentan una confiabilidad del 95%, con suma del error cuadrático medio de 8993.

Figura 28. Regresión, ecuación de *Friis* y dispersión de los datos



Fuente: Autores del Proyecto

Por medio de esta ecuación (Ec. 10) se podría caracterizar la propagación de las señales de potencia emitidas por los *APs*, lo que daría una herramienta muy útil para resolver el problema de la localización del dispositivo móvil, permitiendo dar una solución ya sea resolviendo la ecuación ó aplicando un modelo neuronal, esto, siempre y cuando se pueda garantizar una disminución considerable en la dispersión presentada por datos de potencia.

5. CONCLUSIONES Y RECOMENDACIONES

Este capítulo pretende enumerar las conclusiones, recomendaciones y análisis realizados al culminar la investigación y las pruebas hechas con el sistema de localización diseñado.

5.1 CONCLUSIONES

Los resultados obtenidos en este trabajo constituyen un avance considerable y una base que servirá como etapa inicial para desarrollar trabajos futuros relacionados con la implementación de sistemas de localización en redes inalámbricas. Se podría decir que debido a la poca información encontrada de trabajos relacionados, la investigación realizada en este proyecto sería pionera en Colombia, como herramienta que permite la localización de un dispositivo móvil al interior de una WLAN.

Teóricamente, el sistema de localización tuvo un excelente comportamiento, pues las desviaciones encontradas fueron mínimas y en la mayoría de los casos fue nula. Para las tres opciones aplicadas, solución con comandos de *Matlab*, redes neuronales y algoritmo de búsqueda, la precisión lograda fue totalmente satisfactoria, lo que implica que la parte del diseño del sistema se alcanzó con éxito.

En este proyecto se desarrolló una herramienta que pretende ubicar un dispositivo móvil al interior de una red de área local inalámbrica, basada

en el modelo empírico, en donde la información de la señal fue recopilada y almacenada en una base de datos, a partir de la cual se implementaron dos técnicas programadas en *Matlab*. La primera es un algoritmo de búsqueda que toma las potencias emitidas por cada uno de los puntos de acceso y busca dentro de la base de datos, qué combinación de potencias se ajusta a la introducida por el usuario, dando como respuesta un punto o zona de ubicación. La segunda es una red neuronal previamente entrenada y probada con valores tomados de la base de datos. En esta técnica la red neuronal hace una predicción de la coordenada a partir de los datos de potencia introducidos por el usuario, generando el punto de ubicación del dispositivo.

Experimentalmente, con la infraestructura con que se cuenta, dos puntos de acceso, el resultado no fue óptimo para la solución del problema, puesto que la inestabilidad de las señales condujo a que la incertidumbre en la posición del dispositivo fuese bastante alta. No obstante, se presume que al aumentar la cantidad de puntos de acceso, este problema sería resuelto con mayor exactitud, logrando una considerable disminución en las desviaciones de la predicción.

Para que el sistema de localización tenga un uso práctico está obligado a trabajar en tiempo real. Por esto, los métodos utilizados para determinar la ubicación del dispositivo deben ajustarse a ciertos requerimientos de velocidad de respuesta. Esto implica que las herramientas empleadas para ubicar el dispositivo deben trabajar con un mínimo de carga computacional. Para tal efecto, las técnicas utilizadas en dar solución al problema (red neuronal y algoritmo de búsqueda) fueron desarrolladas buscando un tiempo de respuesta mínimo.

Tener un amplio conocimiento de las características de las redes inalámbricas, de los equipos utilizados y del problema a solucionar

permite hacer un diseño de experimento apropiado, en donde los datos tomados en las mediciones de campo puedan registrar adecuadamente el comportamiento real de las señales de potencia emitidas por los *puntos de acceso*, plasmando sus particularidades, tales como la variación con el tiempo y la distancia, así como los factores que puedan generan problemas de propagación y dispersión. Se encontró durante la etapa de mediciones, que la propagación tiene un tiempo para estabilizarse, puesto que según las observaciones hechas al efectuar la prueba de los *APs* en un punto estático, se advirtió que pasado un tiempo, las intensidades empezaron a ocupar un rango definido. Además de esto, se percibió una particularidad, y es que contrario a lo esperado, al tener los dos *APs* encendidos la intensidad de señal se mantuvo más estable que cuando se hicieron las mediciones por separado.

La utilización de redes neuronales diseñadas específicamente para dar solución al problema de localización permitió obtener un modelo de rápida convergencia hacia los parámetros de diseño y, sin la necesidad de emplear redes demasiado complejas. De esta manera se logró un modelo neuronal sencillo y con bajos tiempos de entrenamiento, disminuyendo considerablemente la carga computacional.

En problemas que puedan ser resueltos aplicando redes neuronales, es recomendable buscar comportamientos o patrones que permitan ser analizados independientemente, con el fin de aplicar un modelo para cada uno de ellos, reduciendo el tamaño de la red. Es importante hacer una adecuada selección de los parámetros de entrenamiento, así como una correcta manipulación de los datos empleados para el entrenamiento, esto permite una mejor y más rápida convergencia del modelo.

En el diseño de una red neuronal no hay un procedimiento estándar a seguir que permita hacer una adecuada selección de sus dimensiones y

parámetros de red, así como no existe una única solución al problema. Sin embargo, siguiendo una serie de pautas encontradas en la literatura y en trabajos realizados dentro del grupo de investigación en conectividad y procesado de señal (CPS), se puede llegar al diseño de un modelo que evite sobredimensionar el sistema y sin llegar a problemas de memorización.

Como resultado final, se consigue una sencilla aplicación, de interfaz amigable y de fácil operación, a la cual se le introducen vía teclado los valores leídos de la herramienta *Netstumbler* y que escogiendo la metodología y ciertos parámetros de la misma, se da una solución a la localización del dispositivo al interior de la red inalámbrica.

5.2 RECOMENDACIONES

También se ha reflexionado acerca de algunas recomendaciones para futuros proyectos relacionados con la localización inalámbrica con miras a mejorar la exactitud alcanzada en este proyecto.

Inicialmente, se sugiere un aumento en el número de puntos de acceso en la parte experimental, ya que, como se mencionó previamente, esto ayudaría de manera considerable a reducir el número de puntos con igual combinación de potencias, lo que conduciría a una disminución de la incertidumbre de la predicción.

Debido a que según las especificaciones de las tarjetas inalámbricas sugieren un mejor desempeño en ambientes interiores (*indoor*), se podría realizar una implementación del sistema de localización en un ambiente así. Con ello se reduciría el área de experimentación y medición y teóricamente se tendrían más consideraciones, las cuales brindan una

mejor caracterización del ambiente de trabajo. A parte de esto, la manipulación y transporte de los equipos se harían más cómodos y el tiempo empleado en las mediciones y las pruebas se reduciría. No obstante, con una caracterización del terreno donde se realicen las pruebas, se podría obtener un soporte más para alcanzar una mejor predicción de la posición del dispositivo.

Como alternativa para futuros estudios, se recomienda que este problema sea tratado como un problema de *tracking.* Es decir, utilizar registros de potencia anteriores y actuales, y limitar la velocidad de desplazamiento del dispositivo móvil al interior del área en estudio, con el fin de predecir la posición del mismo.

Con respecto a la captura de los datos, encontrar una manera de colectarlos y llevarlos automáticamente a la herramienta, sería una mejora para el sistema y daría la posición en tiempo real. También se podría implementar una aplicación en un servidor remoto que pudiese monitorear la posición de un usuario, ya que este proyecto se desarrolló en el mismo cliente.

La fusión con otros proyectos desarrollados por el grupo CPS junto con el trabajo realizado en éste, resultaría más satisfactorio y le daría continuidad a los adelantos logrados.

BIBLIOGRAFÍA

[Wikipedia/WLAN] Wikipedia, La enciclopedia libre. es.wikipedia.org/wiki/WLAN.

[Wikipedia/Access_point] Wikipedia, La enciclopedia libre. es.wikipedia.org/wiki/Access_point.

[Ministerio de Comunicaciones, 2003] Ministerio de Comunicaciones, República de Colombia, "Tecnologías WiFi Bandas Eléctricas sin Licencia", Septiembre de 2003, Colombia.

[Shih, 2003] SHIH Johnny, "Wireless LAN Location System". Noviembre de 2003. Escuela de Tecnología de la Información e Ingeniería Eléctrica. Universidad de Queensland, Australia.

[Saunders, 1999] SAUNDERS S.R., "Antennas and propagation for wireless communication systems", Wiley, 1999.

[Del Brío, Molina, 2002] DEL BRÍO Bonifacio Martín, MOLINA Alfredo Sanz. Redes Neuronales y Sistemas Difusos, 2 Ed. Universidad de Zaragoza. ALFAOMEGA GRUPO EDITOR, S.A. DE C.V. 2002.

[Guzmán, 2005] GUZMÁN Paola. "Trabajo de la gestión en los dispositivos administrables dentro de la red de datos institucional". Febrero de 2005. Facultad de Ingenierías Fisicomecánicas. Escuela de ingeniería Eléctrica,

Electrónica y de telecomunicaciones. Universidad Industrial de Santander. Colombia.

[Hurtado, Reyes, 2001] HURTADO Diana, REYES Oscar. "Predicción de Parámetros de una Planta de Craqueo Catalítico de Fluidos (FCCU – Fluid Catalytic Cracking Unit) Mediante Modelos Neuronales". Facultad de Ingenierías Fisicomecánicas. Escuela de ingeniería Eléctrica, Electrónica y de telecomunicaciones. Universidad Industrial de Santander. Colombia. 2001.

BIBLIOGRAFIA COMPLEMENTARIA

The Institute of Electrical and Electronics Engineers, Inc. "IEEE P802.11, The Working Group for Wireless LANs", http://grouper.ieee.org/groups/802/11/

[Lenihan]Nicola LENIHAN, "WLAN Positioning", Universidad de Limerick, Irlanda. http://www.ul.ie/nlenihan/WLAN%20positioning.pdf

Bahl, P. Microsoft Corp. "RADAR: An In-Building RF-based User Location and Tracking System", Universidad de Rochester, U.S.A. http://research.microsoft.com/~padmanab/papers/infocom2000.pdf

Student Project, "Advanced WaveLan Positioning", Mayo de 2001, Universidad de Tecnología de Lulea, Suecia. http://web.media.mit.edu/~alisa/2001-05-23.pdf

Blake M. Harris, "Amulet: Aproxímate Mobile User Location Tracking System", Universidad de Rochester, U.S.A. http://darkfate.com/bmh/other/pubs/Amulet.pdf

Universidad de Stanford "Halibut: An Infrastructure for Wireless LAN-based Location Tracking", U.S.A. http://fern2.stanford.edu/cs444n/

Ekahau, Inc. "Ekahau Technology and Products", Finlandia.

http://www.vtt.fi/virtual/navi/expo2003/Ekahau030402.pdf

María Acosta, Camilo Zuluaga. "Tutorial sobre redes neuronales aplicadas en ingeniería eléctrica y su implementación en un sitio Web". Octubre de 2000. Facultad de ingeniería eléctrica. Universidad tecnológica de Pereira. Colombia.

ANEXO A

MATLAB

Durante los últimos años, *Matlab* se ha convertido en una de las herramientas más importantes y más útiles en todos los campos de la ingeniería. Para el caso particular de ingeniería electrónica y, más aún, para este proyecto, Sistema de localización, *Matlab* ha sido herramienta fundamental para el desarrollo de la parte software que éste posee. Haciendo uso de sus múltiples y versátiles comandos, sus toolboxes y funciones, se ha llegado a comprender y desplegar con éxito gran parte de este trabajo.

En la tabla se enuncian los comandos y funciones más relevantes de los cuales se ha hecho uso para el sistema de localización.

COMANDO	DESCRIPCIÓN	SINTAXIS
function	Agrega una nueva función	function [salida(s)] = nombrefuncion (parámetros)
fopen	Abre archivo	Variable = fopen (nombrearchivo)
fscanf	Lee datos con formato desde un archivo	[varible(s)] = fscanf (nombrearchivo, formato, tamaño)
fclose	Cierra archivo	Variable = fclose (nombrearchivo)
fprintf	Escribe datos con formato en un archivo	Variable = fprintf (nombrearchivo, formato,

		arreglo, …)
input	Propone un apuntador para una entrada de usuario	Variable = input (‘apuntador ’)
optimset	Crea y/o altera las estructuras optim options	Opciones = optimset (parametro1, valor1, parametr2, valor2, …)
fsolve	Resuelve ecuaciones no lineales de varias variables	Variable = fsolve (funcion, valorinicial)
solve	Da la solución simbólica de ecuaciones algebraicas	Solve ('eqn1','eqn2',...,var1,var2,...,)
eval	Ejecuta un string con una expresión de *Matlab*	Eval (string)
meshgrid	Crea arreglos para gráficas en 3-D	[VAR1, VAR2] = meshgrid (var1, var2)
griddata	Grilla de datos y ajuste de curvas	[variables] = griddata (varX, varY, varZ, varXI, varYI)
pcolor	Grafica una matriz en pseudoscolor (tablero de ajedrez)	Pseudocolor (nombrematriz)
mesh	Grafica en 3-D una malla definida por cuatro argumentos de matriz.	Mesh (rangoX, rangoY, rangoZ, rangocolor)
round	Redondea un los elementos de una matriz hacia el entero más cercano.	Round (nombrematriz)
mean	Encuentra el valor medio o promedio de los elementos de una matriz.	Mean (nombrematriz)
std	Encuentra la desviación	Std (nombrematriz)

	estándar de los elementos de una matriz.	
fliplr	Halla la matriz simétrica respecto a un eje vertical.	Fliplr(nombrematriz)
linspace	Genera un vector con **n** valores igualmente espaciados entre **x1** y **x2.**	Linspace(n, x1, x2)
reshape	Cambia el tamaño de la matriz **A** devolviendo una matriz de tamaño $m \times n$.	Reshape(A,m,n)
prestd	Preprocesa los datos para que la media sea 0 y la desviación estándar 1.	[DatosNormalizados,meanDatos, stdDatos] = prestd(Datos)
trastd	Preprocesa los datos usando media y desviación estándar precalculadas.	[DatosNormalizados] = trastd(Datos,meanDatos,stdDatos)
poststd	Post-procesa los datos que han sido preprocesados con prestd.	poststd(DatosNormalizados, meanDatos,stdDatos)
newff	Crea una red neuronal tipo Backpropagation.	net = newff(PR,[S1 S2...SNl],{TF1 TF2...TFNl},BTF,BLF,PF)
train	Entrena la red neuronal de acuerdo con los parámetros seleccionados.	train(NET,P,T,Pi,Ai,VV,TV)
sim	Simula un modelo neuronal	Sim(NombreRed,Parámetros Entrada)

PR: Matriz Rx2 de valores máximos y mínimos de cada una de las N neuronas de entrada.

Si: Número de neuronas para cada una de las capas.

TFi: Función de transferencia a utilizar en cada una de las capas.
BTF: Algoritmo de entrenamiento a utilizar.
BLF: Función de actualización de pesos.
PF: Función para evaluar el desempeño de la red.
NET: Nombre de la red.
P: Parámetros de entrada.
T: Parámetros de salida.
Pi: Condiciones iniciales de retardo a la entrada.
Ai: Condiciones iniciales de retardo en las capas.
VV: Estructura del vector de validación.
TV: Estructura del vector de prueba.

Además de las funciones y comandos antes mencionados, se hizo uso de la herramienta de *Matlab* conocida como GUIDE (Graphical User Interface Development Environment, Ambiente de Desarrollo de Interfaz Gráfica de Usuario), que permite crear interfaces para que el uso de ciertos programas se haga mas general y amigable para los usuarios.

ANEXO B

EQUIPO UTILIZADO

En este anexo, se presenta el equipo empleado en la realización y conducción del proyecto.

B.1 ACCESS POINTS

B.1.1 DWL – 2000AP+. El AP DWL – 2000AP+ de fabricante DLink, es un dispositivo inalámbrico de alta velocidad diseñado bajo el estándar 802.11 g, pero que a su vez posee la propiedad de operar con equipo inalámbrico de los estándares 802.11b, 802.11b+ obedeciendo a las especificaciones correspondientes para estos estándares.

Sus propiedades más importantes son:

- Estándar 802.11g wireless LAN
- Tasa de transmisión de hasta 54 Mbps
- Tasa de throughput 8 veces mayor que dispositivo 802.11b
- Compatibilidad con dispositivos 802.11b
- Tasa de escalamiento dinámico a 1, 2, 5.5, 6, 9, 11, 12,18, 22, 24, 36, 48, y 54Mbps
- Máxima confiabilidad, throughput y conectividad con conmutación de tasa de datos automática
- Encriptación WEP de 64/128/256 bits elegible por el usuario

- Cuatro modos de operación: access point, puente inalámbrico punto a punto, puente inalámbrico punto a multipunto y como cliente inalámbrico
- Puerto 10/100BASE-TX LAN con MDI/MDIX automático
- Cliente/Servidor DHCP
- Configuración por Web
- Utilidad de administración soporta Win 98 SE, 2000, XP, ME

- **Especificaciones técnicas**

  - **Hardware**

  Estándares: 802.11b, 802.11b+, 802.11g
  Tasas de transmisión: hasta 11 Mbps con 802.11b
  Tecnología de transmisión: DSSS
  Rango de radio frecuencias: 2.4 a 2.4835 GHz
  Canales de operación: de 1 a 11 canales
  Antenas: una antena dipolo externa de 2 dBi de ganancia, una antena interna

Potencia de salida del transmisor: bajo 802.11b, 16 dBm
Rango de operación: interiores: 100 metros, exteriores: 400 metros

- **Físico y ambiental**

Fuente de poder: 5 VDC+/-5%, 2.5 A
Dimensiones: 142 x 109 x 31 mm
Temperatura de operación: de 0 a 55 °C
Humedad: de 5% hasta 95%

B.1.2 DWL – 6000AP. De la empresa fabricante DLink, este punto de acceso es capaz de operar simultáneamente bajo los estándares 802.11a y 802.11b. Con once canales que no se traslapan, este se convierte en una solución para que diferentes usuarios tengan acceso a la red ubicando sus equipos inalámbricos en frecuencias diferentes.

- **Especificaciones técnicas**

  - **Hardware**

  Estándares: 802.11, 802.11a, 802.11b, 802.11d, 802.3, 802.3u
  Tasas de transmisión: 1, 2, 5.5, 11 y 22 Mbps con 802.11b
  Tecnología de transmisión: DSSS
  Rango de radio frecuencias: 2.4 a 2.462 GHz
  Canales de operación: de 1 a 11 canales
  Antenas: una antena dipolo dual de ganancia 5 dBi
  Potencia de salida del transmisor: 15 dBm ± 2 dB
  Rango de operación: interiores: 100 metros, exteriores: 300 metros

  - **Físico y ambiental**

  Fuente de poder: 5 VDC, 2.5 A
  Dimensiones: 91.2 x 54 x 36.4 mm
  Temperatura de operación: de 0 a 55 °C
  Humedad: de 5% hasta 95%

B.2 TARJETAS INALÁMBRICAS

B.2.1 DWL – 120. Esta tarjeta cliente inalámbrica tiene la capacidad de trabajar correctamente con equipos de red que operan bajo el estándar 802.11b.

- **Especificaciones técnicas**

  Tasas de transmisión: 11, 5.5, 2 y 1 Mbps

Rango de operación: en interiores hasta 70 m, en exteriores hasta 300 m.

Fuente de poder: 5 V ± 5%, 350 mA

Potencia de transmisión: 13 dBm

B.2.2 Orinoco USB client. Este adaptador puede conectar un computador a cualquier red operando bajo el estándar 802.11b y alcanza una tasa de transmisión hasta de 11 Mbps a una banda frecuencia de 2.4 GHz, es decir, una banda libre.

- **Especificaciones técnicas**

Interfaz: USB

Canales de frecuencia: 2.4 – 2.4835 GHz

Técnicas de modulación: DSSS

Potencia de salida nominal: 15 dBm
Dimensiones: 13 x 5.7 x 9 cm
Temperatura: 0 – 55 °C
Humedad: 95%
Fuente de poder: 4.75 a 5.25 VDC

B.3 EQUIPOS DE CÓMPUTO

B.3.1 IBM ThinkPad R50

El equipo de cómputo usado fue el IBM ThinkPad R50 al cual fue conectada la tarjeta inalámbrica, tanto en la etapa de campaña de medición como en la actividad de pruebas de campo. Se hizo uso del puerto USB que éste posee, el sistema operativo Windows y el software Netstumbler por medio de este computador portátil.

- **Especificaciones técnicas**
  Procesador: Intel Pentium M de 1.4 GHz
  RAM: 512 MB
  Caché: 512 KB
  Disco duro: 30 GB
  Sistema operativo: Windows XP Professional

ANEXO C

NETWORK STUMBLER v.0.4.0

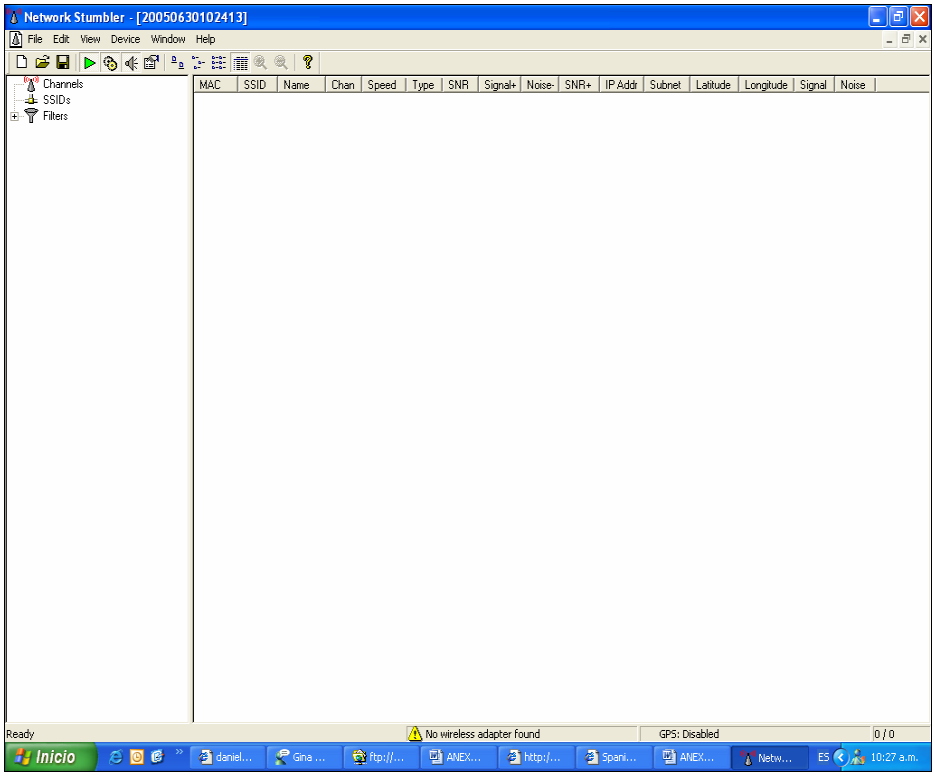
Network Stumbler (Netstumbler) es un software libre de costo, licencia y sin garantía. *Netsumbler* es una herramienta para Windows que permite detectar redes de área local inalámbricas (WLANs) que usan 802.11b, 802.11a y 802.11g. *Netstumbler* posee varios usos, entre ellos:

- Verificar que la red este instalada de la manera que el usuario requiere.
- Encontrar ubicaciones con pobre cobertura dentro de la WLAN del usuario.
- Detectar otras redes que podrían causar interferencia a la red.
- Detectar *APs* no autorizados dentro del lugar de trabajo.
- Ayuda a antenas direccionales auxiliares para enlaces WLAN de largo trayecto.

C.1 VENTANA

Esta ventana posee en su parte superior un menú general de Windows donde están *File*, *Edit*, *View*, *Device*, *Window y Help*. El submenú más importante para este proyecto, de los antes mencionados es *Device*, puesto que es allí donde se selecciona el dispositivo receptor que envía

los datos a *Nestumbler*, que para nuestro caso han sido las tarjetas inalámbricas conectadas al puerto *USB*. Debajo de estos submenús se encuentran unos enlaces rápidos a algunas funciones contenidas en dichos submenús, como en cualquier otra herramienta de *Windows.*



En la parte izquierda de la pantalla, se encuentran tres enlaces: *Channels*, *SSIDs*, *Filters*. Al dar clic sobre *Channels*, se despliegan los números de los canales que están siendo usados, que a su vez, al hacer clic en alguno de ellos, muestran la dirección *MAC* del equipo que está ocupando el respectivo canal. Ahora, al dar clic en *SSDIs*, se muestran los nombres de las redes que son 'administradas' por el *AP*; cuando se da clic sobre uno de esos nombres, nuevamente se despliegan las direcciones *MAC*

correspondientes a cada uno de los *APs*. Y finalmente, cuando se da clic sobre *Filters*, se despliegan opciones de encriptado y tipos de filtros que se pueden implementar para dar seguridad a las redes en estudio.

En la mayor parte de la ventana está la 'zona de trabajo', donde *Netstumbler* muestra la información. La ventana esta organizada por campos que están en la parte superior. Los campos más importantes son:

- *MAC*: muestra la dirección *MAC* del *AP*.
- *SSIDs*: muestra los identificadores de las redes pertenecientes a cada uno de los *APs*.
- *Channel*: muestra el número del canal en el que está la red del AP correspondiente.
- *Speed*: muestra la tasa de transmisión de los datos de la conexión.
- *SNR*: valor actual de relación señal a ruido.
- *Signal+*: muestra la máxima potencia en dBm de señal registrada hasta el momento.
- *Noise-*: muestra el mínimo valor de ruido en dBm registrado hasta el momento.
- *SNR+*: máximo valor de relación señal a ruido registrado.
- *First seen*: hora del primer registro.
- *Last seen*: hora del último registro.
- *Signa*l: valor actual de potencia en dBm.
- *Noise*: valor actual de ruido en dBm.

Adicionalmente, en la parte inferior de la pantalla se presenta el número de *APs* activos, si el *GPS* está habilitado y un racional que indica cuántos *APs* están activos y de cuántos se están haciendo registros.

ANEXO D

ANEXO CÓDIGOS

A continuación se presentan las líneas de código de *MatIab* utilizadas para encontrar tanto la solución inicial como para la solución usando los datos experimentales.

D.1 SOLUCIÓN TEÓRICA

D.1.1 myfunteo.m

```
%función implementada en la solución de la posición

function F = myfunteo(x)
id=fopen('texto.txt','rt');
mat= fscanf(id,'%f',2);
fclose(id);
p1=mat(1);
p2=mat(2);
r1=13-(32.4+20*log10(x(1)/1000)+20*log10(2437))-p1;
r2=13-(32.4+20*log10(x(2)/1000)+20*log10(2437))-p2;
F=[r1; r2];
```

D.1.2 solucteo.m

```
%solución de la posición
```

```
clear all

%Entrada de valores de potencia desde teclado
p1=input('potencia1 ');
p2=input('potencia2 ');
id=fopen('texto.txt','wt');
fprintf(id,'%6.2f  %6.2f',p1,p2);
fclose(id);

%solución de distancias (método iterativo)
x0 = [25;25];
%options = optimset('Jacobian','on');
options=optimset('Display','iter','TolFun',1e-100);
[x,fval] = fsolve(@myfunteo,x0,options)

%solución de las coordenadas
d1=x(1);
d2=x(2);
[x,y]=solve('d1^2=x^2+y^2','d2^2=x^2+(50-y)^2');
x=eval(x)
y=eval(y)
```

D.2 RED NEURONAL

D.2.1 Patrones de entrada tomados de la ecuación de perdidas en espacio libre

- **Matrices para el entrenamiento**

```
%En este código se crean las matrices para el entrenamiento de
%la red  La matrices que contienen las potencias de los Access
%Points se generan a partir de la ecuación de perdidas en
%espacio libre.
```

```
clc
clear all
%crea la matriz de puntos
for j=1:51;
       for k=1:51;
          A(k,j)=(k-1)+i*(j-1);
          dap1(k,j)=abs(A(k,j));
       end
end

%Matriz de potencias creada a partir de la ecuación de
pérdidas
for j=1:51;
    for k=1:51;
   ap1(k,j)=13-(32.4+20*log10(dap1(k,j)/1000)+20*log10(2437));
    end
end
%Matrices  de potencia reflejadas
ap2=fliplr(ap1);

%MATRICES POTENCIA Y POSICIÓN TOMADAS CADA 3 METROS
c=linspace(1,49,17);
c(1,18)=51;
A_1=A(c,:);
A_=A_1(:,c);
p1a=ap1(c,:);
p1=p1a(:,c);
p2a=ap2(c,:);
p2=p2a(:,c);
%coordenadas X-Y
coordx=real(A_);
coordy=imag(A_);
```

```
%Construcción de matrices de potencia y posición con el 80% de
%los datos para el entrenamiento y el 20% para la simulación
b=linspace(4,16,4);
d=[1 2 3 5 6 7 9 10 11 13 14 15 17 18];
%Matriz para entrenamiento
rssi1_1=p1(d,:);
rssi2_1=p2(d,:);
cx1=coordx(d,:);
cy1=coordy(d,:);
%Matriz para simulación
rssi1_2=p1(b,:);
rssi2_2=p2(b,:);
cx2=coordx(b,:);
cy2=coordy(b,:);
save Matrizteo
```

- **Código de la red neuronal**

```
%            ENTRENAMIENTO CON LA ECUACION TEÓRICA
%Programa para el entrenamiento de cada una de las redes para
%predecir la posición del dispositivo móvil, a partir de la
%ecuación de perdidas en espacio libre.

clc
clear all
load Matrizteo

pap1=reshape(p1,[],1);
pap2=reshape(p2,[],1);
pap=[pap1,pap2];
x=reshape(coordx,[],1);
```

```
y=reshape(coordy,[],1);
meanp=mean(pap);
meanp=meanp';
stdp=std(pap);
stdp=stdp';
meanx=mean(x);
meanx=meanx';
meany=mean(y);
meany=meany';
stdx=std(x);
stdx=stdx';
stdy=std(y);
stdy=stdy';

%Construcción de los dos vectores fila de coordenadas y
%potencias para el entrenamiento
vec1=reshape(rssi1_1,1,[]);
vec2=reshape(rssi2_1,1,[]);
vec=[vec1;vec2]; % Matriz de potencias para el entrenamiento
x1=reshape(cx1,1,[]);
y1=reshape(cy1,1,[]);

%Normalización de los datos para el entrenamiento
[pexp] = trastd(vec,meanp,stdp);
[x1n] = trastd(x1,meanx,stdx);
[y1n] = trastd(y1,meany,stdy);

%Construcción de los dos vectores fila de coordenadas y
%potencias para la simulación
pexp1=reshape(rssi1_2,1,[]);
pexp2=reshape(rssi2_2,1,[]);
Pexp_=[pexp1;pexp2];
x2=reshape(cx2,1,[]);
```

```
y2=reshape(cy2,1,[]);

%Normalización vectores potencia para simulación
[psim] = trastd(Pexp_,meanp,stdp);

   %Entrenamiento de la red con los datos de la ecuación de
                          pérdidas

% Predicción de la coordenada X
netexp1a=newff(minmax(pexp),[2 16 1],. . .
{'tansig','tansig','purelin'},'trainlm');
netexp1a=init(netexp1a);
netexp1a.trainParam.epochs=1000;
netexp1a.trainParam.goal=5e-3;
netexp1a.trainParam.lr=0.05;
netexp1a.trainParam.min_grad=1e-030
netexp1a=train(netexp1a,pexp,x1n);
netexp1a_1=netexp1a;
save netexp1a_1

% Predicción de la coordenada Y
netexp2a=newff(minmax(pexp),[2 16 1],. . .
{'tansig','tansig','purelin'},'trainlm');
netexp2a.trainParam.epochs=1000;
netexp2a.trainParam.goal=5e-3;
netexp2a.trainParam.lr=0.05;
netexp2a.trainParam.min_grad=1e-030
netexp2a=train(netexp2a,pexp,y1n);
netexp2a_1=netexp2a
save netexp2a_1

% Comprobación  del modelo neuronal
anx1=sim(netexp1a,pexp);
```

```
[asx1] = poststd(anx1,meanx1,stdx1);
errorx1=abs(asx1-x1);
errorx=[min(min(errorx1)),max(max(errorx1))]

anx2=sim(netexp1a,psim);
[asx2] = poststd(anx2,meanx1,stdx1);
errorx2=abs(asx2-x2);
errorx=[min(min(errorx2)),max(max(errorx2))]



any1=sim(netexp2a,pexp);
[asy1] = poststd(any1,meany1,stdy1);
errory1=abs(asy1-y1);
errory=[min(min(errory1)),max(max(errory1))]

any2=sim(netexp2a,psim);
[asy2] = poststd(any2,meany1,stdy1);
errory2=abs(asy2-y2);
errory=[min(min(errory2)),max(max(errory2))]

save wirelocteo
```

- **Solución teórica aplicando el modelo neuronal**

```
%Programa para determinar la ubicación del dispositivo móvil
%mediante la red neuronal entrenada con los datos obtenidos de
%la ecuación de pérdidas

clear all
load mean_std
load netexp1a_1
load netexp2a_1
```

```
%Wirelocteo= Variable que contiene los parámetros resultantes
%del entrenamiento de la red
% netexp1a_1= red entrenada para predecir la coordenada X
% netexp2a_1= red entrenada para predecir la coordenada Y
% mean_std= Contiene los datos estadísticos de la
normalización

qr1=input('potencia1 ');
qr2=input('potencia2 ');
qr=[qr1;qr2];
[potencia] = trastd(qr,meanp,stdp);

x1=sim(netexp1a_1,potencia);
[X]=abs(poststd(x1,meanx,stdx))

y1=sim(netexp2a_1,potencia);
[Y]=abs(poststd(y1,meany,stdy))
```

D.2.2 Patrones de entrada tomados de lo datos experimentales

- **Matrices para el entrenamiento**

```
%                    MATRICES EXPERIMENTALES
%En este código se crean las matrices para el entrenamiento de
%la red a partir de los datos de potencia emitida por los
Access %Points obtenidos en las mediciones da campo.

clc
clear all
A=zeros(51);
dAP1=zeros(51);
dAP2=zeros(51);
```

```
%Estas son las hipermatices con los datos experimentales.
%hap1(x,y,z)=Hipermatriz del AP1, en z están los 5 datos
%experimentales correspondientes a las potencias tomadas en
cada %punto de coordenada X-Y
hap1(x,y,z)= Hipermatriz de datos experimentales tomados del
AP DWL-6000
hap2(x,y,z)= Hipermatriz de datos experimentales tomados del
AP DWL-2000
hap1=(-1)*hap1;
hap2=(-1)*hap2;
media1=mean(hap1,3);
media2=mean(hap2,3);

%crea la matriz de puntos
for j=1:51;
       for k=1:51;
          A(k,j)=(k-1)+i*(j-1);
       end
end

%MATRICES DE LA POSICION TOMADAS CADA 3 METROS
c=linspace(1,49,17);
c(1,18)=51;
A_1=A(c,:);
A_=A_1(:,c);
%coordenadas X-Y
coordx=real(A_);
coordy=imag(A_);

%Construcción de matrices de potencia y distancia con el 80%
de %los datos para el entrenamiento y el 20% para la
simulación %(DATOS EXPERIMENTALES)
```

```
b=linspace(4,16,4);
d=[1 2 5 6 7 9 11 13 15 17 18];
%Matriz para entrenamiento
rssi1_1=media1(d,:);
rssi2_1=media2(d,:);
cx1=coordx(d,:);
cy1=coordy(d,:);
%Matriz para simulación
rssi1_2=media1(b,:);
rssi2_2=media2(b,:);
cx2=coordx(b,:);
cy2=coordy(b,:);

save Matrizexp
```

- **Código de la red neuronal**

```
% ENTRENAMIENTO DE LA RED NEURONAL
clc
clear all
load Matrizexp

pap1=reshape(media1,[],1);
pap2=reshape(media2,[],1);
pap=[pap1,pap2];
x=reshape(coordx,[],1);
y=reshape(coordy,[],1);
meanp=mean(pap);
meanp=meanp';
stdp=std(pap);
stdp=stdp';
meanx=mean(x);
meanx=meanx';
```

```
meany=mean(y);
meany=meany';
stdx=std(x);
stdx=stdx';
stdy=std(y);
stdy=stdy';

%Construcción de los dos vectores fila de coordenadas y
%potencias experimentales para el entrenamiento
vec1=reshape(rssi1_1,1,[]);
vec2=reshape(rssi2_1,1,[]);
vec=[vec1;vec2]; % Matriz de potencias para el entrenamiento
x1=reshape(cx1,1,[]);
y1=reshape(cy1,1,[]);
%Normalización de los datos experimentales para el
entrenamiento
[pexp] = trastd(vec,meanp,stdp);
[x1n] = trastd(x1,meanx,stdx);
[y1n] = trastd(y1,meany,stdy);

%Construcción de los dos vectores fila de distancias,
%coordenadas y potencias experimentales para la simulación
pexp1=reshape(rssi1_2,1,[]);
pexp2=reshape(rssi2_2,1,[]);
Pexp_=[pexp1;pexp2];
x2=reshape(cx2,1,[]);
y2=reshape(cy2,1,[]);
%Normalización vectores potencia para simulación
[psim] = trastd(Pexp_,meanp,stdp);

    %Entrenamiento de la red con los datos experimentales
% Predicción de la coordenada X
netexp1=newff(minmax(pexp),[2 16 1], . . .
```

```
{'tansig','tansig','purelin'},'trainlm');
netexp1=init(netexp1);
netexp1.trainParam.epochs=1000;
netexp1.trainParam.goal=0.3;
netexp1.trainParam.lr=0.005;
netexp1.trainParam.min_grad=1e-030
netexp1_1=netexp1;
save netexp1_1

% Predicción de la coordenada Y
netexp2=newff(minmax(pexp),[2 16 1], . . .
{'tansig','tansig','purelin'},'trainlm');
netexp2.trainParam.epochs=1000;
netexp2.trainParam.goal=0.1949;
netexp2.trainParam.lr=0.005;
netexp2.trainParam.min_grad=1e-030
netexp2=train(netexp2,Pexp,dist2a);
netexp2_1=netexp2;
save netexp2_1

% Comprobación del modelo neuronal
anx1=sim(netexp1,pexp);
[asx1] = poststd(anx1,meanx1,stdx1);
errorx1=abs(asx1-x1);
errorx=[min(min(errorx1)),max(max(errorx1))]

anx2=sim(netexp1,psim);
[asx2] = poststd(anx2,meanx1,stdx1);
errorx2=abs(asx2-x2);
errorx=[min(min(errorx2)),max(max(errorx2))]

any1=sim(netexp2,pexp);
[asy1] = poststd(any1,meany1,stdy1);
```

```
errory1=abs(asy1-y1);
errory=[min(min(errory1)),max(max(errory1))]

any2=sim(netexp2,psim);
[asy2] = poststd(any2,meany1,stdy1);
errory2=abs(asy2-y2);
errory=[min(min(errory2)),max(max(errory2))]

save wirelocexp
```

- **Solución real aplicando el modelo neuronal**

```
%Programa para determinar la ubicación del dispositivo móvil
%mediante la %red neuronal entrenada con los datos obtenidos
de %las mediciones de campo

clear all
load meanstd
load netexp1_1
load netexp2_1

%Wirelocteo= Variable que contiene los parámetros resultantes
%del entrenamiento de la red
% netexp1_1= red entrenada para predecir la coordenada X
% netexp2_1= red entrenada para predecir la coordenada Y
% meanstd= Contiene los datos estadísticos de la normalización

qr1=input('potencia1 ');
qr2=input('potencia2 ');
qr=[qr1;qr2];
[potencia] = trastd(qr,meanp,stdp);
[potencia] = trastd(potencia1,meanp,stdp);
```

```
x1=sim(netexp1_1,potencia);
[X]=abs(poststd(x1,meanx,stdx))

y1=sim(netexp2_1,potencia);
[Y]=abs(poststd(y1,meany,stdy))
```

D.2.3 Solución alternativa

- **Matrices para el entrenamiento**

```
%                 MATRICES EXPERIMENTALES REFLEJADAS
%En este código se crean las matrices para el entrenamiento de
%la red a partir de los datos de potencia emitida por uno de
%los Access Points obtenidos en las %mediciones da campo. Esta
%matriz de potencia se refleja a diferentes puntos de la
grilla %para simular la presencia de más Access Points, con el
fin de %dar una solución alternativa para el sistema de
localización.

A=zeros(51);
dAP1=zeros(51);
dAP2=zeros(51);

%Estas son las hipermatices con los datos experimentales.
%hap2(x,y,z)=Hipermatriz del AP1, en z están los 5 datos
%experimentales
%correspondientes a las potencias tomadas en cada punto de
%coordenada X-Y

hap2=(-1)*hap2;
ap2=mean(hap2,3);
%Matrices  de potencia reflejadas
ap1=fliplr(ap2);
```

```
ap3=flipud(ap1);
ap4=flipud(ap2);

%crea la matriz de puntos
for j=1:51;
       for k=1:51;
          A(k,j)=(k-1)+i*(j-1);
          dap1(k,j)=abs(A(k,j));
       end
end
dap2=fliplr(dap1);
dap3=flipud(dap1);
dap4=flipud(dap2);

%MATRICES DE POSICION TOMADAS CADA 3 METROS
c=linspace(1,49,17);
c(1,18)=51;
A_1=A(c,:);
A_=A_1(:,c);
%coordenadas X-Y
coordx=real(A_);
coordy=imag(A_);

%Construcción de matrices de potencia y distancia con el 80% de %los datos para el entrenamiento y el 20% para la simulación %(DATOS EXPERIMENTALE %REFLEJADOS)
b=linspace(4,16,4);
d=[1 2 3 5 6 7 9 10 11 13 14 15 17 18];
%Matriz para entrenamiento
rssi1_1=ap1(d,:);
rssi2_1=ap2(d,:);
rssi3_1=ap3(d,:);
rssi4_1=ap4(d,:);
```

```
dap1_1=dap1_(d,:);
dap2_1=dap2_(d,:);
dap3_1=dap3_(d,:);
dap4_1=dap4_(d,:);
cx1=coordx(d,:);
cy1=coordy(d,:);
%Matriz para simulación
rssi1_2=ap1(b,:);
rssi2_2=ap2(b,:);
rssi3_2=ap3(b,:);
rssi4_2=ap4(b,:);
dap1_2=dap1_(b,:);
dap2_2=dap2_(b,:);
dap3_2=dap3_(b,:);
dap4_2=dap4_(b,:);
cx2=coordx(b,:);
cy2=coordy(b,:);

save Matrizref
```

- **Código de la red neuronal**

```
%ENTRENAMIENTO DE LA RED NEURONAL CON LAS MATRICES DE POTENCIA
%REFLEJADAS A DIFERENTES PUNTOS DENTRO DE LA GRILLA
%Programa para el entrenamiento de cada una de las redes para
%predecir la posición del dispositivo móvil, a partir de
%matrices de potencia que %simulan la presencia de varios
Access Points
clc
clear all
load Matrizref

pap1=reshape(ap1,[],1);
```

```
pap2=reshape(ap2,[],1);
pap3=reshape(ap3,[],1);
pap4=reshape(ap4,[],1);
pap=[pap1,pap2,pap3,pap4];
x=reshape(coordx,[],1);
y=reshape(coordy,[],1);
meanp=mean(pap);
meanp=meanp';
stdp=std(pap);
stdp=stdp';
meanx=mean(x);
meanx=meanx';
meany=mean(y);
meany=meany';
stdx=std(x);
stdx=stdx';
stdy=std(y);
stdy=stdy';

%Construcción de los dos vectores fila de coordenadas y
%potencias experimentales para el entrenamiento
vec1=reshape(rssi1_1,1,[]);
vec2=reshape(rssi2_1,1,[]);
vec3=reshape(rssi3_1,1,[]);
vec4=reshape(rssi4_1,1,[]);
vec=[vec1;vec2;vec3;vec4]; % Matriz de potencias para el
%entrenamiento
x1=reshape(cx1,1,[]);
y1=reshape(cy1,1,[]);
%Normalización de los datos experimentales para el
entrenamiento
[pexp] = trastd(vec,meanp,stdp);
[x1n] = trastd(x1,meanx,stdx);
```

```
[y1n] = trastd(y1,meany,stdy);

%Construcción de los dos vectores fila de coordenadas y
%potencias experimentales para la simulación
pexp1=reshape(rssi1_2,1,[]);
pexp2=reshape(rssi2_2,1,[]);
pexp3=reshape(rssi3_2,1,[]);
pexp4=reshape(rssi4_2,1,[]);
Pexp_=[pexp1;pexp2;pexp3;pexp4];
dist1b=reshape(dap1_2,1,[]);
dist2b=reshape(dap2_2,1,[]);
dist3b=reshape(dap3_2,1,[]);
dist4b=reshape(dap4_2,1,[]);
x2=reshape(cx2,1,[]);
y2=reshape(cy2,1,[]);
%Normalización vectores potencia para simulación
[psim] = trastd(Pexp_,meanp,stdp);



    % Entrenamiento de la red con los datos experimentales
                          reflejados
% Predicción de la coordenada X
netexp1=newff(minmax(pexp),[4 16 1], . . .
{'tansig','tansig','purelin'},'trainlm');
netexp1.trainParam.epochs=1000;
netexp1.trainParam.goal=0.0487;
netexp1.trainParam.lr=0.05;
netexp1.trainParam.min_grad=1e-030
netexp1=train(netexp1,pexp,x1n);
netexp1_2=netexp1;
save netexp1_2
```

```
% Predicción de la coordenada Y
netexp2=newff(minmax(pexp),[4 16 1], . . .
{'tansig','tansig','purelin'},'trainlm');
netexp2.trainParam.epochs=1000;
netexp2.trainParam.goal=0.0243;
netexp2.trainParam.lr=0.05;
netexp2.trainParam.min_grad=1e-030
netexp2=train(netexp2,pexp,y1n);
netexp2_2=netexp2;
save netexp2_2

anx1=sim(netexp1,pexp);
[asx1] = poststd(anx1,meanx1,stdx1);
errorx1=abs(asx1-x1);
ex1=[min(min(errorx1)),max(max(errorx1))]

anx2=sim(netexp1,psim);
[asx2] = poststd(anx2,meanx1,stdx1);
errorx2=abs(asx2-x2);
ex2=[min(min(errorx2)),max(max(errorx2))]

any1=sim(netexp2,pexp);
[asy1] = poststd(any1,meany1,stdy1);
errory1=abs(asy1-y1);
ey1=[min(min(errory1)),max(max(errory1))]

any2=sim(netexp2,psim);
[asy2] = poststd(any2,meany1,stdy1);
errory2=abs(asy2-y2);
ey2=[min(min(errory2)),max(max(errory2))]

save wirelocref
```

- **Solución alternativa aplicando el modelo neuronal**

```
%Programa para determinar la ubicación del dispositivo móvil
%mediante la red neuronal entrenada con los datos
experimentales %reflejados a diferentes puntos dentro de la
grilla
clear all
load meanstd
load netexp1_1
load netexp2_1

%Wirelocteo= Variable que contiene los parámetros resultantes
%del entrenamiento de la red
% netexp1_2= red entrenada para predecir la coordenada X
% netexp2_2= red entrenada para predecir la coordenada Y
% meanstd= Contiene los datos estadísticos de la normalización

qr1=input('potencia1 ');
qr2=input('potencia2 ');
qr3=input('potencia3 ');
qr4=input('potencia4 ');

qr=[qr1;qr2;qr3;qr4];
[potencia] = trastd(qr,meanp,stdp);
[potencia] = trastd(potencia1,meanp,stdp);

x1=sim(netexp1_2,potencia);
[X]=abs(poststd(x1,meanx,stdx))

y1=sim(netexp2_2,potencia);
[Y]=abs(poststd(y1,meany,stdy))
```

D.3 Algoritmo de búsqueda

D.3.1 locali_.m

```
function locali (valp)

close all;
load media1_;
load media2;
load stdmax;
load A_;

pot1=input('potencia1 ');
pot2=input('potencia2 ');

valp=[pot1 pot2];

desviacion=stdmax;
k=input('numero de desviaciones estandar ');
figure
subplot(1,2,1)
ap1=wlocal(abs(media1),k,desviacion,A_);
subplot(1,2,2)
ap2=wlocal(abs(media2),k,desviacion,A_);

load vector;

for k=1:length(valp)
    for z=2:length(vector)
        if valp(k)<=vector(z)
            valpz(k)=(z-1)*10;
            break
        end
    end
```

```
end

ap1 = potmedia(ap1,valpz(1));
ap2 = potmedia(ap2,valpz(2));

mapotg=ap1.*ap2.*50;
sparse(mapotg)

figure
% Gráfica de Intervalos
x=real(puntos);
y=imag(puntos);
z=mapotg;
X=[0:1:50];
Y=[0:1:50];
[X,Y] = meshgrid(X,Y);
Z=griddata(x,y,z,X,Y,'cubic');
pcolor(X,Y,Z);
colorbar

%----------------
function ap = potmedia (amedia,zona)
for i=1:length(amedia)
    for j=1:length(amedia)
        if amedia(i,j)==zona
            ap(i,j)=1;
        else
            ap(i,j)=0;
        end
    end
end
```

D.3.2 locali.m

```
function locali (valp)

close all;
load media1;
load media2;
load stdmax;
load A_;

pot1=input('potencia1 ');
pot2=input('potencia2 ');

valp=[pot1 pot2];

desviacion=stdmax;
k=input('numero de desviaciones estandar ');
figure
subplot(1,2,1)
ap1=wlocal(abs(media1_),k,desviacion,A_);
subplot(1,2,2)
ap2=wlocal(abs(media2),k,desviacion,A_);

load vector;

for k=1:length(valp)
    for z=2:length(vector)
        if valp(k)<=vector(z)
            valpz(k)=(z-1)*10;
            break
        end
    end
end
```

```
ap1 = potmedia(ap1,valpz(1));
ap2 = potmedia(ap2,valpz(2));

mapotg=ap1.*ap2.*50;
sparse(mapotg)

figure
% Gráfica de Intervalos
x=real(puntos);
y=imag(puntos);
z=mapotg;
X=[0:1:50];
Y=[0:1:50];
[X,Y] = meshgrid(X,Y);
Z=griddata(x,y,z,X,Y,'cubic');
pcolor(X,Y,Z);
colorbar

%----------------
function ap = potmedia (amedia,zona)
for i=1:length(amedia)
    for j=1:length(amedia)
        if amedia(i,j)==zona
            ap(i,j)=1;
        else
            ap(i,j)=0;
        end
    end
end
```

D.3.3 locali3.m

```
function locali (valp)

close all;
load media1;
load media2;
load media3;
load stdmax;
load A_;

pot1=input('potencia1 ');
pot2=input('potencia2 ');
pot3=input('potencia3 ');

valp=[pot1 pot2 pot3];

desviacion=stdmax;
k=input('numero de desviaciones estandar ');
figure
subplot(2,2,1)
ap1=wlocal(abs(media1),k,desviacion,A_);
subplot(2,2,2)
ap2=wlocal(abs(media2),k,desviacion,A_);
subplot(2,2,3)
ap3=wlocal(abs(media3),k,desviacion,A_);

load vector;

for k=1:length(valp)
    for z=2:length(vector)
        if valp(k)<=vector(z)
            valpz(k)=(z-1)*10;
```

```
            break
        end
    end
end

ap1 = potmedia(ap1,valpz(1));
ap2 = potmedia(ap2,valpz(2));
ap3 = potmedia(ap3,valpz(3));

mapotg=ap1.*ap2.*ap3.*50
sparse(mapotg)

figure
% Gráfica de Intervalos
x=real(puntos);
y=imag(puntos);
z=mapotg;
X=[0:1:50];
Y=[0:1:50];
[X,Y] = meshgrid(X,Y);
Z=griddata(x,y,z,X,Y,'cubic');
pcolor(X,Y,Z);
colorbar

%----------------
function ap = potmedia (amedia,zona)
for i=1:length(amedia)
    for j=1:length(amedia)
        if amedia(i,j)==zona
            ap(i,j)=1;
        else
            ap(i,j)=0;
        end
```

```
    end
end
```

D.3.4 locali4.m

```
function locali (valp)

close all;
load media1;
load media2;
load media3;
load media4;
load stdmax;
load A_;

pot1=input('potencia1 ');
pot2=input('potencia2 ');
pot3=input('potencia3 ');
pot4=input('potencia4 ');

valp=[pot1 pot2 pot3 pot4];

desviacion=stdmax;
k=input('entre el numero de desviaciones estandar que desee
');
figure
subplot(2,2,1)
ap1=wlocal(abs(media1),k,desviacion,A_);
subplot(2,2,2)
ap2=wlocal(abs(media2),k,desviacion,A_);
subplot(2,2,3)
ap3=wlocal(abs(media3),k,desviacion,A_);
subplot(2,2,4)
```

```
ap4=wlocal(abs(media4),k,desviacion,A_);

load vector;

for k=1:length(valp)
    for z=2:length(vector)
        if valp(k)<=vector(z)
            valpz(k)=(z-1)*10;
            break
        end
    end
end

ap1 = potmedia(ap1,valpz(1));
ap2 = potmedia(ap2,valpz(2));
ap3 = potmedia(ap3,valpz(3));
ap4 = potmedia(ap4,valpz(4));

mapotg=ap1.*ap2.*ap3.*ap4.*50
sparse(mapotg)

figure
% Gráfica de Intervalos
x=real(puntos);
y=imag(puntos);
z=mapotg;
X=[0:1:50];
Y=[0:1:50];
[X,Y] = meshgrid(X,Y);
Z=griddata(x,y,z,X,Y,'cubic');
pcolor(X,Y,Z);
colorbar
```

```
%---------------
function ap = potmedia (amedia,zona)
for i=1:length(amedia)
    for j=1:length(amedia)
        if amedia(i,j)==zona
            ap(i,j)=1;
        else
            ap(i,j)=0;
        end
    end
end
```

D.3.5 wlocal.m

```
function mediag =wlocal(media,num,std,puntos)

% Definición de Intervalos según Stdv
vector=media(1,1);
val=0;
maximo=0;
while maximo<=100
    maximo=val+std*num;
    vector = [vector maximo];
    val = maximo;
end
% Asignación de Intervalos
for i=1:length(media)
    for j=1:length(media)
        for z=2:length(vector)
            if media(i,j)<=vector(z)
                mediag(i,j)=(z-1)*10;
                break
            end
```

```
        end
    end
end

save vector;

% Gráfica de Intervalos
x=real(puntos);
y=imag(puntos);
z=mediag;
X=[0:1:50];
Y=[0:1:50];
[X,Y] = meshgrid(X,Y);
Z=griddata(x,y,z,X,Y,'cubic');
pcolor(X,Y,Z);
colorbar
```

ANEXO E

REDES NEURONALES ARTIFICIALES (ANS)

Existen problemas que se resuelven muy bien por medio de un computador y otros en los que el cerebro es mucho más eficiente. Los computadores fueron creados para resolver tareas de alto nivel, como el razonamiento o el cálculo, que son fácilmente resolubles mediante el procesamiento de símbolos; en este tipo de tareas nuestro cerebro actúa en clara desventaja frente a la electrónica. Sin embargo, en tareas de procesamiento de bajo nivel, como las de razonamiento de patrones, percepción, control, etc., los computadores se desenvuelven todavía torpemente. Para esta clase de problemas, esenciales para la supervivencia de un organismo vivo, la naturaleza encontró una excelente solución: el procesamiento autoorganizado que emerge de la interacción de numerosos procesos elementales.

Esta dificultad de los sistemas de cómputo que trabajan bajo la filosofía de los sistemas secuenciales, desarrollados por *Von Neuman*, ha hecho que un gran número de investigadores centren su atención en el desarrollo de nuevos sistemas de tratamiento de información, que permitan solucionar problemas cotidianos, tal como lo hace el cerebro humano; este órgano biológico cuenta con varias características deseables para cualquier sistema de procesamiento digital, tale como:

- Es robusto y tolerante a fallas, diariamente mueren neuronas sin afectar su desempeño.

- Es flexible, se ajusta a nuevos ambientes por medio de procesos de aprendizaje, no hay que programarlo.
- Puede manejar información difusa, con ruido o inconsistente.
- Es altamente paralelo.
- Es pequeño, compacto y consume poca energía.

Las redes neuronales artificiales so capaces de aprender de la experiencia a partir de las señales o datos provenientes del exterior, dentro de un marco de computación paralela y distribuida, fácilmente implementable en dispositivos *hardware* específicos.

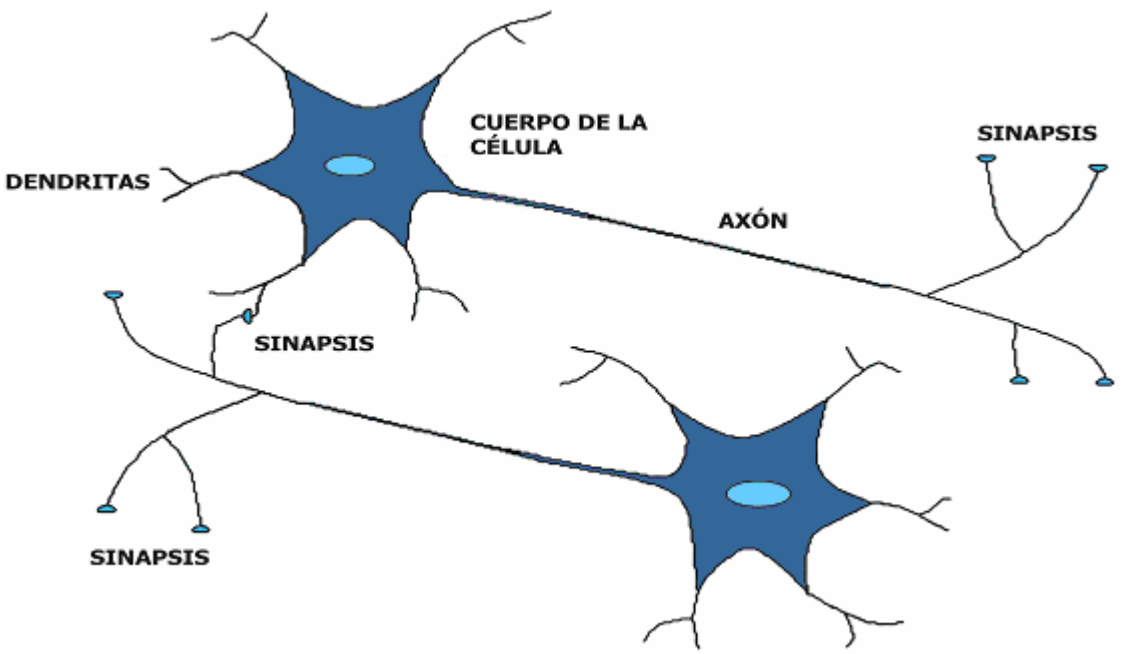
Basados en la eficiencia de los procesos llevados por el cerebro, varios investigadores han desarrollado la teoría de las **ANS**, las cuales emulan el comportamiento de las redes neuronales biológicas, y que se han utilizado para aprender estrategias de solución basadas en ejemplos de comportamiento típico de patrones; estos sistemas no requieren que la tarea a ejecutar se programe, ellos generalizan y aprenden de la experiencia.

Las **ANS** son una teoría que aun esta en proceso de desarrollo; aunque los investigadores han desarrollado potentes algoritmos de aprendizaje de gran valor práctico, las representaciones y procedimientos de que se sirve el cerebro, son aún desconocidas. Tarde o temprano los estudios computacionales del aprendizaje con **ANS** acabarán por converger a los métodos descubiertos por la evolución.

E.1 FUNCIONAMIENTO DE UNA NEURONA BIOLÓGICA

El cerebro consta de un gran número (aproximadamente 10^{11}) de elementos altamente interconectados (aproximadamente 10^4 conexiones

por elemento), llamados neuronas. Estas neuronas tienen tres componentes principales, las dendritas, el cuerpo de la célula o soma y el axón. Las dendritas, son el árbol receptor de la red, son como fibras nerviosas que cargan de señales eléctricas el cuerpo de la célula. El cuerpo de la célula, realiza la suma de esas señales de entrada. El axón es una fibra larga que lleva la señal desde el cuerpo de la célula hacia otras neuronas.



El punto de contacto entre el axón de una célula y una dendrita de otra célula es llamado sinapsis, la longitud de la sinapsis es determinada por la complejidad del proceso químico que estabiliza la función de la red neuronal. En el tipo de sinapsis más común no existe un contacto físico entre las neuronas, estas permanecen separadas por un vacío de unas 0.2 micras. Las sinapsis son direccionales, es decir, la información fluye siempre en un único sentido.

Algunas de las estructuras neuronales son determinadas en el nacimiento, otra parte es desarrollada a través del aprendizaje, proceso en que

nuevas conexiones neuronales son realizadas y otras se pierden por completo. Las estructuras neuronales continúan cambiando durante toda la vida, este cambio consisten en el refuerzo o debilitamiento de las uniones sinápticas.

Todas las neuronas conducen la información de forma similar, esta viaja a lo largo de axones en breves impulsos eléctricos, denominados potenciales de acción; los cuales alcanzan una amplitud máxima de unos 100 mV y duran 1 ms, son resultado del desplazamiento a través de la membrana celular de iones de sodio dotados de carga positiva.

La membrana en reposo mantiene un gradiente de potencial eléctrico de -70 mV, el signo negativo se debe a que el citoplasma intracelular está cargado negativamente respecto al exterior.

Alcanzado un potencial critico denominado umbral, la realimentación positiva produce un efecto regenerativo que obliga al potencial de membrana a cambiar de signo. Es decir, el interior de la célula se torna positivo con respecto al exterior, al cabo de 1ms, la permeabilidad del sodio decae y el potencial de membrana retorna a -70mV, su valor de reposo.

El fenómeno de generación de la señal nerviosa está determinado por la membrana y los iones presentes a ambos lados de ella. La membrana se comporta como un condensador, que se carga al recibir corrientes debidas a las especies iónicas presentes. En esencia, las especies iónicas más importantes, que determinan buena parte de la generación y propagación del impulso nervioso, son Na^+, K^+ y Ca^{2+}, además de los iones de proteínas.

La generación del potencial de acción, que al propagarse a lo largo del

axón da lugar a la transmisión eléctrica de la señal nerviosa. Después de un potencial de acción, la neurona sufre un período refractario, durante el cual no puede generarse uno nuevo. Un hecho importante es que el pulso generado es digital, en el sentido de que existe o no existe pulso, y todos ellos son de igual magnitud.

La intensidad de una sinapsis no viene representada por una cantidad fija, sino que puede ser modulada en una escala temporal mucho más amplia que la del disparo de las neuronas. Esta plasticidad sináptica se supone que constituye, al menos en buena medida, el aprendizaje.

E.2 MODELO DE UNA RED NEURONAL ARTIFICIAL

El modelo de una neurona artificial es una imitación del proceso de una neurona biológica. De la observación detallada del proceso biológico se han hallado los siguientes análogos con el sistema artificial:

- Un conjunto de **entradas** $x_j(t)$
- **Pesos sinápticos** de la neurona *i*, w_{ij} que representa la intensidad de interacción entre cada neurona presináptica *j* y la neurona postsináptica *i*.
- **Regla de propagación** $h_i(t) = \sigma(w_{ij}, x_j(t)$

  $$h_i(t) = \sum w_{ij} x_j$$

  que proporcione le valor del potencial postsináptico de la neurona *i* en función de sus pesos y entradas.
- **Función de activación** $y_i(t) = f_i(h_i(t))$ que representa simultáneamente la salida de la neurona y su estado de activación.

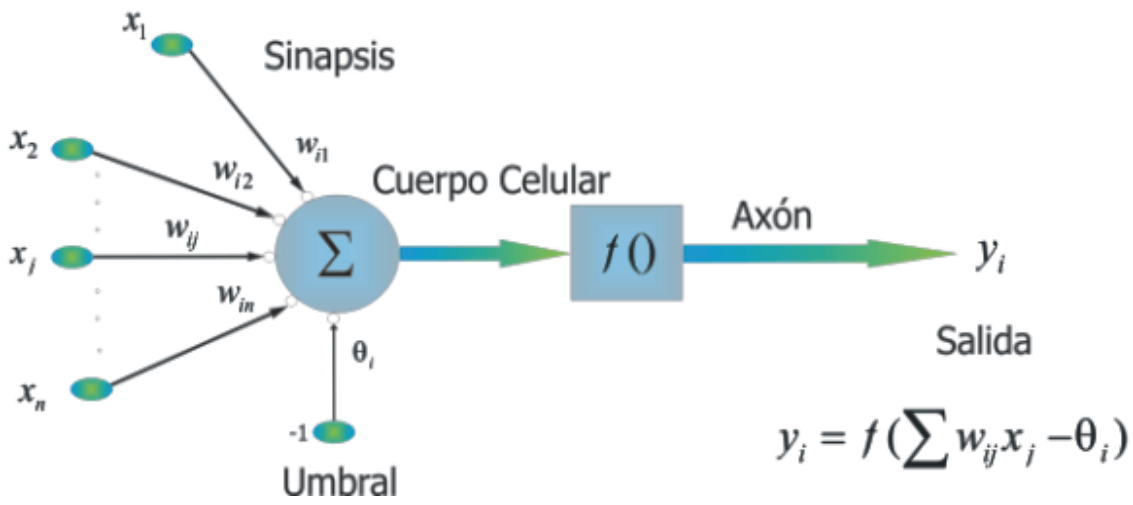
Con frecuencia se añade al conjunto de pesos de pesos de la neurona un

parámetro adicional θ_i, denominado umbral, que se resta del potencial postsináptico, por lo que el argumento de la función de activación queda:

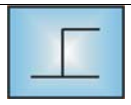
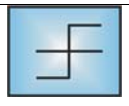
$$\sum w_{ij} x_j - \theta_i$$

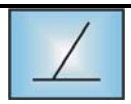
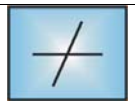
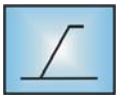
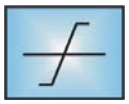
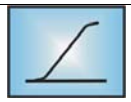
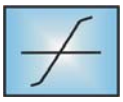
lo que representa añadir un grado de libertad adicional a la neurona. En conclusión, el modelo de neurona estándar queda

$$y_i(t) = f_i(\sum w_{ij} x_j - \theta_i)$$



E.3 FUNCIONES DE ACTIVACIÓN

Nombre	Relación Entrada / Salida	Icono	Función
Limitador Fuerte	a=0 n<0 a=1 n≥0		*Hardlim*
Limitador Fuerte Simétrico	a=-1 n<0 a=+1 n≥0		*Hardlims*

Lineal Positiva	a=0 n<0 a=n 0≤n		*Poslin*
Lineal	a=n		*Purelin*
Lineal Saturado	a=0 n<0 a=n 0≤n≤1 a=1 n>1		*Satlin*
Lineal Saturado Simétrico	a=-1 n<-1 a=n -1≤n≤1 a=+1 n>1		*Satlins*
Sigmoidal Logarítmico	$a = \frac{1}{1+e^{-n}}$		*Logsig*
Tangente Sigmoidal Hiperbólica	$a = \frac{e^{n} - e^{-n}}{e^{n} + e^{-n}}$		*Tansig*
Competitiva	a=1 *Neurona con n máx.* a=0 *El resto de neuronas*		*Compet*

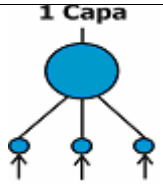
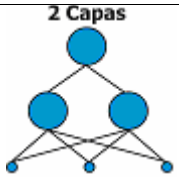
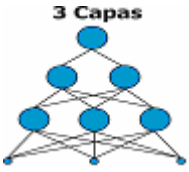
Dentro de una red neuronal, los elementos de procesamiento se encuentran agrupados por capas, de acuerdo a la ubicación de la capa dentro de la red, estas pueden ser, capa de entrada, capas ocultas o capa de salida. Puede haber redes de una sola capa con un número de *s* neuronas, o de varias capas (multicapa), donde cada capa tendrá su propia matriz de pesos y su propio vector de ganancias. Un tipo de redes, un poco diferente a las mencionadas anteriormente son las redes recurrentes, estas contienen una retroalimentación, es decir algunas de sus salidas son conectadas a sus entradas.

E.4 PRINCIPALES TIPOS DE REDES

E.4.1 Perceptrón. Este tipo de red fue inventada por el psicólogo *Frank Rosenblatt* en el año 1957. La estructura del perceptrón se inspira en las primeras etapas del procesamiento de los sistemas sensoriales de los animales.

La importancia del perceptrón radica en su carácter de dispositivo entrenable, pues determina automáticamente los pesos sinápticos que clasifican un determinado conjunto de patrones etiquetados.

El perceptrón es un tipo de red de aprendizaje supervisado, es decir necesita conocer los valores esperados de cada una de las entradas.

Estructura	Regiones de Decisión	Forma general De la región
	Medio Plano Limitado por un Hiperplano.	
	Regiones Cerradas o Convexas.	
	Complejidad Arbitraria Limitada por el Número de Neuronas.	

E.4.2 *Adaline*. Introducida por *Bernard Widrow* y su estudiante *Marcian Hoff* en 1959, cuyo nombre proviene de *Adaptive Linear Neuron*. Este modelo utiliza una neurona similar a la del perceptrón, pero de respuesta

lineal. La *Adaline* incorpora un parámetro adicional denominado *bias* (umbral) que proporciona un grado de libertad adicional.

La red *Adaline* utiliza la regla de aprendizaje de *Widrow-Hoff*, también conocida como regla LMS (*Least Mean Squares*), que conduce a actualizaciones de tipo continuo, siendo la actualización de los pesos proporcional al error que la neurona comete. Al igual que la perceptrón la red *Adaline* es una red de aprendizaje supervisado.

E.4.3 *Backpropagation*. Una solución al problema de entrenar los nodos de las capas ocultas pertenecientes a arquitecturas multicapa la proporciona el algoritmo de retropropagación de errores o BP (*Backpropagation*). Uno de los grandes avances logrados con la *Backpropagation* es que esta red aprovecha la naturaleza paralela de las redes neuronales para reducir el tiempo requerido por un procesador secuencial para determinar la correspondencia unos patrones dados.

La *backpropagation* es un tipo de red de aprendizaje supervisado, que emplea un ciclo propagación – adaptación de dos fases. Una vez se ha aplicado un patrón a la entrada de la red como estímulo, este se propaga desde la primera capa a través de las capas superiores de la red, hasta generar una salida. La señal de salida se compara con la salida deseada y se calcula una señal de error para cada una de las salidas.

Las salidas del error se propagan hacia atrás, partiendo de la capa de salida, hacia todas las neuronas de la capa oculta que contribuyen directamente a la salida. Este proceso se repite, capa por capa, hasta que todas las neuronas de la red hayan recibido una señal de error que describa su contribución relativa al error total. Basándose en la señal de error percibida, se actualizan los pesos de conexión de cada neurona, para hacer que la red converja hacia un estado que permita clasificar

correctamente todos los patrones de entrenamiento.

La importancia de este proceso consiste en que, a medida que se entrena la red, las neuronas de las capas intermedias se organizan a si mismas de tal modo que las distintas neuronas aprenden a reconocer distintas características del espacio total de entrada. Después del entrenamiento, cuando se les presente un patrón arbitrario de entrada que contenga ruido o que esté incompleto, las neuronas de la capa oculta de la red responderán con una salida activa si la nueva entrada contiene un patrón que se asemeje a aquella característica que las neuronas individuales hayan aprendido a reconocer durante su entrenamiento. Y a la inversa, las unidades de las capas ocultas tienden a inhibir su salida si el patrón de entrada no contiene la característica para reconocer, para la cual han sido entrenadas.

- **Regla de aprendizaje.** El algoritmo *backpropagation* para redes multicapa es una generalización del algoritmo LMS, ambos algoritmos realizan actualización de pesos y ganancias con base en el error medio cuadrático.

Las funciones de transferencia utilizadas en este tipo de red deben ser continuas para que su derivada exista en todo el intervalo, ya que las derivadas son requeridas para el cálculo del error.

Las funciones de transferencia más utilizadas son.

- ***logsig*** $f(n) = \frac{1}{1+e^{-n}}$
- ***tansig*** $f(n) = \frac{e^{n} - e^{-n}}{e^{n} + e^{-n}}$
- ***purelin*** $f(n) = n$

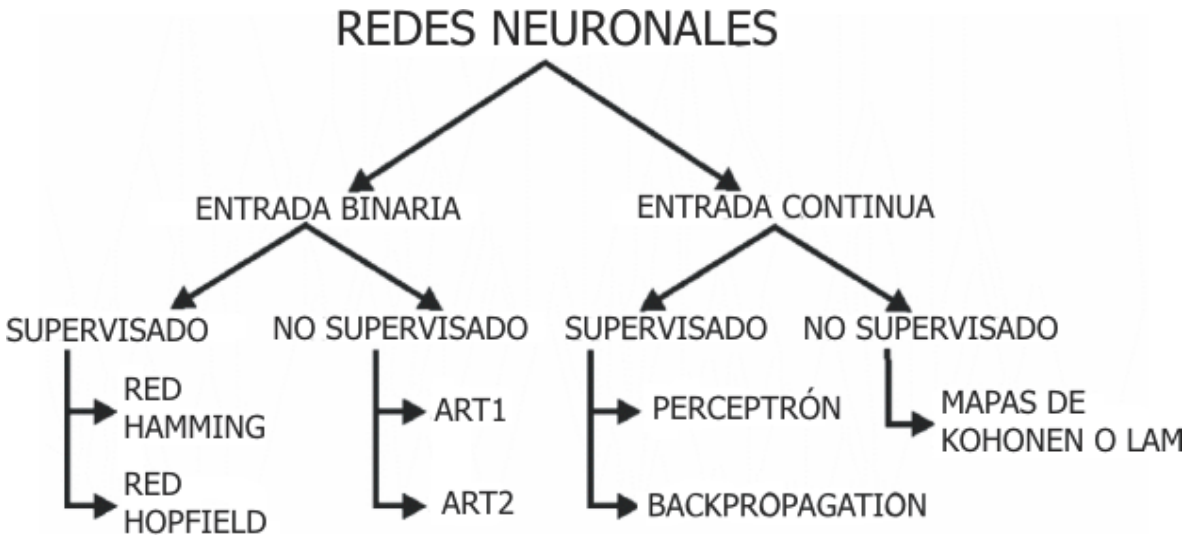
E.4.4 Aprendizaje asociativo. Las redes con aprendizaje no supervisado no requieren influencia externa para ajustar los pesos de las conexiones entre sus neuronas, por ello se dice que estas redes son capaces de autoorganizarce.

Estas redes deben encontrar las características, regularidades, correlaciones o categorías que se puedan establecer entre los datos que se presenten en su entrada, puesto que no hay supervisor que indique a la red la respuesta que debe generar ante una entrada concreta.

En cuanto a algoritmos de aprendizaje no supervisado, en general se consideran dos tipos, que dan lugar a los siguientes aprendizajes:

- **Aprendizaje asociativo**. Pretende medir la familiaridad o extraer características de los datos de entrada.
- **Aprendizaje competitivo**. Se orienta a la cauterización o clasificación de los datos.

En general todas las redes neuronales se pueden clasificar de diversas maneras, según su topología, forma de aprendizaje (supervisado o no supervisado), tipos de funciones de activación. Un resumen de esta clasificación se puede observar en la figura.


REDES NEURONALES
ENTRADA BINARIA
ENTRADA CONTINUA
SUPERVISADO
NO SUPERVISADO
SUPERVISADO
NO SUPERVISADO
RED HAMMING
RED HOPFIELD
ART1
ART2
PERCEPTRÓN
BACKPROPAGATION
MAPAS DE KOHONEN O LAM