UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

POSGRADO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

DESARROLLO DE ALGORITMOS PARA CONOCER LA UBICACIÓN DE TERMINALES MÓVILES Y FIJAS PARA REDES INALÁMBRICAS.

TESIS
QUE PARA OPTAR POR EL GRADO DE:
DOCTOR EN CIENCIAS (COMPUTACIÓN)

PRESENTA:
Luis Francisco García Jiménez

TUTOR
Dr. Javier Gómez Castellanos
Facultad de Ingeniería, Departamento de Ingeniería en Telecomunicaciones

MÉXICO, D.F. MAYO 2015

Sinodales:

Dr. Javier Gómez Castellanos
Dr. Victor Rangel Licea
Dr. Miguel López Guerrero
Dr. Jorge Luis Ortega Arjona
Dr. Rolando Menchaca Méndez

Agradecimientos

Agradezco el apoyo brindado a cada una de las personas que laboran en el Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas de la UNAM. Especialmente el apoyo por parte del Dr. Javier Gómez Castellanos, ya que en todo momento me aconsejó para realizar este trabajo. Asimismo, las aportaciones y recomendaciones que surgieron durante la revisión de este trabajo por parte de: Dr. Víctor Rangel Licea, Dr. Miguel López Guerrero, Dr. Jorge L. Ortega Arjona, Dr. Rolando Menchaca Méndez. Por otro lado, agradezco el apoyo económico brindado por parte del CONACYT y el apoyo recibido dentro del proyecto DGAPA-PAPIIT IN 114813 y 114713.

Índice general

Índice de figuras

Índice de tablas

Índice de algoritmos

Resumen

En los últimos años se ha mostrado que es muy fácil localizar a un nodo móvil, sobre todo cuando éste envía su posición a un servidor remoto. Otros estudios muestran que la ubicación del nodo móvil puede ser estimada por un conjunto de nodos fijos aun cuando éste no transmita su posición. Una consecuencia de este escenario es que la privacidad del usuario móvil puede verse comprometida, entendiendo como privacidad: la garantía que tiene un usuario móvil de que su ubicación este resguardada y no sea usada sin su autorización.

En esta tesis proponemos una técnica para que el nodo móvil mida el riesgo de ser detectado por un conjunto de nodos fijos mediante un algoritmo que emula cómo la red estima la posición actual del nodo móvil, dicho algoritmo sólo se ejecuta en el nodo móvil sin intervención de la red. Este conocimiento, le permite al nodo móvil tener una poderosa herramienta para tomar contramedidas antes de transmitir, tales como: entrar en periodos de silencio, demorar su transmisión, cambiar su trayectoria con el fin de aumentar el error de ubicación visto por la red o viceversa dependiendo de la aplicación. En este trabajo llamamos a este escenario el problema de estimación de la auto-ubicación y hasta donde sabemos, proponemos el primer método llamado ESL (estimating self-location). En ESL, un nodo móvil debe crear un mapa local de la ubicación de los nodos de red, donde éstos se complementan con nodos *virtuales* con el fin de crear diferentes diagramas de Voronoi de la topología de red. Esta técnica le permite al nodo móvil usar un nuevo algoritmo de localización que emula cómo la red calcula la posición actual del nodo móvil. Posteriormente, ESL utiliza las estimaciones hechas por el algoritmo de localización y la posición actual del nodo móvil (obtenida mediante un receptor GPS) para calcular el error de localización incurrido por la red.

Por otro lado, para cuantificar si la estimación hecha por ESL es correcta, se necesita un algoritmo de localización ejecutado por los nodos de red que estimen la posición de un nodo móvil conforme éste atraviesa dicha red. Esta estimación se debe comparar con el valor estimado por ESL con el fin de cuantificar la exactitud de ESL. Para ello, en este trabajo de tesis proponemos un algoritmo de localización ejecutado por la red llamado Ghost. Ghost será emulado desde el nodo móvil por el algoritmo ESL.

En este trabajo de tesis se utiliza un simulador propio programado en el lenguaje de alto nivel Python con el fin de medir el desempeño de ambos algoritmos y computar la diferencia entre el error de ubicación estimado por Ghost y el error de ubicación estimado por ESL. Para ello, se desarrollan un conjunto de pruebas en diferentes condiciones que emulan las condiciones reales de propagación de las señales de radio entre nodos de red y el nodo móvil así como se presenta un experimento real. Tanto las simulaciones como el experimento real muestran que ESL es capaz de reproducir la posición estimada del nodo móvil por la red sin intercambiar información con ésta.

Escenario

Las aplicaciones diseñadas para los dispositivos inalámbricos se han hecho cada vez más comunes y fáciles de usar. Estos dispositivos pueden intercambiar mensajes para tener mayor cobertura y mejorar la movilidad en áreas más grandes. A pesar de estas ventajas, la naturaleza de difusión en las transmisiones inalámbricas facilita que un atacante pueda escuchar el tráfico a través del canal y de esta manera estimar la ubicación de alguna terminal móvil, lo que puede generar un problema de privacidad para el nodo móvil.

Los métodos de localización estiman la posición de una terminal móvil que emite una señal hacia la red. De hecho se suelen utilizar varios nodos fijos (atacantes) ubicados de manera estratégica para aumentar la precisión con que se ubica a la terminal móvil. Si bien existen varios métodos para localizar a una terminal móvil, la técnica más utilizada es conocida como trilateración. Esta técnica requiere de al menos tres nodo fijos no colineales que detecten y estimen la distancia a la terminal móvil.

En general existen tres escenarios donde un nodo fijo podría obtener información de la posición de una terminal móvil.

- El primero escenario se conoce como cobertura-uno (ver Figura 1(a)). En este escenario el nodo fijo (AP_1) sólo puede estimar la distancia al nodo móvil (r_1), pero no puede establecer una posición exacta del mismo, por lo que tiene un conjunto de n puntos posibles como solución geométrica sobre un círculo con radio r_1. R_1 representa la distancia máxima de cobertura del nodo fijo.

- El segundo escenario se presenta cuando dos atacantes detectan a la terminal móvil, es conocido como cobertura-dos (ver Figura 1(b)). En este caso la solución geométrica se convierte en dos posiciones posibles como se muestra en la Figura 1(b) (puntos negros). Uno de estos puntos es la ubicación actual de la terminal móvil, mientras que el otro es una posición falsa. Sin embargo; ambas soluciones son válidas para los atacantes ya que no pueden distinguir la ubicación real de la terminal móvil. r_1 y r_2 son las distancias calculadas por AP_1 y AP_2, respectivamente

- El último escenario, se conoce como cobertura-tres (ver Figura 1(c)). En este caso sólo hay un punto de intersección geométrica entre las circunferencias con radio r_1, r_2 y r_3 cuyos centros son los atacantes AP_1, AP_2 y AP_3, respectivamente. En esta situación, la exactitud de la posición de la terminal móvil es del $100\,\%$ como se muestra en la Figura 1(c).

En este trabajo de tesis se define una red de densidad variable como: un espacio Euclideano en $\mathbb{R}^2$ donde existen áreas con tres o más nodos fijos que detectan al

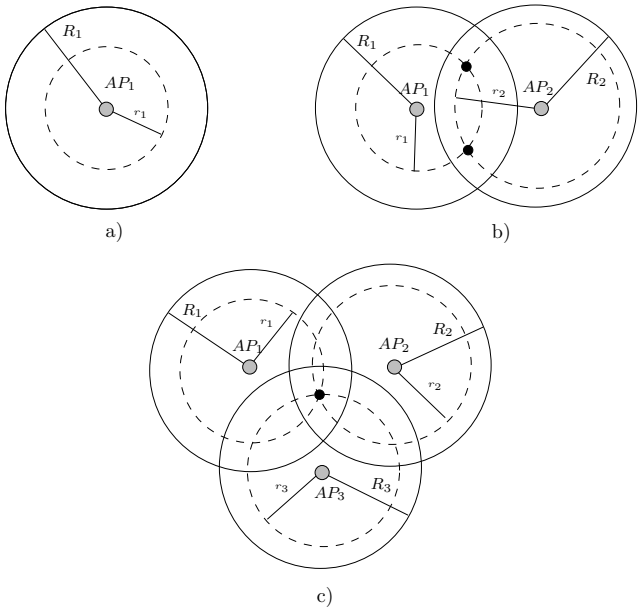


Figura 1: Cobertura-uno,dos y tres

nodo móvil así como existen áreas con menos de tres nodos fijos que detectan al nodo móvil.

Capítulo 1

Introducción

Con la proliferación y popularidad de los dispositivos de posicionamiento geográfico (GPS), la ubicación de un nodo móvil puede ser estimada de manera fácil y precisa. Esta disponibilidad facilita el desarrollo de muchas aplicaciones como son la localización y el seguimiento de terminales móviles [1]. También permite que los nodos móviles interactúen de mejor forma con su entorno. Por ejemplo, los servicios basados en localización (llamados LBS por sus siglas en inglés), requieren la ubicación geográfica del usuario (adquirida por GPS) con el fin de reenviarles información valiosa basada en su posición actual. Sin embargo, este escenario puede generar problemas de privacidad para los usuarios móviles ya que la ubicación de éstos es conocida por el proveedor del servicio (LBS). Para minimizar este riesgo, algunos estudios han centrado su esfuerzo en proteger la ubicación del usuario móvil mientras éste envía peticiones a un LBS [2].

Por otro lado, algunos estudios han mostrado que siempre que el nodo móvil transmita paquetes hacia la red, su ubicación actual puede ser estimada por un conjunto de nodos fijos (típicamente tres nodos no colineales) [3, 4, 1]. Incluso, algunos de estos métodos pueden ubicar al nodo móvil cuando sólo un nodo fijo detecta al nodo móvil [5, 6]. Es claro que este escenario también crea un problema de privacidad para el usuario móvil ya que la red podría almacenar su ubicación conforme pasa el tiempo y predecir sus posiciones futuras. Consideramos que esta vulnerabilidad es crítica para el nodo móvil, por lo que en este trabajo de tesis proponemos una forma de medir dichos riesgos. Dicho de otra manera, proponemos un algoritmo que sólo se ejecuta en el nodo móvil el cual es capaz de emular cómo la red estima su posición actual y de esta forma el nodo móvil puede conocer el error de ubicación que incurre la red. En otras palabras, el nodo móvil puede cuantificar con que precisión es ubicado por la red. A este escenario lo nombramos el problema de *estimación de la auto-ubicación*.

Afirmamos que si el nodo móvil estima su auto-ubicación antes de que éste transmita paquetes hacia la red, el nodo móvil tiene una poderosa herramienta para tomar decisiones y ejecutar alguna contramedida antes de transmitir algún paquete a la red. Por ejemplo, en una situación de emergencia, un usuario móvil que sea capaz de medir su auto-ubicación puede determinar hacia donde debe moverse con el objetivo de obtener la mejor cobertura y estar disponible el mayor tiempo posible. De manera opuesta, un usuario móvil que conozca su auto-ubicación puede variar su potencia de transmisión en su tarjeta de red para reducir la probabilidad de ser detectado o posponer su transmisión para incrementar el error de ubicación visto por la red e inclusive cambiar de trayectoria.

1.1. Definición del problema

El presente trabajo de tesis pretende responder y aportar un conjunto de algoritmos en relación a la siguiente pregunta: ¿Es posible que un nodo móvil reproduzca la posición estimada por un conjunto de nodos fijos que detectaron al nodo móvil? La solución de esta pregunta relaciona dos perspectivas opuestas. Por un lado, la red inalámbrica desea estimar la posición de un nodo móvil mientras éste transmite paquetes. Por otro lado, el nodo móvil desea cuantificar con que precisión es ubicado por la red. Para relacionar ambas perspectivas se debe crear un algoritmo que se ejecute en el nodo móvil y que sea capaz de emular cómo la red estima su posición actual. Si esto último es posible, entonces la posición estimada por el nodo móvil se debe comparar con la posición estimada por la red. Si ambas estimaciones son similares, entonces se puede afirmar que el nodo móvil es capaz de emular a la red.

Desde la perspectiva de la red, en este trabajo de tesis se propone un algoritmo llamado Ghost (ver Capítulo 2) el cual estima la posición de un nodo móvil en una red de densidad variable. Se hace la suposición que cada nodo de red conoce su coordenada geométrica y que es capaz de detectar y estimar la distancia entre su coordenada geométrica y la posición actual del nodo móvil.

Desde la perspectiva del nodo móvil, en este trabajo de tesis se propone un algoritmo llamado ESL; estimating self-location, por sus siglas en inglés, (ver Capítulo 3) el cual emula cómo la red estima la posición actual del nodo móvil en una red de densidad variable. Se hacen las siguientes suposiciones: el nodo móvil tiene un dispositivo GPS que le permite conocer su posición actual en cualquier tiempo dado, el rango de cobertura máximo tanto para los nodos de red como para el nodo móvil es del mismo tamaño. El nodo móvil debe ser capaz de estimar la distancia entre su posición actual y la posición de un nodo de red.

Una vez implementados ambos algoritmos se debe calcular la diferencia entre el error de ubicación estimado por Ghost y el el error de ubicación estimado por ESL. Entre más cercana a cero sea esta diferencia quiere decir que ESL es capaz de emular con precisión cómo la red estima su posición actual. Este escenario lo nombramos estimación de la auto-ubicación.

1.2. Hipótesis

Hipótesis: ”Dado un nodo móvil equipado con un receptor GPS y una red de densidad variable que es capaz de estimar la posición actual del nodo móvil, es posible que el nodo móvil estime su auto-ubicación antes de transmitir a la red.”Suponemos que tanto los nodos de red como el nodo móvil son capaces de medir la potencia de una señal recibida y a su vez estimar la distancia entre su posición geográfica y dicha fuente. Además suponemos que el radio de cobertura máximo tanto del nodo móvil como de los nodos fijos es de la misma longitud.

1.3. Aproximación al problema

Desde la perspectiva del nodo móvil proponemos un algoritmo para estimar la auto-ubicación en redes inalámbricas llamado ESL. Para lograr esta tarea, ESL debe escuchar la transmisión de datos enviados desde los nodos de red, después debe calcular

la distancia entre su posición actual (determinada a través de un receptor GPS) y cada nodo de red al menos en dos posiciones diferentes (por ejemplo, midiendo la potencia de la señal recibida). Esto permite que el nodo móvil construya un mapa local de los nodos de red. Mediante la adición de un conjunto de nodos *virtuales* colocados alrededor de cada nodo de red, ESL crea diferentes diagramas de Voronoi del mapa descubierto con el fin de utilizar un algoritmo de localización que emula cómo la red estima la posición actual del nodo móvil. Después ESL utiliza las posiciones estimadas del nodo móvil y su posición real (obtenida mediante un receptor GPS) con el fin de calcular el error de ubicación visto por la red o dicho en otras palabras, la exactitud con la que la red estima la posición del nodo móvil.

Una suposición común en la mayoría de los algoritmos de localización es que la topología de red es uniforme (es decir, se suponen k nodos de red que cubren cada punto, donde k típicamente es mayor o igual a tres [7]). Sin embargo, esta suposición rara vez es cierta en una topología real. Incluso en las redes inalámbricas densamente pobladas, siempre hay regiones con un menor número de nodos de red que en otras. Además, factores como son: obstáculos, zonas irregulares, cortes de energía, degradaciones de la señal, interferencia entre canales y ruido de otros dispositivos pueden causar pérdida de conectividad en algunas regiones. Debido a todos estos factores, consideramos que es importante estudiar cómo la auto-ubicación se ve afectada conforme el número de nodos de red que detectan al nodo móvil varía con el tiempo. Especialmente, en este trabajo de tesis estudiamos la estimación de la auto-ubicación cuando menos de tres nodos de red detectan al nodo móvil.

Desde la perspectiva de la red se sabe que estimar la posición de un nodo móvil no es un trabajo trivial, ya que existen varios factores que pueden cambian la topología de red como se mencionó anteriormente. Incluso, la mayoría de las redes no se diseñan para garantizar una cobertura de al menos tres nodos fijos que observen cada punto de la red. Por ejemplo, los *hotspots*, las redes de oficinas, casas y universidades, contemplan un sólo punto de acceso (AP) que cubre la mayor área posible. Por estas razones consideramos que estudiar el escenario donde un nodo móvil es detectado por menos de tres nodos fijos es un caso frecuente y, por lo mismo, importante. Para ello en este trabajo se propone un algoritmo llamado Ghost el cual debe ser capaz de estimar la posición de un nodo móvil en un escenario de densidad variable. Para lograr esto se utiliza un conjunto de nodos virtuales colocados alrededor de los nodos fijos con el propósito de crear diferentes diagramas de Voronoi de la topología de red. La combinación de estos diagramas en el tiempo permite estimar la posición actual del nodo móvil para casos cuando un nodo móvil es detectado por más de tres nodos fijos, así como cuando el nodo móvil es detectado por menos de tres nodos fijos.

Finalmente se debe calcular la diferencia entre el error de ubicación estimado por Ghost y el error de ubicación estimado por ESL. Si esta diferencia tiende a cero se puede decir que ESL es capaz de emular con precisión cómo la red estima su posición actual.

En este trabajo de tesis utilizamos un simulador programado en el lenguaje de alto nivel Python con el fin de medir el desempeño de ambos algoritmos y computar la diferencia entre el error de ubicación estimado por Ghost y el error de ubicación estimado por ESL. Para ello, se debe desarrollar un conjunto de pruebas en diferentes condiciones que emulan las condiciones reales de propagación de las señales de radio entre nodos de red y el nodo móvil.

1.4. Objetivo

Diseñar un algoritmo capaz de estimar la auto-ubicación de un nodo móvil que atraviesa una red inalámbrica con el fin de que la terminal móvil conozca la posición estimada por la red y tome una contramedida antes de transmitir.

1.4.1. Objetivos particulares

- Calcular el error de ubicación estimado por Ghost
- Calcular el error de ubicación estimado por ESL
- Calcular la diferencia entre el error de ubicación estimado por Ghost y el error de ubicación estimado por ESL

1.5. Contribución

- En este trabajo introducimos el concepto de estimación de la auto-ubicación al área de redes inalámbricas. Consideramos que si el nodo móvil conoce su auto-ubicación tiene una poderosa herramienta para cuantificar su vulnerabilidad con respecto a los nodos de red que detectan su posición y trayectoria. Esto se debe a que el nodo móvil puede conocer el error de ubicación en el que incurre la red.
- introducimos una técnica para dividir el espacio Euclideano con el uso de nodos virtuales que permiten crear diagramas distintos de Voronoi para diferentes posiciones del nodo móvil bajo la misma topología de red. A su vez esta técnica encierra al nodo móvil en polígonos más pequeños disminuyendo el error de ubicación estimado por la red.
- Proponemos un algoritmo capaz de ubicar a un nodo móvil aun cuando éste es detectado por menos de tres nodos de red.

1.6. Descripción del contenido

Para alcanzar los objetivos planteados anteriormente, es necesario usar conceptos de geometría computacional para dividir el espacio Euclideano y ubicar al nodo móvil en algún polígono convexo así como algunos conceptos estadísticos para modelar los errores de propagación en las señales inalámbricas. Además, implementamos ambos algoritmos en el lenguaje de alto nivel Python ya que es un lenguaje de fácil uso para la creación de algoritmos y especialmente de interfaces gráficas que nos permitan observar el comportamiento y desempeño de los algoritmos ya mencionados. Este trabajo de tesis se presenta de la siguiente manera:

- En el capítulo 2 proponemos un algoritmo llamado Ghost, el cual es capaz de seguir a un nodo móvil aun cuando éste sea detectado por menos de tres nodos de red. En este capítulo introducimos la técnica de nodos virtuales. Estos permiten crear diferentes diagramas de Voronoi de la topología de red con el fin de

obtener el conjunto de polígonos más probables donde el nodo móvil fue detectado, asimismo se presentan las bases geométricas de Ghost. Después definimos el concepto de circulo de máxima velocidad (MSC, maximum speed circle, por sus siglas en inglés), el cual provee información adicional para los casos donde el nodo móvil es detectado por menos de tres nodos de red. También analizamos distintos escenarios variando tanto el número de nodos de red como el número de nodos virtuales con el fin de minimizar el error de ubicación del nodo móvil estimado por la red. Evaluamos el algoritmo Ghost cuantitativamente y comparamos este trabajo con los existentes en la literatura en términos de error de ubicación. Por último presentamos conclusiones y observaciones del algoritmo propuesto.

- En el capitulo 3 proponemos un algoritmo capaz de estimar la auto-ubicación de un nodo móvil llamado ESL. Se explican en detalle los tres escenarios principales que componen a ESL. También presentamos un conjunto de simulaciones bajo diversas condiciones de red, así como un experimento real que muestra que ESL es capaz de reproducir el error de posición del nodo móvil estimado por la red en condiciones reales de tráfico. En este capítulo se utiliza el algoritmo Ghost para comparar el error de ubicación estimado por ESL y cuantificar la exactitud de ESL. Finalmente, damos algunas observaciones finales de ESL.
- En el capitulo 4 presentamos las conclusiones generales y trabajo futuro.

Capítulo 2

Ghost

2.1. Resumen

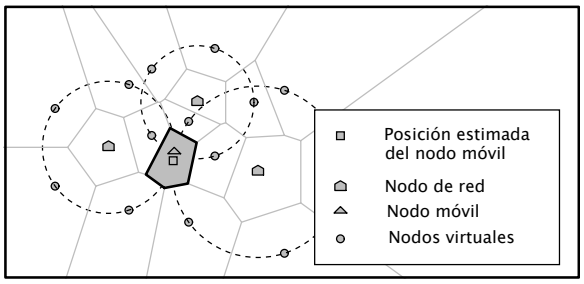
Las técnicas convencionales de localización usan al menos tres nodos de red no colineales para ubicar a un nodo móvil. Sin embargo; no siempre es posible garantizar que existan tres o más nodos de red que detecten al nodo móvil. En este capítulo presentamos un algoritmo llamado Ghost, el cual es un método basado en diagramas de Voronoi, capaz de seguir a un nodo móvil aun cuando éste es detectado por menos de tres nodos de red. Para Ghost, diferentes posiciones del nodo móvil crean diferentes diagramas de Voronoi de la topología de red gracias al uso de nodos *virtuales* colocados alrededor de los nodos de red. Estos diagramas se utilizan para estimar la ubicación actual del nodo móvil a través de intersecar el último diagrama de Voronoi con el penúltimo diagrama. De esta manera la ruta del nodo móvil es construida mediante la búsqueda sistemática de las posiciones más probables a través del tiempo. Las simulaciones realizadas en este capítulo validan que esta metodología garantiza una mayor precisión en términos de la posición estimada del nodo móvil comparado con las propuestas existentes en la literatura. Además, este enfoque no está vinculado a ninguna tecnología específica, por lo que puede ser usada en diferentes plataformas (por ejemplo, WLAN y WSN).

2.2. Introducción

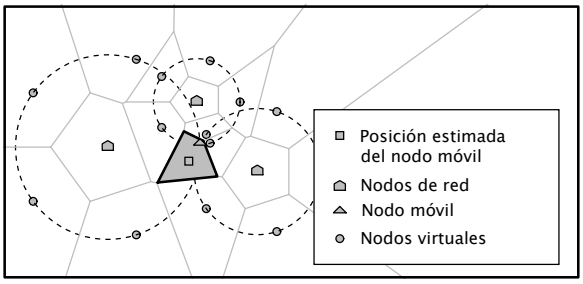
Ghost se compone de tres elementos principales:

- Nodo móvil. Es un nodo cuya tarjeta inalámbrica transmite paquetes hacia la red.
- Nodo de red. Es un nodo fijo dentro de una área que es capaz de detectar al nodo móvil y estimar la distancia hacia éste.
- Nodo *sink*. Es el centro de procesamiento que recibe la información de varios nodos de red y es capaz de calcular la ruta del nodo móvil mediante el uso del algoritmo Ghost.

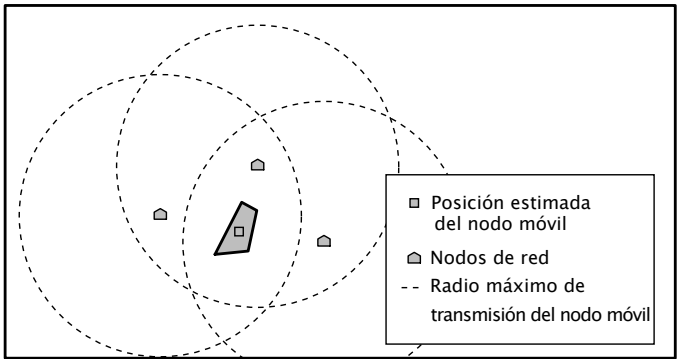
Se requieren tres facetas para que Ghost ubique a un nodo móvil. En primer lugar, los nodos de red que detectan al nodo móvil estiman la distancia Euclideana



(a) Diagrama de Voronoi en el tiempo $t - \Delta t$



(b) Diagrama de Voronoi en el tiempo t



(c) Intersección entre los polígonos seleccionados en el tiempo t y $t - \Delta t$

Figura 2.1: Operación de Ghost: (a) Ghost selecciona el polígono más probable donde el nodo móvil está ubicado para el tiempo $t - \Delta t$ (ver polígono en color gris). (b) Cuando se obtiene una nueva detección del nodo móvil en el tiempo t, Ghost nuevamente selecciona el polígono más probable donde se encuentra el nodo móvil. (c) Ghost interseca el polígono seleccionado en el tiempo t con el polígono seleccionado en el tiempo $t - \Delta t$ con el fin de crear un polígono más pequeño que encierra la ubicación del nodo móvil

al nodo móvil y envían dicha información al nodo *sink*. En segundo lugar, el nodo sink divide la topología de red en regiones convexas mediante el uso del algoritmo de Voronoi [8] el cual requiere dos conjuntos de puntos como entrada. El primer grupo está compuesto por los nodos de red que componen el espacio Euclideano dado por sus coordenadas cartesianas, ver Figuras 2.1(a) y 2.1(b). El segundo grupo está compuesto por puntos equidistantes (nombrados nodos *virtuales*) colocados a lo largo de un círculo con centro en las coordenadas cartesianas de cada nodo de red y cuyo radio es igual a: (i) la distancia estimada entre el i-ésimo nodo de red y el nodo móvil para aquellos nodos de red que detectan al nodo móvil (ver Figuras 2.1(a) y 2.1(b)), (ii) el máximo radio de transmisión para aquellos nodos de red que no detectan al nodo móvil. En tercer y último lugar, Ghost utiliza el diagrama de Voronoi actual para seleccionar el (los) polígono(s) más probable(s) donde el nodo móvil se localiza mediante el uso del algoritmo *point location* [8, 9]. Es importante mencionar que gracias al uso de los nodos virtuales, Ghost puede crear diferentes diagramas de Voronoi para cada posición distinta del nodo móvil.

Como se mencionó anteriormente, debido a que los diagramas de Voronoi de la topología de red cambian con cada posición distinta del nodo móvil (ver Figuras 2.1(a) y 2.1(b)), Ghost puede calcular la ruta del nodo móvil mediante la intersección del polígono elegido en el tiempo t con respecto al polígono seleccionado en el tiempo $t - \Delta t$, donde Δt representa el tiempo transcurrido entre la penúltima detección del nodo móvil y la última detección. El resultado de esta intersección (si existe) es un polígono más pequeño que encierra la ubicación más probable del nodo móvil, ver la Figura 2.1(c). Estos polígonos se utilizan para construir un grafo G donde el centro de cada polígono es un vértice de la ruta seguida por el nodo móvil. Por otra parte, al aumentar el número de nodos virtuales, Ghost puede reducir el área donde se ubica el nodo móvil en una región de Voronoi más pequeña (es decir, un polígono convexo), debido al hecho de que el número de regiones de Voronoi aumenta linealmente con el número de nodos virtuales [8].

2.3. Trabajo relacionado

Los sistemas de localización se pueden dividir en dos grandes categorías:

- *range-based*: basan su medición en alguna propiedad de la señal trasmitida o recibida. Las técnicas mas usadas en estos sistemas son: Time of Arrival (ToA), Time Difference of Arrival (TDoA), Angle of Arrival (AoA) y Received Signal Strength (RSS). Cada una de estas técnicas estima la distancia entre dos nodos al medir: el tiempo que tarda en llegar la señal, la diferencia entre el tiempo de envío y recepción, el ángulo entre la señal enviada y la recibida o la disminución de la potencia de la señal, respectivamente. Sin embargo; la mayoría de los sistemas de localización usan técnicas de trilateración para disminuir el error de ubicación estimado por la red. Por ejemplo, en [10] los autores proponen una técnica de trilateración para disminuir los errores de localización mediante la aplicación de filtros Kalman conforme disminuye la potencia de la señal (RSS) en una red de infraestructura WLAN. En [11], los autores proponen una técnica de trilateración para localizar puntos de acceso (AP) alrededor de un conjunto habitacional mediante el montaje de antenas direccionales en vehículos que circulan en las calles.

- *range-free*: son esquemas de proximidad y conectividad. Por ejemplo, en [12] la ubicación de un conjunto de sensores se estima mediante la medición de la distancia entre algunos sensores seleccionados en intervalos distintos de tiempo con el fin de reconstruir la topología de red. El algoritmo APIT [13], estima la posición de un conjunto de sensores al dividir el espacio Euclidiano en regiones triangulares. Dependiendo si el nodo se encuentra dentro o fuera de un triángulo seleccionado, este algoritmo puede reducir el área donde se localiza un sensor al mover un vértice del triángulo. Para más información de algoritmos de localización, refiérase a [1].

Con la posición del nodo móvil, el siguiente paso es estimar la trayectoria más probable del nodo móvil. La mayoría de los enfoques de este tipo de algoritmos basan su funcionamiento en esquemas de probabilidad utilizando filtros de Kalman [14] [15] [16]. Por ejemplo, en [16] los autores utilizaron filtros de Kalman sobre un conjunto de mediciones ToA con el fin de minimizar el error de ubicación del nodo móvil. Por otro lado, la geometría computacional es una área de las ciencias de la computación que puede ser utilizada para mejorar y simplificar algunas técnicas de localización.

- En [17], los autores proponen un sistema de ubicación humana para entornos en interiores mediante la colocación de sensores infrarrojos y de ultrasonido. Este algoritmo es capaz de distinguir la identidad y ubicación de las personas mediante el uso de técnicas de aprendizaje en el movimiento humano basado en diagramas de Voronoi.

- En [18], los autores proponen el primer algoritmo de localización que divide el espacio Euclideano en regiones poligonales basado en algoritmos de gráficas planas. La idea principal de este algoritmo es seleccionar las aristas más probables por donde el nodo móvil cruza el polígono.

En resumen, ubicar a un nodo móvil no es una tarea fácil, incluso en escenarios simples se complica llevarla a cabo. Además, ninguna de las propuestas mencionadas contemplan el caso cuando menos de tres nodos de red detectan la presencia del nodo móvil. El único trabajo de nuestro conocimiento que da respuesta a este escenario es el propuesto en [5], donde los autores proponen dos algoritmos basados en filtros de Kalman. El primer algoritmo propuesto en [5] se llama ubicación por seguimiento predictivo (PLT, por sus siglas en inglés), éste utiliza la información obtenida de un filtro de Kalman para emular las señales de las estaciones base (BS) faltantes con el fin de utilizar técnicas de trilateración. El segundo algoritmo llamado geométrico asistido PLT (GPLT) ajusta la posición de las estaciones base faltantes mediante una técnica geométrica (GDOP) que garantiza una mayor exactitud en la posición de las estaciones base faltantes. Debido a que ambos métodos mostrados en [5] representan el escenario más cercano a nuestro trabajo, se compara su desempeño con nuestro algoritmo en la Sección 2.6.

2.4. Componentes de Ghost

En esta sección revisamos las condiciones y suposiciones para que opere el algoritmo Ghost, después se discuten los componentes que utiliza para lograr sus objetivos.

Consideramos un conjunto de N nodos de red colocados al azar en un espacio de dos dimensiones ($\mathbb{R}^2$) con un identificador único. Para cada nodo de red situado en

el punto p (es decir, (x, y)), representamos el rango de transmisión máximo como un círculo unitario (UDG) C_u (p, R_c) con centro en las coordenadas del punto p cuyo radio R_c es igual a uno. Además, suponemos que el rango de transmisión es uniforme. Un nodo de red puede detectar a un nodo móvil si y sólo si el nodo móvil está dentro de la cobertura del nodo de red. Modelamos la ruta estimada del nodo móvil como un grafo dirigido G(V, E), donde el centro de un polígono seleccionado por nuestro algoritmo representa un vértice $v \in V$. Dos vértices diferentes comparten una arista en común e $(u, v) \in E$ si y sólo si el vértice u es elegido por Ghost en el tiempo $t - \Delta t$ y v es elegido en el tiempo t. Suponemos que el nodo *sink* conoce la coordenada geográfica de cada nodo de red (por ejemplo, por medio de GPS [19, 20] u otro método de localización indirecto [21, 13, 1]).

2.4.1. División del espacio Euclideano

Cada vez que el nodo móvil es detectado por al menos un nodo de red, Ghost coloca un círculo con centro en las coordenadas de cada nodo de red cuyo radio es igual a: i) la distancia Euclidiana entre el $i-$ésimo nodo de red y el nodo móvil para aquellos nodos de red que detectaron al nodo móvil. ii) el rango de transmisión máximo (R_c) para aquellos nodos de red que no detectaron al nodo móvil. Estos círculos se dividen en m arcos iguales, donde dos arcos comparten un punto llamado nodo virtual como se muestra en las Figuras 2.1(a) y 2.1(b). Tanto los nodos virtuales como los nodos de red se utilizan como entrada al algoritmo de Voronoi con el fin de dividir el espacio euclidiano en regiones convexas. Dicho algoritmo crea $m \times N + N$ polígonos convexos, donde N es el número de nodos de red (es decir, los nodos reales en el espacio Euclideano) y m es el número de nodos virtuales por nodo de red. En otras palabras, cada nodo de red tiene m nodos virtuales asociados. El tiempo de procesamiento del algoritmo de Voronoi es $O(n\ log\ n)$ [8], donde n es igual a $m \times N + N$. Es importante mencionar que el número de nodos virtuales utilizados en cada nodo de red debe elegirse tomando en cuenta que entre mayor sea la precisión en la estimación de la posición del nodo móvil, mayor será el tiempo de procesamiento del algoritmo, esto será discutido en la Sección 2.6.

Localizar a un nodo móvil en Ghost implica tres escenarios posibles: cobertura-{tres, dos, uno}, que se describen en los siguientes apartados. Primero presentamos estos tres escenarios sin tomar en cuenta los errores de estimación en la distancia y suponemos que el espacio euclidiano es dividido por el nodo *sink* utilizando el algoritmo de Voronoi [8, 17, 22]. Después aplicamos un modelo de error en la estimación de la distancia en la Sección 2.6.

2.5. Ghost

En esta sección explicamos el funcionamiento del algoritmo Ghost utilizando los componentes descritos en el apartado anterior. Como se mencionó, localizar a un nodo móvil para Ghost implica tres escenarios posibles dependiendo del número de nodos de red que detectan al nodo móvil. Estos escenarios los llamamos cobertura-tres, cobertura-dos y cobertura-uno.

2.5.1. Cobertura-tres

Ghost utiliza una técnica de trilateración [10] para los casos donde el nodo móvil es detectado por tres o más nodos de red. Esta técnica devuelve la ubicación del nodo móvil como un punto p. Este punto se encuentra estrictamente dentro de un polígono $\mathcal{P}$ dentro del espacio Euclideano dividido por el diagrama de Voronoi. Ghost utiliza el algoritmo point location [8] [9] para encontrar el polígono $\mathcal{P}$. Esta técnicas funciona de la siguiente manera: la subdivisión poligonal del espacio Euclideano (dado por el diagrama de Voronoi en el tiempo presente t) se divide con un trazado de líneas verticales a través de cada vértice del diagrama de Voronoi. Dos líneas verticales consecutivas forman una región en la que la coordenada x de cada vértice se ordena en un arreglo. Este algoritmo utiliza la búsqueda binaria en $O(log\ n)$ para encontrar la coordenada x del punto p. Un proceso similar se utiliza para la coordenada y. Posteriormente, las coordenadas x y y de p se vinculan a un único polígono $\mathcal{P}$ dentro de la subdivisión poligonal. En un escenario con ruido, la ubicación estimada del nodo móvil puede ser diferente a su ubicación real debido a los errores de estimación de distancia. Para ello, Ghost utiliza el centro del polígono elegido $\mathcal{P}$ como la ubicación más probable del nodo móvil.

Para los casos en que el nodo móvil es detectado por tres nodos de red de manera consecutiva, la ruta del nodo móvil se construye de la siguiente manera: sean $\mathcal{P}$ y $\mathcal{Q}$ dos polígonos seleccionados por Ghost en el tiempo t y en el tiempo $t - \Delta t$, respectivamente. Sea $\mathcal{Z}$ el polígono convexo resultado de la intersección entre los polígonos $\mathcal{P}$ y $\mathcal{Q}$. Ghost construye la ruta del nodo móvil al unir el centro del polígono $\mathcal{Q}$ con el centro del polígono $\mathcal{Z}$, y el centro del polígono $\mathcal{Z}$ con el centro del polígono $\mathcal{P}$ (es decir, Ghost une $v_{\mathcal{Q}} \rightarrow v_{\mathcal{Z}} \rightarrow v_{\mathcal{P}}$). En caso que el polígono $\mathcal{Z}$ sea $\emptyset$ (es decir, no existe intersección entre los polígonos $\mathcal{P}$ y $\mathcal{Q}$), Ghost une el centro del polígono $\mathcal{Q}$ con el centro del polígono $\mathcal{P}$ (es decir, $v_{\mathcal{Q}} \rightarrow v_{\mathcal{P}}$). Ghost calcula la intersección entre polígonos en $O(m+n)$ [23], donde m y n son los vértices de los polígonos $\mathcal{Q}$ y $\mathcal{P}$, respectivamente.

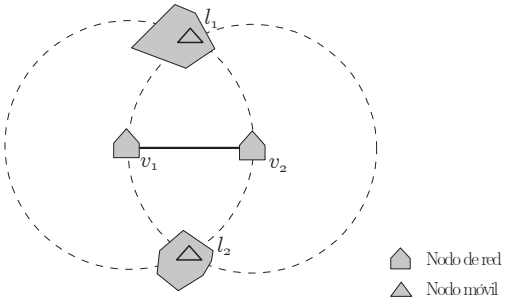
Todas las posiciones del nodo móvil obtenidas en un escenario de cobertura-tres las denominamos puntos *anclas*. Estos puntos pueden ser usados como puntos de anclaje con el fin de reducir la incertidumbre en la posición del nodo móvil cuando menos de tres nodos de red lo detectan.

La operación de Ghost en pseudo código para cobertura-tres se muestra en el Algoritmo 1.

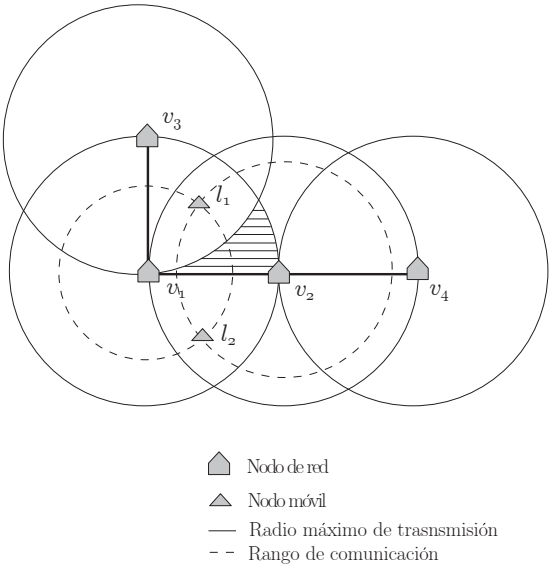
2.5.2. Cobertura-dos

La Figura 2.2(a) muestra el escenario cuando dos nodos de red (v_1, v_2) detectan al nodo móvil. Es fácil ver que se forma una área común al intersecar la cobertura de cada nodo de red. Esta intersección genera dos posiciones posibles del nodo móvil (es decir, l_1 y l_2). Una de estas posiciones representa la posición real del nodo móvil mientras la otra es llamada posición espejo. Note que en este momento no es posible saber cuál corresponde a la posición real y cuál a la posición espejo. A continuación se muestran algunas técnicas para descartar la posición espejo.

La posición espejo del nodo móvil puede ser descartada en algunas situaciones especiales, ver el caso ilustrado en la Figura 2.2(b), donde los nodos de red v_1 y v_2 detectan la presencia del nodo móvil y envían las distancias estimadas al nodo *sink*. El nodo *sink* puede eliminar la ubicación espejo del nodo móvil si y sólo si una de las dos posiciones posibles del nodo móvil se encuentra dentro del área de cobertura



(a)



(b)

Figura 2.2: Dos posiciones posibles del nodo móvil.

Algoritmo 1 Cobertura-tres.

Require: Definir m (número de nodos *virtuales* por nodo de red).

if un nodo móvil es detectado por al menos tres nodos de red: **then**
  Envíar las distancias estimadas al nodo *sink*.
  Generar el diagrama de Voronoi.
  Seleccionar el polígono $\mathcal{P}$ donde se localiza el nodo móvil.
  Conectar el centro del polígono $\mathcal{P}$ (i.e., $v_{\mathcal{P}}$) en el grafo G.
  Concatenar $v_{\mathcal{P}}$ como punto ancla.
end if
if existe intersección entre los polígonos $\mathcal{P}$ y $\mathcal{Q}$ **then**
  Calcular el polígono $\mathcal{Z}$.
  Conectar $v_{\mathcal{Q}} \rightarrow v_{\mathcal{Z}} \rightarrow v_{\mathcal{P}}$.
else
  Conectar $v_{\mathcal{Q}} \rightarrow v_{\mathcal{P}}$.
end if

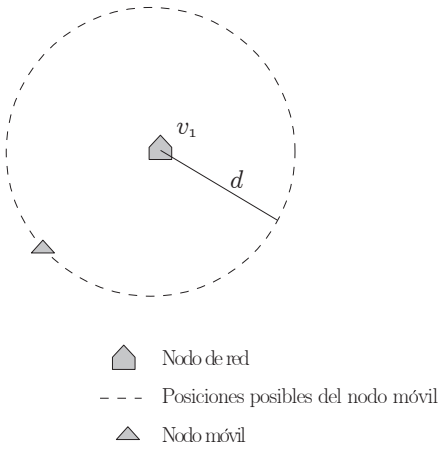
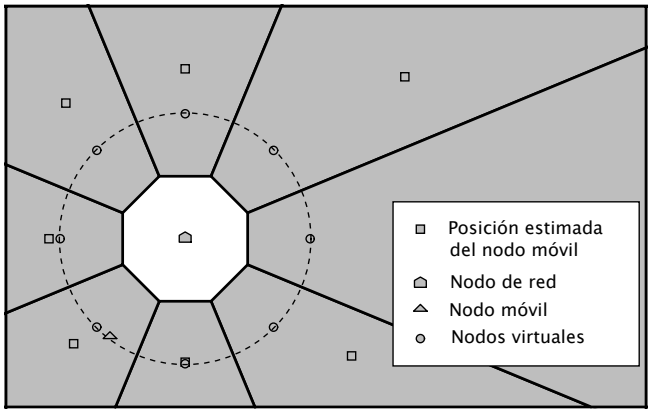
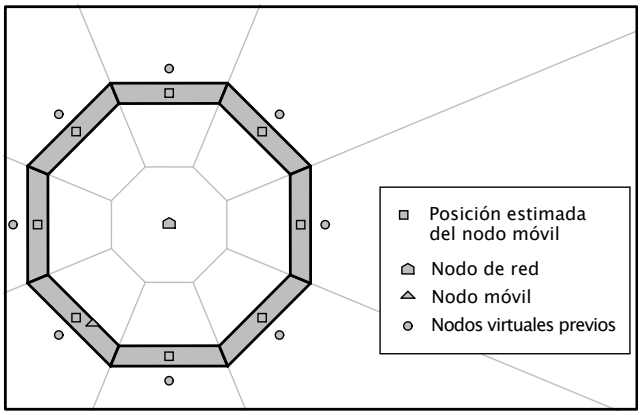


Figura 2.3: Cobertura-uno

de los vecinos más cercanos. Por ejemplo, en la Figura 2.2(b) los nodos de red v_3 y v_4 son vecinos cercanos de v_1 y v_2; dado que los nodos v_3 y v_4 no detectaron la presencia del nodo móvil, implica que la posición l_1 es la posición espejo, por lo que puede ser descartada por el nodo *sink*, mientras la posición l_2 es la posición real del nodo móvil. Sin embargo, supongamos que la posición espejo estuviera en la región sombreada (ver Figura 2.2(b)), en este caso no se podría descartar la posición espejo, ya que ésta se encontraría fuera del área de cobertura de los nodos de red v_3 o v_4. En la Sección 2.5.4 vamos a utilizar el concepto de círculo de máxima velocidad (MSC) para descartar la ubicación espejo en los casos en que el método anterior no pueda ser utilizado.



(a) Diagrama de Voronoi construido por ocho nodos virtuales y un nodo de red



(b) Dona construida al intersecar los polígonos en tiempo $t - \Delta t$ con los polígonos en el tiempo t

Figura 2.4: Cobertura-uno con el uso de nodos virtuales

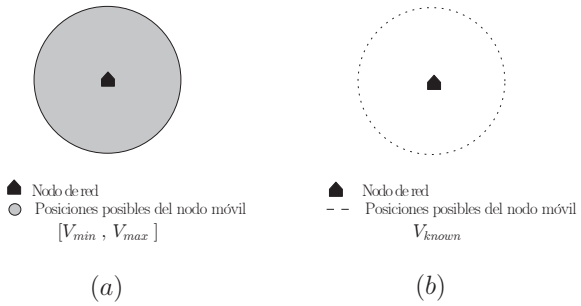


Figura 2.5: Cobertura del MSC

2.5.3. Cobertura-uno

En las redes con pocos nodos (baja densidad), el escenario de cobertura-uno es una situación común en la cual el nodo de red sólo puede estimar la distancia al nodo móvil, pero no puede establecer una posición exacta del mismo, ver la Figura 2.3 donde un nodo móvil es detectado por un sólo nodo de red a una distancia d. Ghost resuelve este problema mediante la colocación de nodos virtuales alrededor del nodo de red. En la Figura 2.4(a) se muestra un diagrama de Voronoi formado por ocho nodos virtuales y un nodo de red. Como podemos ver en esta figura, utilizar el centro de los polígonos seleccionados (dibujados como un □) como la posición probable del nodo móvil puede dar lugar a un error de ubicación de tamaño considerable. Esto ocurre porque algunos de estos polígonos pueden no tener bordes. Para resolver este problema, Ghost limita el tamaño de cada polígono en el escenario de cobertura-uno mediante la intersección de los polígonos calculados en el tiempo t con los polígonos calculados en el tiempo $t - \Delta t$. La Figura 2.4(b) ilustra el resultado de este proceso. Observe que utilizamos los nodos virtuales calculados en el tiempo $t-\Delta t$ para calcular el diagrama de Voronoi en el tiempo t. Este proceso construye un conjunto de polígonos en el que cada uno de ellos está delimitado por cuatro aristas como se muestra en la Figura 2.4(b). Llamamos a este conjunto de polígonos *dona*.

2.5.4. Círculo de máxima velocidad

En esta sección introducimos el concepto de círculo de máxima velocidad (MSC por sus siglas en inglés), el cual disminuye el error de ubicación estimada del nodo móvil cuando éste se encuentra dentro del escenario de cobertura-{uno, dos}. El MSC puede ser visto como un círculo de radio cero situado en el centro del último polígono seleccionado, el cual aumenta su radio a la velocidad máxima (V_{max}) del nodo móvil hasta que la red detecta nuevamente al nodo móvil en el tiempo presente t. Debido a que el nodo móvil es libre de moverse en cualquier dirección a una velocidad en el intervalo [0, V_{max}], el MSC contiene todas las posiciones posibles del nodo móvil entre dos detecciones consecutivas (ver la zona gris de la Figura 2.5(a)). En Ghost, también consideramos el caso cuando se conoce la velocidad del nodo móvil o se puede estimar (V_{known}). En este caso, sólo los puntos del perímetro del MSC son posiciones válidas del nodo móvil como se muestra en la Figura 2.5(b).

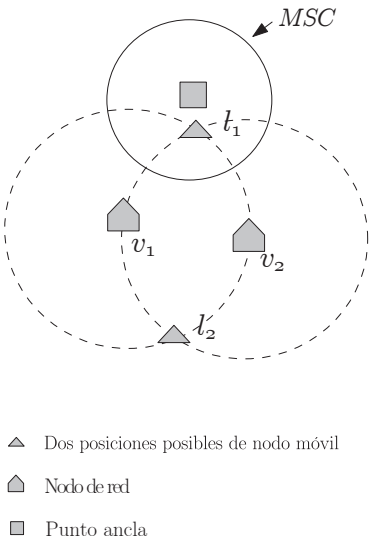


Figura 2.6: Dos nodos de red usando MSC

2.5.5. Cobertura-dos usando MSC

MSC puede ser utilizado para descartar la ubicación espejo del nodo móvil en cobertura-dos de la siguiente manera: el MSC se encuentra inicialmente en la última posición conocida del nodo móvil (por ejemplo, un punto ancla antes de entrar al área de cobertura-dos) con un radio igual a cero. A continuación, el MSC aumenta su radio a una velocidad dada (por ejemplo, V_{max} o V_{known}) hasta que se obtiene una nueva detección. Ghost puede descartar la posición espejo, si y sólo si la posición espejo se encuentra fuera del MSC. Para ilustrar esto, la Figura 2.6 muestra el último punto conocido del nodo móvil (se muestra como □) y la región traslapada entre los dos nodos de red (v_1 y v_2). En este ejemplo, la posición l_2 se encuentra fuera del MSC, por lo que puede ser descartada. Tenga en cuenta que el último punto ancla necesariamente se calculó en el tiempo $t - \Delta t$. En caso de no existir un punto ancla o posiciones conocidas anteriores, Ghost no puede descartar la posición espejo del nodo móvil, ya que ningún MSC puede ser colocado en dicha posición.

La ruta del nodo móvil para cobertura-dos se construye conectando el centro del polígono donde se encuentra el último MSC con el centro del (los) polígono(s) que encierra la posición del nodo móvil (ver la Figura 2.7(a)). En caso de que la posición espejo no pueda ser descartada (es decir, tanto la ubicación real y espejo se encuentran dentro del MSC actual), Ghost une el centro del MSC actual con el centro de los polígonos que encierran las posiciones real y espejo (ver la Figura 2.7(b)). Después, un nuevo MSC con radio igual a cero se coloca en el centro de cada polígono elegido (es decir, un nuevo punto ancla) con el fin de repetir el proceso hasta que el nodo móvil salga del área de cobertura-dos.

Para aquellos casos en que el nodo móvil es detectado por dos nodos de red de manera consecutiva, la ruta del nodo móvil se construye de la siguiente forma: sean $\mathcal{P}$ y $\mathcal{Q}$ dos polígonos seleccionados por Ghost en el tiempo t y el tiempo $t - \Delta t$, respec-

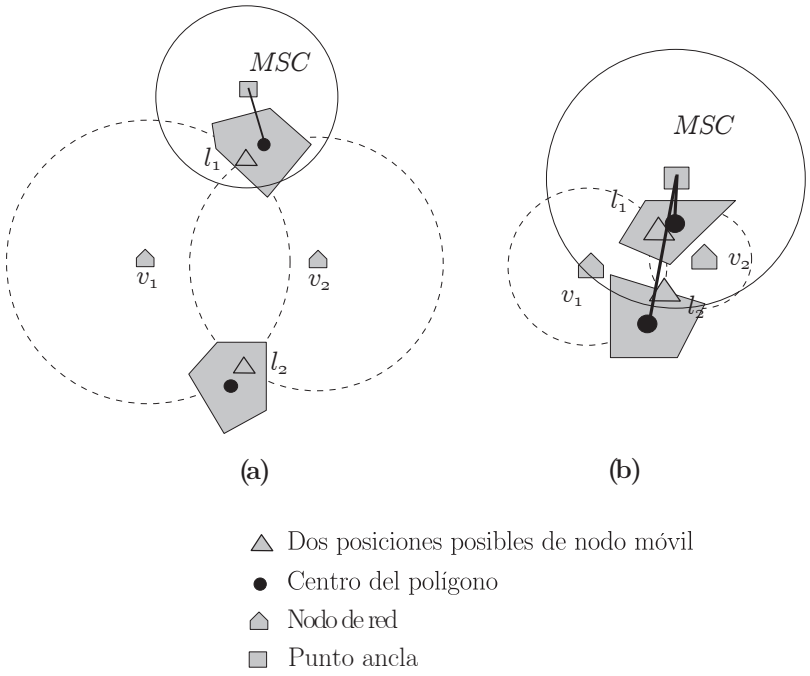


Figura 2.7: Construcción de la ruta del nodo móvil usando MSC en coberturados

tivamente. Ghost construye la ruta del nodo móvil uniendo el centro del polígono $\mathcal{Q}$ con el centro del polígono $\mathcal{P}$ (es decir, Ghost enlaza $v_{\mathcal{Q}} \rightarrow v_{\mathcal{P}}$). La operación de Ghost en pseudo código para cobertura-dos se muestra en el Algoritmo 2.

Algoritmo 2 Cobertura-dos.

Require: Definir m (número de nodos *virtuales* por nodo de red).
1: **if** un nodo móvil es detectado por dos nodos de red **then**
2: Enviar las distancias estimadas al nodo *sink*.
3: Generar el diagrama de Voronoi.
4: **if** la posición espejo es descartada por el actual MSC **then**
5: Seleccionar el polígono $\mathcal{P}$ donde se encuentra el nodo móvil.
6: Conectar el centro del MSC actual al centro de polígono $\mathcal{P}$.
7: Crear un nuevo MSC en el centro del polígono $\mathcal{P}$ con un radio igual a cero.
8: **else**
9: Seleccionar los dos polígonos donde se encuentra el nodo móvil (es decir, posición real y espejo).
10: Conectar el centro del MSC actual con el centro de los dos polígonos seleccionados.
11: Situar un nuevo MSC en el centro de cada polígono elegido (es decir, v_{real} y v_{espejo}) con radio igual a cero.
12: **end if**
13: Ir al paso 1 hasta que el nodo móvil salga del área de cobertura-dos.
14: **end if**

2.5.6. Cobertura-uno usando MSC

Ahora vamos a explicar cómo Ghost utiliza el MSC para construir la ruta del nodo móvil en escenarios de cobertura-uno. Por ejemplo, considere que la última ubicación del nodo móvil es conocida antes de entrar al área de cobertura-uno (por ejemplo, el último punto ancla). Este punto representa el punto raíz en cobertura-uno, en el cual se localiza el MSC actual con radio igual a cero. Después, el radio del MSC aumenta hasta que se obtiene una nueva detección. Esta detección se encuentra necesariamente dentro del área de cobertura-uno donde una nueva dona se construye mediante el uso de los nodos virtuales en el tiempo t y los nodos virtuales en el tiempo $t - \Delta t$. En caso de que no haya nodos virtuales en el tiempo $t - \Delta t$, Ghost crea estos nodos virtuales utilizando un círculo centrado en el nodo de red con un radio igual a la distancia Euclidiana entre el nodo de red y el punto ancla. Para ilustrar esto, la Figura 2.8 muestra el MSC actual centrado en el punto v_0 (es decir, el punto raíz) y la dona actual. Ghost construye la ruta del nodo móvil al conectar el centro del MSC actual con el centro de los polígonos situados dentro del MSC. En este ejemplo, se forman dos aristas. La primera arista conecta los nodos v_0 y v_1, mientras que la segunda arista conecta los nodos v_0 y v_2. Después, un nuevo MSC con radio igual a cero se coloca en los nodos v_1 y v_2 con el fin de repetir el proceso hasta que el nodo móvil salga del área de cobertura-uno. La Figura 2.9 muestra este proceso a medida que transcurre el tiempo.

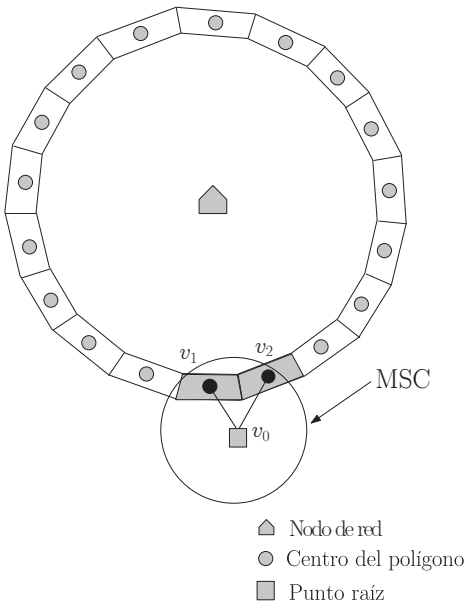


Figura 2.8: Grafo G dentro de cobertura-uno

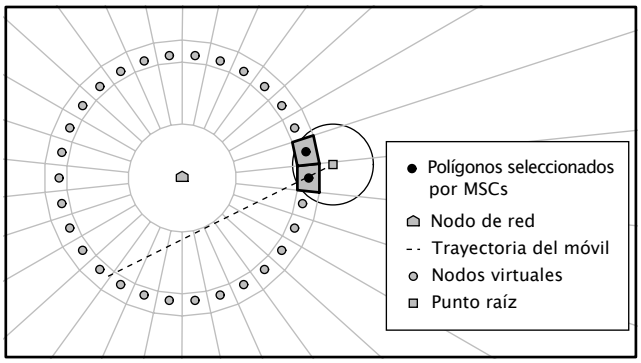
Es importante mencionar que antes de construir la dona, Ghost calcula cuál es su alcance para cobertura-uno. La Figura 2.10 muestra al nodo de red v_1 donde su área de cobertura se traslapa con dos nodos de red (v_2 y v_3). En esta Figura se puede observar que sólo los puntos en el perímetro de cobertura-uno se eligen como nodos virtuales. Ghost utiliza el *algoritmo kd-tree* [24] para encontrar los nodos virtuales y los vecinos más cercanos en $O(n\ log\ n)$. La operación de Ghost en pseudo código para el caso de cobertura-uno se muestra en el Algoritmo 3.

Algoritmo 3 Cobertura-uno.

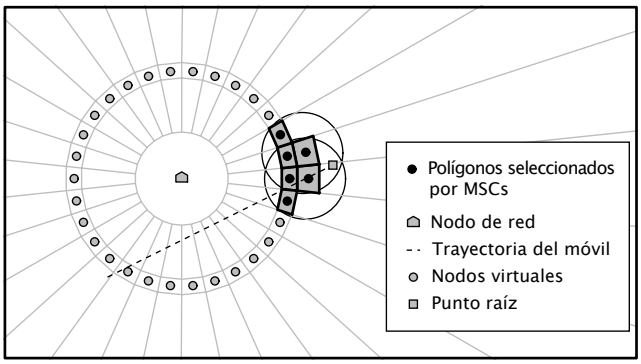
Require: Definir m (número de nodos *virtuales* por nodo de red).

1: **if** un nodo móvil es detectado por un nodo de red **then**
2: Enviar la distancia estimada al nodo *sink*.
3: Crear la *dona* actual.
4: Conectar el centro del MSC con el centro de los polígonos dentro del MSC.
5: Colocar un nuevo MSC con radio igual a cero en cada polígono seleccionado.
6: Ir a 1 hasta que el nodo móvil salga del área de cobertura-uno.
7: **end if**

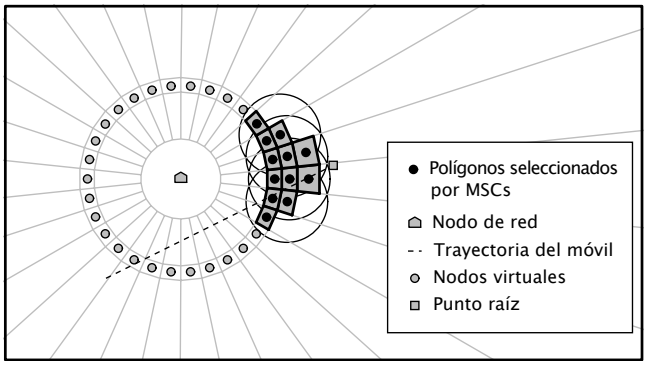
En cobertura-uno, cuanto mayor sea el número de nodos virtuales, mayor será el número de rutas posibles creadas durante la ubicación del nodo móvil. Con el fin de buscar una solución óptima, Ghost utiliza el algoritmo de backtracking (BT) [25]. Pero antes de explicar como funciona el algoritmo BT, es importante explicar cómo Ghost finaliza las rutas posibles dentro de cobertura-uno (es decir, los puntos finales del grafo G). La Figura 2.11 ilustra un nodo móvil (que se muestra como $\triangle$) cerca del límite de cobertura-uno. Los puntos v_j, v_k, v_m, v_n, v_l son el centro de cada MSC



(a) Dona en el tiempo t



(b) Dona en el tiempo $t + \Delta t$



(c) Dona en el tiempo $t + 2\Delta t$

Figura 2.9: Polígonos seleccionados por Ghost mientras transcurre el tiempo

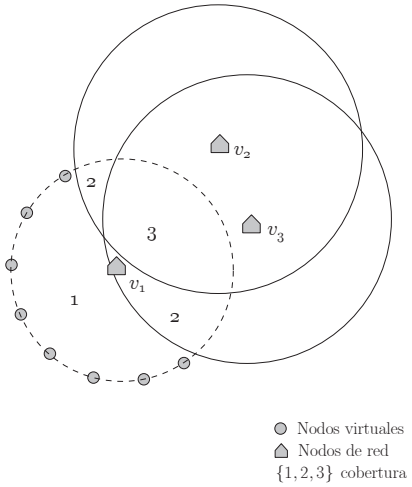


Figura 2.10: Alcance para cobertura-uno

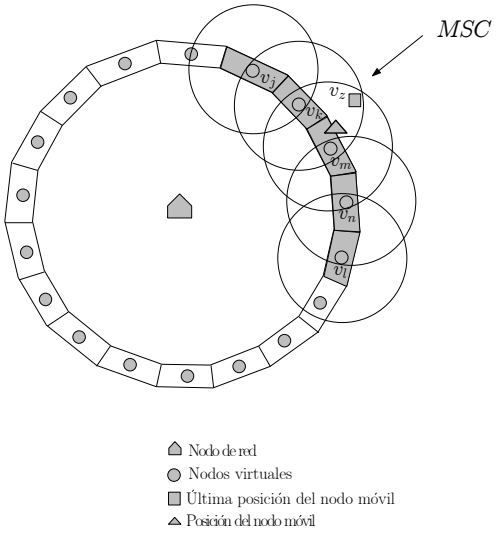


Figura 2.11: Puntos finales en cobertura-uno

actual, respectivamente. Sólo los MSCs que alcanzan la última posición conocida del nodo móvil (v_z) son elegidos como puntos finales del grafo G. Tenga en cuenta que v_z se encuentra necesariamente fuera del área de cobertura-uno. En este ejemplo, los puntos v_k y v_m se eligen como puntos finales de los caminos posibles en cobertura-uno.

En el caso de que no existan puntos anclas antes de entrar en el área de cobertura-uno, Ghost puede utilizar alguna de las siguientes posiciones: i) la última posición descubierta por cobertura-dos (esto incluye la posición real, la posición espejo o ambas), ii) el conjunto de puntos finales del grafo G del último escenario de cobertura-uno. Si aun así no existen puntos conocidos antes de entrar en el área de cobertura-uno, Ghost no puede colocar ningún MSC con el fin de reducir la incertidumbre en la posición del nodo móvil. Sin embargo, éste es un caso poco probable ya que conforme transcurre el tiempo es altamente probable que Ghost encuentre puntos anclas que puedan ser utilizados antes de entrar en un escenario de cobertura-uno.

Una vez que se obtienen los puntos finales, Ghost busca el camino óptimo del grafo creado en cobertura-uno utilizando el algoritmo BT. Este algoritmo busca la mejor solución parcial sistemáticamente a través de satisfacer una función de restricción con el fin de obtener una solución óptima global. En Ghost, esta restricción es el número de detecciones durante el movimiento del nodo móvil dentro del área de cobertura-uno. Esta restricción implica que sólo los caminos cuyas longitudes son iguales al número de detecciones en cobertura-una son caminos válidos. El algoritmo de BT permite que Ghost encuentre: i) todas las rutas entre el punto raíz y todos los puntos finales, ii) el camino más corto entre el punto raíz y un punto final dado, iii) un camino aleatorio. Ya que el algoritmo BT representa el mayor tiempo de procesamiento en Ghost, analizamos el peor de los casos del algoritmo BT en función del tiempo.

Un problema de satisfacción de restricciones (CSP, por sus siglas en inglés), se compone de un conjunto de variables, $X = \{X_1, ..., X_n\}$, donde cada variable $x_i \in X$ tiene un dominio finito asociado $D = \{D(x_1), ..., D(x_n)\}$, y un conjunto de restricciones C, donde cada C_i es de la forma $C_i = \{X_{i1}, X_{i2}, ..., X_{ij}\}$ [26]. Una solución de un problema de restricción es una asignación de un valor $D(x_i)$ que satisface una o todas las restricciones. Si no existe una solución, el CSP se dice que es inconsistente. Por ejemplo, para cualquier solución parcial $(v_1, v_2, ..., v_i)$, el algoritmo BT intenta asignar un valor a la siguiente variable $(v_1, v_2, ..., v_i, v_{i+1})$. Si esta solución parcial satisface las restricciones $C_{j(i+1)} \in C$, entonces el algoritmo BT intenta asignar un nuevo valor a la siguiente variable. En caso de que la solución parcial sea inconsistente, el algoritmo BT intenta asignar otro valor a $D(x_{i+1})$. Si aún así no se satisface, el algoritmo BT da marcha atrás (a la variable x_i) e intenta asignar otro valor a $D(x_i)$. El espacio de búsqueda para asignar un valor al dominio es a lo más $\sum_{i=0}^{n} d^i = \frac{d^{n+1}-1}{d-1}$, donde n es el número de variables en X (en Ghost éste es el número de polígonos seleccionados por el MSC) y d es el tamaño del dominio más grande. Por lo tanto, el tiempo de procesamiento para cada asignación $D(x_i)$ es a lo más $O(d^n)$ y se repite en n vértices. El tiempo total de ejecución para el algoritmo de BT es a lo más $O(nd^n)$. Utilizando una técnica de filtrado se puede optimizar el espacio de búsqueda en CSP. Por ejemplo, cuando un valor $D(x_i)$ es incompatible con los $C_{j(i)}$, podemos eliminar o podar el espacio de búsqueda para futuras variables. En Ghost, cuando el número de aristas en una ruta es mayor que el número de detecciones en cobertura-uno, se puede podar el grafo G para reducir el tamaño del dominio actual. En [26] los autores analizaron

un algoritmo de descomposición de corte cuyo tiempo de ejecución es de $O(nd^2)$.

Por otra parte, es fácil ver que el algoritmo de cobertura-uno es el algoritmo principal en Ghost, mientras que los algoritmos de cobertura-dos y tres son casos particulares del algoritmo de cobertura-uno. De hecho, el algoritmo de BT se puede utilizar en cobertura-dos para descartar la ubicación espejo mediante la búsqueda de una solución óptima en la trayectoria utilizando los puntos extremos de la trayectoria y el número de detecciones del nodo móvil como restricción. El algoritmo 4 muestra el pseudo código de Ghost.

2.6. Simulación y resultados

En esta sección, medimos el error de ubicación del nodo móvil estimado por Ghost. Primero, estudiamos con qué exactitud Ghost calcula la posición del nodo móvil en función del número de nodos virtuales para los casos de cobertura-{tres, dos, uno}. A continuación, estudiamos el desempeño de Ghost para una red con densidad alta (más de 20 nodos de red). Por último, en la Sección 2.6.4, comparamos el desempeño de Ghost contra los algoritmos PLT y GPLT [5] en términos del error estimado en cobertura-{tres, dos, uno}.

2.6.1. Modelo de ruido

Como se mencionó, para cobertura-{tres, dos, uno}, las posiciones del nodo móvil se asocian al centro del (los) polígono(s) seleccionado(s) por Ghost. Por lo tanto, vamos a definir el error como la distancia Euclidiana promedio entre el centro del polígono que encierra la ubicación estimada del nodo móvil y la posición real del nodo móvil para todos los puntos en la simulación. La distancia entre los nodos de red y el nodo móvil se calcula como:

$$r_{i,t} = d_{i,t} + n_{i,t} + e_{i,t} \qquad i = 1, 2, 3, \ldots, N \tag{2.1}$$

donde $r_{i,t}$ denota la distancia estimada entre el i ésimo nodo de red y el nodo móvil en el momento t. $d_{i,t}$ es la distancia Euclideana real entre el i-ésimo nodo de red y el nodo móvil en el momento t. $n_{i,t}$ es el ruido añadido, la cual sigue una distribución Gaussiana con media cero y 10 metros de desviación estándar. Finalmente, $e_{i,t}$ denota el error por línea de vista (NLOS) [27], que está modelada por una distribución exponencial de la siguiente manera.

$$e_{i,t}(v) = \begin{cases} \frac{1}{\lambda_{i,t}} exp(-\frac{v}{\lambda_{i,t}}) & v > 0 \\ 0 & para\ otro\ caso \end{cases} \tag{2.2}$$

donde $\lambda_{i,t} = c \cdot \tau_m (d_{i,t})^{\epsilon} \rho$. El parámetro c es la velocidad de la luz, τ_m es el valor de la mediana de la demora de transmisión RMS entre el i-ésimo nodo de red y el nodo móvil, que se selecciona como $0.1 \mu s$ en la simulación. ϵ es el exponente en la ecuación de la trayectoria de pérdidas (path loss exponent) que se selecciona como 0.5. Finalmente, ρ es el factor de desvanecimiento (shadow fading factor), que es una variable aleatoria log-normal con media cero y una desviación estándar de 4 dB. Los valores utilizados en este modelo de ruido fueron tomados de [5].

Algoritmo 4 Ghost

Require: Definir m (número de nodos *virtuales* por nodo de red).

if un nodo móvil es detectado por al menos tres nodos de red: **then**
  Envíar las distancias estimadas al nodo *sink*.
  Generar el diagrama de Voronoi.
  Seleccionar el polígono $\mathcal{P}$ donde se localiza el nodo móvil.
  Conectar el centro del polígono $\mathcal{P}$ (i.e., $v_{\mathcal{P}}$) en el grafo G.
  Concatenar $v_{\mathcal{P}}$ como punto ancla.
end if
if un nodo móvil es detectado por dos nodos de red **then**
  Enviar las distancias estimadas al nodo *sink*.
  Generar el diagrama de Voronoi.
  if la posición espejo es descartada por el actual MSC **then**
    Seleccionar el polígono $\mathcal{P}$ donde se encuentra el nodo móvil.
    Conectar el centro del MSC actual al centro de polígono $\mathcal{P}$.
    Crear un nuevo MSC en el centro del polígono $\mathcal{P}$ con un radio igual a cero.
  else
    Seleccionar los dos polígonos donde se encuentra el nodo móvil (es decir, posición real y espejo).
    Conectar el centro del MSC actual con el centro de los dos polígonos seleccionados.
    Situar un nuevo MSC en el centro de cada polígono elegido (es decir, v_{real} y v_{espejo}) con radio igual a cero.
  end if
  Ir al paso 8 hasta que el nodo móvil salga del área de exposición-dos.
end if
if un nodo móvil es detectado por un nodo de red **then**
  Enviar la distancia estimada al nodo *sink*.
  Crear la *dona* actual.
  Conectar el centro del MSC con el centro de los polígonos dentro del MSC.
  Colocar un nuevo MSC con radio igual a cero en cada polígono seleccionado.
  Ir a 22 hasta que el nodo móvil salga del área de exposición-uno.
end if

Tabla 2.1: Parámetros de simulación

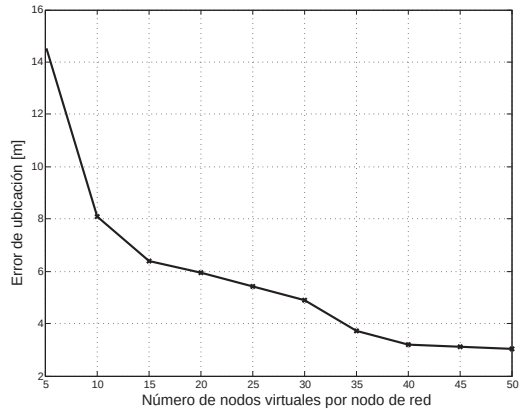
Parametro	Valor
Rango de transmisión	100m
Longitud del área vigilada	1900m
Acho del área vigilada	1100m
Origen de coordenadas	Esquina superior izquierda
Distribución Gaussiana	$\mathcal{N}$(0,100)
Intervalo de velocidad	[1 - 5]m/s
Intervalo de tiempo entre dos detecciones consecutivas	1s
Modelo de movimiento	RWP

En el siguiente apartado evaluamos nuestro sistema mediante la ejecución de un simulador escrito en el lenguaje Python. Primero evaluamos las simulaciones para cobertura-{tres, dos, uno } de forma independiente. El nodo móvil se mueve de acuerdo con el modelo Random Waypoint (RWP) [28] y se ejecutó 50 veces cada experimento para obtener valores promedios. Los parámetros utilizados para estos experimentos se muestran en la Tabla 2.1.

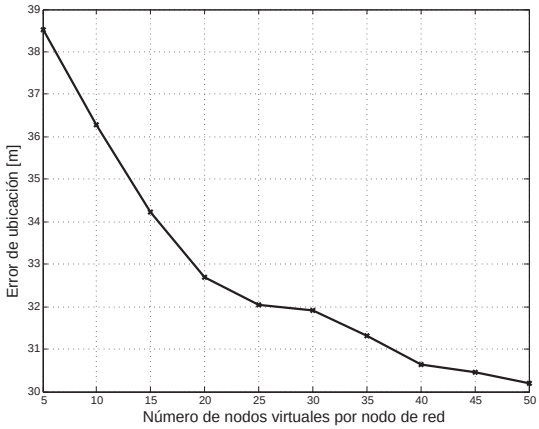
2.6.2. Experimentos aislados para cobertura-{tres,dos y uno}

Para los experimentos en cobertura-tres, el nodo de red n_1 se encuentra en la coordenada (558, 287), el nodo de red n_2 se localiza en (498, 285) y el nodo n_3 está localizado en (530, 333), todas las coordenadas están en metros. La trayectoria del nodo móvil utiliza un modelo RWP cuyos extremos de la trayectoria se encuentran en (501, 339) y (553, 259) en metros. La trayectoria se elige dentro de la zona de traslape creado por los tres nodos de red. Medimos el error de la posición del nodo móvil variando el número de nodos virtuales por nodo de red en un intervalo de 5 a 50 en incrementos de 5. La Figura 2.12(a) muestra la media de error para cobertura-tres vs. el número de nodos virtuales por nodo de red utilizando las coordenadas mencionadas anteriormente. Esta Figura muestra cómo el error estimado disminuye a medida que el número de nodos virtuales aumenta, lo que es el resultado de tener regiones poligonales más pequeñas que encierran la ubicación del nodo móvil a medida que se incrementa el número de nodos virtuales.

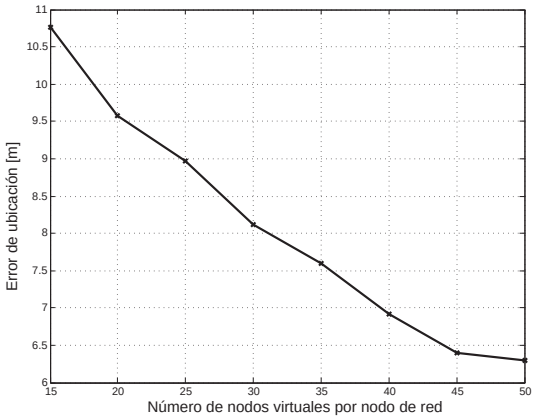
Para los experimentos en cobertura-dos, el nodo de red n_1 se localiza en (521, 300) y el nodo de red n_2 se localiza en (597, 300); las coordenadas se presentan en metros. El nodo móvil utiliza un modelo RWP para describir su trayectoria, cuyo punto inicial y final se encuentran en (529, 356) y (588, 247), respectivamente. Como mencionamos, cuando dos nodos de red detectan al nodo móvil, dos caminos posibles se forman a lo largo de la trayectoria del nodo móvil. Presentamos el error medio mediante la estimación de la distancia entre los centros de ambos polígonos asociados con respecto a la ubicación del nodo móvil. En estos experimentos no se utilizó el concepto de MSC, por tanto, no se descartó ninguna posición espejo. La Figura 2.12(b) muestra cómo los errores de localización disminuyen a medida que el número de



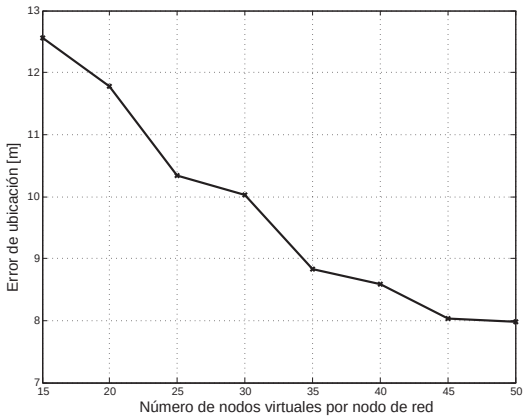
(a) Error en cobertura-tres



(b) Error en cobertura-dos



(c) Error en cobertura-uno con V_{known}



(d) Error en cobertura-uno en el intervalo $[V_{min}, V_{max}]$

Figura 2.12: Error en cobertura-{tres,dos,uno}

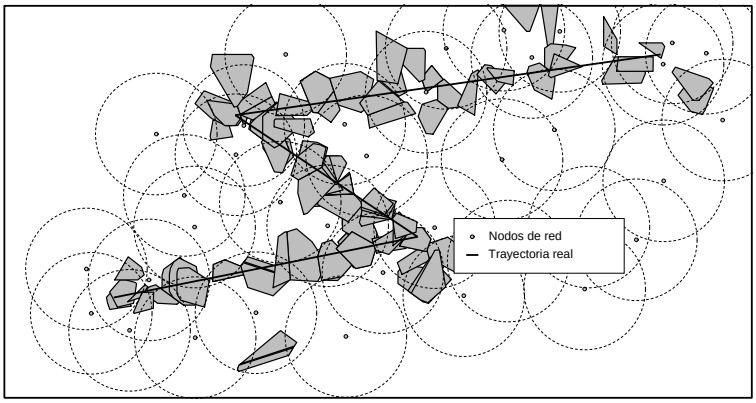
nodos virtuales aumenta, que es el resultado de encerrar al nodo móvil en polígonos cada vez más pequeños.

Para los experimentos en cobertura-uno suponemos que se conoce tanto el punto de entrada (punto raíz del grafo G) y el punto de salida. Los errores de localización para cobertura-uno dependen del número de nodos virtuales como se mencionó anteriormente. Para los casos en que se conoce la velocidad del nodo móvil, el algoritmo de BT pondera la posición más cercana a la velocidad del nodo móvil para cada detección a lo largo del área de cobertura-uno. Por otro lado, cuando la velocidad del nodo móvil es desconocida, pero cae dentro del intervalo de $[0, V_{max}]$, el algoritmo de backtracking selecciona todos las aristas que caen dentro del MSC actual(es) a lo largo del área de cobertura-uno. En ambos casos se calcula la media del error de localización mediante la estimación de la distancia Euclidiana entre la ubicación real y el centro del (los) polígono(s) seleccionado(s) por cada detección dentro del área de cobertura-uno. Para este experimento el nodo de red se encuentra en (603, 325), y la velocidad del nodo móvil se considera de $4m/s$. La trayectoria del nodo móvil utiliza el modelo RWP cuya entrada está situada en el punto (523, 394) y la salida se encuentra en (673, 404); en metros. La Figura 2.12(c) muestra cómo los errores de localización disminuyen a medida que el número de nodos virtuales aumenta. Nuestras simulaciones muestran que tener más de 15 nodos virtuales disminuyen el error estimado de forma monótona. Sin embargo, el costo de calcular una ruta óptima aumenta exponencialmente a medida que el número de nodos virtuales aumenta (como se mostró en la Sección 3.4.3.1). Es importante mencionar que los casos con menos de 15 nodos virtuales por nodo de red no se reportan en la Figura 2.12(c) y 2.12(d). Esta situación surge porque la distancia entre dos nodos virtuales consecutivos en la dona era mayor que el radio del MSC. Como resultado, el MSC no alcanzó el centro de algún polígono dentro de la *dona*, por lo que las aristas en el grafo G no se conectaron.

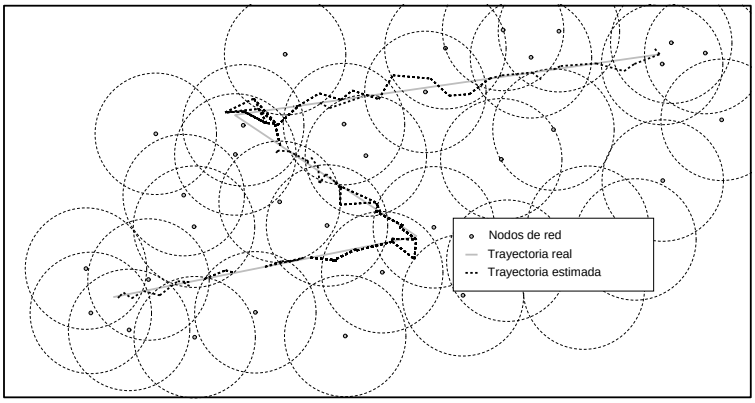
En la Figura 2.12(d) se muestran los resultados de las simulaciones para el caso cuando el nodo móvil puede variar su velocidad en el intervalo [1, 5] m/s. En esta Figura, se puede observar que para menos de 15 nodos virtuales por nodo de red, el grafo G esta desconectado, similar al experimento anterior cuando se conoce la velocidad del nodo móvil. También podemos observar en esta figura que tener más de 15 nodos virtuales por nodo de red disminuye el error de localización monotónicamente.

2.6.3. Red con alta densidad de nodos

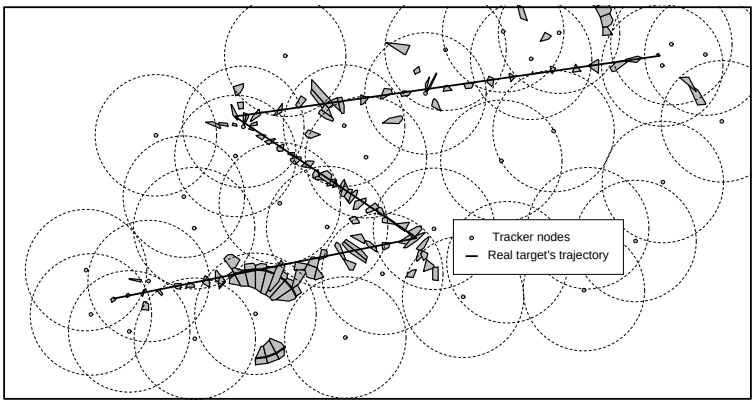
En esta sección se muestra un experimento con 36 nodos de red ubicados al azar en una área rectangular, se utilizan los parámetros mostrados en la Tabla 2.1. La Figura 2.13(a) ilustra la ruta real del nodo móvil y los polígonos generados por Ghost usando cinco nodos virtuales por nodo de red, mientras que la Figura 2.13(b) muestra la trayectoria (línea continua) del móvil vs. la trayectoria estimada por Ghost (línea discontinua). En la Figura 2.13(b) podemos ver que el grafo G se desconecta en algunos lugares debido a que se utilizan muy pocos nodos virtuales (en particular en el caso de cobertura-uno). La Figura 2.13(d) muestra el mismo escenario de red pero ahora se usaron 25 nodos virtuales por nodo de red. En esta figura se observa que la ruta estimada del nodo móvil no se desconecta, y es similar a la trayectoria real del nodo móvil. En la Figura 2.13(c) podemos ver, como se esperaba, que el tamaño de los polígonos seleccionados por Ghost son más pequeños en comparación con los polígonos seleccionados en la Figura 2.13(a). La Figura 2.14 ilustra cómo el error de localización disminuyen a medida que el número de nodos virtuales aumenta



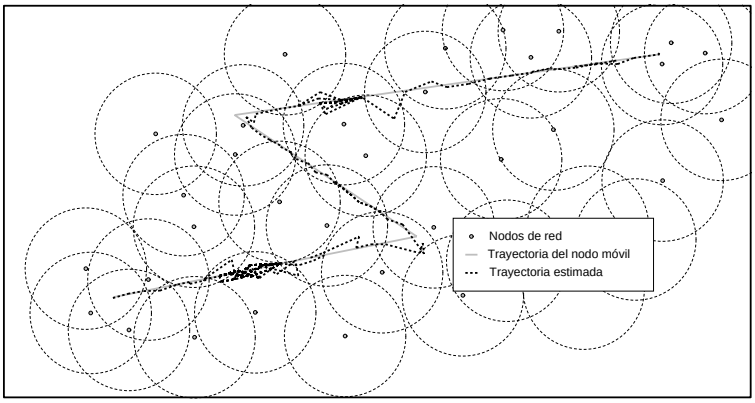
(a) Red con 5 nodos virtuales por nodo de red



(b) Red con 5 nodos virtuales por nodo de red



(c) Red con 25 nodos virtuales por nodo de red



(d) Red con 25 nodos virtuales por nodo de red

Figura 2.13: Escenario de una red de alta densidad

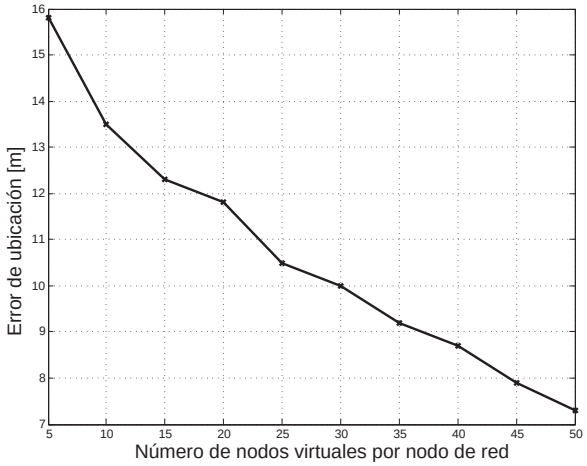
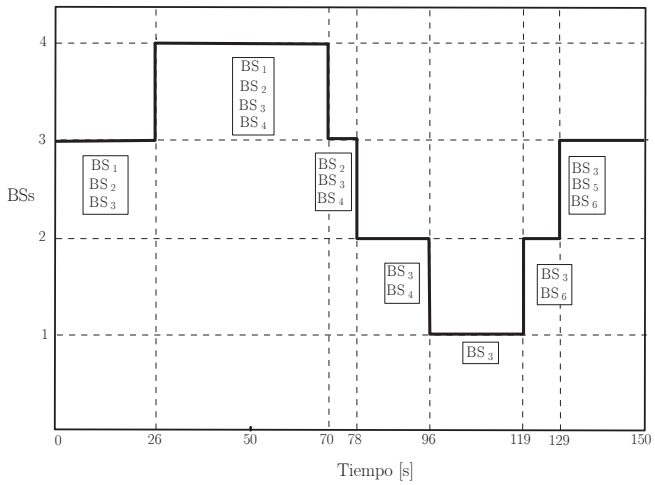
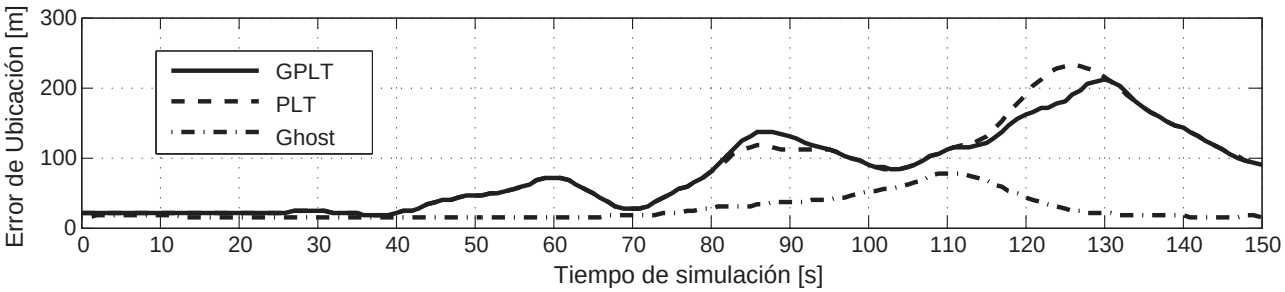


Figura 2.14: Estimación del error para una red de alta densidad



(a) Tiempo de Simulación



(b) Desempeño

Figura 2.15: Ghost vs. PLT y GPLT

en un escenario de red de gran tamaño. Es importante mencionar que el MSC para cobertura-{uno, dos} fueron utilizados en este experimento.

2.6.4. Ghost vs. PLT y GPLT

Los algoritmos Ghost, GPLT y PLT [5] centran su esfuerzo en ubicar a un nodo móvil en escenarios con menos de tres nodos de red (es decir, cobertura-{uno, dos}). Por ello comparamos el desempeño de Ghost con los algoritmos GPLT y PTL [5]. En [5] los autores propusieron un escenario en el que el número de estaciones base que detectan al nodo móvil cambia en el tiempo. La Figura 2.15(a) ilustra este arreglo, donde BS_i denota la i-ésima estación base que detecta al nodo móvil en un tiempo específico. Se utilizaron los mismos parámetros del modelo de ruido ($\mathcal{N}$ (0,100)), las mismas dimensiones del área vigilada, una trayectoria similar del nodo móvil, el mismo modelo de velocidad y el mismo rango de transmisión utilizado en [5] para comparar el error de localización a través del tiempo. En este experimento se utilizaron 30 nodos virtuales por nodo de red ya que en las simulaciones se observó que esta configuración proporciona un buen balance entre el error de ubicación y el tiempo de procesamiento para la mayoría de las aplicaciones civiles. La Figura 2.15(b) muestra el error medio en función del tiempo de simulación. En esta Figura podemos ver que Ghost supera en términos de error de ubicación a los algoritmos GPLT y PLT en la mayor parte de la simulación. Esto es muy claro en el intervalo de $[78 - 129]$ segundos, donde sólo una o dos estaciones base detectan al nodo móvil. El hecho de que Ghost utiliza el algoritmo de backtracking cuya función criterio es el número de detecciones durante los escenarios de cobertura-{uno, dos} resulta en un conjunto más pequeño de rutas probables seleccionadas por el nodo móvil. En contraste, los algoritmos PLT y GPLT no aplican una función criterio para reducir los caminos probables en cobertura-{uno, dos}. Además, si se aumenta el número de nodos virtuales, la superficie por polígono se reduce y como resultado, la distancia euclidiana entre el centro de los polígonos seleccionados y la ubicación real del nodo móvil se reduce como fue mostrado en la Sección 2.6.2.

2.7. Conclusiones

En este capítulo se propuso un modelo para localizar un nodo móvil en redes inalámbricas con menos de tres nodos de red. Se demostró que incluso para casos en donde sólo un nodo de red detecta al nodo móvil, un conjunto de caminos posibles pueden construirse. Los resultados de las simulaciones muestran que el uso de diagramas dinámicos de Voronoi disminuyen el error estimado de la posición del nodo móvil en comparación con otras propuestas encontradas en la literatura. Finalmente, los resultados demuestran que al reducir el tamaño de las regiones de Voronoi mediante el uso de nodos virtuales diminuye el error estimado por Ghost en los escenarios de cobertura-{uno, dos, tres}.

Capítulo 3

ESL

3.1. Resumen

En los últimos años se ha mostrado que es muy fácil localizar a un nodo móvil, sobre todo cuando éste envía su posición a un servidor remoto. Otros estudios muestran que la ubicación del nodo móvil puede ser estimada por un conjunto de nodos fijos aun cuando éste no transmita su posición. Una consecuencia de este escenario es que la privacidad del usuario móvil puede verse comprometida, entendiendo como privacidad: la garantía que tiene un nodo móvil de que su ubicación este resguardada y no sea usada sin la autorización del usuario móvil.

En esta tesis proponemos una técnica para que el nodo móvil mida el riesgo de ser detectado por un conjunto de nodos fijos mediante un algoritmo que emula cómo la red estima la posición actual del nodo móvil, dicho algoritmo sólo se ejecuta en el nodo móvil sin intervención de la red. Este conocimiento, le permite al nodo móvil tener una poderosa herramienta para tomar contramedidas antes de transmitir, tales como: entrar en periodos de silencio, demorar su transmisión, cambiar su trayectoria con el fin de aumentar el error de ubicación visto por la red o viceversa dependiendo de la aplicación. En este trabajo llamamos a este escenario el problema de estimación de la auto-ubicación y hasta donde sabemos, proponemos el primer método llamado ESL (estimating self-location). En ESL, un nodo móvil debe crear un mapa local de la ubicación de los nodos de red, donde éstos se complementan con nodos *virtuales* con el fin de crear diferentes diagramas de Voronoi de la topología de red. Esta técnica le permite al nodo móvil usar un nuevo algoritmo de localización que emula cómo la red calcula la posición actual del nodo móvil. Posteriormente, ESL utiliza las estimaciones hechas por el algoritmo de localización y la posición actual del nodo móvil (obtenida mediante un receptor GPS) para calcular el error de localización incurrido por la red.

Por otro lado, también estudiamos en este capítulo cómo se ve afectada la auto-ubicación de un nodo móvil cuando el número de nodos de red varia con el tiempo. En particular, se estudian los casos cuando el nodo móvil es detectado por menos de tres nodos de red.

3.2. Trabajo relacionado

En esta sección se resume el trabajo más relevante relacionado con la privacidad en la localización de redes inalámbricas. En este trabajo definimos *privacidad* como la

garantía por la cual la ubicación del nodo móvil no es utilizada por la red para otros fines no autorizados por el nodo móvil.

En los últimos años algunas investigaciones se han centrado en estudiar el caso cuando un nodo móvil envía su posición actual a los servicios basados en localización (LBS) con el fin de obtener información dependiente de su ubicación. Sin embargo, estos servicios plantean graves problemas de privacidad ya que la ubicación del nodo móvil está disponible al proveedor del servicio en cualquier momento. Por ejemplo, en [4] los autores utilizaron un proceso de regresión de Gauss para reconstruir la trayectoria del nodo móvil usando sus posiciones enviadas a través del tiempo. Además, los mismos autores proponen un método que minimiza el riesgo de reconstruir la trayectoria del nodo móvil al seleccionar las posiciones enviadas al LBS. Otro problema relacionado con la ubicación del usuario móvil es que esta información puede ser utilizada para otros fines, tales como analizar el comportamiento y gustos del usuario móvil a través del tiempo. En [29] por ejemplo, los autores estudiaron algunas situaciones de riesgo para los usuarios móviles con el uso de LBS que no son de confianza.

Por otro lado, existen trabajos que buscan una estrategia para estimar el número y la posición de nodos de red que observan una región (también conocido como el problema de cobertura). Por ejemplo, en [30], los autores propusieron un algoritmo en el que la incertidumbre de la posición de los nodos de red se modela como círculos concéntricos con el fin de determinar el radio mínimo que garantiza cobertura-k. Otras investigaciones se centran en reducir al mínimo el número de nodos de red al optimizar las zonas de traslape. Por ejemplo, la red de panal hexagonal se sabe que es óptima para cobertura-1 [31]. Sin embargo, cualquier pequeño desplazamiento de un nodo de red puede crear agujeros, lo que resulta en regiones ciegas (sin cobertura) dentro de la red inalámbrica. En [31], se propuso un estudio de regiones hexagonales donde los autores contraen la estructura hexagonal para garantizar una cobertura total, incluso cuando algunos nodos de red no se encuentren en sus posiciones designadas. Los autores en [32] propusieron un protocolo para proteger caminos vulnerables en áreas vigiladas mediante el uso de algunos sensores móviles. Este algoritmo utiliza diagramas de Voronoi para encontrar puntos vulnerables en el área vigilada con el fin de proteger regiones importantes denominadas TBP (para-ser-protegida, por sus siglas en inglés) con el fin de mejorar la cobertura de las áreas TBP. Todos estos estudios ejemplifican los problemas de privacidad de un nodo móvil, ya que éste puede ser detectado incluso cuando no revela su posición a un servidor remoto.

Por otro lado, la exposición de un nodo móvil se puede definir como la habilidad que tiene la red para observar a un nodo móvil a medida que se mueve en una trayectoria arbitraria [3]. Los autores en [33] definen formalmente el problema de exposición como la integral de una función de la intensidad en el intervalo $[t_1, t_2]$ a lo largo de una trayectoria dada $p(t)$. El problema de exposición está directamente relacionado con la ruta de exposición mínima [7], donde un nodo móvil busca un camino entre dos puntos dados tal que la exposición total recibida sea mínima. Por ejemplo, los autores en [34] propusieron un algoritmo llamado anti-detección en el que un nodo móvil evalúa la mejor trayectoria con el fin de atravesar una área vigilada sin ser detectado. Este algoritmo utiliza los principios de enrutamiento para moverse a través de la red basado en el nivel local de exposición y su posición actual con respecto a su destino final. Sin embargo, este algoritmo asume que el nodo móvil tiene conocimiento parcial/completo de la topología de red y además del destino final de antemano. Consideramos que este supuesto no es del todo cierto, ya que la mayoría de los nodos

de red no suelen compartir su posición a los nodos móviles.

Aunque los problemas de privacidad de un nodo móvil que revela o no su posición están asociados a diferentes escenarios, ninguna de las propuestas anteriores estudia el caso cuando un nodo móvil conoce el error con el que la red estima su posición. De esta manera, el nodo móvil será capaz de medir su auto-ubicación antes de que éste transmita. Este conocimiento proporciona al nodo móvil la posibilidad de tomar contramedidas, como posponer las transmisiones o cambiar su trayectoria entre otros.

3.3. Componentes principales de ESL

En esta sección primero revisamos las condiciones para que opere el algoritmo ESL, después se discuten los componentes que serán utilizados por ESL para lograr sus objetivos en la sección 3.4.

Consideramos un conjunto de N nodos de red dispersos al azar en un espacio $\mathbb{R}^2$ con identificadores únicos. Para cada nodo de red situado en el punto p (es decir, (x, y)), se representa el rango máximo de transmisión como un disco unitario UDG (p, R_c) centrado en el punto p con radio unitario R_c. Asumimos que el rango de transmisión es uniforme (es decir, cada nodo tiene una antena omni-direccional). Un nodo v puede recibir la señal de un nodo u si y sólo si el nodo v está dentro del UDG del nodo emisor u. También asumimos que cada vez que el nodo móvil envía un paquete a la red en el tiempo t, la red es capaz de detectar al nodo móvil en el mismo instante. Además, se asume que un nodo móvil que atraviesa la red puede obtener su posición geográfica por medio de un receptor GPS.

ESL asume que los nodos móviles son capaces de detectar la presencia de los nodos vecinos de red a través de escuchar sus paquetes transmitidos. También se considera que un nodo móvil es capaz de estimar la distancia entre su posición actual y cada nodo de red (mediante el uso de métodos de estimación de potencia de la señal). En un escenario real; estas estimaciones son muy sensibles a las condiciones ambientales, por ello modelamos estas condiciones mediante el uso de una variable aleatoria con una distribución Gaussiana con el fin de evaluar los efectos de propagación. También se realizaron experimentos reales para medir el comportamiento de ESL en condiciones de propagación y de operación. A continuación, se describen los componentes básicos que comprenden el funcionamiento de ESL.

3.3.1. Detección del nodo de red

ESL debe descubrir la topología de red para conocer la ubicación de los nodos de red vecinos (es decir, k nodos de red, donde $k \geq 1$). Para lograr esta tarea, la interfaz inalámbrica del nodo móvil debe operar en modo monitor para escuchar el tráfico transmitido por los nodos de red al menos en dos lugares diferentes. Sea l_1 y l_2 la ubicación del nodo móvil donde se produjo el primer y el segundo encuentro con el k ésimo nodo de red, donde d_1 y d_2 denota la distancia estimada entre el nodo móvil y el k ésimo nodo de red (ver la Figura 3.1); $l_1 \neq l_2$. En este punto, el nodo móvil puede calcular dos soluciones geométricas para la ubicación del k ésimo nodo de red mediante la intersección de los dos círculos superpuestos centrados en las posiciones l_1 y l_2 con radios d_1 y d_2, respectivamente. Una solución geométrica se encuentra en el lado izquierdo mientras que la otra se localiza en el lado derecho (triángulos en la

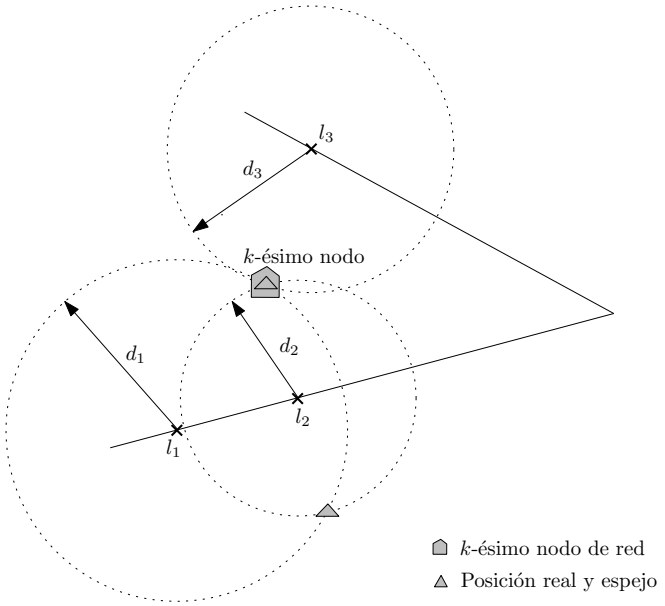


Figura 3.1: Detección del nodo de red

Figura 3.1) con respecto a la trayectoria del nodo móvil, es decir, $l_1 \rightarrow l_2$. Definimos la solución del lado izquierdo como la posición real, mientras que la otra se convierte en la posición espejo. Tenga en cuenta que las posiciones real y espejo se utilizan sólo para fines descriptivos ya que en este punto el nodo móvil no tiene ningún medio para distinguir entre ambas posiciones.

Para diferenciar entre la posición real y espejo del k-ésimo nodo de red, ESL requiere que el nodo móvil cambie su trayectoria y que el k-ésimo nodo de red esté dentro de su área de cobertura (UDG). Si ambos requisitos se cumplen, una de las dos soluciones geométricas del k-ésimo nodo de red puede ser descartada, véase el ejemplo representado en la Figura 3.1 donde el nodo móvil cambia su trayectoria (es decir, $l_2 \rightarrow l_3$), mientras que el nodo móvil detecta al k-ésimo nodo de red en la posición l_3. En este ejemplo d_3 indica la distancia entre el nodo móvil y el k-ésimo nodo de red. Se puede observar en esta figura que la solución que se encuentra fuera de los límites del círculo centrado en la posición l_3 puede ser descartada. En este punto, no hay ambigüedad sobre la posición del k-ésimo nodo de red, ya que una de las dos soluciones geométricas fue descartada. De lo contrario, la ambigüedad de la posición del k-ésimo nodo de red seguirá presente.

3.3.2. Creación del árbol binario

Un árbol binario (BT, por las siglas en inglés de binary tree), es un grafo conexo, acíclico y no dirigido construido de tal forma que sólo existe un camino entre dos nodos y con grado menor a tres. ESL utiliza un BT para construir un mapa local de los nodos de red conforme éste recorre la red inalámbrica. En ESL un árbol T es una estructura de datos sin ciclos en los que un nodo de red n_i es una entidad con los siguientes campos:

- key[n_i]: un identificador único.
- left[n_i]: un puntero al subárbol izquierdo.

- right[n_i]: un puntero al subárbol derecho.
- color[n_i]: una función criterio.

En ESL cada nodo de red puede tener tres colores diferentes.

- color blanco: este color denota que ocurrió el primer encuentro entre el nodo móvil y el nodo de red n_i.
- color gris: este color denota que ocurrió el segundo encuentro entre el nodo móvil y el nodo de red n_i, es decir, se pueden calcular las posiciones real y espejo de n_i.
- color negro: este color denota el caso cuando se descartó la ambigüedad en la posición del nodo de red n_i y el nodo móvil conoce la ubicación real del nodo n_i.

Un nodo n_j que es insertado en T, debe satisfacer las siguientes reglas:

- el nodo n_j está en el subárbol izquierdo del nodo n_i si y sólo si el atributo key[n_j] $<$ key[n_i]
- el nodo n_j está en el subárbol derecho del nodo n_i si y sólo si el atributo key[n_j] $>$ key[n_i]

Para cumplir las reglas antes citadas, ESL utiliza el procedimiento Inserta-actualiza (ver Algoritmo 5) que funciona de la siguiente manera: Una vez que se produjo el primer encuentro con el nodo de red n_i, el algoritmo utiliza la función Node(id) para insertar un nuevo nodo n_i en T y establecer sus atributos key[n_i] = id y color[n_i] = blanco. Si se produce un segundo encuentro con el nodo de red n_i, la función Color actualiza el atributo color[n_i] a gris. Si se produce un tercer encuentro y la ambigüedad en la posición del nodo n_i se pueda descartar, la función Color actualiza el atributo color[n_i] a negro. ESL nombra el primer nodo insertado en T como nodo raíz. El tiempo de procesamiento para actualizar el color de un nodo de red n_i requiere $\Theta(n)$; donde n es el número de nodos en T, esto se debe a que cada nodo es visitado a lo más una vez. El tiempo en términos de procesamiento para el algoritmo BT depende de la altura de T. Sin embargo, se sabe que la altura esperada de un árbol binario es proporcional a $O(log\ n)$ [35].

Por ejemplo, supongamos que en el tiempo t los nodos 1, 4 y 6 son de color blanco. Los nodos 3, 7 y 10 son de color gris y el nodo 15 es de color negro (ver la Figura 3.2(a)). Supongamos que en el tiempo $t + \Delta t$, el nodo móvil detecta a los nodos 4 y 6 por segunda vez, mientras que el nodo 8 es detectado por primera vez. Esto implica que los nodos 4 y 6 se colorean en gris (es decir, se pueden establecer las dos solución geométricas de cada nodo) mientras que el nodo 8 se inserta en color blanco como un nuevo nodo en T. Los nodos 1, 3, 7, 10 y 15 no presentan ningún cambio de color (ver Figura 3.2(b)).

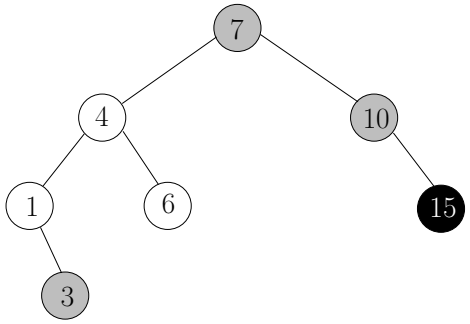
3.3.3. Creación de la matriz $M(t)$

Sea $N_{in}(t) = \{n_1, n_2, \ldots, n_k\}$, el conjunto de k nodos de red detectados por el nodo móvil en el momento actual t (i.e., k nodos de red dentro del UDG del nodo móvil), en el que cada nodo de red es de color gris o negro. Sea $M(t)$ una matriz compuesta de

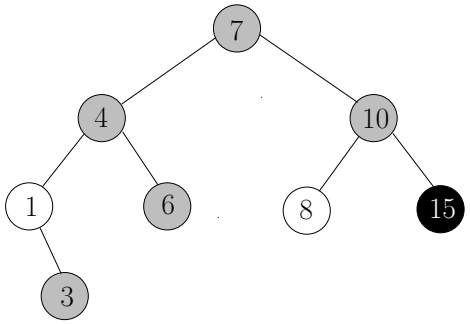
Algoritmo 5 Inserta-actualiza

Require: raíz[T]

```
1: Inserta-actualiza (n_i , id)
2: if id < key[n_i] then
3:    if left[n_i] = NIL then
4:       left[n_i] ← Node(id)
5:    else
6:       Inserta-actualiza(left[n_i], id)
7:    end if
8: end if
9: if id > key[n_i] then
10:   if right[n_i] = NIL then
11:      right[n_i] ← Node(id)
12:   else
13:      Inserta-actualiza(right[n_i], id)
14:   end if
15: else
16:   Color[n_i]
17: end if
```



a)



b)

Figura 3.2: Árbol T en ESL

$2^{k-negro}$ filas y k columnas, donde $negro$ denota el número de nodos de red de color negro $\in N_{in}(t)$. Cada fila en $M(t)$ está compuesta de un número binario de longitud k en el intervalo [0, $2^{k-negro}-1$] en incrementos de una unidad. Sea r_i = [$n_1, n_2, \ldots, n_k$] la i-ésima fila de $M(t)$ tal que cada $n_j, 1 \leq j \leq k$ representa la posición real o espejo del j-ésimo nodo de red, indicado por "0" o "1", respectivamente. Las columnas de $M(t)$ representan el k-ésimo nodo de red $\in N_{in}(t)$ ordenado por su atributo key[n_k] = id de forma ascendente de derecha a izquierda. Sólo si existen nodos negros $\in N_{in}(t)$, éstos deben ser colocados en las columnas de la extrema izquierda de $M(t)$ ordenados en forma ascendente por su atributo key. Por ejemplo, supongamos que el nodo móvil detecta cuatro nodos de red en el momento actual t, es decir, $|N_{in}(t)| = 4$, cuyos atributos key[id] son 4, 6, 9 y 15, respectivamente, los nodos 4 y 9 son de color gris mientras que los nodos 6 y 15 son de color negro. La matriz $M(t)$ para este ejemplo se muestra a continuación:

$$M(t) = \begin{array}{c} \begin{array}{cccc} n_6 & n_{15} & n_4 & n_9 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix} \end{array}$$

Es importante notar que las columnas a la extrema izquierda de $M(t)$ contienen "0" denotando la ubicación real de los nodos 6 y 15 (nodos negros), respectivamente, observe que las filas de $M(t)$ se componen de un número binario de longitud cuatro en el intervalo [0, 2^{4-2}-1].

Para mostrar otro ejemplo, ahora vamos a considerar que el nodo móvil detecta tres nodos de red en el tiempo t cuyos atributos key[id] son 3, 7 y 10, respectivamente. Supongamos que en este caso todos los nodos de red están coloreados en gris, la matriz $M(t)$ para este ejemplo se muestra a continuación:

$$M(t) = \begin{array}{c} \begin{array}{ccc} n_3 & n_7 & n_{10} \end{array} \\ \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix} \end{array}$$

En este caso, las filas de $M(t)$ se componen de un número binario de longitud tres cuyo valor está en el rango de cero a siete.

3.3.4. División del espacio Euclidiano

Este procedimiento muestra cómo ESL divide el espacio Euclidiano. Sea $N_{out}(t)$ el conjunto de nodos de red en T (sólo nodos en color gris o negro) que no pertenecen a $N_{in}(t)$. En otras palabras, $N_{out}(t)$ es el complemento de $N_{in}(t)$ tal que $N_{out}(t) \cap N_{in}(t) = \emptyset$. Para cada $r_i \in M(t)$, el espacio euclidiano se divide mediante el algoritmo de Voronoi [8] el cual requiere dos conjuntos de puntos como entrada:

1. Para cada nodo de red $n_j \in r_i$, ESL coloca m nodos equidistantes a lo largo de un círculo C centrado en las coordenadas del nodo de red n_j (i.e, ubicación real o espejo asociado a "0" o "1", respectivamente) que tiene un radio igual a la distancia estimada entre las coordenadas del nodo n_j y la posición actual del nodo móvil en el tiempo t. Llamamos al conjunto de m nodos como nodos *virtuales*. Vea el ejemplo ilustrado en la Figura 3.3.

2. Para cada nodo de red $n_i \in N_{out}(t)$, ESL coloca m nodos virtuales equidistantes de la siguiente manera:

   a) si el atributo color[n_i] = gris, ESL coloca un círculo C centrado en las posiciones real y espejo, cada uno con un radio igual a la distancia máxima de transmisión (es decir R_c), de tal manera que cada círculo C tiene m nodos virtuales equidistantes.

   b) si el atributo color[n_i] = negro, ESL coloca m nodos virtuales a lo largo de un círculo C centrado sólo en la posición real del nodo n_i que tiene un radio igual a la distancia máxima de transmisión.

Es importante mencionar que la división del espacio euclidiano es una operación atómica, en otras palabras, todo el procedimiento debe ser procesado en el tiempo t.

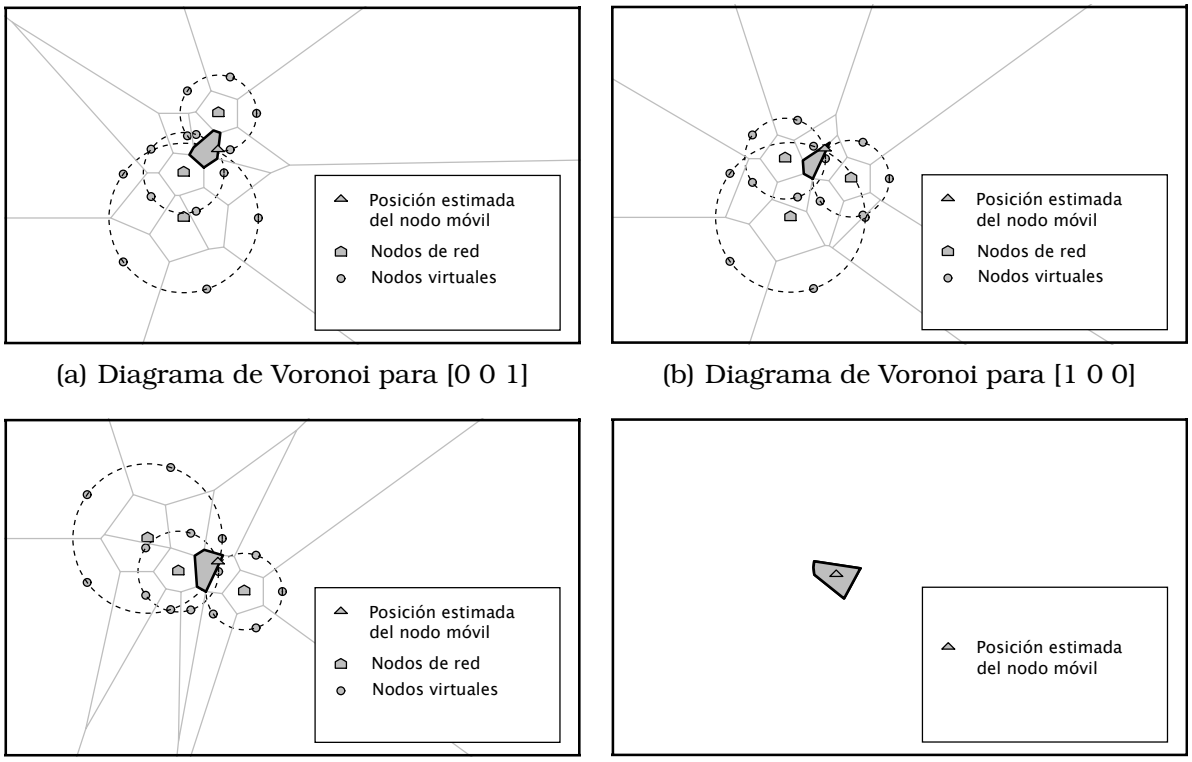
En la siguiente sección, se describe cómo ESL utiliza estos componentes básicos para estimar su auto-ubicación.

3.4. ESL

En esta sección explicamos el funcionamiento del algoritmo ESL utilizando los componentes descritos en el apartado anterior. ESL consta de tres escenarios principales de acuerdo con el número de nodos de red que detectan al nodo móvil. Estos escenarios los llamamos exposición-tres, exposición-dos y exposición-uno.

3.4.1. Exposición-tres

Ahora vamos a explicar cómo ESL funciona cuando el nodo móvil detecta tres o más nodos de red en el tiempo actual t. De acuerdo con la división del espacio euclidiano descrito en la sección 3.3.4, por cada $r_i \in M(t)$, ESL calcula el diagrama de Voronoi asociado. Por ejemplo, considere la matriz $M(t)$ mostrada en la sección 3.3.3 cuyo número de filas es igual a ocho, $|N_{in}(t)| = 3$ y $N_{out}(t)$ está vacío. El diagrama de Voronoi para las filas [0 0 1], [1 0 0] y [1 1 0] se muestran en las Figuras 3.3(a), 3.3(b) y 3.3(c), respectivamente. Observe que cada nodo de red tiene m nodos virtuales (representados como círculos grises). Por cuestiones de simplicidad en la Figura 3.3 sólo se muestran tres filas de un total de ocho. Por cada diagrama de Voronoi asociado



(a) Diagrama de Voronoi para [0 0 1]

(b) Diagrama de Voronoi para [1 0 0]

(c) Diagrama de Voronoi para [1 1 0]

(d) Polígono resultante usando CH

Figura 3.3: ESL para casos cuándo $N_{in}(t) \geq 3$: (a) ESL selecciona el polígono más probable donde se localiza el nodo móvil en el tiempo t para la fila [0 0 1]. De manera similar en (b) y (c), ESL selecciona los polígonos más probables para las filas [1 0 0], [1 1 0] en el tiempo t, respectivamente. Finalmente, ESL usa el algoritmo cierre convexo (CH) para crear un único polígono resultante que representa la posición más probable del nodo móvil en el tiempo t.

a cada $r_i \in M(t)$, ESL utiliza un algoritmo de trilateración [36, 10] para localizar al nodo móvil, el cual regresa un punto p que necesariamente se encuentra dentro de un polígono convexo del diagrama de Voronoi asociado a la fila r_i. A través del algoritmo *point location* [8], ESL selecciona dicho polígono, ver Figuras 3.3(a), 3.3(b) y 3.3(c). Hacemos hincapié en que el algoritmo de trilateración puede ser usado si y sólo si $|N_{in}(t)| \geq 3$. Finalmente, con cada polígono convexo asociado a cada $r_i \in M(t)$, la posición más probable del nodo móvil se obtiene mediante el cálculo de un *centroide*. Para lograr esto, se utiliza el algoritmo de *Convex Hull* [8] [35] (CH) que toma como entrada el centro de cada polígono convexo asociado a cada $r_i \in M(t)$. El resultado de este proceso es un polígono convexo que representa el área espacial media donde la red inalámbrica localiza al nodo móvil. A partir de ahora, nos referimos al centro de este polígono como *punto ancla*.

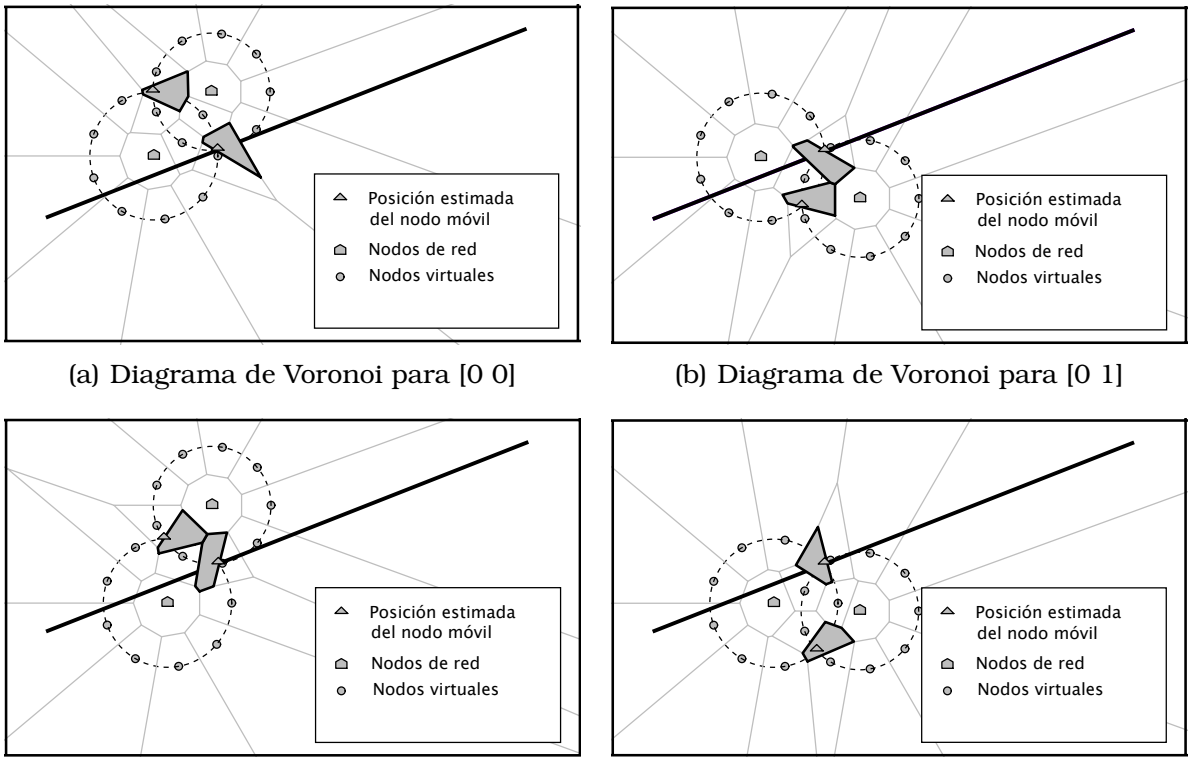
Por ejemplo, en la Figura 3.3(d) se muestra el polígono resultante asociado a la matriz mencionada en el párrafo anterior, donde el triángulo gris representa la posición más probable del nodo móvil (es decir, el punto ancla) visto por la red y el polígono en color gris representa el área espacial media donde la red localiza al nodo móvil. Tenga en cuenta que este procedimiento emula cómo la red localiza el nodo móvil cuando tres o más nodos de red detectan al nodo móvil. Es importante mencionar que la matriz $M(t)$ representa todos los posibles escenarios donde los nodos de red en $N_{in}(t)$ pueden localizarse. Como se mencionó en la sección 3.3.4, todo este procedimiento es una operación atómica que debe ser ejecutada en el tiempo t.

3.4.1.1. Casos especiales

Utilizar el algoritmo de CH requiere al menos tres puntos no colineales como entrada. En ESL, dos casos especiales se consideran antes de aplicar este algoritmo:

Supongamos que uno de los nodos de red $\in N_{in}(t)$ es de color gris, mientras que los demás son de color negro. De acuerdo a la Sección 3.3.3, el número de filas en $M(t)$ es igual a dos. De ello se desprende que tanto el número de polígonos encontrados por el algoritmo *point location* y sus centros asociados son igual a dos. Claramente, el algoritmo CH no puede ser usado en este caso. ESL resuelve esta situación al intersecar los dos polígonos como el área espacial media donde la red inalámbrica detecta al nodo móvil.

Consideremos ahora el caso en que todos los nodos en $N_{in}(t)$ son de color negro y su cardinalidad de $|N_{in}(t)| \geq 3$. La posición más probable del nodo móvil es el centro del único polígono encontrado por el algoritmo *point location* ya que sólo existe una fila en $M(t)$. Esta situación aparece sólo cuando la ambigüedad de la posición de todos los nodos de red en $N_{in}(t)$ fue descartada en el tiempo t. Este escenario es el caso ideal, ya que se conoce la posición real de los nodos de red y ESL puede reproducir la misma posición que genera la red, de otra forma se debe calcular un centroide con todos los posibles escenarios asociados a la matriz $M(t)$.



(a) Diagrama de Voronoi para [0 0]

(b) Diagrama de Voronoi para [0 1]

(c) Diagrama de Voronoi para [1 0]

(d) Diagrama de Voronoi para [1 1]

Figura 3.4: ESL para casos cuando $N_{in}(t) = 2$: ESL selecciona el polígono más probable donde está ubicado el nodo móvil en el tiempo t para las filas [0 0], [0 1], [1 0] y [1 1] (ver polígonos en color gris en las figuras (a), (b), (c) y (d), respectivamente.

3.4.2. Exposición-dos

Ahora vamos a explicar cómo ESL funciona cuando el nodo móvil detecta dos nodos de red en el tiempo t. De acuerdo con la sección 3.3.3, cada vez que el nodo móvil detecta un nodo de red, una matriz $M(t)$ se construye con los nodos de red $\in N_{in}(t)$. Para los casos cuando dos nodos de red son detectados por el nodo móvil, esto implica que $|N_{in}(t)| = 2$. Por ejemplo, considere que los dos nodos de red $\in N_{in}(t)$ son de color gris y que $N_{out}(t) = \emptyset$, la matriz asociada $M(t)$ para este ejemplo es:

$$M(t) = \begin{pmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{pmatrix}$$

El espacio Euclidiano para las filas [0 0], [0 1], [1 0] y [1 1] se muestra en las Figuras 3.3(a), 3.3(b), 3.3(c) y 3.3(d), respectivamente. En estas figuras se puede ver que por cada $r_i \in M(t)$ la posición estimada del nodo móvil tiene dos soluciones geométricas. Estas soluciones se generan mediante la intersección de las dos zonas

de cobertura de cada nodo de red. Denotamos estas soluciones como $\hat{M}_r^i$ y $\hat{M}_m^i$, donde el subíndice r y m indican las posiciones real y espejo del nodo móvil, respectivamente (representados como triángulos grises), y el súper índice representa la i-ésima fila de $M(t)$. Por cada $r_i \in M(t)$ ESL selecciona los polígonos convexos más probables donde el nodo móvil se encuentra (es decir, $\hat{M}_r^i$ y $\hat{M}_m^i$) por medio del algoritmo *point location* [9] (ver la Figura 3.3). También es importante mencionar que las posiciones real y espejo se utilizan sólo para fines descriptivos ya que en este punto la red no tiene manera de distinguir qué solución corresponde a la posición real y cual corresponde a la posición espejo.

Por otro lado, podemos observar que las posiciones $\hat{M}_m^1$ y $\hat{M}_m^4$ de las filas [0 0] y [1 1], son simétricas con respecto a la trayectoria del nodo móvil (ver triángulos en Figura 3.5). También podemos observar en la misma figura que las posiciones $\hat{M}_m^2$ y $\hat{M}_m^3$ asociadas a las filas [0 1] y [1 0] también son simétricas con respecto a la trayectoria del nodo móvil (es decir, $l_1 \rightarrow l_2$). Las posiciones $\hat{M}_r^1$, $\hat{M}_r^2$, $\hat{M}_r^3$ y $\hat{M}_r^4$ coinciden en la posición l_2 denota por $\hat{M}_r^*$ (ver el triángulo en la Figura 3.5). Debido a estas condiciones de simetría, ESL descarta las filas [1 0] y [1 1] de $M(t)$ para los casos donde $|N_{in}(t)| = 2$, ya que desde la perspectiva de la red, las posiciones $\hat{M}_m^4$ y $\hat{M}_m^3$ producen errores de localización similares a las posiciones $\hat{M}_m^1$ y $\hat{M}_m^2$ con respecto a la trayectoria del nodo móvil. Como resultado, esta matriz se reduce a:

$$M(t) = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$$

Ahora, vamos a explicar cómo ESL calcula la posición más probable del nodo móvil para los casos donde $M(t)$ es igual a [[0 0], [0 1]] y $|N_{in}(t)| = 2$. Como se mencionó, para cada $r_i \in M(t)$ existen dos polígonos asociados, uno de ellos siempre se encuentra sobre la trayectoria del nodo móvil, ver Figuras 3.3a y 3.3b, mientras que el otro puede estar situado en el lado izquierdo o el lado derecho con respecto a la trayectoria del nodo móvil, ver Figuras 3.3(a) y 3.3(b), respectivamente. En este momento ESL no puede descartar ninguno de estos polígonos (después utilizaremos el método de MSC descrito en la sección 2.5.4 para reducir la ambigüedad en la posición del nodo móvil) por lo que ESL calcula tres polígonos posibles que representan la posición del nodo móvil en el tiempo t. El primer polígono se construye por la intersección de los polígonos situados sobre la trayectoria del nodo móvil (es decir, los polígonos asociados a las posiciones $\hat{M}_r^1$ y $\hat{M}_r^2$), mientras que los otros dos, son los polígonos asociados a las posiciones $\hat{M}_m^1$ y $\hat{M}_m^2$, respectivamente.

Como se mencionó en el procedimiento de la sección 3.3.4 la técnica de exposición-dos es una operación atómica que debe ser procesada en el tiempo t.

3.4.2.1. Casos especiales para exposición-dos

Supongamos que todos los nodos de red en $N_{in}(t)$ son de color negro y su cardinalidad es $|N_{in}(t)| = 2$, en este caso la matriz asociada $M(t)$ es [0 0]. Como resultado de ello el nodo móvil sólo tiene una posición real y una posición espejo asociadas a las posiciones $\hat{M}_r^1$ y $\hat{M}_m^1$, respectivamente. Este es el caso ideal, ya que ESL estimaría la misma posición real y espejo que la red ya que no existe ambigüedad en la posición de los nodos de red.

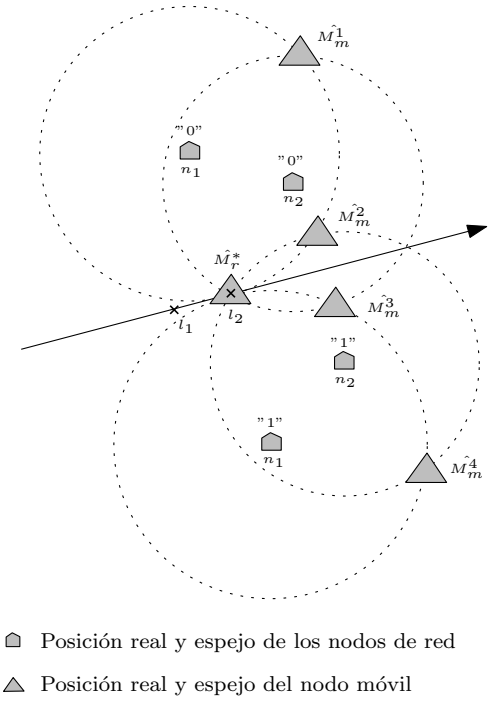


Figura 3.5: Posiciones posibles en exposición-dos

3.4.2.2. Exposición-dos utilizando MSC

El MSC se puede utilizar para descartar la ambigüedad en la posición del nodo móvil en escenarios de exposición-dos de la siguiente manera: el MSC se encuentra inicialmente en la última posición conocida del nodo móvil (por ejemplo, un punto ancla en el tiempo $t - \Delta t$.) Con una radio igual a cero. A continuación, el MSC aumenta su radio a velocidad constante de V_{max} hasta que el nodo móvil transmite un nuevo paquete a la red en el momento actual t como se mencionó en la sección 2.5.4 (ver el ejemplo en la Figura 3.6). ESL puede reducir la ambigüedad en la posición del nodo móvil si y sólo si algunas de las posiciones probables del nodo móvil se encuentran fuera del MSC actual. En el ejemplo de la Figura 3.6, los polígonos asociados a las posiciones $\hat{M}_m^1$ y $\hat{M}_m^2$ se descartan ya que ambos se encuentran fuera del MSC actual. A continuación, un nuevo MSC con radio igual a cero se coloca en el centro del polígono seleccionado por ESL en el tiempo t como un nuevo punto ancla (ver la Figura 3.6 en la posición $\hat{M}_r^*$). Para los casos en que no existan puntos anclas en el tiempo $t - \Delta t$, ESL no puede descartar la ambigüedad de la posición del nodo móvil. Sin embargo, incluso en estos casos, ESL debe colocar un nuevo MSC en el centro de cada polígono como un nuevo conjunto de puntos anclas. Por ejemplo, supongamos que en la Figura 3.6 no existe ningún punto ancla en el tiempo $t - \Delta t$, entonces ESL deberá colocar tres puntos anclas en el centro de cada polígono asociado a las posiciones $\hat{M}_r^*$, $\hat{M}_m^1$ y $\hat{M}_m^2$, respectivamente en el tiempo t.

3.4.3. Exposición-uno

Ahora vamos a explicar cómo ESL funciona cuando sólo un nodo de red detecta al nodo móvil en el tiempo t. De acuerdo con la sección 3.3.3, cada vez que un nodo

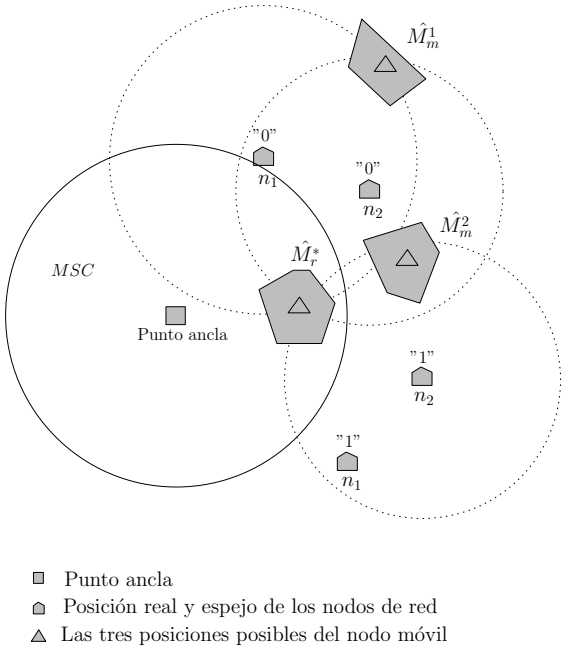


Figura 3.6: Escenario exposición-dos con MSC

móvil detecta un nodo de red, una matriz $M(t)$ se construye mediante el uso de los nodos de red $\in N_{in}(t)$. Para este caso $|N_{in}(t)| = 1$, por lo que se deduce que existen dos posibles escenarios: si la posición espejo del único nodo de red se descarta, implica que $M(t)$ = [0], de lo contrario $M(t)$ = [[0], [1]]. En forma similar a lo explicado en la sección 3.4.2, y debido a las condiciones de simetría, ESL descarta la posición del nodo de red situado al lado izquierdo con respecto a la trayectoria del nodo móvil ya que la posición del lado derecho genera errores similares de localización. Por lo tanto, cuando $|N_{in}(t)| = 1$, la matriz se reduce a $M(t) = [0]$.

Desde la perspectiva de ESL cuando $|N_{in}(t)| = 1$, implica que existe un número infinito de posibles posiciones del nodo móvil a lo largo del perímetro de la zona de cobertura del nodo de red, por lo que el nodo de red no puede descartar ninguna posición del nodo móvil, ver la Figura 3.7. ESL resuelve esta situación colocando nodos virtuales alrededor del nodo de red. La Figura 3.8(a) muestra un diagrama de Voronoi formado por ocho nodos virtuales y un nodo de red. Como podemos ver en esta figura, utilizar el centro de cada polígono (ver cuadrados dentro de los polígonos) como posiciones posibles del nodo móvil puede generar un error considerable ya que algunos de estos polígonos no tienen bordes (ver la Figura 3.8(a)). ESL resuelve esta situación al limitar el tamaño de cada polígono mediante la intersección de los polígonos creados en el tiempo t con los polígonos creados en el tiempo $t - \Delta t$. La Figura 3.8(b) muestra el resultado de este proceso. Observe que se utilizan los nodos virtuales creados en el tiempo $t - \Delta t$ para calcular el diagrama de Voronoi en el tiempo t. Este proceso construye un conjunto de polígonos en el que cada polígono está delimitado por cuatro bordes, como se muestra en la Figura 3.8(b). Como se mencionó en la sección 2.5.3, llamamos a este conjunto de polígonos *dona*.

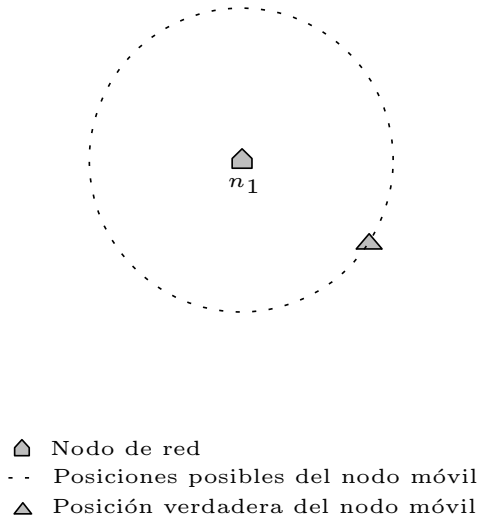
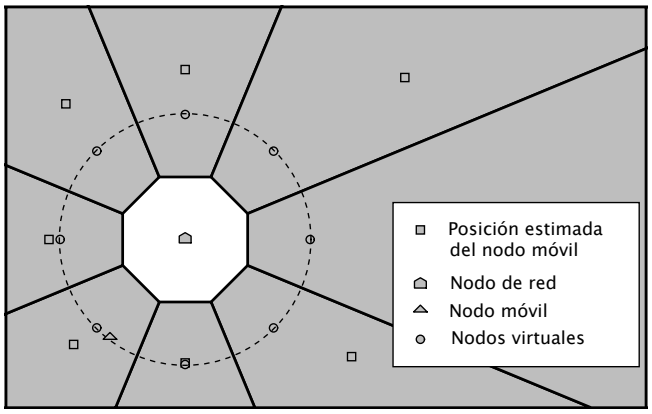
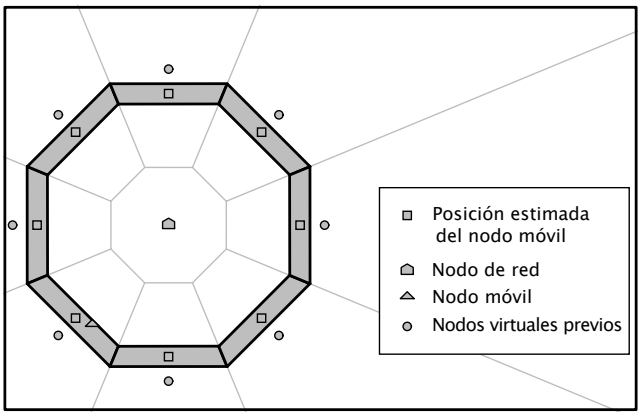


Figura 3.7: Exposición-uno



(a) Diagrama de Voronoi construido por ocho nodos virtuales



(b) Dona construida al intersecar los polígonos del tiempo $t - \Delta t$ con los polígonos del tiempo t

Figura 3.8: Exposición-uno con el uso de nodos virtuales

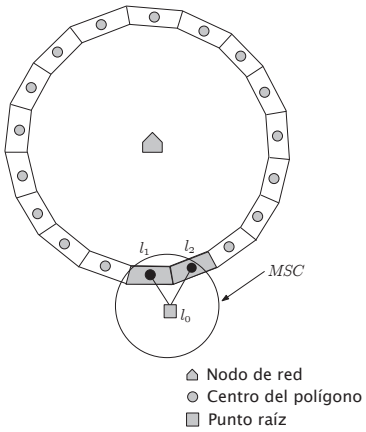
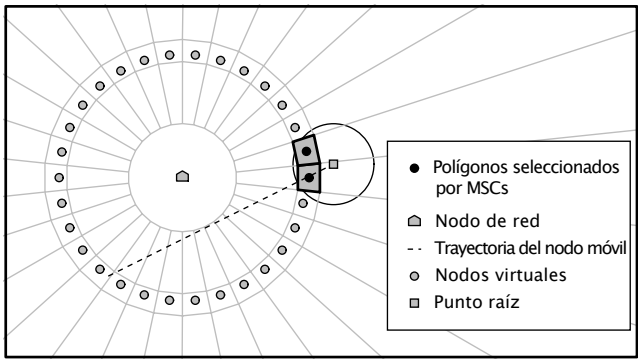


Figura 3.9: Grafo G dentro de exposición-uno

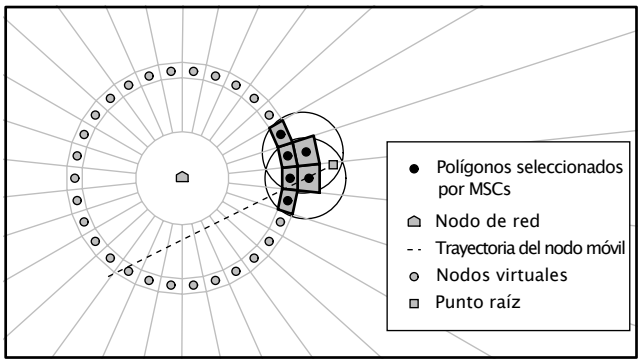
3.4.3.1. Exposición-uno usando MSC

Ahora vamos a explicar cómo ESL usa el MSC para descartar la ambigüedad del nodo móvil para el escenario de exposición-uno. Por ejemplo, considere que ESL conoce la última posición del nodo móvil antes de entrar en el área de exposición-uno (por ejemplo, un punto ancla calculado en el tiempo $t - \Delta t$). Nos referimos a este punto como punto raíz. De acuerdo con la sección 2.5.4, el MSC asociado al punto ancla tiene un radio cero en el tiempo $t - \Delta t$. Después, el radio del MSC aumenta en proporción de V_{max} hasta que el nodo móvil envía un nuevo paquete a la red en el tiempo t. En este momento una nueva dona puede construirse utilizando los nodos virtuales en el tiempo t y los nodos virtuales en el tiempo $t - \Delta t$. En caso de que no existan nodos virtuales en el tiempo $t - \Delta t$, ESL calcula este conjunto de nodos virtuales mediante el uso de un círculo centrado en el nodo de red que tiene un radio igual a la distancia Euclideana entre el nodo de red y el punto ancla. Un ejemplo de este proceso se ilustra en la Figura 3.9, donde el MSC actual se centra en la posición l_0 (es decir, el punto raíz) y la dona actual se centra en el nodo de red. Después ESL calcula la posición probable del nodo móvil mediante la conexión del centro del MSC con el centro de los polígonos situados dentro del MSC. El ejemplo en la Figura 3.9 une dos aristas, la primera arista conecta las posiciones l_0 y l_1, mientras que la segunda arista se forma al conectar las posiciones l_0 y l_2. Luego, un nuevo MSC se coloca en las posiciones l_1 y l_2 con el fin de repetir el proceso hasta que el nodo móvil abandone el escenario de exposición-uno. ESL utiliza este proceso para construir un grafo G que comprende las probables posiciones del nodo móvil a medida que pasa el tiempo, ver la Figura 3.10.

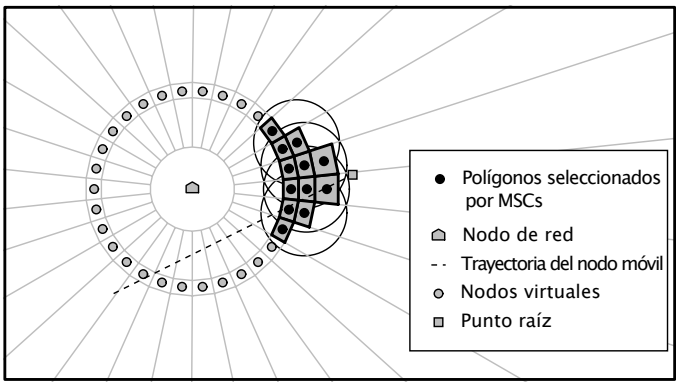
ESL utiliza el algoritmo de backtracking (BT, ver sección 3.4.3.1) para buscar una solución óptima en G. Sin embargo, ESL debe cerrar los posibles caminos en G antes de utilizar el algoritmo de backtracking. La Figura 3.11 ilustra un nodo móvil (que se muestra como $\triangle$) fuera de los límites del alcance de exposición-uno. Las posiciones l_j, l_k, l_m, l_n, l_l son el centro de los MSC actuales, respectivamente. Sólo los MSCs que alcanzan la posición actual del nodo móvil (l_z) se eligen como puntos finales de las trayectorias posibles en G. En el ejemplo de la Figura 3.11, los puntos l_k y l_m se eligen como puntos finales de los caminos posibles en exposición-uno. En caso que no existan puntos anclas antes de entrar al escenario de exposición-uno, ESL no



(a) Dona en el tiempo t



(b) Dona en el tiempo $t + \Delta t$



(c) Dona en el tiempo $t + 2\Delta t$

Figura 3.10: Polígonos seleccionados conforme el tiempo transcurre

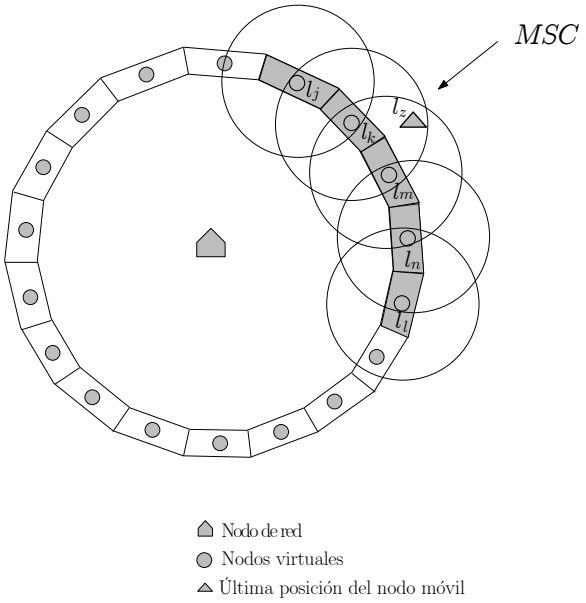


Figura 3.11: Puntos finales en exposición-uno

puede colocar ningún MSC con el fin de reducir la incertidumbre en la posición del nodo móvil. Sin embargo, este caso es poco probable ya que en algún momento ESL puede encontrar puntos anclas que pueden ser usados como puntos raíz para los escenarios de exposición-dos y de exposición-uno. El algoritmo ESL en pseudo código se presenta en el Algoritmo 6.

3.4.4. Limitaciones de ESL

Como se mencionó en la Capítulo 1, el número esperado de nodos de red que cubre cada punto en una red de densidad alta, es igual o mayor a tres con el fin de utilizar un algoritmo de trilateración. Sin embargo, esta suposición rara vez es cierta en las redes reales. Por ejemplo, véase el ejemplo de la Figura 3.12 donde existen regiones cubiertas por tres, dos y un nodo de red. En esta Figura podemos ver que el nodo de red n_1 utiliza sólo los nodos virtuales localizados en el perímetro de la zona de cobertura-1, ya que el nodo de red n_1 puede comunicarse con sus vecinos cercanos y eliminar los nodos virtuales creados en las áreas de intersección entre los nodos vecinos (en este ejemplo n_2 y n_3). Sin embargo, ESL no puede utilizar este procedimiento ya que no es capaz de comunicarse con los nodos de red a fin de ajustar el alcance en exposición-uno. Como resultado, ESL puede generar un error de ubicación mayor con respecto al generado desde la perspectiva de la red.

3.5. Pruebas de rendimiento para ESL

Esta sección analiza el rendimiento de ESL como un método de auto-ubicación que emula cómo la red estima la posición actual de un nodo móvil. Para ello construimos

Algoritmo 6 ESL

Require: Definir m (número de nodos *virtuales* por nodo de red).

if un nodo móvil detecta al menos tres nodos de red: **then**
  Actualizar el árbol T y generar la matriz $M(t)$.
  for cada $r_i \in M(t)$ **do**
    Generar el diagrama de Voronoi y seleccionar el polígono $\mathcal{P}$ donde se localiza el nodo móvil.
  end for
  Generar el polígono resultante $\mathcal{R}$ usando el algoritmo CH.
  Concatenar el centro del polígono $\mathcal{R}$ como punto ancla.
end if
if un nodo móvil detecta dos nodos de red **then**
  Actualizar el árbol T y generar la matriz $M(t)$.
  for cada $r_i \in M(t)$ **do**
    Generar el diagrama de Voronoi y seleccionar los polígonos donde se localiza el nodo móvil.
  end for
  Genera los polígonos asociados a las posiciones $\hat{M}_r^*$, $\hat{M}_m^1$ y $\hat{M}_m^2$.
  if la posición espejo es descartada por el actual MSC **then**
    Conectar el centro del MSC actual al centro de polígono seleccionado.
    Crear un nuevo MSC en el centro del polígono seleccionado con un radio igual a cero.
  else
    Seleccionar los polígonos donde se encuentra el nodo móvil (es decir, los polígonos asociados a las posiciones $\hat{M}_r^*$, $\hat{M}_m^1$ y $\hat{M}_m^2$).
    Conectar el centro del MSC actual con el centro de los polígonos.
    Situar un nuevo MSC en el centro de cada polígono con radio igual a cero.
  end if
  Ir al paso 9 hasta que el nodo móvil salga del área de cobertura-dos.
end if
if un nodo móvil detecta a un único nodo de red **then**
  Actualizar el árbol T y generar la matriz $M(t)$.
  Crear la *dona* actual.
  Conectar el centro del MSC con el centro de los polígonos dentro del MSC.
  Colocar un nuevo MSC con radio igual a cero en cada polígono seleccionado.
  Ir a 25 hasta que el nodo móvil salga del área de cobertura-uno.
end if

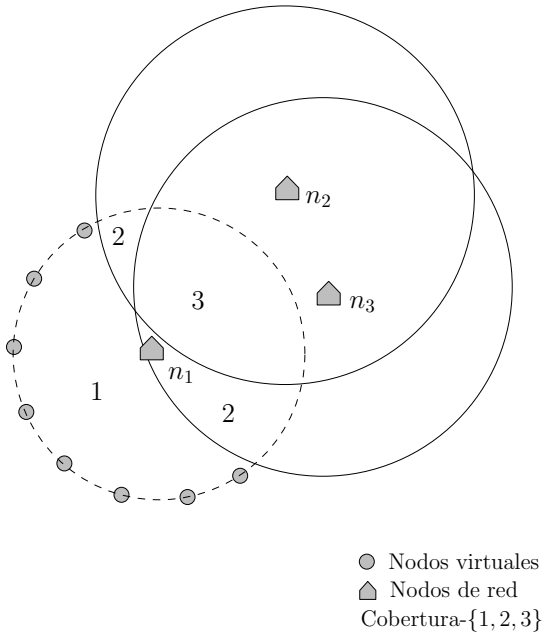


Figura 3.12: Cobertura en exposición-uno

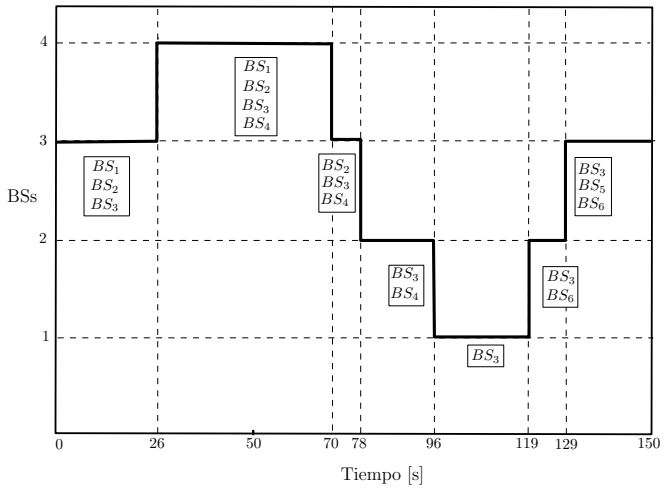
una serie de simulaciones, así como un experimento real con el fin de evaluar la precisión y las limitaciones de ESL como un método de auto-ubicación.

3.5.1. Simulaciones para ESL

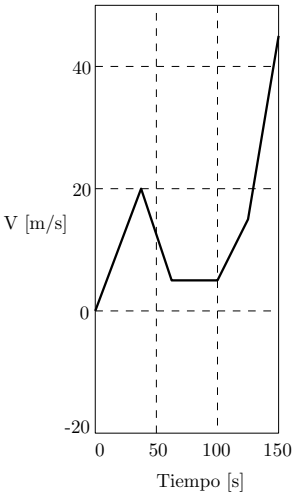
En este apartado se utilizó un simulador escrito en el lenguaje de alto nivel Python para realizar una serie de simulación en el escenario propuesto por [5] y [6]. Utilizamos dos algoritmos llamados Ghost [6] y GPLT [5] para demostrar que ESL es capaz de estimar posiciones similares del nodo móvil calculadas por Ghost y GPLT. Se seleccionaron estos algoritmos, ya que ambos se centran en seguir a un nodo móvil cuando menos de tres nodos de red detectan al nodo móvil.

3.5.1.1. ESL vs Ghost y GPLT

En esta simulación se utilizó el escenario propuesto en [5] donde el número de estaciones base que detecta al nodo móvil cambia a través del tiempo mientras éste atraviesa la red. La Figura 3.13(a) ilustra este arreglo, donde BS_i denota la i-ésima estación base que detecta al nodo móvil en un tiempo específico. En nuestras simulaciones se utilizó el mismo modelo de ruido, las mismas dimensiones de la área vigilada, el mismo rango de transmisión, una trayectoria similar del nodo móvil, el mismo modelo de velocidad (ver la Figura 3.13(b)) propuesto en [5] con el fin de comparar la media del error de posición calculada por ESL con respecto a la media del error de posición del nodo móvil estimada por GPLT y Ghost. En este experimento se utilizaron 30 nodos virtuales por nodo de red ya que este número de nodos virtuales generó el menor error para la mayoría de los escenarios en [6]. La Figura 3.14 muestra el promedio de error de ubicación en función del tiempo para ESL, Ghost y GPLT. Podemos observar en esta figura que Ghost tiene un menor error de ubicación en la mayor parte de la simulación con respecto al generado por GPLT, especialmente en el intervalo de $[78 - 129]$



(a)



(b)

Figura 3.13: Escenario propuesto por GPLT

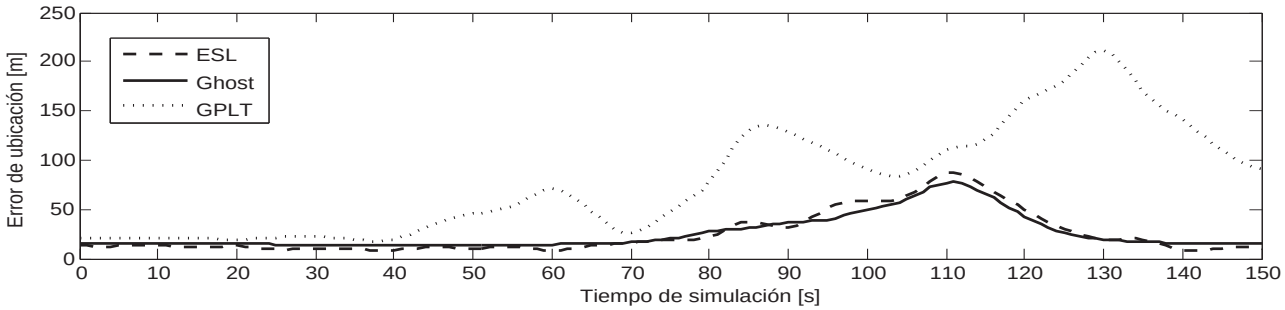
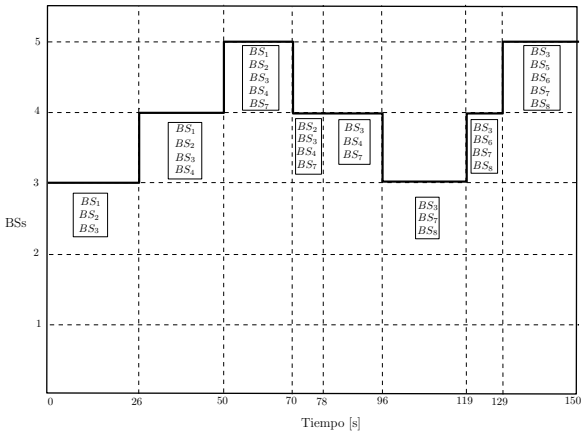
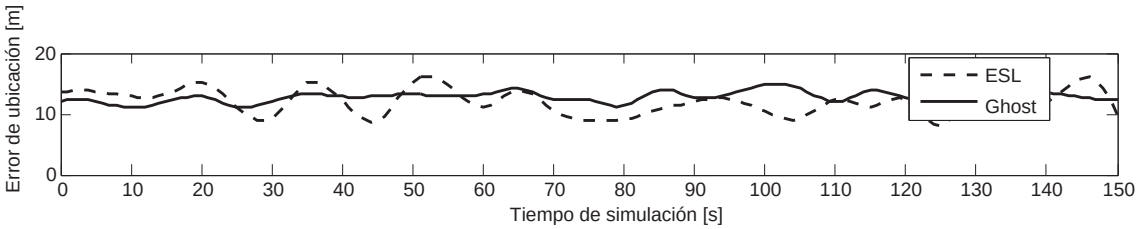


Figura 3.14: ESL vs. Ghost y GPLT



(a)

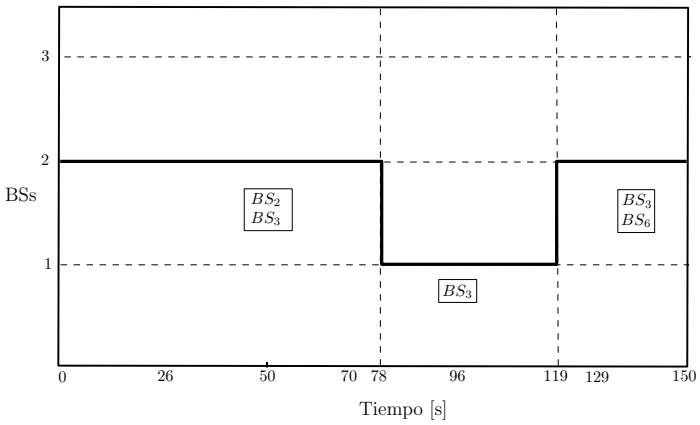


(b)

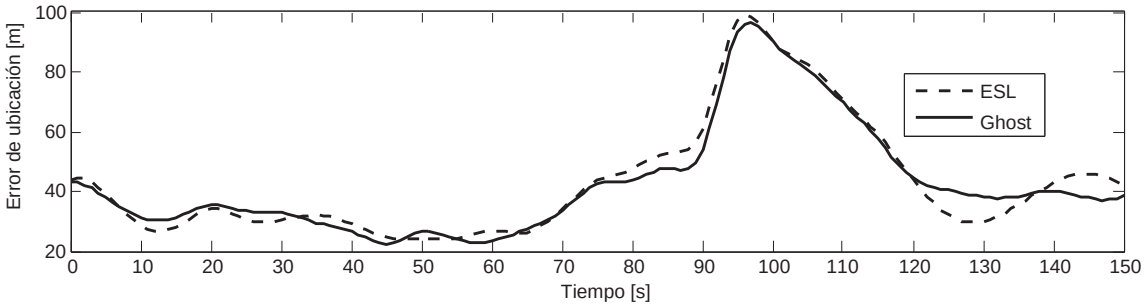
Figura 3.15: Tres o más nodos de red

donde dos y una estación base detectan al nodo móvil. Además, podemos observar en esta figura que ESL y Ghost tienen errores de ubicación similares para la mayoría de los puntos simulados. En otras palabras, ESL es capaz de reproducir el error de ubicación estimado por la red para la mayor parte de los puntos en la simulación. También observamos que cuando el nodo móvil es detectado por tres o más nodos de la red, el error calculado por ESL es similar al error calculado por GPLT y Ghost. A partir de ahora, unicamente comparamos ESL con el algoritmo Ghost [6] ya que éste último generó los errores de localización más bajos al compararlo con GPLT [5], sobre todo cuando sólo dos o un nodo de red detectan al nodo móvil.

También llevamos a cabo dos experimentos utilizando el escenario propuesto en [5]. En el primer experimento, añadimos dos BS (BS_7 y BS_8 ubicados en (1800, 0) y (3000, 4000), respectivamente) con el fin de garantizar que el nodo móvil siempre es detectado por al menos tres nodos de red en toda la simulación. La Figura 3.15(a) muestra este arreglo, mientras la Figura 3.15(b) muestra el error calculado por ESL y Ghost durante la simulación. Podemos ver en esta figura que ESL es capaz de reproducir con exactitud la ubicación estimada por la red para los casos donde existen tres o más nodos de red. En el segundo experimento se quitaron las estaciones base BS_2, BS_4 y BS_5 para garantizar que el nodo de red siempre es detectado por menos de tres nodos de red. La Figura 3.16(a) ilustra este arreglo, mientras que la Figura 3.16(b) muestra el error calculado por ESL y Ghost durante la simulación. También vemos en esta figura que ESL es capaz de reproducir el error de localización calculado por la red para la mayoría de los puntos en la simulación.



(a)



(b)

Figura 3.16: Menos de tres nodos de red

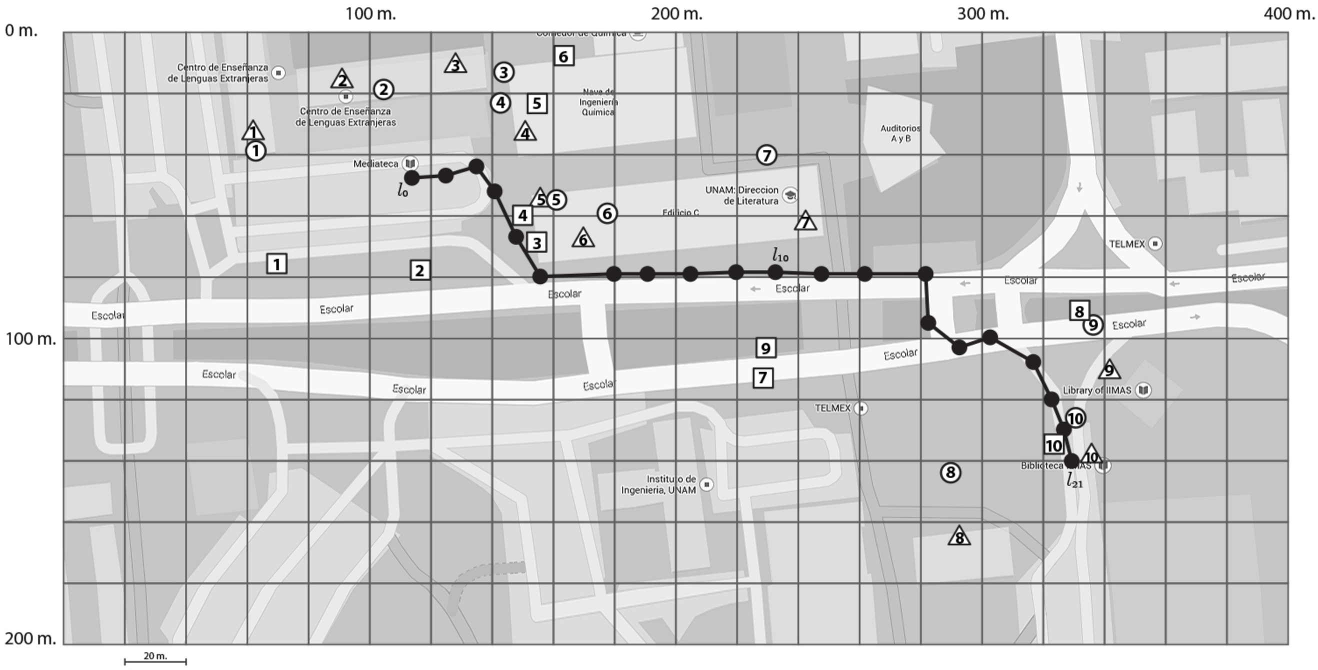


Figura 3.17: Escenario real en nuestro campus universitario

3.5.2. Experimento real

En esta sección se presenta el desempeño de ESL como un método de auto-ubicación en condiciones reales. Para ello creamos una serie de experimentos en nuestro campus universitario en el que el nodo móvil descubrió un conjunto de nodos de red mientras éste atravesaba la red inalámbrica de nuestro campus. Esta región corresponde a un rectángulo que mide 400 x 200 m^2 aproximadamente, el origen de coordenadas de esta región se encuentra en la esquina superior izquierda (véase la Figura 3.17). Este escenario se compone de edificios, árboles, automóviles y otros tipos de vehículos de transporte que interfieren con la línea de vista entre los nodos de red y el nodo móvil. Se midió el alcance de transmisión inalámbrica (es decir, utilizando el método de RSSI sobre 802.11g) en diferentes lugares para varios nodos de red antes de ejecutar el experimento real. Como resultado de estas mediciones obtuvimos la gráfica mostrada en la Figura 3.18, donde Real denota los valores obtenidos experimentalmente. Cada punto se obtuvo al promediar 100 mediciones RSSI en dBm. Se utilizó el modelo de espacio libre [37] para obtener el valor de $n = 2.649$ (*pathloss exponent*) y de esta manera obtener las distancias con base a la medición de potencia obtenida. Fijamos el rango de transmisión máximo en $R_c = 63m$. Representamos a la verdadera posición de los nodos de red como $\triangle$, la ubicación real estimada de los nodos de red por ESL como $\bigcirc$ y la ubicacion espejo de los nodos de red como $\square$, respectivamente (véase la Figura 3.17). La tabla 3.1 muestra estas coordenadas, mientras que la Tabla 3.2 muestra las coordenadas del nodo móvil a medida que atraviesa la red (adquiridas por GPS). Estas coordenadas se representan como puntos negros en la Figura 3.17. En este experimento, también utilizamos 30 nodos virtuales por nodo de red. Utilizamos un teléfono inteligente con Android para implementar ESL cuya interfaz inalámbrica

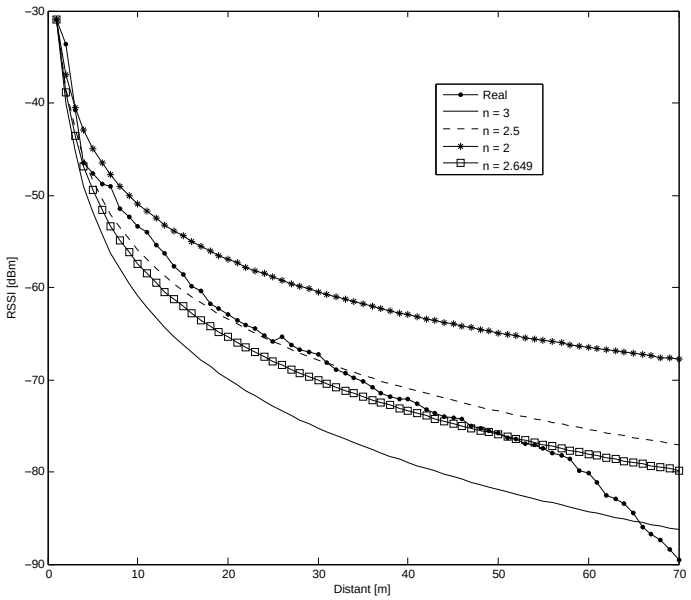


Figura 3.18: Mediciones experimentales de RSSI

se configura en modo monitor para oír el tráfico transmitido por los nodos de red. La Figura 3.19 ilustra el número de nodos de red que detectan al nodo móvil mientras éste atraviesa la red inalámbrica de nuestro campus, donde n_i denota el i-ésimo nodo de red detectado por el nodo móvil en diferentes posiciones. La Figura 3.20(a) muestra el error de ubicación calculado por ESL y Ghost en diferentes posiciones. En esta Figura es fácil ver que el error de auto-ubicación calculado por ESL es similar al error de ubicación calculado por Ghost para la mayor parte de la traza. La Figura 3.20(b) muestra la diferencia en la estimación de la ubicación entre ESL y Ghost, se puede ver que dicha diferencia oscila entre $0m$ y $4m$. Esta diferencia en la estimación de ubicación se debe principalmente a las degradaciones de la propagación creados por múltiples rutas de reflexión, interferencia entre señales y las limitaciones de ESL mencionados en la sección 3.4.4. Sin embargo, consideramos que estas diferencias pueden ser aceptables por la mayoría de las aplicaciones civiles.

3.6. Conclusión

En este trabajo, se presenta un nuevo algoritmo llamado ESL el cual es capaz de reproducir el error de ubicación del nodo móvil estimado por la red sin intervención alguna de ésta. ESL es capaz de crear un mapa de la red sin conocimiento previo de su topología. Este procedimiento requiere que el nodo móvil esté equipado con un GPS para trabajar correctamente. Las simulaciones y experimentos reales muestran que ESL es capaz de reproducir con precisión el error de ubicación calculado por la red en diferentes escenarios donde hay tres o más nodos de red, así como para aquellos casos donde hay dos o un nodo de red detectando al nodo móvil. En general, se muestra que ESL calcula con precisión el error calculado por la red en la mayoría de los escenarios mostrados por las simulaciones, así como en el escenario real. Además, ESL le permite conocer a los nodos móviles su auto-ubicación antes de transmitir con el fin de tomar

Tabla 3.1: Posiciones de los nodos de red

Nodo de red	Verdadera [m]	Real [m]	Espejo [m]
n_1	(62,32)	(63,39)	(70,76)
n_2	(91,15)	(105,19)	(117,78)
n_3	(128,10)	(144,13)	(155,69)
n_4	(151,32)	(143,23)	(150,60)
n_5	(156,54)	(161,55)	(155,23)
n_6	(170,67)	(178,59)	(163,8)
n_7	(242,61)	(229,40)	(229,113)
n_8	(293,164)	(290,144)	(332,91)
n_9	(342,110)	(337,96)	(230,103)
n_{10}	(336,138)	(331,125)	(323,135)

Tabla 3.2: Posición del nodo móvil

Posición del nodo móvil	Coordenadas [m]
l_0	(114,48)
l_1	(125,47)
l_2	(135,44)
l_3	(141,52)
l_4	(148,67)
l_5	(156,80)
l_6	(167,80)
l_7	(180,79)
l_8	(191,79)
l_9	(205,79)
l_{10}	(220,78)
l_{11}	(233,78)
l_{12}	(248,79)
l_{13}	(262,79)
l_{14}	(282,79)
l_{15}	(283,95)
l_{16}	(293,103)
l_{17}	(303,100)
l_{18}	(317,108)
l_{19}	(323,120)
l_{20}	(327,130)
l_{21}	(328,140)

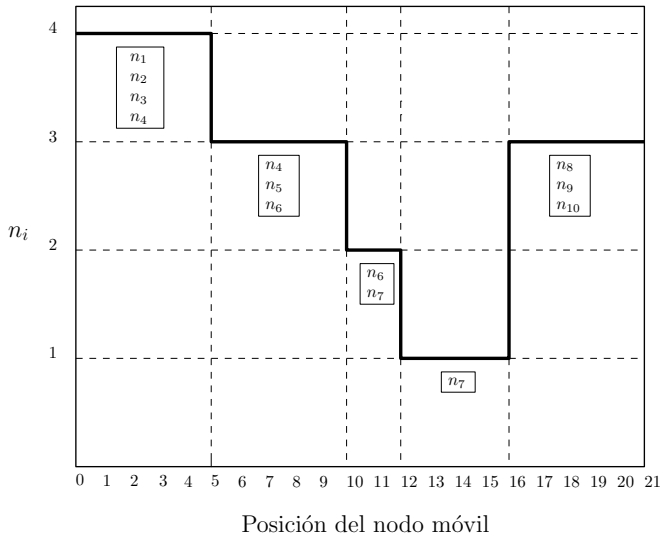
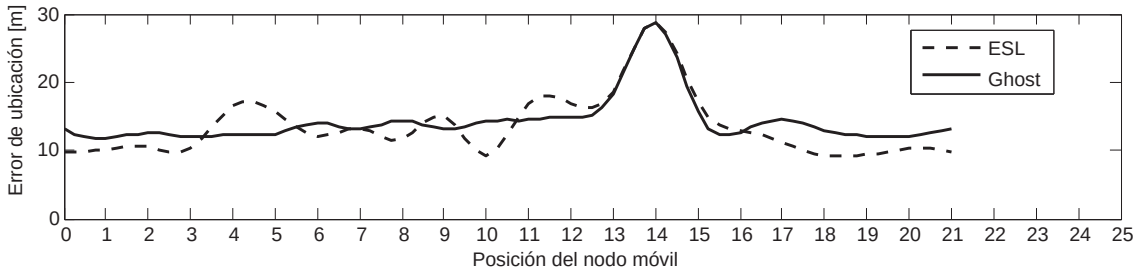
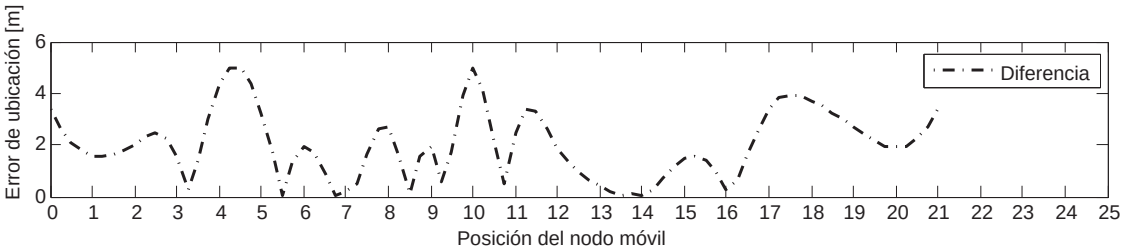


Figura 3.19: Número de nodos de red detectados por el nodo móvil



(a) ESL vs. Ghost



(b) Error entre ESL y GHOST

Figura 3.20: ESL vs. Ghost

contramedidas dependiendo de la aplicación. Hasta donde sabemos, este es el primer trabajo que estudia el problema de la estimación de la auto-ubicación.

Capítulo 4

Conclusiones

En esta sección presentamos las conclusiones generales de este trabajo de tesis, la conclusión de la hipótesis e interpretamos los resultados tanto de las simulaciones como del experimento real. Finalmente se presenta una sección de trabajo futuro.

4.1. Conclusión general

En esta tesis proponemos un algoritmo llamado ESL que permite al nodo móvil conocer su auto-ubicación y de esta manera tomar una contramedida antes de transmitir con el fin de que interactue de mejor forma con su entorno. Por otro lado ESL proporciona al nodo móvil el error de ubicación que incurre la red, valor que la misma red no puede conocer ya que ésta usualmente no conoce la posición real del nodo móvil, lo que implica que no puede estimar dicho error. Las simulaciones y experimentos reales muestran que ESL puede reproducir el error de ubicación estimado por la red para escenarios con tres o más nodos que detectan al nodo móvil, así como aquellos escenarios donde dos o un nodo de red detectan al nodo móvil. Hasta donde sabemos, éste es el primer algoritmo que estudia la estimación de la auto-ubicación. Además, también proponemos un algoritmo llamado Ghost el cual localiza a un nodo móvil en escenarios con tres o más nodos de red, así como aquellos donde menos de tres nodos detectan al nodo móvil. Este algoritmo utiliza una técnica introducida en este trabajo que usa nodos virtuales para generar diagramas dinámicos de Voronoi de la misma topología de red. Esta técnica permite reducir el área media donde la red localiza al nodo móvil y al mismo tiempo permite encontrar rutas posibles cuando un nodo móvil es detectado por menos de tres nodos de red. Tanto las simulaciones como los experimentos reales muestran que Ghost genera un menor error de ubicación comparado con los trabajos encontrados en la literatura.

4.2. Conclusión de la hipótesis

Permítanos retomar la hipótesis presentada en la Sección 1.2.

”Dado un nodo móvil equipado con un receptor GPS y una red de densidad variable capaz de estimar la posición actual del nodo móvil, es posible que el nodo móvil estime su auto-ubicación antes de transmitir a la red.” Suponemos que tanto los nodos de red como el nodo móvil son capaces de medir la potencia de una señal recibida y a su vez estimar la distancia entre su posición geográfica y dicha fuente. Además

suponemos que el radio de cobertura máximo tanto del nodo móvil como de los nodos fijos es de la misma longitud.

En la Sección 3.5.2 se demostró que ESL puede reproducir el error estimado por la red en ambientes simulados así como en escenarios reales. En la Figura 3.20(b) se puede observar que la diferencia de error de ubicación estimado entre Ghost y ESL se encuentra en el intervalo de cero a cinco metros. Consideramos que este error es aceptable para muchas aplicaciones civiles, especialmente porque en este tipo de sistemas intervienen muchas variables que interfieren en la estimación de la distancia entre las terminales móviles. Por lo que consideramos que ESL es un algoritmo capaz de estimar su auto-ubicación antes de transmitir a la red. En cuanto a las suposiciones hechas, consideramos que hoy en día muchos de los sistemas móviles tienen un sistema de localización GPS y que son capaces de estimar la distancia a otra terminal midiendo la potencia de la señal recibida. Por lo que podemos concluir que la hipótesis es cierta.

4.3. Interpretación de los resultados

En [38] los autores proponen un algoritmo llamado DoE, en el cual un conjunto de nodos móviles construyen un mapa local de las posiciones probables de un conjunto de nodos fijos. Cuando estos nodos móviles cruzan su camino, comparten dicho mapa con el fin de procesar esta información y eliminar la ambigüedad en la posición de las terminales fijas. ESL puede utilizar esta técnica con el fin de reducir la ambigüedad en la posición de los nodos fijos y de esta manera reducir su propio error de ubicación. Si ESL comparte su mapa local con otros nodos y elimina la ambigüedad de algunos nodos fijos, implicaría que puede reducir el número de nodos grises en $M(t)$ y por lo tanto el error de auto-ubicación se reduce. El caso ideal se da cuando $M(t)$ sólo tiene nodos negros. En ese momento el diagrama de Voronoi construido por ESL y Ghost son iguales, por lo que el error estimado debe ser igual. Sin embargo, si el nodo móvil no puede intercambiar su mapa local con otros nodos móviles, puede cambiar frecuentemente de dirección y reducir la ambigüedad de la posición de los nodos fijos como se mencionó en la Sección 3.3.1 y de esta manera reducir el número de nodos grises en la matriz $M(t)$. Los dos escenarios mostrados anteriormente pueden reducir el error mostrado en la Figura 3.20(b) y garantizar que el error de ubicación entre Ghost y ESL tienda a cero.

4.4. Trabajo futuro

Como se mencionó en la sección anterior, el error generado por ESL tiene que ver con el número de nodos grises que intervienen en el cálculo de la matriz $M(t)$. Si esta matriz unicamente contiene nodos de color negro, el error de ubicación entre la red y ESL tiende a cero. En el trabajo futuro, se propondrá una técnica en la cual el nodo móvil pueda eliminar la ambigüedad en la posición de los nodos fijos sin intercambiar su mapa local ni cambiar frecuentemente de dirección. Está técnica deberá ser realizada unicamente por el nodo móvil con el fin de reducir problemas de privacidad.

Bibliografía

[1] G. Han, H. Xu, T.Q. Duong, J. Jiang, and T. Hara. Localization algorithms of wireless sensor networks: a survey. *Telecommunication Systems*, 2011.

[2] Tao Shu, Yingying Chen, Jie Yang, and A Williams. Multi-lateral privacy-preserving localization in pervasive environments. In *INFOCOM,IEEE*, pages 2319–2327, April 2014.

[3] V. Phipatanasuphorn and P. Ramanathan. Vulnerability of sensor networks to unauthorized traversal and monitoring. *Computers, IEEE Transactions on*, 53(3):364–369, Mar 2004.

[4] Li Ma, Jiangchuan Liu, Limin Sun, and O.B. Karimi. The trajectory exposure problem in location-aware mobile networking. In *Mobile Adhoc and Sensor Systems (MASS), 2011 IEEE 8th International Conference on*, pages 7–12, 2011.

[5] T. Po-Hsuan, F. Kai-Ten, L. Yu-Chiun, and Chao-Lin C. Wireless location tracking algorithms for environments with insufficient signal sources. *IEEE Transactions on Mobile Computing*, 8(12):1676–1689, 2009.

[6] F. Garcia, J. Gomez, M. Gonzalez, M. Lopez-Guerrero, and V. Rangel. Ghost: Voronoi-based tracking in sparse wireless networks using virtual nodes. *Telecommunication Systems Springer*, 2015.

[7] D.P. Mehta, M.A. Lopez, and Lan Lin. Optimal coverage paths in ad-hoc sensor networks. In *Proc. on Communications ICC IEEE*, volume 1, pages 507–511 vol.1, 2003.

[8] T. M. Chan. Point location in O(log n) time, Voronoi diagrams in O(n log n) time, and other transdichotomous results in computational geometry. In *Proc. of Foundations of Computer Science*, pages 333–344, Oct 2006.

[9] A.Z.B. Haji Talib, M. Chen, and P. Townsend. Three major extensions to Kirkpatrick's point location algorithm. In *Proc. of Computer Graphics International*, pages 112–121, Jun 1996.

[10] N. Kumar. A weighted center of mass based trilateration approach for locating wireless devices in indoor environment. In *Proc. of the Forth ACM International Workshop on Mobility Management & Wireless Access, Terromolinos, Spain, October 2*. ACM, 2006.

[11] A.P. Subramanian, P. Deshpande, J. Gaojgao, and Das S.R. Drive-by localization of roadside WiFi networks. In *Proc. of INFOCOM*, pages 718–725, Apr 2008.

[12] Z. Zhong and T. He. MSP: Multi-sequence positioning of wireless sensor nodes. In *Proc. of International Conference on Embedded Networked Sensor Systems, Sydney, Australia, November 6-9*, pages 15–28, 2007.

[13] T. He, C. Huang, B.M. Blum, J.A. Stankovic, and T. Abdelzaher. Range-free localization schemes in large-scale sensor networks. In *Proc. of Mobi-Com*, 2003.

[14] I. Guvenc, C.T. Abdallah, R. Jordan, and O. Dedeoglu. Enhancements to RSS based indoor tracking systems using Kalman filters. In *Proc. of GSPx and International Signal Processing Conference*, 2003.

[15] E.L. Souza, E.F. Nakamura, and H.A.B. F. de Oliveira. On the performance of target tracking algorithms using actual localization systems for wireless sensor networks. In *Proc. of International Symposium o Modeling Analysis and Simulation of Wireless and Mobile Systems, Tenerife, Canary Islands, Spain, October 26-19*, pages 418–423, 2009.

[16] C. Chao-Lin and F. Kai-Ten. Hybrid location estimation and tracking system for mobile devices. In *Proc. of the Vehicular Technology Conference, IEEE 61st*, volume 4, pages 2648–2652, 2005.

[17] L. Liao, D. Fox, J. Hightower, H. Kautz, and D. Schulz. Voronoi tracking: Location estimation using sparse and noisy sensor data. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2003.

[18] M. Z. A. Bhuiyan, G. Wang, and J. Wu. Polygon-based tracking framework in surveillance wireless sensor networks. In *Proc. of Parallel and Distributed Systems*, pages 174–181, 2009.

[19] N. Bulusu, J. Heidemann, and D. Estrin. GPS-less low-cost outdoor localization for very small devices. *IEEE Personal Communications*, 7(5):28–34, Oct 2000.

[20] D. Niculescu and B. Nath. DV based positioning in ad hoc networks. *Telecommunication Systems*, 22:267–280, 2003.

[21] D. Ma, M.J. Er, B. Wang, and H. B. Lim. Range-free localization based on hop-count quantization in wireless sensor networks. *Telecommunication systems*, pages 1–6, Jan 2009.

[22] W. Alsalih, K. Islam, Y. Nunez-Rodriguez, and H. Xiao. Distributed Voronoi diagram computation in wireless sensor networks. In *Proc. of Annual ACM Symposium on Parallel Algorithms and Architectures*, Jun 2008.

[23] C. Yang, P. Shi, W. Zao, L. Wang, X. Meng, and Wang J. New intersection algorithm of convex polygons based on Voronoi diagrams. In *Proc. of International Symposium on Voronoi Diagrams in Science and Engineering*, pages 224–231, Jul 2006.

[24] A. Yershova and S.M. LaValle. Improving motion-planning algorithms by efficient nearest-neighbor searching. *IEEE Transactions on Robotics*, 23(1):151–157, 2007.

[25] L. Hong-Bo, L. Zhan-Shan, A. Yang, and D. Hui-Ying. On research of optimization strategy for dynamic backtracking. In *Proc. of International Conference on Machine Learning and Cybernetics*, volume 1, pages 266–271, Jul 2009.

[26] R. Dechter. Enhancement schemes for constraint processing: Backjumping, learning, and cutset decomposition. *SciencieDirect, Artificial Intelligence*, 41:273–312, 1990.

[27] F. Benedetto, G. Giunta, A. Toscano, and L. Vegni. Dynamic LOS/NLOS statistical discrimination of wireless mobile channels. In *Proc. of IEEE Vehicular Technology Conference*, pages 3071–3075, Apr 2007.

[28] W. Navidi and T. Camp. Stationary distributions for the random waypoint mobility model. *IEEE Transactions on Mobile Computing*, 3(1):99–108, 2003.

[29] M.L. Damiani and C. Cuijpers. Privacy challenges in third-party location services. In *Mobile Data Management (MDM), 2013 IEEE 14th International Conference on*, volume 2, pages 63–66, June 2013.

[30] K. Vu and R. Zheng. Robust coverage under uncertainty in wireless sensor networks. In *Proc. of INFOCOM 2011 IEEE*, pages 2015–2023, 2011.

[31] M.P. Johnson, D. Sarioz, A. Bar-Noy, T. Brown, D. Verma, and Chai Wah Wu. More is more: The benefits of denser sensor deployment. In *Proc. INFOCOM 2009, IEEE*, pages 379–387, 2009.

[32] Kuei-Ping Shih, Chun-Chih Li, and Yen-Da Chen. An intruder avoidance vulnerable path adjustment protocol for wireless mobile sensor networks. In *Proc. on Pervasive Computing (JCPC)*, pages 125–130, 2009.

[33] S. Meguerdichian, F. Koushanfar, G. Qu, and M. Potkonjak. Exposure in wireless ad-hoc sensor networks. In *Proc. of MobiCom*, pages 139–150, 2001.

[34] W. Zhang and M. Li. Anti-detection: how does a target traverse a sensing field. In *Embedded Software and Systems, 2005. Second International Conference on*, 2005.

[35] M. Sharif, S. Khan, S.J. Khan, and M. Raza. An algorithm to find convex hull based on binary tree. In *Proc. on Multitopic Conference INMIC. IEEE 13th International*, pages 1–6, 2009.

[36] E. Elnahrawy, Li. Xiaoyan, and R.P. Martin. The limits of localization using signal strength: a comparative study. In *Proc. of Sensor and Ad Hoc Communications and Networks, First Annual IEEE Communications Society Conference on*, pages 406–414, Oct 2004.

[37] O. Oguejiofor and A. Adewale. Outdoor localization system using rssi measurement of wireless sensor network. In *Proceedings of International Journal of Innovative Technology and Exploring Engineering*, 2013.

[38] M. Pascoe, J. Gomez, V. Bonilla, M. Lopez-Guerrero, V. Rangel, and E. Rodriguez-Colina. DoE: A mobility-based location method for wireless networks. *IEEE Transactions on Mobile Computing*, 2013.