UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA

La Universidad Católica de Loja

Escuela de Ciencias de la Computación

"Implementación de modelos de movilidad para redes móviles Ad Hoc"

Proyecto de fin de carrera previo a la obtención del título de Ingeniero en Informática.

AUTOR: Juan Carlos Sánchez Landin

DIRECTOR: Ing. Rommel Vicente Torres Tandazo

DIRECTORA: Ing. Liliana Elvira Enciso Quispe

Loja –Ecuador

2012

Ing.

Rommel Vicente Torres Tandazo

DIRECTOR DE TESIS

CERTIFICA:

Que el Sr. Juan Carlos Sánchez Landin, autor de la tesis "Implementación de modelos de movilidad para redes móviles Ad Hoc", ha cumplido con los requisitos estipulados en el Reglamento General de la Universidad Técnica Particular de Loja, la misma que ha sido coordinada y revisada durante todo el proceso de desarrollo desde su inicio hasta la culminación, por lo cual autorizo su presentación.

Loja, 16 de Mayo del 2012

……

Ing. Rommel Vicente Torres Tandazo

DIRECTOR DE TESIS

Ing.

Liliana Elvira Enciso Quispe

DIRECTORA DE TESIS

CERTIFICA:

Que el Sr. Juan Carlos Sánchez Landin, autor de la tesis "Implementación de modelos de movilidad para redes móviles Ad Hoc", ha cumplido con los requisitos estipulados en el Reglamento General de la Universidad Técnica Particular de Loja, la misma que ha sido coordinada y revisada durante todo el proceso de desarrollo desde su inicio hasta la culminación, por lo cual autorizo su presentación.

Loja, 16 de Mayo del 2012

…….

Ing. Liliana Elvira Enciso Quispe

DIRECTORA DE TESIS

AUTORÍA

Las ideas, opiniones, conclusiones, recomendaciones y más contenidos expuestos en el presente informe de tesis son de absoluta responsabilidad del autor.

.....

Juan Carlos Sánchez Landin

CESIÓN DE DERECHOS

Yo, **Juan Carlos Sánchez Landin,** declaro ser autor del presente trabajo y eximo expresamente a la Universidad Técnica Particular de Loja y a sus representantes legales de posibles reclamos o acciones legales.

Adicionalmente declaro conocer y aceptar la disposición del Art. 67 del Estatuto Orgánico de la Universidad Técnica Particular de Loja, que en su parte pertinente textualmente dice: "Forman parte del patrimonio de la Universidad la propiedad intelectual de investigaciones, trabajos científicos o técnicos y tesis de grado que se realicen a través o con el apoyo financiero, académico o institucional (operativo) de la Universidad".

…….

Juan Carlos Sánchez Landin.

AGRADECIMIENTO

Agradezco a todos los acompañantes de esta aventura. Primeramente a Dios por su constante apoyo, a mi familia por su desprendimiento y fuerza, a mis amig@s porque hicieron más fácil el camino y finalmente, a aquellos docentes que marcaron la diferencia.

DEDICATORIA

Dedico este trabajo a todo aquél que no se rinde cuando un obstáculo se le presenta.

ÍNDICE DE CONTENIDOS

ARTE I: MARCO DE REFERENCIA

CAPITULO 1

PARTE II: MODELOS PROPUESTOS

CAPITULO 2

CAPITULO 3

PARTE III: IMPLEMENTACIÓN Y VALIDACIÓN

CAPITULO 4

CAPITULO 5

CONCLUSIONES Y RECOMENDACIONES

ANEXOS

ÍNDICE DE FIGURAS

CAPÍTULO 1

PARTE II

CAPITULO 2

CAPITULO 3

PARTE III

CAPITULO 4

CAPITULO 5

ANEXOS

Anexo2

RESUMEN

El presente trabajo inicia con el estudio de algunos de los modelos más representativos de movilidad empleados en la simulación de las redes móviles *Ad hoc* (*MANET*). Se analizan también, conceptos de escenarios de emergencia y rescate reales, que indican las oportunidades que tienen las redes *MANET* para ser aplicadas en zonas de desastre.

En base a lo anterior, se propone la construcción de dos modelos de movilidad de restricción geográfica denominados: Unmarked point model (*UPM*) y Adjacency vertex model (*AVM*), que están orientados a la movilidad dentro de una imagen de formato *XBM* dentro de un segmento del mapa de la ciudad de Loja (1000px X 2000 px) que se ha simulado un desastre. *UPM* está desarrollado a partir del análisis de puntos libres y ocupados de la imagen. *AVM* realiza su movimiento mediante el uso de una matriz de adyacencia y la evaluación comparativa de la longitud de los caminos. Los dos modelos se implementan en la herramienta de simulación *Scengen* para generar escenarios de movilidad que posteriormente van a ser utilizados para el análisis de la red.

ORGANIZACIÓN DE LA TESIS

El presente trabajo de tesis se encuentra dividido en tres partes. En la primera parte, conformada por el Capítulo 1, se hace referencia y un ligero análisis de las redes *MANET* y su importancia en la comunicación moderna. Además de los elementos que componen los escenarios de movilidad y su clasificación respectiva.

En la segunda parte, se planifican y desarrollan dos modelos de movilidad. Además del análisis y selección de la herramienta a implementarse. De tal forma que, en el Capítulo 2, se presenta la planificación y análisis de un modelo denominado *UPM* que está basado en el movimiento dentro de los cuatro puntos comunes de direccionamiento (arriba, abajo, derecha, izquierda) para la evaluación de la disponibilidad para realizar el siguiente movimiento. Se define también, el modelo *AVM*, el cual realiza sus movimientos a través de una matriz de adyacencia, mediante la cual selecciona el camino más corto hacia un destino. En el Capítulo 3, se realiza el desarrollo y la implementación de los modelos. Así mismo, se detallan las características del funcionamiento y estructura de la herramienta Scengen.

En la tercera parte, se implementan los modelos, se genera el escenario de prueba y se evalúan los patrones de movilidad generados por *UPM* y *AVM* midiendo tanto la eficiencia computacional y la eficiencia del protocolo de enrutamiento *AODV.* En el Capítulo 4, se muestra el proceso para la obtención, edición, formateo, generación de obstáculos, la transformación a código binario de la imagen del mapa; y, la obtención de los vértices vecinos de un segmento de un mapa de una ciudad real. En el Capítulo 5, utilizando indicadores como procesamiento de *CPU*, tamaño del archivo de escenarios generado y tiempo de ejecución se evaluó la eficiencia computacional. A su vez utilizando métricas como retardo, rendimiento del protocolo de enrutamiento, rendimiento del protocolo de aplicación, paquetes borrados se evaluó el comportamiento de protocolo de enrutamiento *AODV* cuando utiliza los patrones de movilidad generados por *UPM* y *AVM*.

INTRODUCCIÓN

La comunicación en redes móviles ha suscitado un gran interés desde hace más de una década que perdu¬ra hasta nuestros días y según estimaciones aumentará su trascendencia en el futuro [1]. Con el estudio de las redes de celulares [2] y la indagación de su conveniencia en la comunicación humana, las redes móvi¬les fueron labrando su consolidación de la mano con el aumento de la demanda, la miniaturización de los aparatos móviles y la diversificación de sus productos. Dando como resultado grandes ganancias a las empre¬sas productoras y generadoras de servicios móviles que incitaron al aumento del atractivo de la inversión en este tipo de segmento de mercado que obligó a la contratación de una mayor cantidad de especialistas del área de telecomunicaciones para que se encargará de un estudio más minucioso del tema [3] [4].

La tecnología sigue evolucionando rápidamente por lo que se sumarán más aparatos a las redes móviles en un futuro cercano. Las redes móviles *Ad hoc* (*MANET)* dominadas por: celulares, PDA's, blackberries, etc., se podrían integrar en un futuro con nuevos aparatos inteligentes lo cual maximizará el espectro comunicacional que puede ser aprovechado para generar utilidades para el beneficio de la sociedad [2] [3] [6].

Para comprobar su efectividad sin incurrir en gastos cuantiosos es conveniente simular las redes móviles, tal como es el caso de las redes *MANET*, lo cual hace imprescindible que se manejen escenarios de prueba, además de los protocolos y recursos necesarios para generar la comunicación. Los escenarios de redes móviles se fundamentan en el uso de modelos de movilidad que buscan simular la realidad mediante métricas y comportamientos pre-definidos que van a regir a todo nodo de la red.

Se ha incrementado el interés al análisis de los modelos de movilidad orientados a representar la movilidad humana en diversos escenarios con o sin obstáculos porque tiene una amplia gama de aplicaciones especialmente en situaciones que pueden ayudar a minimizar situaciones de riesgo. Los escenarios de emergencia y rescate cargados de una gran cantidad de obstáculos suscitan, hasta la actualidad, un desafío para los estudiosos de las redes móviles debido a su gran aplicabilidad en escenarios de crisis donde el único medio

de comunicación posible puede ser un dispositivo electrónico móvil de uso cotidiano.

OBJETIVO GENERAL

• Implementar un modelo de movilidad en la herramienta Scengen que permita generar escenarios de emergencia y rescate en un área con obstáculos.

OBJETIVOS ESPECÍFICOS

• Determinar el tipo movilidad que se utiliza para obtener escenarios de emergencia y rescate con obstáculos.
• Analizar la herramienta *Scengen* en escenarios de emergencia y rescate.
• Lograr la compatibilidad funcional del modelo de movilidad y la herramienta.
• Determinar cuál es la estructura y ubicación de la herramienta para *NS2*.
• Validar los escenarios obtenidos de forma visual y cuantificable.
• Realizar un análisis con los modelos de movilidad implementados, para determinar cuál de ellos da una mayor prestación a la red.

RESULTADOS ESPERADOS

Una vez terminada la presente tesis se espera conseguir los escenarios de emergencia y rescate generados en la herramienta *Scengen* empleando los dos modelos de movilidad desarrollados para el efecto.

PARTE I
MARCO DE REFERENCIA

CAPÍTULO 1. FUNDAMENTO TEÓRICO

1.1 Redes *MANET*

Las redes *MANET* nacen de la necesidad de comunicación entre las personas que se encuentran en sitios inhóspitos o en situaciones delicadas, en las cuales se tiene que montar una especie de infraestructura lógica en la cual cada miembro de la red se convierte en un transmisor o un medio de enlace con otro dispositivo móvil. Su topología es altamente dinámica debido a la cantidad de conexiones y desconexiones que se producen a lo largo del tiempo [6]. Además, por la naturaleza de sus hosts se debe tener en cuenta que poseen capacidades limitadas lo que hace que se deba buscar el uso eficiente de los recursos con que cuentan. Debido a esto, el estudio real del comportamiento de estas redes resulta muy costoso especialmente en una zona de desastre. La Figura 1.1 muestra una configuración de una red móvil *Ad hoc*. Para determinar comportamientos que se generan en redes *MANET* se utiliza cierto software que permite de una manera experimental simularlos. La ventaja de las simulaciones es que se puede manipular los parámetros o las variables según lo que se pretenda analizar[2]. La simulación de una *MANET* necesita de la especificación de un escenario para la determinación de los valores para los parámetros relacionados a su movilidad.

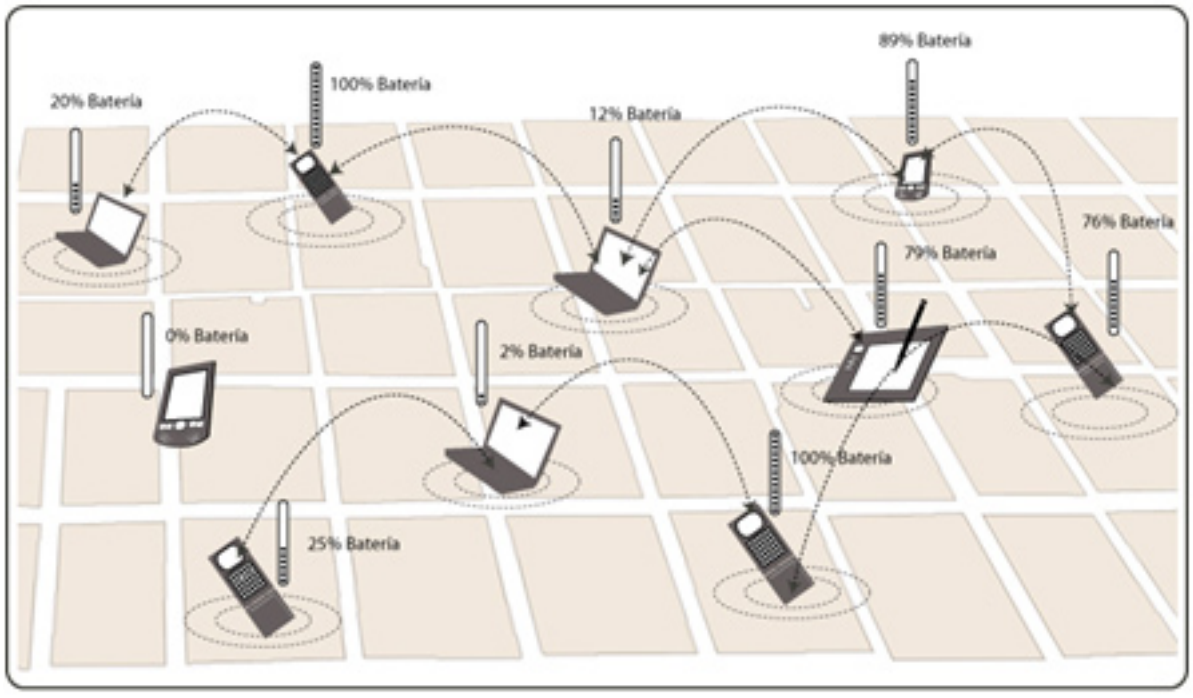


Figura 1.1 Comunicación entre dispositivos móviles

1.1.1 Escenario

Un escenario en una red *MANET,* es una representación controlada, limitada y previamente establecida de un ambiente determinado. Estos escenarios habitualmente constan de las siguientes partes: área, nodos, modelo(s) de movilidad, camino(s) y objetivo(s).

Área: Un área en un escenario de movilidad es el marco generalmente definido por un largo y un ancho, en el cual se desarrolla toda la actividad generada por la interacción entre los nodos según lo dictan sus modelos de movilidad asociados. Las formas de áreas pueden ser diversas y puestas a conveniencia, aunque las más utilizadas son las circulares y rectangulares como lo indica la Figura 1.2. Hay dos tipos de composiciones de escenarios en las simulaciones: escenario libre de obstáculos y escenario con obstáculos.

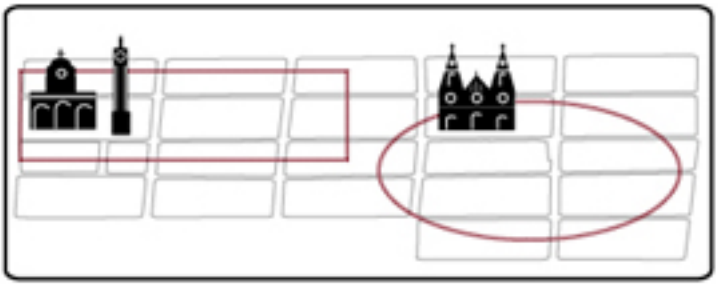


Figura 1.2 Áreas más usadas en los escenarios de movilidad.

Nodos: Son objetos móviles, a los cuales se les asigna un modelo movilidad en forma individual o grupal. Ocupan una posición en el área *(x,y)* y cumplen un rol representativo de un objeto real en el escenario. [7] La naturaleza de los nodos pueden ser de varios tipos de dispositivos electrónicos, inclusive los que se manejan en la actualidad como lo indica la Figura 1.3: *Laptops*, *tabletas*, *PDA's*, *celulares*, etc., pero a la vez se puede considerar a un ser vivo como un nodo si lleva consigo por lo menos un dispositivo que pueda establecer o permita unirse a una red móvil. Como se observa en la Figura 1.3 se presentan 4 dispositivos de uso extendido en todo el mundo.

Figura 1.3 Los aparatos móviles actuales.

1.2 Modelos de movilidad

Los modelos de movilidad son importantes porque determinan el comportamiento de los nodos siendo de esta manera un facilitador o una traba para la consecución de una comunicación eficiente[8]. Los modelos de movilidad permiten forjar camino(s) y alcanzar objetivo(s) dentro de un escenario de movilidad.

Camino(s): Son las rutas obtenidas por procedimientos matemáticos especiales para que el nodo logre su objetivo. Los caminos pueden ser abstracciones de disposiciones reales como mapas o de conceptos idealizados como líneas horizontales y verticales. Según el modelo de movilidad que adopten y la naturaleza de la concepción del escenario, podrían determinar condicionantes de movilidad, como por ejemplo: sentido de las vías, señales de tránsito, etc.

Objetivo(s): Es el destino que se dispone con anterioridad de forma estática en el área del escenario. Este debe ser alcanzado mediante la movilidad y el cumplimiento de parámetros de simulación.

Se pueden encontrar dos tipos de modelos de movilidad: los basados en trazas que hacen referencia a *logs de movimientos reales* [10] y los sintéticos que emulan la realidad mediante ecuaciones matemáticas que posteriormente serán implementadas en un simulador de redes dentro de un computador que permita generar movimientos controlados tal como lo ilustra la Figura 1.4. Estos modelos sintéticos, para generar su movimiento característico, emplean un conjunto de parámetros. Entre los más comunes están:

- Tiempo mínimo y máximo.
- Tiempo de pausa.
- Velocidad mínima y máxima.
- Dirección.
- Aceleración.
- Tipo de distribución estadística.

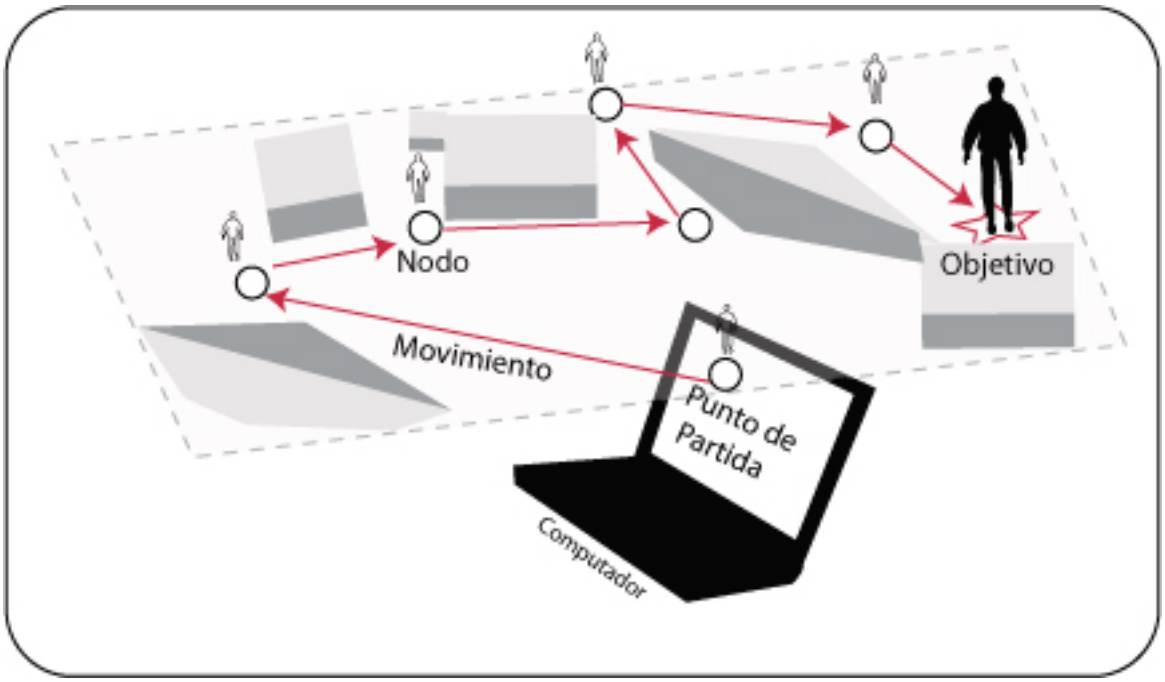


Figura 1.4 Movilidad sintética.

Los modelos de movilidad sintéticos se clasifican en:

- Modelos de movilidad sintético no realista
- Modelos de movilidad sintético realista

1.2.1 Modelos de movilidad sintéticos no realistas

Los modelos de movilidad sintéticos no realistas se los pueden considerar aquellos que no son aplicables para la simulación de la movilidad de las personas debido a que sus movimientos resultantes están aplicados a la movilidad de otras realidades físicas o estructurales de entes radicalmente opuestos al comportamiento humano.

1.2.1.1. Modelos aleatorios

Los modelos aleatorios son representaciones que obtienen todos sus parámetros como dirección, velocidad, la aceleración, etc., de valores escogidos al azar. Entre sus principales representantes tenemos:

Random walk mobility model: Fue el primer modelo de movilidad creado con el objetivo de imitar el comportamiento de ciertos organismos erráticos. El modelo antes de iniciar asigna a cada nodo una velocidad tomada del rango *[Vmin – Vmax]*, una dirección del rango *[0 - 2Π]* y una posición aleatoria. [9] Inmediatamente se inicia su movimiento, el cual prosigue su camino sin alterarlo durante un desplazamiento constante *(d)* o en un intervalo de tiempo fijo*(t)*. Al llegar a su destino o al haber consumido todo su tiempo, se calcula una nueva dirección y velocidad. Cuando el nodo esta cerca al límite del escenario, rebota y es asignada una nueva dirección. Este proceso se lo resume en la Figura 1.5. A este modelo se lo conoce también como movimiento *Browniano*.

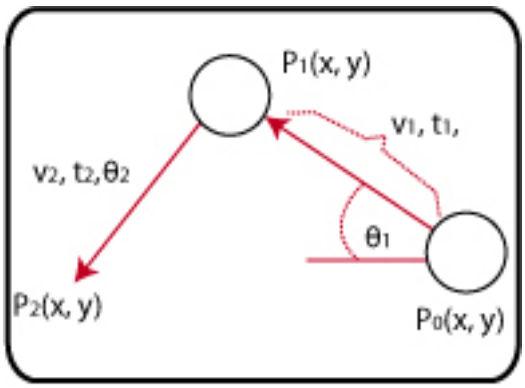


Figura 1.5 Movimiento de un nodo bajo random walk model

Random waypoint mobility model: Este modelo inicia ubicando a cada nodo en una posición aleatoria y con una velocidad tomada del rango *[Vmin – Vmax]*. El nodo se desplaza a lo largo de un período de tiempo y luego, cuando este acaba, se realiza una pausa y se calcula una nueva velocidad y dirección. El proceso se lo sintetiza en la Figura

1.6. Este modelo es muy utilizado para la evaluación del rendimiento y de los parámetros (velocidad y dirección). Especialmente, se lo usa en la herramienta *NS2* por su sencillez y además, por ser el origen de otros modelos. Por su uso extendido, los investigadores han llegado a formular mejoras, tales como: *Random direction mobility model* (disminuye la concentración central en el desplazamiento de los nodos) [8], *Trip mobility model* , entre otros.

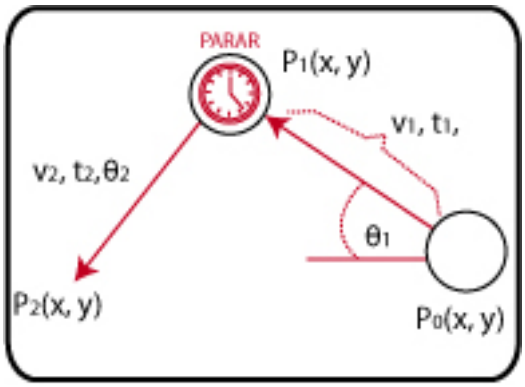


Figura 1.6 Movimiento de un nodo bajo waypoint model

Boundless simulation area mobility model: El modelo establece una relación entre la velocidad y la dirección anterior con las actuales. El vector velocidad está representado por *vector V (v,θ)* donde v es la velocidad y θ es la dirección. La posición del nodo está definida por las coordenadas *(x,y)*. Todos estos valores son cambiados en cada variación del tiempo *(t)* empleando las siguientes fórmulas:

$$v\,(t + \Delta t) = min\,[max(v(t) + \Delta v,\, 0), Vmax\,] \quad (1.1)$$

$$\theta\,(t + \Delta t) = \theta(t) + \Delta\theta\,; \quad (1.2)$$

$$x\,(t + \Delta t) = x(t) + v(t) * cos\,\theta(t); \quad (1.3)$$

$$y\,(t + \Delta t) = y(t) + v(t) * sin\,\theta(t); \quad (1.4)$$

Donde,

$\Delta v = [-Amax * \Delta t,\ Amax * \Delta t]$, *Amax* es la aceleración máxima

$\Delta\theta = [-\alpha * \Delta t,\ \alpha * \Delta t]$, α es la variación angular de la dirección

Este modelo trabaja en un escenario sin límites, por lo que si un nodo sale del escenario reaparece en el lado opuesto. Este comportamiento se lo puede observar en la Figura 1.7.

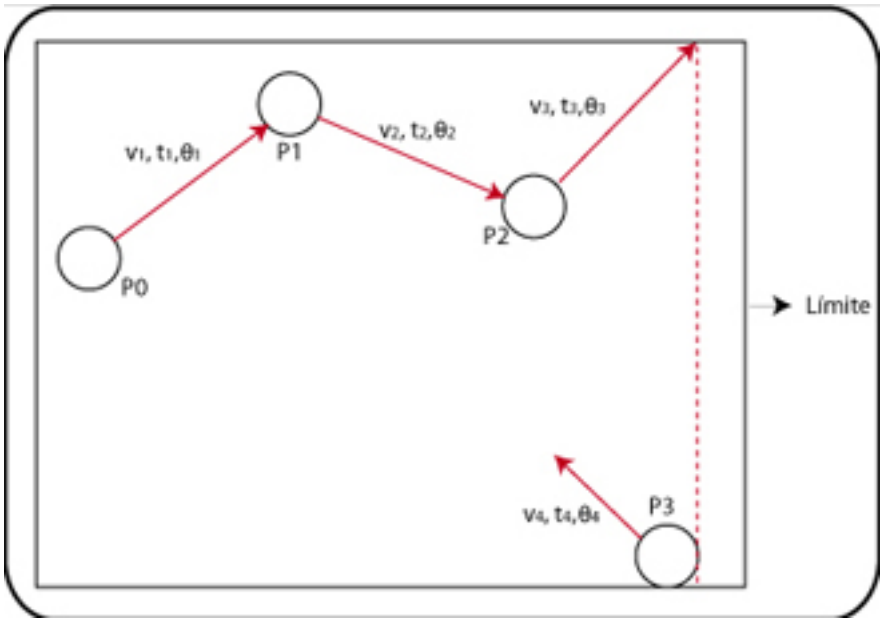


Figura 1.7 Movimiento boundless simulation mobility area

1.2.1.2 Modelos de dependencia temporal

Los modelos de dependencia temporal tratan de corregir las debilidades de realismo que poseen los modelos anteriores como es la parada súbita, el cambio drástico de velocidad y direccionamiento; empleando un proceso comparativo entre sus valores actuales y los valores siguientes estimados para el próximo movimiento.

Gauss-Markov mobility model: Este modelo fue adaptado para una amplia gama de aleatoriedad variando su principal parámetro (*tuning parameter*). Un nodo inicia con una velocidad y una dirección asignada, que luego de cierto periodo de tiempo son reasignadas, se lo muestra en la Figura 1.8. Para calcular el valor de la n-ésima velocidad y dirección del siguiente movimiento *(n)* se toma en cuenta sus valores anteriores *(n-1)* [11] y una variable al azar. Se aplican las siguientes ecuaciones:

$$sn = \alpha\, sn-1 + (1-\alpha)\bar{s} + \sqrt{(1-\alpha^2)}\, s_{x_{n-1}} \quad (1.5)$$

Donde,

sn = Nueva velocidad del nodo en un intervalo de tiempo n

α = Variación de aleatoriedad ($0 \leq \alpha \leq 1$)

$\bar{s}$ = Constante del valor medio de la velocidad cuando $n \rightarrow \infty$

$s_{x_{n-1}}$ = Variable de aleatoriedad gaussiana

$$dn = \alpha dn - 1 + (1 - \alpha)\bar{d} + \sqrt{(1 - \alpha^2)}\, d_{x_{n-1}} \quad (1.6)$$

Donde

dn = Nueva dirección del nodo en un intervalo tiempo n

α = Variación de aleatoriedad ($0 \leq \alpha \leq 1$)

$\bar{d}$ = Constante del valor medio de la dirección cuando $n \rightarrow \infty$

$d_{x_{n-1}}$ = Variable de aleatoriedad gaussiana [16]

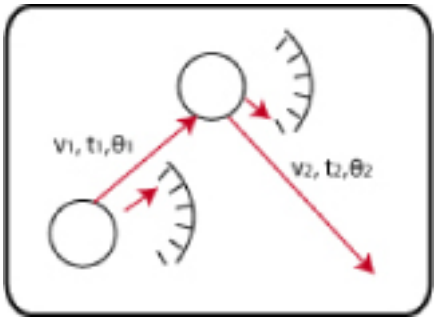


Figura 1.8 Movimiento gauss-markov

Smooth random mobility model: Este modelo suaviza el movimiento de los nodos controlando los cambios abruptos de velocidad al realizar un cambio de dirección mediante la ejecución de una desaceleración antes de que se acabe el intervalo de tiempo tal como se lo indica en la Figura 1.9. Todas sus nuevas velocidades y direcciones son calculadas a partir sus respectivos valores actuales. Los cambios de frecuencia y de velocidad son realizados mediante un recurso denominado *Proceso Poisson* donde el cambio de velocidad de aceleración y desaceleración es estimado de una función de probabilidad de distribución de velocidades que son tomados de valores uniformemente distribuidos de los rangos: *[0, Amax]* y *[-Amin, 0]*, respectivamente.

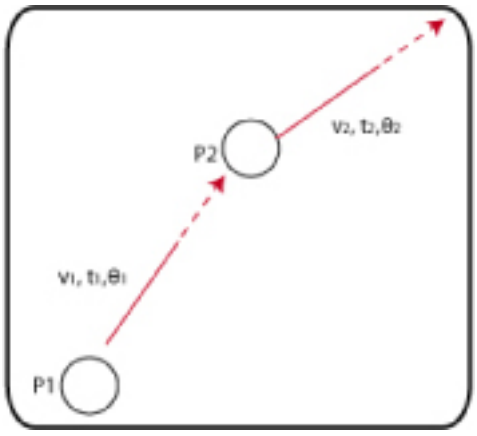


Figura 1.9 Movimiento Smooth Random Mobility Model

1.2.2 Modelos de movilidad sintéticos realistas

Son aquellos modelos que simulan de forma más fiel a la realidad el movimiento de los seres humanos mediante la aplicación de estrategias de movilidad.

1.2.2.1 Modelos de dependencia espacial

Contiene los modelos de movilidad que manejan grupos de nodos que comúnmente se encuentran regidos por el movimiento de un nodo principal que define el tipo de movilidad del grupo como una unidad.

Reference point group mobility: Su movimiento grupal está determinado por un modelo de movimiento arbitrario mientras que su movimiento interno (cada nodo) está asociado a un punto de referencia como se lo grafica en la Figura 1.10. La posición actual del nodo es el resultado de la aplicación de la aleatoriedad conjuntamente con la posición de su punto de referencia [12]. La posición de los puntos de referencia puede cambiar.

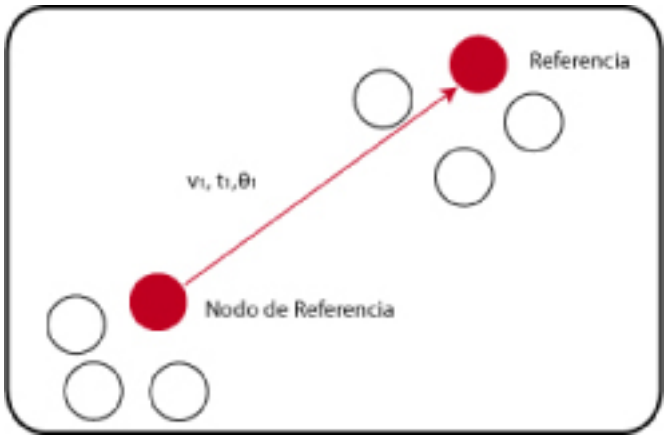


Figura 1.10 Movimiento *RPGM*

Column mobility model: En este modelo de movilidad, un nodo hace el papel de líder y el resto lo siguen componiendo una recta (*reference grid*) similar a un grupo de soldados en formación. Los nodos siguen a su punto de referencia aplicando un modelo aleatorio. El *reference grid*, es una línea que se mueve calculando una distancia y aplicando un ángulo entre el rango *[0-Π]*, los nodos se desplazan a la par con el *reference grid* aunque siempre manteniendo su movimiento con respecto al punto de referencia [8] como se lo muestra en la Figura 1.11.

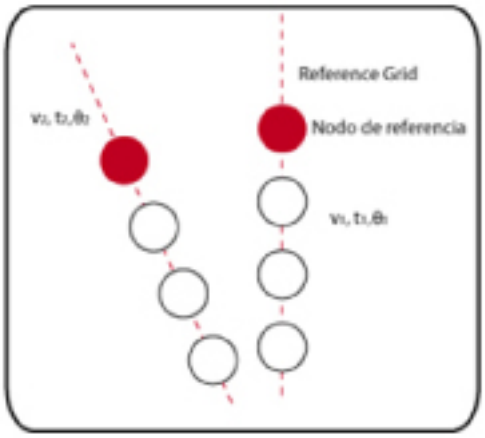


Figura 1.11 Movimiento column model

Pursue mobility model: El movimiento utiliza la misma estrategia que la utilizan los policías al seguir a un ladrón [11]. Al ser implementado en el modelo, los nodos persiguen a un objetivo, esto se lo representa gráficamente en la Figura 1.12.

Cada nodo regula su movimiento aleatorio con el fin de seguir al nodo objetivo. Las posiciones de los nodos se obtienen aplicando la siguiente ecuación:

$$\text{nueva posición} = \text{posición anterior} + \text{aceleración} \quad (1.7)$$
$$= (\text{objetivo} - \text{posición anterior}) + \text{vector}$$

Donde, el vector es obtenido mediante un modelo randómico.

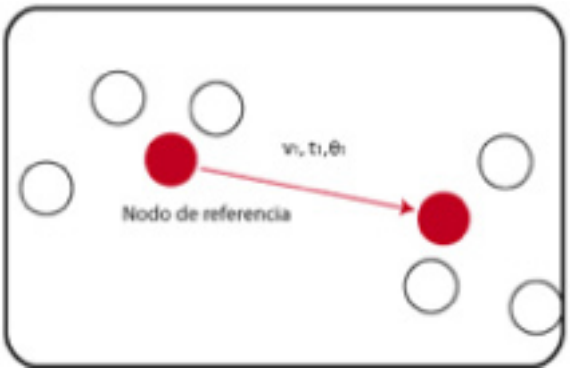


Figura 1.12 Movimiento persue model

Nomadic mobility model: Su nombre evoca al comportamiento humano de épocas remotas donde se viajaba en grupos errantes gobernados por un líder. Al implementar el modelo se abstrae esta característica. Por lo que todos los nodos persiguen a un nodo de referencia pero manteniendo su espacio personal donde se mueven de forma randómica. Este proceso se lo muestra en la Figura 1.13. Cuando se cambia un punto de referencia los

parámetros del modelo se ponen como inalcanzables dejando tiempo para que los nodos se actualicen. [11]

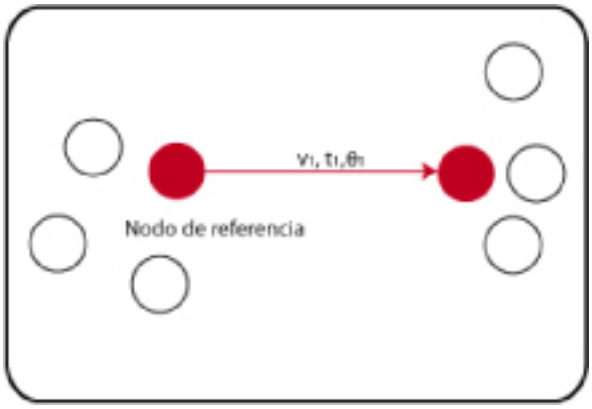


Figura 1.13 Movimiento nomadic model

1.2.2.2. Modelos de restricción geográfica

Los modelos de restricción geográfica ponen énfasis en dar las soluciones a las limitantes que ofrecen un escenario, en los cuales se puede encontrar una variedad de restricciones que obligan a plantear una eficiente movilidad de los nodos. **Pathway models:** Estos tipos de modelos son desarrollados para escenarios que se fundamentan en la obtención de caminos para la circulación de los nodos hasta alcanzar su objetivo, estos caminos pueden ser representaciones de rutas, carreteras o autopistas de ciudades que dependiendo de la flexibilidad del escenario establecido deberán acatar restricciones de los caminos como límite de velocidad, señales de tránsito, etc. Algunos modelos que pertenecen a este grupo son: City Section, Manhattan Model, entre otros [10] [14]. En la Figura 1.14 se representa la movilidad de un modelo de esta clase.

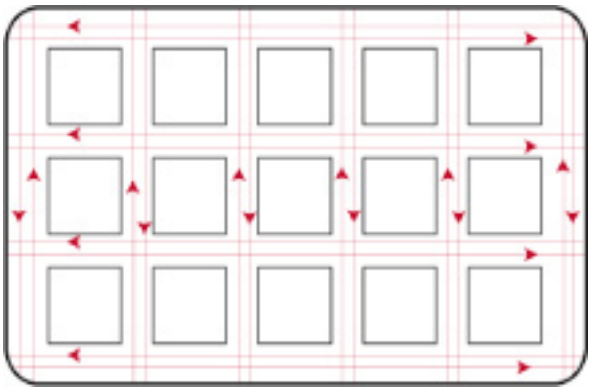


Figura 1.14 Manhattan model

Obstacle Models: Estos modelos se fundamentan en obtener soluciones a partir de los obstáculos que son definidos en el área del escenario [14]. Su objetivo se logra aplicando soluciones basadas en movimientos desde vértices como: *Diagramas de Voronoi* que se lo muestra en la Figura 1.15 y *el camino más corto* mostrado en la Figura 1.16.

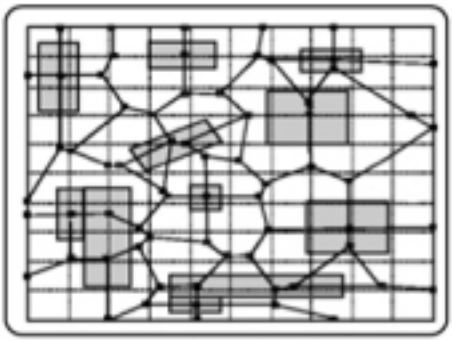


Figura 1.15 Diagrama de voronoi [13]

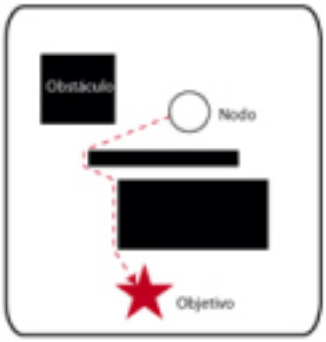


Figura 1.16 El Camino más corto

A este tipo de modelos de movilidad pertenecen: *Human Mobility Obstacle* (*HUMO) Model*, *Obstacle Advoidance Mobility Model* (*OAM*), entre otros más.

HUMO: El propósito de este modelo es obtener un movimiento real de una red *Ad hoc* operada por humanos en una zona de desastre como: terremotos, incendios, etc., donde los equipos de salvamento tales como: médicos, bomberos, policías y más, se enfrentan a un escenario lleno de obstáculos. El nodo primeramente establece la posición del destino *d* y luego, analiza el vértice *P1* del obstáculo *P* más cercano al nodo. En caso de que este vértice sea inaccesible porque otro obstáculo *Q* lo impide. El nodo selecciona un vértice más cercano *Q2* del obstáculo *Q* al vértice *P1* de *P*. Este proceso se realiza hasta alcanzar el destino *d* [15]. La Figura 1.17 muestra este proceso.

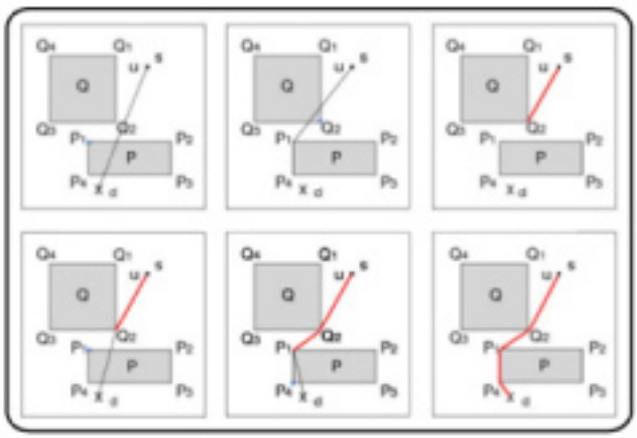


Figura 1.17 Modelo *HUMO*[15]

OAM: El modelo es una variante mejorada del modelo *HUMO*, igual que éste, tiene que moverse hacia el objetivo pero el modelo *OAM* implementa la búsqueda del camino más corto evitando los obstáculos de un área previamente establecida. Los obstáculos son formas poligonales cuyos vértices ocupan una posición geográfica *(x,y)*. Los nodos conocen la ubicación de cada vértice en el área del escenario. Por lo que antes de realizar el movimiento a su objetivo randómicamente escogido, hace un análisis y selección de vértices próximos que formen una ruta corta hacia el destino como se lo muestra en la Figura 1.18. Cuando se ha obtenido el camino más corto, el nodo se mueve y llega al destino. Posteriormente, inicializa nuevamente el proceso anterior [13].

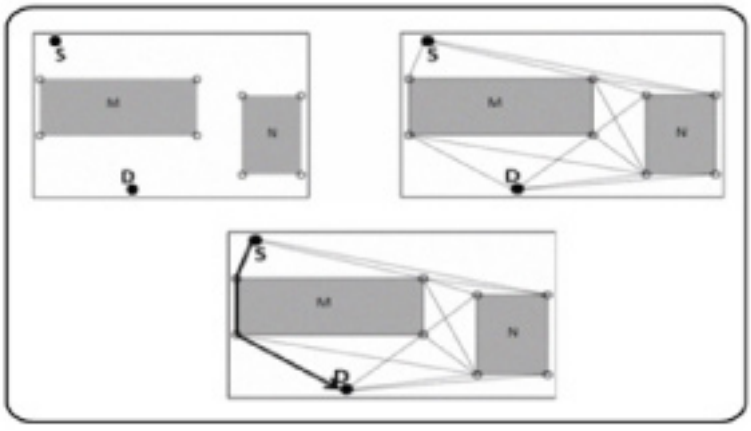


Figura 1.18 Modelo OAM [13]

1.3 Escenarios de movilidad de ayuda y rescate de centros urbanos

Es un ambiente de representación de desastres de distinta índole en donde se trata de evaluar la eficiencia de los diferentes modelos de movilidad existentes o creados. Los elementos integrantes del escenario ya toman nombres representativos como: damnificados, cuerpo de rescate, etc.; y a la vez, se abstraen criterios, estrategias y experiencias de la realidad para modelar, restringir y validar su comportamiento sobre el escenario. Por lo que se realizan simulaciones sobre comportamientos característicos en ambientes de desastre o de salvamento dentro de un área urbana.

1.4 Perspectivas actuales para el estudio de los escenarios de ayuda y rescate de personas

Centros especializados de investigación de varias partes del mundo se han dedicado a realizar estudios acerca de modelos, sistemas y ambientes de desastres con la finalidad de aportar con senderos de exploración en el campo de ayuda y rescate. Una línea de investigación que está abierta es la de evaluación de las herramientas de socorro usadas y la posterior optimización mediante la propuesta de mejoras tentativas. Éste es el caso del sistema de Red de Ayuda de Desastres, *DAN*[15][16], que es el resultado de la abstracción de la estrategia real de: *Mass Casualty Events*[17][18] empleada para desastres por parte de la *OOC*[4][16] de Alemania. Mediante el análisis del producto se hicieron algunas propuestas de optimización, una de ellas es la implementación de dos *filtros bayesianos*[18] basados en los algoritmos de estimación de la posición en el proceso del rastreo de los pacientes[16]. Otra propuesta es la inserción de un algoritmo llamado *Range-Based Monte Carlo*[17] para rastrear óptimamente a una gran cantidad de pacientes.

En algunos casos se ha optado por implementar conceptos exitosos y comprobados en otras ramas de las ciencias como es el caso de la inteligencia artificial mediante la cual se propone la integración de agentes autónomos que actúan como sensores intermedios, como es el caso del *Meta- Modelo de Movilidad Framework Ghost* que permite modelar y simular ambientes de desastre, empleando un paradigma basado en reglas de comportamiento del nodo. [19]

Otra forma, es realizando simulaciones prototipo en herramientas tales como la ampliamente utilizada con fines académicos *Network Simulator 2* (*NS2*), que permitirá evaluar las capacidades de nuevas implementaciones compatibles con la aplicación que pueden ser creadas,

manipuladas o descargadas de la red con el objetivo de analizar de forma comparativa, detallada y visual, en base de los datos encontrados. Pudiéndose determinar de esta manera, si se ajusta a un panorama útil dentro del área de ayuda y rescate. Cabe indicar que en la actualidad existe una gran cantidad de simuladores desarrollados en múltiples lenguajes de programación con amigables interfaces que pueden ser un buen argumento para probar y comparar las herramientas.

1.5 Factores restrictivos para la creación de escenarios de ayuda y rescate para sobrevivientes en centros urbanos

Los escenarios de ayuda y rescate en la vida real son complejos, puesto que se trabaja con grupos humanos que siguen una planificación determinada donde el principal objetivo es precautelar las vidas de las personas que hayan sobrevivido al desastre. La clave está dada en la planificación de estrategias [17][18] y restricciones, que deberán seguir los cuerpos de salvamento como los Bomberos, Cruz Roja, Policía, etc. [20]. Por lo general los modelos de movilidad creados lo hacen desde el punto de vista del grupo de salvamento. Esta tesis se enfoca en el punto de vista del sobreviviente, en el cual se debe comprender su realidad y plasmarla en una movilidad que lo represente. Además, diversos factores se puede considerar al desarrollar el modelo como: la idiosincrasia y la educación preventiva de desastres.

Varios desastres naturales dantescos han ocurrido el año 2011, como el terremoto y posterior tsunami que barrieron la ciudad de Sendai afectando a otras localidades de archipiélago nipón. Además, que provocó el desastre nuclear en la planta de Fukushima. A pocos días de esto, se desencadenó una serie de voraces tornados en el medio oeste centros urbanos. Esto no es nuevo, solo debe de hacer un análisis cronológico sobre desastres en el mundo. Por ejemplo: La erupción volcánica del Vesubio que hizo desaparecer a la floreciente Pompeya, el pavoroso incendio de la ciudad de San Francisco a inicios del siglo XX, el bombardeo aliado a la ciudad de Dresde en la segunda guerra mundial. Si esto parece muy internacional solo vasta revisar un libro de historia para tener ejemplos locales. Lo que lleva a afirmar que los desastres en poblaciones siempre ha habido y habrán, convirtiendo a los ciudadanos en los actores principales de estos eventos. Por ello, este trabajo se orienta a la simulación de los sobrevivientes de un desastre.

Los sistemas de comunicaciones por lo general y por referencias históricas se conoce que colapsan o son nulos después de un desastre esto nos demostró principalmente las tragedias vidas en Japón este año (2011) y la de Chile hace un año atrás. Por lo que generar una red inalámbrica móvil Ad-hoc (MANET) empleando los dispositivos móviles que pueda contar los sobrevivientes sería algo factible para un panorama de esta naturaleza. Aquí se produce una metamorfosis de términos en la cual el ciudadano luego del incidente se convierte en un sobreviviente y cuando se establece la red pasa a ser un nodo móvil (MN). En las zonas de desastres, puede haber más de una persona en una misma posición geográfica como se lo puede apreciar en la Figura 1.19.

Figura 1.19. Terremoto en Japón 2011[1]

Las ciudades están llenas de múltiples construcciones de distintas formas, materiales y tamaños que dan ese brillo tan particular a las urbes. Pero luego de suscitado el desastre de grandes proporciones, solamente quedan en ruinas, bastas zonas inaccesibles y algunas estructuras en pie. La fisonomía de la ciudad cambia drásticamente y algunas vías quedan bajo los escombros lo que constituyen obstáculos a las vías de evacuación. Los medios de transporte dejan de funcionar en su gran mayoría por lo anteriormente mencionado o forman parte de los daños. Inclusive la cantidad de cadáveres podría ser un problema en ciertos escenarios. Una amplia variedad de materiales dispuestos en un mismo lugar se

1 http://www.avivamiento.com/blog/?p=1895

constituyen en obstáculos. Cabe hacer notar que los obstáculos no tienen una forma definida pero por el propósito de abstracción para simulación podría tomar una forma de cuadrilátero generalmente un cuadrado o rectángulo, como lo toman muchos juegos como el *Age Empires*, *Comandos*, *SimCity*, etc. Además, se puede aducir que es argucia de simplicidad estética basada en las leyes de la *Gestalt* (leyes de diseño) [21] utilizada en diversas utilidades ergonómicas reales como textos, envolturas, etc. Debido a esto en el presente trabajo de tesis se busca encaminar a los modelos que trabajan sobre objetos de formas no ideales.

Finalmente, señalar que se debería considerar dentro del modelo el espíritu de solidaridad que prima en el ser humano cuando se producen estos eventos que puede hacerle decidirse ayudar a otro sobreviviente incapacitado de movilidad (mal heridos, bajo los escombros) que estaría en competencia directa con el espíritu de supervivencia que señala que no sería factible que los supervivientes ingresen o estén sobre los obstáculos debido a que la primera reacción de las personas es salir de los lugares donde se encuentran y alejarse de zonas de peligro. Como también lo recomiendan las campañas de seguridad en muchos países (Defensa Civil, Cruz Roja, etc.).

Conociendo estas características y lo valioso que serían las redes *MANET* en áreas de desastre aún más después de haber decidido que los modelos de movilidad (*UPM*, *AVM*) van interactuar dentro de escenarios de este tipo. Se tomó como punto de partida, investigar el comportamiento real de un damnificado para que de esta forma se pueda emplear de manera efectiva en el diseño de la movilidad del nodo. Los datos fueron proporcionados mediante una entrevista al Cuerpo de Bomberos de la Ciudad de Loja, como se lo puede ver en el Anexo 1. Las conclusiones sobre el comportamiento de un damnificado son las siguientes:

- Por instinto de supervivencia el ser humano busca su propia salvación cuando surge un desastre.
- Cuando la persona está buscando ir a un determinado sitio, esta no ingresa a los escombros para acortar camino sino busca los espacios libres.
- Los únicos medios de comunicación que luego que pase un gran desastre en la ciudad de Loja, sólo los manejarán los equipos de emergencia y rescate.

- Loja se encuentra desprovista de planes de salvamento y comunicación.
- Las personas después de un desastre acuden a pedir ayuda a los centros de salvamento locales.

Esta información permitió determinar que:

- La movilidad de los nodos no es grupal.
- Los nodos no deben ingresar en los obstáculos por lo que son inaplicables métodos como: el diagrama de Voronoi entre otros.
- La movilidad debe de estar dirigida a un punto objetivo.

PARTE II
MODELOS PROPUESTOS

CAPITULO 2. ANÁLISIS DE LOS MODELOS UNMARKED POINT MODEL (*UPM*) Y ADJACENCY VERTEX MODEL (*AVM*).

2.1 Unmarked point model (*UPM*)

El modelo UPM se lo desarrolló pensando en el análisis que tiene que hacer una persona en un escenario atestado de obstáculos como es el caso de un escenario de emergencia y rescate. Debido a que el individuo debe de analizar un rango de posibilidades para realizar su próximo movimiento tal como se lo hace al cruzar un río. El movimiento de los nodos se hace a través de la verificación y uso de espacios libres adyacentes a la posición del nodo en cualquiera de los cuatro puntos de direccionamiento (arriba, abajo, derecha, izquierda).

Dada una posición del nodo para definir la próxima posición se usa: obstáculos, movimiento y camino.

Obstáculos

Los obstáculos son randómicamente distribuidos en una matriz binaria que es generada a partir del mapa de la ciudad de Loja para simular un área de desastre, como se lo podrá observar más adelante en esta sección la generación del escenario de desastre. Cada espacio ocupado está representado con 1 y el espacio libre con un 0. Permitiendo diferenciar la presencia de un obstáculo o un punto libre que puede convertirse en parte del camino al punto objetivo.

Movimiento

El nodo parte de una posición aleatoria seleccionada mediante la aplicación de una distribución randómica dentro del rango (1000 px X 2000 px) que se encarga de buscar en la matriz un punto al azar que tenga el valor de 0. Luego, evalúa los cuatro valores próximos a la posición actual estimando siempre la manera de acercarse a un punto objetivo definido, que en caso de un escenario de emergencia y rescate podría ser tomado como un punto de evacuación o de ayuda. Este punto objetivo constituye el destino a ser alcanzado. Los movimientos que realiza los hace en relación a fila o a columna.

Camino

UPM es muy minucioso para moverse pues trabaja en zonas que presentan bastantes obstáculos. A diferencia de modelos tales como *HUMO* u *OAM*, no se basa en los vértices del polígono sino en el estado de la siguiente posición libre. En el punto de evacuación de este modelo, si un nodo se encuentra en una posición (x_a, y_b) para acercarse su punto objetivo que está a (x_n, y_n), tendrá que elegir el espacio libre de sus posiciones próximas (x_{a-1}, y_b), (x_{a+1}, y_b), (x_a, y_{b-1}) y (x_a, y_{b+1}); que sea conveniente para alcanzar la posición (x_n, y_n). Este proceso se lo ilustra en la Figura 2.1 y en la Figura 2.2.

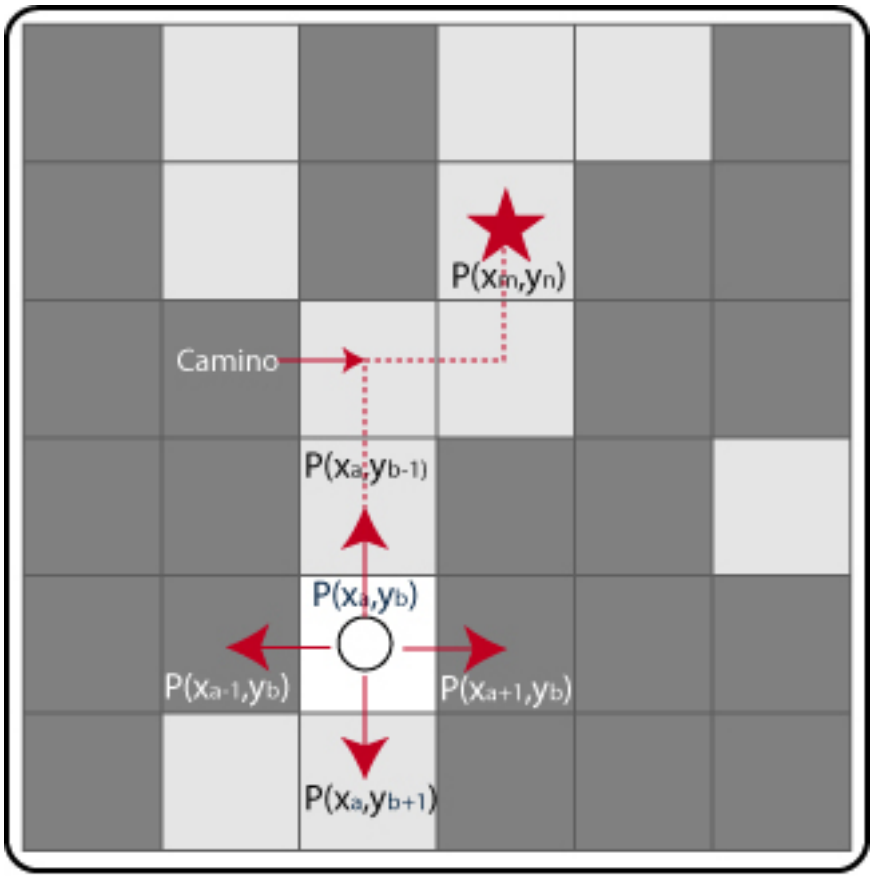


Figura 2.1 Movimiento del modelo *UPM*

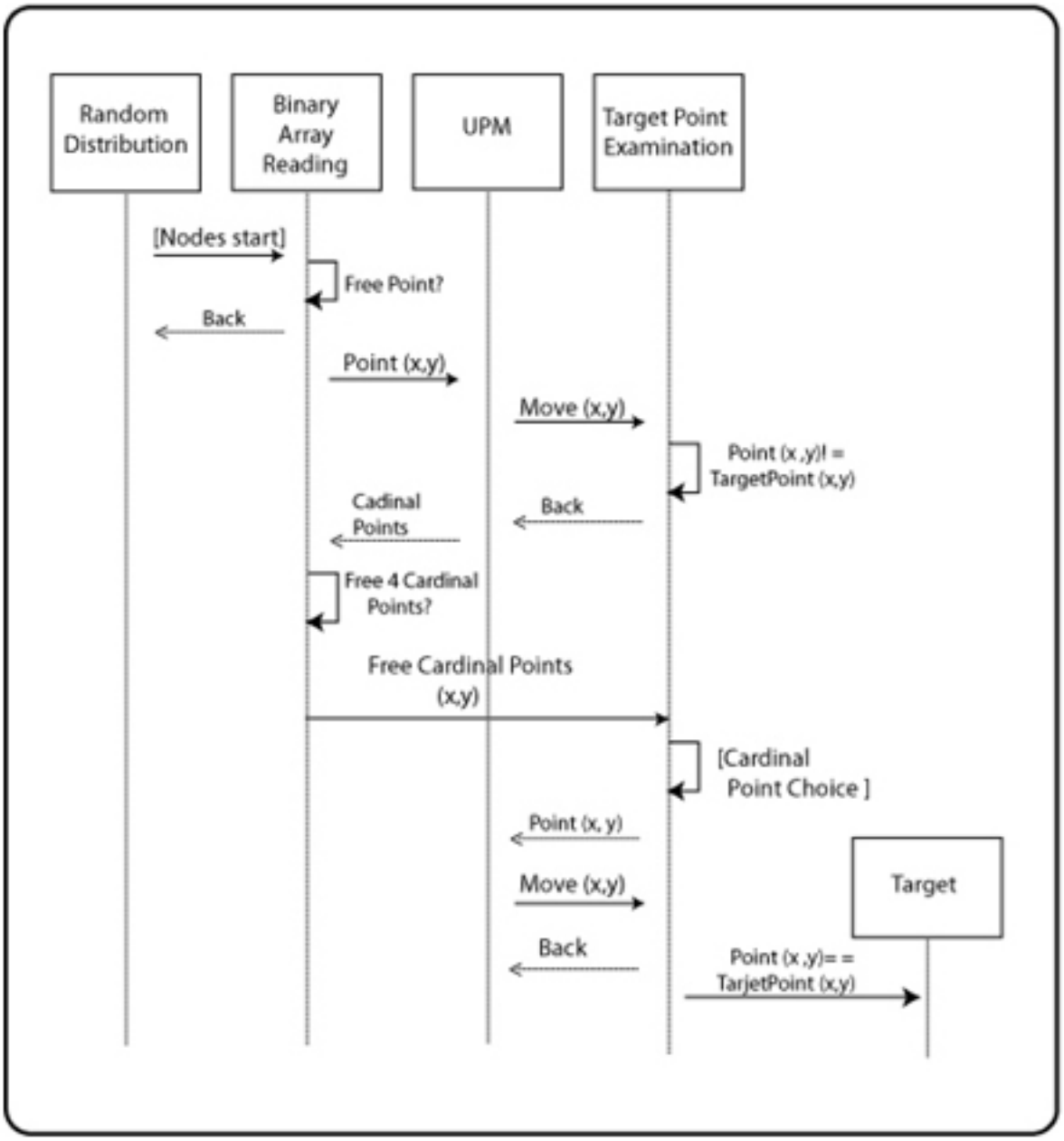


Figura 2.2 Ejecución del modelo *UPM*

A continuación se detalla en la Figura 2.3 el algoritmo en pseudocódigo del movimiento de los nodos implementado en la clase *Walk.* En la Figura 2.4 se muestra una tarjeta *CRC* de la clase *Walk* y en la Figura 2.5 el diagrama de objetos de la clase *Walk*.

Algorithm 1 UPM node movement.

Require: A binary matrix where it is detailed free and occupied spaces.

Ensure: Next nodes movement point (x,y).

```
1: if (((pX < pY) ∧ (p_target_X <> pX)) ∨ (p_target_Y == pY))
   then
2:   if (p_target_X < pX) then
3:     if (left_free) then
4:       (node_left_move)
5:     else
6:       (node_down_move) ∨ (node_up_move) ∨ (node_right_move)
7:     end if
8:   else
9:     if (right_free) then
10:      (node_right_move)
11:    else
12:      (node_down_move) ∨ (node_up_move) ∨ (node_left_move)
13:    end if
14:  end if
15: else
16:   if (p_target_Y < pY) then
17:     if (down_free) then
18:       (node_down_move)
19:     else
20:       (node_left_move) ∨ (node_up_move) ∨ (node_right_move)
21:     end if
22:   end if
23: end if
```

Figura 2.3 Algoritmo de movimiento de los nodos implementado en la clase *Walk.*

Walk	
Super Classes: Model	
Sub Classes:	
Description: Inicializa las variables de tiempo, velocidad y tipo de área. Asigna y evalúa los valores de los pares ordenados según la disponibilidad del espacio cercano (arriba, abajo, derecha e izquierda).	
Attributes:	
Name	**Description**
Px	Coordenada *x* de la siguiente posición
Py	Coordenada *y* de la siguiente posición
Random_Pos	Valor booleano para comprobar la aleatoriedad
T_dist_	Tipo de distribución randómica para obtener el tiempo en la simulación
T_max_	Tiempo mínimo
T_min_	Tiempo máximo
V_dist_	Tipo de distribución randómica para determinar los valores de la velocidad en la simulación.
V_max_	Velocidad máxima
V_min_	Velocidad mínima
Responsabilities	**Colaboration**
Inicializar variables (init)	modelspec
Obtener la próxima posición del nodo (dato)	
Asignar la siguiente posición al nodo (MakeMove)	random

Figura 2.4 Tarjeta CRC de la clase *Walk*.

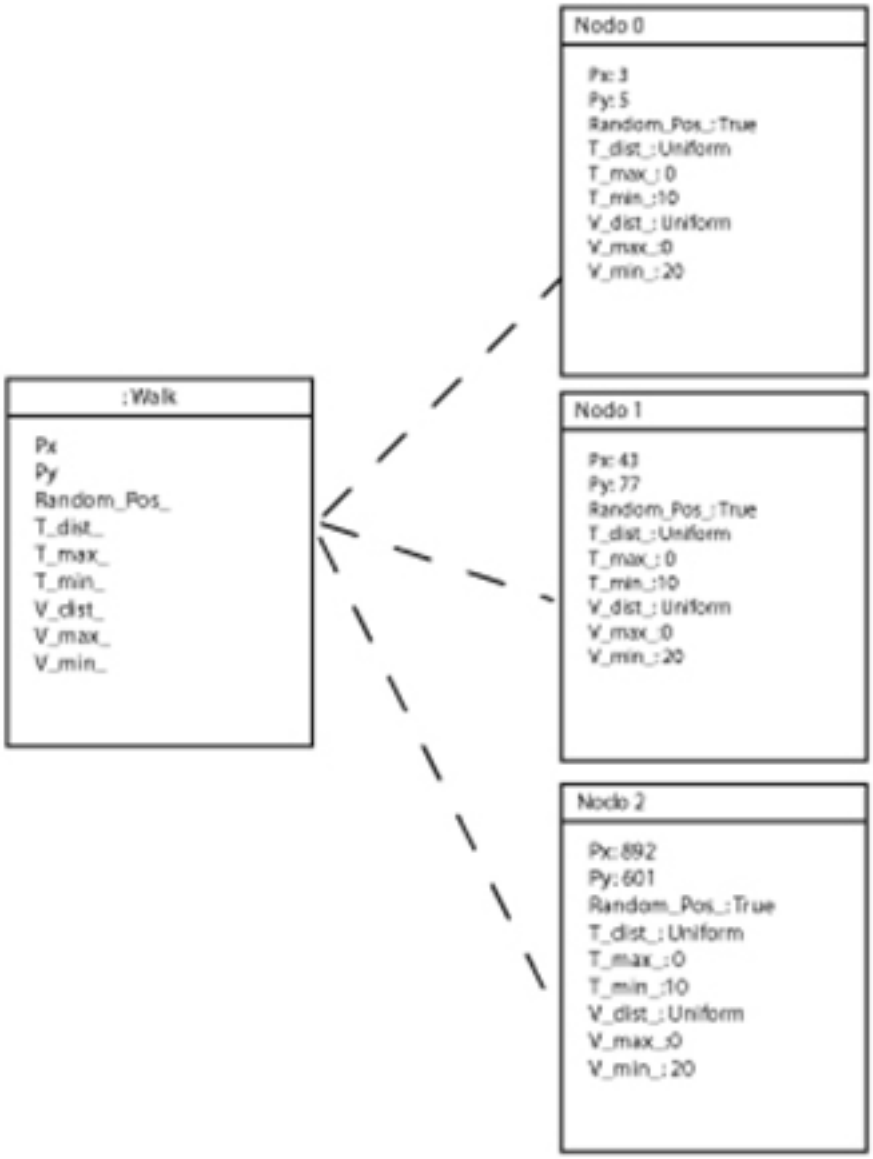


Figura 2.5 Diagrama de Objetos de la clase Walk.

2.2 Adjacency vertex model (*AVM*)

AVM es un modelo parecido al modelo *OAM* porque está basado en el modelo de movilidad humana (*HUMO*) y también aplica el concepto de camino más corto para ir hacia el objetivo. Pero a diferencia de este, se estructura en dos estrategias de funcionamiento que lo convierten en un modelo híbrido, es decir, en un modelo basado en obstáculos y caminos a la vez, la primera estrategia se usa para la generación de los obstáculos y de sus caminos, y la segunda estrategia para generar el movimiento de los nodos distribuidos randómicamente luego de un desastre.

Dada una posición del nodo para definir la próxima posición se usa: obstáculos, movimiento y camino.

Obstáculos

Los obstáculos son obtenidos de una matriz binaria que es generada mediante un proceso de inclusión al azar de patrones de 1's en el archivo del segmento del mapa en binario, formándose de esta manera polígonos que difieren de su forma original o unos nuevos dispuestos sobre áreas libres del mapa. Cada vértice del obstáculo cuenta con un conjunto de vértices adyacentes.

Movimiento

El nodo parte de una posición aleatoria, seleccionada mediante una distribución uniforme e inmediatamente se calcula la posición del vértice más cercano, haciendo la comparación cuantitativa del resultado de la distancia entre la posición actual y el vértice objetivo o punto de evacuación con el uso de la fórmula 2.1 de *distancia euclidiana* [2] entre dos puntos.

$$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \qquad (2.1)$$

Camino

El camino que se genera parte de un punto inicial hasta otro punto objetivo a través del salto más corto entre vértices obviando los obstáculos como bien lo define el algoritmo de la distancia más corta, pero al utilizar una matriz de adyacencia se puede basar únicamente en la

2 http://www.sectormatematica.cl/contenidos/distancia.htm

comparación de valores de distancias y obviar si se ha visitado una vez algún vértice debido a que los saltos ya están enlazados y lo que hace el nodo es armar dinámicamente *el camino más corto*[3] hasta su destino. Se lo indica a este proceso gráficamente en la Figura 2.6 y en forma de reglas en la Figura 2.7.

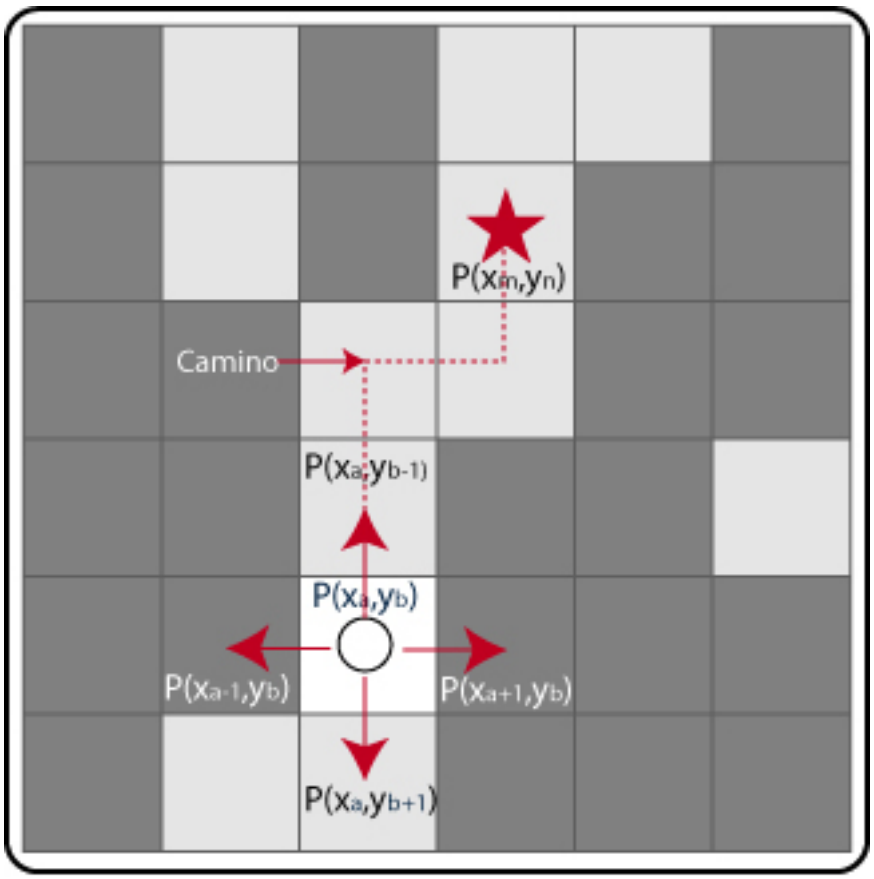


Figura 2.6 Movimiento del modelo *AVM*

3 http://bioinfo.uib.es/~joemiro/aenui/procJenui/ProcWeb/actas2001/saalg223.pdf

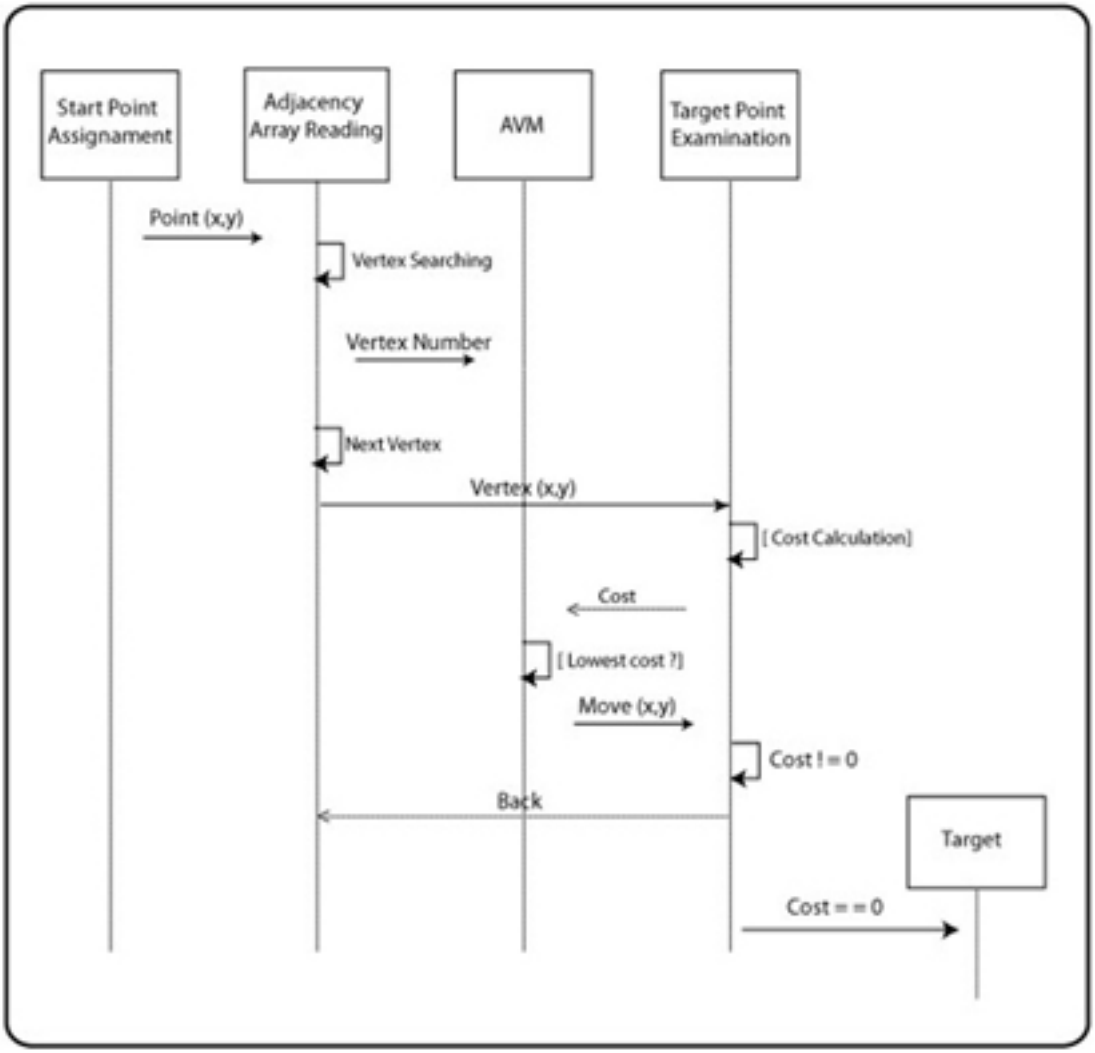


Figura 2.7 Ejecución del modelo AVM

A continuación se detalla en la Figura 2.8 el algoritmo en pseudocódigo del movimiento de los nodos implementado en la clase *Walkd*. En la Figura 2.9 se muestra una tarjeta *CRC* de la clase *Walkd* y en la Figura 2.10 el diagrama de objetos de la clase *Walkd*.

Algorithm 2 AVM node movement.

Require: An adjacency matrix

Ensure: Next nodes movement point (x,y).

1: **for** $(var2 \leftarrow 0, ..., num\_max\_nodes)$ **do**
2: **if** $((pX == max[var2][0]) \wedge (pY == max[var2][1]))$ **then**
3: $start\_node2 \leftarrow var2$
4: $var2 \leftarrow num\_max\_nodos$
5: **end if**
6: **if** $(pX <> max[target\_node][0]) \wedge (pX <> max[target\_node][1])$ **then**
7: **for** $(var32 \leftarrow 0, ..., num\_max\_colum[2])$ **do**
8: $a = max[start\_node2][var3] - max[target\_node][0]$
9: $b = max[start\_node2][var3 + 1] - max[target\_node][1]$
10: $cost \leftarrow \sqrt{[(a)^2 + (b)^2]}$
11: **if** $(var3 == 2))$ **then**
12: $mincost \leftarrow cost$
13: $pX \leftarrow max[start\_node2][var3]$
14: $pY \leftarrow max[start\_node2][var3 + 1]$
15: **else**
16: **if** $(cost < min\_cost)$ **then**
17: $min\_cost = cost$
18: $pX \leftarrow max[start\_node2][var3]$
19: $pY \leftarrow max[start\_node2][var3 + 1]$
20: **end if**
21: **end if**
22: **end for**
23: **end if**
24: **end for**

Figura 2.8 Algoritmo de movimiento de los nodos implementado en la clase *Walkd*.

Walkd	
Super Classes: Model	
Sub Classes:	
Description: Inicializa las variables de tiempo, velocidad y tipo de área. Convertir las cantidades a tipo entero. Seleccionar la arista de menor valor para realizar	
Attributes:	
Name	**Description**
filtro	Controla si es la primera vez que pasa por ese vértice.
nodo_fin	Coordenada (x,y) de la posición anterior
nodo_ini	Coordenada (x,y) de la posiciónsiguiente
Random_Pos_	Valor booleano para comprobar la aleatoriedad
Px	Coordenada *x* de la siguiente posición
Py	Coordenada *y* de la siguiente posición
T_dist	Tipo de distribución randómica para obtener el tiempo en la simulación
T_max_	Tiempo mínimo
T_min	Tiempo máximo
V_dist	Tipo de distribución randómica para determinar los valores de la velocidad en la simulación.
V_max_	Velocidad máxima
V_min_	Velocidad mínima
Responsabilities	**Colaboration**
Inicializar variables (init)	modelspec
Formatear cantidades para que sean operables	
Buscar la arista de menor peso	
Obtener la próxima posición del nodo (dato)	
Punto_evacuación	
Asignar la siguiente posición al nodo (MakeMove)	random

Figura 4.9 Tarjeta CRC de la clase *Walkd*.

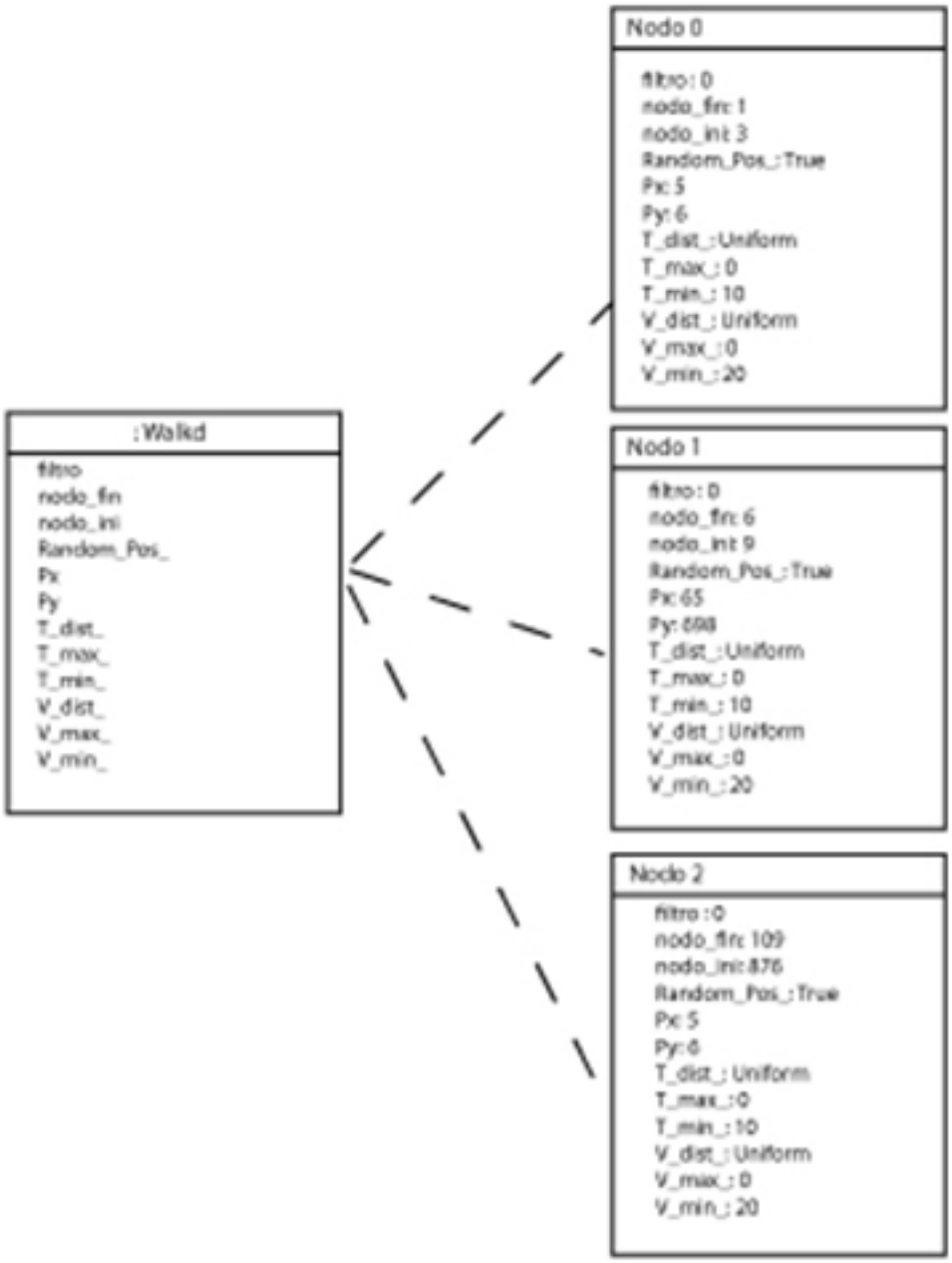


Figura 2.10 Diagrama de Objetos de la clase *Walkd*

CAPITULO 3. HERRAMIENTAS ANALIZADAS PARA LA IMPLEMENTACIÓN DE LOS MODELOS *UPM* Y *AVM*

3.1 Herramientas analizadas

Existen varias herramientas de simulación de redes *MANET*, las cuales pueden ser muy básicas o complejas de acuerdo a los requerimientos o al expertíz del mismo. Estas herramientas otorgan sin duda múltiples beneficios para el análisis del rendimiento de la red de comunicación. Entre las herramientas estudiadas tenemos:

Herramientas de simulación

Network Simulator 2 (*NS2*)

Es el ambiente de simulación de redes dirigido a eventos de código abierto siendo el más utilizado en ambientes académicos en todo el mundo. Es un producto creado por *US Defense Anvanced Research Projects Agency* (*DARPA*). Está desarrollado en *C++* y *TLC*. Se pueden simular protocolos unicast, multicast y varios protocolos propios de las redes *MANET.*[22]

NS3

Es la versión mejorada del *NS2* en la cual se eliminan los conflictos de interacción entre *C++* y *TLC*. Por lo que el software puede simular escenarios complejos cargados de una gran cantidad de nodos aproximadamente hasta unos 10 000 nodos móviles. Es catalogada como una herramienta moderna y que tiene mucho potencial. [25]

QualNet

Es una herramienta de simulación desarrollado por la *Universidad de California*, que empezó como una aplicación gratuita pero que en la actualidad es un producto comercial. Es un entorno capaz de simular una gran cantidad de nodos y ofrece una poderosa interfaz para el usuario [29]. Sus características son:

- Posee un conjunto potente de protocolos de red y modelos dispositivos para la simulación de redes alámbricas e inalámbricas.

• Optimiza la velocidad y escalabilidad del procesador.

• Es una herramienta diseñada para incrementar su velocidad de acuerdo a la cantidad de procesadores.

• La interfaz gráfica es robusta que permite generar animaciones y análisis estadísticos de una simulación.

• Ha sido comprobado su alta fidelidad de simulación para modelos de redes inalámbricas con una cantidad 50 000 nodos .

• Es una aplicación multiplataforma.

JiST/SWANS

Es un entorno de alto rendimiento de ingeniería de simulación de eventos discretos desarrollado en *Java* por la *Universidad de Cornell*. Es utilizado para realizar cualquier simulación aunque está específicamente diseñado para *MANETS*. Es un ambiente de alta escalabilidad y es capaz de simular escenarios hasta con 10 000 nodos móviles. Una de sus ventajas es que ayuda a evaluar programas desarrollados en *Java*[4] .

OMNeT++

Es un framework de simulación de redes de eventos modulares discretos orientados a objetos que ofrece una solución comprobada para:

• El modelamiento de redes alámbricas e inalámbricas.

• Modelamiento de protocolos

• Modelamiento de multiprocesadores y otros sistemas de hardware.

• Validación de arquitecturas de hardware.

• Evaluación de complejos sistemas de software.

OMNeT++ por sí solo no es un simulador pero es una aplicación que provee infraestructura y herramientas para realizar simulaciones[5] .

4 http://jist.ece.cornell.edu/

5 http://omnetpp.org/doc/omnetpp/manual/usman.html

Generadores de mobilidad

Boon Motion

Es un software desarrollado en *Java* por la *Universidad de Boon* siendo su objetivo crear y analizar escenarios de movilidad. Los escenarios pueden ser exportados para ser integrados a múltiples herramientas de simulación entre las cuales se encuentra *NS2*. Tiene implementado varios modelos de movilidad como: *Random waypoint*, *Brawniano,* etc. [23]

Ad hockey

Es un programa que permite crear y simular escenarios de forma interactiva. Esta desarrollado en *Perl/Tk* y la creación de escenarios se realiza con o sin obstáculos aunque sus movimientos solamente se aplica la movilidad Waypoint. Su distribución libre en la página del *proyecto Monarch*[6].

Scengen

Es una aplicación programada en *C++* cuyo único objetivo es crear escenarios para *NS2*. Su código es libre y puede ser fácilmente descargado de la página del autor. Tiene implementados varios modelos de movilidad. Su código viene empaquetado junto con un directorio que trae el código del programa de simulación *Ad hockey*[7].

MobiREAL

Es un simulador desarrollado por el *Instituto Tecnológico de Georgia* y está compuesto de dos independientes programas que se denominan: *MobiREAL Behavior* and *MobiREAL network simulator*. El primero simula el comportamiento de los nodos y el segundo se encarga de la simulación del tráfico. Al ser inicializadas ambas partes interactúan y generan la movilidad. También permite generar o suprimir nodos dinámicamente y manipular la velocidad, posición o dirección de los nodos móviles [24].

3.2 Elección de la herramienta para simulación y determinación del modelo de movili-

6 http ://www.monarch.cs.rice.edu

7 http://isis.poly.edu/~qiming/scengen/index.html

dad adecuado

De las herramientas anteriormente mencionadas, se ha seleccionado la herramienta *NS2* y *Ad hockey* para la validación visual del escenario generado; y, Sc*engen* para la implementación de los modelos propuestos. La selección se ha basado en:

1. Restricción: Ha sido un requerimiento del Laboratorio de Redes que la solución cumpla con los siguientes requerimientos: que sea desarrollado para una distribución del Sistema Operativo Linux y que el escenario de movilidad de nodos generado sea utilizable en *NS2*.

2. *Scengen* es una programa mono-propósito, es decir, tiene una sola función que es la de crear escenarios de movilidad. Al ser de este tipo se convertiría en un ambiente ideal para implementar un nuevo modelo ya que no posee una gran cantidad de código como las herramientas que brindan más servicios y de esta manera disminuir el tiempo de análisis de la herramienta. Además, cumple con todas la métricas acotadas en la parte superior lo que permite dar una integridad con el trabajo que realiza el grupo de docentes-investigadores del laboratorio.

3.2.1 Descripción de la herramienta Scengen

Scengen es una herramienta que genera escenarios de movilidad pre-formateados en NS2. Esta aplicación fue desarrollada por el *PHD Ing. Quiming Li* [8] en Singapur y su código se lo puede descargar en forma libre. El paquete descargable se encuentra integrado por los siguientes archivos y directorios dispuestos en la Figura 3.1:

8 http://isis.poly.edu/~qiming/

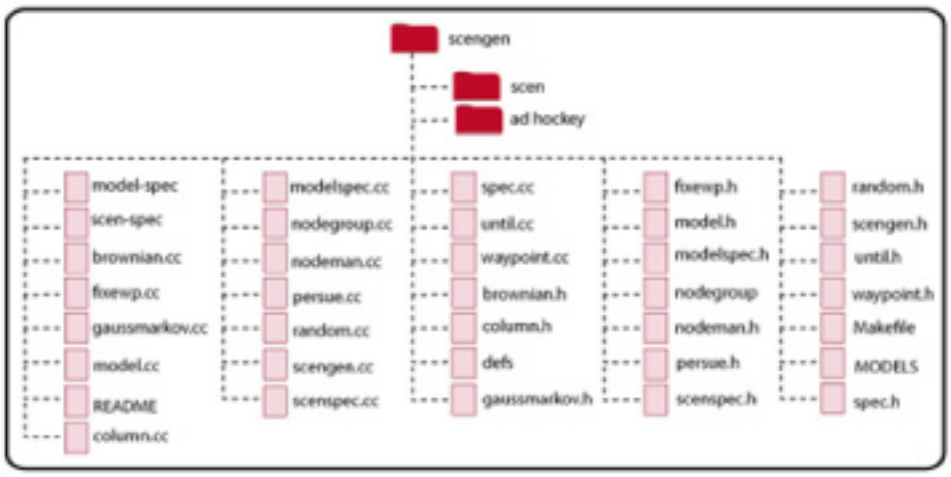


Figura 3.1 Archivos y directorios de Scengen

A continuación se detalla brevemente cada uno de los archivos y directorios que se aprecia en la imagen.

README: Es un documento que contiene la redacción de sencillas instrucciones sobre ciertos archivos y tópicos de la herramienta.

Makefile: Archivo que posee una serie de instrucciones que se activarán cuando se realice la compilación del archivo en el terminal.

MODELS: Instrucciones de cómo agregar un nuevo modelo a la aplicación.

defs: Archivo donde se encuentran definidas varias de las librerías que utilizan todos los archivos *.h* y *.cc*

until. h: Cabecera que define la *clase List* que define métodos de selección de caracteres de los archivos *scen-spec* y *model spec*. La variable *key* maneja la parte anterior al igual de las variables encontradas en esos archivos, mientras que la variable *value* almacena los valores numéricos.

until.cc: Almacena el desarrollo del código de los métodos definidos en la clase entre los cuales se encuentra el procedimiento que convierte los caracteres almacenados en la variable *value* a diferentes tipos de datos.

random.h: Cabecera donde se definen las constantes y las clases utilizadas para generar la números aleatorios.

random.cc: Métodos de generación de números aleatorios a partir de un valor numérico denominado semilla y distribuciones estadísticas como: Gauss, uniforme, etc.

scen-spec.h, scen-spec.cc, model-spec.h y model-spec.cc son los encargados de manipular y discernir cada uno de los caracteres, palabras que son leídos de los archivos *scen-spec* y *model-spec*. Además se encargan de almacenar los datos necesarios para enviarlos como parámetros a otros archivos de la herramienta y desechar los datos irrelevantes.

model.h: En este archivo se hallan las clases más usadas e importantes de la aplicación como: *Vector*, *Area* y *Model*.

model.cc: Se encarga de reconocer el tipo de área estipulada en el archivo *scen-spec* y señalar el procedimiento a seguirse. Aquí se toman los valores randómicos obtenidos del archivo *random.cc* y asignan dinámicamente como valores de los parámetros a usarse. Finalmente los envía dinámicamente a la clase *Scengen*.

nodeman.h : En este encabezado se encuentra establecida la clase nodeman que controla a los procedimiento de cada uno de los nodos.

nodeman.cc: Inicializa, elimina, administra y otras funciones para la gestión de nodos.

nodegroup.h: Cabecera donde se define la clase nodegroup y sus respectivos procedimientos aplicados a controlar grupos de nodos.

nodegroup.cc: Contiene el código de los métodos que permiten obtener los parámetros de los grupos de nodos y manipular por programación sus características.
scengen.h: Define a la clase principal del programa denominada *ScenGen*.
scengen.cc: Se concentran todos los procedimientos que tienen que ver con la captura y armado del pre-formato del escenario que posteriormente se van a visualizar en la pantalla.

Los archivos *.h* y *.cc* de los 6 modelos implementados: *Brownian*, *Column*, *FixedWaypoin*t,*Gauss-Markov*, *Persue*, y *Waypoint*. Son empleados en la herramienta para generar escenarios que reproducen movimientos bajo un área libre de obstáculos.

model-spec: Archivo donde se encuentran parametrizados todos los modelos que trae la herramienta.

scen-spec: Archivo donde se encuentra detallado los valores del área, tipo de área, el número de nodos, nombre del modelo movilidad que usarán los nodos, grupos de nodos, valores de ciertos parámetro, etc.

Scen: Es un directorio que almacena 3 ejemplos de escenarios de movilidad (*scen-conference*, *scen-convention* y *scen-disaster*) y sus respectivos archivos fuente (*scen-spec.conference*, *scen-spec.convention* y s*cen-spec.disaster*). El programa viene por defecto integrado con el archivo *scen-spec.disaster*.

***Ad hockey*:** Es el directorio que contiene el código del programa de simulación visual y creación de escenarios para observar de forma detallada esta herramienta ir a Anexo 2.

3.2.2 *Ad hockey* versus *Scengen*

Las aplicaciones *Ad hockey* y *Scengen* son herramientas que en ciertos puntos compiten porque ofrecen similares servicios y en otros se complementan. Por lo que para determinar las ventajas y desventajas que ofrece Ad-hockey frente Scengen se propone la Tabla 3.1:

Características	*Scengen*	*Ad hochey*
Obtención del programa	Internet	Internet
Documentación	Ninguna	Casi nula
Instalación	Fácil	Compleja
Programación	C++	Perl, C++
Interfaz gráfica	No	Si
Interactividad con el usuario	Poca	Mucha
Servicios	Uno	Varios
Creación de escenarios	Automática	Manual
Trabaja con obstáculos	No	Si
Uso de imágenes	No	Si
Visualización de resultados	Compleja	Fácil
Modelos de Movilidad	Varios	Uno

Tabla 3.1 Scengen vs. Ad hokey

Según este pequeño cuadro se podría decir que es más conveniente el uso de *Ad hockey* que el *Scengen*, pero el hecho que permita trabajar sobre un solo modelo de movilidad y que obligue al usuario a modificar o crear un nuevo escenario cada vez que quiera obtener otro escenario nos deja en claro que es una herramienta didáctica valiosa que está diseñada más para la visualización que para la creación de escenarios. Motivo por el cual se puede deducir que el creador del programa *Scengen* vio la conveniencia de anexarla en el paquete distribuido en el Internet.

3.2.3 Diagramas de la interacción de los archivos y directorios Pre-Durante y Post compilación del *Makefile*

En la Figura 3.2, muestra cuatro bloques que corresponden a la estructura interna de la herramienta *Scengen* antes ser compilado a través del archivo *Makefile*. El bloque inferior contiene a los 8 archivos de código fuente desarrollados en *C++* que no son modelos de movilidad pero que son esenciales para el funcionamiento de la herramienta ya que en ellos se encuentra el código fuente de la aplicación. El segundo bloque lo integran los 7 modelos de movilidad implementados. En el paquete superior se encuentra integrado por 2 directorios que son: *Ad hockey* y *scen.*, y 4 archivos de texto plano. El archivo *MODELS* proporciona los pasos que se debe realizar para agregar un nuevo modelo. El archivo *scen-spec*, es el lugar donde se encuentran definidos los parámetros que se utiliza para generar escenarios. Un respaldo de este archivo se encuentra el directorio *scen* con el nombre de *scen-spec.disaster*. El archivo

model-spec, contiene los parámetros de cada modelo implementado.

El directorio scen, tiene 3 archivos con diferentes tipos de parámetros cada uno orientados para diferentes entornos de simulación y 3 archivos más que son los escenarios generados por cada uno de estos archivos.
El directorio *ad hockey* tiene en su interior el programa de simulación visual *Ad hockey*. El archivo readme detalla unos cortas ideas sobre la herramienta. El bloque que dice *Makefile* representa el archivo make que contiene todas las sentencias para generar en línea de comandos los ejecutables para que funcione la herramienta. En este recuadro se encuentra todo el significado cromático empleado en cada cuadro y círculo que se coloca sobre los gráficos que representan los *archivos .cc*. Cada uno de los colores expresados representa una cabecera (*.h*). En cada archivo se colocan cuadros con un color respectivo que quiere representar la cabecera que está definida en el código. Los círculos encima del recuadro que representa la cabecera del mismo nombre de archivo .cc, simbolizan las cabeceras que contiene la cabecera en cuestión.

La Figura 3.3, recrea la interacción que tiene el archivo Makefile al momento de compilarse con el resto de archivos, directorios y el compilador g++. Se puede observar un recuadro puesto en la parte inferior que contiene todos los *archivos .cc* y *.hh* que trae la herramienta. Los cuales son imprescindibles para que el compilador genere los ejecutables encerrados en el recuadro superior al recuadro del compilador *g++*. Estos archivos ejecutables los maneja el comando make porque se encuentran definidos dentro de la estructura de las instrucciones del *Makefile* para generar un ejecutable de la aplicación. Los tres archivos que se encuentran en un recuadro aparte no juegan ningún papel importante en la compilación, son archivos que contienen información sobre la aplicación.

En la Figura 3.4, se puede observar cómo queda la interacción de los archivos y directorios después de la ejecución del comando *make* que permite elaborar el ejecutable de la aplicación *Scengen* (./scengen). Este va trabajar de ahora en adelante solo con los archivos ejecutables generados en la complicación y no con los *archivos fuentes*.

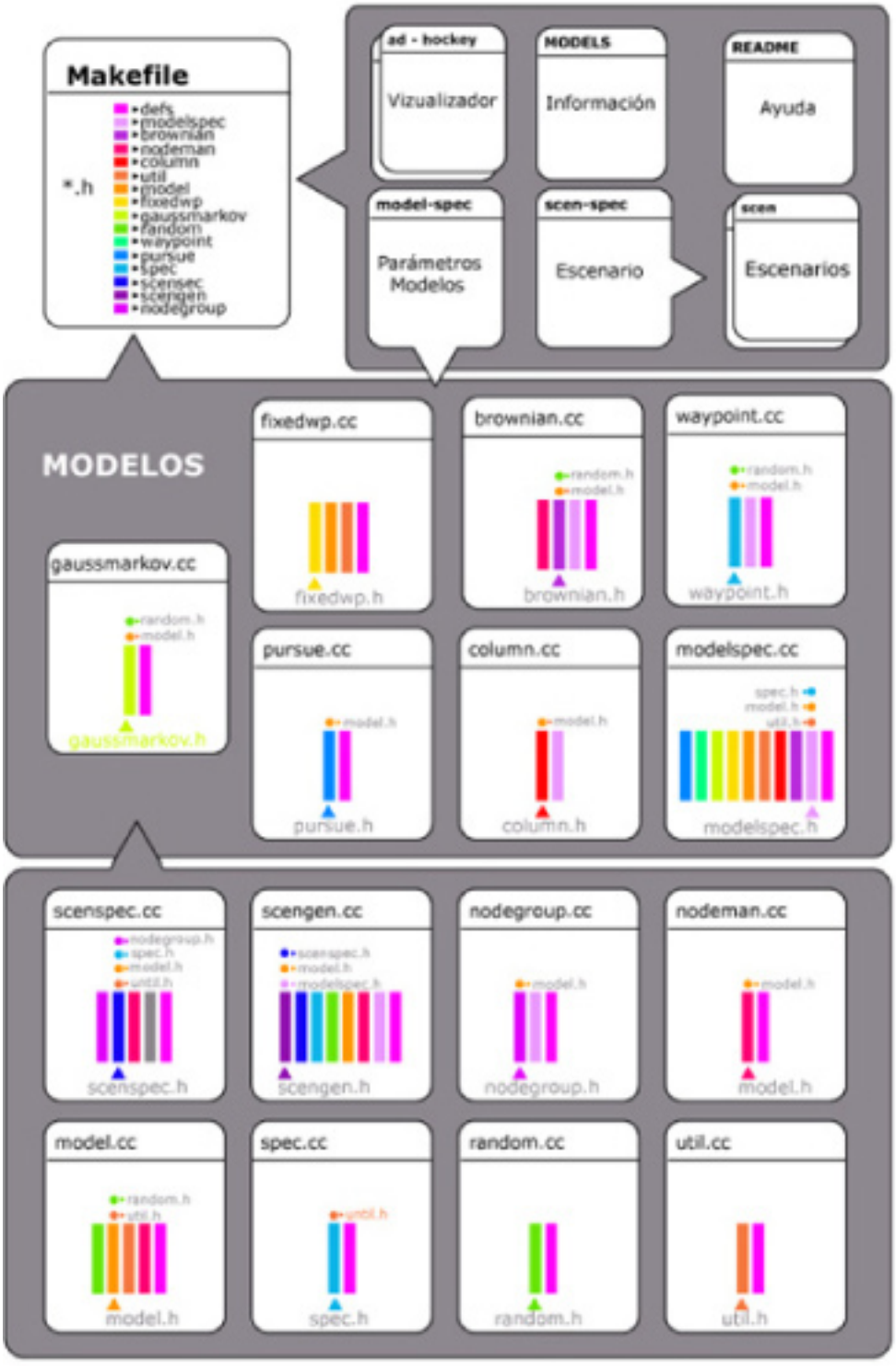


Figura 3.2 Pre-compilación

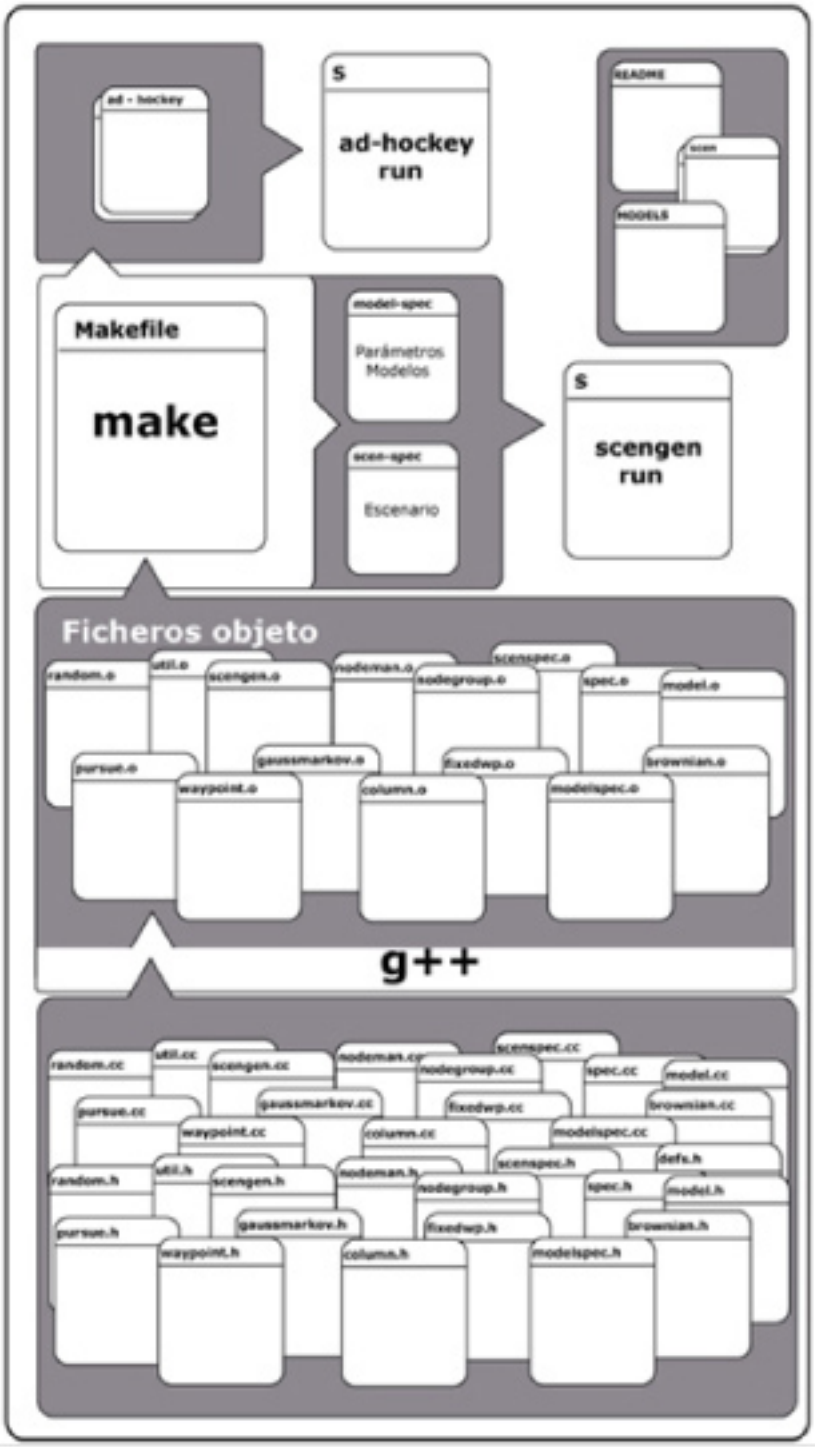


Figura 3.3 Compilación

Figura 3.4 Post-compilación

3.2.4 Diagrama de Clases

En la Figura 3.5, se puede contemplar el diagrama de clases de la aplicación *Scengen*. En él se pueden observar que está conformado por más 20 clases. Los modelos implementados en la herramienta se encuentran estrechamente relacionados con la clase *Model* debido a que es la clase padre de cada clase que se debe de crear para cada modelo. Las clases: *ModelSpec* y *ScenSpec*, se encargan de leer y evaluar cada palabra o caracter de los archivos model-spec y scen-spec respectivamente, para obtener las palabras claves (*key*) y sus valores (*value*) que son los parámetros de entrada para el funcionamiento de los modelos y generación de los escenarios. Otras clases importantes son las clases: *Area*, *Move*, *Vector*, *Node* y *Node Group*; que rigen el comportamiento de cada modelo. Las clases *AreaType* y *ModelType,* contienen el conjunto de modelos y áreas que trabaja la herramienta. Los atributos de estas clases son obtenidos mediante herencia en las clases *Area* y *Model* respectivamente. La clase principal de la aplicación se denomina *ScenGen*.

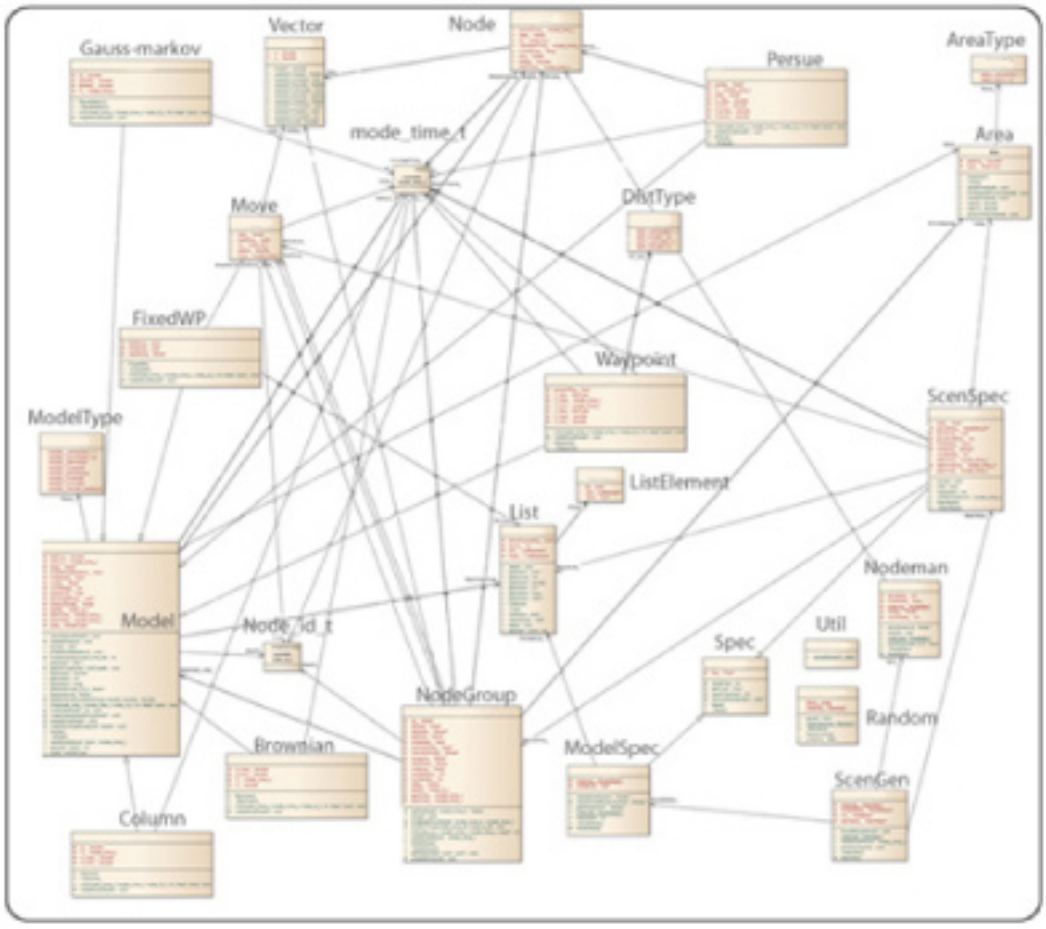


Figura 3.5 Diagrama de clases de la estructura *Scengen*.

PARTE III
IMPLEMENTACIÓN Y VALIDACIÓN

CAPITULO 4. IMPLEMENTACIÓN DE LOS MODELOS DE MOVILIDAD *UPM* Y *AVM*

Para comprobar la eficiencia de los modelos propuestos fue necesario implementarlos en la herramienta de simulación y evaluar su comportamiento. Para el efecto, se implementan los modelos *UPM* y *AVM* en la herramienta *Scengen*.

4.1 Datos importantes para implementación en la herramienta.

En la tabla 4.1, se lista una serie de características relevantes con sus respectivos valores que son de utilidad para la implementación de los modelos.

Característica	Valores
Nombre de la herramienta a modificarse	Scengen
Lenguaje de programación necesario	C++
Modelos implementados	FixedWP, Brownian, Column, Persue, Waypoint y Gauss-Markov
Cuenta conmodelos basados en obstáculos	No
Forma de áreas permitidas	Rectangular y circular
Forma de generación de los valores	Aleatoria
Modelos de distribución implementados	Uniforme y Gauss
Nombres de los modelos a implementarse	*UPM* y *AVM*
Tipo de modelos a implementarse	Modelos de restricción geográfica
Los modelos a implementarse manejan obstáculos	Si
Estrategia de movilidad del modelo UPM	Movimientos geográficos empleando la matriz binaria del mapa.
Estrategia de movilidad del modelo AVM	Aplicar el algoritmo de camino más corto utilizando una matriz de adyacencia de los vértices del mapa
Obtención de la matiz binaria	Conversión automática de código .XBM
Obtención de la matriz de adyacencia	Manual de cada vértice de los obstáculos
Area implementarse	Segmento de mapa de la ciudad de Loja
Formato de la imagen del mapa	.XBM
Dimensiones del mapa	2km x 1km
Forma del área	Rectangular
Resolución de la imagen	2000px x 1000px
Colores	Blanco y negro
Parámetros del modelo	Velocidad, pausa
Generación de área de desastre	Automática
Herramientas de Simulación	*NS2* y *Ad hockey*

Protocolo a usar para la simulación NS2	*AODV*
Tiempo total	150 segundos
Número de nodos ha probar	20, 40 , 60, 90
Computador	Compaq Presario C700,
Sistema Operativo	Ubuntu Natty

Tabla 4.1. Datos claves de la implementación

4.2 Selección de un segmento de una urbe real.

Para la determinación del área sobre la cual se a validar los modelos *UPM* y *AVM* se planteó el uso de un segmento de un plano de la ciudad de Loja-Ecuador. La dimensión escogida es de 2 Km x 1 Km porque se tendrá un espacio suficiente para lograr evaluar la movilidad de los nodos como para generar el mapa de desastre. Cómo la calle 10 de Agosto mide aproximadamente 2 kilómetros, se la usa como módulo para obtener el segmento de tales dimensiones sin permitir que dentro del segmento queden cuadras incompletas. La zona establecida va desde Norte hasta la calle Quito y al Sur hasta la calle Catacocha. Al Occidente hasta la calle 18 de Noviembre y al oriente va hasta la calle Juan José Peña. Esto se lo detalla en la Figura 4.1.

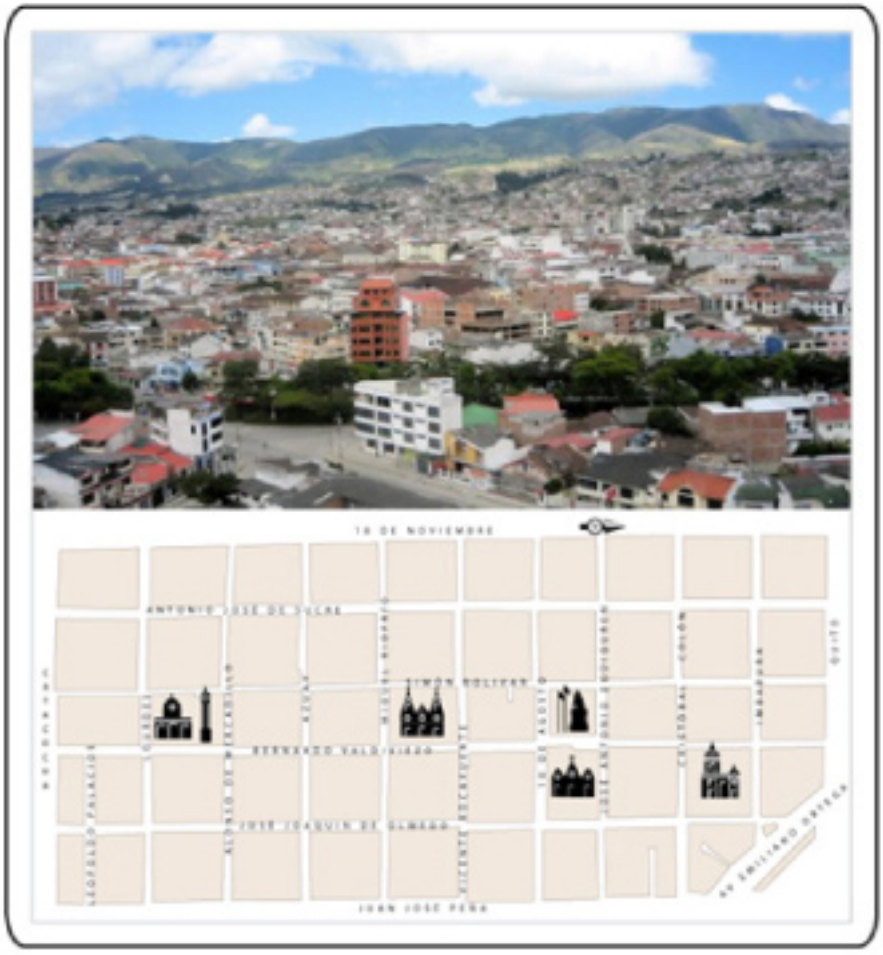


Figura 4.1 Segmento 2 Km x 1 Km de la ciudad de Loja

4.2.1 Edición, formateo gráfico y simulación del área de desastre

En base a un plano real de la urbe del año 2011 proporcionado por el *Ilustre Municipio del Cantón Loja* se procede a seleccionar el área idónea y se comienza a editar la imagen, eliminando detalles arquitectónicos dispuestos en dicho recurso gráfico de tal forma que se abstraiga de todo lo innecesario y queden solo bloques de las estructuras como se lo indica en la Figura 4.2. La imagen se encuentra en formato *jpg* , siendo posteriormente cambiado al formato de mapa de bits monocromático denominado *Exchangable BitMAP* (*XBM)* por la facilidad que presenta su composición interna en forma de matriz binaria que representa el escenario. Para conocer más profundamente sobre la composición interna de este tipo de herramientas ir al Anexo 3.

Figura 4.2 Obtención de la imagen .XBM

El código interno de la imagen es transformado mediante un programa desarrollado en C++ a datos binarios que van a ayudar en el control de la movilidad del modelo *UPM*. Observe la Figura 4.3 y revise el Anexo 3 para profundizar sobre el cambio de la codificación de la imagen.

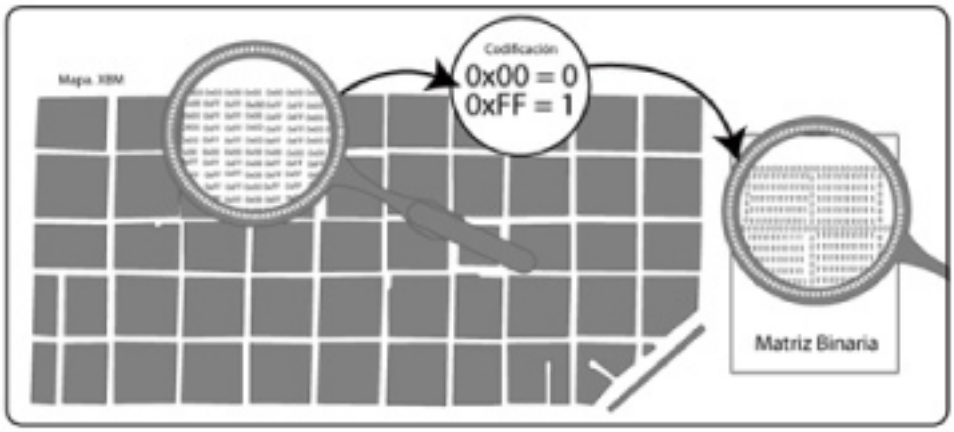


Figura 4.3 Representación binaria

Manipulando la matriz binaria obtenida anteriormente y mediante el desarrollo de un programa en C++ se generan obstáculos aleatoriamente como lo puede apreciar en la Figura 4.4.

Figura 4.4 Área de desastre

Bajo esta imagen obtenemos más de 30 obstáculos y alrededor de 234 vértices, como se lo aprecia en la Figura 4.5.

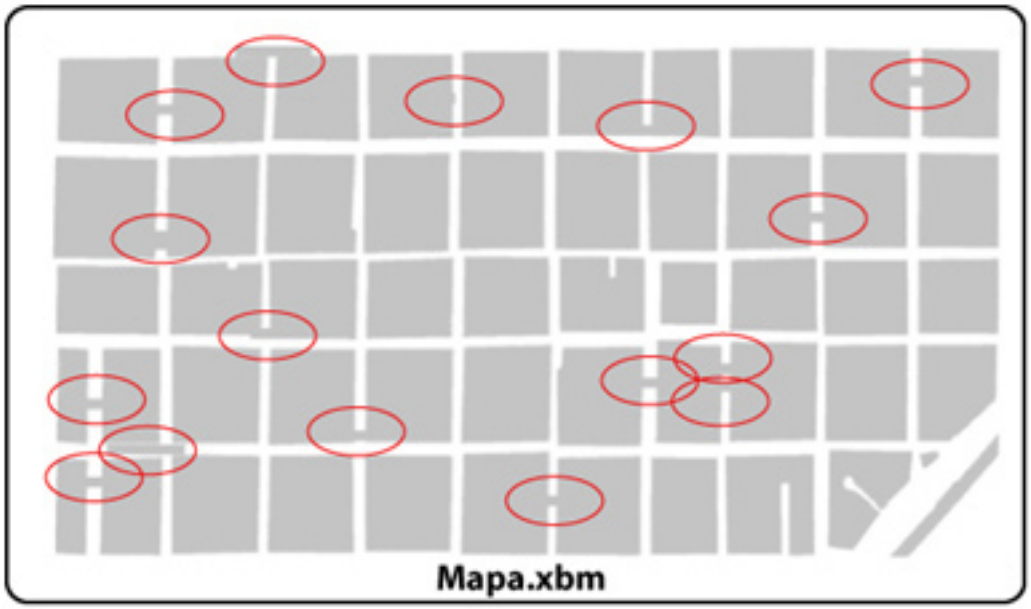


Figura 4.5 Obstáculos generados

Posteriormente, mediante la observación de la imagen resultante, se genera la matriz de adyacencia para poder determinar los posibles caminos y calcular los respectivos pesos que son utilizados por el modelo *AVM*. En la Figura 4.6 se muestra los vértices adyacentes para el punto 141.

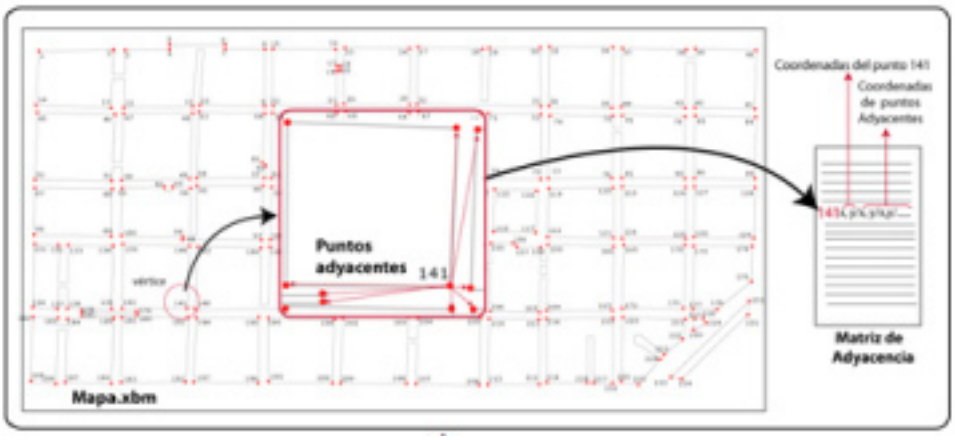


Figura 4.6 Generación de la Matriz de Adyacencia

El formato como se encuentra establecido los vértices la matriz de adyacencia es el siguiente:

$$P_{1x},P_{1y/}\,P_{sigx},P_{sigy}\,/\,P_{sigx},P_{sigy}\,/\,P_{sigx},P_{sigy}\,/\ldots\ldots\ldots\ldots/\,P_{nx},P_{ny}$$

$$P_{2x},P_{2y}/\,P_{sigx},P_{sigy}\,/\,P_{sigx},P_{sigy}\,/\,P_{sigx},P_{sigy}\,/\ldots\ldots\ldots\ldots/\,P_{nx},P_{ny}$$

……

4.3 Diagrama de clases para el modelo *UPM*

El la Figura 4.7, se puede observar el diagrama de clases de la herramienta *Scengen* en donde ya se encuentran implementados ambos modelos. La clase *Model* se constituye en una super clase para todos los modelos de la cual se hereda sus atributos.

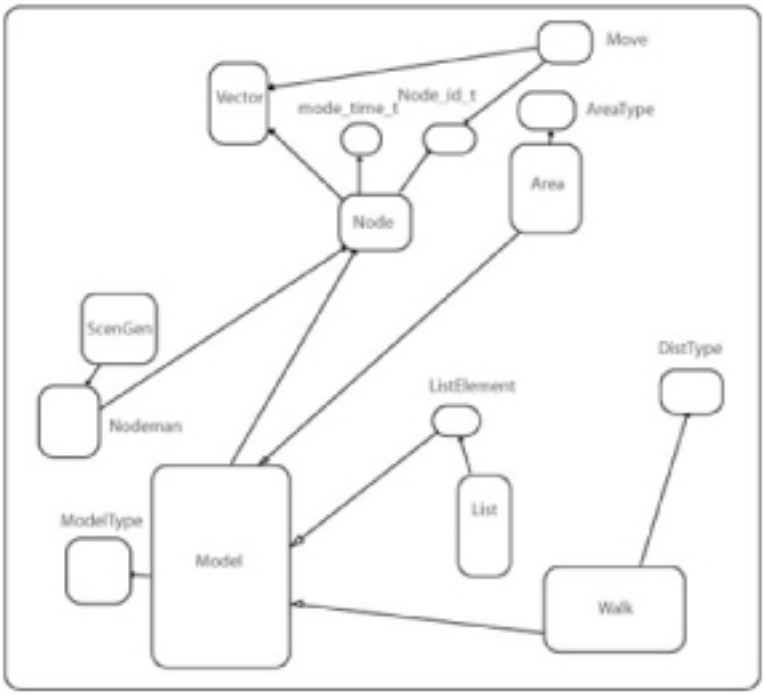


Figura 4.7 Diagrama de clases del modelo de movilidad *UPM* implementado bajo la clase *Walk*.

Para implementar el modelo de movilidad *UPM*, se debe de realizar una definición en la clase *Random* donde se inserta un método tipo flotante denominado *pixel()* en el cual se generan y seleccionan números enteros de forma aleatoria almacenados en dos variables tipo integer. Los valores generados para la primera variable van a fluctuar de 0 hasta el largo de la *matriz binaria* y la segunda de 0 hasta el ancho de la *matriz binaria*.

var1 = selec_aleatorio(0 hasta largo_matriz_bin);
var2 = selec_aleatorio(0 hasta ancho_matriz_bin);

Ambas variables se constituyen en un punto *P(var1,var2)* de la matriz binaria, que luego se verificará que si esta posición está *libre(0)* o si está *ocupada (1)*. En caso de que este libre se retorna la posición para ser posteriormente usada. Caso contrario se la desecha y se repite el proceso anterior.

```
Si (p[var1][var2]))== '0'
    return punto[var1][var2]
Caso contrario
    generar var1, var2
```

En la clase *Area* se define el método *randomPos(vector)*.Primeramente se verifica si es una área rectangular y se ha definido el modelo *UPM* en el archivo scen-spec (member_model = *UPM*). Si la respuesta es afirmativa, se llama al método *pixel()* de donde se recibe el *punto(var1,var2)* y se asigna a cada coordenada del vector una coordenada del punto respectivamente, como a continuación se lo señala:

vector_x= var1

vector_y= var2

Si la respuesta es negativa, se aplica otro tipo de generación randómica y otro modelo de movilidad.

Clase Walk:

La clase *Walk*, se trabaja con un método denominado *makeMove(nodo)*. En este método recibe como parámetro a nodo, que es una variable tipo estructura donde se almacenan el punto localización actual y el destino del nodo móvil. Cada nodo móvil, inicia en el punto establecido por las coordenadas del vector obtenidas previamente, por lo se realiza lo siguiente:

punto_actual_x= vector_x

punto_actual_y= vector_y

Para obtener la posición destino se llama desde este método al método datos(), cual toma las coordenadas del punto actual y evalúa la disponibilidad de sus 4 posiciones cercanas según lo define el modelo *UPM*. Si es hallada por lo menos una posición libre se decide la posición destino haciendo una selección de acuerdo a la proximidad de la posición actual con la posición del punto de evacuación mediante la elección de un criterio búsqueda por fila o columna. El punto de evacuación está dispuesto en forma estática dentro del programa. Los valores de las coordenadas encontrados son asignados de la forma siguiente:

punto_destino_x = punto_encontrado_x

punto_destino_y = punto_encontrado_y

El proceso anterior de encontrar el punto de destino se repite hasta terminar el tiempo dispuesto en el archivo *scen-spec (stop_time)*. Es conveniente señalar que cuando se inicia nuevamente este proceso se asigna las coordenadas del punto de destino anterior en las del punto actual, así:

punto_actual_x= vector_x

punto_actual_x= punto_destino_x

punto_actual_y= punto_destino_y

4.4 Diagrama de clases para el modelo *AVM*

El diagrama de clases, ver Figura 4.8, para el modelo *AVM* es similar al modelo *UPM*.

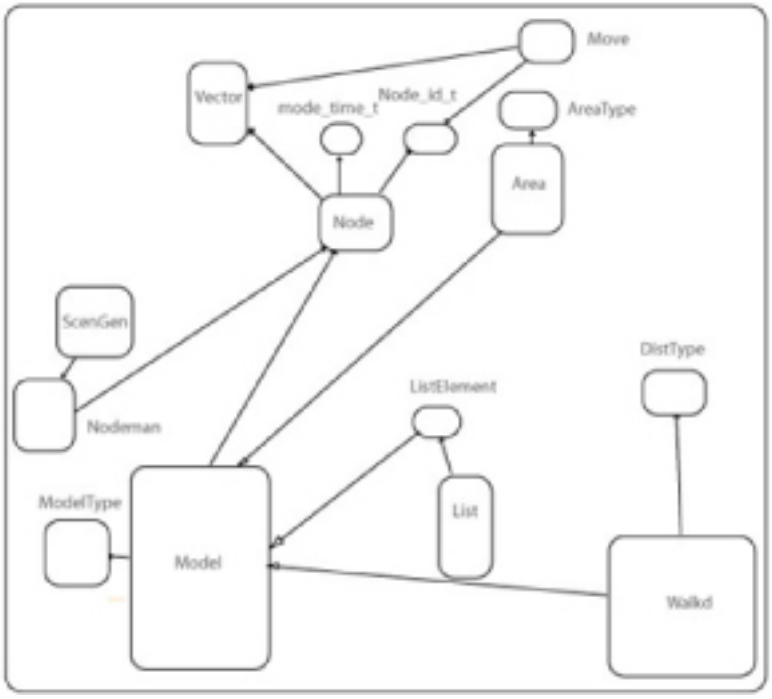


Figura 4.8 Diagrama de clases del modelo de movilidad *AVM* implementado bajo la clase *Walkd*.

El modelo *AVM* adopta los números generados aleatoriamente por la herramienta sin crear uno nuevo que fue imprescindible para el modelo anterior. En este caso, el modelo, toma únicamente la parte entera del vector y luego, buscar el punto cercano que se encuentre en la matriz de adyacencia. El punto encontrado se convierte en la posición actual del nodo móvil.

punto_actual_x= parte entera (vector_x)

punto_actual_y= parte entera (vector_y)

A diferencia con el proceso del modelo *UPM*, los valores de los vértices del mapa son almacenados como tipo de dato entero en una matriz de adyacencia de vértices. Para el efecto se utiliza el procedimiento *numerar()* que se encuentra bajo el *método datos()* en la clase *Walkd*. Las coordenadas de posicionamiento del vértice destino están ingresadas estáticamente en el procedimiento *nodo_dest()*.

De acuerdo a los valores del punto actual que dan la ubicación del nodo móvil, se procede a leer la línea del vértice que coincide con este valor. Para el efecto se inicia aplicando la fórmula de distancia euclidiana entre los valores del vértice y el primer par de valores después del carácter " / " que indica que es un vértice vecino. Si es el primer cálculo se debe de guardar su resultado en la variable y se almacena el par de valores del vértice vecino como las coordenadas del punto de destino.

Costo= Raíz_Cuadrada[Exp_Cuadrado(punto_actual_x-vertice_vecino_x)+
Exp_Cuadrado(punto_actual_y-vertice_vecino_y)]
punto_destino_x=punto_encontrado_x
punto_destino_y=punto_encontrado_y

Este procedimiento se repite con el siguiente par de valores de la lista cuyo resultado se compara con el anterior. Si es menor, se asigna a la variable caso y se actualiza los valores del punto de destino contrario no. Esto se repite hasta el último vértice vecino puesto en la línea.

Si Costo< Costo_calculado
Costo= Costo_calculado
punto_destino_x=punto_encontrado_x
punto_destino_y=punto_encontrado_y
Caso contrario
Ir hacia el siguiente vértice vecino

Cuando se ha terminado de analizar todos los vértices vecinos de la línea se procede a buscar el punto de destino en la matriz de adyacencia. Se repite todo el proceso con la siguiente línea encontrada hasta llegar al punto de evacuación.

A continuación, se presenta el diagrama de clases de la herramienta Scengen con los dos modelos ya implementados en la Figura 4.9. Las dos clases de los modelos, *Walk* (*UPM*) y *Walkd* (*AVM*), forman parte de la estructura de la aplicación al igual que el resto de modelos que vienen implementados con la herramienta como lo son: *Waypoint*, *FixedWP*, *Gauss-markov*, *Persue* y *Column*. Además, se puede observar que está implementado también el programa de creación de mapa de desastre siendo factible vincularlo mediante programación a los modelos creados o a otros modelos y generar el área de simulación de forma gráfica y dinámica.

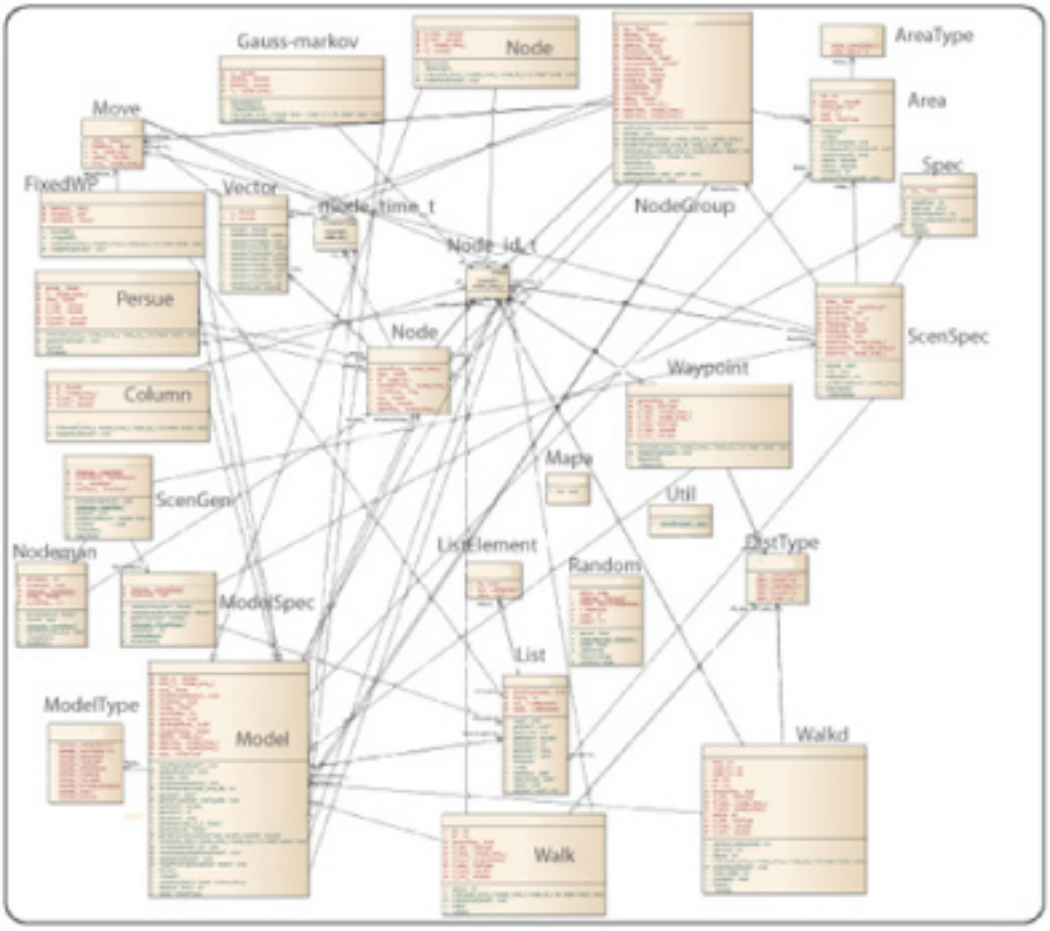


Figura 4.9 Diagrama de clases de la herramienta Scengen con los modelos: *UPM, AVM*.

CAPITULO 5. DISCUSIÓN DE RESULTADOS

Los dos modelos implementados se los sometió a tres tipos de experiencias para conocer su eficiencia y rendimiento de manera que sea factible realizar una comparación entre dichos modelos, que son:

- Eficiencia computacional
- Eficiencia empleando el protocolo *AODV*
- Validación visual

Eficiencia computacional

La medición de la eficiencia de los dos modelos propuestos se ha basado en tres parámetros: tiempo que demora en la generación del escenario, el porcentaje de uso de recursos del computador por cada escenario y el número de líneas de código de cada archivo generado. Los resultados obtenidos se muestran en la Tabla 5.1 en el caso *UPM* y en la Tabla 5.2 para *AVM*. Esta evaluación se la desarrolla para la cantidad de 20, 40, 60, 90 nodos, en un período de tiempo total de 150 segundos. El desplazamiento se realiza sobre un mapa de una ciudad real de 2 Km x 1 Km con obstáculos. El área simula una zona de desastre. Los parámetros del *Scengen* aplicados para ambos modelos(*UPM*, *AVM*): velocidad mínima/máxima, tiempo mínimo/máximo y pausa mínima/máxima.

El computador en el cual se trabaja posee un procesador Intel ® Celeron ® 550 @ 2.00 GHZ 2.00 HZ, y 991,7 MiB de memoria. La implementación se la hace sobre el sistema operativo Ubuntu Natty (11.04) y utilizando el compilador gcc 4.5.2.

De acuerdo a los parámetros de eficiencia anteriormente determinado se obtiene los siguientes resultados Tabla 5.1 y Tabla 5.2:

Número de nodos	Tiempo generación del escenario (s)	Uso de los recursos del CPU (%)	Número de líneas del escenario
20	7.261	100	440
40	13.476	100	904
60	19.698	100	1361
90	28.225	100	2032

Tabla 5.1 Datos obtenidos con el modelo *UPM*

Número de nodos	Tiempo generación del escenario (s)	Uso de los recursos del CPU(%)	Número de líneas del escenario
20	0.137	25	203
40	0.287	41	419
60	0.459	62	698
90	0.568	80	920

Tabla 5.2 Datos obtenidos con el modelo *AVM*

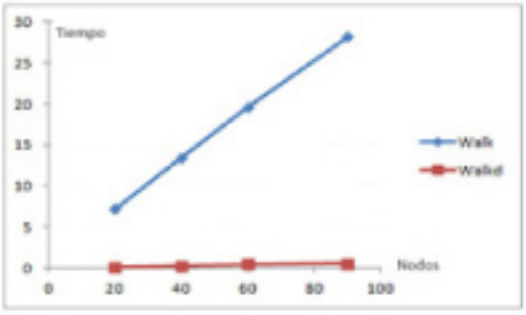


Figura 5.1 Consumo de tiempo de ejecución

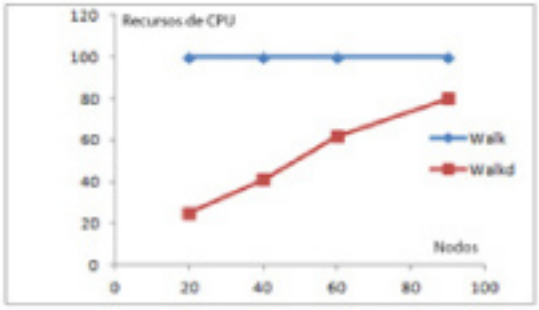


Figura 5.2 Consumo de recursos *CPU*

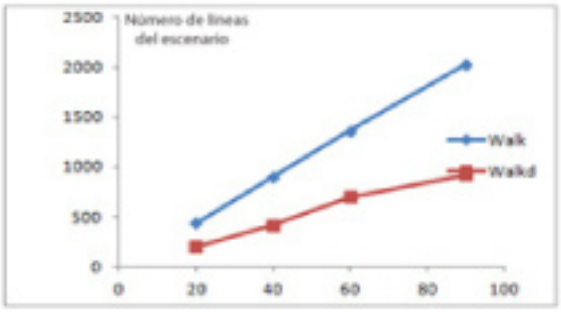


Figura 5.3 Número de líneas de generadas

En la Figura 5.1, se puede observar una marcada diferencia entre las gráficas de ambos mo-

delos. El elevado consumo de tiempo del modelo *UPM* en cada ejecución tiende a incrementarse de forma sostenida en relación con el incremento de número de nodos, mientras que en el modelo *AVM* se incrementa mínimamente y en comparación con el otro modelo el uso del tiempo es mínimo.

En la Figura 5.2, se indica el consumo voraz de recursos de *CPU* por parte del modelo *UPM* que tiende a ser constante sin importar el aumento o disminución de nodos a diferencia del modelo *AVM*, que el aumento de la utilización de recursos de la máquina va ligado al aumento del número de nodos.

Finalmente, en la Figura 5.3 se aprecia que el modelo *UPM* genera una mayor cantidad de líneas de código que el modelo *AVM* que afecta directamente a las simulaciones realizadas en el simulador *NS2*. Sin embargo al ser tan minuciosa la movilidad que ofrece el modelo *UPM*, es útil cuando los obstáculos carecen de una forma geométrica definida.

Eficiencia empleando el protocolo *AODV*
Con los mismos escenarios de movilidad cuando el nodo 0 se mueve (usados anteriormente) y también se generó unos nuevos escenarios haciendo que el nodo 0 no se moviera. En ambos casos se persigue descubrir la eficiencia de cada modelo aplicando el protocolo *AODV*.

Para medir la eficiencia utilizando el protocolo *AODV* se utilizó los siguientes escenarios:

- Nodo 0 que se mueve (Tabla 5.1 y 5.2)
- Nodo 0 sin movimiento (Se generó nuevos escenarios).

En la Figura 5.4, se puede apreciar que en las dos experiencias que el modelo *AVM* es un movimiento más eficiente hasta llegar a los 60 nodos donde comienza alejarse de los valores cercanos a 1. En cambio *UPM*, que parte de un valor distante del valor de 1 pero que a lo largo de su movimiento tiende a tatar de alcanzar valores cercanos a 1 siendo en los 60 nodos donde se precipita hasta alcanzar valores más próximos a este.

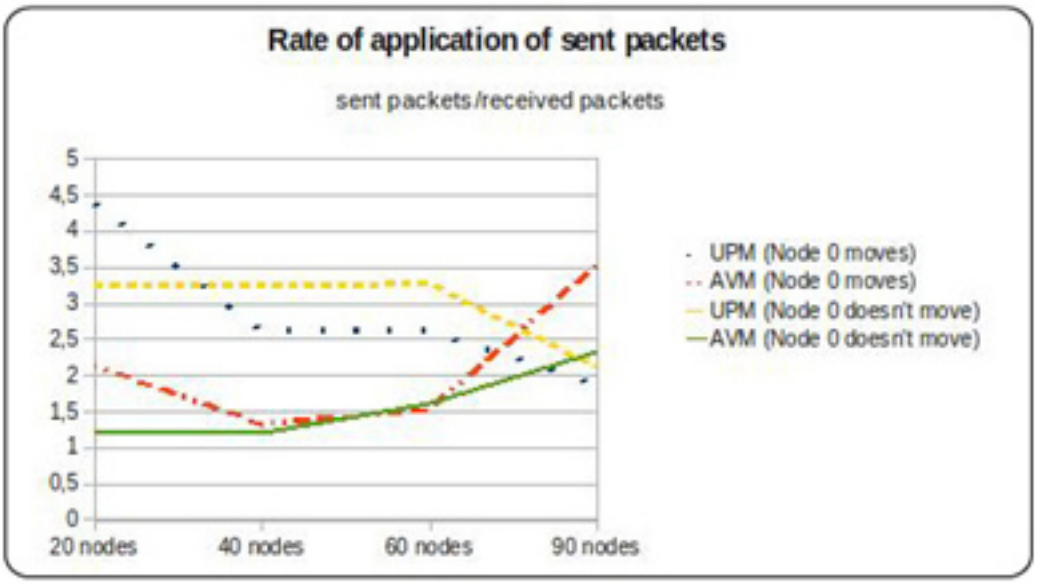


Figura 5.4 Tasa de paquetes enviados en la capa de aplicación

La Figura 5.5 indica que *UPM,* es un modelo que presenta mayor beneficio frente al *AVM*. Debido a que, *UPM* se mantiene una mayor cantidad de tiempo dando valores relativamente constante y más apegado a valores cercanos a 1; a diferencia de *AVM* que se aleja sustancialmente.

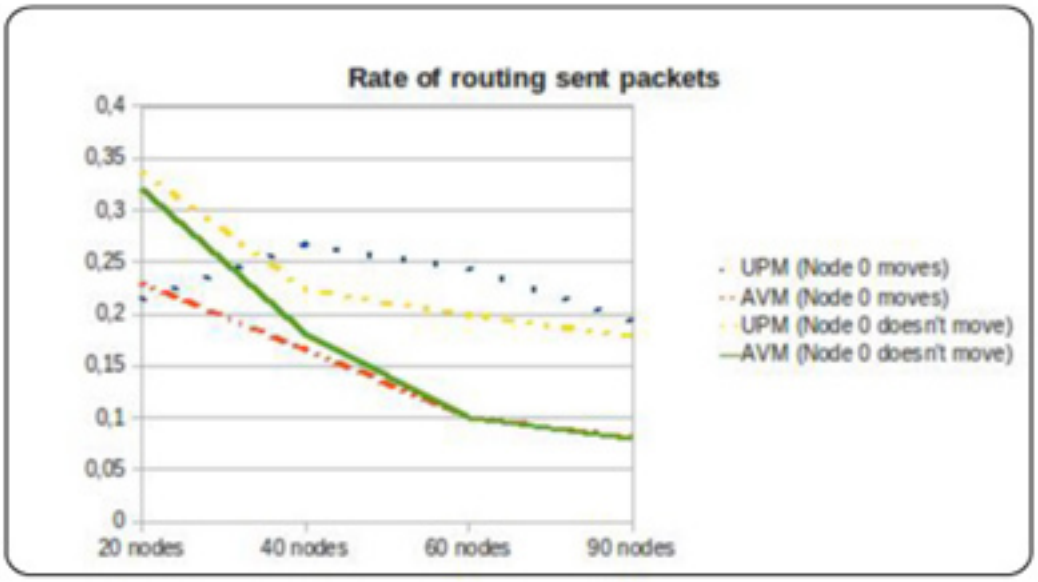


Figura 5.5 Tasa de paquetes enviados en la capa de red

En la Figura 5.6, se observa que el modelo *AVM* maneja mejores valores de retardo que *UPM*, porque *UPM* tiende a permanecer con un retardo de 0 lo que significa que el 100%

de los paquetes llegan a su destino ó que no se reciben las respuestas a los paquetes enviados. Mientras AVM se mantiene antes de llegar 60 nodos pero al incrementarse el número de nodos tiende a precipitarse.

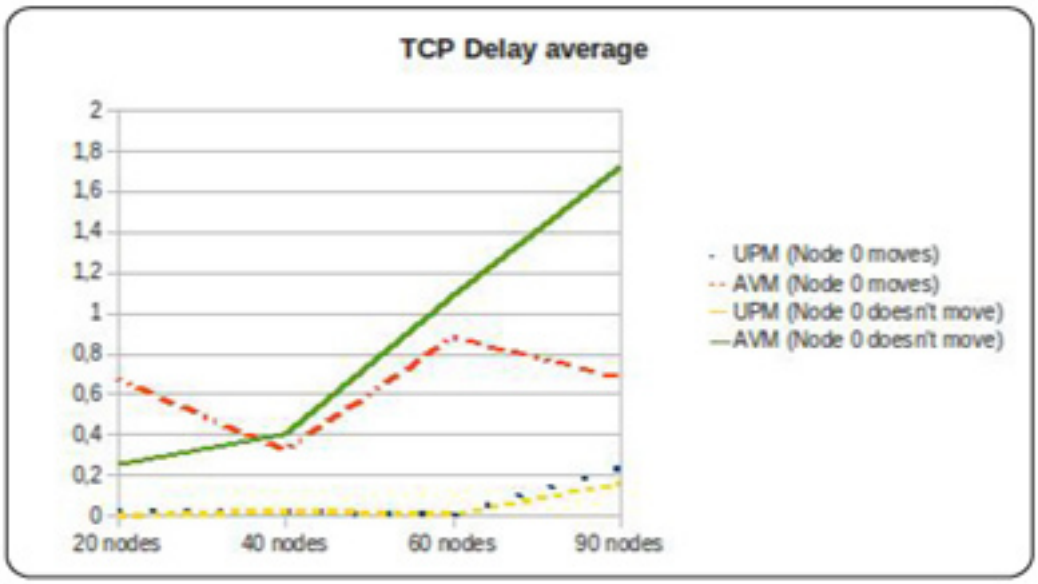


Figura 5.6 Promedio de retardp TCP

La Figura 5.7 muestra que el modelo *UPM* tiene un nivel relativamente constante de paquetes borrados. *AVM* borra menos paquetes, aunque al llegar a los 90 nodos tiende a dispararse la cantidad de paquetes borrados.

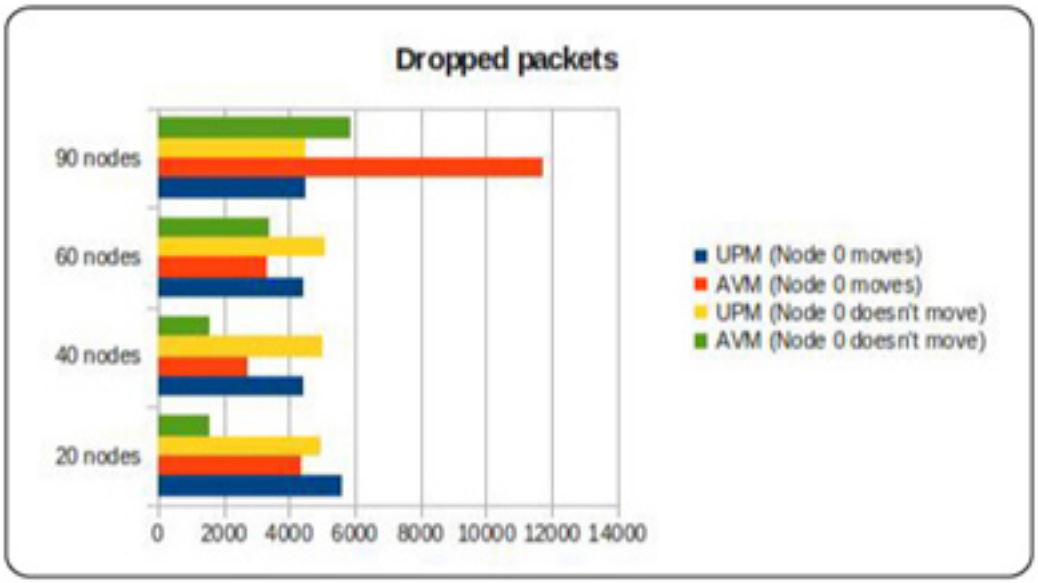


Figura 5.7 Paquetes borrados

Validación Visual

Todos los escenarios obtenidos se los validó visualmente en los simuladores *Ad hockey* y *NS2* para poder observar de una manera gráfica y dinámica la movilidad. En la Figura 5.8, se puede ver el comportamiento que tienen los nodos al ejecutarse el escenario de movilidad creado bajo el modelo *UPM* en la herramienta *Ad hockey* y en la Figura 5.9 se observa el resultado de la ejecución en la herramienta de visualización nam de *NS2*, del escenario de movilidad según el modelo *AVM*. Se puede notar las diferencias visuales de ambas aplicaciones aunque poseen el mismo cometido nam es superior a *Ad hockey* porque ofrece más detalle visual en su ejecución.

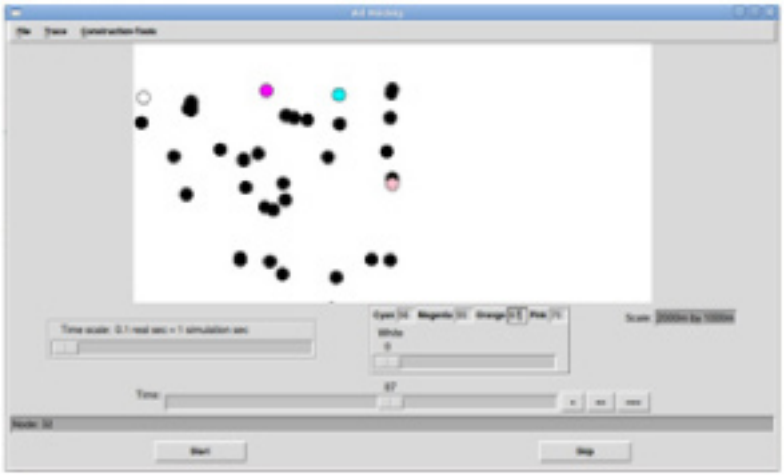


Figura 5.8 Ejecución del modelo *UPM* en la herramienta *Ad hockey*

Figura 5.9 Ejecución del modelo *AVM* en la herramienta *Ad hockey*

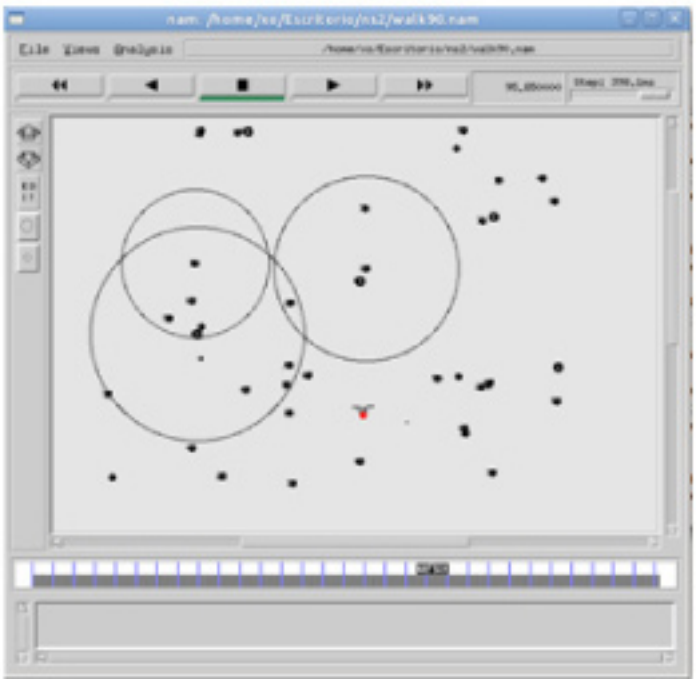


Figura 5.10 Ejecución del modelo *UPM* en *NS2*

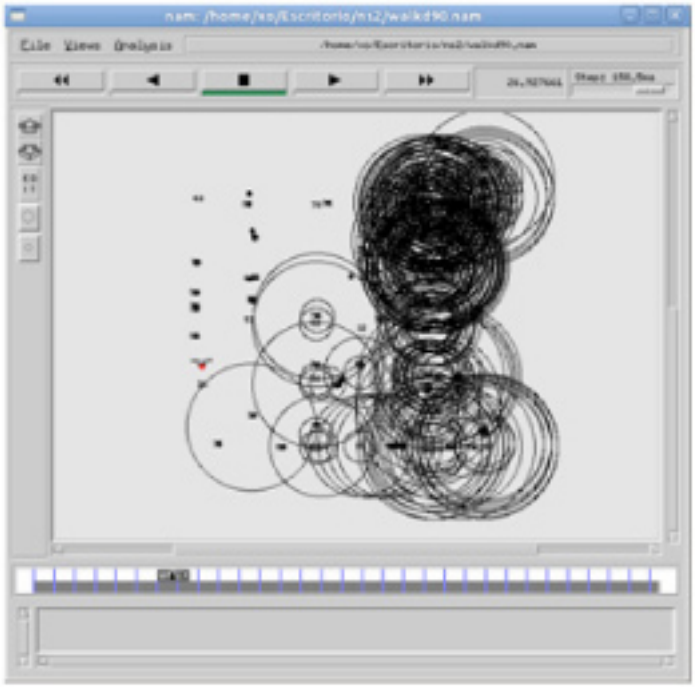


Figura 5.11 Ejecución del modelo *AVM* en *NS2*

Aportaciones y líneas futuras de trabajo

Aportes

Para iniciar el trabajo y poder alcanzar el objetivo principal de la tesis, se comenzó con un análisis del código y de la estructura de la herramienta Scengen para poder entender su funcionamiento, determinar los elementos que intervienen en su procesamiento y los resultados que genera. Además, permitió conocer que la herramienta crea escenarios de movilidad para áreas libres de obstáculos. Es decir, ninguno de los modelos está definido para ser empleado en un área con obstáculos y mucho menos lo sería en una simulación de desastre en una ciudad como se lo pretende al querer utilizar un plano real como base de la movilidad. Por lo que fue necesario planificar, analizar y crear unos nuevos modelos los que los denomine: *UPM* y *AVM* con el propósito de ser implementados en la herramienta. El área de obstáculos se definió en un segmento del mapa de la ciudad de Loja obtenido del departamento de Centro Histórico del *Ilustre Municipio del Cantón Loja*, dado en formato *PDF*. El segmento seleccionado fue de 2km x 1km de una zona al azar. Se la editó y se le cambio al formato a *.XBM* porque permitía simplificar la imagen a solo dos colores (blanco y negro) y siendo su ventaja que la estructura interna de la imagen es una matriz manipulable mediante programación, especialmente usando el lenguaje C++. Por lo que desarrollé un programa en este lenguaje, para poder recodificar la imagen a una forma que facilite su uso en la generación de la simulación de una zona de desastre, para el efecto al código se lo convirtió en código hexadecimal y luego a código binario (0,1). El mismo programa insertó de forma aleatoria un número limitado de porciones de 1's de dimensiones previamente definidas para obtener el área de desastre del segmento elegido. Solamente para ayudar a la visualización del producto final de la imagen, se incluyó también en ese programa la opción de codificar nuevamente el resultado de la imagen en binario del área de desastre a *.XBM* como se lo puede apreciar en las figuras: Figura 4.4 y Figura 4.5. Se logró crear este programa gracias al estudio minucioso que se hizo de la estructura de codificación de la imagen *XBM* y la periocidad de la repetición de los códigos en el archivo como se lo señala en el Anexo3. A partir de la matriz binaria, se obtuvo de manera manual los valores de la matriz de adyacencia imprescindible para la movilidad en el modelo *AVM*. La implementación de los modelos en la herramienta requirió crear dos clases para *UPM* y *AVM* a las que las denominó: *Walk* y *Walkd*, respectivamente. Esto se lo puede apreciar en la Figura 4.7 y Figura 4.8. Estas clases tomaron esos nombres porque fueron los nombres iniciales de ambos modelos que por conveniencia posteriormente se los modificó para el desarrollo de un artículo internacional.

El modelo *UPM* obligó a la inserción de un pequeño método para la obtención de números aleatorios basados en las dimensiones del segmento del mapa (1000px x 2000px). Se generó para cada modelo 4 escenarios de movilidad (20, 40 ,60 y 90 nodos) para evaluar el rendimiento de los dos modelos creados y se lo hizo mediante el análisis comparativo de: la eficiencia computacional, eficiencia empleando el protocolo *AODV* y una validación visual en los simuladores nam de *NS2* y *Ad hockey.* La eficiencia de cada uno de los dos modelos que se hace mediante el empleo del protocolo *AODV* se la realizó utilizando de base el archivo de trazas *.tr* generado en *NS2* y luego se le aplicó un conjunto de archivos filtro .sh dados por el Laboratorio de Redes y Telecomunicaciones para obtener los valores de la compleja composición interna del archivo *.tr*. A estos valores se les aplicó las fórmulas respectivas y se obtuvo las gráficas (Figura 5.4, Figura 5.5, Figura 5.6 y Figura 5.7). Todo el proceso anterior se lo resume en la Figura 5.12. Finalmente se elaboró un artículo *PDF* con la ayuda del Coordinador del Laboratorio resumiendo el trabajo hecho y se lo envió a un journal "*Ad Hoc & Sensor Wireless Networks*" (Ver Anexo 4).

Líneas futuras de trabajo:

El presente trabajo de tesis es un pequeño ejemplo del amplio campo de estudio que se puede encontrar en la actualidad enfocado en las redes móviles y su aplicación en la vida real. Siendo su principal interés formar parte de la solución de eventos tan complejos como un desastre porque están en juego la pérdida de vidas y/o recursos.

Al crear estos dos modelos de movilidad pude observar que sería interesante y obviamente deseable que se avancen y se creen modelos que:

- Permitan que los nodos guarden una tabla de rutas dinámicamente para que puedan conocer de antemano la calidad del camino y poder elegir de manera inteligente el mejor camino sin desperdiciar recursos como: el tiempo; que están valioso en las redes *MANET*.

• Crear modelos que puedan utilizar imágenes de mapas de ciudades a todo color de centros urbanos o zonas de desastre sin importar su tamaño ó formatos gráficos para definirlos como áreas de los escenarios de movilidad.

• Incluir una interacción entre los nodos, es decir aumentar el grado de realismo en los modelos de movilidad en la cual permita no solo centrarse en la movilidad como tal, sino también darle características de comportamiento realista cuando un nodo esta frente a otro.

• Capturar realidades locales para formular modelos de movilidad que puedan formar parte de la solución de los problemas de salvamento, circulación peatonal, etc., y permitan crear en los cuerpos de salvamento un criterio de proactividad tecnológica y telecomunicacional.

• La herramienta *Scengen* se puede tomar como la base para el desarrollo de una aplicación más completa y robusta para la simulación pudiendo integrar a su estructura módulos de representación gráfica y estadística que la llevarían a otro nivel.

Es importante que se considere optimizar *Scengen* mediante la integración de nuevas funcionalidades que le permita evolucionar y estar a la par con otras herramientas que existen en la actualidad, con la finalidad de tener una aplicación que sea fácilmente manipulable y adaptable a las necesidades internas.

Todo lo anterior se puede lograr tomando en cuenta los trabajos hechos por grupos investigadores de todo el mundo que se ponen en consideración en el Internet de forma libre. Además estos grupos de trabajo están conformados de estudiantes y docentes de universidades con un gran espíritu colaborativo. Observar y tomar en cuenta las necesidades locales, discernir los comportamientos clave que se deben tomar en cuenta de un ente mediante el análisis de la información recolectada de la fuente y analizar el funcionamiento de las herramientas libres que se encuentran en el Internet especialmente las monopropósito porque pueden aportar ideas valiosas cuando se pretende crear una herramienta robusta. Se aconseja revisar las restricciones del lenguaje frente a la plataforma elegida porque se debe conocer las limitantes

presentan como por ejemplo, el compilador C++ de Ubuntu carece de la librería graphics y sus compiladores de C++ difieren uno del otro como sucede en Ubuntu Natty 11.04.

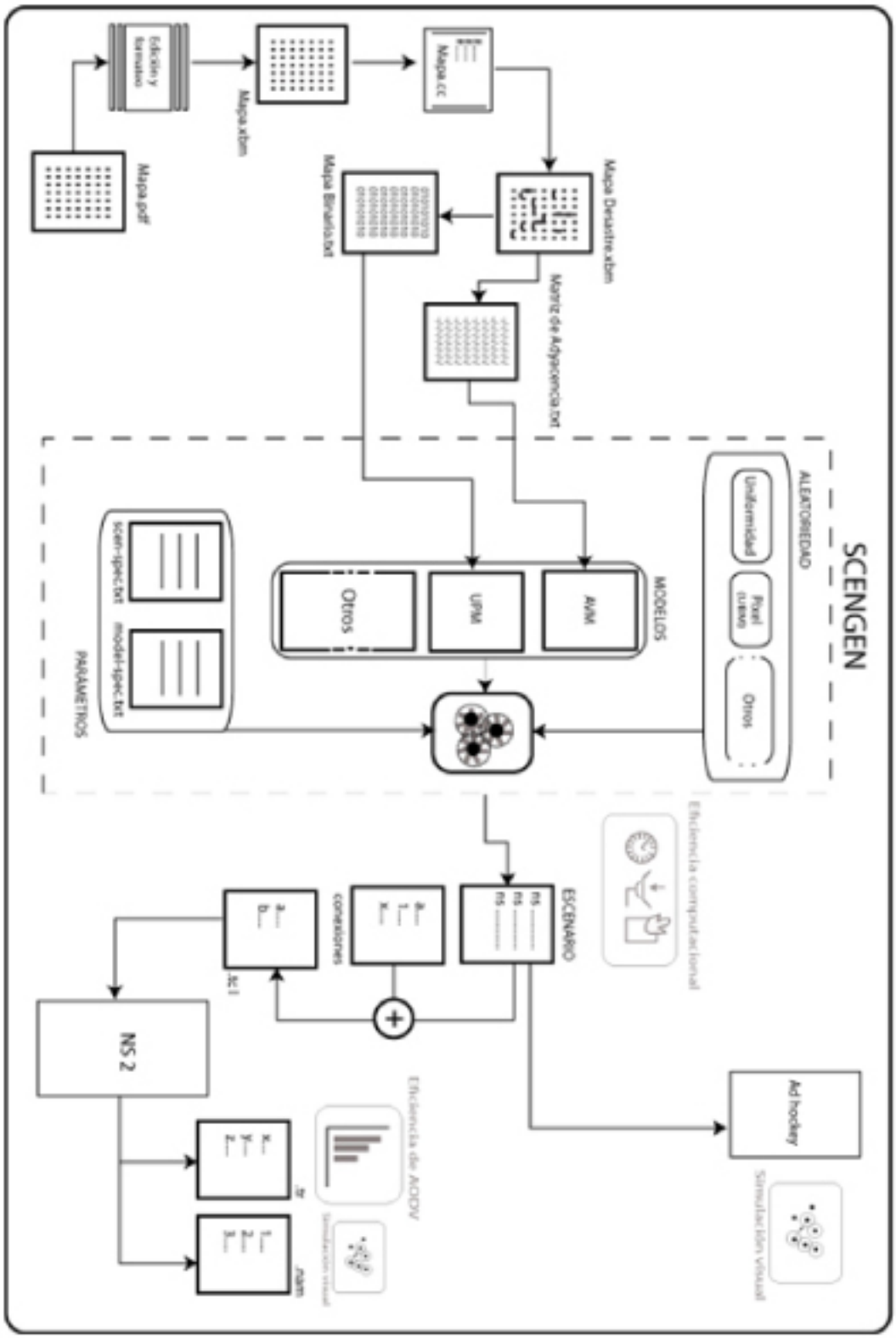


Figura 5.12 Aportes del trabajo de tesis

CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES

• Los modelos de movilidad son representaciones de comportamientos biológicos sobre una determinada área con o sin obstáculos, regidos por parámetros tales como: velocidad, tiempo, entre los más importantes. Donde su manipulación incide en la eficiencia de la comunicación de la red.

• Los patrones de movilidad generados por un ser humano demandan un análisis previo para ser captado en toda su dimensión y posteriormente implementados para ser utilizados para la generación de escenarios como es el caso de los de emergencia y rescate.

• El modelo de movilidad *UPM*, basa su solución en la determinación del próximo espacio libre, únicamente monitoreando los espacios vecinos ubicados a la derecha, izquierda, arriba y abajo. Tomando siempre en cuenta el punto de evacuación especificado.

• El modelo *AVM*, genera su siguiente paso determinando dinámicamente el valor mínimo de las diferentes aristas vecinas aplicando la fórmula de distancia euclidiana y el concepto del *algoritmo de Dijsktra*.

• El gran universo de herramientas de simulación tienden a guardar compatibilidad de sus salidas con la herramienta *NS2* como es el caso de *Scengen* que sus resultados se constituyen en entradas para ser analizadas en *NS2*.

• *Scengen* es una herramienta escalable y audazmente estructurada porque permite implementar nuevos módulos sin afectar el funcionamiento del resto del programa.

• Los escenarios de emergencia y rescate necesitan ser simulados en áreas llena de obstáculos que afectan a las conexiones y cobertura de los dispositivos de comunicación móvil.

• *UPM* y *AVM*, son dos modelos que tienen un mismo fin que es la de dar movilidad en escenarios de emergencia y rescate, pero que son diferentes en la forma de conseguir la movilidad del nodo y en la dependencia que tienen de los obstáculos para generarlo.

• *UPM* debido a la minuciosidad en la generación de los patrones de movilidad demanda gran capacidad de recursos computacionales en comparación con lo que necesita *AVM*.

• Las tasas de eficiencia alcanzadas por el protocolo *AODV* son superiores en el modelo de movilidad *AVM* que en *UPM* porque se mantiene más cercana al valor de 1 que constituye la expresión ideal de comunicación de dichas tasas.

• Los modelos desarrollados pueden mejorar su desempeño mediante la inclusión de nuevos conceptos informáticos tales como: la inclusión de registros de desplazamiento, registros de pesos de las aristas, colaboración entre nodos, etc. Que permitirán optimizar recursos y llevar a un nivel más inteligente la movilidad.

• Scengen al tener una estructura clara y escalable, permite incluir nuevos modelos de movilidad de forma independiente del resto de la estructura del programa. Por lo que es útil cuando se quiera evaluar modelos de movilidad de forma relativamente sencilla.

RECOMENDACIONES

- Para mejorar el rendimiento de los modelos *UPM* y *AVM*, sería conveniente implementar un sistema de lectura-escritura de registros de movilidad de cada nodo tanto en la clase *Walk* y *Walkd* respectivamente.

- La implementación de un nuevo modelo se debe de hacer tal como lo indica el archivo *MODEL* de la herramienta *Scengen*. Observando la forma como se ha escrito o estructurado dentro del código de los otros modelos en cada uno de los pasos detallados en dicho archivo. Se debe tener especial cuidado en la digitación de las variables globales que se deba añadir ya que unas están en mayúsculas, otras en minúsculas y algunas de las dos formas. Esto afecta tanto en el desarrollo del modelo y en la compilación de la aplicación.

- Los parámetros incluidos en el archivo modelspec influyen en el modelo de movilidad y el archivo scenspec hace lo propio en el escenario. Debido a ello se debe tener en claro primero que parámetros se quiere manipular. Para cambiar el número de nodos o si se desea probar un nuevo modelo deberá hacérselo en el archivo *scenspec*.

- La clase principal *Scengen*, preformatea los parámetros resultantes de los procesos hechos por la herramienta, por lo que si se desea incluir dentro del archivo plano generado que se lo denomina escenario, se debe modificar e incluir lineas de código en esta clase, que permita incluir archivos para ser reconocidos en otras herramientas, por ejemplo: *Ad hockey*.

- Los valores obtenidos mediante procesos aleatorios son tomados en la clase model y enviados a la clase del modelo de movilidad como una estructura denominada *Vector* por lo que se debe de instanciar a esta estructura para conocer y manipular dichos valores.

• Cuando se necesite como es el caso del modelo *Walk* de un proceso de obtención de números aleatorios enteros se debe incluir un nuevo modelo de distribución randómica instanciando a la clase random, debido que los números obtenidos son con decimales y no poseen un rango específico, ya que dichas distribuciones están implementadas para trabajar en áreas libres de obstáculos.

BIBLIOGRAFÍA

[1] Jorge Eduardo Treviño, Jaime Leonardo Bobadilla y Miguel Anulfo Saumett, **Simulación y evaluación de redes de ad hoc bajo diferentes modelos de movilidad**. Universidad Nacional de Colombia. 2003

[2] Xiaoyan Hong, Mario Gerla, Guangyu Pei , Ching-Chuan Chiang , **A Group MobilityModel for Ad Hoc Wireless Networks**. Computer Science, Department, California University, USA. 1999.

[3] Carles Gómez Montenegro, Josep Paradells Aspas. **REDES AD-HOC: EL PRÓXIMO RETO. Wireless Networks Group,** Entel Dept., Technical University ofCatalonia (UPC). Barcelona, España. 2004

[4] Bhavyesh Divecha, Ajith Abraham, Crina Grosan, Sugata Sanyal , **Impact of Node Mobility on MANET Routing Protocols Model**s. Mumbai University, Mumbai , India,Centre for Quantifiable Quality of Service in Communication Systems Norwegian University of Science and Technology , Norway., School of Technology and Computer Science Tata Institute of Fundamental Research , India. 2007.

[5] Nils Aschenbruck, Elmar Gerhards-Padilla, Peter Martini , Jens Tölle. **A survey on mobility models for performance analysis in tactical mobile networks**. Institute of Computer Science IV. Bonn, Germany. 2008.

[6] Ashok-Kumar, Chandra-Sekaran, Pascal Weisser, Klaus D. Müller-Glaser. A **Comparison of Bayesian Filter Based Approaches for Patient Localization During Emergency Response to Crisis**. 2009.

[7] Samuel C. Nelson, Albert F. Harris III, and Robin Kravets , Event-driven, **Role-based Mobility in Disaster Recovery Networks** , Department of Computer Science, University of Illinois at Urbana–Champaign, Urbana, IL, USA. 2007.

[8] Daniel Ramirez S. **Simulación de redes AD-HOC utilizando AODV**. Universidad el Bosque, Instituto Alberto Merani, Bogotá, Colombia. 2007.

[9] Bhavyesh Divecha, Ajith Abraham, Crina Grosan, Sugata Sanyal. **Impact of Node Mobility on MANET Routing Protocols Models**. Mumbai University, Mumbai , India. Centre for Quantifiable Quality of Service in Communication Systems, Norwegian University of Science and Technology, Norway. School of Technology and Computer Science, Tata Institute of Fundamental Research, India. 2007.

[10] Andrea Clementi, Angelo Monti, Riccardo Silvestri. **Modelling mobility: A discrete revolution**. Dipartimento di Matematica, Università di Roma Tor Vergata, Via della Ricerca Scientifica,Dipartimento di Informatica, Università di Roma La Sapienza, Italia. 2011

[11] Tracy Camp , Jeff Boleng, Vanessa Davies . **A Survey of Mobility Models for Ad Hoc Network Research**. Dept. of Math. and Computer Sciences , Colorado School of Mines, Golden, CO, USA. 2002

[12] S. Cristaldi, A. Ferro, R. Giugno, G. Pigola, A. Pulvirenti. **Obstacles constrained group mobility models in event-driven wireless networks with movable base stations**. Department of Mathematics and Computer Science, University of Catania, Italy. 2011.

[13] Ibrahim Khider, Wang Furog, Yin Wei Hua, Sacko. **A Survey of Geographic Restriction Mobility Model**. Department of Communication and Information Systems. Huazong University of Science and Technology. Hubei, China. 2007.

[14] Mirco Musolesi, Cecilia Mascolo. **Chapter 1: Mobility Models for Systems Evaluation A Survey**. Dartmouth College, USA y University of Cambridge, UK. 2009.

[15] Christos Papageorgiou, Konstantinos Birkos, Tasos Dagiuklas, Stavros Kotsopoulos. **An obstacle-aware human mobility model for ad hoc networks**. Dept. of Electrical and Computer Engineering, University of Patras. Grecia. 2009.

[16] Yu Chenchen, Li Xiaohong, Zhang Dafang. **An Obstacle Avoidance Mobility Model. Software School**, School of Computer and Communication Hunan University Changsha, China. 2010.

[17] Ashok-Kumar Chandra-Sekaran1, Gunnar Ste¬fansson, Christophe Kunze, Klaus D. Müller- Glaser1, Pascal Weisser1. **A Range-Based Mon¬te Carlo Patient Localization during Emergency Response to Crisis.** Institute for Information Pro¬cessing Technology (ITIV), University of Karl¬sruhe (TH), Germany. FZI Research Centre for Information Technology, Karlsruhe, Germany. 2009.

[18] Nils Aschenbruck, Aarti Munjal, Tracy Camp. **Trace-based mobility modeling for multi-hop wireless networks.** University of Bonn, Institute of Computer Science, Alemania. Colorado School of Mines, Dept. of Math. and Computer Sciences, USA. 2010.

[19] Franck Legendre, Vincent Borrel, Marcelo Dias de Amorim, Serge Fdida . **Modeling Mobility with Behavioral Rules: The Case of Incident and Emergency Situations**. Laboratoire d'Informatique de Paris 6 (LIP6/CNRS) ,Universit´ Pierre et Marie Curie – Paris, Francia. 2006.

[20] Nils Aschenbruck, Elmar Gerhards-Padilla, and Peter Martini . **A survey on mobility models for performance analysis in tactical mobile networks**. 2008.

[21] Nelson Reinoso. **DISEÑO PUBLICITARIO**, II edición. ISBN-9978-40-234-9. Gráficas Iberia. Bogotá, Colombia. 2003. pp. 70-123.

[22] Juan Pablo Hernández, Daniel Márquez. **Monografía: Redes Móviles Adhoc. Sistemas Distribuidos**. Universidad Nacional de Rosario Facultad de Ciencias Exactas, Ingeniería y Agrimensura. Argentina. 2006

[23] Waal C. Gerharz M.. **BonnMotion a Mobility Scenario Generation and Analysis Tool Documentation**, Universidad de Bonn, Alemania. 2011

[24] Kazuki Konishi, Kumiko Maeda, Kazuki Sato, Akiko Yamasaki, Hirozumi Yamaguchi, Keiichi Yasumoto, Teruo Higashino. **MobiREAL Simulator – Evaluating MANET Applications in Real Environments**. Osaka, Japon. 2005.

[25] Razvan Stanica, Emmanuel Chaput, André-Luc Beylot. **Simulation of vehicular ad-hoc networks: Challenges, review of tools and recommendations**. Department of Communications and Networking, ENSEEIHT, IRIT, University of Toulouse, France.2011.

ANEXOS

ANEXO 1

CUESTIONARIO PARA EL CUERPO DE BOMBEROS DE LA CIUDAD DE LOJA

1. ¿Qué servicios ofrece su institución a la ciudadanía?
2. ¿Se han presentado últimamente desastres en la ciudad en los cuáles haya ayudado su organización? ¿Cuáles? ¿Dónde?
3. ¿Las personas de los sectores adyacentes han ayudado?
4. ¿Qué desastres tienen la mayor probabilidad de presentarse en la ciudad de Loja?

    Naturales:

    Provocados:

En caso de desastre de grandes proporciones:

5. ¿Qué instituciones tomarían el control y dirigirían las tareas de ayuda y rescate?
6. ¿Su institución está preparada para hacerle frente? ¿De qué forma?
7. ¿La ciudad quedaría desprovista de medios de comunicación?
8. ¿Existe alguna estrategia de movilidad para desastre de esta índole?
9. ¿Cuál sería la estrategia de comunicación a usarse para mantener informada y calmada a la población?
10. ¿Existe alguna organización ciudadana para brindar ayuda en tales casos?
11. ¿Cuáles serían las zonas más vulnerables de la ciudad? ¿Existe algún estudio al respecto?
12. ¿Cuáles son zonas seguras para hacer evacuaciones en caso de ser necesario?
13. ¿La ciudad de Loja, cuenta con algún refugio?

En estos escenarios el comportamiento de la persona:

14. ¿El sobreviviente tiene algún patrón de comportamiento tras un desastre?
15. ¿Permanece en un sitio en espera de ayuda o sale a buscar ayuda? ¿Lo hace en grupo? ¿Tienen algún tipo de relación sanguínea?
16. ¿Hacia qué lugares se dirigen?
17. ¿Tiende ayudar a los demás? ¿De qué forma?

18. ¿Considera que la sociedad lojana está suficientemente instruida sobre la forma correcta de actuar frente a un desastre?

INFORME DE ENTREVISTA CON LA ENTIDAD DE SOCORRO

Nombre de la entidad: Cuerpo de Bomberos de la ciudad de Loja.
Persona entrevistada: Sgto. Luna. Encargado de del Área de Comunicación y Promoción del Cuerpo de Bomberos.

Resolución del Bloque de Preguntas Formuladas:

1. ¿Qué servicios ofrece su institución a la ciudadanía?

El cuerpo de Bomberos de esta ciudad se encarga de colaborar, dar servicios de emergencia, dar agua en caso de escasez, atención médica, prevención de incendios, capacitación entre otras actividades.

2. ¿Se han presentado últimamente desastre en la ciudad en los cuales haya ayudado su organización? ¿Cuáles? ¿Dónde?

Se han realizado salidas a pedidas de auxilio a la comunidad. Abastecer de recursos a zonas en desastre.

3. ¿Las personas de los sectores adyacentes han ayudado?

Frente a desastres como incendios, deslaves y otras calamidades las personas de los alrededores si acostumbra a ayudar a los damnificados.

4. ¿Qué desastres tienen la mayor posibilidad de presentarse en la ciudad de Loja?

Naturales:

Incendios forestales, deslaves, inundaciones.

Provocados:

Incendios forestales

En caso de desastre de grandes proporciones:

5. ¿Qué instituciones tomarían el control y dirigirían las tareas de ayuda y rescate?

En caso de desastres grandes se conforma un grupo de ayuda entre: bomberos, policía, cruz roja, oficina de gestión de riesgos, el ejército y por la propia comunidad. El que generalmente, frente a un desastre estaría a cargo como cabeza organizativa sería el gobernador de la provin-

cia y la oficina de gestión de riesgos.

6. ¿Su Institución está preparada para hacerle frente? ¿De qué forma?

La Institución está preparada para hacerle frente a la situación, aunque en caso de ser necesario se pide ayuda a los cuerpos de los bomberos de las ciudades vecinas o a nivel nacional.

7. ¿La ciudad quedaría desprovista de medios de comunicación?

Si, dentro de un desastre de grandes proporciones no habría comunicación excepto la brinden los cuerpos de salvamento.

8. ¿Existe alguna estrategia de movilidad para desastre de esa índole?

No existe ningún tipo de estrategia establecida para hacerle frente a un desastre debido a nuestra cultura de que "Eso nunca nos pasará".

9. ¿Cuál sería la estrategia de comunicación a usarse para mantener informada y calmada la población?

No existe estrategia de ningún tipo.

10. ¿Existe alguna organización ciudadana para brindar ayuda en estos casos?

No, aparte de los organismos gubernamentales señalados.

11. ¿Cuáles serían las zonas más vulnerables de la ciudad? ¿Existe algún estudio al respecto?

En caso de inundaciones las zonas bajas de la ciudad tales como Carigán, Memphis entre otras. Si existen unos mapas de zonas vulnerables de la ciudad facilitados por el Municipio de Loja y que lo tenemos a disposición actualmente.

12. ¿Cuales son zonas seguras para hacer evacuaciones en caso de ser necesario ?

Espacios abiertos, estadios y lugares libres de infraestructuras que puedan causar daños a las personas, tales como: tendido eléctrico, edificaciones, etc.

13. ¿La ciudad de Loja cuenta con algún refugio?

No existe.

En estos escenarios el comportamiento de la persona:

14. ¿El sobreviviente tiene algún patrón de comportamiento tras un desastre?

No. Luego de un terremoto o cualquier otro desastre la persona entra en estado de shock.

15. ¿Permanece en un sitio en espera de ayuda o sale a buscar ayuda? ¿Lo hace en grupo? ¿Tiene algún tipo de relación sanguínea?

Generalmente, la persona queda devastada y no sabe qué hacer. En caso de núcleos familiares siguen a una cabeza que es el padre o el hermano mayor.

16. ¿Hacia qué lugares se dirigen?

Quedan moviéndose en los alrededores de su casa destruida.

17. ¿Tienden ayudar a los demás? ¿De qué forma?

No, por su estado anímico y las angustias de su porvenir frente a tal calamidad.

18. ¿Considera que la sociedad lojana está suficiente instruida sobre la forma correcta de estar frente a un desastre?

No, la sociedad lojana no se encuentra informada y es indolente frente al tema.

Comentarios:

- Las personas deben conocer que existe un espacio en cada inmueble que se puede constituir en la mejor manera de salvarse frente a desastre en caso de un terremoto se debe ubicar bajo un pilar de la casa, a un lado bajo un mueble para estar dentro del triángulo de vida.
- El centro histórico de la ciudad es una zona vulnerable pues son edificaciones antiguas y están conti

  guas unas con otras
- No existe ningún sitio seguro en caso de evacuaciones en el centro histórico.

• Los planes de contingencia los establecen los organismos de socorro cuando suceden según la gravedad del caso.

• La cultura de desastre de la comunidad lojana está muy distante de otras comunidades.

• Los desastres que se presentan con más frecuencia en nuestra ciudad son las inundaciones en las zonas bajas e incendios.

• La ciudad no cuenta con planes de contingencia establecidos sino que los cuerpos de salvamentos los generan en el momento mismo que se presenta el desastre.

• En un desastre en la ciudad es aconsejable que las personas se desplacen a espacios abiertos como el

parque Jipiro, el parque lineal de la Tebaida o al coliseo “Ciudad de Loja”.

Material recolectado:

- Mapas de Riesgos de la ciudad de Loja (Cuerpo de Bomberos).
- Mapa del centro histórico de la ciudad de Loja (Municipio de Loja).

Anexo2

Manual de instalación la aplicación *Ad hochey*

Es un script desarrollado en *Perl* que mediante el empleo de la librería gráfica *TK* (*Tool Kit*) permite generar una interfaz simple que facilita al usuario la creación y simulación de escenarios para mediante la manipulación de parámetros y contenidos de manera interactiva. Por el uso de estos recursos de la programación se dice que es una aplicación *Perl/Tk.*

Su código se encuentra distribuido de forma libre en la red de dos formas:

1. Empaquetado dentro de la aplicación Scengen en la siguiente dirección electrónica:
   - **http://isis.poly.edu/~qiming/scengen/index.html**

2. Directamente en la página de sus desarrolladores del *Monarch Project* (Proyecto Monarca) en:

   - **http://www.monarch.cs.rice.edu/cmu-ns.html**

En dicha página se puede obtener los paquetes de Perl/Tk adecuados para cada una de las versiones de la aplicación. Además, de valiosos comentarios y breves documentos que pueden minimizar los errores de compilación y uso.

Versiones y Contenido

Existen dos versiones del Adhockey disponibles en Internet, cuya diferencia radica obviamente en sus características nuevas mejoradas u omitidas o; y en la versión de Perl/Tk que utiliza.

En la primera se emplea Tk400.200 se la puede obtener en la dirección:

- **http://www.monarch.cs.rice.edu/ftp/monarch/wireless-sim/cmu-extendedns-1.1.2.tar.gz**

Su contenido se lo puede ver en la Figura a.2.1

Figura a.2.1 Segmento del árbol de directorio del paquete cmu-extendedns-1.1.2.tar.gz

y en la segunda se usa *Tk800.015*, en las direcciones:

- **http://www.monarch.cs.rice.edu/cmu-ns.html**
- **http://isis.poly.edu/~qiming/scengen/index.html**

El contenido se lo ve en la Figura a.2.2

Figura a.2.2 Segmento del árbol de directorio del paquete *Scengen*.tar relacionado

En este trabajo de tesis se va emplear la versión que usa *Tk800.015,* que es la distribución más reciente de la aplicación y que viene además junto al *Scengen*.

Generación del Archivo Ejecutable (ad-hockey)

Para la obtención del archivo ejecutable de la aplicación se necesita abrir la terminar e ingresar al directorio que contiene los archivos de Ad-hockey de tal forma:

xo@ubuntu:~$ **cd juan/scengen/ad-hockey**

Una vez dentro del directorio se tipea la palabra ***make*** para que proceda la compilación del **Makefile** que instanciará a los demás archivos para obtener el archivo ejecutable ad-hockey.

xo@ubuntu:~/juan/scengen/ad-hockey$ **make** (Figura a.2.3)

```
xo@ubuntu:~/juan/scengen/ad-hockey$ make
gcc -c   -g -Wall -o what-time.o what-time.c
gcc -o what-time  what-time.o
xo@ubuntu:~/juan/scengen/ad-hockey$
```

Figura a.2.3 Corrida del comando *make* en un terminal

Problemas en la compilación:

Existen dos errores que se van a presentar en el momento de querer de ejecutar el archivo ejecutable las mismas que son:

1. Error por la mala escritura de la palabra reservada *borderwidth*.(Figura a.2.4)

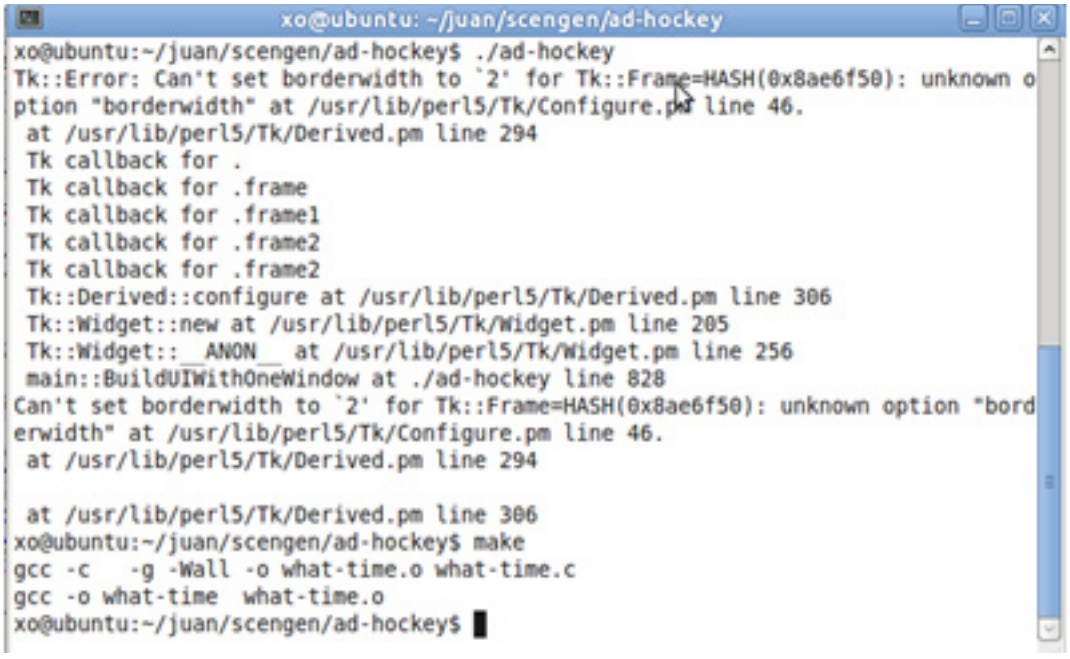


Figura a.2.4 Error de compilación.

• Editar el archivo Ad-hockey y reemplazar la palabra borderwidth por – borderwidth, de la forma expresado en Figura a.2.5.

```
##############################################################################
# Construct the UI
##############################################################################
sub BuildUIWithOneWindow {
    $MW = MainWindow->new;
    $MW->title('Ad Hockey');

    MakeMenus($MW);

    $CANVAS = $MW->Canvas(
                          -width => '15c',
                          -height => '15c',
                          -background => $normal_canvas_bkgnd,
                          );
    $CANVAS->pack;

    my $speed_id_frame = $MW->Frame(-borderwidth => 2,);
    my $timepos_frame = $MW->Frame(borderwidth => 2,);
    my $msg_frame =  $MW->Frame(borderwidth => 2,);
    my $controls_frame = $MW->Frame(-borderwidth => 0,);

    MakeControls($controls_frame);

    $info_display = $msg_frame->Text(
```

Figura a.2.5 Corrección de la palabra reservada *–boderwidth*.

2. Error por no encontrar el archivo what-time. (Figura a.2.6)

```
xo@ubuntu:~/juan/scengen/ad-hockey$ ./ad-hockey
open2: exec of what-time failed at ./ad-hockey line 1648
```

Figura a.2.6 Error de compilación.

• Editar el archivo *Ad hockey* y escribir la ruta donde se encuentra alojado el archivo *what-time* como se la puede ver en Figura a.2.7 y Figura a.2.8.

```
#########################################################################
##### Read in Scenario files provided to the command line
if ($default_trace) {
    if (OpenTraceFile($default_trace) >= 0) {
        $trace_on = 1;
        $show_agt = 1;
        $show_rtr = 1;
        $show_pkt_lines = 1;
    }
}
if ($default_scenario) {
    ReadScenario($default_scenario);
    if ($default_commpattern ne "") {
        ReadCommunicationPattern($default_commpattern);
    }
}
                                    cambiar   "/home/xo/juan/scengen/ad-hockey/what-time"
#########################################################################
### setup the time source
open2( \*ReadTime, \*WriteTime, "what-time" or die;
WriteTime->autoflush();

#########################################################################
### begin running things
```

Figura a.2.7 Cambiar la ruta de acceso al archivo *what-time*.

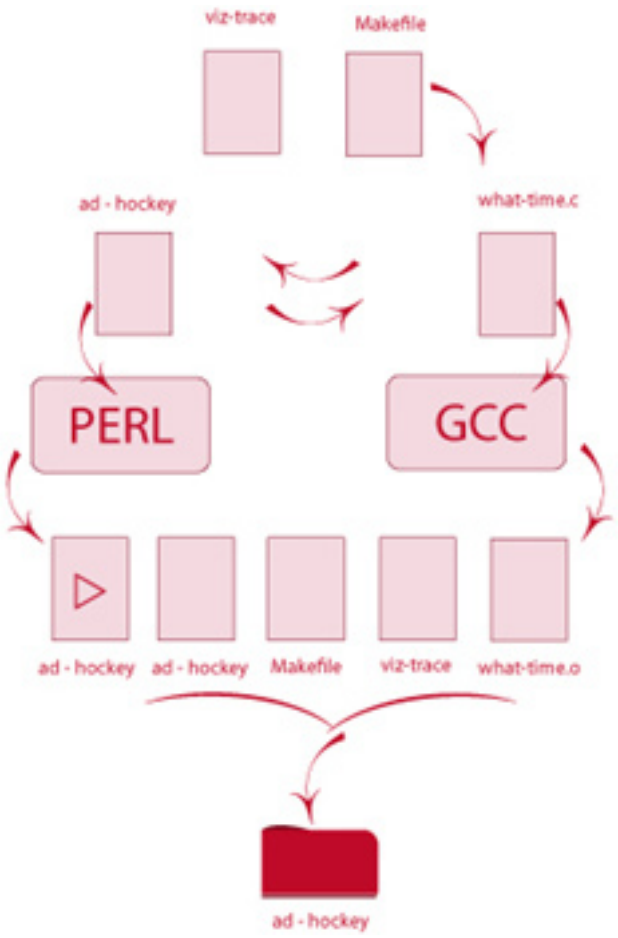


Figura a.2.8 Interacción de los elementos durante la ejecución de: make y *.ad-hockey*.

FUNCIONAMIENTO

Trabaja conjuntamente con la herramienta *NS*, aunque no la ejecuta directamente la aplicación por si sola sino que necesita que el usuario realice un trabajo independiente en ella para cuyo resultado integrarlo a la aplicación. Se lo demuestra gráficamente en la Figura a.2.9

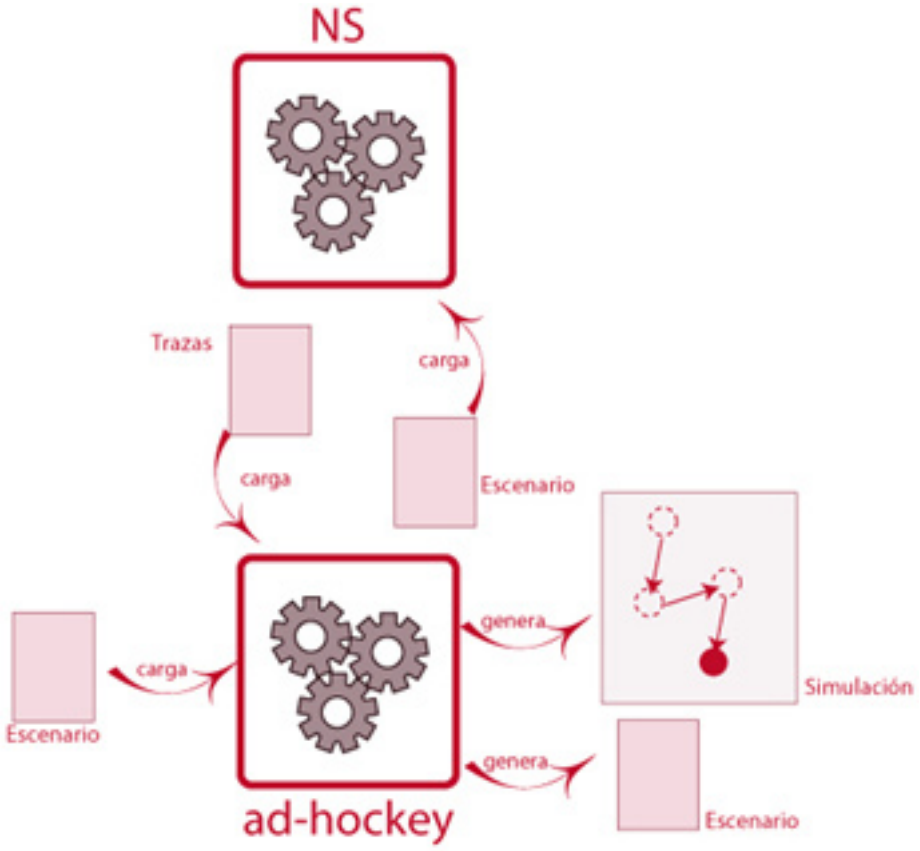


Figura a.2.9 Funcionamiento de *ad-hockey*

Funcionamiento de la interfaz de usuario

Se ejecuta el programa desde el terminal, ingresando primeramente al directorio *ad-hochey* y luego, añadiendo **./ad-hockey.** Esto se lo ve en la Figura a.2.10.

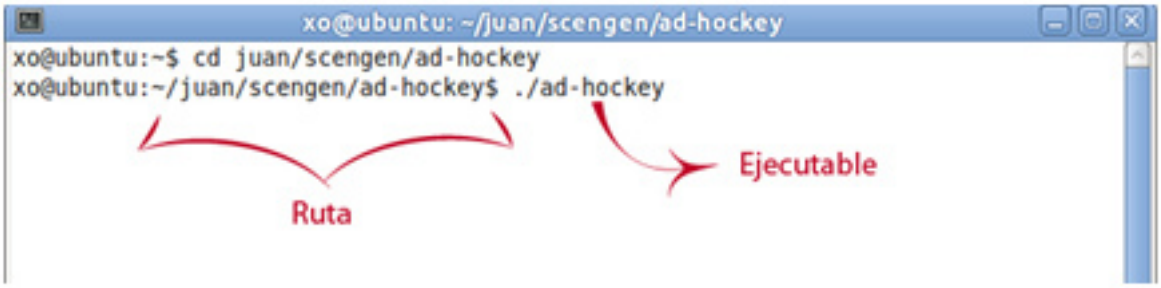


Figura a.2.10 Partes necesarias para ejecutar el programa ad-hockey

Inmediatamente se visualiza la interfaz que contiene la siguiente composición según la Figura a.2.23:

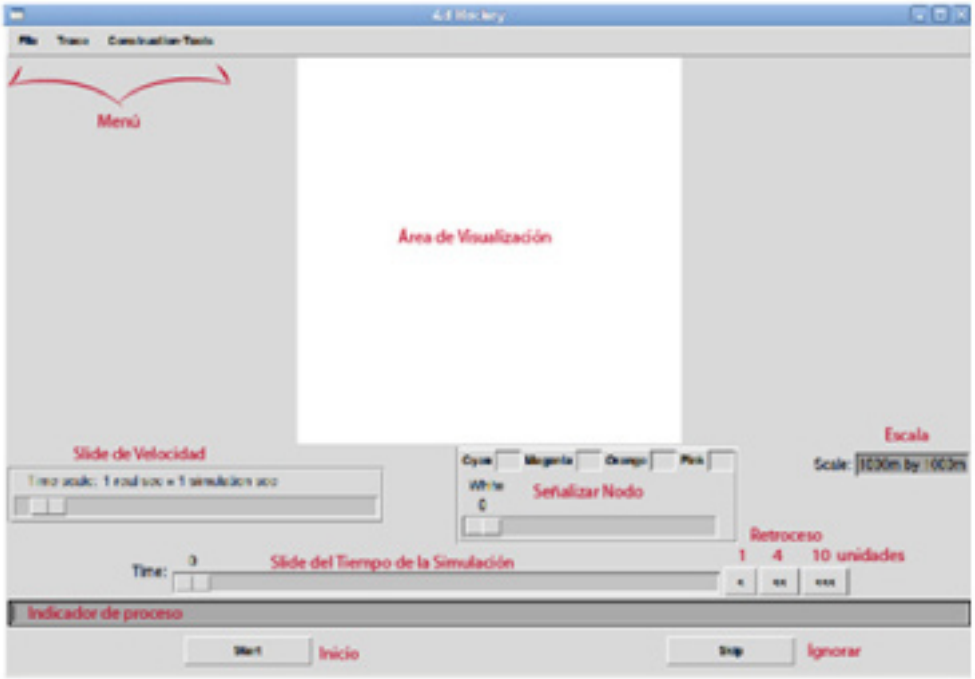


Figura a.2.11 Elementos de la interfaz gráfica de ad-hockey.

Simulación a partir de un escenario

1. Se carga el archivo que contiene el escenario según Figura a.2.12, Figura a.2.13.

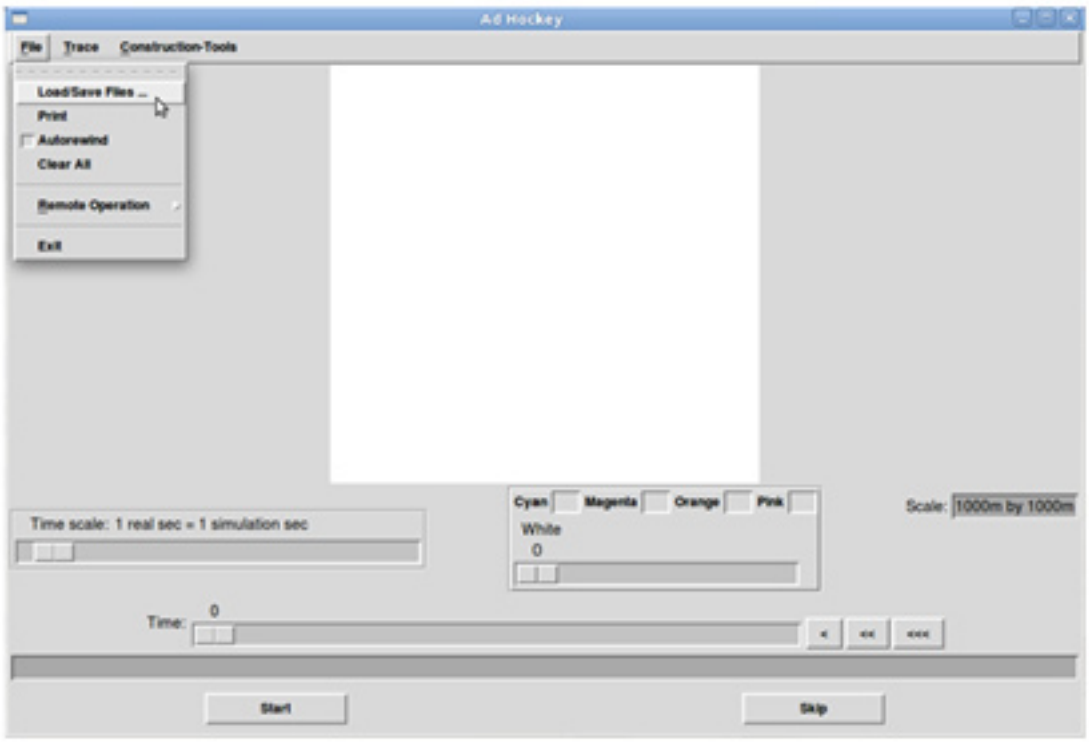
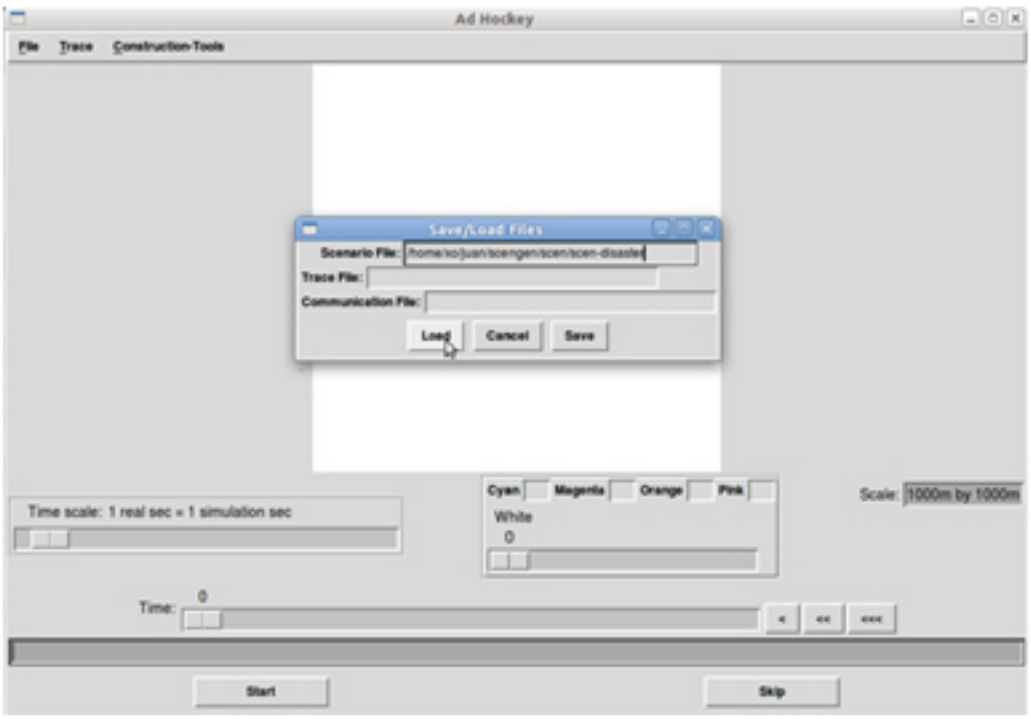


Figura a.2.12 Selección de la opción Load/Save Files (cargar y guardar archivos)



ra a.2.13 Cargar el archivo de un escenario.

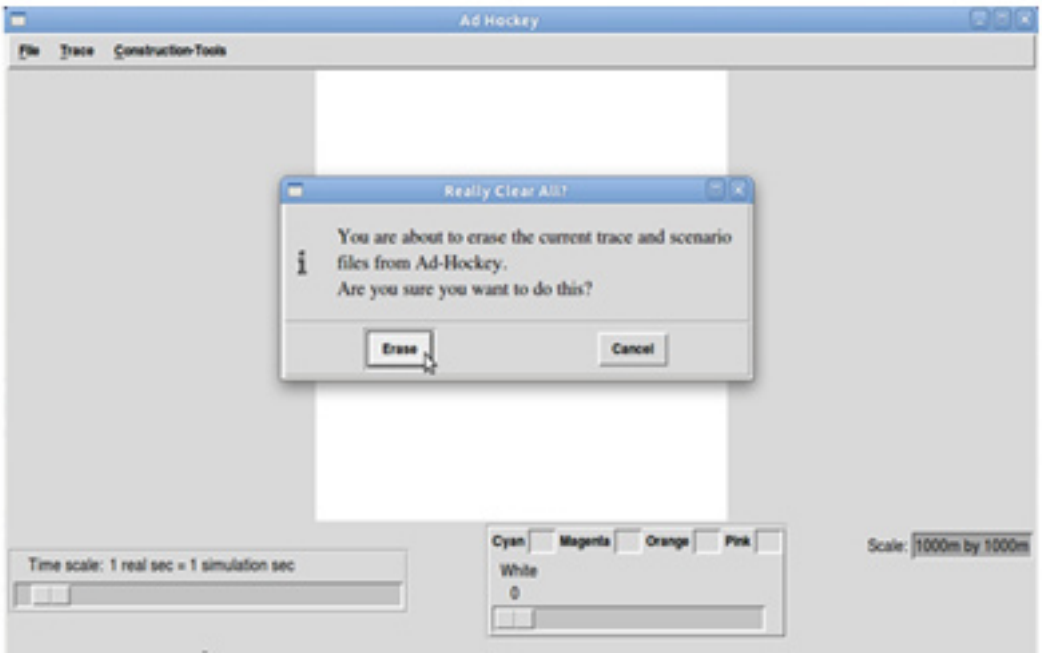


Figura a.2.14 Borrar el contenido anterior del Área de Simulación.

2. Se señala el nodo(s) cromáticamente para que sea más fácil identificarlos durante la simulación. Por defecto, el programa pinta al nodo 0 como en la Figura a.2.15.

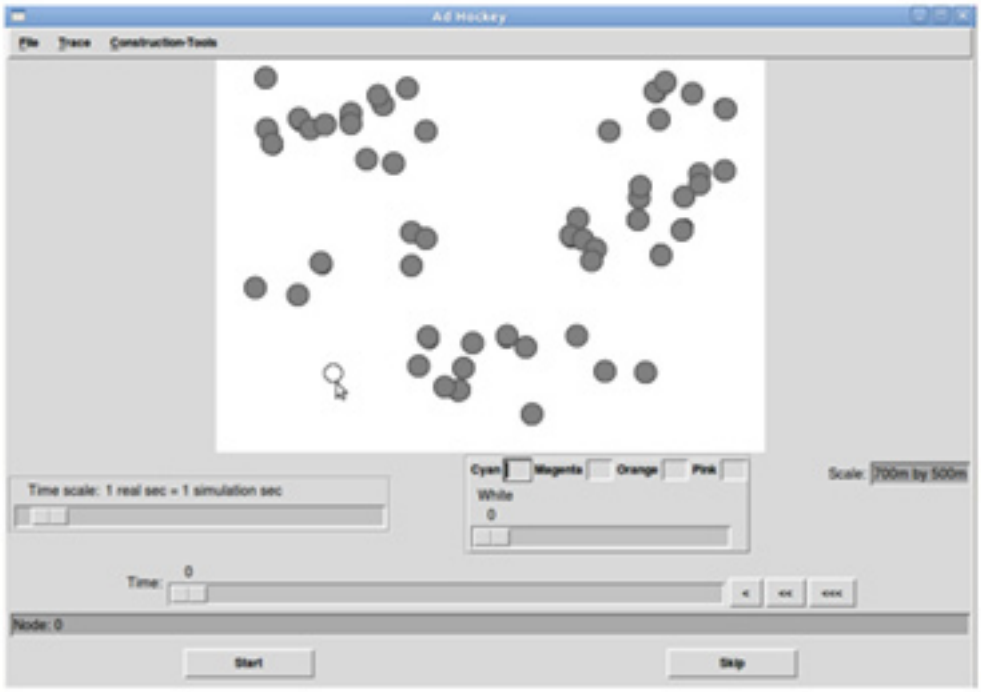


Figura a.2.15 Señalización del nodo que tiene asignado el enfoque por defecto.

Si se pone el puntero del ratón sobre algún nodo, tomará un color amarillo intenso y mostrará su range circle (círculo de cobertura) como se lo ve en la Figura a.2.16.

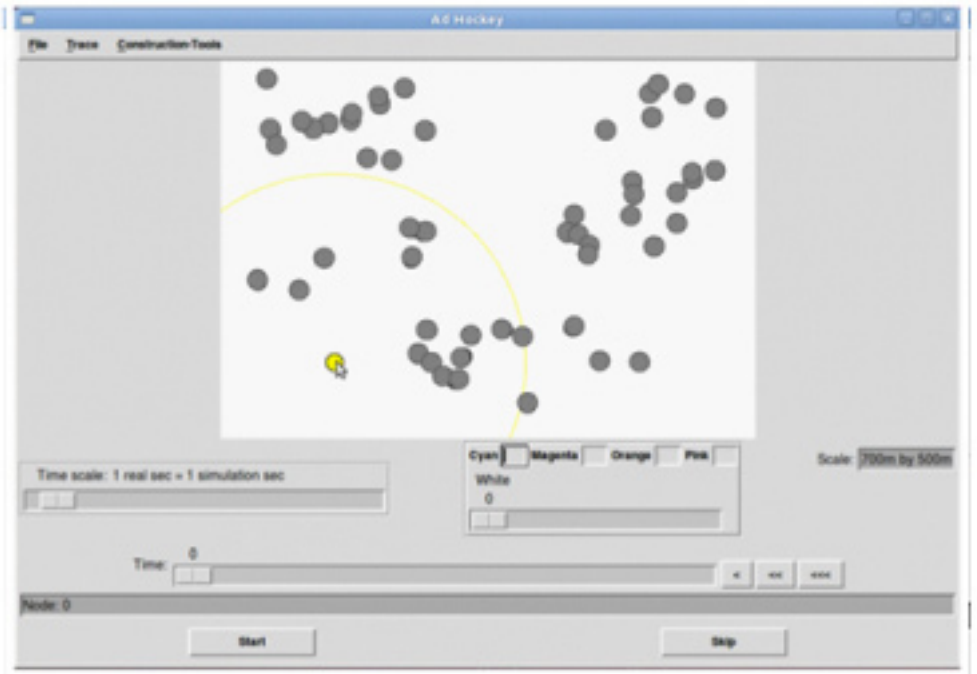


Figura a.2.16 Visualización del círculo de cobertura del nodo.

Se puede cambiar la señalización por defecto a otro nodo utilizando el slide ubicado en la zona de Señalizar Nodo. Se hace un movimiento ascendente o descendente hasta encontrar el nodo deseado como se lo puede notar en Figura a.2.17

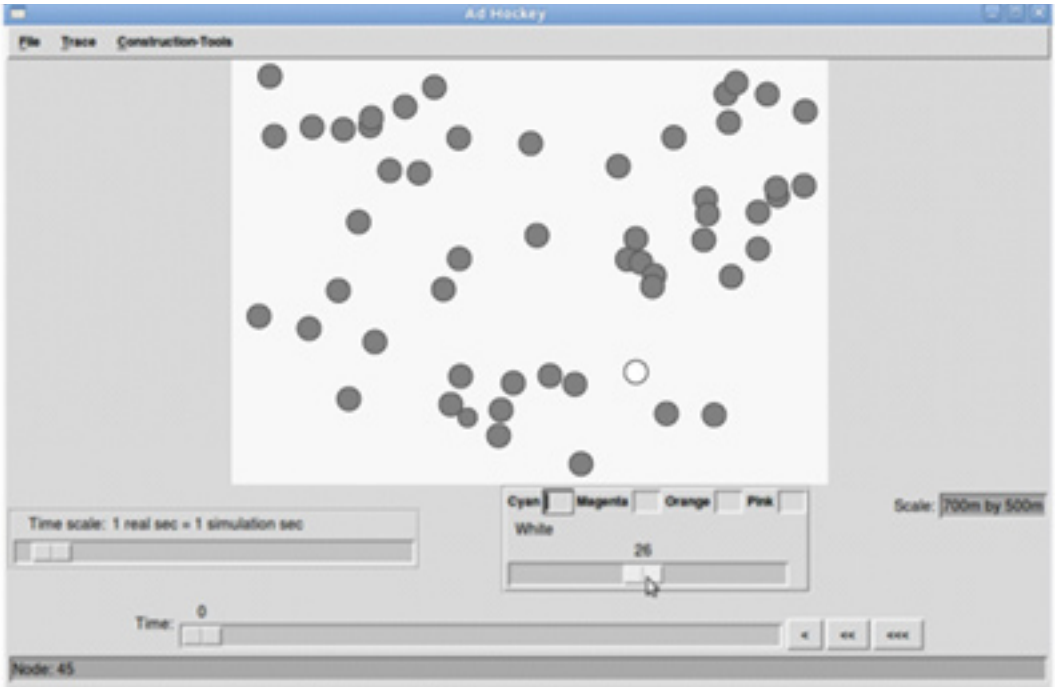


Figura a.2.17 Manipulación del Slide que pone la enfoque por defecto sobre un nodo.

También se puede señalar hasta un máximo de 4 nodos más, escribiendo su número en los casilleros de los colores establecidos en la zona de Señalizar Nodo. Los colores son: cian (cyan), magenta, anaranjado (orange) y rosa (pink). Esto se lo ve la Figura a.2.18.

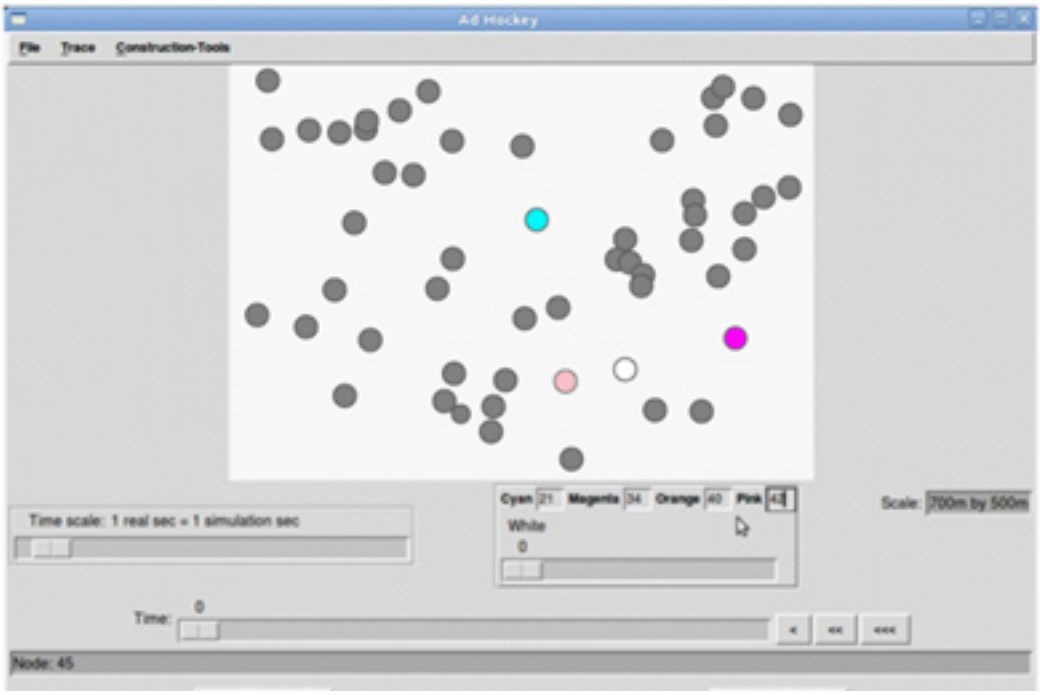


Figura a.2.18 Asignación cromática a 4 nodos de la simulación.

3. Para observar el movimiento que está estipulado en el escenario se tiene que presionar el botón *Inicio* (*Start)*. Donde Figura a.2.19

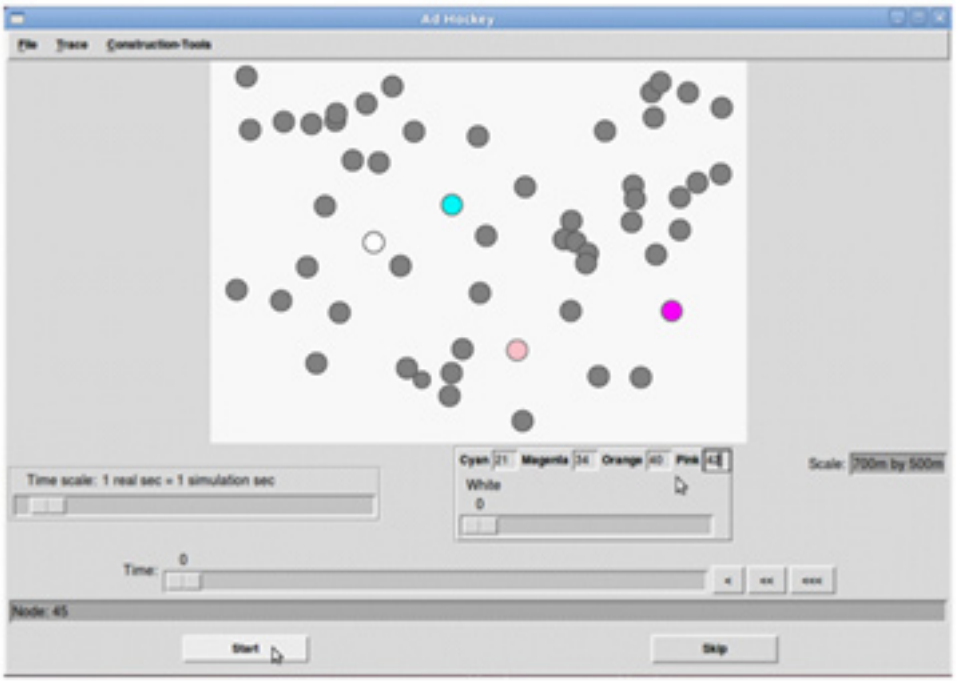


Figura a.2.19 Botón *Start*

Inmediatamente que se ejecuta, el label cambia su contenido por la palabra Parar (*Stop*). Por lo que si se debe presionar nuevamente el botón. Como lo indica la Figura a.2.20.

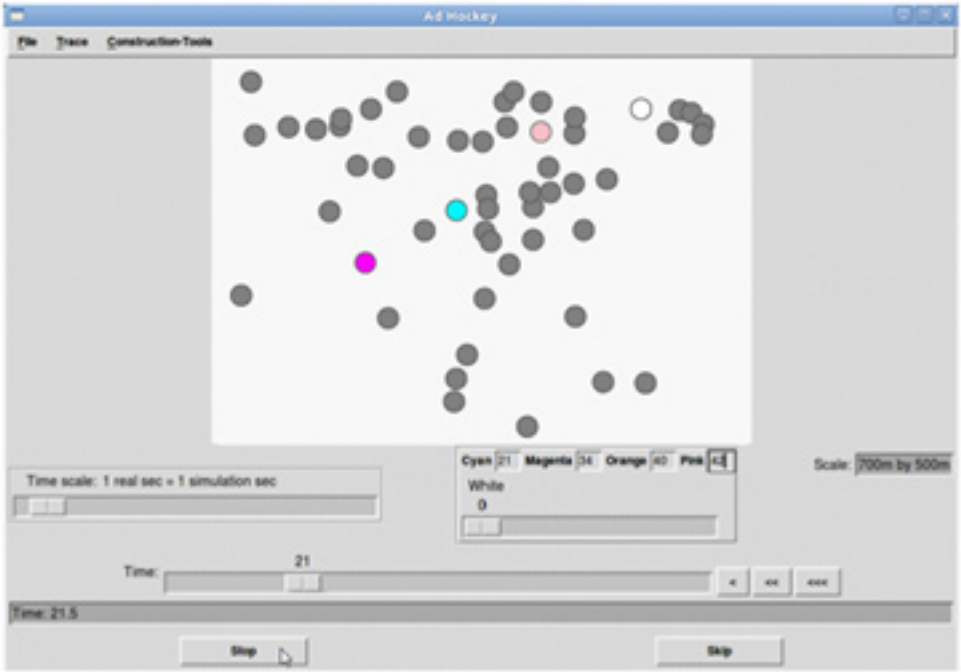


Figura a.2.20 Botón *Stop*

4. Se manipula la velocidad de la simulación a través del *Slide de Velocidad*. Obsérvese la Figura a.2.21

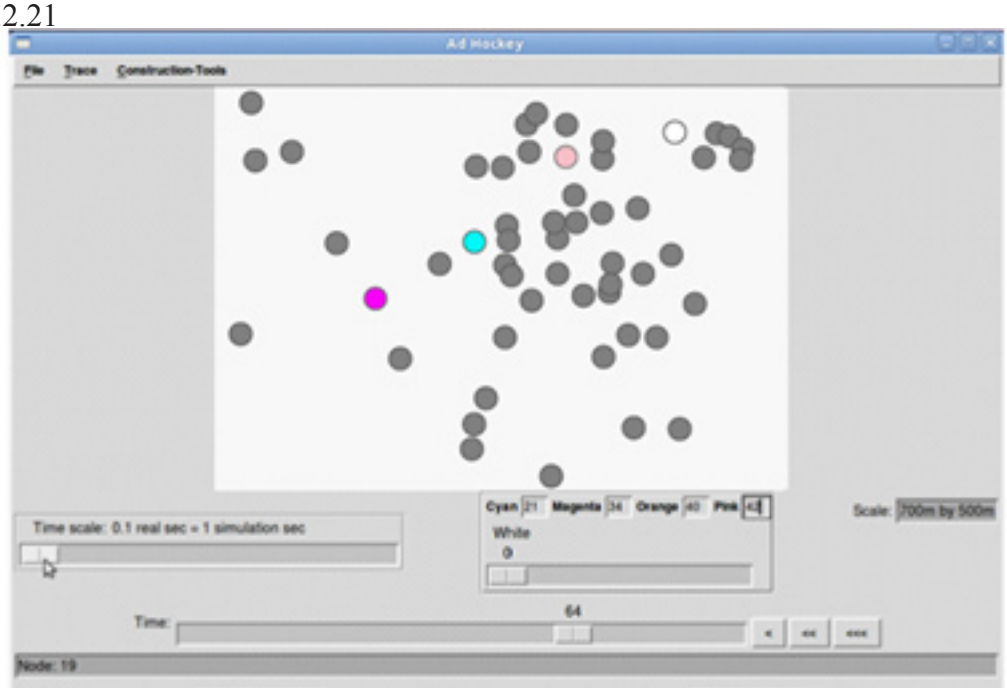


Figura a.2.21 Manipulación del *Slide de Velocidad*

5. Se manipula el tiempo de la simulación a través del *Slide de Tiempo* y sus botones adyacentes con saltos fijos. Obsérvese la Figura a.2.22

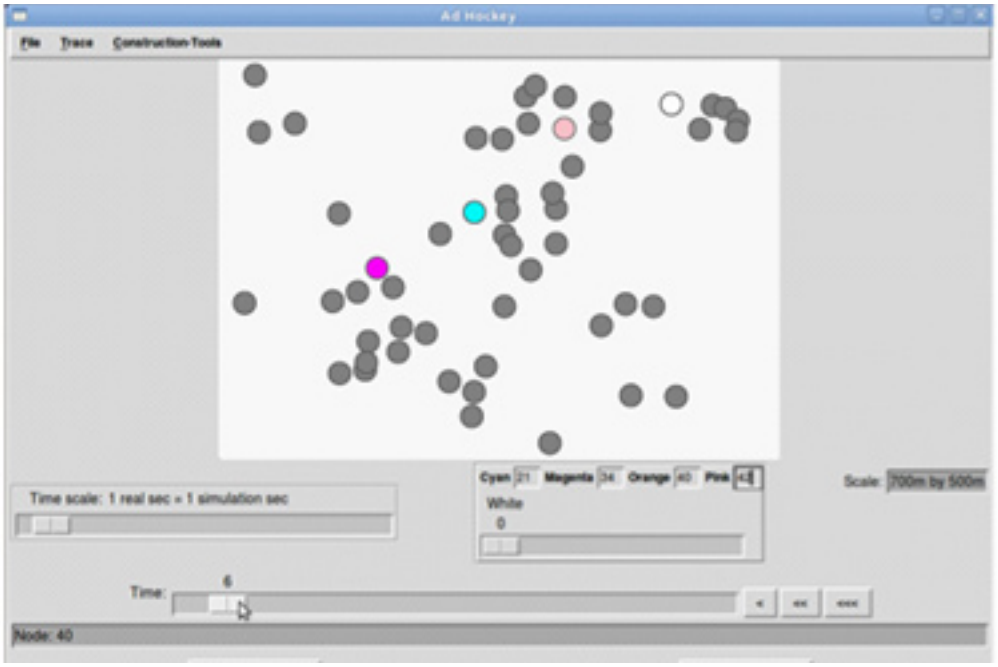


Figura a.2.22 Manipulación del *Slide de Tiempo.*

6. Observar el Indicador de procesos para obtener información de cada acción que se realice

o se ejecute. Obsérvese la Figura a.2.23

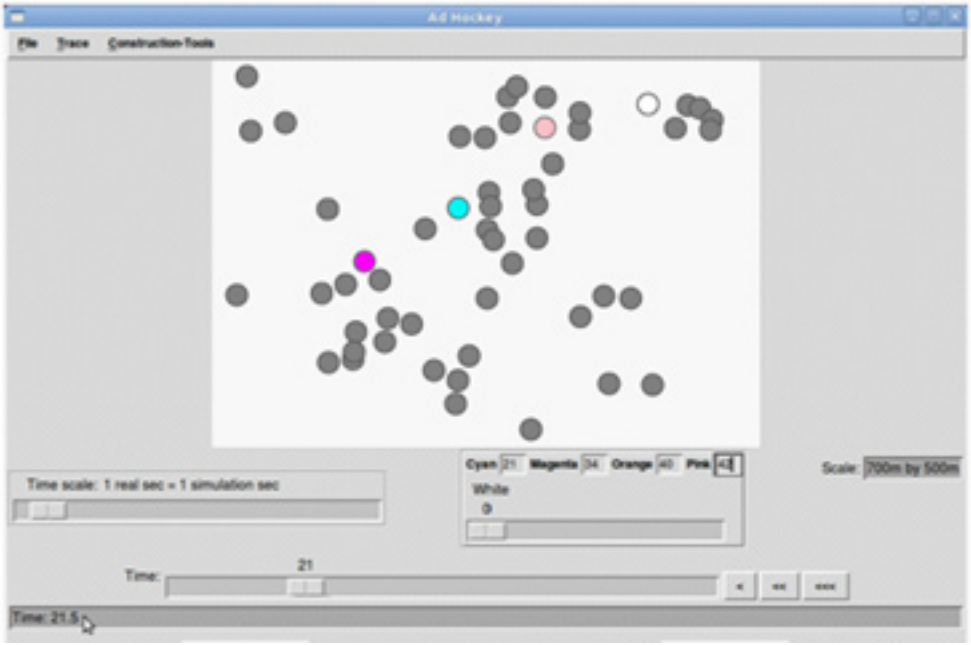


Figura a.2.23 Información presentada durante una simulación en el *Indicado de Procesos*.

7. Marcar Autorewind (autorecorrido) para ver en forma indefinida la ejecución de la simulación según Figura a.2.24.

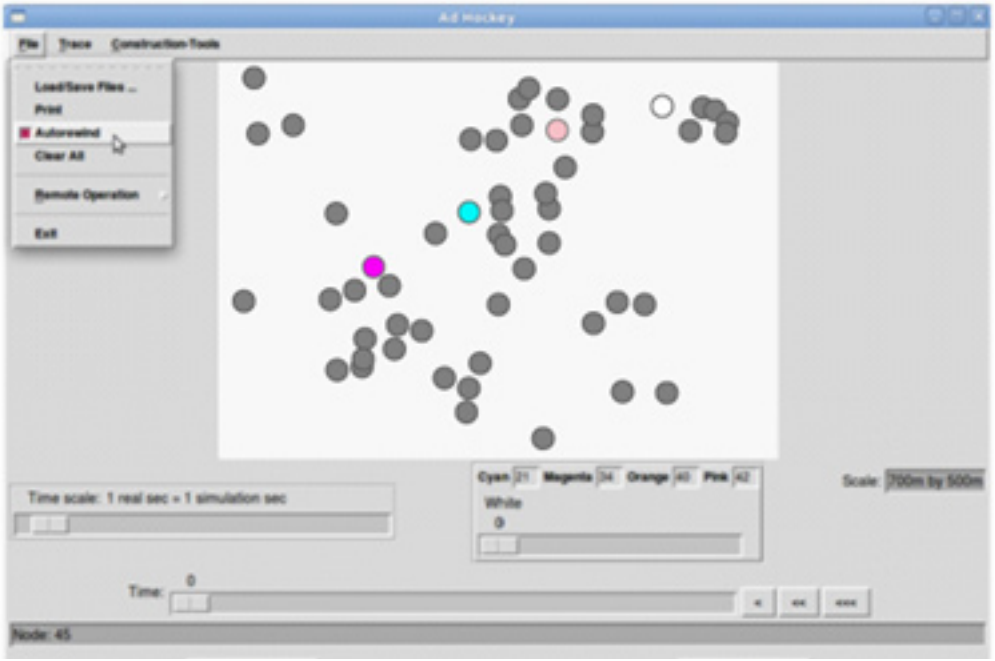


Figura a.2.24 Activación del *checkbox* de la opción *Autowind*.

8. Activar la visualización de los de *range circles* (círculos cobertura) de todos los nodos.

De la forma como se lo demuestra en la Figura a.2.25 y Figura a.2.26

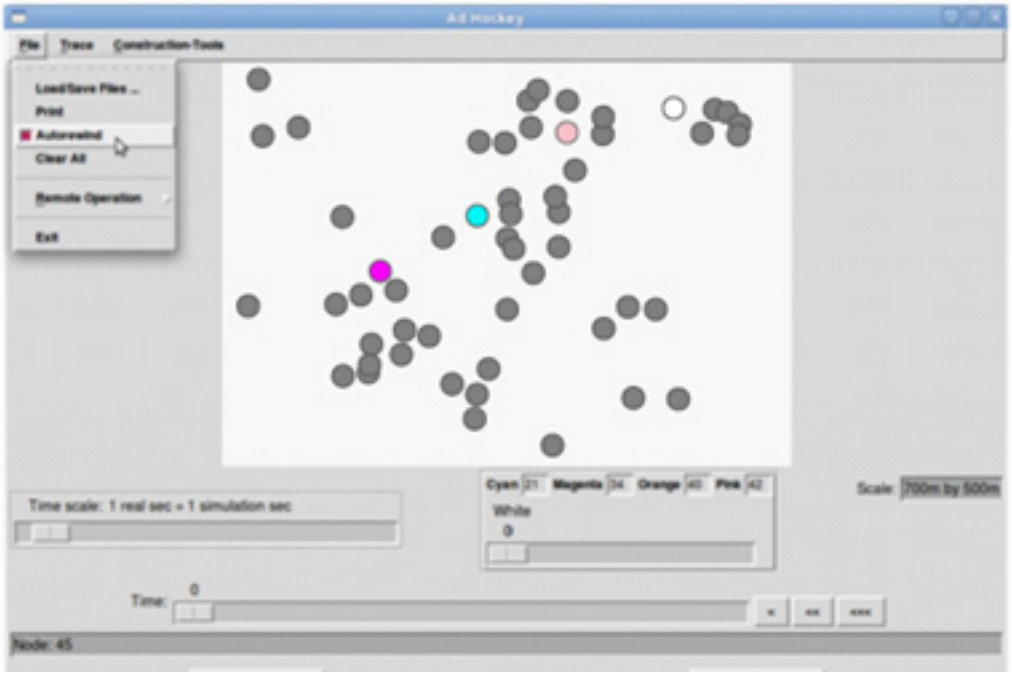


Figura a.2.25 Activar la opción *range circles*

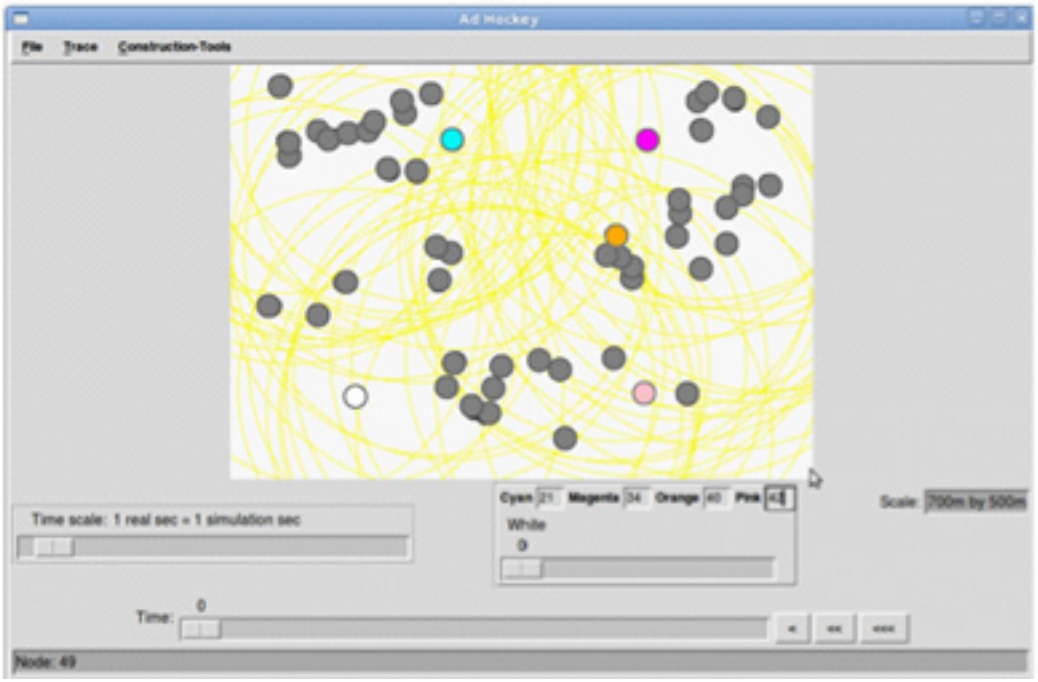


Figura a.2.26 Visualización de todos los range circles de la simulación.

9. Activar la visualización de *cobwebs* (*clustering*) de los nodos como se puede apreciar en Figura a.2.27 y Figura a.2.28

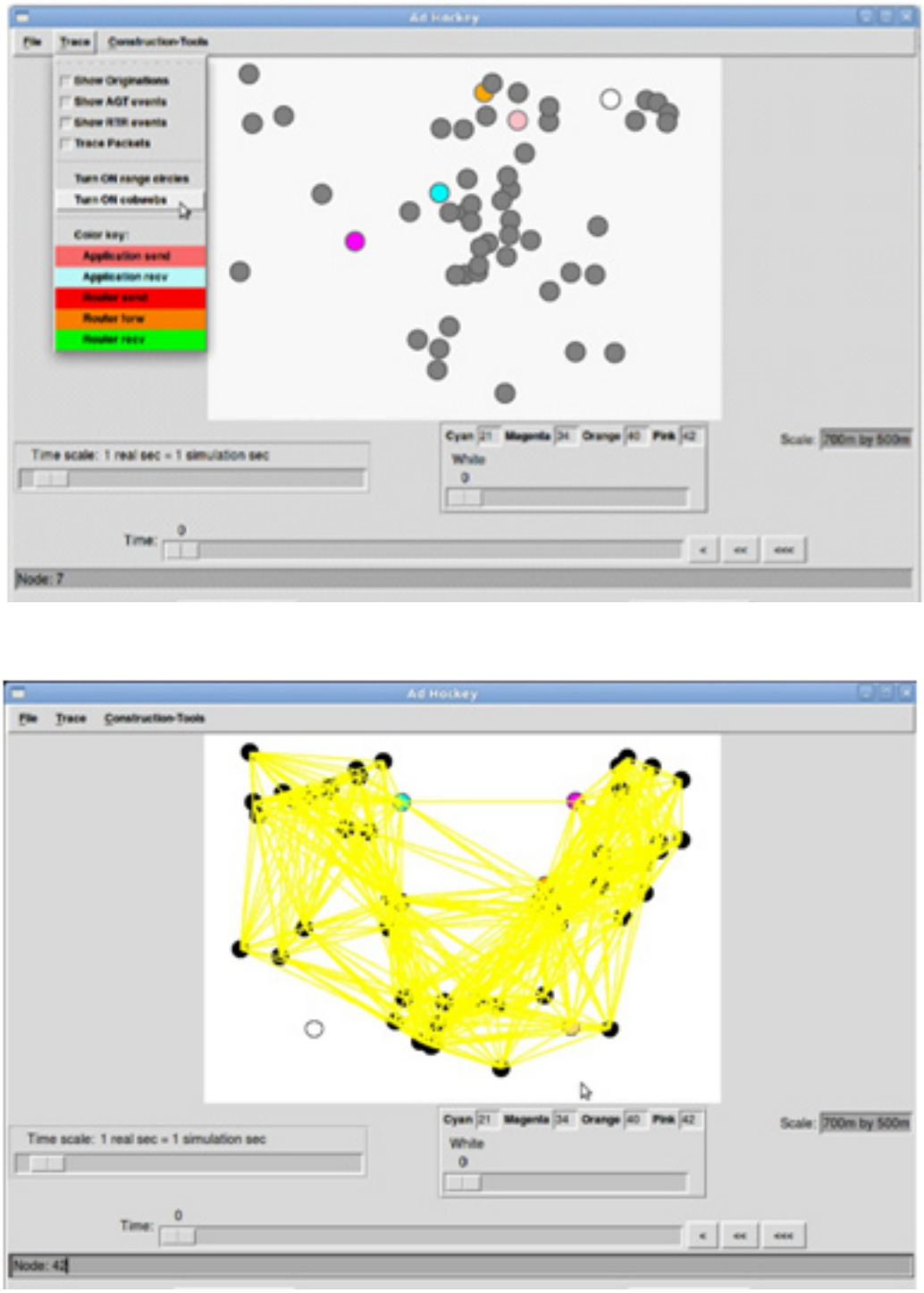


Figura a.2.28 Cobwebs de todos los nodos de la simulación.

10. Al hacer clic sobre cada nodo se puede visualizar su movimiento sobre el Área de Simulación. Obsérvese la Figura a.2.29

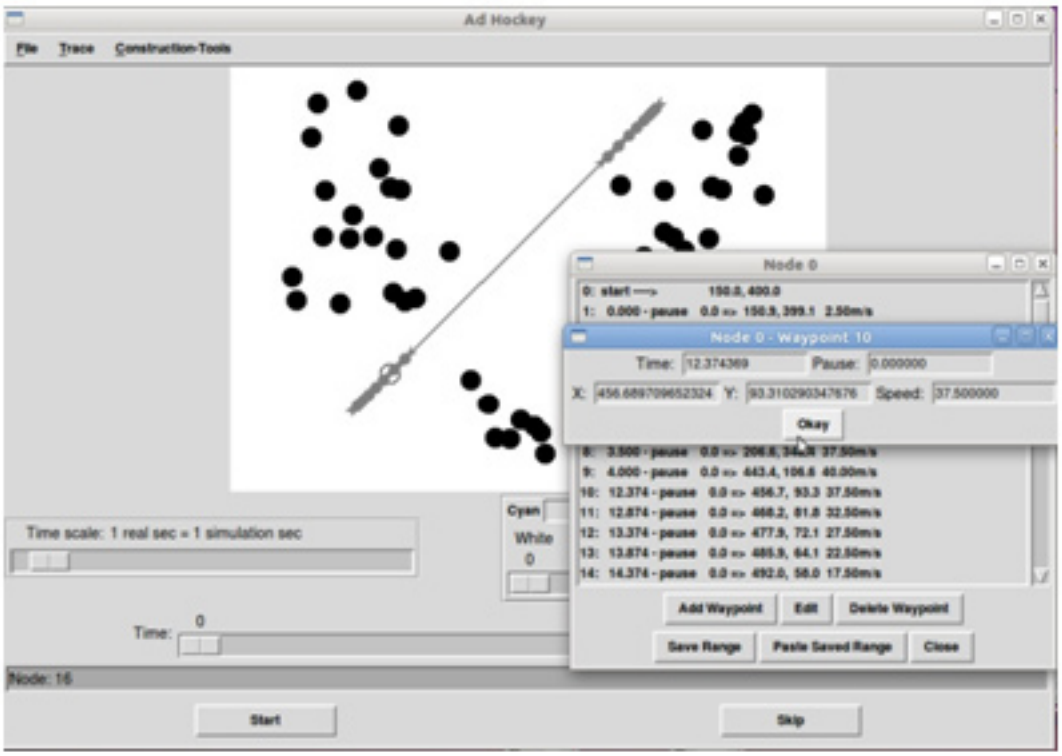


Figura a.2.29 Manipulación de los parámetros del desplazamiento del nodo.

Describe de forma visual el trayecto del nodo. Existe la posibilidad de poder modificar los parámetros para alterar su desplazamiento en el *Área de Simulación*, añadir más puntos de movimiento, copiar y pegar segmentos de desplazamientos de otros nodos.

Creación de escenarios

El programa permite al usuario crear escenarios de forma interactiva sobre el *Área de Simulación* y posteriormente almacenar este resultado en un archivo de texto que puede ser utilizado en otro programa tal como el *NS*. Estos escenarios pueden ser solo de nodos o de nodos y obstáculos

Para crear un escenario debemos de seguir los siguientes pasos:

1. Revisar y/ modificar los parámetros del archivo en blanco, de acuerdo a lo que señala la Figura a.2.30 y la Figura a.2.31.

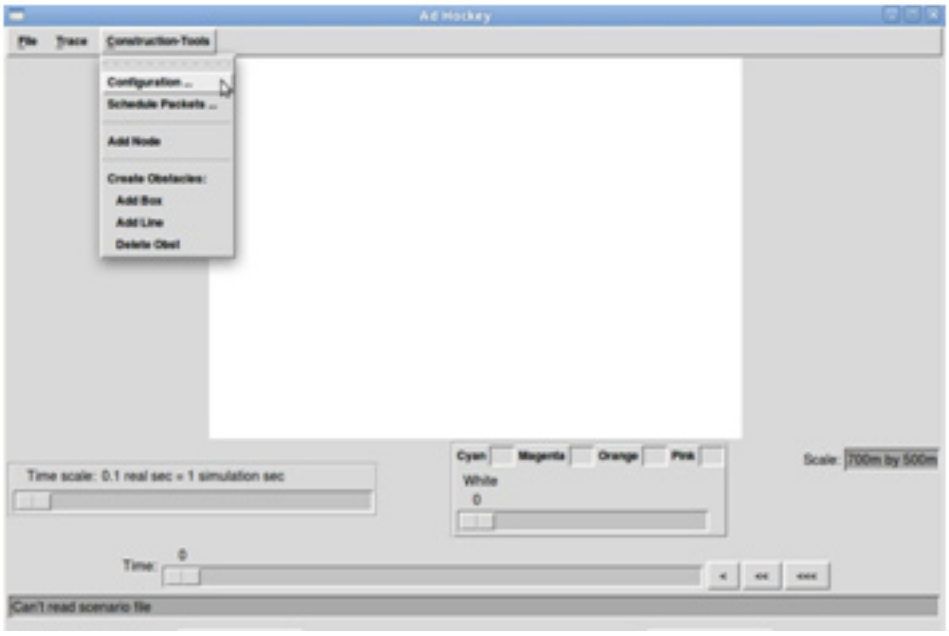


Figura a.2.30 Selección del ítem *Configuration* (Configuración)

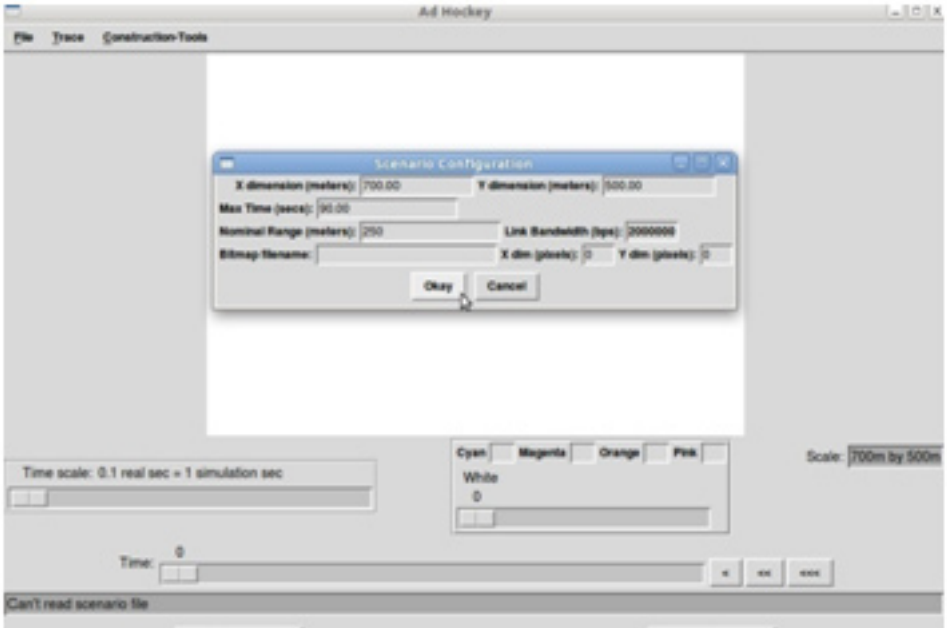


Figura a.2.31 Revisión de los parámetros compositivos del *Área de Simulación.*

2. Crear un el nodo (s) y generar su movimiento en el *Área de Simulación*. Esto se muestra en: Figura a.2.32, Figura a.2.33, Figura a.2.34, Figura a.2.35 y Figura a.2.36

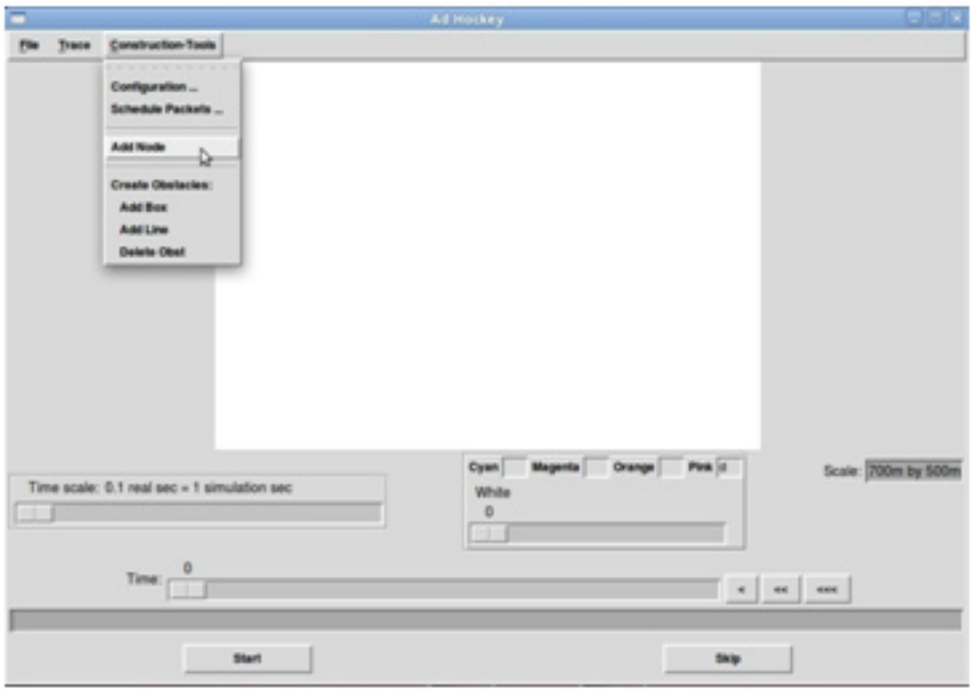


Figura a.2.32 Elección del ítem del Menú: Add Node.

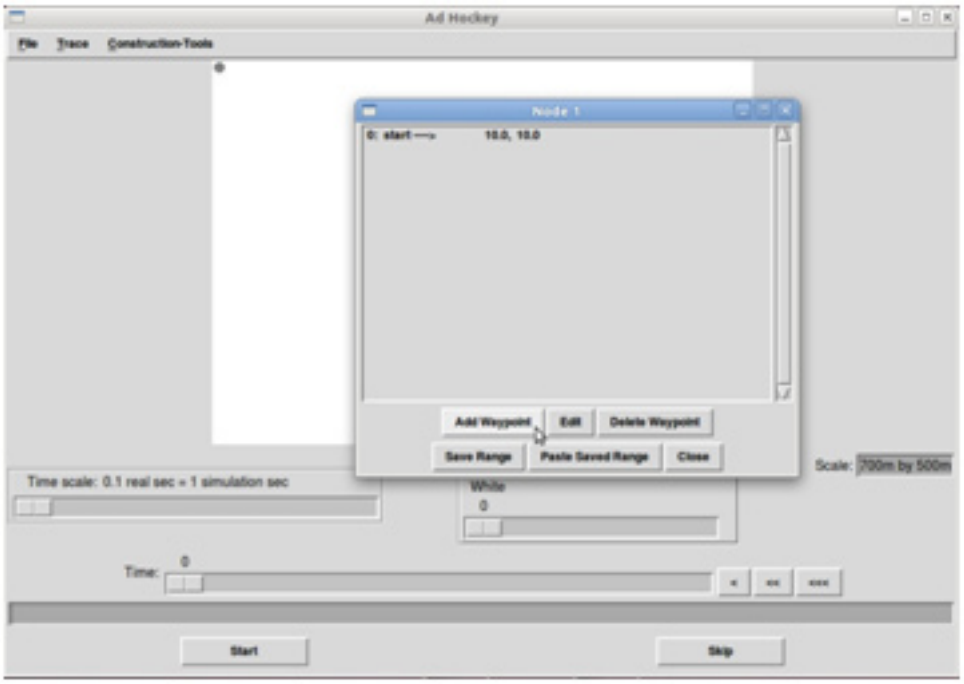


Figura a.2. 33 Añadir un nuevo punto de movimiento a la trayectoria.

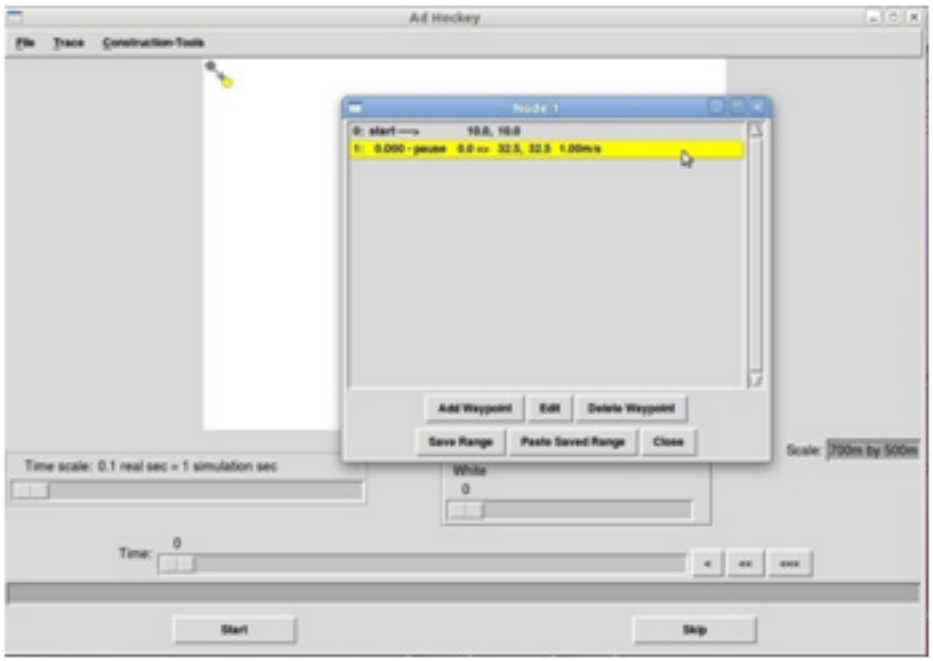


Figura a.2.34 Manipulación de los parámetros del nodo

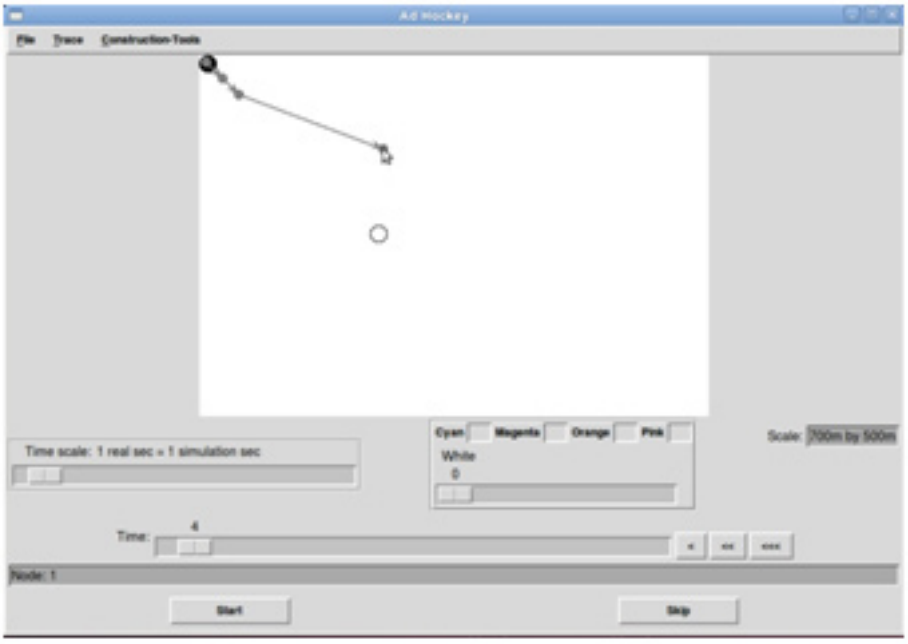


Figura a.2.35 Generación de la trayectoria del nodo.

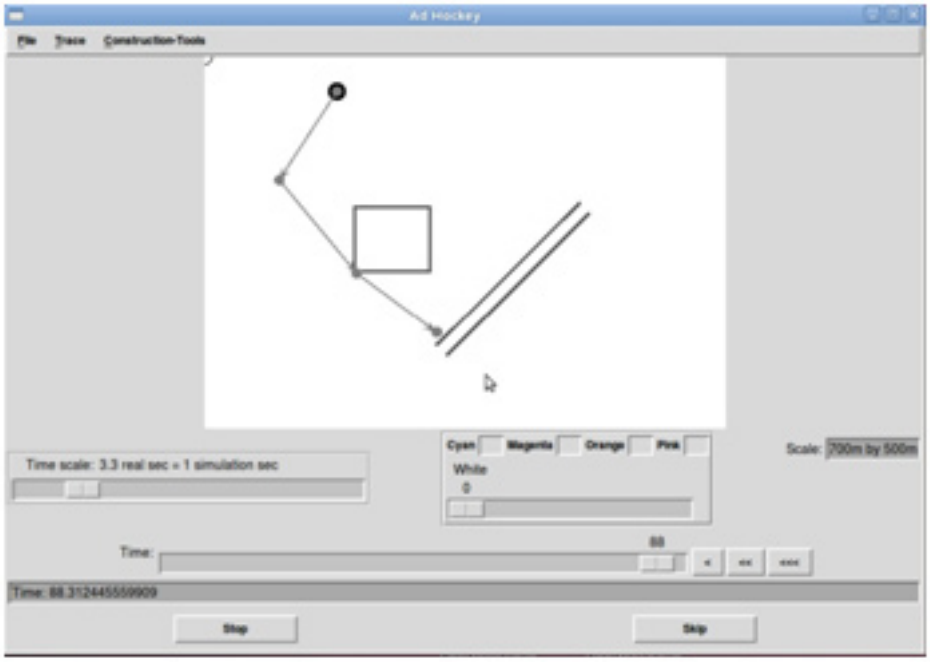


Figura a.2.36 Creación de obstáculos en *Ad hockey*

ANEXO 3

Generación dinámica de imágenes de desastres en áreas urbanas empleando el formato XBM

INTRODUCCIÓN

La representación dinámica y/o aleatoria de obstáculos a partir de una imagen de un segmento de un plano urbano real con el fin de ir guiando al observador y restringiendo el libre desplazamiento en una simulación de movilidad, demanda de un formato con cierta flexibilidad y facilidad para su comprensión interna sin reparar relativamente en la calidad visual de la imagen. Por lo que es aceptable trabajar en imágenes visualmente simples como las *XBitMap* (*XBM*) donde su formato compositivo maneja términos conocidos para cualquier programador como lo son: fila, columna y códigos hexadecimales.

En los libros o en el ciberespacio existe un tratado superficial del formato XBM debido a su simplicidad gráfica en comparación con el resto de formatos inclusive de su misma especie como: Escala de grises, BMP, etc. Debido a esto, el presente artículo es el producto de un análisis hecho en base al trabajo minucioso sobre la estructura de su formato, que inició con la premisa de poner en claro algunos aspectos de la composición de su formato en relación con su representación visual sobre el lienzo (área de la imagen) y la obtención visual de los obstáculos en forma aleatoria sobre la imagen.

CONTENIDO

IMÁGENES XBM

XBM es un formato gráfico que representa sus valores monocromáticos (blanco y negro) expresados en píxeles. En la informática se los considera como archivos binarios porque pueden ser representados como 0's o 1's, respectivamente.

FORMATO XBM

Los archivos XBM son editables como texto plano simplificando la observación de su estructura interna compuesta de la siguiente manera:

- Las primeras líneas, detallan el nombre del archivo, sus dimensiones y la definición de su matriz, entre otras cosas.

#define hip_width 2000

#define hip_height 1000

static char hip_bits[] = {

- El detalle de la representación gráfica se encuentra determinado por una sucesión de códigos que se disponen entre llaves {}.

{ 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,

0xAF….....};

ANÁLISIS DE LA MATRIZ DE CÓDIGOS

La matriz consta de un número n filas y de 12 columnas como se lo puede apreciar en la Figura a.3.1, pero su disposición gráfica sobre el lienzo de la imagen es diferente que obedecen a la siguiente fórmula:

nLinea = (nfilas x 12)/ largo en pixeles de la imagen (Figura a.3.1)

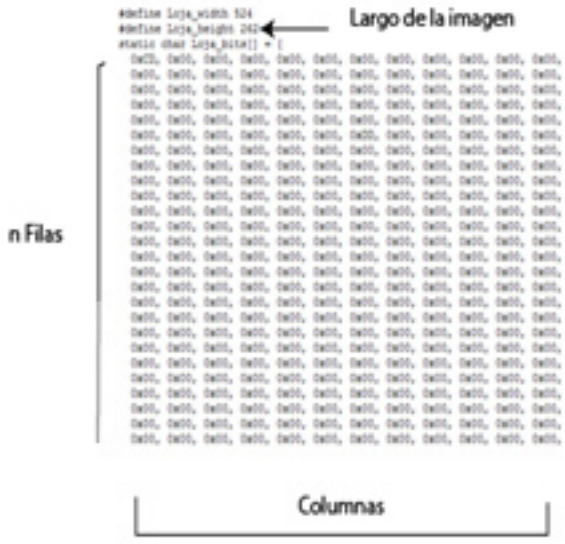


Figura a.3.1 Matriz binaria

N Línea es la cantidad de códigos que componen una línea de la imagen visualizada sobre el lienzo como se lo puede ver en la Figura a.3.2

.

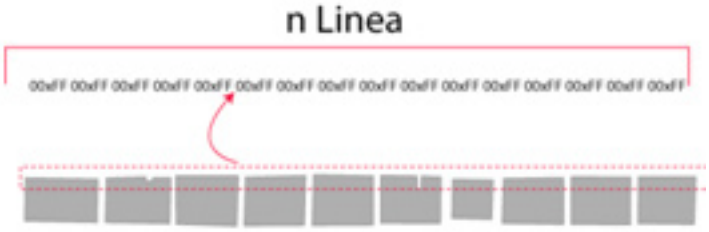
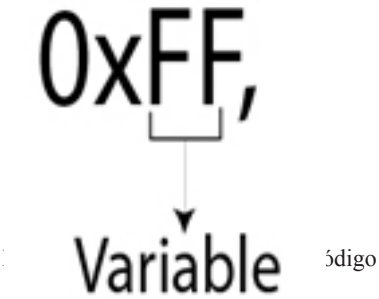


Figura a.3.2 Análisis de los códigos

Cada uno de los códigos ocupa el espacio de un Byte, en cada bit se aloja un píxel. Según la Figura a.3.3.

Figura a.3.3 Representación en binario de un bit de imagen en blanco

Los códigos se forman de la unión de 5 caracteres, 3 de los cuales son fijos y 2 variables como se indica la Figura a.3.4



ódigo

Los caracteres variables son valores hexadecimales que definen si un píxel se pinta o no de negro en el lienzo de la siguiente manera que ilustra Figura a.3.5:

Figuraa.3.5 Codificación binaria

GENERACION DINÁMICA DE OBSTÁCULOS SOBRE UNA IMAGEN XMB

Comprendidos los aspectos anteriores se puede comenzar a manipular la imagen en programación para inducir a la creación de los obstáculos de manera dinámica conociendo de antemano que:

- Los Obstáculos están definidos por los espacios ocupados en la imagen por las áreas pintadas de color negro o blanco dependiendo del manejo de la monocromía que se utilice para realizar el trabajo.
- La generación de los obstáculos en un desastre no es predefinida sino que son espacios ocupados de manera fortuita para ello se puede aplicar la utilidad randómica del lenguaje (C, C++,etc.) ó mecanismos que proporcionen la aleatoriedad por programación.
- De acuerdo a la forma como se disponga el valor de los obstáculos se debe de asignar su valor hexadecimal en los caracteres variables del código y de igual manera mediante el número y el tamaño de los obstáculos

DISCUSIÓN

El manejo de las imágenes a niveles bajos de resolución permite obtener un mayor dominio del formato en el momento de la programación que obviamente dista mucho de la facilidad de obtenerlos de manera estática a partir de herramientas gráficas a la cual se deberá recurrir para hacer un cambio. Se puede convertir los códigos hexadecimales en términos de 1 y 0, para trabajar con una gran matriz binaria donde cada número es un píxel y represente el criterio de ocupado o lleno.

Al utilizar una imagen de un segmento de una área urbana real conlleva a trabajar con formas poligonales que las encontramos en cualquier parte de nuestra ciudad a diferencia de la representaciones idealizadas mediante bloques cuadriláteros como es el caso del modelo Manhattan cuya movilidad está sujeta a este tipo de escenarios idealizados.

CONCLUSIONES

- El conocimiento de la relación entre la representación gráfica sobre el lienzo y el formato interno de la imagen facilita la manipulación de una imagen.
- El descubrimiento interno del trabajo de un formato de una imagen es complejo por lo que se debe trabajar con formatos simples.

- El formato XBM está compuesto de una estructura similar a la definición de una matriz estática del lenguaje C.
- Cada línea del lienzo de la imagen está conformado por una cantidad fija de bytes.
- La graficación de los obstáculos en el lienzo está determinada por la repetición de unidades de píxeles con el mismo valor cromático que la de las áreas ocupadas del plan.

ANEXO 4

Unmarked Point and Adjacency Vertex, mobility models for the generation of emergency and rescue scenarios in urban areas

Unmarked Point and Adjacency Vertex, mobility models for the generation of emergency and rescue scenarios in urban areas

ROMMEL TORRES[1]*, JUAN CARLOS SANCHEZ[1]†, LUIS MENGUAL GALAN[2]‡

[1] *Networks and Telecommunications Laboratory, Universidad Técnica Particular de Loja, ECUADOR*

[2] *Departmenot de Lenguajes y Sistemas Informáticos e Ingeniería de Software, Universidad Politécnica de Madrid, SPAIN*

In disaster situations, where ad hoc mobile networks are normally used, the location and quantity of existing obstacles is random. Most existing mobility models have not been developed with consideration of the obstacles that exist in a disaster environment. This paper proposes two methods of mobility that realistically represent movement in an environment with obstacles. Unmarked Point Model (UPM) uses a high granularity strategy and Adjacency Vertex Model (AVM) uses a method that selects the shortest pathway. UPM consumes more resources than AVM to generate node mobility patterns in an ad hoc network, on a real map area of an urban area, where obstacles have been placed to simulate an emergency and rescue scenario. In addition, a comparative analysis of both models in its routing performance using AODV[16].

Key words: Wireless, ad hoc, mobility models, simulation, urban areas, emergency and rescue.

* email: rovitor@utpl.edu.ec
† email: jcsanchez@utpl.edu.ec
‡ email: lmengual@fi.upm.es

1 INTRODUCTION

In an emergency and rescue scenario the ad hoc and MANET mobile networks can create a temporary wireless network from mobile nodes without requiring a preset network infrastructure[6].

The use of MANET networks in a disaster area allows the use of everyday technological devices such as cell phones, PDAs, tablets and other mobile devices [12] to establish communication between both rescue teams and survivors. So that, independently and collaboratively, the generation of a temporary network is achieved in order to lessen the effects of the disaster.

The use of simulation tools allows us to determine and analyze the behavior of MANET networks without incurring the expenditure of resources [12]. A key element in the simulation is the generation and use of a mobility model to determine the behavior of each displacement node [6].

Mobility models must simulate human movement [15], because in emergency and rescue scenarios, especially in urban areas, there are obstacles that prevent the movement of vehicles and limit the movement of people. [6] Each of these obstacles is not only a problem for the free movement of people, but also in attenuating signals and even preventing network connections between nodes.

There are tools to generate node movement in emergency and rescue scenarios in visual form, such as Ad-hockey [1], however they use perfect geometric shapes such as quadrilaterals and simple lines for obstacles, causing the node to interact in an artificial environment.

This paper proposes two models of human mobility: the first is called UPM, inspired by basic deductions of "free space - occupied space" and the second is called AVM and is based on human movement (HUMO) [6] and the use of the shortest path concept (Dijkstra). The implementation of the two proposed models is performed in the SCENGEN [17] scenario generation tool.

To validate the models, we propose a method of generating emergency and rescue scenarios using real maps to simulate the state of the city after a disaster.

The paper is structured as follows: Section II discusses the types of mobility models. In Section III, we propose the UPM mobility model. Section IV proposes the AVM mobility model. Section V shows the process of generating an emergency and rescue scenario and finally, in Section VI, we compare the proposed models by measuring the computational performance and AODV behavior within a simulation in Network Simulator 2 (NS2)[7].

2 RELATED WORK

Mobility models generally fall into two groups: models based on traces that are based on actual movements [7] captured by electronic devices and stored in large databases of information [20] [14] [4]. There are also synthetic models, whose behavior is based on the use of mathematical models [14].

Synthetic models can be classified according to their similarity to human movement, in realistic synthetic models and unrealistic synthetic models. The realistic synthetic models are divided into models with spatial dependency and models with geographical dependency. Within the unrealistic synthetic models, random models and models with temporal dependency can be found.

Random Models: These are models where the nodes have free movement in the simulation area. Its direction, speed, and acceleration characteristics, etc. are randomly generated causing abrupt changes in node movement. These models include Random Walk Mobility Model, Random Way-Point Model, Random Direction Model and Boundless Simulation Area Model. Random Walk Mobility Model is the basis of several random patterns and its movement is considered the most unpredictable of all [5][19]. The Random Way-Point Model is the model used for the simulation of protocols in ad-hoc networks [13].The Random Direction Model seeks to break the concentration of dots in the center of the area [5]. The Boundless Simulation Area Model uses the relationship between the previous direction and speed with the current one and works within a limitless area so that a node that leaves the area's boundary goes back to its opposite side [5].

Models with Temporal Dependency: These models try to correct the weaknesses of realism of previous models, such as the sudden stop and the drastic change of speed and direction, using a comparison process between the current and next values of speed, acceleration and direction. They are representations of mobility that do not conform to human patterns [5] [9].Some of these models are: the Gauss-Markov Mobility Model and the Smooth Random Mobility Model. The Gauss-Markov Mobility Model was designed to adapt to different levels of randomness via tuning parameters. A node starts with a speed and an assigned direction, which after a certain period of time is reassigned [5]. The Smooth Random Mobility Model, controls sudden changes of speed to make a change in direction through the use of a slowdown before time runs out [3].

Models with Spatial Dependency: These include the mobility models that handle node groups that are commonly governed by the movement of a node [6] [10]. Among the models that belong to this group are: Reference Point

Group Mobility Model, Column Mobility Model, Pursue Mobility Model and Nomadic Mobility Model. In the Reference Point Group Mobility Model, the group movement is determined by an arbitrary motion model while its internal movement (each node) is associated with a reference point [10][8]. In the Column Mobility Model, the node model is based on movement along a straight line (grid reference) that moves in a given direction [5]. The Pursue Mobility Model uses the same strategy used by police to track a thief. Once implemented in the model, the nodes chase after a target [5].The Nomadic mobility model, its name evokes the distant, past human behavior when people traveled in wandering groups governed by a leader [5].

Models with Geographic Dependency: The models with geographic restriction emphasize the solution of stage limitations, which force an efficient mobility of nodes. There are two variants of models with geographic dependency: the Pathway Models and the Obstacle Models. Pathway type models are developed for scenarios that are based on obtaining pathways for the movement of nodes to reach their target. These pathways can be representations of city streets, roads or highways, which depending on the flexibility of the established scenario, must abide by restrictions on the roads such as speed limits and traffic lights, among others [14][7].The Obstacle Models are models based on solutions obtained from the defined obstacles in a scenario. Generally, this achieves the implementation of solutions based on movements in its vertex, such as the Human Obstacle Mobility Model(HUMO), using mathematical models or the use of graphical modeling [14].

The two proposed models, UPM and UVM, belong to the group of realistic synthetic mobility models with geographical restriction.

The UPM is a type of Obstacle Model because it handles the analysis of "free - busy" space to make its movement and thus generates its pathway. The UPM is different from the rest of the models because it can work on scenarios that possess obstacles of non-geometrical or amorphous shapes, because its movement is based in the low level treatment of the image, that is, in scenarios whose images of obstacles present great granularity or pixelation, as shown in Figure 1.

The AVM uses the Pathway Model and Obstacle Model strategies because its movement is given by the existence of points defined on the obstacles and also because the node has to make a quantitative comparison of distances to make its next move. This movement is similar to HUMO movements [15] [6] and its improved variant, Obstacle Avoidance Mobility (OAM) [6]. AVM is also Pathway because nodes move according to previously established path-

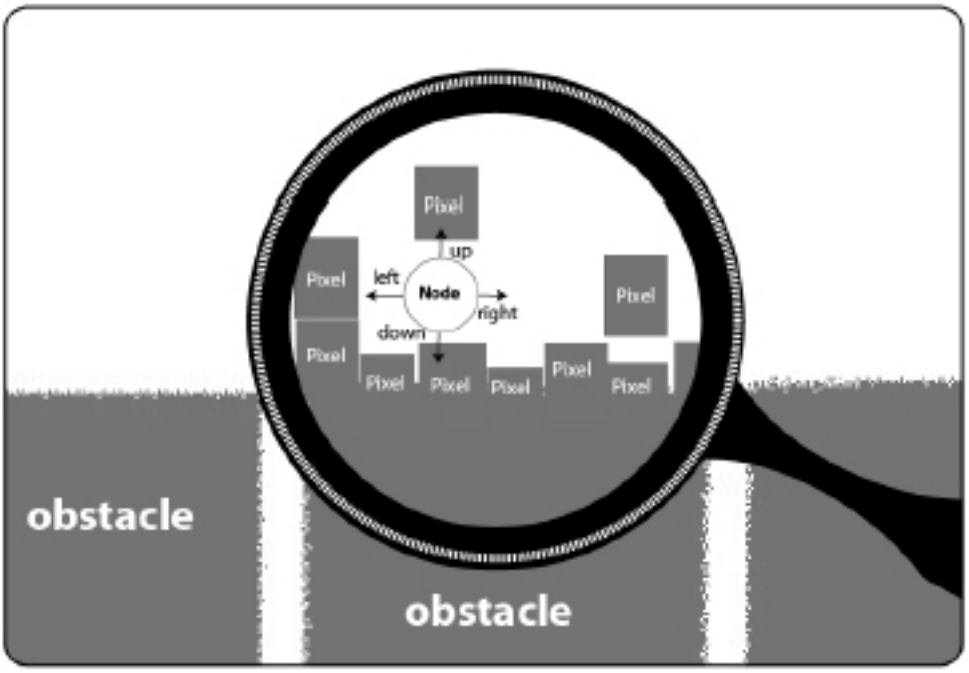


FIGURE 1
Pixelated surfaces

ways on an adjacency matrix [17].

The AVM differs from the other models because it uses an adjacency matrix of vertex, which enables the formation of an area of very simple or complex obstacles, through modifying adjacency points or eliminating vertex.

3 UNMARKED POINT MODEL (UPM)

The UPM was developed by taking into consideration the analysis that a person has to make in a scenario crowded with obstacles similar to those in an emergency and rescue scenario. This is because an individual must analyze a range of possibilities before making their next move, much the same as they would do before crossing a river. Node movement is done through verification and use of open spaces adjacent to the node's position in any of the four cardinal points.

Given a node's current position, the following is used to define its next position:

Obstacles

The obstacles are randomly distributed in a binary matrix generated by a map of a real city, to simulate a disaster area, as shown in Section V, which describes the generation of a disaster scenario. Each occupied space is represented with a 1 and the free space is represented with a 0. This makes it possible to distinguish the presence of an obstacle or a point that can become part of the pathway to the target point.

Movement

The node starts from a randomly selected position through the application of a uniform distribution that is responsible for finding a point on the matrix with a value of 0. It then evaluates the four closest values to the current position, always estimating the pathway to approaching a defined target point, which in the case of an emergency and rescue scenario, could be an evacuation or aid point. The target point is the destination to be reached. The movements performed are made in a row or a column. This process is shown in Figure 3.

Pathway

UPM is a very thorough way to move as it works in areas with many obstacles. Unlike models such as HUMO or OAM, it is not based on the vertex of the polygon, but on the state of the next free position as indicated by Algorithm 1. In the evacuation point of this model, if a node is in a position (x_a, y_b) to

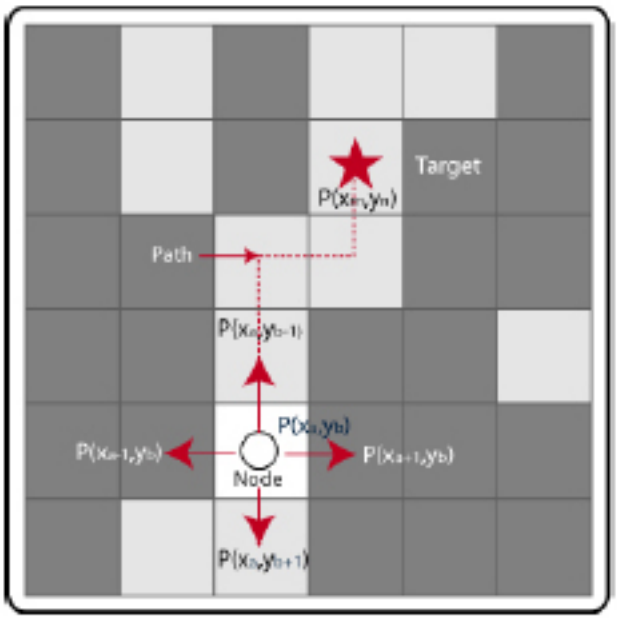


FIGURE 2
UPM node movement

approach its target point in (x_n, y_n), it has to choose the free space of its next positions (x_{a-1}, y_b), (x_{a+1}, y_b), (x_a, y_{b-1}) y (x_a, y_{b+1}); which is convenient to reach the position (x_n, y_n) as shown in Figure 2.

4 ADJACENCY VERTEX MODEL(AVM)

The AVM is similar to the OAM model as it is based on the human mobility model (HUMO) and also applies the concept of the shortest path to reach the target. However, it is organized into two operating strategies that make it a hybrid model, i.e. a model simultaneously based on obstacles and pathways. The first strategy is used for the generation of the obstacles and their pathways, and the second strategy is to generate the movement of randomly distributed nodes after a disaster.

Given a node's current position, the following is used to define the next position:

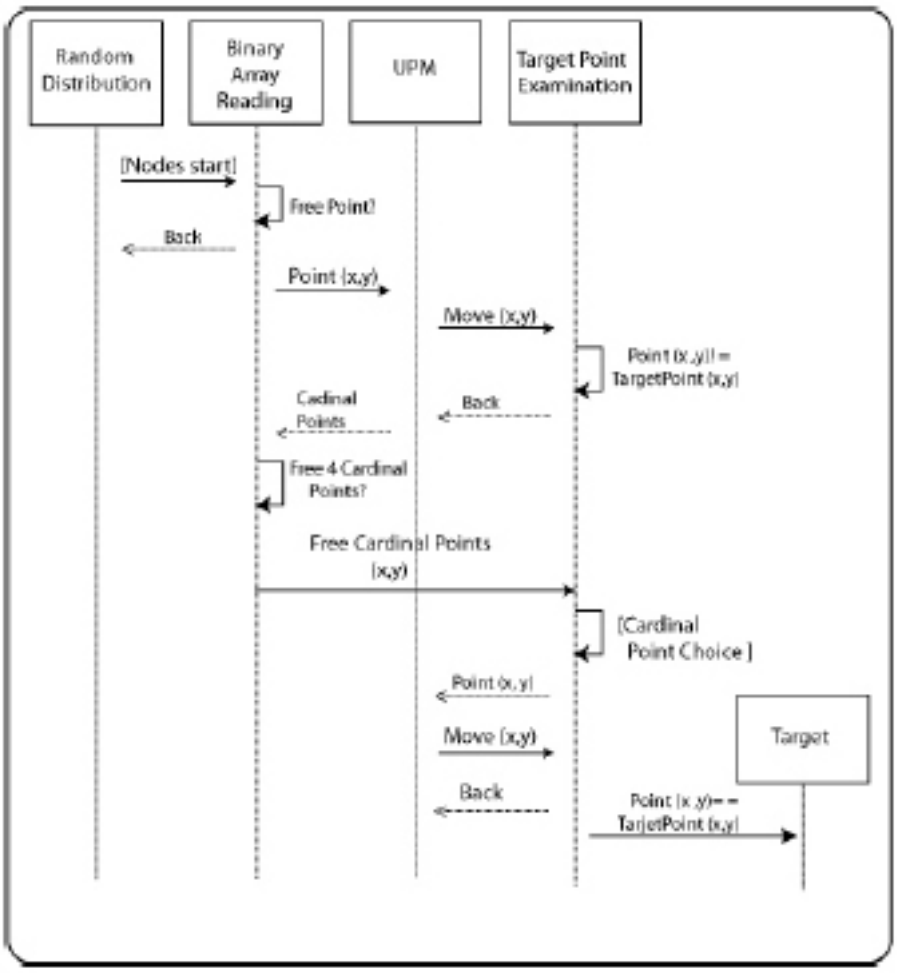


FIGURE 3
UPM node movement diagram

Algorithm 1 UPM node movement.

Require: A binary matrix where it is detailed free and occupied spaces.

Ensure: Next nodes movement point (x,y).

1: **if** $(((pX < pY) \wedge (p\_target\_X <> pX)) \vee (p\_target\_Y == pY))$ **then**
2: **if** $(p\_target\_X < pX)$ **then**
3: **if** $(left\_free)$ **then**
4: $(node\_left\_move)$
5: **else**
6: $(node\_down\_move) \vee (node\_up\_move) \vee (node\_right\_move)$
7: **end if**
8: **else**
9: **if** $(right\_free)$ **then**
10: $(node\_right\_move)$
11: **else**
12: $(node\_down\_move) \vee (node\_up\_move) \vee (node\_left\_move)$
13: **end if**
14: **end if**
15: **else**
16: **if** $(p\_target\_Y < pY)$ **then**
17: **if** $(down\_free)$ **then**
18: $(node\_down\_move)$
19: **else**
20: $(node\_left\_move) \vee (node\_up\_move) \vee (node\_right\_move)$
21: **end if**
22: **end if**
23: **end if**

Obstacles

These are randomly distributed obstacles in a file that contains a binary matrix which represents a segment of a real city map, simulating a disaster area. The vertex of each polygon are marked with a digit other than 0 and 1. Each vertex contains a number of adjacent vertex that are close to the visible points which can be reached directly. This information is stored in an adjacency matrix.

Movement

The node starts from a position selected randomly by a uniform distribution and immediately calculates the nearest vertex position by quantitative comparison of the results of the distance between the vertex position and the target orevacuation point. It does this using the Euclidean distance formula between two points, as indicated by Algorithm 2.

Pathway

The pathway is generated from a starting point to another target through the shortest jump between vertex of the obstacles, as defined in the algorithm as the shortest distance. However, it must do this by using an adjacency matrix based solely on the comparison values of distances and avoid it if it a vertex has already visited it, because the hops are already linked and the node is dynamically built to find the shortest pathway to its destination, as indicated in the Figure 4 and Figure 5.

5 GENERATION OF DISASTER SCENARIOS BASED ON REAL MAPS

For the generation of the scenario on which to validate the UPM and AVM models, we will use a segment plan of a real city, 2 km x 1 km, located in Loja city, Ecuador in South America, a panorama of the city shown in Figure 6.

Based on a real map of the city, we will proceed to select the appropriate area and start to edit the image, removing the architectural details in the resource graph in a way that removes everything that is unnecessary and only leaves blocks of the structures. After that, it is changed to a bitmap format called Exchange monochrome BitMAP (XBM) for the ease of its internal composition as a binary matrix representing the scenario. This procedure is summarized in Figure 7.

The internal code of the image is transformed using a program developed in C++ binary data that will help in controlling the mobility of the UPM.

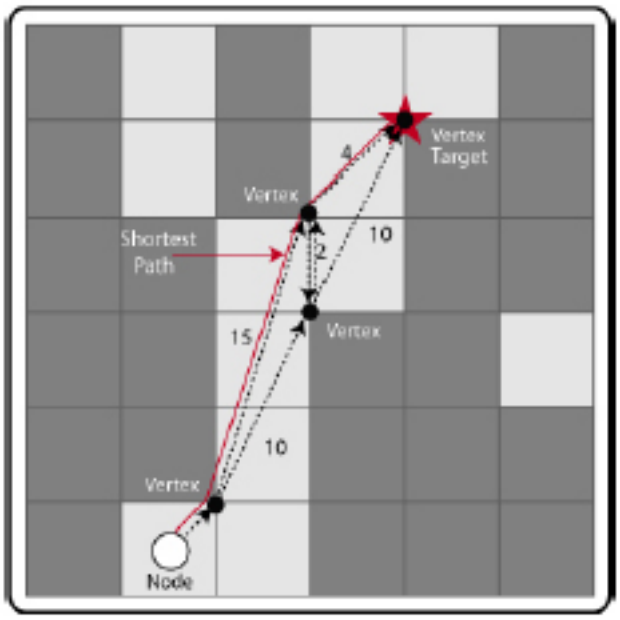


FIGURE 4
AVM node movement

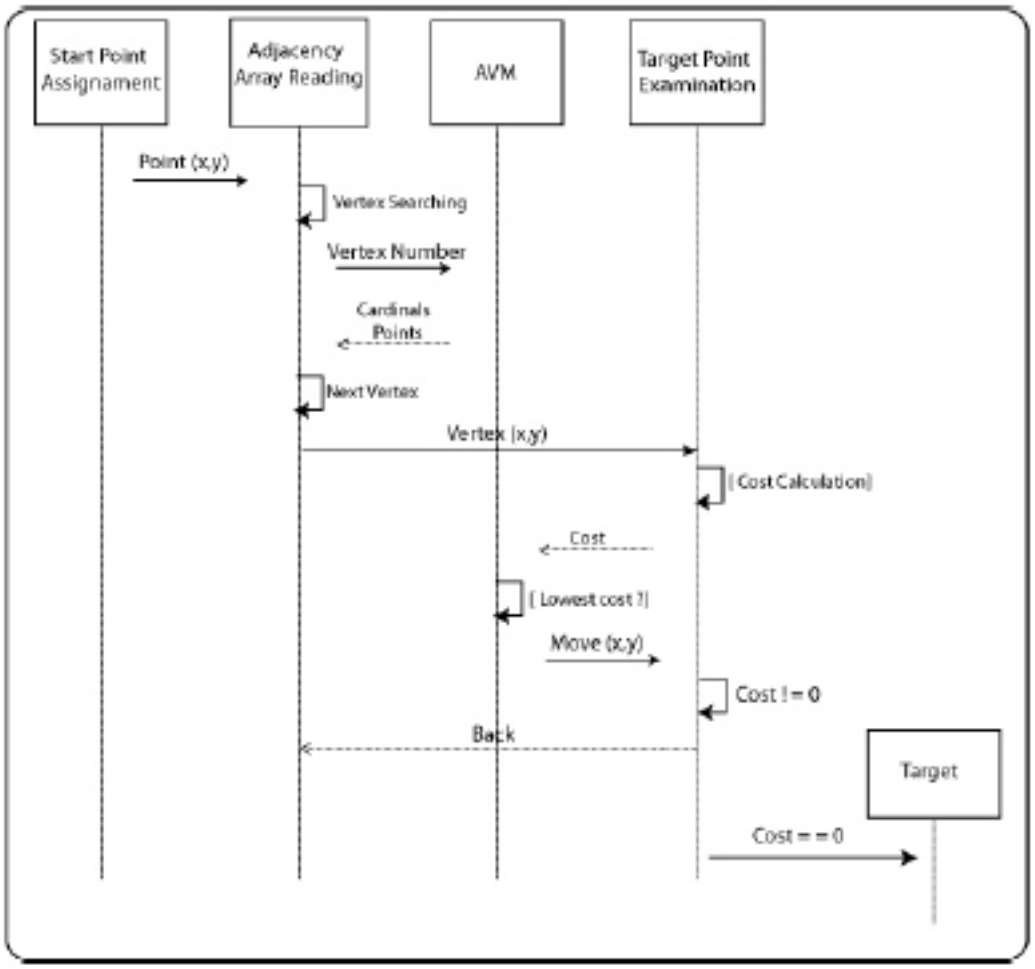


FIGURE 5
AVM node movement diagram

Algorithm 2 AVM node movement.

Require: An adjacency matrix

Ensure: Next nodes movement point (x,y).

1: **for** $(var2 \leftarrow 0, ..., num\_max\_nodes)$ **do**
2: **if** $((pX == max[var2][0]) \wedge (pY == max[var2][1]))$ **then**
3: $start\_node2 \leftarrow var2$
4: $var2 \leftarrow num\_max\_nodos$
5: **end if**
6: **if** $(pX <> max[target\_node][0]) \wedge (pX <> max[target\_node][1])$ **then**
7: **for** $(var32 \leftarrow 0, ..., num\_max\_colum[2])$ **do**
8: $a = max[start\_node2][var3] - max[target\_node][0]$
9: $b = max[start\_node2][var3 + 1] - max[target\_node][1]$
10: $cost \leftarrow \sqrt{[(a)^2 + (b)^2]}$
11: **if** $(var3 == 2))$ **then**
12: $mincost \leftarrow cost$
13: $pX \leftarrow max[start\_node2][var3]$
14: $pY \leftarrow max[start\_node2][var3 + 1]$
15: **else**
16: **if** $(cost < min\_cost)$ **then**
17: $min\_cost = cost$
18: $pX \leftarrow max[start\_node2][var3]$
19: $pY \leftarrow max[start\_node2][var3 + 1]$
20: **end if**
21: **end if**
22: **end for**
23: **end if**
24: **end for**

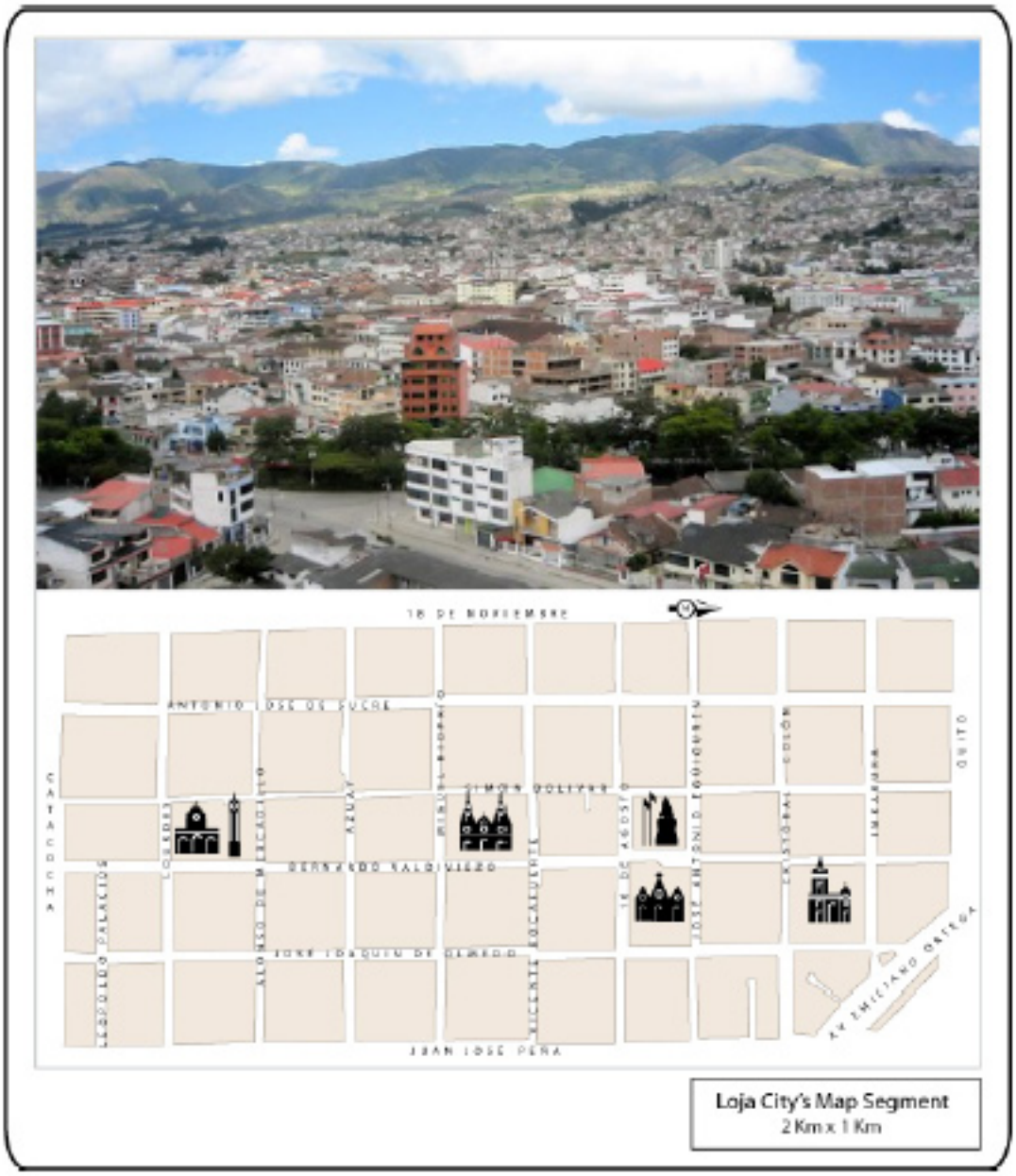


FIGURE 6
Abstraction of one segment of the map of Loja city-Ecuador

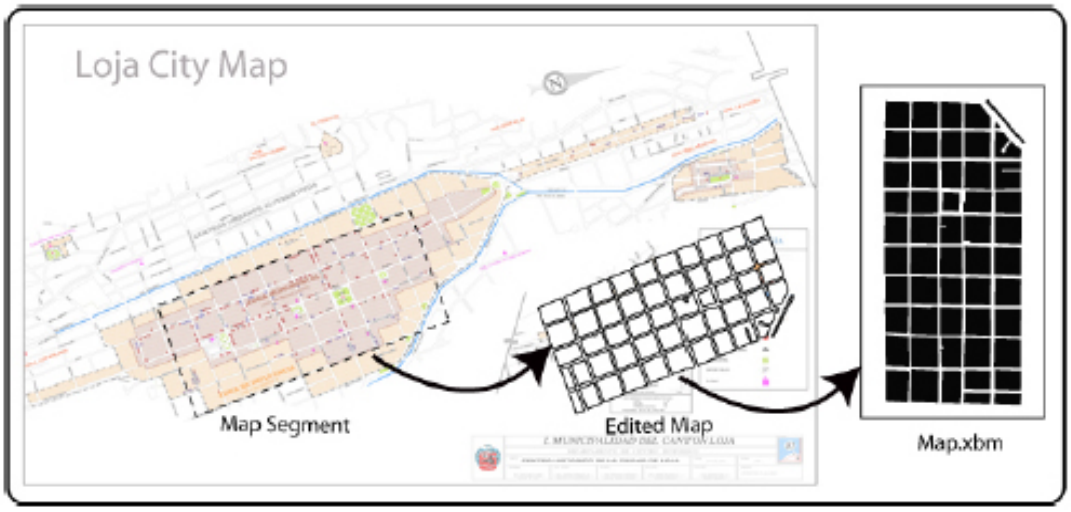


FIGURE 7
Obtaining XBM image file

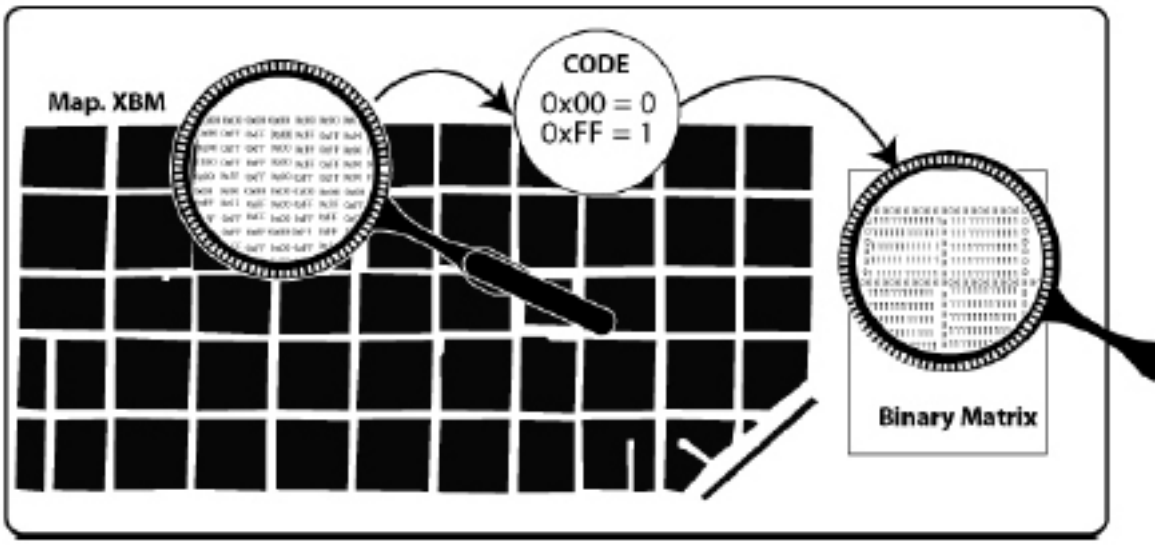


FIGURE 8
Binary representation of the XBM image file

Figure 8 summarizes the expressed idea.

Manipulating the binary file obtained above and using a program developed in C++ randomly generates obstacles in the image. This result is indicated in Figure 9.

Below this image we receive over 30 obstacles Figure 10 and about 234 vertex. Then, by observing the resulting image, an adjacency matrix is generated that is used by the AVM.It shows the process by taking as its starting point the vertex 141.

UPM and AVM have been implemented in the SCENGEN tool. The software developed in C++ generates scenarios for the NS2 tool. SCENGEN has implemented 6 models of mobility, none is of the geographic restriction type and data are generated randomly using uniform distribution or the Gauss-Markov distribution. There are many simulation tools, among them are:

- Scengen is an application developed in C++ to generate mobility scenarios in a fast and simple way.
- NS2 environment was developed C++ and TCL. It's widely used for academic research community for creating, protocol evaluating, Models communication simulating and more [18].

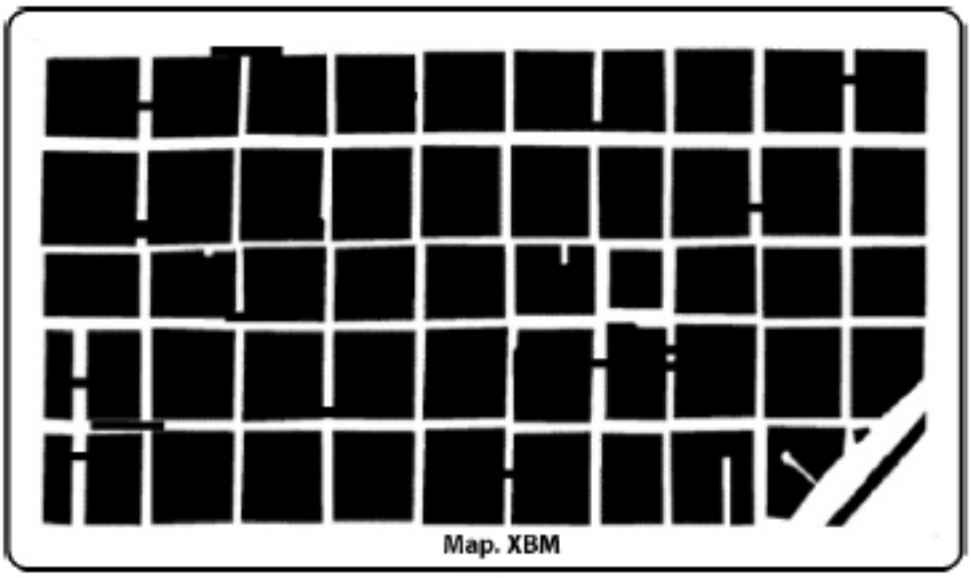


FIGURE 9
Disaster area with obstacles generated (Map.XBM)

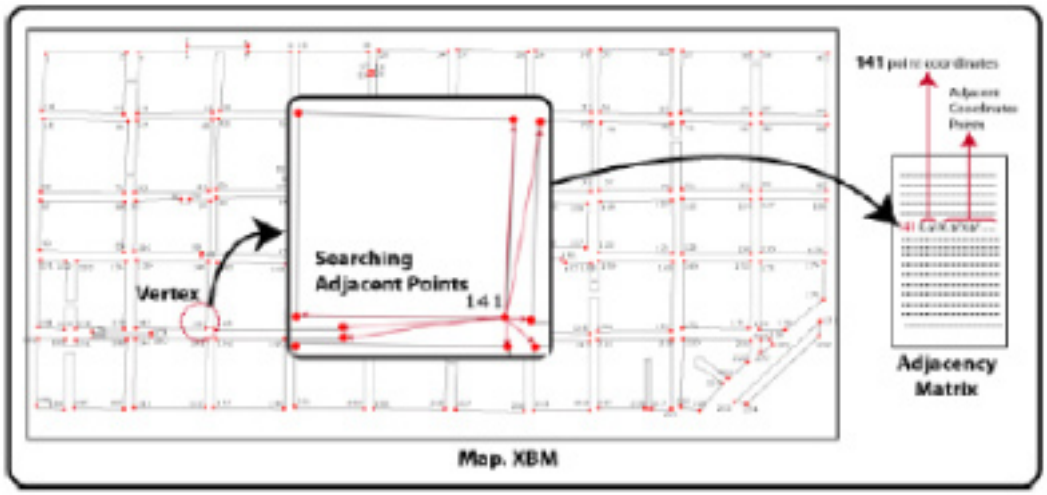


FIGURE 10
Obtaining of adjacency points matrix file

- NS3 is more modern and eliminating complex produced by C++ and TLC. It has a few mobility model implemented than NS2. [18].
- BonnMotion is a program made in JAVA oriented for the creation and analysis of mobility scenarios [2].
- Mobi-Real is a tool for realistic simulation of human and vehicle movement with variation of behavior depending on the context of the application [11].

The SCENGEN tool was chosen for the following features:

- It is an application that creates mobility scenarios for the NS2 simulation tool.
- Its code is open, light, available online and fully developed in C++.
- It allows new patterns of mobility.
- It facilitates the visualization of generated movements.

6 EXPERIMENT RESULTS

6.1 Computational Performance

The computational performance is the evaluation of the efficiency of resource use hardware and software of a computer in front of a task. We use three metrics: the scenario generation time, processor usage and number of generated lines. Table 1 and Table 2 show the computational performance for UPM and AVM changing the node density.

Node density	Scenario generation time (s)	Processor Usage (%)	Number of generated lines
20	7.261	100	440
40	13.476	100	904
60	19.698	100	1361
90	28.225	100	2032

TABLE 1
Data obtained with UPM

In Figure 11, we can see a marked difference in runtime between the two models. The high consumption of time in each UPM runtime tends to

Node density	Scenario generation time(s)	Processor Usage(%)	Number of generated lines
20	0.137	25	203
40	0.287	41	419
60	0.459	62	698
90	0.568	80	920

TABLE 2
Data obtained with the AVM

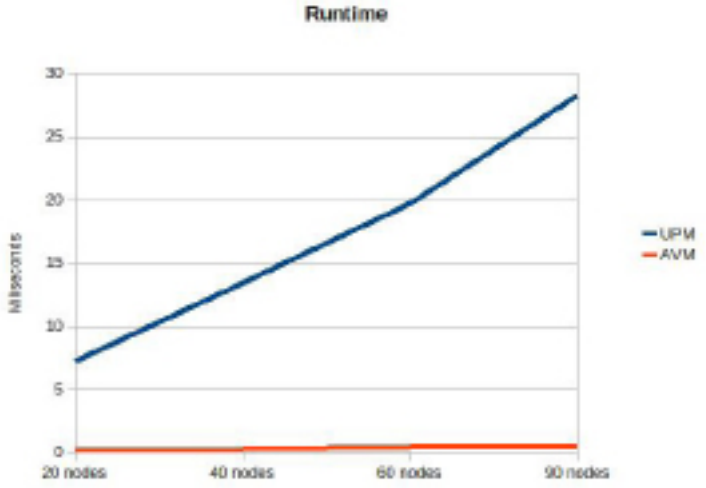


FIGURE 11
Generation time of a scenario

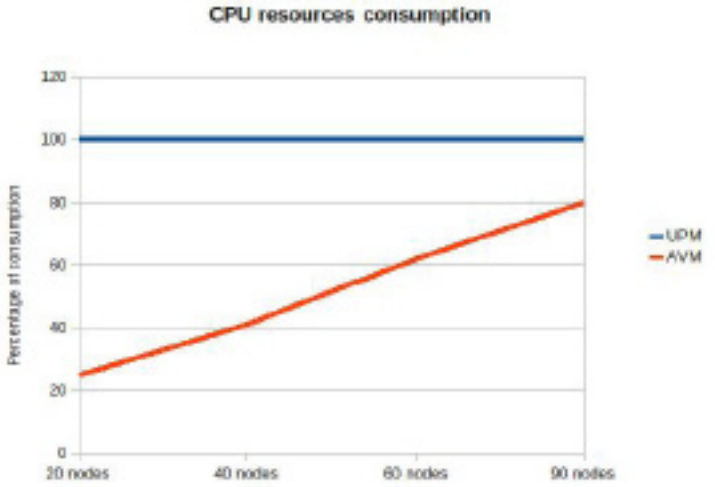


FIGURE 12
CPU utilization

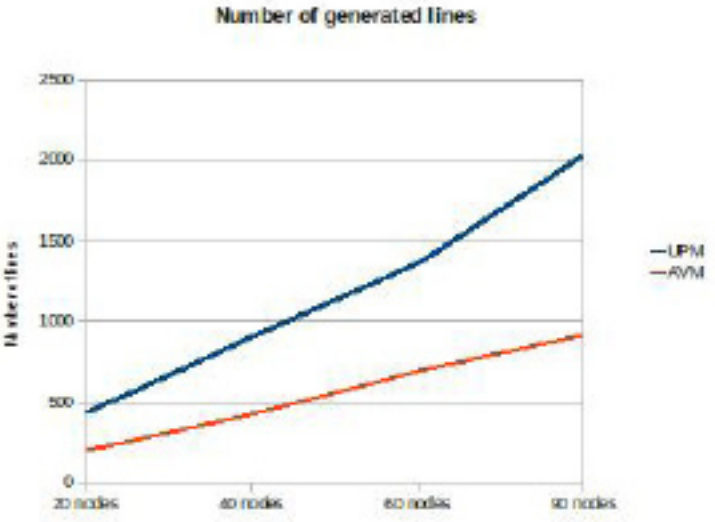


FIGURE 13
Number of generated lines

increase steadily in relation to the increasing number of nodes, while in the AVM, runtime is minimal.

Figure 12 indicates the consumption of processor resources. The UPM uses all the processor resources, while for AVM, the use of processor resources is directly related to the number of nodes.

Finally, Figure 13 shows that the UPM generates a greater amount of movements than the AVM. The amount of movement directly affects the simulations performed in the NS2 simulator.

In conclusion it can be determined from the above that AVM is a more efficient use of resources against UPM because it has a movement so detailed that very resource demanding.

6.2 Relationship with the ad hoc routing protocol

Since the purpose of mobility models is a direct influence on the efficiency of a protocol in a network is necessary to measure their behavior by analyzing a controlled simulation.This is going to use the parameters to the list in Table 3 below:

Item	Detalle
Routing protocol	AODV
Nodes density	20, 40, 60, 90
Mobility models	UPM, AVM
Evacuation point	Static in the map
Number of connections	20 connections (All to Node 0)
Dynamic Viewpoint Static Viewpoint	Node 0 moves Node 0 doesn't move

TABLE 3
Simulation parameters

The simulation was developed in the NS2 using the previously created scenarios and 20 TCP connections for evaluating the AODV protocol performance.

Evaluation of each viewpoint are done for responding the following metrics: rate received packets, delays average, application layer efficiency, routing layer efficiency and dropped packets.

According to Dynamic viewpoint, Node 0 and rest of nodes are moving simulating the behavior of the survivors in a disaster area that communicate with each other for mutual aid to reach the destination which is an evacuation point. In the Static viewpoint Node 0 is located at the evacuation point

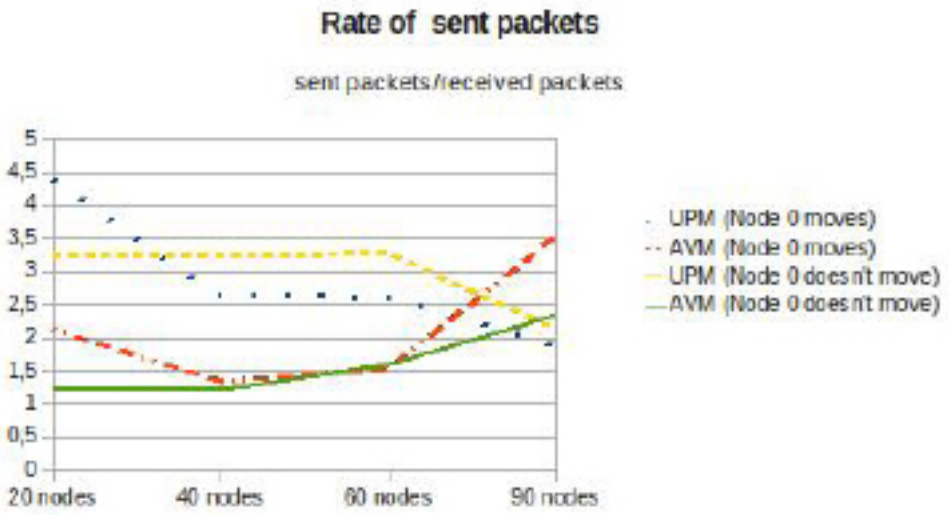


FIGURE 14
Rate of sent packets

without moving anywhere but stay connected to all nodes pretending to be an emergency and rescue equipment. The following is an analysis of the two models of mobility through the two points of view indicated.

Figure 14 illustrates the behavior of both models tend to stay in both cases. UPM will be stabilized first and then for values greater than 60 nodes plummets approaching the ideal value of the connection. AVM part of relatively more convenient because they are closer to the ideal value but on reaching 60 nodes tends to increase steadily away from the ideal value.

Figure 15 shows UPM suffers less delay packets within communication but upon reaching 60 nodes tends to increase slightly its rate. By having such low values may indicate the beginning there is no communication. While AVM maintains high values being when Node 0 does not move the value soars.

According to Figure 16, with high values UPM starts sending packets until around 40 nodes and 60 nodes stabilizes tends to fall to ideal values. Unlike AVM that starts with values relatively close to the ideal value but increases as the number of nodes tends to rise until it reaches 60 nodes where their value increases substantially.

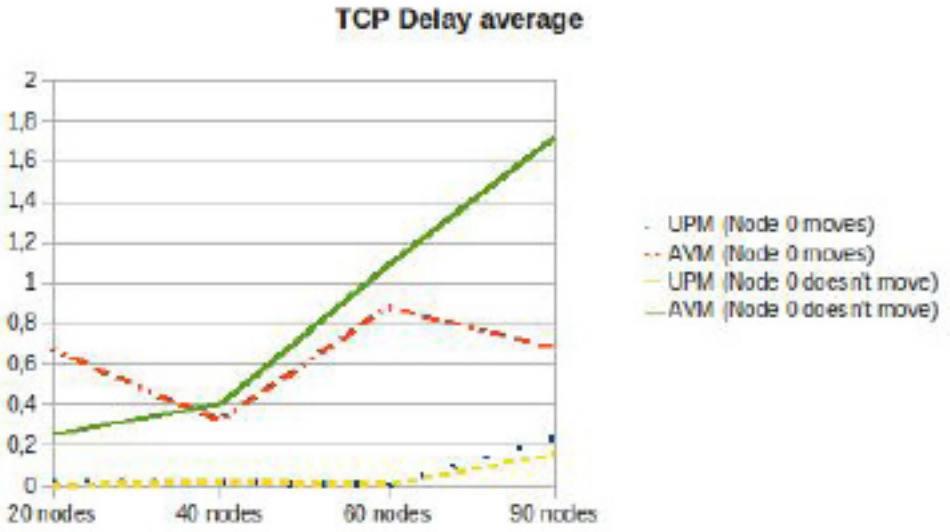


FIGURE 15
TCP delay average

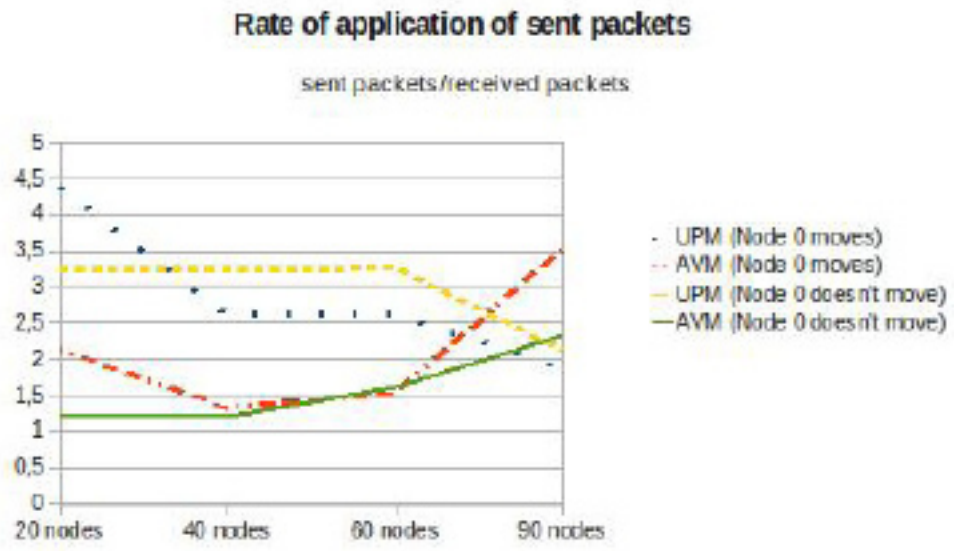


FIGURE 16
Application sent packets

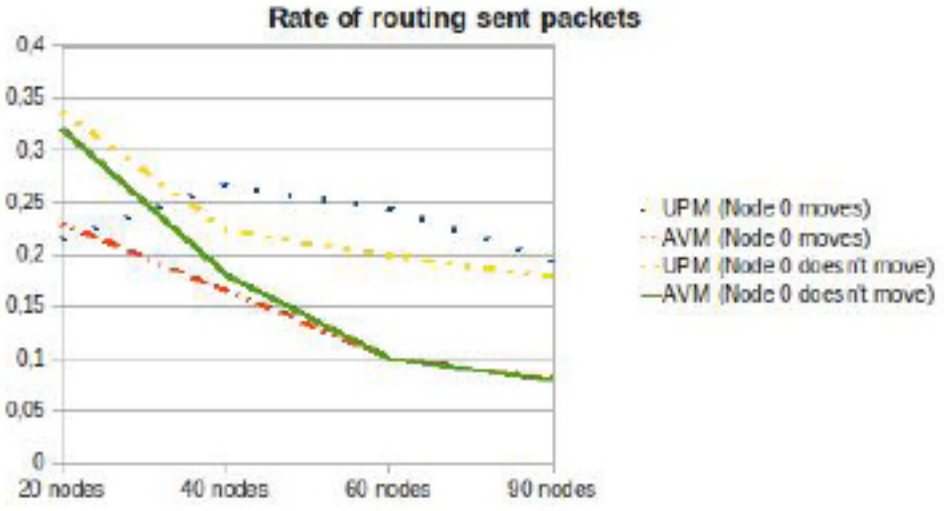


FIGURE 17
Rate of routing sent packets

As displayed in Figure 17, the two models are a greater number of packets they send. UPM, is closer to the ideal value when the node 0 does not move but falls like AVM to reach 60 nodes but maintains a broad range of distance between the graphs of the two models.

As seen in Figure 18, UPM maintains a very stable behavior within the network and the changes that occur are minimal compared to AVM which tends to increase according to set of nodes increased up enough when reaching node 0 moves and the number of nodes is greater than 60 nodes.

7 CONCLUSIONS

In this paper we present two models of human mobility that can be applied to the simulation of emergency and rescue scenarios. Both handle a large volume of obstacles and pathways. They are perfectly scalable and depending on the model, require more or less machine resources. The AVM constitutes the most efficient model because it uses fewer resources and is faster than the UPM. It is therefore the most suitable for the generation of simulated emer-

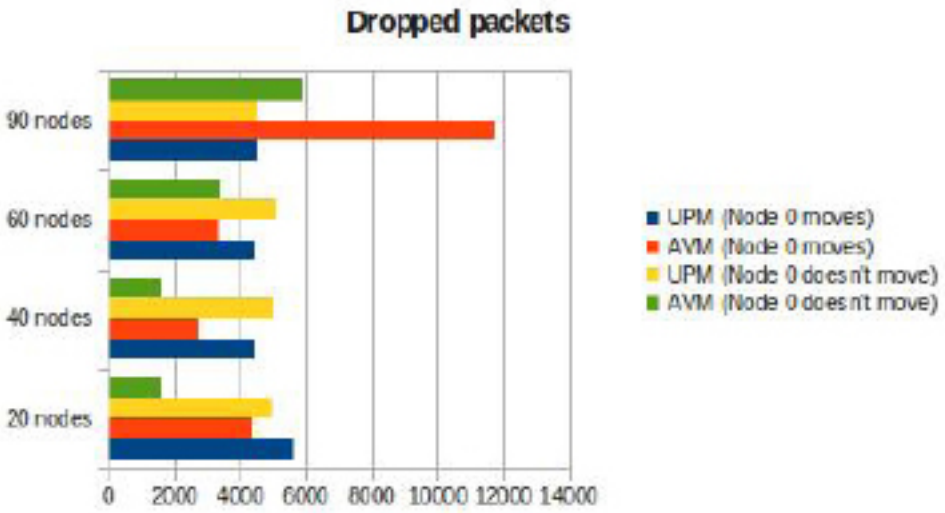


FIGURE 18
Dropped packets

gency and rescue scenarios. The UPM is useful for areas with obstacles that do not have a clear geometric shape such as emergency and rescue scenarios where the intensity of the disaster may directly affect the shape of the obstacle.

The convenience of AVM shown during the simulation using AODV protocol confirms the effectiveness and efficiency advantages offered by this model against the UPM for low node density. Although as seen UPM offers advantages enabling it to provide detailed work on the scenario mainly maintaining a good flow of communication with low delay rates and management of greater stability in the network.

REFERENCES

[1] The Monarch Project `http://http://www.monarch.cs.rice.edu`.

[2] Nils Aschenbruck, Raphael Ernst, E Gerhards-Padilla, and Matthias Schwamborn. (2010). Bonnmotion - a mobility scenario generation and analysis tool. *Work*, page 110.

[3] Nils Aschenbruck, Elmar Gerhards-Padilla, and Peter Martini, (2008). A survey on mobility models for performance analysis in tactical mobile networks.

[4] Nils Aschenbruck, Aarti Munjal, and Tracy Camp. (May 2010). Trace-based mobility modeling for multi-hop wireless networks. *Computer Communications*, 34(6):704–714.

[5] Tracy Camp, Jeff Boleng, and Vanessa Davies. (2002). A survey of mobility models for ad hoc network research. *Wireless Communications and Mobile Computing. Special issue on Mobile Ad Hoc Networking. Research, Trends and Applications*, 2:483–502.

[6] Yu Chenchen, Li Xiaohong, and Zhang Dafang. (oct. 2010). An obstacle avoidance mobility model. In *Intelligent Computing and Intelligent Systems (ICIS), 2010 IEEE International Conference on*, volume 3, pages 130 –134.

[7] Andrea Clementi, Angelo Monti, and Riccardo Silvestri. (August 2011). Modelling mobility: A discrete revolution. *Ad Hoc Networks*, 9(6):998–1014.

[8] S. Cristaldi, a. Ferro, R. Giugno, G. Pigola, and a. Pulvirenti. (May 2011). Obstacles constrained group mobility models in event-driven wireless networks with movable base stations. *Ad Hoc Networks*, 9(3):400–417.

[9] Bhavyesh Divecha, Ajith Abraham, Crina Grosan, and Sugata Sanyal. Impact of node mobility on manet routing protocols models.

[10] Yanying Gu, R. Venkatesha Prasad, and Ignas Niemegeers. (December 2009). Mobility Modeling for Personal Networks. *Wireless Personal Communications*, 58(2):169–196.

[11] K. Konishi, K. Maeda, K. Sato, A. Yamasaki, H. Yamaguchi, K. Yasumoto, and T. Higashino. (sept. 2005). Mobireal simulator-evaluating manet applications in real environments. In *Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2005. 13th IEEE International Symposium on*, pages 499 – 502.

[12] Franck Legendre, Vincent Borrel, Marcelo Dias de Amorim, and Serge Fdida. (2006). Modeling mobility with behavioral rules: The case of incident and emergency situations. In *AINTEC*, pages 186–205.

[13] Guolong Lin, Guevara Noubir, and Rajmohan Rajaraman, (2004). Mobility models for ad hoc network simulation.

[14] Mirco Musolesi and Cecilia Mascolo. (2009). Mobility models for systems evaluation. *Middleware for Network Eccentric and Mobile Applications*, pages 1–28.

[15] C. Papageorgiou, K. Birkos, T. Dagiuklas, and S. Kotsopoulos. (sept. 2009). An obstacle-aware human mobility model for ad hoc networks. In *Modeling, Analysis Simulation of Computer and Telecommunication Systems, 2009. MASCOTS '09. IEEE International Symposium on*, pages 1 –9.

[16] Charles E. Perkins and Elizabeth M. Royer. (1997). Ad-hoc on-demand distance vector routing. In *IN PROCEEDINGS OF THE 2ND IEEE WORKSHOP ON MOBILE COMPUTING SYSTEMS AND APPLICATIONS*, pages 90–100.

[17] The Scenario Generator. `http://http://ttp://isis.poly.edu/~qiming/scengen/index.html`.

[18] Razvan Stanica, Emmanuel Chaput, and André-Luc Beylot. (June 2011). Simulation of vehicular ad-hoc networks: Challenges, review of tools and recommendations. *Computer Networks*, 55(14):3179–3188.

[19] Chen Zhao and Mihail L. Sichitiu. (March 2011). Contact time in random walk and random waypoint: Dichotomy in tail distribution. *Ad Hoc Networks*, 9(2):152–163.

[20] Qunwei Zheng, Xiaoyan Hong, and Sibabrata Ray. (2004). Recent advances in mobility modeling for mobile ad hoc network research. In *Proceedings of the 42nd annual Southeast regional conference*, ACM-SE 42, pages 70–75, New York, NY, USA. ACM.