UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Facultad 4

Algoritmo para generar posiciones candidatas de enrutadores de una WSAN en entornos interiores

Tesis presentada en opción al título de Máster en Informática Aplicada

Autor: Ing. Angel Alberto Vazquez Sánchez

Tutor: Msc. Ismael Armando Nodarse Mora

La Habana, diciembre de 2014

DEDICATORIA

A mis padres, mi esposa y mi hijo.

AGRADECIMIENTOS

En primer lugar deseo agradecer al MSc. Ismael Armando Nodarse Mora, tutor de esta investigación.

Además, deseo agradecer a los integrantes del grupo de investigación Andrómeda, a mis compañeros de trabajo del departamento y la facultad.

A mis padres y hermanos por todo su apoyo.

A Lisset por darme todo su amor y regalarme el orgullo de ser padre.

A mi otra familia Elvira, Beto y Bético.

Así como a colegas y amigos que no he mencionado, mi más sincero agradecimiento.

DECLARACIÓN JURADA DE AUTORÍA

Declaro por este medio que yo Angel Alberto Vazquez Sánchez, con carné de identidad 84022526763, soy el autor principal del trabajo final de maestría “Algoritmo para generar posiciones candidatas de enrutadores de una WSAN en entornos interiores”, desarrollada como parte de la Maestría en Informática Aplicada y que autorizo a la Universidad de las Ciencias Informáticas a hacer uso de la misma en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Y para que así conste, firmo la presente declaración jurada de autoría en La Habana a los ___ días del mes de diciembre del año 2014.

_____

Ing. Angel Alberto Vazquez Sánchez

RESUMEN

El despliegue de redes inalámbricas de sensores y actuadores (WSANs, por sus siglas en inglés) en entornos interiores representa un reto para los diseñadores; requiere experiencia en el tema y varias iteraciones de ensayo y error para encontrar un diseño optimizado. Como parte del grupo de investigación Andrómeda, se está desarrollando una herramienta que sugiera el diseño de WSANs en entornos interiores. Uno de los principales problemas identificados en el desarrollo de la herramienta fue la generación de las posiciones candidatas de los enrutadores. En el desarrollo de la solución se empleó una red auto-organizada particularmente una modificación del algoritmo Growing Neural Gas. En la validación de la propuesta de solución se aplicaron pruebas t-Student sobre los datos obtenidos del experimento realizado para las métricas producto topográfico, promedio del cuantificador del error y cantidad de componentes conexas.

Palabras claves: redes inalámbricas de sensores y actuadores, posiciones candidatas de enrutadores, red auto-organizada, Growing Neural Gas

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

El empleo de dispositivos electrónicos diseñados para la medición de algún fenómeno físico, es una tecnología que lleva varias décadas en explotación. Una Red Inalámbrica de Sensores (del inglés Wireless Sensor Network, WSN) es una red de nodos físicos, donde cada uno es capaz de medir una magnitud física o química de su entorno y donde los nodos están vinculados a través de un medio inalámbrico [1].

Para el siglo XXI se avizoraba un desarrollo vertiginoso en la utilización de las WSN en varias esferas. En febrero de 2003 el MIT[1] identificó las diez tecnologías emergentes que cambiarán el mundo y en el listado las WSN aparecen en primera posición. Producto a su gran interoperabilidad con otros dispositivos, se definen luego como Redes Inalámbricas de Sensores y Actuadores (del Inglés Wireless Sensor and Actor Networks, WSAN) [2]. En este tipo de red se define un actor o actuador como un dispositivo que convierte una señal eléctrica en una acción física con repercusión en el entorno [3].

Los enrutadores (conocidos también como *relays* o repetidores) son dispositivos que tienen como objetivo propiciar las comunicaciones que ocurren en la red. Utilizado fundamentalmente en WSANs de mediano y gran tamaño, los enrutadores favorecen aspectos como la conectividad, la escalabilidad, el balance de cargas y el tiempo de vida de la red [4].

Otro dispositivo presente en las WSAN es el coordinador, el cual dependiendo de la arquitectura de red, puede funcionar como pasarela *(gateway)* entre la WSAN y una red externa, o como dispositivo central que procesa la información obtenida de los sensores y decide qué acción se debe llevar a cabo por los actuadores [2].

Entre los beneficios del uso de las WSAN se encuentran la velocidad de mejora, el mantenimiento simplificado, costos del ciclo de vida reducidos, transiciones discretas ante las actualizaciones y la flexibilidad [5]. Su instalación puede ser realizada en cualquier lugar y en cualquier momento, permitiendo un ahorro del 20 al 80 por ciento. Además, este tipo de redes están basadas en estándares de la IEEE por lo que son auto-configurables y pueden ser instaladas con el mínimo de experiencia. Los beneficios se extienden adicionalmente en que son altamente confiables, auto-reparables y reducen los costos de mantenimiento [6].

En la actualidad las WSAN se encuentran desplegadas en aplicaciones agrícolas, militares, ambientales, para la salud, la construcción civil, entre muchas otras. Otro uso muy importante es su utilización en los sistemas de gestión de energía en edificios (BEMS, según siglas en inglés)[7]. Factores externos como el ahorro energético, emparejado a las ventajas que aporta esta tecnología a los BEMS actuales, han influido en su gran aceptación en este campo de aplicación.

Existen otros factores que han acelerado la utilización de redes inalámbricas en los BEMS. Por ejemplo, su empleo puede significar una reducción de los costos si se tiene en cuenta el cableado requerido de muchos sistemas de gestión de energía actuales. Además de la reducción de los costos por cableado, la flexibilidad y versatilidad que las caracteriza, representa también una disminución en los costos de despliegue y mantenimiento.

[1] Instituto de Tecnología de Massachusetts (http://web.mit.edu/)

Según [5] y [8] entre los principales problemas en el uso de WSAN en los BEMS se encuentran la cobertura de la red, la interferencia, obstrucciones físicas, el ancho de banda, la eficiencia energética y la inter-funcionalidad entre los componentes del sistema de control.

Otra problemática que enfrentan las redes de sensores en este campo de aplicación, es el despliegue de esta tecnología en entornos interiores. Actualmente, los despliegues de las WSAN dentro de edificaciones, suelen realizarse siguiendo un procedimiento *ad hoc*. Para validar la conexión entre los diferentes nodos de la red, los diseñadores se basan en la estrategia básica de "instalar y probar". Esta técnica puede resultar factible cuando el despliegue es realizado por diseñadores experimentados y en lugares relativamente pequeños. Al aplicarse este tipo procedimiento, surge la siguiente problemática: a medida que aumenta el tamaño de la edificación donde se va a realizar el despliegue, la complejidad a la que se enfrenta el diseñador es cada vez mayor. Por otra parte, cuando se aplica esta técnica en grandes escenarios, se derrocha mucho tiempo y las probabilidades de que se diseñen soluciones deficientes son elevadas, lo cual puede repercutir de forma negativa en la calidad del despliegue [8].

Otra deficiencia que posee la estrategia anterior, es que cuando se realiza un diseño fortuito de la red, su rendimiento puede ser significativamente menor que el obtenido siguiendo una estrategia de diseño mejor fundamentada. Por otra parte, el diseñador debe tener presente varios factores cuando va a realizar un despliegue de una WSAN dentro de un edificio, en los que se incluyen: las características del entorno, la tecnología que será usada, la cobertura de la señal, las zonas específicas que requieren de la medición, la latencia y costos de la red, entre otros. Producto al elevado número de elementos que deben ser considerados, se plantea la necesidad de una herramienta software que le permita al diseñador enfrentarse a la difícil tarea de diseñar una efectiva, robusta y fiable WSAN.

Por lo anteriormente expresado, en [8] se propone el desarrollo de una herramienta software que permita, entre otras funcionalidades, generar propuestas de despliegues de WSAN en el interior de una edificación.

El grupo de investigación Andrómeda, producto de una colaboración entre la Universidad de las Ciencias Informáticas y la Universidad de Málaga, se encuentra desarrollando una herramienta que proponga el diseño de una WSAN basada en el estándar ZigBee[2] [9]. Para ello deberá tener en cuenta las características del entorno y otras restricciones especificadas por el diseñador de la red.

Como características fundamentales de la herramienta se encuentran en primer lugar que está basada en un modelo de despliegue propuesto específicamente para WSAN [10] y en segundo lugar le permite al diseñador especificar la ubicación física de los dispositivos terminales (sensores y actuadores) y del coordinador.

Según [11], para el despliegue de una WSAN se necesita realizar las siguientes fases: captura de requisitos, diseño automático y optimización, despliegue y verificación del correcto funcionamiento de la red. Una herramienta informática estaría enmarcada en las fases de captura de requisitos y de diseño automático y optimización.

[2] Especificación de un conjunto de protocolos de alto nivel de comunicación inalámbrica para su utilización con radiodifusión digital de bajo consumo, basada en el estándar IEEE.

Como entrada, la herramienta recibirá el modelo tridimensional del edificio donde se desea desplegar la red, la ubicación del coordinador (ZigBee Coordinator) y las ubicaciones de los dispositivos terminales (ZigBee End Devices). A partir de estos elementos se generarán las posiciones candidatas que pueden ocupar los enrutadores (ZigBee Routers), de manera que quede garantizada la conectividad entre los dispositivos de la red y un posicionamiento adecuado con relación al entorno. Posteriormente se optimiza el posicionamiento de los enrutadores. Finalmente, el resultado es presentado al diseñador como una propuesta optimizada de la WSAN. (Ver Figura 1)

Figura 1 Funcionamiento de la herramienta.

Una etapa fundamental para el funcionamiento de la herramienta es la generación de posiciones candidatas de los enrutadores, dado que no resulta factible evaluar el rendimiento y comportamiento de un nodo en cada posible posición dentro de un entorno determinado. Como finalidad tiene que garantizar la conectividad entre los dispositivos terminales y el coordinador, ubicando enrutadores en posiciones válidas de acuerdo al modelo del edificio. Las posiciones de los enrutadores deben encontrarse acorde a las posiciones de los dispositivos finales y el coordinador, en otras palabras, las posiciones candidatas en su conjunto deben describir el área delimitada por los dispositivos finales y el coordinador.

Para que una posición candidata sea válida esta debe cumplir las siguientes restricciones:

- Se debe garantizar la conectividad entre los dispositivos terminales y el coordinador;
- El conjunto de posiciones candidatas generadas preserva la topología descrita por los dispositivos finales y el coordinador.

El cumplimiento de estas restricciones para cada posición candidata constituye un proceso complejo y engorroso si se toma en cuenta que, se debe distribuir equitativamente las posiciones candidatas a través de ambientes complejos. Por tanto, no es factible que el diseñador especifique cada posible ubicación de los enrutadores, siendo entonces necesario que este proceso sea llevado a cabo de forma automática.

En este contexto se plantea como **problema de investigación:**

¿Cómo generar posiciones candidatas válidas de los enrutadores en el diseño de una WSAN para entornos interiores?

Se presenta como **objeto de estudio** el proceso de diseño de WSANs en entornos interiores y como **campo de acción** la generación de posiciones candidatas de enrutadores de una WSAN en entornos interiores.

El **objetivo general** de esta investigación es desarrollar un algoritmo para generar posiciones candidatas válidas de enrutadores de una WSAN en entornos interiores.

El objetivo general se articula en los siguientes **objetivos específicos**:

1. Establecer los fundamentos teóricos y metodológicos relacionados con el diseño de WSANs y la generación de posiciones candidatas de enrutadores.
2. Desarrollar un algoritmo para la generación de posiciones candidatas de enrutadores de una WSAN en entornos interiores.
3. Validar la propuesta desarrollada.

La **investigación** es de tipo **descriptiva**. Por lo que se plantea la siguiente **hipótesis**:

Si se desarrolla un algoritmo para la generación de posiciones candidatas de enrutadores de una WSAN en entornos interiores entonces se garantiza la conectividad entre los dispositivos terminales y el coordinador; y se preserva la topología descrita por los dispositivos finales y el coordinador.

Operacionalización de las variables dependientes e independientes

Variable independiente: algoritmo para la generación de posiciones candidatas de enrutadores de una WSAN en entornos interiores.

Variables dependientes:

- Conectividad entre los dispositivos terminales y el coordinador.
- Preservación de la topología descrita por los dispositivos finales y el coordinador.

Tabla 1 Operacionalización de las variables.

Variable independiente	**Dimensión**	**Indicadores**	**Unidades de medida**
Algoritmo para la generación de posiciones candidatas de enrutadores de una WSAN en entornos interiores	Eficacia	Capacidad de lograr un efecto deseado, esperado o anhelado	Alta, Media, Baja
Variable dependiente	**Dimensión**	**Indicadores**	**Unidades de medida**
Conectividad entre los dispositivos terminales y el coordinador	Conectividad	Cantidad de componentes conexas	Valor Numérico Discreto
Preservación de la topología descrita por los dispositivos finales y el coordinador	Preservación de la topología	Producto topográfico[3]	Valor Numérico Continuo
		Promedio del Cuantificador de Error	Valor Numérico Continuo

Métodos de investigación:

Analítico-Sintético: Permitió descomponer el problema de investigación en varios elementos; de esta forma fue más sencillo profundizar en el estudio de cada elemento, para que luego fueran sintetizados en la solución de la propuesta.

Histórico-Lógico: El empleo de este método permitió llevar a cabo el estudio crítico del proceso de diseño de una WSAN, además permitió constatar teóricamente cómo han evolucionado los algoritmos de generación de posiciones candidatas.

Inductivo-deductivo: Se empleó para identificar la técnica con que se desarrollará el algoritmo.

Experimento: Será utilizado para la verificación de la hipótesis.

Medición: A partir de las mediciones se pudo realizar evaluaciones al algoritmo propuesto.

Aporte y Novedad de la investigación:

A partir de la presente investigación **se contará** con un algoritmo que podrá ser empleado por los sistemas de diseño y simulación de WSAN. Sus características permitirán que sea usado en entornos interiores.

[3] Es una medida de la preservación de las relaciones de vecindad entre diferentes espacios.

El resultado de esta investigación puede ser extrapolado fácilmente al posicionamiento de enrutadores en redes inalámbricas de sensores (WSN, por sus siglas en inglés).

Lo **novedoso** de la investigación está reflejado en el empleo de la información referente a la estructura de entornos interiores para la generación automática de posiciones candidatas de los enrutadores de una WSAN. Las posiciones candidatas generadas constituirán un espacio de búsqueda reducido para ser utilizado por cualquier algoritmo de diseño de una WSAN en entornos interiores.

En ese sentido, la solución pretende apoyar la validación de una novedosa estrategia de despliegue propuesta específicamente para WSANs [10] que pudiera ser utilizada con éxito en la práctica y/o servir como banco de pruebas para próximas investigaciones.

Otro aspecto de novedad es el empleo de un algoritmo de aprendizaje de redes neuronales auto-organizadas para la generación de posiciones candidatas de enrutadores de una WSAN en entornos interiores.

Organización de las tesis:

El documento se encuentra organizado en tres capítulos de la forma siguiente:

Capítulo 1 Preliminares. En este capítulo se expresan los elementos fundamentales relacionados el diseño de WSAN en entornos interiores, así como de los algoritmos usados en la generación automática de posiciones candidatas. Se realiza un análisis de los principales trabajos realizados en la generación de posiciones candidatas de enrutadores en entornos interiores.

Capítulo 2 Propuesta de solución. En este capítulo se expresan los elementos fundamentales a tener en cuenta para la generación de posiciones candidatas de enrutadores de una WSAN en entornos interiores. Se desarrolla un algoritmo para generar las posiciones candidatas de enrutadores en entornos interiores.

Capítulo 3 Validación de la propuesta. En este capítulo se realiza la evaluación de la calidad del algoritmo desarrollado. Además se realiza un análisis del costo computacional del algoritmo desarrollado.

Finalmente se presentan las Conclusiones, Recomendaciones, el Glosario de Términos y las Referencias Bibliográficas.

CAPÍTULO 1 PRELIMINARES

En este capítulo se brinda una panorámica sobre el proceso de diseño de una WSAN en entornos interiores. Además se enfoca en la generación automática de posiciones candidatas de dispositivos inalámbricos de una WSAN en entornos interiores; realizando un estudio de los principales algoritmos usados en dicha actividad. Se exponen elementos a tener en cuenta a la hora de generar las posiciones candidatas de los dispositivos inalámbricos de una WSAN en entornos interiores. También se realiza un estudio sobre los principales algoritmos empleados en la construcción de redes auto-organizadas.

1.1 Fundamentos sobre WSANs

Una Red Inalámbrica de Sensores (WSN) en su forma más simple puede ser definida como una red de dispositivos denotados como *nodos* que pueden monitorear el ambiente y comunicar la información obtenida del campo de monitoreo a través de enlaces inalámbricos; los datos son reenviados, posiblemente a través de múltiples saltos en repetidores *(router)*, a un *sumidero* (*sink*), en algunas ocasiones denotados como *controlador* o *monitor,* que puede ser usado localmente o conectado a otras redes a través de una pasarela (*gateway*) [12]. (Ver Figura 2)

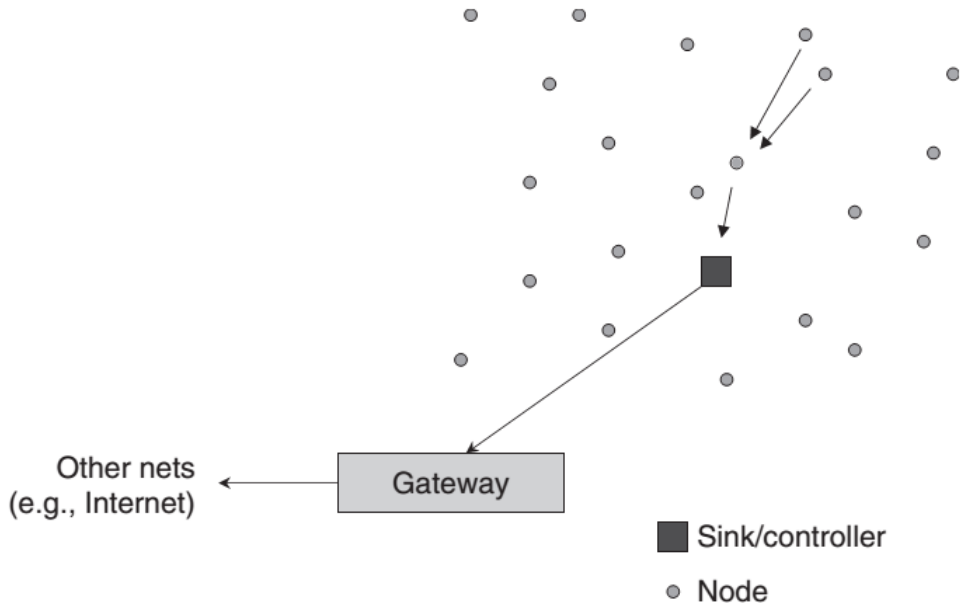

Figura 2 Ejemplo de WSN. (Fuente:[12])

Cuando se incluye la presencia de *actuadores* o sea dispositivos que son capaces de manipular el ambiente, entonces se debe hablar en términos de Red Inalámbrica de Sensores y Actuadores (WSAN, por sus siglas en inglés). Los elementos básicos de una WSAN son los *nodos* (tanto sensores, enrutadores como actuadores), los *sumideros*, y las *pasarelas* [12]. (Ver Figura 3)

De forma general podemos describir a una WSAN como una red de nodos que miden el ambiente cooperativamente y puede controlarlo, habilitando la interacción entre las personas o computadoras con el ambiente circundante.

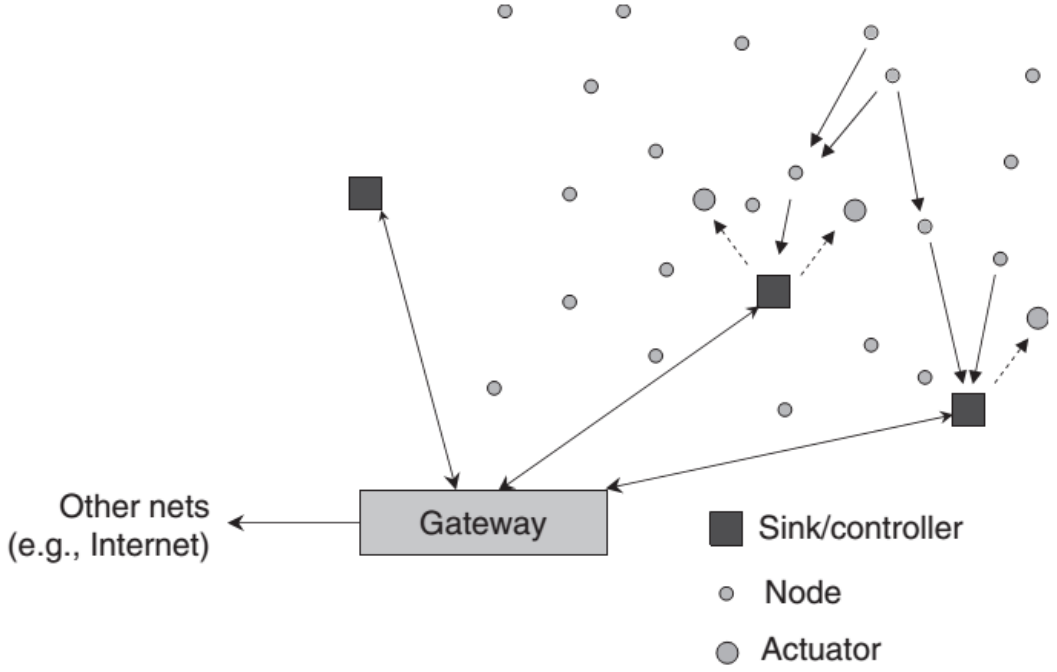

Figura 3 Ejemplo de una WSAN. (Fuente:[12])

Los nodos sensores son de bajo costo, dispositivos de bajo poder con capacidades limitadas de monitoreo, computación y comunicación inalámbrica. Los actuadores son nodos ricos en recursos equipados con mejores capacidades de procesamiento, mayor poder de transmisión y mayor tiempo de vida de la batería.

Las WSANs tienen las siguientes características únicas [2]:

- *Requerimiento en tiempo real*: dependiendo del tipo de aplicación puede existir la necesidad de responder a la entrada de un sensor. Por ejemplo, en una aplicación contra incendio, la acción debe ser inicializada en el área de evento lo más pronto posible. Además, el dato coleccionado y distribuido por el sensor debe permanecer válido en el momento de actuar.
- *Coordinación*: A diferencia de las WSN donde la entidad central (*sink*) desarrolla la función de coleccionar los datos y coordinarlos, en las WSAN un nuevo fenómeno de red llamado coordinación sensor-actuador y actuador-actuador puede ocurrir.
- *Despliegue de sensores y actuadores*: Según el caso, la red puede necesitar cientos de sensores y pocos actuadores en los puntos del entorno que lo requieran, o cientos de actuadores y unos pocos sensores.

Los enrutadores tienen la misma función en estos tipos de redes; no importa qué tipos de dispositivos se estén comunicando; su objetivo es garantizar la conectividad, mejorar la escalabilidad, el balance de carga y el tiempo de vida [13].

Desde el punto de vista de su arquitectura, en [14] se plantea que las WSANs se dividen en dos grupos: las de arquitectura semiautomática y las de arquitectura automática. Cuando los sensores de una WSAN de arquitectura semiautomática detectan un fenómeno estos transmiten su lectura hacia el coordinador; este la procesa y se encarga de comunicar a los actuadores la acción que deben ejecutar. Por otro lado, las de arquitectura automática transmiten su lectura a los actuadores; estos procesan la información y ejecutan la acción apropiada. (Ver Figura 4)

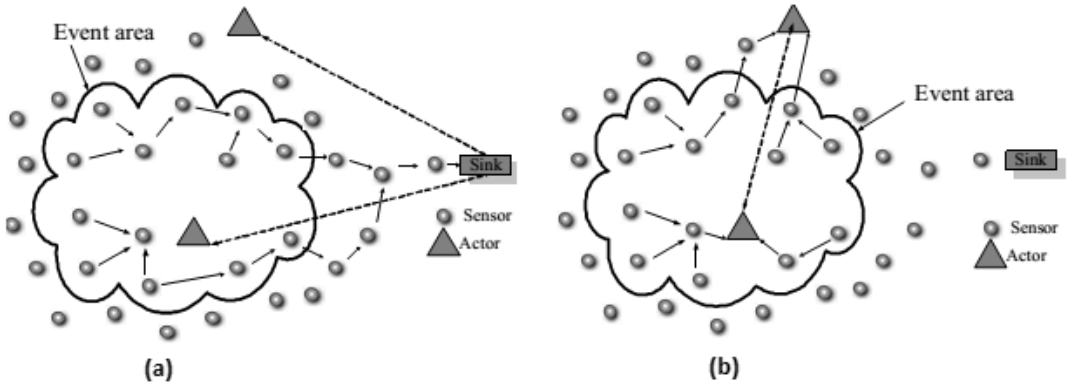

Figura 4 Arquitecturas de las WSANs: (a) semiautomática (b) automática. (Fuente:[14])

Las WSANs de arquitecturas semiautomática requieren algoritmos de comunicación y coordinación simples y su arquitectura es similar a las WSNs. Por otro lado, las de arquitectura automática presentan tasas de transferencias más bajas, tienen un menor consumo de energía y un tiempo de vida más largo [2]. Tomando en cuenta la naturaleza del sistema en el que se pretende desplegar la WSAN y las características antes mencionadas los diseñadores deben elegir entre una u otra arquitectura para el diseño de la red.

1.2 Diseño de WSAN en entornos interiores

Según [11] para lograr emplear una WSN para soportar un sistema de monitoreo de energía de un edificio es necesario un proceso de cuatro fases. Dichas fases son:

- Captura de requisitos.
- Diseño automático y Optimización.
- Despliegue.
- Verificación.

Estableciendo que la herramienta de diseño propuesta en dicho artículo solamente se enmarcaría en las dos primeras fases. Durante la primera fase de Captura de requisitos, se plantea el uso del modelo de datos IFC[4] ya que constituye una forma estándar para compartir e intercambiar información acerca de los datos de los edificios; pero dicho modelo es reducido hasta el punto que solamente se toma en cuenta para la geometría de las plantas las paredes, las puertas y las ventanas. Durante esta etapa también se cuenta con una biblioteca de componentes (los diferentes tipos de dispositivos inalámbricos) con las características de cada uno, dichas características constituyen restricciones adicionales durante el proceso de diseño de la WSN.

Además se expresa que la segunda fase se realizará en dos pasos, a través de algoritmos de pre-procesamiento y de optimización de las posiciones candidatas. En el primero se realiza una reducción del espacio de búsqueda al generar una serie de posiciones candidatas. El segundo algoritmo se encarga de encontrar un subconjunto de los dispositivos generados de forma tal que permita soportar una red con las características requeridas.

[4] Siglas del inglés Industry Foundation Classes (IFC), modelo de datos que describe edificios y datos de la industria de la construcción.

En [15] se propone el desarrollo de una herramienta de diseño de una WSN que use como protocolo ZigBee, la misma consta de tres elementos fundamentales:

1. *Despliegue de la red*. Basado en la configuración de hardware de la antena y la descripción del área de despliegue este componente exporta la representación del espacio tridimensional del entorno interior, la localización de los nodos sensores desplegados, medidas de los patrones de radiación de antena, y modelos de interferencia y perdida de la señal.
2. *Generación de la topología*. Basado en las posiciones de los nodos sensores desplegados y la información de la pérdida de señal de los componentes desplegados así como las configuraciones de carga de tráfico y parámetros de ZigBee, se genera un árbol topológico de agrupaciones de ZigBee que satisface todas las restricciones.
3. *Funciones de evaluación de rendimiento*. A través de un simulador de la red y basado en las salidas de los componentes anteriores esta funcionalidad evalúa el rendimiento de la red, incluyendo la demora punto a punto y el promedio de perdida de paquetes.

Por otra parte en [16] se presenta una herramienta para el modelado, diseño y evaluación con el objetivo de asistir a los diseñadores cuando llevan a cabo la tarea de diseñar una WLAN[5]. En dicho artículo se plantea que de forma general las herramientas que llevan estas tareas realizan dos pasos de diseño fundamentales. El primer paso requiere la definición de los requisitos específicos del lugar, cada herramienta requiere diferentes niveles de detalles desde una descripción general del entorno hasta las definiciones de muros mapeados en base de datos de propiedades de materiales. El segundo paso involucra el diseño de la WLAN usando modelos de propagación.

Para el diseño de una red inalámbrica de infraestructura para edificios inteligentes se propone en [17] un proceso que consiste en dos fases. La primera consiste en la captura de los requerimientos del sistema, recibiendo como entrada un modelo IFC y permitiendo al diseñador elegir las zonas de demanda. La segunda fase se encarga del pre-procesamiento y la optimización, el primer paso se encarga de la generación de posiciones candidatas de los nodos sensores y los sumideros; ya en el segundo paso se optimiza el conjunto de posiciones candidatas de modo que se optimice algún parámetro como puede ser por ejemplo la confiabilidad de la comunicación.

En [18] se hace referencia a una metodología de diseño para WSN la cual consiste en las siguientes etapas: requerimientos del sistema, análisis, diseño de la solución, y desarrollo de la arquitectura de software, implementación y prueba. Hay que mencionar que dicha metodología no propone el uso de alguna herramienta que automatice el proceso.

En [8] y [11] se plantea para el proceso de despliegue de una WSAN un proceso de cuatro fases: Captura de Requisitos, el Diseño Automático y Optimización, el Despliegue y por último la Verificación del correcto funcionamiento de la red.

En todos los procesos estudiados se tiene a la captura de requisitos y al diseño automático y optimización, como los pasos fundamentales a ser automatizados mediante una herramienta informática. A consideración del autor el proceso

[5] WLAN (*Wireless Local Area Network*), se toma en cuenta debido a la similitud entre este tipo de red inalámbrica y las WSAN.

propuesto por [8] y [11] es el que más se ajusta al problema planteado. Específicamente en la fase de diseño y optimización se establecen los siguientes pasos necesarios para completarla (según [11]):

- Importar la descripción del entorno.
- Generación de la red candidata.
- Módulo de optimización.

Según [8] en la fase de diseño y optimización se establecen los siguientes pasos:

- Selección del modelo de propagación de la señal de RF[6].
- Generación de posiciones candidatas.
- Generación del diseño de la red (mediante la optimización de las posiciones candidatas).

En ambos enfoques un aspecto fundamental es la generación de las posiciones candidatas de los enrutadores. Luego se pasará a analizar cómo se ha sido abordado este proceso por la comunidad científica, siendo este el objetivo del siguiente epígrafe.

1.3 Generación automática de posiciones candidatas de enrutadores de WSANs en entornos interiores

Con el fin de reducir el espacio de búsqueda para posteriores algoritmos de optimización es necesario realizar la generación automática de posiciones candidatas de los enrutadores de una WSAN en entornos interiores. Los enfoques son escasos y generalmente se hace únicamente mención del algoritmo utilizado con dicho fin, por lo que el acceso al código o seudocódigo de los algoritmos empleados es prácticamente nulo.

El enfoque más básico de todos es el de limitar el espacio de búsqueda representando el área como una cuadrícula y considerando como posiciones candidatas solamente los puntos de intersección [4]. Con este enfoque no se toman en cuenta las restricciones del diseñador, la cantidad de posiciones que pueden ser generadas puede ser mayor que las generadas siguiendo otro tipo de proceso, dependiendo de la distancia que se dé entre los diferentes puntos de la cuadrícula se incurre en el riesgo de obtener una cuadrícula dónde los nodos no estén conectados o exista redundancia y/o solapamiento entre las funciones que deben realizar los diferentes nodos. Por otro lado, la topología que se obtiene de aplicar este procedimiento es fija, no necesariamente representando el entorno de monitoreo.

La generación aleatoria de las posiciones candidatas constituye otra vía para lograr el objetivo planteado. En [19] se plantea que esta técnica es empleada no solamente para generar posiciones candidatas, sino para generar toda la WSAN. Esta es generada por la colocación de un grupo de sensores en un patrón de área específica, tales como rectángulos o círculos. La posición de los N nodos es aleatoriamente determinada (por ejemplo, mediante de la selección aleatoria de dos o tres coordenadas) y son independientes unos de otros.

[6] Radio Frecuencia, se aplica a la porción menos energética del espectro electromagnético, situada entre unos 3 kHz y unos 300 GHz.

A la perspectiva anterior, en [4] se le llama *distribución de forma aleatoria* de los sensores; en dicho artículo se referencia a [20] donde se plantea que siguiendo este acercamiento existen tres enfoques o patrones principales: *difusión simple* (distribución normal en dos dimensiones), *uniforme* y *R-aleatorio* donde los nodos son uniformemente diseminados respecto a la dirección radial o angular desde la estación base. Siguiendo este modelo en [21] se considera una arquitectura de red en dos capas donde los sensores son agrupados alrededor de repetidores que se comunican directamente con la estación base; en dicho artículo se muestra que esta distribución no extiende el tiempo de vida de la red, debido a que los nodos repetidores consumen su energía a diferentes ritmos dependiendo de su proximidad a la estación base. Estos enfoques han sido utilizados principalmente para el despliegue de sensores en entornos exteriores, saliendo del objetivo de esta investigación. De todas maneras siguiendo este enfoque no se asegura el cubrimiento del entorno de monitoreo dada la aleatoriedad de la selección.

En [22] se plantea el uso de un algoritmo genético para resolver el problema del posicionamiento de los sensores, donde cada uno de los cromosomas de la población se corresponde con un posicionamiento candidato de los dispositivos inalámbricos de la WSAN. En este enfoque es modelado el posicionamiento de los sensores como un problema de optimización combinatorio. Otro artículo donde se emplea una técnica de este tipo para el problema del posicionamiento de los sensores en la WSAN es [23], en este caso es aplicado en el contexto de un sistema en un entorno exterior. Al igual que el enfoque anterior este ha sido usado principalmente en entornos exteriores, además tiene como otra característica tiene que este enfoque busca obtener las posiciones candidatas pero luego de un proceso de optimización donde las mismas ya están en su posición óptima por lo que no se ajusta al objetivo de esta investigación.

Finalmente el acercamiento más empleado ([11, 16, 17]) es el de usar una red auto-organizada. De esta manera una topología de red es generada incrementalmente, ofreciendo como ventaja que se va distribuyendo de manera uniforme las posiciones candidatas a través de ambientes complejos, mientras se obtienen posiciones candidatas deseables, por ejemplo en los muros, etc. Mediante estas técnicas se obtiene una malla de posiciones candidatas que contiene un número finito de posiciones candidatas, las cuales reducen el espacio de búsqueda y aumentan la velocidad de los algoritmos de optimización a emplear en etapas posteriores.

En la Tabla 2 se detalla la relación de artículos donde se hace mención de los diferentes enfoques en la generación de posiciones candidatas.

Tabla 2 Resumen de artículos sobre la generación de posiciones candidatas.

	Nombre artículo	**Año**	**Estrategia empleada**
[4]	Strategies and techniques for node placement in wireless sensor networks: A survey	2008	Cuadrícula Generación aleatoria Algoritmo Genético Red auto-organizada
[19]	Applications, Models, Problems, and Solution Strategies	2010	Generación aleatoria
[20]	Performance study of node placement in sensor networks	2004	Generación aleatoria
[13]	Relay node deployment strategies in heterogeneous wireless sensor networks: multiple-hop communication case	2005	Generación aleatoria

[22]	Sensor placement in sensor and actuator networks	2010	Algoritmo genético
[23]	Multi-objective genetic algorithm for the automated planning of a wireless sensor network to monitor a critical facility	2004	Algoritmo genético
[11]	A wireless sensor network design tool to support building energy management	2009	Redes auto-organizadas
[16]	A wireless local area network modeling tool for scalable indoor access point placement optimization	2010	Redes auto-organizadas
[17]	Design of underlying network infrastructure of smart buildings	2008	Redes auto-organizadas
[8]	Utilización de WSANs en Sistemas de Control de Edificios	2010	Redes auto-organizadas

En la revisión bibliográfica no se encontraron estrategias que difieran de las expresadas en la tabla anterior en el período posterior a los años señalados.

Analizando las características mencionadas para cada uno de los enfoques planteados, se determina que el método que mejor se adapta al problema es el de las redes auto-organizadas. Entre las características deseables de este enfoque se encuentra su capacidad de crear una red que se adapta a ambientes complejos como pueden ser los entornos interiores. Otro elemento de importancia es que los elementos generados describen la topología del área de monitoreo de donde surgen las señales que recibe el algoritmo.

En la siguiente sección se estudiará en mayor profundidad las redes auto-organizadas.

1.4 Redes auto-organizadas

Las redes auto-organizadas emplean su capacidad de aprendizaje adaptativo para auto-organizar la información que reciben durante el aprendizaje y/o la operación. La auto-organización consiste en la modificación de la red neuronal completa para llevar a cabo un objetivo específico. Cuando las redes neuronales se usan para reconocer ciertas clases de patrones, estas auto-organizan la información usada [24].

En [25] se plantea que la auto-organización provoca la *generalización*: facultad de las redes neuronales de responder apropiadamente cuando se les presentan datos o situaciones a las que no habían sido expuestas anteriormente. El sistema puede generalizar la entrada para obtener una respuesta. Esta característica es muy importante cuando se tiene que solucionar problemas en los cuales la información de entrada no es muy clara; además permite que el sistema dé una solución, incluso cuando la información de entrada está especificada de forma poco clara o incompleta.

Una propiedad muy importante es la planteada por [26] donde se explica que las redes formadas mediante auto-organización son capaces de describir relaciones topológicas entre las señales de entrada; de modo que las relaciones de semejanza más importantes entre las señales de entrada son convertidas en relaciones espaciales entre las neuronas.

Existe gran variedad de modelos neuronales auto-organizativos. La mayor parte posee una topología y funcionamiento básico semejante. En [26] se describe en

qué consiste el comportamiento común de cada uno de los modelos neuronales auto-organizados.

Una red neuronal auto-organizada está formada por un conjunto A de n neuronas:

$$A = \{c_1, c_2, \dots, c_n\} \quad (1)$$

Donde cada una de las neuronas tiene asociado un vector de referencia w_c perteneciente al espacio de las señales de entradas V.

$$w_c \in V \quad (2)$$

que indica la zona del espacio de entrada a la cual esta neurona es más receptiva.

Las neuronas están conectadas entre sí mediante conexiones de vecindad que son simétricas y permiten establecer una relación topológica entre las diferentes neuronas pertenecientes a la red neuronal.

$$C \subset A \times A \quad (3)$$

$$(i, j) \in C \Leftrightarrow (j, i) \in C$$

De modo que una neurona c posee un conjunto de vecinos topológicos N_c:

$$N_c = \{i \in A / (i, c) \in C\} \quad (4)$$

El aprendizaje se realiza a partir de un conjunto de señales de entrada n-dimensionales que son generadas siguiendo una función de densidad de probabilidad.

$$p(\xi), \xi \in V \quad (5)$$

Para cada señal de entrada ξ, mediante un proceso competitivo entre las n neuronas, se obtiene la neurona ganadora $s(\xi)$ definida como la neurona que posee vector de referencia más cercano a ξ

$$s(\xi) = \arg min_{c \in A} \ \|\xi - w_c\| \quad (6)$$

donde $\|\cdot\|$ representa, normalmente, la distancia de Euclides.

Posteriormente, se produce un proceso de adaptación de los vectores de referencia de todas o una parte de las neuronas de la red (en dependencia de la relación de vecindad) con el objetivo de aproximar sus vectores de referencia a la señal de entrada siguiendo la Ley de Hebb [24].

$$\Delta w_c = \alpha \cdot (w_c - \xi) \quad (7)$$

donde α pondera el paso de adaptación.

Una vez finalizado el proceso auto-organizativo, se obtiene un mapeado del espacio de las señales de entrada V en la red neuronal A:

$$\phi_W : V \to A, \xi \in V \to \phi_W(\xi) \in A \quad (8)$$

Donde $\phi_W(\xi)$ se obtiene a partir de la siguiente condición.

$$\left\|w_{\phi_w(\xi)} - \xi\right\| = min_{c \in A}\|\xi - w_c\| \quad (9)$$

Los modelos auto-organizativos se pueden diferenciar en dos grandes grupos desde el punto de vista estructural: aquellos que tienen dimensiones fijas y topología de la red preestablecida; y los que varían sus dimensiones durante el aprendizaje [27]. Entre el primer grupo se encuentran los mapas auto-organizados de Kohonen y las Growing Cell Structures. En el segundo grupo se encuentran el Neural Gas y el Growing Neural Gas.

1.4.1 Mapas auto-organizados de Kohonen

Las Redes de Kohonen (también llamados Mapas Auto-Organizados de Kohonen o KSOM por sus siglas en inglés) son sistemas de clasificación no supervisados que permiten encontrar individuos de una población que comparten características comunes[28]. Dicha característica los hace ideales para explorar espacios vectoriales en los que se desconoce la estructura de clasificación de los vectores.

Esta red es compuesta por dos capas de neuronas, una capa lineal y otra de forma matricial (ver Figura 5), la primera capa (entrada) tiene como número de neuronas el número de dimensiones que tenga el espacio de entrada, y la segunda capa, la cual se ajusta después del entrenamiento a los patrones de entrada, tiene la cantidad de neuronas que se desee, de manera, que la adaptación sea total o por lo menos abarque la mayor parte del espacio de entrada (cantidad de patrones de entrenamiento)[29]. Este tipo de red posee una estructura de tipo *feed-forward* y el proceso de entrenamiento de este tipo de red es competitivo.

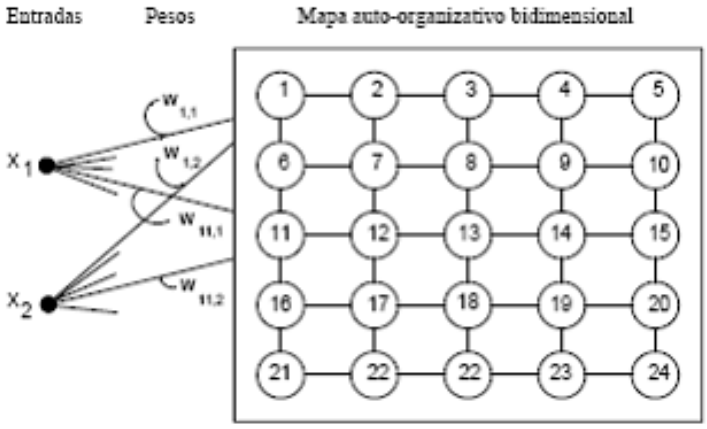

Figura 5 Mapa auto-organizado de Kohonen. (Fuente: [29])

Según [29] este tipo de redes son comúnmente utilizadas para la reducción de dimensiones, reconocimiento de texto, reconocimiento de voz, cuantificación vectorial, etc.

En [30] se plantea que este tipo de red tiene como limitaciones que:

- Tienen un tamaño fijo y la topología debe ser predeterminada.
- Algunas celdas están localizadas donde la probabilidad de densidad $p(x)$ del vector de entrada x es cero.
- Algunas celdas que son vecinos directos topológicamente tienen posiciones más distantes en $p(x)$.

Estas limitaciones resultan en un pobre rendimiento para el agrupamiento de patrones cuando se usa KSOM para algunas aplicaciones. Para superar estas limitaciones B. Fritzke en [31] propone Growing Cell Structures y Growing Neural Gas.

1.4.2 Growing Cell Structures (GCS)

Este modelo posee una estructura flexible, un número variable de neuronas y una topología k-dimensional donde k es escogido arbitrariamente y a priori. Las neuronas están conectadas entre sí, formando hipertetraedros. El caso más simple y usual es en el que la red es bidimensional y las neuronas están unidas formando triángulos [29]. En la Figura 6 se muestra cómo es el proceso de acomodación e inserción de neuronas mediante el proceso de entrenamiento, este proceso es no supervisado.

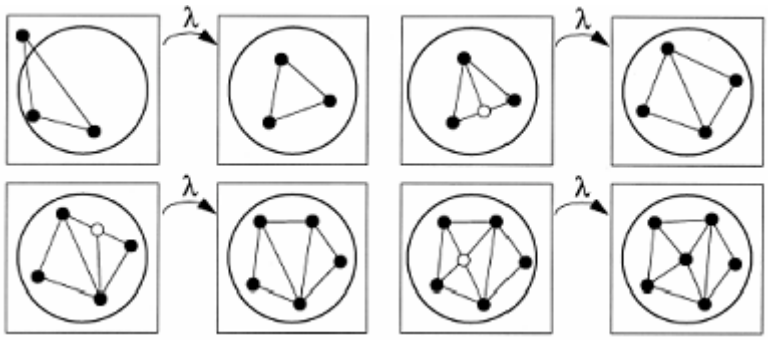

Figura 6 Growing Cell Structure. (Fuente:[29])

El GCS comienza con un triángulo de tres células en posiciones aleatorias en $p(x)$, las células son ubicadas sobre áreas de probabilidad de densidad distinta de cero. La inserción de nuevas células debe mantener la estructura de conectividad triangular. La estructura resultante es entonces redistribuida a través de un proceso adaptativo, que es análogo a KSOM [30].

Como bien se plantea en [31] la principal ventaja respecto a KSOM es la habilidad de la red de automáticamente encontrar una estructura y tamaño de red adecuado. Esto se logra a través de un proceso controlado de crecimiento que también incluye eliminación ocasional de unidades.

Las redes GCS y los KSOM permiten la proyección en subespacios de muestras discretas no lineales de una dimensionalidad escogida a priori. Dependiendo de las relaciones inherentes entre la dimensionalidad de los datos y la dimensionalidad del espacio buscado, puede provocar que alguna información de la disposición topológica de los datos de entrada se pierda en el proceso. Se puede decir que, generalmente, no existe un mapeo reversible de una dimensionalidad alta a una baja. La búsqueda de estructuras que permitiesen estos mapeos reversibles, permitió alcanzar un nuevo objetivo dentro del aprendizaje no supervisado, el aprendizaje topológico [32].

Dada una distribución de datos de alta dimensionalidad $P(\xi)$ se tiene que encontrar una estructura topológica que refleje lo más fielmente posible la distribución topológica de los datos. Una de las formas más elegantes de construir este tipo de estructuras es mediante el *aprendizaje competitivo de la regla de Hebb* (CHL, por sus siglas en inglés) [32]. Este aprendizaje requiere uso de un método de cuantización vectorial. Para estos fines se propuso en [33] el método de Neural Gas y basándose en este Fritzke presentó el método Gas Neuronal Creciente en [34].

1.4.3 Gases Neuronales

En [26] se expresa que bajo este término se engloban diferentes modelos auto-organizativos que: realizan su proceso de adaptación según una función de energía (espacio de los vectores de entrada), a diferencia de los mapas auto-organizativos de Kohonen, en el que viene determinado por la estructura de la red, y convergen rápidamente a errores de cuantización pequeños, menores que los obtenidos con los modelos anteriores.

Estos pretenden evitar las restricciones de los modelos anteriores que requieren conocimiento a priori sobre la dimensión topológica del espacio de los vectores de entrada. Es por eso que no preestablecen ninguna topología de red, sino que es durante el aprendizaje cuando las neuronas se conectan o desconectan reajustando su red de interconexión.

1.4.3.1 Neural Gas (NG)

Este modelo posee las características mencionadas anteriormente. No establece a priori ninguna topología de la red, conecta las dos neuronas más cercanas para cada uno de los patrones escogidos en el aprendizaje. Posee la desventaja de tener que predeterminar el tamaño de la red, es decir, el número de neuronas que posee [26]. En la Figura 7 se muestra cómo evoluciona la red, donde el gas neuronal se adapta o toma la forma de una estructura en tres dimensiones, después de su entrenamiento.

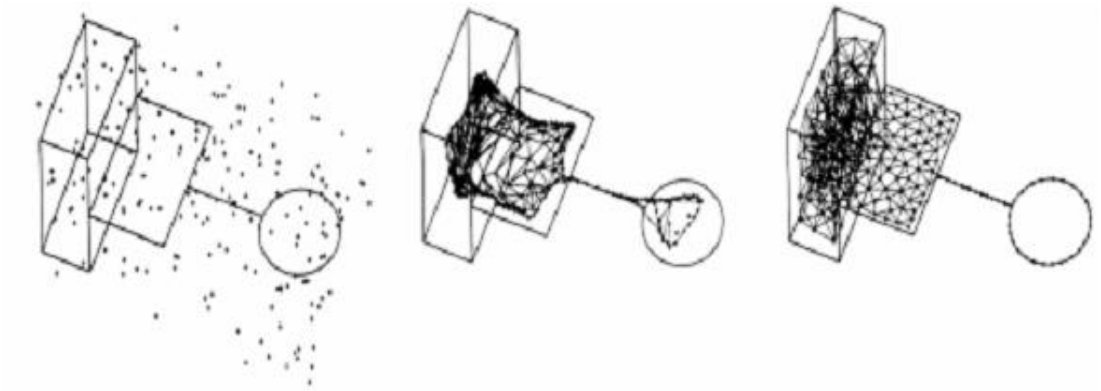

Figura 7 Adaptación de un gas neuronal a una forma de tres dimensiones. (Fuente:[26])

La ordenación de las neuronas, según la distancia al patrón de entrada, y la posterior modificación de sus vectores de referencia, produce la expansión de las mismas dentro del espacio de entrada. Posteriormente, mediante la inserción y eliminación de aristas se establece una triangulación entre los diversos elementos de procesamiento.

Este tipo de red auto-organizada ha sido empleada en la predicción de series temporales, control de robots, clasificación de escritura, etc. [35, 36]

1.4.3.2 Growing Neural Gas (GNG)

En algunos casos poca o ninguna información está disponible sobre la distribución o el tamaño del conjunto del tamaño de entrada, en estos casos es difícil determinar a priori el número de nodos a usar, tal es el caso de los Mapas auto-organizados de Kohonen, el algoritmo Neural Gas y también en el agrupamiento clásico K-means [37].

Debido a esto como ya se había planteado, en [34] Fritzke propone el método Gas Neuronal Creciente (GNG de las siglas en inglés). Para ello se vale de las bondades del Neural Gas y de Growing Cell Structures. En dicho método no se establece topología de unión entre las neuronas y tampoco el número de neuronas de la red, sino que a partir de dos neuronas, el gas se va reproduciendo y ubicando, generando y uniendo neuronas hasta que toma la forma exacta del espacio de entrada [29]. En la Figura 8 se muestra el proceso de adaptación a la misma figura de 3 dimensiones.

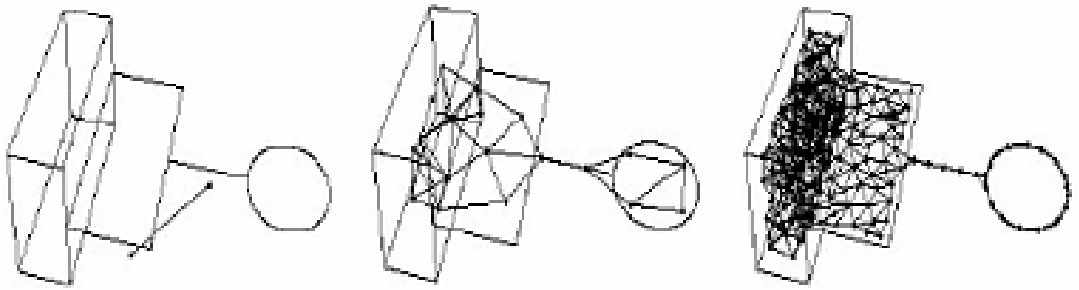

Figura 8 Adaptación de un Gas Neuronal Creciente a una forma de tres dimensiones. (Fuente:[29])

Este modelo ha sido generalmente utilizado en la robótica, identificación de gestos y reconstrucción de superficies [38-40].

1.4.4 Comparación entre los diferentes algoritmos

Para la evaluación de los algoritmos se decide basarse en dos criterios fundamentales: en primer lugar la técnica que mejor preserve la topología, y en segundo lugar el coste computacional.

El primer criterio está estrechamente relacionado con los conceptos de región de Voronoi y de triangulación de Delaunay. La región de Voronoi de una neurona está formada por todos los puntos del espacio de entrada para el que ésta resulta la neurona ganadora. Por tanto, como resultado final del aprendizaje competitivo se obtiene un grafo (red neuronal), cuyos vértices son las neuronas de la red y cuyas aristas son las conexiones entre las mismas, que representa la triangulación de Delaunay del espacio de entrada correspondiente a los vectores de referencia de las neuronas de la red [26].

La triangulación de Delaunay preserva la topología del espacio de entrada, en [41] Martinetz introduce una terminología que restringe esta cualidad. Se dice que el mapeado ϕ_w de V en A preserva la vecindad si vectores de características similares, vectores cercanos en el espacio de entrada V, son mapeados en neuronas cercanas de A.

De igual manera, se señala que el mapeado inverso

$$\phi_w{}^{-1}: A \rightarrow V, c \in A \rightarrow w_c \in V \tag{10}$$

preserva la vecindad si neuronas cercanas en A tienen asociados vectores de características próximos dentro del espacio de entrada [26].

De la combinación de ambas definiciones, se establece como **Red Preservadora de la Topología (RPT)** a aquella red A cuyos mapeados ϕ_w y $\phi_w{}^{-1}$ son preservadores de la vecindad.

En [26] se plantea que a partir de esta definición de RPT, y analizando los diferentes modelos se obtiene que:

- Los KSOM no son preservadores de la topología como tradicionalmente han sido considerados. Dado a que esto solamente ocurriría en el caso de que la topología del mapa y del espacio de entrada coincidieran. Esto se debe a que la topología de la red es establecida a priori, desconociendo posiblemente la topología del espacio de entrada.
- Las GCS de Fritzke tampoco son RPT ya que la topología de red está establecida a priori (triángulos, tetraedros, etc.). Aunque sí mejora el funcionamiento respecto a los mapas de Kohonen, debido a su capacidad de inserción y eliminación de neuronas.
- El grafo resultante del aprendizaje competitivo de las NG y de las GNG es una triangulación de Delaunay inducida (ver Figura 9), subgrafo de la triangulación de Delaunay, en el que únicamente se encuentran las aristas de la triangulación de Delaunay cuyos puntos pertenezcan al espacio de entrada de V. En [41] se demuestra que ambos modelos son RPT.

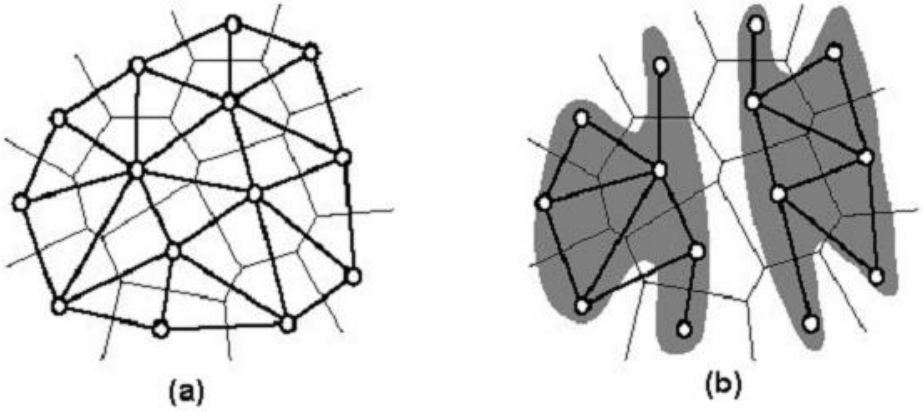

Figura 9 (a) Triangulación de Delaunay, (b) Triangulación de Delaunay inducida. (Fuente: [41])

Por tanto para el primer aspecto a considerar los modelos que mejor representan la topología del espacio de entrada serían el NG y el GNG, coincidiendo con lo planteado en [26] y [29].

El segundo aspecto a considerar es el relacionado con el coste computacional que requiere cada uno de dichos modelos para confeccionar la red neuronal. En estudios realizados en este sentido tanto en [29] como en [26] se hace una comparación entre los diferentes modelos analizados.

Tabla 3 Resultados de pruebas a los modelos de aprendizaje

	Tiempo (seg) ESTRELLA	Tiempo (seg) CIRCULO	Tiempo (seg) ESPIRAL
KSOM	36	28	42
GCS	12	7	21
NG	109	95	148
GNG	14	8	25

En [29] se comparan los diferentes modelos usando como entrada de las pruebas espacios que representan diferentes figuras, el resultado de estas pruebas realizadas se muestra en la Tabla 3.

En [26] se hace un estudio similar reportando como resultado que el proceso de aprendizaje para el modelo NG posee una complejidad muy superior a la GNG, por ejemplo para una red con un número final de 100 neuronas, se comportan de la siguiente manera (Tabla 4):

Tabla 4 Tiempos de aprendizaje de los diferentes modelos auto-organizativos con 100 neuronas

Modelo auto-organizativo	Tiempo de aprendizaje (seg)
KSOM	38
GCS	9
NG	117
GNG	12

Se puede decir que las redes auto-organizativas que mejor preservan la topología de un espacio de entrada son los Gases Neuronales y entre estos las GNG tienen un mejor rendimiento respecto al tiempo de aprendizaje.

1.5 Detalles del algoritmo Growing Neural Gas

Según lo expresado por [34] para definir una red se considera que la misma consiste en:

- Un conjunto A de unidades (o nodos). Cada unidad $c \in A$ tiene asociado un vector de referencia $w_c \in \mathbb{R}^n$. El vector de referencia puede ser visto como la posición del espacio de entrada de la unidad correspondiente.
- Un conjunto N de conexiones (o aristas) entre pares de unidades. Estas conexiones no son ponderadas. Su único propósito es la definición de una estructura topológica.

Además, existe un número (posiblemente infinito) de señales de entrada obedeciendo alguna función de densidad de probabilidad $P(\xi)$ desconocida.

La idea principal de este método es el de sucesivamente adicionar nuevas unidades a una red inicialmente pequeña por la evaluación local de medidas estadísticas obtenidas durante los pasos previos de adaptación. Este es el mismo acercamiento seguido por el modelo de GCS el cual, sin embargo, posee una topología con una dimensionalidad fija.

En este acercamiento la topología de red es generada incrementalmente mediante CHL y tiene una dimensionalidad que depende de los datos de entrada y que puede variar localmente.

A continuación se brinda el algoritmo GNG:

1. Comenzar con dos unidades a y b en posiciones w_a y w_b en $\mathbb{R}^n$.
2. Generar una señal de entrada de acuerdo con $P(\xi)$.
3. Encontrar la unidad s_1 más cercana y la unidad s_2 segunda más cercana.
4. Incrementar la edad de todas las aristas que emanan de s_1.
5. Adicionar el cuadrado de la distancia entre la señal de entrada y de la unidad más cercana en el espacio de entrada a una variable contador local.
   $$\Delta error(s_1) = \left\| w_{s_1} - \xi \right\|^2$$
6. Mover s_1 y sus vecinos directos topológicamente a través de ξ por las fracciones ϵ_b y ϵ_n, respectivamente, de la distancia total:
   $$\Delta w_{s_1} = \epsilon_b(\xi - w_{s_1})$$
   $$\Delta w_n = \epsilon_n(\xi - w_n)\ \forall n \in N_{s_1}$$

Donde N_{s_1} son los vecinos topológicos de s_1.

7. Si s_1 y s_2 están conectados por una arista, establecer la edad de esta arista a cero. Si dicha arista no existe crearla.
8. Eliminar las aristas con una edad superior a a_{max}. Si esta operación resulta en un nodo del que no emanan aristas entonces eliminarlo también.
9. Si el número de señales de entradas generadas es un entero múltiplo de un parámetro λ, insertar una nueva unidad como sigue:
    - Determine la unidad q con el error acumulado máximo.
    - Insertar una nueva unidad r a medio camino entre q y su vecino f con la variable de error mayor:
    $$w_r = 0.5\ (w_q + w_f)$$
    - Insertar aristas conectando la nueva unidad r con las unidades q y f, y eliminar la arista original entre q y f.
    - Disminuir la variable de error de q y f multiplicándola con una constante α. Inicializar la variable de error de r con el nuevo valor de la variable de error de q.
10. Disminuir todas las variables de error multiplicándolas por la constante d.
11. Si un criterio de parada (por ejemplo tamaño de la red o alguna medida de rendimiento) no es alcanzado ir al paso 2.

Los pasos de adaptación a través de las señales de entrada (6) llevan a un movimiento general de todas las unidades hacia aquellas áreas del espacio de entrada de donde vienen las señales ($P(\xi) > 0$). La inserción de aristas (7) entre la unidad más cercana y la segunda más cercana con respecto a una señal de entrada genera una conexión simple de la "triangulación de Delaunay inducida" con respecto a la posición actual de todas las unidades.

La eliminación de aristas (8) es necesaria para librar aquellas aristas que ya no forman parte de la triangulación de Delaunay inducida porque sus puntos finales se han movido. Esto es logrado por el envejecimiento local de aristas (4) alrededor de la unidad más cercana combinada con reseteado de la edad (7) de esas aristas que ya existen entre la unidad más cercana y la segunda más cercana.

Con la inserción y la eliminación de aristas el modelo trata de construir y luego seguir la triangulación de Delaunay inducida que es un movimiento lento del objetivo debido a la adaptación de los vectores de referencia.

La acumulación de los cuadrados de las distancias (5) durante la adaptación ayuda a identificar unidades que reposan en áreas del espacio de entrada donde el mapeo de señales a unidades provoca un gran error. Para reducir ese error, nuevas unidades son insertadas en esas regiones.

1.6 Conclusiones del capítulo

A partir de los elementos expuestos anteriormente se arriban a las siguientes conclusiones:

- En el estudio realizado sobre el diseño de WSANs en entornos interiores se tiene la captura de requisitos y al diseño automático y optimización como los pasos fundamentales a ser automatizados. Un aspecto central en los procesos de diseño de WSAN es la generación de posiciones candidatas.
- En la literatura existen varios enfoques para generar de forma automática de posiciones candidatas de enrutadores de WSANs en entornos interiores. Por

sus características las redes auto-organizadas son las más adecuadas para resolver el problema planteado.

- Las redes auto-organizativas que mejor preservan la topología de un espacio de entrada son los Gases Neuronales. Entre los Gases Neuronales las Growing Neural Gas tienen un mejor rendimiento respecto al tiempo de aprendizaje.

CAPÍTULO 2 PROPUESTA DE SOLUCIÓN

En el presente capítulo se realiza la propuesta de un algoritmo para la generación de las posiciones candidatas de los enrutadores de una WSAN para entornos interiores. Esta propuesta se sustenta en el algoritmo de aprendizaje de redes auto-organizadas Growing Neural Gas, así como en la modificación de varias de sus características. Se formaliza el problema a resolver y se describe el algoritmo que permite la solución a dicho problema.

2.1 Modelación del problema

Para modelar el problema se asumen las siguientes premisas:

- El modelo del edificio (IFC) es importado por la herramienta Andrómeda y el área de monitoreo es delimitada por el usuario.
- Un nodo sensor n_k (dispositivo final o *end device*), incluye varios tipos de sensores (temperatura, CO_2, etc.). Cada propuesta de diseño de red incluye un total de l (≥ 1) nodos sensores. El conjunto de nodos sensores es denotado por $N = \{n_k \mid 1 \leq k \leq l\}$.
- Un enrutador r_i trabaja como intermediario entre N y un Coordinador c. Cada propuesta de diseño de red obtenida incluye un total de δ (≥ 0) nodos enrutadores. El conjunto de nodos enrutadores es denotado por $R = \{r_i \mid 0 \leq i \leq \delta\}$.
- Los elementos en $N \cup \{c\}$ serán colocados manualmente por el diseñador, y las posiciones candidatas serán generadas solamente para los elementos en R.

Definición 1 Restricciones de posicionamiento: Se denotará por C al conjunto de restricciones para las posibles ubicaciones de los elementos en R. Para cada r_i, se denota a su conjunto de restricciones como C_{r_i} ($C_{r_i} \subseteq C$). Se asumirá que la siguiente restricción siempre se cumple:

$$\left|C_{r_i}\right| \geq 1, \forall r_i \in R \tag{1}$$

Una vez establecido lo anterior se puede plantear como **Formulación del problema:**

Sea A un área 3D en un entorno interior, y $A_s \subseteq A$ un área de monitoreo que tiene la intención detectar magnitudes físicas o químicas; se quiere obtener una discretización de A_s mediante la obtención de un grafo $G = (V, E)$.

En esta instancia $V(\neq \emptyset)$ representa las posiciones de los elementos en N, las posiciones candidatas donde los elementos en R (sujetos a la **Definición 1**) pueden ser ubicados y la ubicación de c ($V = N \cup R \cup \{c\}$); las aristas en E ($\neq \emptyset$) especifican las conexiones entre ellos. En este caso d ($\in V$) y d' se comunican, en otras palabras $(d, d') \in E$, cuando:

$$\theta(d, d') = true \tag{2}$$

Donde θ es una función booleana que implementa el modelo de propagación seleccionado para la simulación (por ejemplo los empleados en [42]).

2.2 Estructura general de la propuesta

La solución propuesta (ver Figura 10) está basada en el algoritmo GNG. Esta técnica permite crear un grafo a partir de una distribución de datos en $\mathbb{R}^n$, haciendo posible la tarea de discretizar el área de monitoreo.

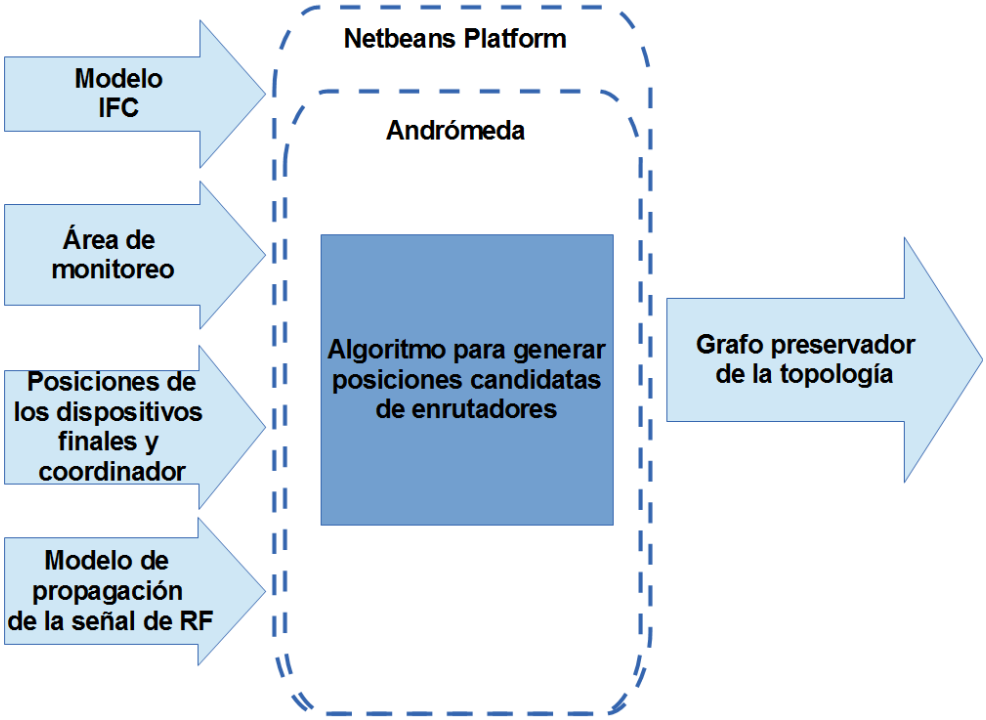

Figura 10 Esquema general de la propuesta.

El algoritmo deberá integrarse en la plataforma Andrómeda la que a su vez estará basada en la plataforma de cliente rico de Netbeans.

Para el funcionamiento del algoritmo propuesto se tiene establecido como entradas el modelo IFC de la edificación sobre la que se va a proponer el diseño de la WSAN, el área que será objeto del monitoreo, las posiciones de los dispositivos finales y el coordinador y por último el modelo de propagación de la señal de RF que permitirá concluir si dos dispositivos se pueden comunicar.

Como resultado de la ejecución del algoritmo se obtendrá un grafo denominado *Grafo preservador de la topología*. Dicho elemento se define como el grafo cuyos nodos están situados en los puntos asociados a cada uno de los elementos de procesamiento, estableciendo una arista entre aquellos nodos cuyos elementos de procesamiento respectivos están conectados [26].

2.3 Estructura del algoritmo propuesto

El algoritmo propuesto estará compuesto por seis pasos fundamentales. De forma general el algoritmo parte de la modificación del GNG para adaptarlo a las condiciones propias del problema planteado. De esta manera los pasos 1, 2 y 3 del algoritmo poseen modificaciones respecto al original y además se agregan los pasos 5 y 6. En la Figura 11 se esbozan sus pasos, que serán definidos posteriormente.

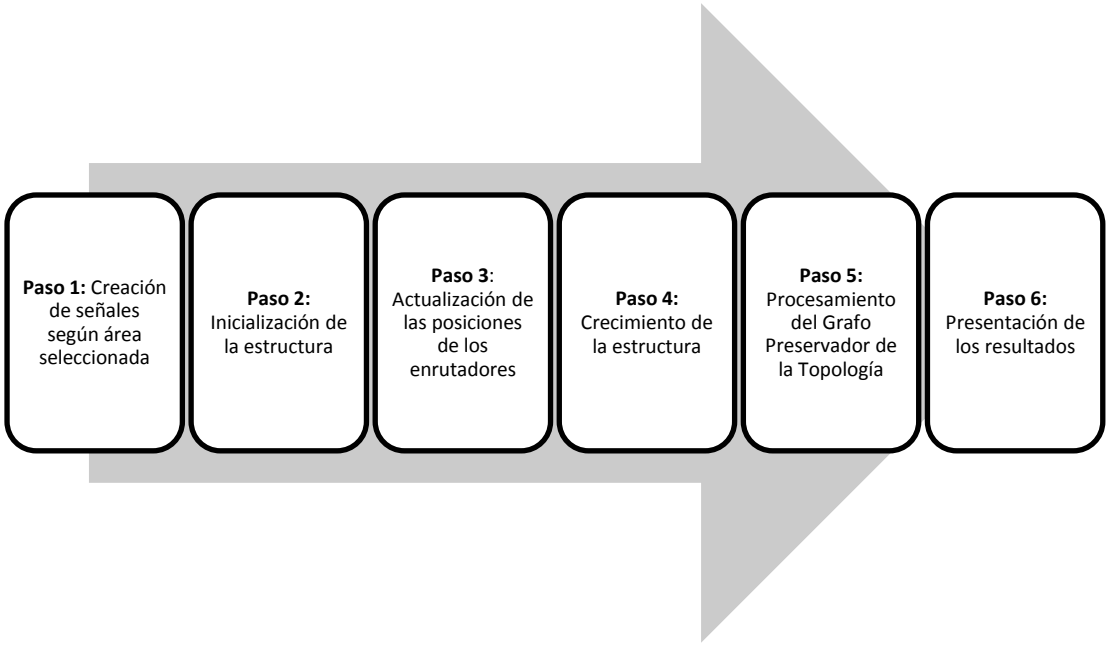

Figura 11 Pasos lógicos para la construcción del algoritmo.

Estos componentes del algoritmo se relacionarán formando un ciclo con los pasos 3 y 4, hasta que se cumpla el criterio de parada. El resto de los pasos se ejecutarán de forma secuencial.

2.3.1 Elementos principales del algoritmo propuesto

El algoritmo propuesto utiliza los siguientes elementos para su funcionamiento:

1. Mecanismo de obtención de señales de entrada según el área seleccionada y la posición de los dispositivos finales y el coordinador.
2. Inicialización de la estructura. En este aspecto se crea las condiciones iniciales que poseerá la estructura de la red.
3. Actualización de las posiciones de los nodos de la estructura. En este paso las posiciones de los nodos de la estructura se mueven en dirección de la señal generada por el mecanismo propuesto en el paso 1.
4. Para alcanzar la cantidad de posiciones candidatas de enrutadores deseada es necesario que la estructura crezca. Este paso es el encargado de llevar a cabo esta tarea.
5. En este paso de manera opcional se puede realizar tareas de transformación del grafo propuesto hacia diferentes tipos de grafos que puedan ser de interés para el diseñador.
6. El objetivo de esta etapa es la presentación de los resultados obtenidos en la herramienta Andrómeda; así como la tarea de exportar este resultado a un formato estándar para que sea usado por posteriores etapas del diseño de la WSAN.

2.3.2 Precondiciones

Para el correcto funcionamiento del algoritmo propuesto en la herramienta Andrómeda es necesario tener como elementos definidos:

1. El modelo IFC del edificio donde se va a realizar el despliegue.
2. Definición del área de monitoreo.
3. Determinación de las posiciones de los dispositivos finales y el coordinador deben estar establecidas.

4. Definición del modelo de propagación de la señal de Radio Frecuencia.

2.3.3 Entradas y salidas del algoritmo propuesto

Las **entradas** del algoritmo son el área de monitoreo así como las posiciones de los dispositivos finales y el coordinador de la WSAN. Ambos elementos deben ser definidos por el diseñador sobre el modelo IFC que fue previamente cargado por la herramienta Andrómeda. Es necesario qué el diseñador elija que modelo de propagación será usado. Es necesario, también, definir por parte del diseñador la cantidad de posiciones candidatas que desea.

Como **salida** del algoritmo se obtendrá un grafo no dirigido el cuál representará el conjunto de posiciones candidatas de los enrutadores, el conjunto de dispositivos finales y el coordinador que conformarán la WSAN.

2.4 Descripción formal del algoritmo propuesto

En el presente epígrafe se definirán formalmente los pasos que componen el algoritmo propuesto.

2.4.1 Paso 1 Creación de señales según área seleccionada

Como se explicó en el epígrafe 2.3.3 para comenzar el algoritmo es necesario tener como entrada el área de monitoreo y las posiciones candidatas de los dispositivos finales. El objetivo principal de los enrutadores consiste en asegurar la conectividad entre los dispositivos finales y el controlador (*sink*) por tanto no tiene sentido ubicar enrutadores por fuera del área que comprende la envolvente convexa originada por los puntos que representan los dispositivos finales y el coordinador (ver Figura 12). Todas las señales que se generen deben estar dentro de la envolvente convexa que forman los dispositivos finales.

Los aspectos necesarios para realizar este paso del algoritmo propuesto son:

1. Obtener la envolvente convexa que determinan las posiciones de los dispositivos finales.
2. Obtener un mecanismo que permita generar señales dentro de la envolvente convexa.

Esta manera de generar las señales de entrada del algoritmo constituye una novedad de la presente investigación.

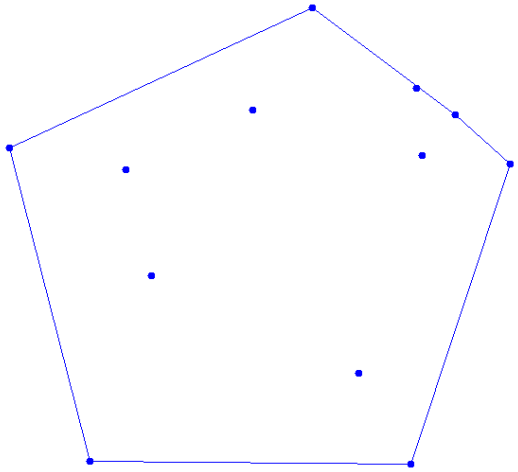

Figura 12 Envolvente convexa de una nube de puntos.

Envolvente convexa

La envolvente convexa de un subconjunto S del plano es el menor polígono convexo que contiene a todos los puntos de S [43]. Para obtener la envolvente convexa existen múltiples acercamientos. En la siguiente tabla se muestra una comparativa entre los métodos:

Tabla 5 Algoritmos para obtener la envolvente convexa.

Referencia	Algoritmo	Complejidad temporal (caso peor)
[44]	Graham's Scan	$O(n \log n)$
[45]	Quick Hull	$O(n \log n)$ (en el caso peor $O(n^2)$)
[46]	Sharif	$O(n \log n)$

En [46] se realiza un estudio comparativo del rendimiento de la ejecución de estos algoritmos para diferentes cantidades de puntos (ver Figura 13). En esta figura la etiqueta "*New technique"* se refiere al algoritmo de Sharif. Como se puede apreciar el rendimiento del algoritmo de Sharif posee mejores resultados para el conjunto de puntos analizados, por esta razón se procede a definir los elementos necesarios para utilizar dicho algoritmo para obtener la envolvente convexa a partir del conjunto de dispositivos finales más el coordinador de la WSAN.

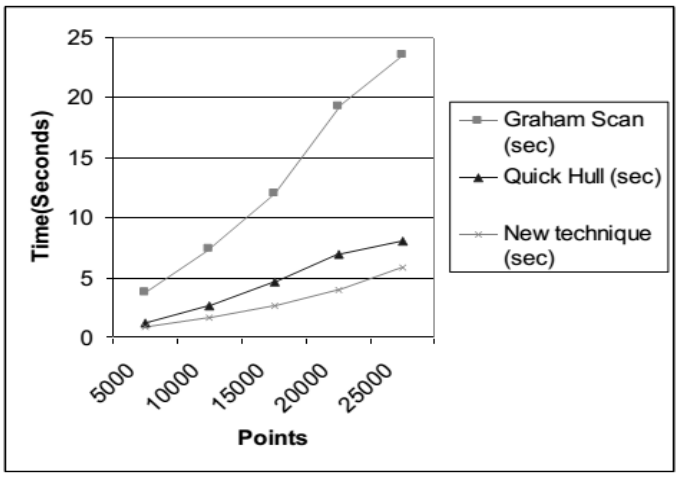

Figura 13 Comparativa entre los algoritmos para determinar envolvente convexa. (Fuente: [46])

El seudo-código del Algoritmo de Sharif es el siguiente:

Tabla 6 Seudo-código del algoritmo Sharif para obtener la envolvente convexa.

```
Algoritmo Sharif(P)
Entrada: Un conjunto de puntos P en el plano.
Salida: Una lista conteniendo los puntos de la envolvente convexa.
1-Encontrar los puntos con X_min, X_max, Y_min, Y_max del conjunto de puntos.
2-Constuir un cuadrilátero usando esos puntos.
3-Decartar los puntos dentro del cuadrilátero formando un nuevo conjunto
n_new.
4-Ordenar los puntos (respecto al eje x), denotados como (P_1,P_2,...,P_n).
5-Añadir P_1 y P_2 a una pila.
6-Mientras i ≤ n {
       Si (x_i realiza una giro a la izquierda relativo a los dos elementos
                                                    en el tope de la pila)
      entonces {
             Añadir xi a la pila.
             Incrementar i.
       } sino {
             Eliminar los dos últimos puntos añadidos a la pila y
                                                              descartar.
       }
}
```

Como resultado del algoritmo anterior se obtiene un polígono P representado por un arreglo de n puntos $P_0, P_1, \dots, P_n = P_0$, el cual constituye la envolvente convexa del conjunto de dispositivos terminales de la WSAN.

Obtención de señales dentro de la envolvente convexa

Luego que se tiene determinada el área en la cual se deben obtener las señales de entrada para el algoritmo, es necesario poseer un mecanismo que permita obtener las señales dada la envolvente convexa. Para ello es necesario saber si un punto pertenece al interior del polígono que forma la envolvente convexa.

En [47] se realiza una clasificación de los algoritmos que permiten determinar la presencia de un punto dentro de un polígono en: “sin pre-procesamiento” y “con pre-procesamiento”.

En la categoría “sin pre-procesamiento” la inclusión dentro del polígono es determinada mediante cierto parámetro como *números de cruce del rayo (ray crossing number)*, *signo de desbalance (the sign of offset)* o la orientación de un punto respecto a los lados del polígono.

Los algoritmos en la categoría “con pre-procesamiento” surgen para probar contra el mismo polígono un número grande de puntos. Estos algoritmos igualmente obtienen la prueba de inclusión de cierto parámetro, pero la evaluación del parámetro es acelerado mediante el pre-procesamiento del polígono. Generalmente, el paso de pre-procesamiento descompone el polígono de interés en componentes más simples como cuadrículas, trapezoides, triángulos, polígonos convexos, polígonos en forma de estrella, celdas de tamaño uniforme o aristas de nivel. Entonces una consulta de inclusión puede ser eficientemente obtenida por el componente en la vecindad de un punto de prueba.

Esta última categoría es la que más se ajusta al problema, debido a que es necesario probar una gran cantidad de puntos. Según [47] el método de pre-

procesamiento más usado es el de subdivisión uniforme; por ejemplo se emplea en [48], [49] y el propio [47].

En la Tabla 7 se muestra una comparación de la complejidad temporal de estos métodos. En dicha tabla se considera que n es la cantidad de vértices del polígono en el que se debe realizar la prueba.

Como se puede apreciar no existen grandes diferencias en cuanto a la complejidad temporal de estos algoritmos pero como se muestra en [48] el rendimiento del algoritmo que utiliza los puntos centrales de la cuadrícula presenta mejores resultados en cuanto a los tiempos de ejecución.

Tabla 7 Comparación entre los algoritmos que implementan la prueba de inclusión en un polígono.

Referencia	Algoritmo	Complejidad temporal Pre-procesamiento	Complejidad temporal Prueba de inclusión[7]
[47]	Basado en el punto casi cercano	$O(n\sqrt{n})$	$O(1)$ o $O(\sqrt{n})$
[48]	Puntos centrales de la cuadrícula (GCP, por sus siglas en inglés).	$O(n\sqrt{n})$	$O(1)$ o $O(\sqrt{n})$
[49]	Contención en células (CBCA, de las siglas en inglés)	$O(n\sqrt{n})$	$O(1)$

Debido a lo antes planteado se decide utilizar el algoritmo GCP para determinar la presencia de un punto dentro del polígono que representa el área de monitoreo.

El algoritmo GCP de forma general se divide en dos etapas: el pre-procesamiento y la prueba de inclusión. En la Tabla 8 se muestra el algoritmo para el pre-procesamiento del polígono.

Tabla 8 Algoritmo para pre-procesamiento según el algoritmo GCP.

```
Algoritmo pre-procesamiento GCP(P)
Entrada: Un conjunto de puntos P en el plano que representa un polígono.
Salida: Se obtiene una cuadrícula con los puntos centrales de cada celda y
la información de si está dentro o fuera del polígono.

 1. Decidir la resolución de la cuadrícula para subdividir el área en
    celdas de tamaño uniforme.
 2. Registrar cada celda con lados del polígono que la atraviese.
 3. Examinar cada eje para encontrar cuáles celdas cruza (para ello se usa
    el método descrito en [50]).
 4. Determinar el estado de los puntos centrales del polígono
    4.1 Establecer el estado para las celdas en la frontera de la
        cuadrícula.
    4.2 Partiendo de estas celdas en la frontera, se tratan las celdas
        restantes en secuencia hasta que todas hayan sido tratadas.
```

[7] La segunda alternativa se refiere al peor caso, sin embargo estos algoritmos se comportan como $O(1)$ en la generalidad de los casos.

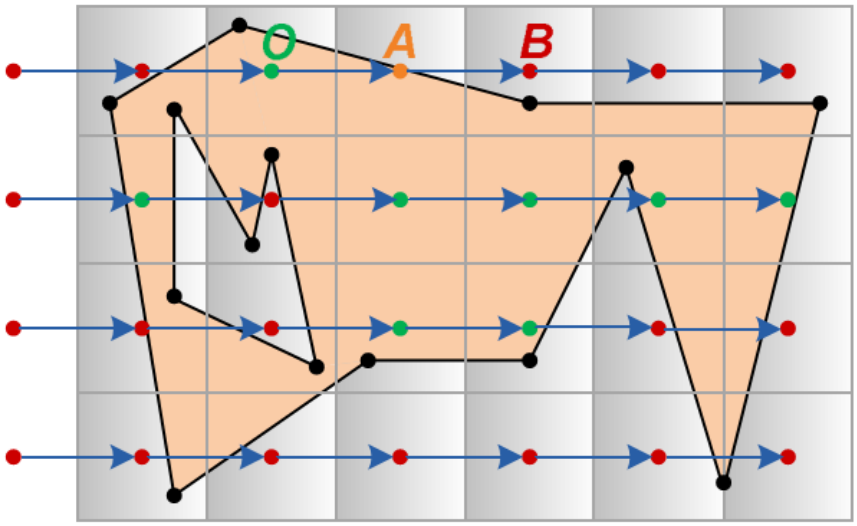

Figura 14 Ejemplo de cuadrícula donde se aplica el algoritmo GCP. (Fuente: [48])

En el paso 1 del algoritmo descrito anteriormente es necesario establecer los límites de la cuadrícula un poco más grandes que los límites del polígono. En el paso 4.1 se inicia por las celdas en la frontera izquierda de la cuadrícula porque es sencillo obtener el valor de estos puntos usando los puntos vecinos que están fuera de la cuadrícula (ver Figura 14). En el paso 4.2 se van analizando las celdas vecinas según el orden (ver flechas azules de la Figura 14).

En la Tabla 9 se muestra el algoritmo para la prueba de inclusión del punto en el polígono:

Tabla 9 Algoritmo para la prueba de inclusión GCP.

Algoritmo prueba de inclusión **GCP**(pto)
Entrada: Coordenadas de un punto para comprobar que se encuentra en dentro del polígono
Salida: Un booleano que representa la inclusión o no del punto en el polígono.

1. Si el centro de la celda es singular[8] entonces
   Se busca una celda desde la celda del punto de prueba en dirección horizontal o vertical, hasta que se encuentre un punto central no singular.
   sino
   Elegir el centro de la celda que contiene el punto de prueba
2. Crear un segmento entre el punto de prueba y el punto central no singular elegido y determinar la propiedad de inclusión para el punto de prueba.

[8] Un punto es singular si se encuentra en una arista del polígono.

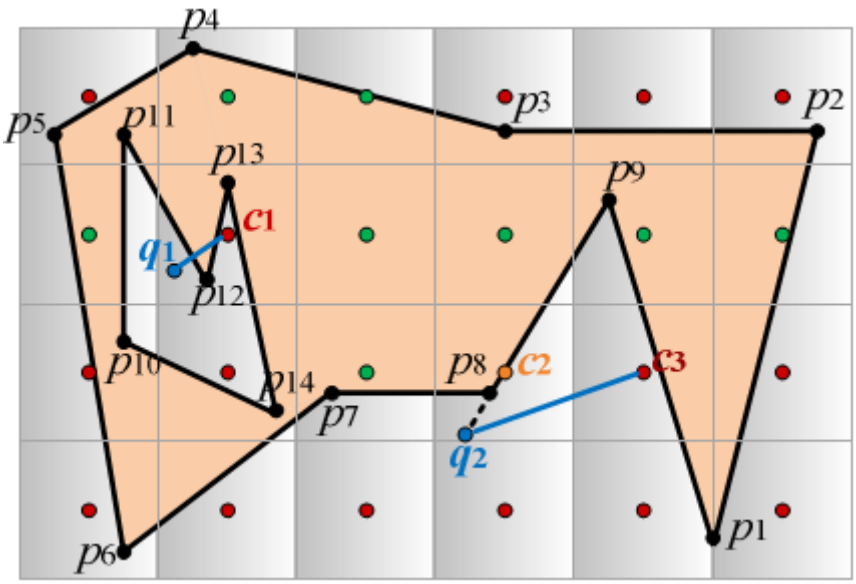

Figura 15 Prueba de inclusión del algoritmo GCP. (Fuente:[48])

En el algoritmo propuesto se refiere como singular a un punto central que se encuentre sobre un lado del polígono; por ejemplo del punto $c2$ en la Figura 15. La acción que se realiza si el punto de prueba se encuentra en una celda con un punto central singular se ve ejemplificada en la Figura 15 a través del punto $q2$ cuyo centro es $c2$ y se debe trasladar hasta el punto $c3$.

Finalmente para generar las señales de entrada para el algoritmo se procede a obtener la envolvente convexa a partir de los dispositivo finales y el coordinador utilizando el algoritmo de Sharif [46] y luego se generan las señales mediante la creación de puntos aleatorios comprobando que se encuentren dentro del polígono utilizando el algoritmo GCP propuesto en [48].

2.4.2 Paso 2 Inicialización de la estructura

Para la inicialización de la estructura según el algoritmo GNG se debe comenzar el proceso usando dos unidades ubicadas en las posiciones aleatorias obtenidas a partir de los patrones de entrada del algoritmo. En el algoritmo se inicializará la estructura con las dos posiciones iniciales, pero además se añadirá como elementos iniciales de la estructura los dispositivos finales y el coordinador, los cuales constituirán elementos fijos dentro del grafo.

2.4.3 Paso 3 Actualización las posiciones de los enrutadores

El tercer paso dentro del proceso general es la de actualizar las posiciones candidatas de los enrutadores; también se podría referir a este proceso como actualización de la estructura. El algoritmo para actualizar las posiciones de los enrutadores es:

Tabla 10 Algoritmo para actualizar las posiciones de los enrutadores.

Algoritmo para actualizar las posiciones de los enrutadores **Entrada**: Grafo con las n unidades (vértices), señal de entrada (ξ). **Salida**: Grafo con n unidades con su posición actualizada (vértices). 1. Encontrar la unidad s_1 más cercana y la unidad s_2 segunda más cercana. 2. Incrementar la edad de todas las aristas que emanan de s_1. 3. Adicionar el cuadrado de la distancia entre la señal de entrada y de la unidad más cercana en el espacio de entrada a un contador local.

$$\Delta\text{error}(s_1) = \left\|w_{s_1} - \xi\right\|^2$$

4. Mover s_1 y sus vecinos directos topológicamente a través de ξ por las fracciones ϵ_b y ϵ_n, respectivamente, de la distancia total:
$$\Delta w_{s_1} = \epsilon_b(\xi - w_{s_1})$$
$$\Delta w_n = \epsilon_n(\xi - w_n)$$
para todos los vecinos directos n de s_1
5. Si s_1 y s_2 están conectados por una arista, establecer la edad de esta arista a cero. Si dicha arista no existe crearla.
6. Eliminar las aristas con una edad superior a a_{max}. Si esta operación resulta en un nodo del que no emanan aristas entonces eliminarlo también.

En el paso 1 se utilizará como función de distancia para determinar la cercanía entre dos dispositivos el modelo de propagación de la radio frecuencia (RF) que previamente haya sido elegido por el diseñador de red. Esta selección del modelo de RF como función de distancia es debido a que es necesario asegurar la comunicación entre los dispositivos que conformen la red y puede suceder que a pesar que los dispositivos se encuentren cerca según la distancia euclidiana estos queden incomunicados mediante algún obstáculo que impida la comunicación entre ellos (por ejemplo, una pared). Para las operaciones relacionadas con los pasos 3 y 4 se utilizará como medida de distancia la euclidiana. Otro detalle importante a declarar es que en los movimientos que se efectúen en el paso 4 si el elemento que se va a mover es un dispositivo final o el coordinador se aborta el movimiento. De igual manera en el paso 6 si el nodo que quedó aislado es un dispositivo final no se elimina.

2.4.4 Paso 4 Crecimiento de la estructura

Es necesario que la estructura o grafo crezca a lo largo del tiempo. Para ello cada cierta cantidad de generación de señales de entrada (múltiplo de una variable λ) se debe añadir una nueva unidad en un punto intermedio entre la unidad con error acumulado máximo y su vecino topológico con mayor error acumulado. El algoritmo para lograr este crecimiento se muestra en la siguiente tabla:

Tabla 11 Algoritmo para el crecimiento de la estructura.

Algoritmo para el crecimiento de la estructura
Entrada: Grafo con las n unidades (vértices), constante de control de crecimiento (λ), constante de disminución de error (α), constante de disminución del error global (d).
Salida: Grafo con n+1 unidades con su posición actualizada (vértices).
1. Si el número de señales de entradas generadas es un entero múltiplo de un parámetro λ, insertar una nueva unidad como sigue:
   1.1. Determine la unidad q con el error acumulado máximo.
   1.2. Insertar una nueva unidad r a medio camino entre q y su vecino f con la variable de error mayor:
$$w_r = 0.5\,(w_q + w_f)$$
   1.3. Insertar aristas conectando la nueva unidad r con las unidades q y f, y elimine la arista original entre q y f.
   1.4. Disminuir la variable de error de q y f multiplicándola con una constante α. Inicialice la variable de error de r con el nuevo valor de la variable de error de q.
2. Disminuir todas las variables de error multiplicándolas por la constante d.

Luego que se termine este proceso es necesario comprobar que se ha alcanzado el criterio de parada para terminar el ciclo que forman los pasos 3 y 4 del algoritmo general. Entre estos criterios de parada se pueden tener:

- Tamaño de la red (número de unidades).
- Cantidad de señales generadas.
- Número de iteraciones

La decisión de cuál criterio de parada emplear queda a consideración del diseñador de la WSAN.

2.4.5 Paso 5 Procesar el Grafo Preservador de la Topología

El grafo obtenido del proceso realizado por los pasos anteriores se denomina Grafo Preservador de la Topología (GPT). En [26] se expresa que un GPT establece una triangulación de Delaunay inducida de los vértices del grafo.

Además se establece la siguiente relación entre diferentes tipos de grafos:

$$NNG \subseteq MST \subseteq RNG \subseteq GG \subseteq DT$$

donde NNG es el grafo de vecinos más cercanos, MST es el árbol de expansión mínimo, RNG es el grafo de vecindad relativa, GG es el grafo de Gabriel y DT es la triangulación de Delaunay. Cada grafo tiene sus propias características entre las que se encuentran:

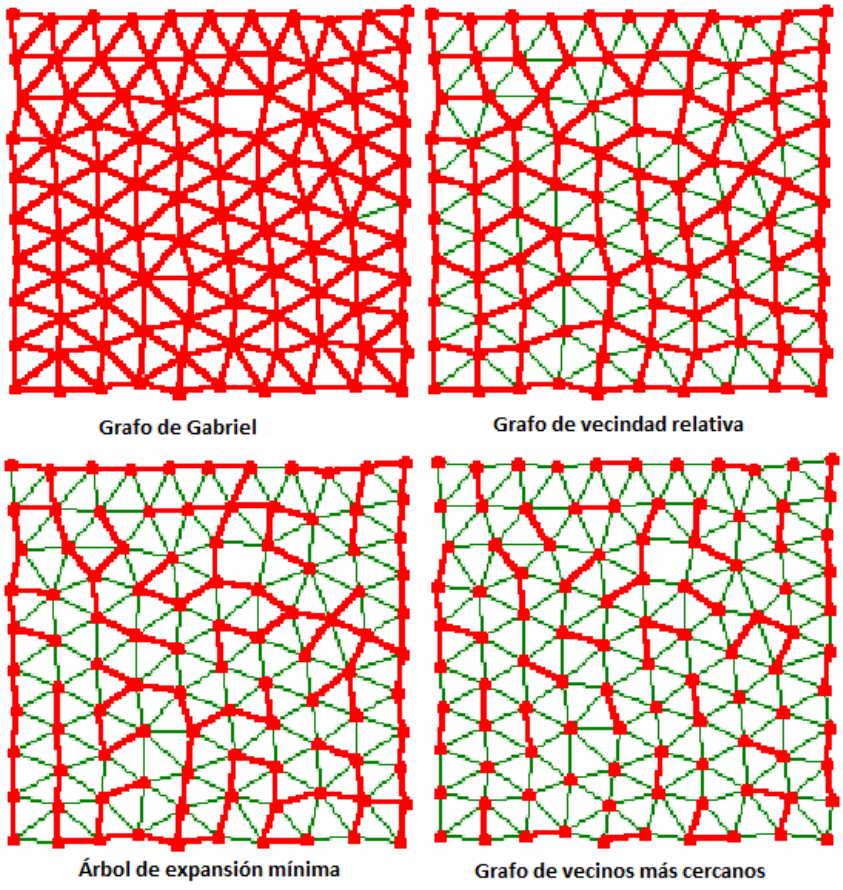

Figura 16 Subgrafos de la triangulación de Delaunay. (Fuente: [26])

- El Grafo de Gabriel (GG) de un conjunto S de puntos expresa una noción de proximidad o cercanía de esos puntos. Formalmente, este es el grafo con un conjunto de vértices S donde cualesquiera sean los puntos P y Q en S, son adyacentes exactamente si estos son distintos y donde el disco cerrado cuyo diámetro es el segmento $\overline{PQ}$ no contiene otros puntos de S [51]. Entre algunas de sus características más importantes se encuentra que este grafo tiene la distancia más corta entre todos los pares de nodos, lo cual es de interés en el área de las redes de comunicaciones. Es conocido que estos grafos pueden ser obtenidos en tiempo lineal si se tiene la triangulación de Delaunay [52].
- El Grafo de Vecindad Relativa (RNG, por sus siglas en inglés) es un sub-grafo de la Triangulación de Delaunay. Es un grafo no dirigido definido por un conjunto de puntos conectando dos puntos p y q mediante una arista donde quiera que no exista un tercer punto r que sea más cercano a ambos de lo que son entre ellos. De igual manera este tipo de grafo puede ser obtenido en tiempo lineal partiendo de la triangulación de Delaunay [53].
- Dado un grafo conectado no dirigido, un *árbol de expansión* de ese grafo es un sub-grafo que es un árbol y conecta todos los vértices. Un Árbol de Expansión Mínimo *(*MST*,* de las siglas en inglés*)* es entonces un árbol de expansión con peso menor o igual que cada uno de los restantes árboles de expansión. Un subtipo utilizado es el *Árbol de Expansión Mínimo Euclidiano* (EMST, de las siglas en inglés) que se define como un MST donde el peso de la arista entre cada par de puntos es la distancia entre ellos. Este tipo de grafo ha tenido aplicaciones directas en el diseño de redes, incluyendo redes de computadores, redes de telecomunicaciones y redes eléctricas. Para construir este tipo de grafo se puede usar el algoritmo propuesto en [54].
- El Grafo de Vecinos más Cercanos (NNG, de las siglas en inglés) para un conjunto de n objetos P es un grafo no dirigido con P siendo su conjunto de vértices y con una arista de p a q donde quiera que q sea el vecino más cercano de p. El NNG encuentra su aplicación en diversas ramas entre las que se encuentran la minería de datos y aprendizaje automático. Para la construcción de este tipo de grafo es factible seguir la propuesta de [55]

Se le permitirá al criterio del diseñador cuál de estos tipos de grafos presentar a la siguiente etapa del algoritmo.

2.4.6 Paso 6 Presentar resultados

Una vez que se haya terminado de procesar el GPT es necesario entonces realizar dos operaciones fundamentales: Visualizar el GPT obtenido en la herramienta Andrómeda y Exportar el GPT a un formato estándar para presentarlo al algoritmo de optimización. Se pasará a analizar cada una de estas tareas por separado.

Visualizar el GPT en la herramienta Andrómeda

Para poder visualizar el GPT en la herramienta Andrómeda se necesita inicialmente dominar la estructura que posee dicha herramienta. La misma está implementada usando la plataforma Netbeans; su estructura es basada en diferentes módulos como se puede apreciar en la Figura 17. El módulo que se encarga de realizar la representación gráfica de los elementos correspondientes al modelo IFC es el visor (Viewer), para ello hace uso de los elementos de los módulos 3D-Model e IfcModel. Para poder representar en el visor de Andrómeda es necesario entonces usar dos clases fundamentales: *NonStructuralElement* y *LineBetweenPoints* pertenecientes

al módulo 3D-Model las cuales permitirán agregar los nodos y las aristas del GPT al modelo IFC y por tanto representar el grafo en el visor.

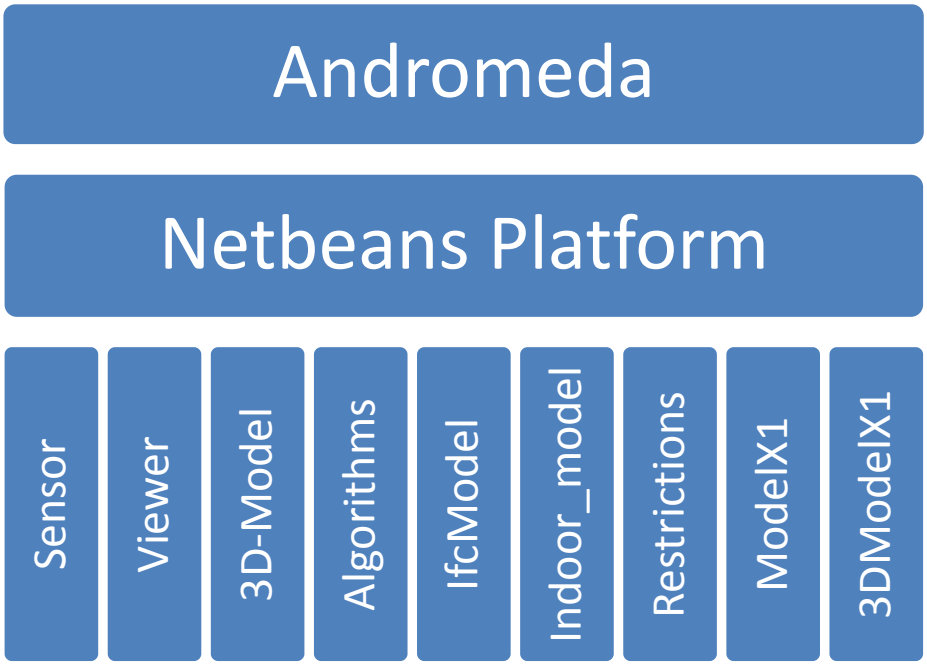

Figura 17 Estructura de la herramienta Andromeda. (Fuente: Elaboración propia)

El algoritmo para representar gráficamente el GPT en el visor será el siguiente:

Tabla 12 Algoritmo para representar gráficamente el GPT en el visor de Andrómeda.

Algoritmo para representar gráficamente el GPT en el visor de Andrómeda
Entrada: Grafo Preservador de la Topología (n vértices y m aristas).
Salida:
1. Para cada vértice hacer:
   - Crear un nuevo objeto *NonStructuralElement* con las coordenadas del vértice que se esté analizando.
   - Añadir el nuevo objeto al modelo IFC que se encuentre cargado en el proyecto en curso
2. Para cada arista hacer:
   - Crear un nuevo objeto *LineBetweenPoints* pasándole como parámetro los dos vértices que forma cada arista.
   - Añadir el nuevo objeto al modelo IFC que se encuentre cargado en el proyecto en curso

Exportar el GPT a un formato estándar

Una vez que el GPT ha sido visualizado correctamente en la herramienta Andrómeda es necesario exportar dicho grafo para que el mismo sea procesado por los algoritmos posteriores o por cualquier otra herramienta que quiera hacer uso del mismo.

En [56] se plantea que a pesar de los esfuerzos referentes a la confección de un estándar para representar grafos ningún acuerdo sobre un formato determinado ha sido ampliamente aceptado y de hecho muchas herramientas soportan solamente un número limitado de formatos los cuales son restringidos típicamente en su expresividad y son específicos a un área de aplicación. Para enfrentar los

problemas tratados anteriormente surge en el año 2000 el formato GraphML [57]. GraphML es un formato de fichero basado en XML y soporta todo el rango de posibles estructuras de grafos incluidos grafos dirigidos, no dirigidos y mixtos.

Un fichero GraphML consiste en un XML conteniendo un elemento <graph>, en la que se encuentra una secuencia no ordenada de elementos <node> y <edge>. Cada elemento <node> debe tener un atributo *id* distinto, y cada elemento <edge> tiene atributos *source* y *target* que identifican los puntos finales de cada arista al tener los *id* de esos puntos finales. Otras características del lenguaje GraphML permiten a los usuarios especificar cuando una arista o vértice es dirigido o no, y permite además asociar datos adicionales con los vértices o aristas. Este formato ha sido empleado en múltiples investigaciones entre las que se encuentran [58-60]. En la Tabla 13 se muestra un ejemplo de un fichero con el formato de GraphML.

Tabla 13 Estructura de un archivo GraphML.

```
<?xml version="1.0" encoding="UTF-8"?>
<graphml xmlns="http://graphml.graphdrawing.org/xmlns"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://graphml.graphdrawing.org/xmlns
     http://graphml.graphdrawing.org/xmlns/1.0/graphml.xsd">
  <graph id="G" edgedefault="undirected">
    <node id="n0"/>
    <node id="n1"/>
    <edge id="e1" source="n0" target="n1"/>
  </graph>
</graphml>
```

En la herramienta Andrómeda se hace uso del marco de trabajo JUNG[9] [61]. Este marco de trabajo (en lo adelante *framework*) permite la representación de un grafo usando el API de Java. Entre el grupo de funcionalidades presentes en este framework se encuentra la de exportar grafos en el formato GraphML. Para ello se cuenta con una clase llamada *GraphMLWriter* la cual posee el método *save* que recibe como parámetros el grafo que se desea salvar y un objeto de la clase *PrintWriter* y salva el grafo usando el formato de fichero planteado. En la Tabla 14 se muestra el método a seguir para salvar el grafo al formato GraphML usando el marco de trabajo JUNG.

Tabla 14 Algoritmo para salvar un grafo a GraphML usando el framework JUNG.

```
GraphMLWriter<LearningModelNode, LearningModelEdge> graphWriter
                                          = new GraphMLWriter<>();
PrintWriter out = new PrintWriter(new BufferedWriter(
                                           new FileWriter(file)));
graphWriter.save(graph, out);
```

2.5 Conclusiones del capítulo

- El algoritmo propuesto permite la generación de las posiciones candidatas de los enrutadores para una WSAN en entornos interiores.
- La estructura denominada Grafo Preservador de la Topología permitió almacenar las posiciones candidatas de los enrutadores de la WSAN.
- El algoritmo propuesto es novedoso debido a que las señales de entrada son generadas haciendo uso de la envolvente convexa y del método puntos centrales de la cuadrícula; se hace uso de modelos de propagación de la

[9] De las siglas en inglés de Java Universal Network/Graph Framework

señal de RF como medida de distancia entre dos nodos; se realiza la inicialización de la estructura con más de dos elementos y se usa el estándar GraphML para exportar el resultado obtenido.

- El uso del marco de trabajo JUNG así como la plataforma de cliente rico Netbeans facilitaron integración del algoritmo propuesto en la herramienta Andrómeda.

CAPÍTULO 3 VALIDACIÓN DE LA PROPUESTA

En este capítulo se presenta la aplicación del algoritmo en la herramienta Andrómeda y se realiza la validación del algoritmo propuesto. Se plasma una valoración de la complejidad temporal del algoritmo propuesto y se realiza la validación de la propuesta de solución, para ello se aplican diferentes pruebas estadísticas t-Student sobre los resultados del experimento realizado.

3.1 Solución propuesta

El algoritmo propuesto fue integrado en la herramienta Andrómeda. Como se plantea en el epígrafe 2.2 las entradas al algoritmo son: el modelo IFC previamente cargado, las áreas de monitoreo, las posiciones de los dispositivos finales y el coordinador, y el modelo de propagación de las señales de RF. En la Figura 18 se muestran tres de las cuatro precondiciones que necesita el algoritmo. Una vez que estos elementos están listos, se permite al diseñador elegir el modelo de propagación de la señal y la cantidad de posiciones candidatas que desea (ver Figura 19).

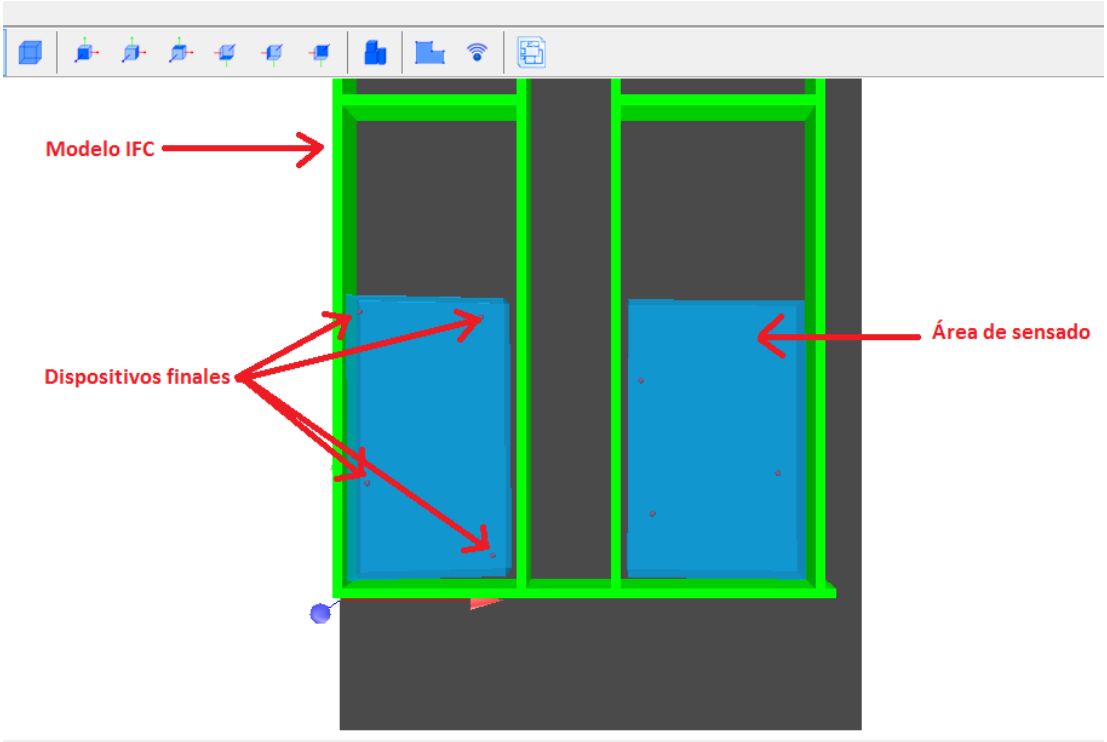

Figura 18 Precondiciones para el algoritmo propuesto en la herramienta Andrómeda.

Figura 19 Selección del modelo de propagación y cantidad de posiciones candidatas.

Una vez ejecutado el mismo se generan las posiciones candidatas para los enrutadores de forma tal que se asegura que todos los dispositivos se encuentren conectados entre sí, y que la topología formada describa el área formada por la envolvente convexa de los dispositivos finales.

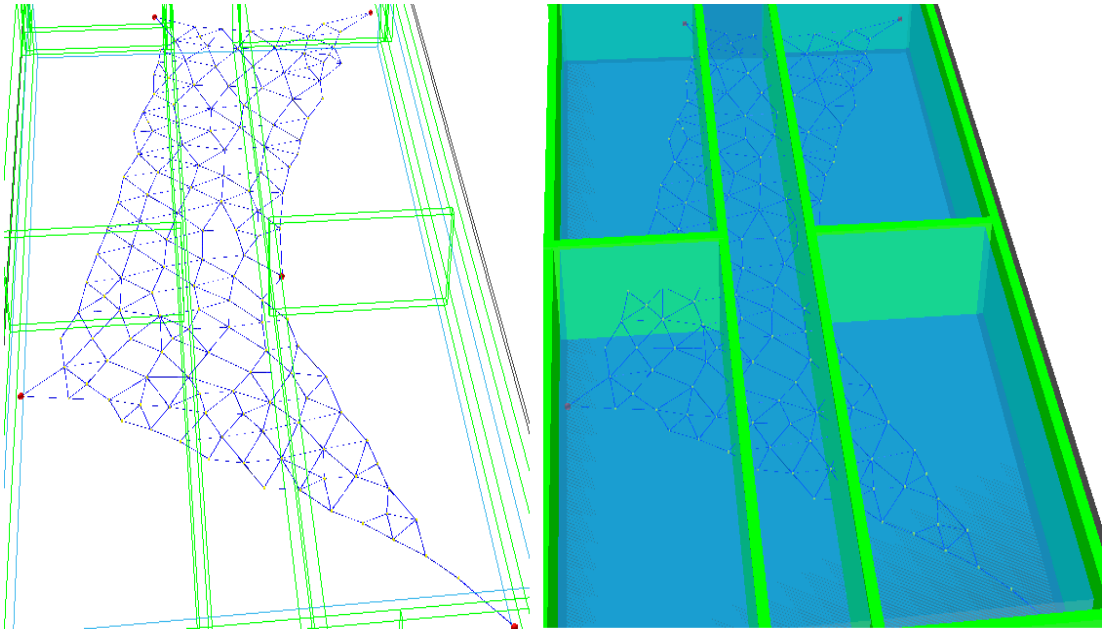

Figura 20 Resultado de la ejecución del algoritmo.

3.2 Valoración del costo computacional del algoritmo propuesto

El uso eficiente de recursos suele medirse en dos parámetros fundamentales: espacio y tiempo. Ambos aspectos representan los costos que supone encontrar la solución al problema planteado mediante el algoritmo.

La complejidad espacial es una medida de la cantidad de memoria de trabajo que el algoritmo necesitará. En la actualidad esta medida no constituye una limitante en cuanto al rendimiento de los algoritmos, por tanto se analiza la complejidad temporal del algoritmo propuesto.

3.2.1 Análisis del tiempo de ejecución del algoritmo propuesto

Para realizar un análisis del caso peor del algoritmo propuesto se determinó el tiempo requerido para cada operación básica dentro de la propuesta (ver Tabla 15).

Tabla 15 Resumen de los tiempos de ejecución por cada operación básica.

Paso del algoritmo	Operación básica	Coste temporal
Paso 1	Envolvente convexa	$O(n \log n)$
	Generación de señales - Preprocesamiento - Prueba de inclusión	 $O(m\sqrt{m})$ $O(1)$
Paso 2	Inicialización de la estructura	$O(1)$
Paso 3	Actualizar las posiciones de los enrutadores	$O(k)$
Paso 4	Crecimiento de la estructura	$O(k)$
Paso 5	Procesar el Grafo Preservador de la Topología - Grafo de Gabriel	$O(k)$ $O(k)$

	- Grafo de Vecindad Relativa - Árbol de Expansión Mínima - Grafo de Vecinos más Cercanos	$O(k \log k)$ $O(k^{1.14})$
Paso 6	Visualizar el Grafo Preservador de la Topología	$O(k^2)$
	Exportar el Grafo Preservador de la Topología	$O(k^2)$

En la tabla anterior, se define n como la cantidad de dispositivos finales insertados por el usuario, m es la cantidad de vértices de la envolvente convexa y k es la cantidad de nodos del Grafo Preservador de la Topología (GPT). Los resultados en cuanto al coste computacional establecidos en el paso 5 están dados por los tiempos computacionales reportados por los autores de los algoritmos elegidos.

En el paso 1 hay que tomar en cuenta que la prueba de inclusión es un método auxiliar que realmente será usado en el paso 3 del algoritmo, pero como su complejidad temporal es $O(1)$ según la propiedad del producto de O grande se queda como cota para ese paso $O(k)$.

Los pasos 3 y 4 forman un ciclo que tiene como condición de parada alcanzar la cantidad de posiciones candidatas señaladas por el diseñador (k). Una posición es añadida cada λ iteraciones, por tanto los pasos 3 y 4 se repetirán un total de λk veces siendo λ una constante. Por lo que la complejidad temporal del ciclo formado por los pasos 3 y 4 será de $O(k^2)$, aplicando primeramente la propiedad de la suma de O grande y luego la propiedad del producto de O grande.

El paso 5 del algoritmo es opcional, pero considerando el caso peor se incurre en un costo computacional de $O(k \log k)$.

Finalmente el paso 6 tiene como coste computacional conjunto $O(k)$.

Por todo lo antes expresado podemos obtener como coste computacional total del algoritmo:

$$O(n \log n) + O\left(m\sqrt{m}\right) + O(1) + O(k^2) + O(k \log k) + O(k^2) \quad (1)$$

Al aplicar la propiedad de la suma de O grande en (1) se obtiene una nueva expresión:

$$O(n \log n) + O\left(m\sqrt{m}\right) + O(k^2) \quad (2)$$

Los vértices de la envolvente convexa están acotados por la cantidad de dispositivos finales que se reciben como entrada del algoritmo, por lo que se cumple la siguiente relación:

$$m \leq n \quad (3)$$

Debido a (3) podemos decir, sin perder generalidad, que el algoritmo propuesto está acotado por la siguiente expresión:

$$O(n \log n) + O\left(n\sqrt{n}\right) + O(k^2) \quad (4)$$

Y nuevamente aplicando la propiedad de la suma de O grande en (4) obtenemos finalmente el coste computacional del algoritmo propuesto:

$$O(n\sqrt{n} + k^2) \quad (5)$$

Podemos decir entonces que el coste computacional del algoritmo propuesto dependerá del tamaño de las entradas correspondientes a la cantidad de dispositivos finales y a la cantidad de posiciones candidatas de enrutadores que desee obtener el diseñador.

3.3 Validación del algoritmo propuesto

En esta sección se realiza la validación del algoritmo propuesto. Se definen los elementos necesarios para realizar la validación de la hipótesis propuesta y se detallan los resultados obtenidos.

3.3.1 Muestreo

La población estará definida por los conjuntos de dispositivos finales (de hasta 20 dispositivos) con posiciones generadas aleatoriamente dentro de un área de monitoreo en un entorno interior representado por su modelo IFC.

La muestra será el 100% de los conjuntos de dispositivos finales generados. Para elegir la muestra se utilizará Muestreo Aleatorio Simple.

3.3.2 Diseño del experimento

Se utilizará un diseño experimental de tipo pre-experimento, siguiendo la tipología establecida en [63]. En especial se empleará pre-experimento de tipo 1 *estudio de casos con una sola medición*. La decisión de realizar este tipo de diseño experimental se debe en lo fundamental a que no es posible contar con un grupo de control sobre el cual realizar las comparaciones posteriores.

Para probar la efectividad de la variable dependiente Preservación de la topología usaremos como métricas el Producto Topográfico (TP) [64] y el Promedio del Cuantificador del Error [65]. Ambas métricas han sido ampliamente usadas para comprobar la preservación de la topología en mapas auto-organizados [64-67]. Para probar la efectividad de la variable dependiente conectividad entre dispositivos y controlador se usará como métrica la cantidad de componentes conexas presentes en el grafo obtenido.

Realizaremos la comprobación de las hipótesis planteadas utilizando la prueba estadística t-Student debido a que el tamaño de la muestra es pequeño respecto al total de posibles combinaciones de las variables de entrada del algoritmo, la varianza es desconocida y los datos siguen una distribución normal.

3.3.3 Análisis de los resultados

Para realizar las pruebas de hipótesis se realizaron un total de 120 ejecuciones al algoritmo propuesto en la herramienta Andrómeda, usando la siguiente distribución para los parámetros de entrada:

Tabla 16 Valores de los parámetros de entrada para los experimentos.

Cantidad de dispositivos finales	Cantidad de posiciones candidatas
5,10,15,20	50, 100,150

Para cada posible combinación de los parámetros se realizaron 10 ejecuciones lo que da el total de ejecuciones planteada.

Para cada una de las corridas del algoritmo se obtuvo el tiempo de ejecución, el promedio del cuantificador del error, el producto topográfico y la cantidad de componentes conexas. El conjunto de datos obtenidos de la ejecución de las pruebas antes planteada puede ser apreciado en el Anexo 1 Resultados de las pruebas realizadas.

Producto Topográfico

Según [64] si el grafo obtenido de la ejecución del algoritmo preserva la topología el valor del Producto Topográfico (TP) está en la cercanía de 0. Por tanto, se pasa a probar que como promedio el TP se encuentra en el rango:

$$-0.5 < TP < 0.5 \quad (1)$$

Para demostrar (1) lo realizaremos en dos etapas: primero demostraremos que se cumple que $-0.5 < TP$ (2) y en una segunda etapa demostraremos que $0.5 > TP$ (3). La distribución para el producto topográfico se comporta de la siguiente manera:

Tabla 17 Resumen de la variable Producto Topográfico en el conjunto de prueba.

Min	1st Q	Median	Mean	3rd Q	Max
-0.71760	-0.52230	-0.45610	-0.43590	-0.34250	-0.07203

Antes de establecer qué tipo de prueba de hipótesis vamos a usar, debemos comprobar que tipo de distribución describen los datos. Para ello realizaremos la prueba Shapiro-Wilk[10] [68] para probar si el conjunto de datos del Producto Topográfico describe una distribución normal (ver Tabla 18).

Tabla 18 Aplicando prueba Shapiro-Wilk para comprobar la normalidad del atributo TopographicProduct.

```
> shapiro.test(datos$TopographicProduct)
        Shapiro-Wilk normality test

data:  datos$TopographicProduct
W = 0.9882, p-value = 0.3914
```

Como el valor-p es mayor que 0.05 se asume que los datos provienen de una distribución normal. En las siguientes imágenes (Figura 21) se muestra la distribución de los datos según la densidad y el gráfico QQ de dicha métrica; se aprecia la cercanía a la distribución normal de los datos obtenidos.

[10] Esta prueba parte de la hipótesis nula de que “los ejemplos vienen de una distribución normal” contra la hipótesis alternativa de que “los datos no vienen una distribución normal”.

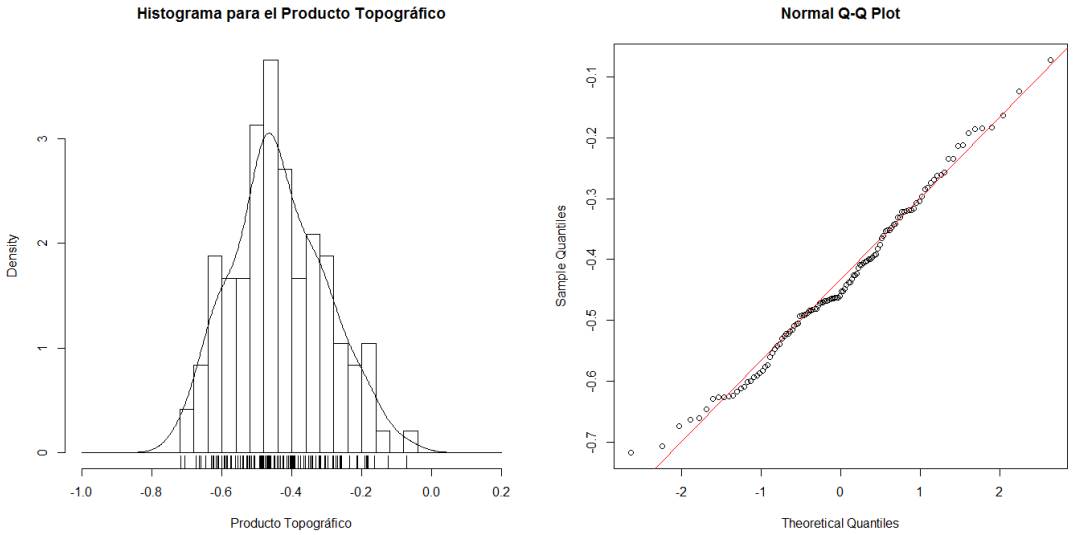

Figura 21 Histograma y gráfico QQ para el Producto Topográfico.

Pasamos entonces a probar (2) estableciendo la siguiente hipótesis nula y alternativa:

$$H_0: \mu \leq -0.5$$

$$H_1: \mu > -0.5$$

Como no es conocida la varianza para toda la población, dado que técnicamente es intratable obtener y comprobar todas las posibles combinaciones de ubicaciones de los dispositivos finales, utilizaremos la prueba t-Student para una muestra, con el fin de aprobar o rechazar la hipótesis nula. Además el tamaño de la muestra elegida es pequeña respecto al total de posibles combinaciones de cantidad de dispositivos finales y de cantidad de posiciones candidatas.

Para realizar la prueba utilizaremos el lenguaje de programación R y ejecutando la siguiente prueba:

Tabla 19 Prueba t-Student con una muestra para probar (2).

```
t.test(datos$TopographicProduct, alternative="greater",mu=-0.5)
```

Y como resultado de esta prueba obtenemos:

Tabla 20 Resultado de la prueba para (2).

```
      One Sample t-test

data:  datos$TopographicProduct
t = 5.2349, df = 119, p-value = 3.6e-07
alternative hypothesis: true mean is greater than -0.5
95 percent confidence interval:
 -0.456191       Inf
sample estimates:
 mean of x
-0.4358885
```

Como se puede apreciar se rechaza la hipótesis nula y por tanto la hipótesis alternativa es verdadera.

De forma análoga probaremos (3) a través de las siguientes hipótesis nula y alternativa:

$$H_0: \mu \geq 0.5$$

$$H_1: \mu < 0.5$$

De igual manera usando la prueba t-Student de una muestra (ver Tabla 21) obtenemos el resultado mostrado en la Tabla 22.

Tabla 21 Prueba t-Student con una muestra para probar (3).

```
t.test(datos$TopographicProduct, alternative="less",mu=0.5)
```

Tabla 22 Resultado de la prueba para (3).

```
      One Sample t-test

data:  datos$TopographicProduct
t = -76.4182, df = 119, p-value < 2.2e-16
alternative hypothesis: true mean is less than 0.5
95 percent confidence interval:
       -Inf -0.4155861
sample estimates:
 mean of x
-0.4358885
```

De igual manera se rechaza la hipótesis nula y por tanto la hipótesis alternativa es verdadera.

De esta manera queda probado lo planteado en (1) y podemos afirmar que el grafo obtenido por el algoritmo propuesto preserva la topología.

Promedio del Cuantificador del Error

Según [69] la precisión con que un mapeado representa su entrada puede ser medido mediante el promedio del cuantificador del error. El cuantificador del error está dado por el cuadrado de la distancia de la señal de entrada al punto más cercano a la misma dentro del espacio de salida. En la siguiente ecuación podremos ver cómo quedaría esta función de error para el algoritmo Growing Neural Gas[34]:

$$E = \|w_b - \xi\|^2$$

Donde ξ es la señal de entrada y w_b es el vector de referencia del nodo más cercano a ella dentro del espacio de salida. Y luego el promedio del cuantificador del error sería [66]:

$$E = \sum_{\forall\, \xi \in \mathbb{R}^d} \|w_b - \xi\|^2 * p(\xi)$$

Donde $p(\xi) = 1/n$ y n es la cantidad de señales generadas.

La distribución para esta métrica, obtenida de la ejecución de las pruebas, puede ser apreciada en el Anexo 1 Resultados de las pruebas realizadas. En la Tabla 23 se muestra el resumen de la distribución y en la Figura 22 se muestran el histograma y la gráfica QQ.

Tabla 23 Resumen de la variable Promedio del Cuantificador del Error en el conjunto de prueba.

Min	1st Q	Median	Mean	3rd Q	Max
0.007601	0.008328	0.009817	0.010520	0.012920	0.014990

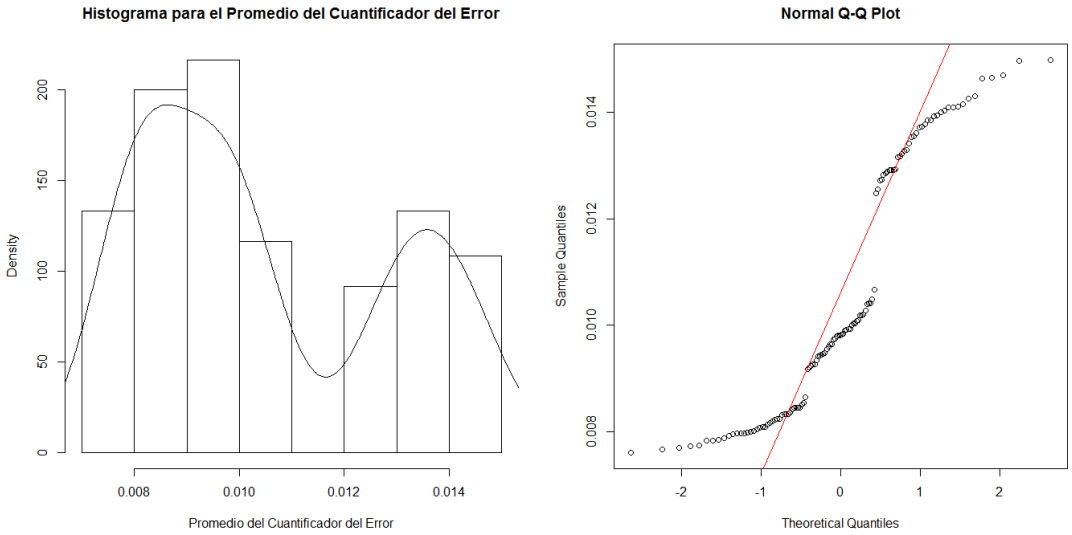

Figura 22 Histograma y gráfica QQ para la métrica Promedio del Error de Cuantificación.

Como se puede apreciar en este caso los datos no siguen una distribución normal. Por lo que pasamos a dividir la distribución de datos agrupándolos según la cantidad de posiciones candidatas a generar (Tabla 24). En la Tabla 25 se encuentra la aplicación de las pruebas de Shapiro-Wilk para cada uno de los conjuntos de datos obtenidos y como se puede apreciar el valor-p para cada uno es mayor que 0.05 y por tanto podemos afirmar que siguen una distribución normal y podemos aplicar la prueba t-Student de manera independiente a cada uno de los conjuntos obtenidos.

Tabla 24 Comando para dividir el conjunto de datos según la cantidad de nodos a generar.

```
avg50<-datos$AverageQuantizationError [ datos$CantNodos == 50]
avg100<-datos$AverageQuantizationError [ datos$CantNodos == 100]
avg150<-datos$AverageQuantizationError [ datos$CantNodos == 150]
```

Tabla 25 Prueba de normalidad para cada conjunto de datos.

```
> shapiro.test(avg50)

      Shapiro-Wilk normality test

data:  avg50
```

```
W = 0.9632, p-value = 0.2154

> shapiro.test(avg100)

      Shapiro-Wilk normality test

data:  avg100
W = 0.9738, p-value = 0.4718

> shapiro.test(avg150)

      Shapiro-Wilk normality test

data:  avg150
W = 0.9743, p-value = 0.4859
```

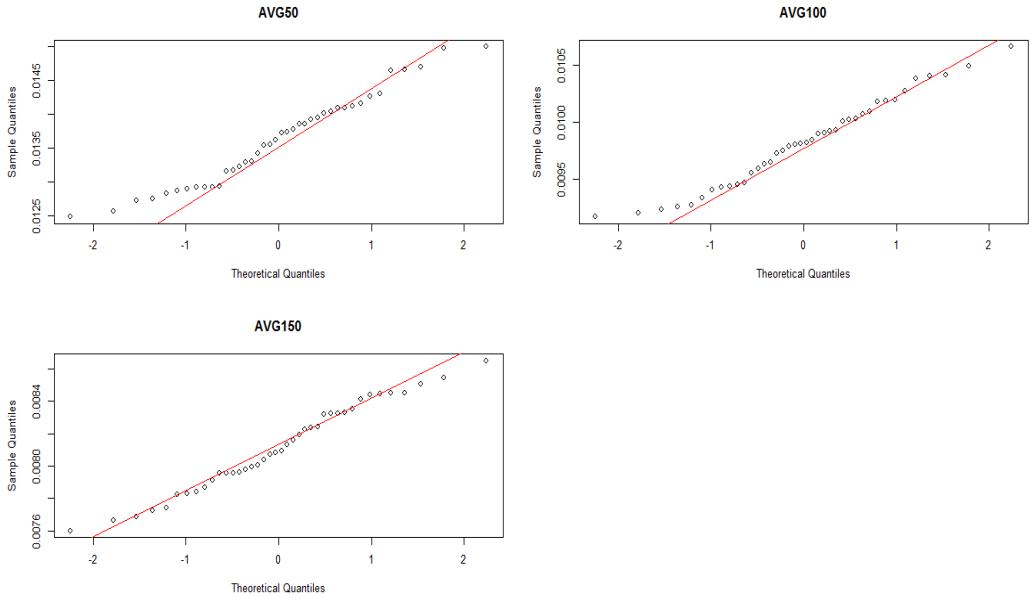

Figura 23 Gráficos QQ para las distribuciones de datos obtenidas.

Mientras menor sea el error de cuantificación del error como promedio mejor el algoritmo propuesto mapeará señales de entrada a uno de los elementos en el espacio de salida. En [70] empieza a ser considerado como buen valor de esta métrica valores por debajo de 0.2; por otro lado en [71] se utiliza como buenos valores por debajo de 0.05. Para comprobar el cumplimiento de este indicador diremos que aceptamos como bueno el algoritmo si el promedio del cuantificador de error está por debajo de 0,05.

Pasaremos a probar entonces la siguiente hipótesis nula y alternativa para cada conjunto de datos obtenido:

$$H_0: \mu \geq 0.05$$
$$H_1: \mu < 0.05$$

En la Tabla 26 se muestra el resultado de aplicar la prueba t-Student con la hipótesis nula y alternativa propuesta.

Tabla 26 Pruebas t-Student para cada conjunto de datos generado.

```
> t.test(avg50, alternative="less",mu=0.05)

      One Sample t-test

data:  avg50
t = -338.5274, df = 39, p-value < 2.2e-16
alternative hypothesis: true mean is less than 0.05
95 percent confidence interval:
       -Inf 0.01380022
sample estimates:
 mean of x
0.01361915

> t.test(avg100, alternative="less",mu=0.05)

      One Sample t-test

data:  avg100
t = -645.0706, df = 39, p-value < 2.2e-16
alternative hypothesis: true mean is less than 0.05
95 percent confidence interval:
        -Inf 0.009923171
sample estimates:
 mean of x
0.00981822

> t.test(avg150, alternative="less",mu=0.05)

      One Sample t-test

data:  avg150
t = -975.9485, df = 39, p-value < 2.2e-16
alternative hypothesis: true mean is less than 0.05
95 percent confidence interval:
        -Inf 0.008185327
sample estimates:
  mean of x
0.008113013
```

Como se puede observar cada conjunto de datos rechazan la hipótesis nula y por tanto se prueba que la media de cada uno de los conjuntos de datos está por debajo del valor propuesto. Por lo anteriormente expresado podemos concluir que el algoritmo se comporta de manera correcta respecto a la métrica Promedio del Cuantificador del Error.

Cantidad de componentes conexas

Para poder medir la conectividad entre los dispositivos finales y el coordinador, se utilizará como medida la cantidad de componentes conexas formadas por la aplicación del algoritmo. Si existe más de una componente conexa en el grafo obtenido como parte del algoritmo propuesto entonces existen al menos dos puntos que no serán alcanzables entre ellos.

Para comprobar la existencia de más de una componente conexa se utiliza el algoritmo Búsqueda Primero a lo Ancho (BFS) que posee implementado el marco de trabajo JUNG. Si al finalizar la ejecución de este algoritmo sobre el grafo resultante de la aplicación del algoritmo propuesto, existe algún dispositivo final que no ha sido visitado entonces el grafo tiene más de una componente conexa.

En el Anexo 1 Resultados de las pruebas realizadas se puede apreciar los resultados obtenidos para este indicador. Como se puede observar en todos los casos se obtuvo como resultado una componente conexa. Por lo que podemos concluir que se cumple con lo planteado para este indicador.

3.4 Conclusiones del capítulo

Como resultados de la validación del algoritmo propuesto se obtuvo que:

- Se demostró la capacidad del algoritmo para generar las posiciones candidatas de los enrutadores de una WSAN en entornos interiores.
- El algoritmo propuesto posee una complejidad temporal de tiempo polinomial igual a $O(n\sqrt{n} + k^2)$.
- La aplicación del pre-experimento permitió evaluar el algoritmo propuesto comprobando el cumplimiento de los objetivos propuestos.
- La aplicación de las métricas Producto Topográfico, Promedio del Cuantificador del Error y la Cantidad de Componentes Conexas permitió validar el comportamiento del algoritmo. Como resultado se comprobó el cumplimiento de la hipótesis planteada.
- La aplicación de la prueba estadística t-Student a los resultados obtenidos del experimento permitió medir las diferentes métricas empleadas.

CONCLUSIONES GENERALES

Después de realizar la presente investigación se concluye:

- A partir de la sistematización de los referentes teóricos sobre el proceso de diseño de WSANs en entornos interiores y en particular de la generación de las posiciones candidatas de los enrutadores se encontraron varios enfoques para resolver el problema planteado. Aparecen las redes auto-organizadas como el mejor enfoque existente, en especial el algoritmo Growing Neural Gas destaca como el método más adecuado.
- Se obtuvo un algoritmo que se adapta a las necesidades del problema planteado, utilizando un total de seis pasos generales y el mismo se encuentra integrado sobre la plataforma Andrómeda.
- El algoritmo propuesto es novedoso debido a que: las señales de entrada son generadas haciendo uso de la envolvente convexa y del método puntos centrales de la cuadrícula; se hace uso de modelos de propagación de la señal de RF como medida de distancia entre dos nodos; se realiza la inicialización de la estructura con más de dos elementos y se usa el estándar GraphML para exportar el resultado obtenido.
- El algoritmo propuesto posee una complejidad temporal de tiempo polinomial igual a $O(n\sqrt{n} + k^2)$.
- La aplicación del pre-experimento permitió evaluar el algoritmo propuesto comprobando el cumplimiento de los objetivos propuestos.
- El algoritmo fue probado mediante el uso de las métricas Producto Topográfico, Error del cuantificador y Cantidad de componentes conexas que miden el comportamiento de las variables dependientes.

RECOMENDACIONES

- Valorar el uso de nuevos mecanismos de propagación de la señal de RF para medir la distancia entre los dispositivos enrutadores.
- Aplicar la métrica Función Topográfica para medir el grado de preservación de la topología.
- Valorar el uso de nuevas métricas para medir la conectividad entre los elementos de la WSAN.
- Probar el algoritmo propuesto sobre en nuevos entornos y medir su comportamiento.
- Realizar la implementación de otros mecanismos para la generación de posiciones candidatas de enrutadores que permitan realizar comparaciones con la propuesta de esta investigación.

GLOSARIO DE TÉRMINOS

BEMS: Siglas en inglés para Sistemas de Gestión de Energía en Edificios.

Envolvente Convexa: La envolvente convexa de un subconjunto S del plano es el menor polígono convexo que contiene a todos los puntos de S.

IFC: Es una especificación neutral y abierta que no es controlada por un solo proveedor o grupo de proveedores. Es un formato de fichero basado en objetos con un modelo de datos desarrollado por buildingSMART para facilitar la interoperatibilidad en la industria de la arquitectura, ingeniería y la construcción, y es un formato comúnmente usado para el modelado de información de edificios (Building Information Modeling, BIM). La especificación del modelo IFC está disponible y abierta. Se encuentra registrada por ISO y es un estándar internacional oficial bajo el nombre ISO 16739:2013.

JUNG: De las siglas en inglés de Java Universal Network/Graph Framework.

Producto topográfico: Es una medida de la preservación de las relaciones de vecindad entre diferentes espacios.

RF: Radio Frecuencia, se aplica a la porción menos energética del espectro electromagnético, situada entre unos 3 kHz y unos 300 GHz.

WSN: Siglas en inglés de Red de Sensores Inalámbricos (Wireless Sensor Network). Consiste en sensores autónomos espacialmente distribuidos que miden condiciones físicas y ambientales, tales como temperatura, sonido, presión, etc. y de forma cooperativa pasan sus datos de forma cooperativa a través de la red hacia un punto principal.

WSAN: Siglas en ingles de Red Inalámbrica de Sensores y Actuadores (Wireless Sensor and Actuators Network). Usualmente consiste en un grupo de nodos sensores que son usados para obtener información del ambiente, y nodos actuadores que son usados para cambiar el comportamiento del ambiente[19].

ZigBee: Especificación de un conjunto de protocolos de alto nivel de comunicación inalámbrica para su utilización con radiodifusión digital de bajo consumo, basada en el estándar IEEE 802.15.4 de redes inalámbricas de área personal (*wireless personal area network*, WPAN).

REFERENCIAS BIBLIOGRÁFICAS

1. Jankowska, A., M.C. Schut, and N. Ferreira-Schut, *A Wireless Actuator-Sensor Neural Network for Evacuation Routing.* Sensor Technologies and Applications, International Conference on, 2009. **0**: p. 139-144.
2. Akyildiz, I.F. and I.H. Kasimoglu, *Wireless sensor and actor networks: research challenges.* Ad Hoc Networks, 2004. **2**(4): p. 351 - 367.
3. *Wireless Sensor and Actor Networks (WSAN)*, 2011.
4. Younis, M. and K. Akkaya, *Strategies and techniques for node placement in wireless sensor networks: A survey.* Ad Hoc Networks, 2008. **6**(4): p. 621-655.
5. Guo, W. and M. Zhou. *An emerging technology for improved building automation control*. in *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*. 2009.
6. TAC, *Wireless Controller Networks for Building Automation. Benefits and opportunities for facility owners*, 2010, http://www.tac.com/data/internal/data/07/39/1220381406396/TAC$+$Wireless$+$WP_A4.pdf.
7. Giannopoulos, N., C. Goumopoulos, and A. Kameas. *Design Guidelines for Building a Wireless Sensor Network for Environmental Monitoring*. 2009. Informatics, Panhellenic Conference: IEEE Computer Society.
8. Nodarse Mora, I.A., *Utilización de WSANs en Sistemas de Control de Edificios*, 2010, Universidad de las Ciencias Informáticas.
9. Alliance, Z., *ZigBee specification*, 2006.
10. Mora, I.A.N. and M.D. Rodríguez, *Redes inalámbricas de sensores y actuadores en sistemas de control de edificios.* Serie Científica, 2011. **4**(4).
11. Guinard, A., A. McGibney, and D. Pesch. *A wireless sensor network design tool to support building energy management*. in *Proceedings of the First ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*. 2009. New York, NY, USA: ACM.
12. Verdone, R., et al., *Wireless sensor and actuator networks: technologies, analysis and design*2010: Academic Press.
13. Xu, K., H. Hassanein, and G. Takahara. *Relay node deployment strategies in heterogeneous wireless sensor networks: multiple-hop communication case*. in *Sensor and Ad Hoc Communications and Networks, 2005. IEEE SECON 2005. 2005 Second Annual IEEE Communications Society Conference on*. 2005. IEEE.
14. Ruiz-Ibarra, E., L. Villasenor-Gonzalez, and R.A. Santos, *Design Issues and Considerations for Coordination Mechanisms in*

Wireless Sensor and Actuator Networks. Electronics, Robotics and Automotive Mechanics Conference, 2007. **0**: p. 80-88.

15. Huang, Y.-K., et al. *An Integrated Deployment Tool for ZigBee-Based Wireless Sensor Networks*. in *Embedded and Ubiquitous Computing, 2008. EUC '08. IEEE/IFIP International Conference on*. 2008.
16. Mc Gibney, A., M. Klepal, and D. Pesch. *A wireless local area network modeling tool for scalable indoor access point placement optimization*. in *Proceedings of the 2010 Spring Simulation Multiconference*. 2010.
17. Mc Gibney, A., M. Klepal, and J.T. ODonnell. *Design of underlying network infrastructure of smart buildings*. in *Intelligent Environments, 2008 IET 4th International Conference on*. 2008.
18. Martínez, D., et al., *Formal Specification and Design Techniques for Wireless Sensor and Actuator Networks.* Sensors, 2011. **11**(1): p. 1059-1077.
19. Liu, H., A. Nayak, and I. Stojmenovic, *Applications, Models, Problems, and Solution Strategies.* Wireless Sensor and Actuator Networks, 2010: p. 1.
20. Ishizuka, M. and M. Aida. *Performance study of node placement in sensor networks*. in *Distributed Computing Systems Workshops, 2004. Proceedings. 24th International Conference on*. 2004.
21. Xu, K., et al. *Relay node deployment strategies in heterogeneous wireless sensor networks: single-hop communication case*. in *Global Telecommunications Conference, 2005. GLOBECOM'05. IEEE*. 2005.
22. Li, X., et al., *Sensor placement in sensor and actuator networks.* Wireless Sensor and Actuator Networks: Algorithms and Protocols for Scalable Coordination and Data Communication, 2010: p. 263.
23. Jourdan, D.B. and O.L. de Weck. *Multi-objective genetic algorithm for the automated planning of a wireless sensor network to monitor a critical facility*. in *Defense and Security*. 2004.
24. Martín, B. and A. Sanz, *Redes neuronales y sistemas borrosos*, 2006, Alfaomega.
25. Matich, D.J., *Redes Neuronales: Conceptos basicos y aplicaciones.* Cátedra de Informática Aplicada a la Ingeniería de Procesos--Orientación, 2001.
26. Flórez Revuelta, F., *Modelo de representación y procesamiento de movimiento para diseño de arquitecturas de tiempo real especializadas*, 2002, Universidad de Alicante. Departamento de Tecnología Informática y Computación.

27. Mora Gimeno, F.J., *Modelo para la integración de técnicas heterogéneas de detección de intrusos en sistemas distribuidos*2010: Universidad de Alicante.
28. Kuri, A., *Redes de Kohonen y la Determinacion Genetica de las Clases.* Instituto Tecnológico Autónomo de México, 2001.
29. Muñoz, D.J., *Proceso de reconocimiento de objetos, asistido por computador (visión artificial), aplicando gases neuronales y técnicas de minería de datos.* Scientia Et Technica, 2006. **12**(30): p. 385-390.
30. Cheng, G., et al., *Rapid Training for Self-Organizing Neural Networks with Incremental.* Intelligent Systems Design and Applications, International Conference on, 2006. **1**: p. 28-31.
31. Fritzke, B., *Growing Cell Structures - A Self-organizing Network for Unsupervised and Supervised Learning.* Neural Networks, 1993. **7**: p. 1441-1460.
32. MARTIN, B. and A. Sanz Molina, *Redes neuronales y sistemas difusos*2002.
33. Martinetz, T., K. Schulten, and others, *A" neural-gas" network learns topologies*1991: University of Illinois at Urbana-Champaign.
34. Fritzke, B. *A Growing Neural Gas Network Learns Topologies*. in *Advances in Neural Information Processing Systems 7*. 1995. MIT Press.
35. Ouyang, Y. and H. Yin, *A neural gas mixture autoregressive network for modelling and forecasting FX time series.* Neurocomputing, 2014. **135**: p. 171-179.
36. Coman, H., E. Barth, and T. Martinetz, *Sparse Coding Neural Gas Applied to Image Recognition*, in *Advances in Self-Organizing Maps*2013, Springer. p. 105-114.
37. Holmström, J., *Growing neural gas.* Experiments with GNG, GNG with Utility and Supervised GNG, Uppsala Master's Thesis in Computer Science, Uppsala University Department of Information Technology, 2002.
38. Shamwell, J., et al. *The robot baby and massive metacognition: Early steps via growing neural gas*. in *Development and Learning and Epigenetic Robotics (ICDL), 2012 IEEE International Conference on*. 2012. IEEE.
39. Cui, Y., Y. Zhang, and Y. Cui. *A growing neural gas (GNG) model for hand gesture recognition*. in *Natural Computation (ICNC), 2013 Ninth International Conference on*. 2013. IEEE.
40. Mueller, C.A., N. Hochgeschwender, and P.G. Ploeger. *Surface Reconstruction with Growing Neural Gas*. in *Proc. of the Workshop on Active Semantic Perception and Object Search in the Real World held at the Conference on Intelligent Robots and Systems (IROS), San Francisco, USA*. 2011.

41. Martinetz, T. and K. Schulten, *Topology representing networks.* Neural Networks, 1994. **7**(3): p. 507 - 522.
42. Thiel, M. and K. Sarabandi, *3D-wave propagation analysis of indoor wireless channels utilizing hybrid methods.* Antennas and Propagation, IEEE Transactions on, 2009. **57**(5): p. 1539-1546.
43. Goodman, J.E. and J. O'Rourke, *Handbook of discrete and computational geometry*2010: CRC press.
44. Graham, R.L., *An efficient algorith for determining the convex hull of a finite planar set.* Information processing letters, 1972. **1**(4): p. 132-133.
45. Mücke, E., *Quickhull: computing convex hulls quickly.* Computing in Science & Engineering, 2009. **11**(5): p. 54-57.
46. Sharif, M., *A new approach to compute convex hull.* Innovative Systems Design and Engineering, 2011. **2**(3): p. 186-192.
47. Yang, S., et al., *A point-in-polygon method based on a quasi-closest point.* Computers & Geosciences, 2010. **36**(2): p. 205-213.
48. Li, J. and W. Wang. *Point-in-polygon tests by determining grid center points in advance*. in *Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2013 Asia-Pacific*. 2013. IEEE.
49. Žalik, B. and I. Kolingerova, *A cell-based point-in-polygon algorithm suitable for large sets of points.* Computers & Geosciences, 2001. **27**(10): p. 1135-1145.
50. Cleary, J.G. and G. Wyvill, *Analysis of an algorithm for fast ray tracing using uniform space subdivision.* The Visual Computer, 1988. **4**(2): p. 65-83.
51. Norrenbrock, C., *Percolation threshold on planar Euclidean Gabriel Graphs.* arXiv preprint arXiv:1406.0663, 2014.
52. Toussaint, G.T., *The relative neighbourhood graph of a finite planar set.* Pattern recognition, 1980. **12**(4): p. 261-268.
53. Lingas, A., *A linear-time construction of the relative neighborhood graph from the Delaunay triangulation.* Computational Geometry, 1994. **4**(4): p. 199-208.
54. March, W.B., P. Ram, and A.G. Gray. *Fast euclidean minimum spanning tree: algorithm, analysis, and applications*. in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2010. ACM.
55. Dong, W., C. Moses, and K. Li. *Efficient k-nearest neighbor graph construction for generic similarity measures*. in *Proceedings of the 20th international conference on World wide web*. 2011. ACM.
56. Tamassia, R., *Handbook of graph drawing and visualization.* AMC, 2011. **10**: p. 12.
57. Brandes, U., et al., *Graph markup language (GraphML)*2010: Bibliothek der Universität Konstanz.

58. McGill, J.R., E.A. Walkup, and M.K. Kuhner, *GraphML specializations to codify ancestral recombinant graphs.* Frontiers in genetics, 2013. **4**.
59. Shields, T.L., *Generating GraphML XML Files for Graph Visualization of Architectures and Event Traces for the Monterey Phoenix Program*, 2012, DTIC Document.
60. Lim, E.H., J.N. Liu, and R.S. Lee, *Ontology Graph Generation Process*, in *Knowledge Seeker-Ontology Modelling for Information Search and Management*2011, Springer. p. 99-119.
61. O'Madadhain, J., et al., *The jung (java universal network/graph) framework.* University of California, Irvine, California, 2003.
62. Sedgewick, R. and P. Flajolet, *An introduction to the analysis of algorithms*2013: Addison-Wesley.
63. Sampier, R.H., *Metodología de la investigación*2006: Editorial Ciencias Médicas.
64. Bauer, H.-U. and K.R. Pawelzik, *Quantifying the neighborhood preservation of self-organizing feature maps.* Neural Networks, IEEE Transactions on, 1992. **3**(4): p. 570-579.
65. Uriarte, E.A. and F.D. Martín, *Topology preservation in SOM.* International Journal of Mathematical and Computer Sciences, 2005. **1**(1): p. 19-22.
66. Revuelta, F.F., et al., *Analysis of the topology preservation of accelerated Growing Neural Gas in the representation of bidimensional objects*, in *Current Topics in Artificial Intelligence*2004, Springer. p. 167-176.
67. Villmann, T., et al., *Topology preservation in self-organizing feature maps: exact definition and measurement.* Neural Networks, IEEE Transactions on, 1997. **8**(2): p. 256-266.
68. Yap, B. and C. Sim, *Comparisons of various types of normality tests.* Journal of Statistical Computation and Simulation, 2011. **81**(12): p. 2141-2155.
69. Kaski, S. and K. Lagus, *Comparing self-organizing maps*, in *Artificial Neural Networks—ICANN 96*1996, Springer. p. 809-814.
70. Nadarajah, S., *An Expression for the Average Quantization Error.* Wireless personal communications, 2009. **49**(4): p. 575-585.
71. Jegou, H., M. Douze, and C. Schmid, *Product quantization for nearest neighbor search.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2011. **33**(1): p. 117-128.

ANEXO 1 RESULTADOS DE LAS PRUEBAS REALIZADAS

Tabla 27 Resultados de las pruebas realizadas.

No	Cantidad Dispositivos	Cantidad de Nodos	Tiempo	Promedio del Cuantificador del Error	Producto Topográfico	Componentes conexas
1	5	50	1449	0,014693043	-0,365344249	1
2	5	50	255	0,014298952	-0,260565636	1
3	5	50	262	0,014036379	-0,352956471	1
4	5	50	321	0,014639418	-0,476486841	1
5	5	50	200	0,013539844	-0,573666448	1
6	5	50	211	0,014092082	-0,318777601	1
7	5	50	196	0,014988955	-0,262550984	1
8	5	50	216	0,013717942	-0,317054174	1
9	5	50	215	0,013922042	-0,072029252	1
10	5	50	205	0,014002637	-0,268743312	1
11	10	50	192	0,01415003	-0,400206455	1
12	10	50	196	0,013847774	-0,257514162	1
13	10	50	192	0,01315688	-0,162902501	1
14	10	50	189	0,013299873	-0,462887998	1
15	10	50	197	0,013847473	-0,234307822	1
16	10	50	217	0,012919224	-0,451961344	1
17	10	50	214	0,014967855	-0,186110498	1
18	10	50	191	0,012821965	-0,348237573	1
19	10	50	193	0,014652213	-0,212363444	1
20	10	50	193	0,014111176	-0,319540989	1
21	15	50	186	0,012928492	-0,184177193	1
22	15	50	183	0,014093712	-0,426262762	1
23	15	50	184	0,012894276	-0,382047257	1
24	15	50	185	0,012916088	-0,235161169	1
25	15	50	191	0,013943319	-0,376222978	1
26	15	50	178	0,013283939	-0,284149452	1
27	15	50	189	0,012558035	-0,351970667	1
28	15	50	186	0,012912441	-0,214112818	1
29	15	50	176	0,013214522	-0,191956385	1
30	15	50	179	0,01373506	-0,296835721	1
31	20	50	195	0,012740274	-0,463118564	1
32	20	50	167	0,013774269	-0,321523658	1
33	20	50	172	0,012480888	-0,464883883	1
34	20	50	171	0,013617312	-0,304350532	1
35	20	50	168	0,013417369	-0,124310399	1
36	20	50	167	0,014253407	-0,39634392	1
37	20	50	166	0,012862697	-0,464250753	1
38	20	50	165	0,013551799	-0,32216891	1

39	20	50	162	0,013168015	-0,281522026	1
40	20	50	166	0,012714296	-0,468761556	1
41	5	100	802	0,009921185	-0,612424169	1
42	5	100	786	0,0099014	-0,518810515	1
43	5	100	789	0,010096949	-0,472673166	1
44	5	100	812	0,010187459	-0,58338536	1
45	5	100	782	0,010416158	-0,34159071	1
46	5	100	784	0,010664051	-0,541607245	1
47	5	100	778	0,010412647	-0,522666363	1
48	5	100	791	0,010275613	-0,554403979	1
49	5	100	798	0,010387911	-0,466583273	1
50	5	100	773	0,009820762	-0,673799486	1
51	10	100	778	0,009807231	-0,432174437	1
52	10	100	790	0,009905325	-0,717579916	1
53	10	100	776	0,010025726	-0,484344074	1
54	10	100	788	0,010034326	-0,515238984	1
55	10	100	787	0,009813534	-0,59982456	1
56	10	100	783	0,00992699	-0,49254819	1
57	10	100	781	0,009791815	-0,480774671	1
58	10	100	778	0,010492864	-0,663888903	1
59	10	100	798	0,009745902	-0,39912406	1
60	10	100	802	0,009634012	-0,40911585	1
61	15	100	759	0,009725781	-0,4642225	1
62	15	100	770	0,009332221	-0,59337304	1
63	15	100	787	0,010196031	-0,392752287	1
64	15	100	770	0,009436444	-0,451979493	1
65	15	100	755	0,0100084	-0,490386849	1
66	15	100	750	0,010179678	-0,438072185	1
67	15	100	787	0,009257143	-0,182780331	1
68	15	100	780	0,009840317	-0,539280891	1
69	15	100	782	0,010071739	-0,414686933	1
70	15	100	818	0,009452126	-0,624011245	1
71	20	100	839	0,009270065	-0,487855311	1
72	20	100	823	0,00959161	-0,425472917	1
73	20	100	777	0,009196859	-0,319977453	1
74	20	100	786	0,009646529	-0,44238133	1
75	20	100	768	0,009555451	-0,391758446	1
76	20	100	761	0,009406643	-0,438243681	1
77	20	100	803	0,009466915	-0,352558337	1
78	20	100	779	0,009431214	-0,505567432	1
79	20	100	785	0,009233623	-0,616804426	1
80	20	100	815	0,00916814	-0,506868188	1
81	5	150	1833	0,008503863	-0,661200853	1
82	5	150	1871	0,008091578	-0,629545829	1
83	5	150	1779	0,008544193	-0,547194523	1

84	5	150	1824	0,008225611	-0,6092825	1
85	5	150	1962	0,008350527	-0,330797833	1
86	5	150	1852	0,008329438	-0,402096681	1
87	5	150	1832	0,008441216	-0,471443336	1
88	5	150	1760	0,00844802	-0,530058201	1
89	5	150	1869	0,008316488	-0,330865098	1
90	5	150	1823	0,008649144	-0,49252429	1
91	10	150	1748	0,008450852	-0,626140788	1
92	10	150	1779	0,008236312	-0,576037392	1
93	10	150	1772	0,007911828	-0,44790272	1
94	10	150	1779	0,008325341	-0,482569123	1
95	10	150	1769	0,008005872	-0,460257381	1
96	10	150	1817	0,00799293	-0,586301422	1
97	10	150	1788	0,008194773	-0,481290254	1
98	10	150	1785	0,008324774	-0,591066517	1
99	10	150	1778	0,008413554	-0,625029638	1
100	10	150	1749	0,008159134	-0,42307268	1
101	15	150	1763	0,008241472	-0,274792519	1
102	15	150	1805	0,007829365	-0,626464883	1
103	15	150	1755	0,007688482	-0,40861153	1
104	15	150	1735	0,008451263	-0,405754431	1
105	15	150	1761	0,007958228	-0,706806021	1
106	15	150	1794	0,007740347	-0,645945312	1
107	15	150	1762	0,007960194	-0,30717994	1
108	15	150	1747	0,007953871	-0,483797824	1
109	15	150	1758	0,007665273	-0,560738891	1
110	15	150	1911	0,008037242	-0,522136917	1
111	20	150	1767	0,00795649	-0,470802949	1
112	20	150	1751	0,007601411	-0,601240391	1
113	20	150	1708	0,008083183	-0,526642526	1
114	20	150	1951	0,007977404	-0,403700728	1
115	20	150	2019	0,008133756	-0,492828919	1
116	20	150	1776	0,007823399	-0,361667754	1
117	20	150	1788	0,007867197	-0,342868797	1
118	20	150	1789	0,008069479	-0,509079393	1
119	20	150	2071	0,00783993	-0,462646302	1
120	20	150	1772	0,007727083	-0,468310926	1