Revista Educación en Ingeniería
ISSN 1900-8260
Enero a Junio de 2012, Vol. 7, N°. 13, pp. 23-31 • © 2012 ACOFI • http://www.educacioneningenieria.org
Recibido: 15/10/2011 • Aprobado: 17/05/2012 • Versión final: 25/05/2012

BIO-ROUTE: UN SIMULADOR PARA REDES DE SENSORES INALÁMBRICOS

BIO-ROUTE: A SIMULATOR FOR WIRELESS SENSOR NETWORKS

Juan Carlos Blandón Andrade
Pontificia Universidad Javeriana, Cali (Colombia) • jcblandon@javerianacali.edu.co

Jesús Alfonso López Sotélo
Universidad Autónoma de Occidente, Cali (Colombia) • jalopez@uao.edu.co

Resumen

En este trabajo se presenta un simulador de redes de sensores inalámbricos desarrollado como parte de una investigación sobre la aplicación de algoritmos bioinspirados en el problema de enrutamiento en dichas redes. El simulador diseñado permite, entre otras acciones, crear una red de sensores inalámbricos completamente personalizada, modificar dicha red, generar eventos, resolver el enrutamiento entre sensores, transmitir datos y guardar información en archivos planos para su posterior utilización. El simulador también se puede usar en procesos de enseñanza-aprendizaje como herramienta didáctica por docentes y estudiantes interesados en esta tecnología.

Palabras clave: redes de sensores inalámbricos, enrutamiento, software para pedagogía

Abstract

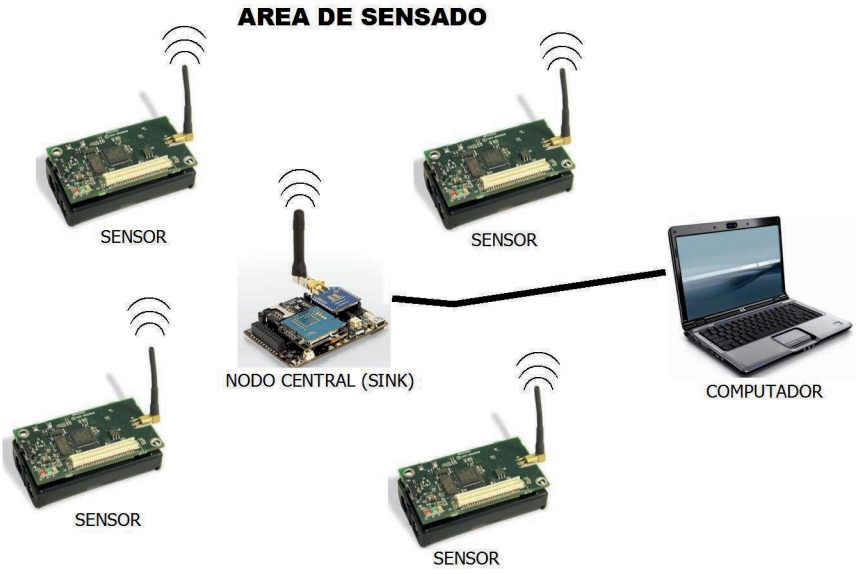
This paper presents a simulator of wireless sensor networks developed during a research on the application of bioinspired algorithms to the routing problem of this kind of networks. The proposed simulator can create a custom wireless sensor network, modify it, add new sensors, generate events, routing and store data in text files to reload it. The simulator can be used as an educational tool for teaching to people interested in this technology.

Keywords: wireless sensor networks, routing, education software

Introducción

La tecnología de redes de sensores inalámbricos consiste básicamente en una serie de dispositivos electrónicos denominados nodos sensores (en general, son dispositivos con capacidades de cómputo, transmisión inalámbrica y sensado, básicos), desplegados en un área determinada (Beltran, 2007). Estos sensores, una vez ubicados dentro del área a inspeccionar, informan acerca de cambios que ocurren en la misma con el objetivo de transmitir estos eventos a un nodo central llamado colector o *sink*, el cual también es un sensor, pero tiene comunicación con un computador personal o portátil en el cual están centralizados los datos de la red (ver figura 1).

Figura 1. Red de sensores inalámbricos.



Fuente: (Mohammad & Imad, 2005).

Es importante mencionar que estos dispositivos son pequeños, cuentan con poca capacidad de cómputo y tienen una fuente de energía que se agota. Sin embargo, esto no es una limitante a la hora de realizar las labores para los cuales fueron diseñados. Entre sus aplicaciones se encuentran el monitoreo de variables tales como: temperatura, humedad relativa y sonidos, entre otras variables en ambiente de difícil acceso (Akyildiz et al., 2002).

Las redes de sensores inalámbricos también se caracterizan por la ubicación de cientos o miles de dispositivos en un área geográfica; éstos a su vez tienen un alcance limitado y cuando quieren informar un evento al nodo central, la información debe saltar a varios sensores antes de llegar a su destino. Es decir, se deben trazar unas posibles rutas, para cuando sucede un determinado evento en una zona específica, esto también es llamado el problema de enrutamiento en las redes de sensores inalámbricos.

Dado que varias facultades y escuelas, en particular de ingeniería, investigan y proponen nuevos algoritmos para tratar de resolver el problema del enrutamiento, se hace necesaria la creación de una herramienta computacional que permita probar estos algoritmos desarrollados y, de esta forma, poder medir su eficiencia.

En un trabajo sobre un simulador específico (González et al., 2009), donde se diseñó un simulador para red de sensores cuyos nodos estuvieran basados en microprocesadores Texas Instruments® de la serie MSP430 (debido a que no hay simuladores que modelen el comportamiento de cada nodo), sus diseñadores trabajaron en la adaptación del simulador MSPSIM (Eriksson et al., 2008) que es un simulador para los microprocesadores mencionados y desarrollaron la modificación de una herramienta de simulación de redes de sensores inalámbricos en ambiente Java denominada COOJA.

Otro trabajo que presenta una visión global sobre las redes de sensores inalámbricos (Chio Cho et al., 2009) menciona algunas herramientas como Castalia desarrollada por *Networks and Pervasive Computing de National,* ICT Australia; J-SIM es útil tanto para simular como emular una red de sensores inalámbricos. También se encuentra GloMoSim que es un entorno de simulación escalable escrito en lenguaje C y Parsec, el cual puede simular en forma paralela el evento de simulación discreta. El *software* OMNeT++ que ayuda a estudiar el efecto de escala, la arquitectura a nivel de nodo, la eficiencia energética, la arquitectura de las comunicaciones y el sistema.

El proyecto realizado en el lenguaje de programación Python® (García et al., 2009) hace referencia a la construcción de un entorno de simulación aplicado en la extinción de incendios forestales. El simulador está compuesto por varios núcleos independientes y a su vez interconectados. Los sensores se distribuyen en una zona geográfica para medir periódicamente la temperatura, presión, humedad y la velocidad del viento. Según la propuesta, los bomberos portan dispositivos móviles que reciben el resultado de los algoritmos.

Estas visiones demuestran que para cada problema en particular es conveniente desarrollar la herramienta que permita la simulación de la red de acuerdo con sus propósitos; una investigación realizada sobre algoritmos bioinspirados, en redes de sensores inalámbricos en la Pontificia Universidad Javeriana, Cali (Colombia) requería de un sistema de prueba para los algoritmos diseñados. Para esto, se desarrolló la herramienta llamada BIO-ROUTE versión 1.0, que permitió realizar las pruebas y evaluar los algoritmos desarrollados. A su vez, esta herramienta permite servir como aporte para futuras investigaciones sobre enrutamiento en redes de sensores inalámbricos, pues con el simulador desarrollado se cuenta con un *software* que se puede tomar como punto de partida para probar los nuevos modelos.

Metodología

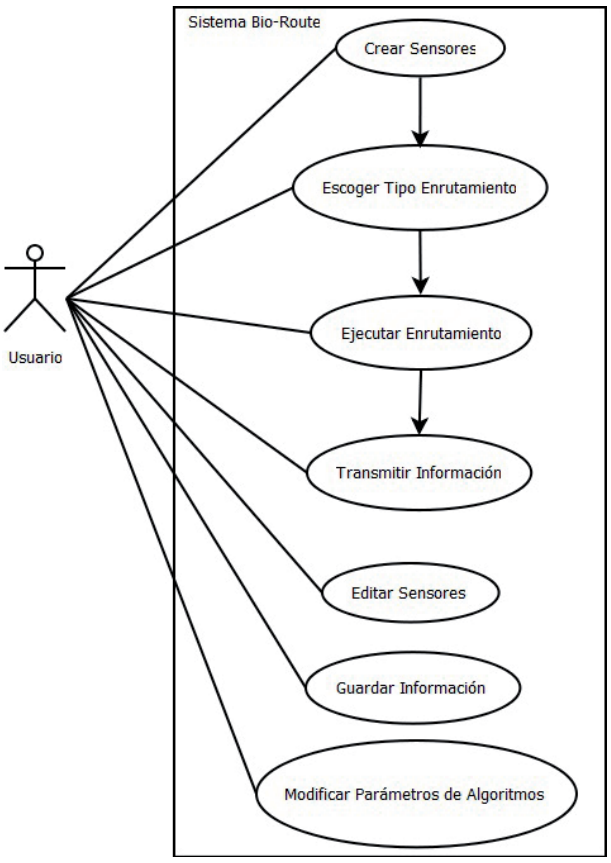
Para la realización del simulador se utilizó el método del ciclo de vida clásico de *software*. Este ciclo comprende: análisis, diseño, implementación y pruebas del sistema.

Análisis: en el análisis se definieron las características que debe tener el *software* a desarrollar, entre ellas:

- El sistema debe permitir la creación de un espacio donde se puedan ubicar los sensores y dispositivos en cualquier parte del área y la cantidad que el usuario desee. Se debe crear una opción para que el usuario pueda cargar información sobre una red que se haya guardado con anterioridad. Después de ubicar los sensores en el plano bidimensional, el *software* debe calcular las distancias euclidianas entre los diferentes sensores y almacenarla en memoria, para ello, debe utilizar la ubicación XY, y de esta forma, realizar el proceso.
- Se debe crear una opción para modificar la información de los sensores, por ejemplo, la ubicación de los sensores en el plano, la energía, la cobertura, el estado y mostrar la información sobre las variables que monitorea.
- Se debe contar con la opción de poder agregar un nuevo sensor a la red existente.
- Se debe tener la posibilidad de crear eventos en la red, es decir, que al ejecutar una rutina, se debe generar aleatoriamente un evento. Esto a su vez producirá la necesidad de enrutamiento entre el sensor donde se generó el evento y el nodo colector.
- La aplicación debe permitir escoger el tipo de enrutamiento deseado y aplicarlo para encontrar la ruta entre el nodo generador del evento y el nodo colector.
- Debe mostrar el resultado del proceso de enrutamiento y dibujar el recorrido en el plano, al igual que informar a qué sensores debe saltar antes de llegar a su destino.
- Con una ruta definida, el simulador debe tener la capacidad de transmitir información, desde el sensor inicial hasta el nodo final. Para ello, se debe simular el envío de información a cada uno de los sensores de la ruta encontrada y descontar la energía consumida en cada sensor.
- Se debe permitir guardar en archivos, información sobre los sensores, las variables de monitoreo y enrutamientos realizados por el sistema.
- Finalmente, la aplicación debe dejar modificar los parámetros de los algoritmos de enrutamiento que se utilicen para hallar nuevas rutas.

Diseño: para la creación de este *software* se diseñó un diagrama de casos de uso que se muestra en la figura 2.

Figura 2. Diagrama de casos de uso del sistema Bio-Route.



Fuente: Elaboración propia

El diagrama de casos permitió detectar las principales acciones entre el usuario y el sistema con lo cual se tuvo una visión clara de lo que se esperaba de la aplicación.

Implementación: Los lenguajes de programación permiten al desarrollador de *software* compilar sus algoritmos para que una computadora pueda ejecutarlos. Existen varias herramientas en el mercado las cuales pueden utilizarse para desarrollar soluciones informáticas. A continuación se mencionan, brevemente, algunas de ellas con sus respectivas características.

Java®: Es un lenguaje de programación orientado a objetos que permite realizar aplicaciones para Internet y dispositivos móviles. La tecnología Java se basa en dos elementos: el lenguaje Java y la plataforma, que es la máquina virtual de Java (JVM). El código Java tiene la capacidad de funcionar en cualquier plataforma *software* o *hardware;* Por ejemplo, no importa se ejecuta el código desde el sistema operativo Linux o Windows®, el programa Java se ejecutará correctamente en ambos, sólo basta tener instalada la máquina virtual (García de Jalón, 2000). Algunas de las características del Lenguaje Java son:

- Lenguaje interpretado
- Fuertemente tipado.
- Sintaxis similar a C / C++.
- Sin punteros: garbage collection
- 100% portable
- Integra bibliotecas estándar para:
  - Interfaces de usuario
  - Objetos distribuidos
  - Threads
  - XML
- Ejecutable desde navegadores web (Mozilla, IExplorer, etc)
- Origen: aumentar html para páginas web más dinámicas
- Una limitante es que al ser un lenguaje interpretado no es muy rápido

Netlogo®: Es un ambiente programable para simular sistemas naturales y sociales, principalmente de tipo complejo que evolucionan con el tiempo. Contiene una biblioteca de modelos "Models Library" que es una colección grande de simulaciones que pueden usarse y modificarse. Estas simulaciones se dirigen a muchas áreas tales como: ciencias naturales y sociales; biología y medicina; física y química; matemática e informática; economía y psicología social (Vidal, 2011). Algunas de las características de Netlogo® son las siguientes:

- Completamente programable
- Estructura de lenguaje simple
- En la parte matemática, doble precisión en punto flotante
- Es intuitivo de usar
- Es multiplataforma

PHP® (Hypertext Preprocessor): Es un lenguaje de programación interpretado al lado del servidor que es utilizado para generar páginas Web dinámicas y su código se puede incluir en páginas html. Como es un lenguaje que se ejecuta en el servidor, no necesita que lo soporte el navegador; pero, es necesario que el servidor donde se encuentra alojada la página soporte PHP (Martín & Graciani, 2008). Algunas de las características del lenguaje PHP® son:

- Su rapidez
- Soporte Multiplataforma
- Facilidad de aprendizaje
- Permite trabajar con muchas Bases de Datos. (Oracle, Informix, MySql, PostgreSQL)
- Es gratuito bajo una licencia abierta

MATLAB® (MATrix LABoratory): Es un lenguaje para realizar cálculos numéricos con vectores y matrices. Como caso particular, puede también trabajar con números escalares tanto reales como complejos, con cadenas de caracteres y con otras estructuras de información más complejas. Una de las capacidades más atractivas es la de realizar una amplia variedad de gráficos en dos y tres dimensiones. MATLAB® tiene también un lenguaje de programación propio. También se puede decir que es un programa de cálculo técnico y científico, cuenta con la vectorización de algoritmos, lo cual le da velocidad de procesamiento. En otras aplicaciones resulta bastante más lento que el código equivalente desarrollado en lenguajes como C/C++ o Fortran (García & Rodriguez, 2005). A continuación se mencionan algunas de sus características:

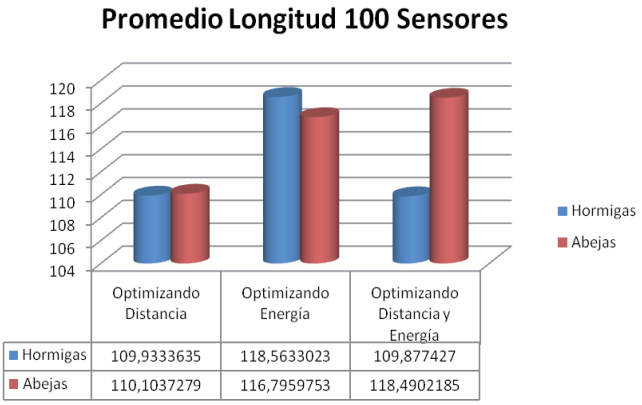
- Cálculo numérico rápido y con alta precisión
- Manejo simbólico
- Graficación y visualización avanzada
- Programación mediante un lenguaje de alto nivel
- Programación estructurada y orientada a objetos
- Soporte básico para diseño de interfaz gráfica
- Extensa biblioteca de funciones
- Integración con aplicaciones externas
- Aplicaciones especializadas para algunas ramas de ciencias e ingeniería (toolboxes)

Bio-Route fue desarrollado en el lenguaje de programación MATLAB®. La implementación se realizó por módulos, utilizando la interfaz gráfica de MATLAB®, también utilizando las bibliotecas. En muchas partes de los algoritmos de enrutamiento se utilizó la vectorización, esto le dio más rapidez a la entrega de resultados de los procesos. Un aspecto a resaltar es que el *software* al ser desarrollado en una plataforma como MATLAB® (que es una de las plataformas mas extendidas para realizar simulaciones en ingeniería), permitirá al simulador ser usado por una gran cantidad de instituciones educativas, lo cual hará posible su aplicabilidad como herramienta de enseñanza.

Pruebas: Adicionalmente a la implementación de algunos algoritmos bioinspirados para el enrutamiento de la red de sensores inalámbrico, también se realizó la de un algoritmo de enrutamiento tradicional. Esto con fin de poder realizar comparaciones entre el método tradicional y los propuestos en la investigación desarrollada. Por el diseño modular del simulador, realizar las implementaciones de dichos algoritmos consistió en adicionar un módulo a la aplicación en cada caso.

El simulador permitió verificar de manera rigurosa el desempeño de los diferentes algoritmos implementados; para tal fin se realizaron enrutamiento de redes de hasta 1000 nodos y, debido a la característica del simulador de generar los datos relevantes de las diferentes simulaciones, se pudieron hacer comparaciones muy exhaustivas de diferentes aspectos del enrutamiento tales como: la longitud, el gasto de energía y la eficiencia (medida en la cantidad de sensores activados) de la ruta encontrada. En la figura 3 se muestra la comparación de los algoritmos bioinspirados considerando la longitud lograda por el enrutamiento para una red de 100 sensores. No se entrará en detalle de los algoritmos implementados pues no constituye el objetivo de este trabajo.

Figura 3. Ejemplo de comparación obtenida con los datos generados por el simulador

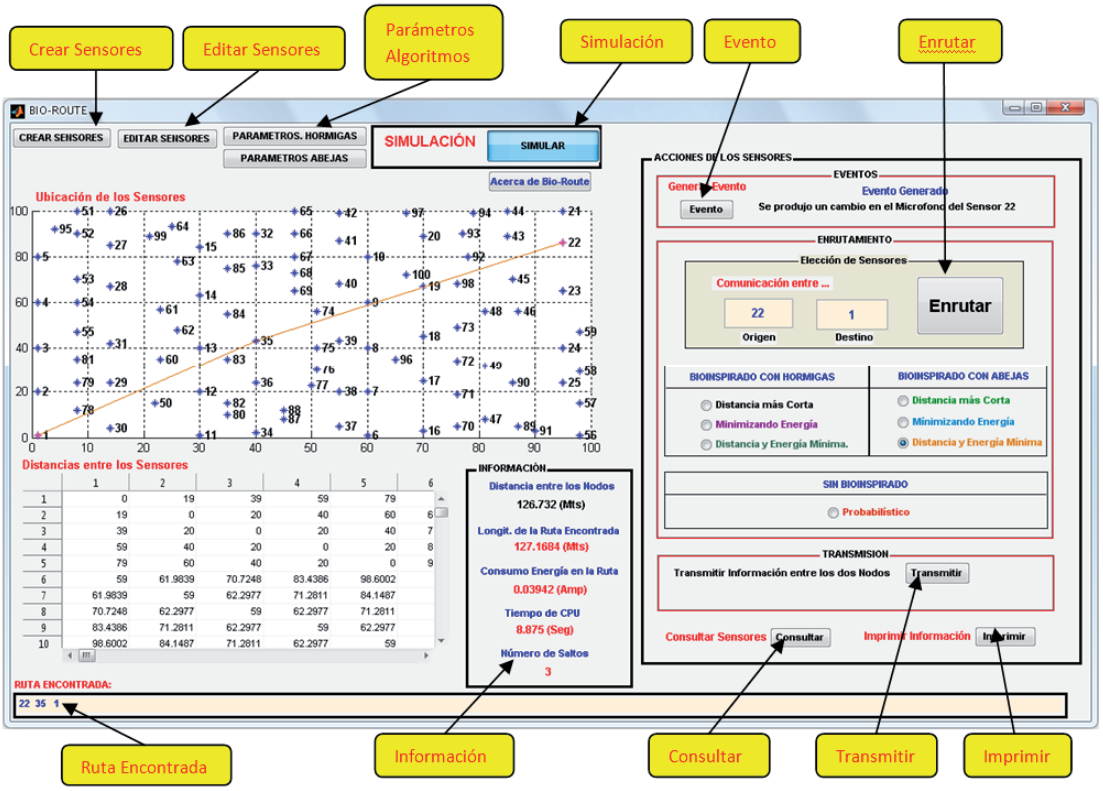


	Optimizando Distancia	Optimizando Energía	Optimizando Distancia y Energía
Hormigas	109,9333635	118,5633023	109,877427
Abejas	110,1037279	116,7959753	118,4902185

Resultados

A continuación se hará una breve descripción de la herramienta desarrollada. La pantalla principal del simulador desarrollado se presenta en la figura 4.

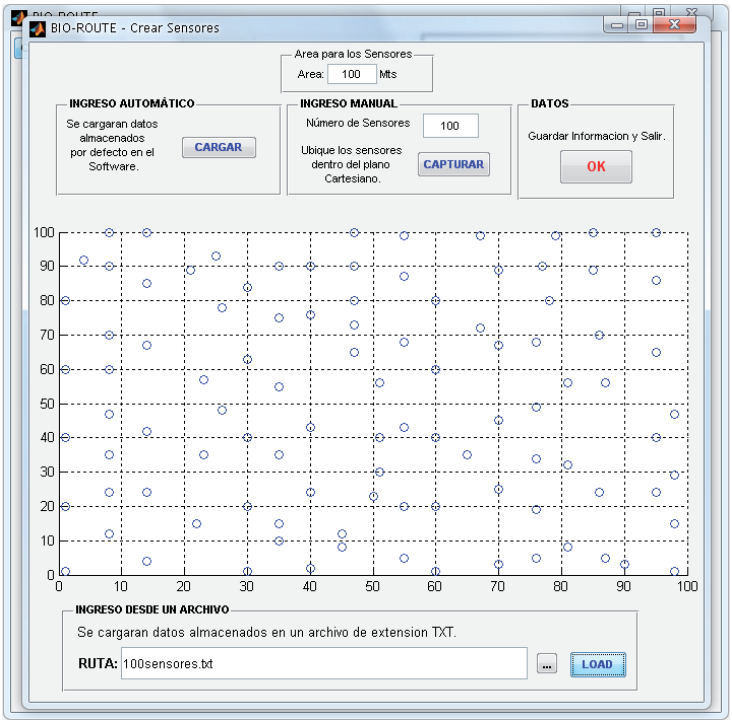
Figura 4. Pantalla principal Bio-Route.



Fuente: elaboración propia.

El botón **"*crear sensores*"** llama a otra ventana en donde se realiza el proceso (ver figura 5).

Figura 5. Pantalla crear sensores.

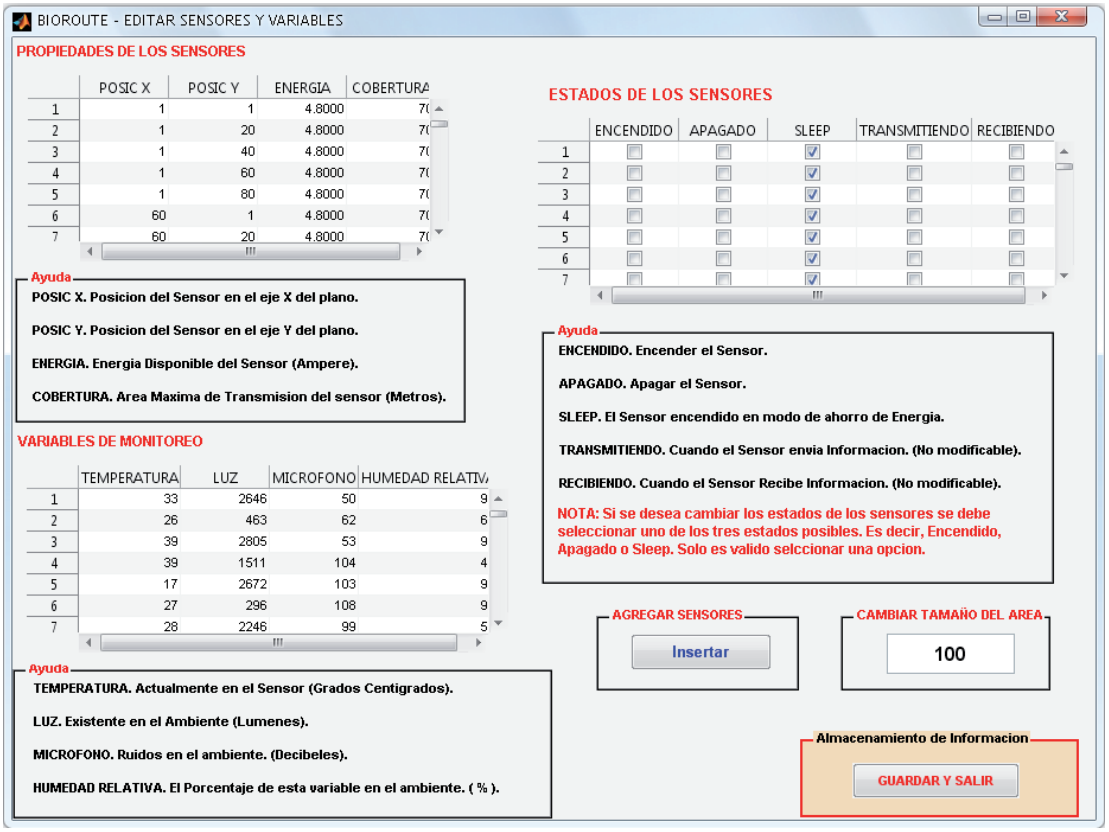


Fuente: Elaboración propia

Existen tres formas de crear sensores en el plano. La primera está en la opción de ingreso automático; allí se carga información predefinida para la aplicación cuando se realizaron las pruebas. La segunda es la forma manual; al escoger esta opción el usuario podrá definir el número de sensores a utilizar y capturar manualmente los puntos que el usuario seleccione dentro del plano. La tercera es cargar la información de los sensores desde un archivo "txt".

Al tener creados los sensores el programa principal habilita los botones de la aplicación, calcula las distancias euclidianas entre los sensores y los almacena en una tabla. El botón ***"editar sensores"*** permite llamar otra ventana donde se podrá modificar la información sobre los sensores o de las variables que monitorean (ver figura 6).

Figura 6. Pantalla editar sensores.



Fuente: Elaboración propia

Allí se puede cambiar la posición del sensor, la cobertura, apagar el sensor, la variables de monitoreo. También es posible cambiar el tamaño del área donde se encuentran ubicados los sensores. Además, tiene una opción para agregar un sensor nuevo a los existentes. Finalmente, cuenta con la opción para guardar los cambios realizados.

Al seleccionar el botón **"simulación",** aparecerán todas las opciones para simular el enrutamiento en una red de sensores inalámbricos. Con el botón "**evento**" se puede generar un nuevo evento en la red; el *software* escogerá un sensor aleatoriamente, también un evento que suceda en ese dispositivo. Al generar el evento, el simulador resalta el sensor generador del evento y el nodo colector. Entre estos dos nodos posteriormete se realizará el enrutamiento.

El usuario puede seleccionar el tipo de algoritmo de enrutamiento a utilizar, después de ello debe pulsar el Botón "**enrutar**" y esperar que se presente el resultado que será dibujado en el plano. El *software*

presenta información sobre la longitud en metros de la solución encontrada, cálculo estimado de consumo de energía en esa ruta (no se desglosa por nodos), el tiempo de cómputo de esa solución y el número de saltos que debe realizar la información desde su origen hasta el destino. También presenta la ruta encontrada detalladamente (los sensores a los que hay que saltar).

Después de realizar el enrutamiento la aplicación permite simular el envío de información por medio del botón "**transmitir**". Al escoger esta opcion, el usuario ve que se van encendiendo y apagando los sensores de la ruta encontrada desde el inico hasta el fin. Cuando se realiza esta acción, de la información existente de esos sensores, se descuenta la energía gastada.

La aplicación cuenta con un botón "**imprimir**", que permite almacenar la información de la simulación actual en archivos de extensión txt. Esto permite al usuario volver a cargar esta información sin tener que adecuarla nuevamente.

Conclusiones

Uno de los principales aportes de este trabajo fue el de implementar un simulador para redes de sensores inalámbricos en una plataforma, tan extendida para la simulación de aplicaciones en ingeniería, como el MATLAB® lo cual contrasta con los simuladores existentes desarrollados en plataformas muy específicas.

Bio-Route permitió implementar de manera eficiente diferentes algoritmos de enrutamiento y, debido a sus características, fue posible realizar una comparación bastante detallada entre los mismos.

La aplicación desarrollada, Bio-Route v. 1.0, se ideó pensando en un diseño modular el cual permitirá implementar nuevos algoritmos de enrutamiento y parametrizar, de mejor manera, la red de sensores inalámbricos para que sus resultados sean los más cercanos a la realidad lo cual convierte al software desarrollado, en una opción a usarse en procesos de enseñanza aprendizaje como herramienta didáctica por docentes y estudiantes interesados en esta tecnología.

Referencias

Akyildiz, I. F., Su, W., Sankarasubramaniam, Y. & Cayirci, E. (2002). Wireless sensor networks: a survey. *Computer Networks Journal*, 38(4), 393-422.

González, Asencio, Núñez, Arturo M., & Pascual G., Javier. (2009). *Diseño de un simulador para redes de sensores*. Madrid, España: Facultad de Informática, Universidad Complutense de Madrid.

Beltrán, F. (2007). Wireless Sensor Networks. *Bit* , 61.

Chio Cho, N., Tibaduiza B., Diego, Aparicio Z., Laura y Caro O., Luis M. (2009). Redes de Sensores Inalámbricos. *II Congreso Internacional de Ingeniería Mecatrónica, Universidad Autónoma de Bucaramanga*, Bucaramanga, Colombia.

Eriksson, J., Dunkels, A., Finne, N., Österlind, F.,Voigt, T. & Tsiftes, N. (2008). Demo abstract: Mspsim - an extensible simulator for msp430-equipped sensor boards. In *Proceedings of the 5th European Conference on Wireless Sensor Networks (EWSN 2008)*, Bologna, Italy.

García de Jalón, J. (2000). *Aprenda Java como si estuviera en primero*. San Sebastian, España: Escuela Superior de Ingenieros Industriales, Universidad de Navarra.

García de Jalón, J., Rodriguez, J. I., Vidal, J. (2005). Aprenda Matlab 7.0 como si estuviera en primero. Madrid, España: Escuela Técnica Superior de Ingenieros Industriales, Universidad Politécnica de Madrid.

García, E. M., Serna, M. Á., Bermúdez, A., y Casado, R. (2009). *Simulación de un sistema de apoyo en la extinción de incendios forestales basado en WSN*. La Mancha, España: Instituto de Investigación en Informática de Albacete, Departamento de Sistemas Informáticos, Universidad de Castilla.

Martín M., F. J., y Graciani D., C. (2008). *PHP: Lenguaje de programación*. Sevilla, España: Departamento de Ciencias de la Computaciónn e Inteligencia Artificial, Universidad de Sevilla.

Mohammad, I., y Imad, M. (2005). Handbook of Sensor Networks:Compact Wireless and Wired Sensing Systems. EUA: *CRC Press*.

Vidal, José. M. (2011). Netlogo For Building Prototype Multiagent Systems. EUA: Departament of Computer Science and Engineering, University of South California.

Sobre los autores

Juan Carlos Blandón Andrade
Pontificia Universidad Javeriana, Cali (Colombia)
jcblandon@javerianacali.edu.co

Jesús Alfonso López Sotelo
Ingeniero Electricista. Universidad del Valle. Magíster en Automática. Universidad del Valle. Doctor en Ingeniería. Universidad del Valle. Actualmente se desempeña como Director del Programa de Ingeniería Mecatrónica de la Universidad Autónoma de Occidente, Cali, Colombia, Sus áreas de interés son la Inteligencia Computacional (Redes Neuronales, Lógica Difusa, Computación Evolutiva) y sus aplicaciones en el control automático, el reconocimiento de patrones, el modelado de sistemas y a la optimización.
jalopez@uao.edu.co