Casa abierta al tiempo

UNIVERSIDAD AUTÓNOMA METROPOLITANA

POSGRADO EN CIENCIAS Y TECNOLOGÍAS DE LA INFORMACIÓN

Sincronización de Tiempo en Redes Submarinas de Sensores Inalámbricos

Idónea Comunicación de Resultados

para la obtención del grado de

MAESTRO EN CIENCIAS Y TECNOLOGÍAS DE LA INFORMACIÓN

presentada por

Sergio Arturo Elizalde Hernández

Asesor:

Dr. Víctor Manuel Ramos Ramos

22 de noviembre de 2017

Sustentada ante el jurado calificador:

Dr. Enrique Stevens Navarro, UASLP, Presidente

Dra. Graciela Román Alonso, UAM-I, Vocal

Dr. Víctor Manuel Ramos Ramos, UAM-I, Secretario

Agradecimientos

Agradezco a CONACyT por brindarme el apoyo económico que gracias a eso se ha hecho posible la realización de mi proyecto de investigación. A la Universidad Autónoma Metropolitana por darme la oportunidad de continuar con mis estudios profesionales en el Posgrado de Ciencias y Tecnologías de la Información que pertenece al Programa Nacional de Posgrados de Calidad.
Quiero agradecer ampliamente al Dr. Víctor Manuel Ramos Ramos por darme la oportunidad de pertenecer a su equipo de trabajo y por la confianza que ha depositado en mí. Agradezco todos sus consejos académicos y personales que han ayudado a mejorar muchos aspectos para mi formación como investigador y en mi persona.
También quiero agradecer a los profesores que me han guiado con ética y profesionalismo. Su ayuda ha sido indispensable para completar mi formación como investigador. Gracias a mis compañeros de PROSECOM que me han orientado durante este proyecto. Así mismo, también agradezco a mis compañeros de generación y a Katia que me han acompañado en todo el camino.
Por último, quiero extender el agradecimiento a mis padres por apoyarme siempre en mi formación académica. Sus ánimos, motivación, compañía y su confianza que siempre me han demostrado, han sido de gran importancia para concluir este proyecto. Mis hermanos y sobrinos son otro factor importante para mantenerme firme en el proceso de la maestría. Y en general, a mi familia que siempre ha confiado en mí.

Contenido

Lista de Figuras I

Lista de Tablas III

Acrónimos V

Resumen VII

1. Introducción 1

1.1. Objetivo General 2

1.2. Objetivos Particulares 3

1.3. Estructura del documento 3

2. Trabajo Relacionado 5

2.1. TSHL [2] 6

2.2. MU-Sync [3] 6

2.3. D-Sync [4] 8

2.4. JSL [5] 9

2.5. Light-Sync [6] 10

2.6. Mobi-Sync [7] 11

2.7. ROCS [8] 12

2.8. Descripción general 13

3. Protocolos de Sincronización de tiempo 15

3.1. Tipos de protocolos de sincronización 15

3.1.1. Protocolos Proactivos 15

3.1.2. Protocolos Reactivos . . . 16
3.1.3. Protocolos de encaminamiento geográfico . . . 16
3.2. Protocolos MAC para UWSN . . . 16
3.2.1. R-MAC . . . 17
3.2.2. Broadcast . . . 18
3.3. Protocolos de encaminamiento . . . 18
3.3.1. Reenvío Basado en Vectores (VBF) . . . 18
3.3.2. Reenvío basado en vectores de extremo a extremo (HH-VBF) . . . 19

4. Simuladores para UWSN **21**
4.1. OMNet++ . . . 21
4.2. UWSim . . . 23
4.2.1. Configuración del entorno . . . 24
4.2.2. Soporte de múltiples robots . . . 25
4.2.3. Sensores simulados . . . 25
4.2.4. Interfaces de red . . . 25
4.3. Aqua-Sim basado en NS-2.30 . . . 27
4.3.1. Capa de Red . . . 28
4.4. Aqua-Sim basado en NS-3 . . . 30
4.5. Simulador seleccionado . . . 30

5. Evaluación **31**
5.1. Escenarios de simulación . . . 31
5.2. Modelo de movilidad . . . 32

6. Resultados **35**

7. Conclusiones **43**
7.1. Trabajo a futuro . . . 45

Referencias **45**

Bibliografía **47**

Apéndices **51**

A. Parámetros de simulación **53**

B. Descripción de trazas **55**

C. Regresión lineal **57**

Lista de Figuras

3.1. Protocolos de encaminamiento basado en vectores. 20

5.1. Escenario de simulación. 32

6.1. Regresión lineal. 38
6.2. Primera vs Segunda regresión lineal. 39
6.3. Calibración de la regresión lineal. 40
6.4. Primera regresión lineal con 120 nodos. 41
6.5. Segunda regresión lineal con 120 nodos. 41

Lista de Tablas

4.1. Bloques de XML. 26

6.1. Valores para la regresión y modelo lineal. 36
6.2. Aproximación de la regresión lineal. 37
6.3. Calibración para la sincronización de tiempo en UWSN. 39

Acrónimos

UWSN	Redes de Sensores Inalámbricas Submarinas
TCP/IP	Protocolo de Control de Transmisión/Protocolo de Internet
MAC	Control de Acceso al Medio
IMM	Modelo Múltiple Interactivo
SR	Envía – Recibe
RO	Sólo Recibe
WLSE	Estimación de los Mínimos Cuadrados Ponderados
OR	Encaminamiento Oportunista
RTS	Solicitud para Enviar
CTS	Limpiar para Enviar
ARQ	Solicitud de Repetición Automática
ACK	Acuse de Recibido
HIL	Herramienta de Simulación de Bucle
ROV	Vehículo con Control Remoto
OSG	Gráfico de Escena Abierta
ROS	Sistema Operativo de Robot
DOF	Grados de Libertad

URDF	Formato de Descripción de Robot Unificado
FIFO	Primero que Entra Primero que Sale

Resumen

El planeta Tierra se encuentra cubierto por dos terceras partes de agua, este es uno de los principales puntos que hace al medio marino un área interesante de investigación en muchos ámbitos como son: el control remoto en la industria petrolera, cuidados ambientales, detección de catástrofes y alerta temprana, seguridad nacional, así como nuevos descubrimientos de recursos naturales. Con ayuda de las redes de sensores inalámbricas submarinas (UWSN: Underwater Wireless Sensor Networks) se pueden resolver varios problemas como los antes mencionados, ya que las UWSN son muy importantes para conocer de una mejor manera lo que está pasando en este medio. Es por eso, que se tiene un gran interés en desarrollar aplicaciones con este tipo de redes, ya que es un tema al que aún no se le ha indagado mucho para la resolución de problemas.

Este trabajo de investigación se centra en la sincronización de tiempo para las redes submarinas de sensores inalámbricos. En un método de sincronización de sensores (nodos) en una red, se tienen más de dos nodos (B y C) que se quieren comunicar con el nodo destino (A). Es decir, B y C cuentan con un oscilador de frecuencia el cual es utilizado como el reloj de cada nodo. De esta manera, cada uno tiene un reloj que avanza de diferente manera en comparación con cualquier otro. Debido a que ningún reloj oscila de la misma manera, cada trama enviada puede llegar en cualquier instante de tiempo con respecto a A. Así, se pueden generar varias colisiones en la red y los nodos nunca podrán comunicarse. Es por esto por lo que es de gran importancia la sincronización de tiempo para que la comunicación entre los nodos de una red se efectúe con el menor número de problemas.

La sincronización de tiempo en las redes cumple un papel importante, como asignar el tiempo de habla para cada nodo. Cuando los nodos se encuentran sincronizados ya saben cuál es su tiempo de habla asignado. En el momento que se tienen redes densas y todos los nodos quieren transmitir al mismo tiempo a un nodo receptor, lo primero que se debe tener en cuenta

es esperar su turno de habla para que las tramas que se envíen no se confundan con las de algún otro nodo vecino.

Para las redes submarinas, la sincronización de tiempo es de gran importancia, ya que es un problema fundamental bajo el agua debido a su alto retardo de propagación. Existen muchos protocolos de sincronización para redes terrestres, sin embargo, no pueden ser aplicados directamente en las redes submarinas debido a los problemas existentes (alto retardo de propagación, la utilización de la comunicación acústica y la movilidad de los nodos). Con estos problemas, han surgido algunas propuestas para su solución en un ambiente marino, pero el campo aún sigue siendo amplio. Es por ello, que en este trabajo se hace un estudio de protocolos propuestos en redes de sensores submarinos inalámbricos para conocer y entender la función de estos protocolos. Posteriormente, se busca un simulador de redes submarinas, para que de esta manera se obtengan trazas de tiempo necesarias para la sincronización de tiempo. Una vez obtenidas las trazas de tiempo se filtran los datos para contar con la información necesaria para la sincronización.

Para la sincronización de redes se utiliza el número de nodos que se implementan en la red, así como el tiempo que tarda cada nodo en tener una comunicación con el nodo fuente que es quien va a ser el que sincronice a toda la red. Estos datos son importantes para la sincronización debido que, para la regresión lineal necesitamos un valor dependiente y uno independiente, de esta manera el nodo será la variable dependiente del tiempo. Es decir, cada nodo dependerá del tiempo en que se comunica con el nodo fuente, lo cual se obtiene de algunos cálculos estadísticos para determinar la pendiente de la curva, así como también el offset que se tiene en cada escenario.

Capítulo 1

Introducción

En la mayoría de las UWSN se considera la movilidad debido a que la ubicación de cada sensor cambia constantemente. Por este motivo, la sincronización de tiempo se utiliza para determinar su ubicación, y también los mensajes de marca de tiempo se propagarán entre los nodos, con el fin de determinar la proximidad relativa entre ellos. La sincronización se utiliza también para ahorrar energía, esto permite que los nodos puedan dormir durante un tiempo determinado y luego despertar periódicamente para recibir y enviar una señal. Además, el contar con un temporizador común entre los nodos permite determinar la velocidad de un nodo móvil. Es por ello, que la sincronización de tiempo es importante para la comunicación en las redes de sensores, puesto que se pueden ejecutar algoritmos de movilidad, localización, proximidad, entre otras cosas. Por eso es por lo que muchas aplicaciones de redes de sensores requieren sincronización en el reloj local de cada nodo sensor con respecto a un nodo de referencia.

La necesidad de la sincronización es evidente, además de sus múltiples usos como la determinación de la ubicación, la proximidad y la velocidad, es necesaria porque los relojes no son perfectos. Para cada reloj hay intervalos de tiempo distintos en cada transmisión que genere un nodo, por consecuencia, los eventos serán vistos de diferente manera en el nodo receptor.

Las redes de sensores submarinas han causado un gran interés en la investigación para nuevas aplicaciones y conocimiento del medio marino, pero no se puede aplicar un protocolo de sincronización terrestre en el medio marino. Existe un contraste entre estos dos medios: los canales de comunicación terrestres y los acústicos en UWSN, ya que pueden verse afectados seriamente por el medio marino, el ruido, recursos de ancho de banda, potencia limitada y por las duras condiciones ambientales bajo el agua. Por lo tanto, el canal de comunicación bajo

el agua a menudo experimenta atenuación severa de la señal, el efecto de trayectos múltiples, dispersión de frecuencia y potencia limitada, por mencionar algunos problemas [1]. A su vez, el canal de comunicación bajo el agua es uno de los canales inalámbricos más complejos de la naturaleza. En esta investigación, sólo se consideran los problemas de retardos de propagación para poder la regresión lineal de la sincronización.

Ante los problemas antes presentados, se hace esta investigación, en la cual primero se buscan trabajos relacionados a éste, para buscar una alternativa de resolución hacia la sincronización. Posteriormente, se hace una búsqueda de un simulador que tome en cuenta los problemas necesarios para realizar una sincronización de red submarina, como lo son: una simulación en un espacio tridimensional. Esta debe soportar los protocolos de encaminamiento y control de acceso al medio (MAC: Medium Access Control) para UWSN, nodos móviles, entre otros. El simulador ayuda a obtener trazas de tiempo suficientes para poder hacer el análisis estadístico y calcular la regresión lineal.

Para entender de una mejor manera el funcionamiento de la red, es importante mencionar que en la capa de enlace de datos del modelo TCP/IP (Transmission Control Protocol/Internet Protocol), se encuentra la subcapa inferior MAC (Control de Acceso al Medio por sus siglas en inglés), la cual se encarga de ofrecer direccionamiento y da acceso al canal de control lo cual hace posible que varios nodos de una red puedan comunicarse. Existen protocolos en la capa MAC para el intercambio de tramas, como son R-MAC (Reservation Medium Access Control) [12] y BroadcastMAC. Posteriormente se encuentra a la capa de red, la cual se encarga de ofrecer un camino entre el nodo fuente y el nodo destino, por lo tanto, la capa de red es la responsable de encaminar los datagramas con la dirección lógica o IP. En la capa de red hay protocolos geográficos como: VBF (Vector-Based Forwarding) [9] y HH-VBF (Hop-by-Hop Vector-Based Forwarding) [10]. Para la sincronización de nodos submarinos también existen algoritmos como los citados en [2–8], los cuales también serán analizados posteriormente.

1.1. Objetivo General

El interés de la investigación en este tema ha crecido no sólo en un ámbito académico, sino de manera industrial y militar [1]. Para el desarrollo de este trabajo de investigación, se persigue mediante el siguiente objetivo general:

- Evaluar e implementar un protocolo de sincronización para UWSN en un simulador de eventos discretos

1.2. Objetivos Particulares

Para ello, se siguen los siguientes objetivos particulares:

- Revisar el estado del arte de la sincronización de tiempo en UWSN
- Definir y evaluar el escenario de simulación
- Obtener las muestras de tiempo requeridas para la ejecución del algoritmo de sincronización
- Trazar gráficas de desempeño de calibración de la pendiente del tiempo.
- Realizar un reporte sobre las UWSN y sobre los resultados obtenidos.

1.3. Estructura del documento

Esta idónea comunicación de resultados se explica de la siguiente manera:

En el capítulo 2, se presentan algunos algoritmos de sincronización para redes de sensores inalámbricas encontrados durante el estado del arte de manera cronológica. Los primeros algoritmos de sincronización en el medio submarino (de acuerdo con lo encontrado en la literatura) están implementados para redes de sensores anclados. Posteriormente se implementan escenarios en 2D, los datos generados por las simulaciones de estos algoritmos son poco confiables debido a que no se acercan a lo que sucede de manera real. En este capítulo también se menciona al algoritmo de sincronización Mobi-Sync que es el algoritmo de base para el desarrollo de este tema de investigación. En el capítulo 3, se presentan los tipos de protocolos utilizados para diseñar algoritmos de sincronización que se clasifican en: proactivos, reactivos y geográficos. Para el diseño de algoritmos, se necesitan de protocolos MAC y de encaminamiento (capa dos y tres del protocolo TCP/IP), por lo que se realiza un estudio por cada capa con los protocolos utilizados en los algoritmos encontrados. En el capítulo 4, se realiza una comparación entre simuladores para redes que funcionen en una ambiente submarino, además de soportar simulaciones en tres dimensiones. Sólo se encuentra a Aqua-Sim que es un simulador que cuenta con

estas especificaciones el cual está soportado por NS-2.30 y existe una versión beta para NS-3. En el capítulo 5, se evalúa un escenario de simulación con Aqua-Sim basado en NS-2.30 utilizando un protocolo geográfico en capa de encaminamiento y dos protocolos MAC (Broadcast y R-MAC) para hacer una comparación en su funcionamiento. También, se busca el modelo de movilidad y la manera de manipularlo dentro del simulador. En el capítulo 6, se describen los resultados obtenidos en la investigación, en donde se puede apreciar que el protocolo Broadcast tiene un mejor funcionamiento para redes densas y R-MAC para redes no densas. Por último, en el capítulo 7 se presentan las conclusiones de esta investigación y se describe el trabajo a futuro.

Capítulo 2

Trabajo Relacionado

En [18], se hace un estudio de las capas de red utilizando TCP/IP, sin embargo, para esta investigación nos enfocaremos principalmente en la capa de enlace de datos y en la capa de red. En la capa de enlace de datos es la responsable del intercambio de tramas entre los nodos que forman la red, permitiendo una comunicación entre las capas superiores. Dentro de esta capa hay dos subcapas inferiores: subcapa de control de enlace lógico (LLC: Logical Link Control) y la subcapa de control de acceso al medio (MAC). Los protocolos MAC realizan operaciones propias para las redes inalámbricas, la capa MAC es encargada de gestionar y mantener las comunicaciones entre puntos de acceso a adaptadores de red. La capa MAC coordina el acceso a un canal de radio compartido y utilizar su capa física para detectar la señal portadora, también debe de ser capaz de detectar la señal de transmisión y recepción de tramas. Sin un protocolo adecuado en la subcapa MAC la comunicación con las capas superiores puede ser malo y esto degrada el rendimiento de la red. En esta capa también se enfrentan algunos otros problemas tales como la eficiencia energética, la escalabilidad y latencia.

La capa de red, es la capa encargada de seleccionar los caminos más convenientes con el fin de entregar los datos al nodo destino. Los protocolos de encaminamiento en las redes terrestres se dividen en dos tipos: los protocolos proactivos y los reactivos.

Actualmente existen diversos algoritmos de sincronización utilizando diferentes tipos de protocolos MAC y de encaminamiento, sin embargo, se han elegido sólo algunos algoritmos de sincronización para conocer la forma en que obtienen las tranzas de tiempo para el algoritmo de sincronización.

2.1. TSHL [2]

Este protocolo consta de dos fases, la primera fase realiza el sesgo de inclinación del tiempo de cada nodo para que sea sincronizado. El cálculo del sesgo se realiza sin el conocimiento del retardo de propagación de la señal transmitida por el nodo fuente y mediante una regresión lineal sobre varios valores de tiempo de su nodo principal. Posteriormente, los nodos sincronizados pueden operar con diferente sesgo en sus relojes ya que todos los nodos comparten un patrón de frecuencia y son capaces de mantener un temporizador relativo para eventos adicionales. Para esto, se hace el supuesto de que la propagación es constante durante el intercambio de mensajes, para contar con un nodo guía que sincronice a los demás. Así, los nodos tendrán que comunicarse con una referencia de tiempo externa, como un receptor GPS o una boya de superficie.

En la Fase II del algoritmo, se realiza un intercambio de mensajes bidireccional para compensar el sesgo del reloj. Cuando ambas fases se completan se obtiene un mapeo del reloj local, pero el reloj no es exacto con la base de tiempo de referencia. Además de que los relojes de los nodos sólo son estables a cortos plazos. Así, la frecuencia de reloj y el sesgo de inclinación deben ser de un periodo corto de tiempo.

Es por eso por lo que TSHL es un protocolo poco confiable, ya que no contempla movilidad de nodos, tampoco el retardo por el efecto Doppler y su simulación es realizada en un plano bidimensional.

2.2. MU-Sync [3]

Este algoritmo está diseñado para minimizar la desviación de tiempo entre nodos, estimando y compensando tanto el desvío como el desplazamiento por medio de dos fases. En la primera fase, el sesgo y el offset de tiempo se obtienen aplicando dos veces la regresión lineal sobre un conjunto de balizas de referencia. La primera regresión lineal permite que el nodo principal extraiga el retardo de propagación que encuentre cada paquete de referencia. Después de ajustar las temporizaciones de los paquetes recibidos con sus respectivos retardos de propagación, se realiza una segunda regresión lineal sobre este nuevo conjunto de puntos, a partir de los cuales se puede estimar el sesgo y el desplazamiento de cada nodo.

En contraste con TSHL, en el que cada nodo calcula su propio sesgo y desplazamiento de tiempo, el nodo de referencia de MU-Sync asume la responsabilidad de iniciar el proceso de sincronización y calcular el sesgo y desplazamiento de tiempo de sus vecinos.

Este algoritmo supone que los nodos sensores se mueven aleatoriamente dentro de un área de 1000 x 1000 m^2, por lo tanto, no se considera un espacio tridimensional. También considera que la velocidad del sonido es constante (1500 m/s), y no hay variación de la inclinación debido a un cambio en el ambiente. Por otro lado, los errores no determinísticos encontrados durante el intercambio de mensajes se modelan usando la distribución gaussiana. Los parámetros de la simulación son los siguientes:

1. La velocidad máxima ($V_{\max}$) de movilidad es de 2 m/s
2. La inclinación del reloj se establece a 40 ppm
3. La compensación del reloj es de 10 ppm
4. Para la regresión lineal se utilizaron 25 guías
5. El tiempo en que se tarda un nodo común en reexpedir el acuse de recibido (ACK) desde que recibe el paquete es de 0 segundos
6. El intervalo de tiempo entre dos paquetes sucesivos es de 5 segundos
7. Los nodos tienen un rango de velocidad de $[0, V_{\max}]$ con un intervalo de tiempo de 600 segundos
8. Los ángulos de movilidad son de $[-45°, 45°]$
9. La granularidad del reloj es de 1 μs

De acuerdo con MU-Sync, se realizaron 1000 simulaciones para obtener datos confiables para la sincronización. Pero observamos que los parámetros de simulación aún se alejan de los datos reales con los que se puede trabajar para una sincronización más apegada a la realidad en un medio marino.

2.3. D-Sync [4]

D-Sync sincroniza los nodos con sólo uno de ellos dentro del rango de transmisión de cada nodo, este nodo es llamado nodo faro o guía. El nodo esencialmente toma el papel de líder o maestro y es responsable de estimar el desplazamiento del reloj y sesgo para todos los nodos que pueden comunicarse con él. El nodo guía puede ser un súper nodo con mayor energía o también puede ser elegido periódicamente entre los nodos de la red. Una vez que los demás nodos se sincronizan con el nodo guía, también, se sincronizan entre sí y se logra la sincronización de la red.

El protocolo de D-Sync se puede describir en cuatros pasos para lograr su sincronización, como se describe a continuación:

1. El nodo guía inicia el proceso de sincronización transmitiendo un mensaje de petición y almacena el tiempo de envío en el mensaje en un tiempo t_1

2. Algún nodo común (nodo B) registra el tiempo de recepción del mensaje según su hora local (t_2). El mensaje enviado por el nodo guía puede ser utilizado por el nodo B para estimar su velocidad relativa basada en el desplazamiento estimado del efecto Doppler

3. El nodo B espera un tiempo aleatorio (T_{regreso}), antes de responder al nodo guía. Después, el nodo B transmite un mensaje de respuesta al nodo guía en el tiempo t_3

4. Posteriormente, el nodo guía estima su velocidad relativa al nodo B del mensaje de respuesta y también registra el tiempo de recepción de este mensaje (t_4)

Se realiza una simulación con D-Sync en un espacio bidimensional de un área de 1000 x 1000 [m^2], donde cada nodo se mueve en una trayectoria curva dentro del área con una velocidad relativa máxima de 2 [m/s] y una aceleración relativa máxima de 0.1 [m/s^2]. D-Sync establece la velocidad del sonido en 1500 [m/s] y esta permanece constante durante toda la sincronización. Se considera al reloj del nodo guía como un reloj preciso, mientras que el resto de los nodos tienen una inclinación de 80 ppm y un desplazamiento de 0.00006. La capa MAC utilizada por esta red está basada en una contención con ranura para obtener acceso al canal. Al igual que MU-Sync, soporta simulaciones en dos dimensiones, lo cual hace al algoritmo poco confiable para un escenario real.

2.4. JSL [5]

Este algoritmo consta de cuatro fases principales: (I) intercambio de mensajes, (II) sincronización, (III) localización y (IV) iteración.

1. La Fase I inicia con un nodo sensor ordinario que es el que obtiene el tiempo de referencia y la información de la ubicación de los nodos vecinos.

2. En la Fase II, el proceso de sincronización se lleva a cabo por los nodos ordinarios sobre la información obtenida en la Fase I. La Fase II consta de cuatro pasos.

   a) La posición aproximada de un nodo se calcula utilizando el método de diferencia de tiempo de llegadas (TDOA: Time Difference Of Arrivals).

   b) En el efecto de la estratificación hay compensación de tiempo para poder calcular el retardo de propagación.

   c) JSL realiza la regresión lineal para sincronizar el nodo sensor ordinario.

   d) Actualización de los retardos de propagación correspondientes.

3. Durante la Fase III, se lleva a cabo el proceso de localización basado en los retardos de propagación estimados en la Fase II. Además, se utiliza un algoritmo de seguimiento del modelo múltiple interactivo (IMM) para mejorar la precisión de la localización.

4. Por último, en la Fase IV hay un proceso de iteración. La posición estimada en la Fase III actúa como entrada a la Fase II para reemplazar la posición aproximada. A continuación, la fase II y fase III se repiten hasta tener las ubicaciones y la compensación del sesgo del reloj.

Las simulaciones de JSL son realizadas con una herramienta matemática (Matlab) considerando 20 sensores submarinos en una distribución dentro de una región de 200 x 200 x 1000 [m^3]. Dos de estos 20 nodos sensores (cualquiera) pueden comunicarse entre sí mientras no tengan establecida una comunicación con otro nodo. JSL sólo contempla un nodo de referencia, el cual es el encargado de sincronizar a los demás nodos y considera que los patrones de movilidad son uniformes, con maniobra de giro coordinado. Por lo tanto, es necesario que los nodos utilicen

un rastreador IMM con los dos modos anteriores. Estos se describen por el filtro de Kalman y el filtro de Kalman extendido, respectivamente.

Este trabajo sólo se enfoca en la Fase II de este algoritmo, que se refiere a la sincronización de nodos, la cual consta de tres pasos principales:

1. Estimación de retardo de propagación
2. Regresión Lineal
3. Actualización de retardo de propagación

2.5. Light-Sync [6]

Este algoritmo de sincronización combina dos paradigmas de sincronización, envía – recibe (SR) y sólo recibe (RO), para estimar el desplazamiento de reloj y desplazamiento de cualquier nodo con respecto al nodo de referencia. SR y RO se utilizan en dos fases distintas del algoritmo propuesto. En la Fase 1, el paradigma SR estima parámetros de reloj para después ser explotados durante la Fase 2 a través del uso del paradigma RO.

En el paradigma SR, dos nodos están involucrados activamente en el proceso de intercambio de mensajes. Se supone que el nodo A tiene por objetivo la sincronización del nodo B. Para ello, el nodo B inicia el proceso de cambio enviando un mensaje de solicitud al nodo A, que contiene su marca de tiempo de transmisión. Después de que el nodo A ha recibido el mensaje, transcurre un lapso predefinido, posteriormente el nodo A responde al nodo B con un mensaje que lleva las marcas de tiempo de recepción y transmisión.

A diferencia de SR, en el paradigma de RO al menos tres nodos están involucrados. Específicamente: el nodo C, ya que se supone que está en el rango de cobertura de ambos nodos, de manera que recibe y decodifica los mensajes transmitidos por los dos nodos A y B. El nodo explota estos mensajes con el fin de estimar sus parámetros de reloj y, por lo tanto, sincronizarse.

Light-Sync deduce la inclinación y el desplazamiento estimado para cualquier nodo con respecto al nodo de referencia con las siguientes suposiciones: los nodos están atados al fondo del mar y los retardos de propagación son estadísticamente distribuidos de acuerdo con una variable aleatoria gaussiana, una varianza desconocida y una media conocida que depende de la distancia del nodo y la velocidad del sonido.

Light-Sync se compone de dos fases: en la Fase 1, el objetivo es permitir que un nodo en el fondo marino se sincronice con el nodo de referencia estimando su reloj de desplazamiento y determinando el intervalo de tiempo con respecto a la boya de superficie utilizando el paradigma de SR. En la Fase 2, los resultados de la fase anterior se utilizan para permitir que todos los nodos restantes en el fondo marino se sincronicen con el nodo de referencia.

2.6. Mobi-Sync [7]

Mobi-Sync consta de tres fases importantes para la sincronización.

1. En la Fase I, se obtiene la estimación del retardo de propagación. Para la estimación se toman en cuenta dos pasos, uno es el intercambio de mensajes y otro paso es el cálculo de retardo.

   a) En la etapa de intercambio de mensajes, un nodo ordinario lanza la sincronización de tiempo mediante la difusión de la solicitud de un mensaje. Al recibir el mensaje de solicitud, cada súper nodo vecino envía dos mensajes de respuesta. Estos mensajes contienen el vector de velocidad registrada por el súper nodo y la estampa de tiempo de la capa MAC.

   b) En la etapa de cálculo de retardo, se hace una suposición razonable, el nodo ordinario calcula automáticamente su vector de velocidad utilizando las correlaciones espaciales contenidas en los vectores de velocidad de los súper nodos vecinos. El nodo ordinario continúa transmitiendo mensajes de solicitud hasta que se obtiene la cantidad de puntos suficientes para realizar la regresión lineal.

2. En la Fase II, el nodo ordinario ejecuta la primera ronda de regresión lineal con un conjunto de marcas de tiempo recibida de los súper nodos y los retardos de propagación correspondientes. Esto proporciona una estimación de la inclinación proyectada del reloj y del offset. Esta regresión emplea un procedimiento avanzado en la estimación de los mínimos cuadrados ponderados (WLSE) para reducir el impacto de la suposición en la Fase I.

3. En la Fase III, para mejorar aún más la precisión de la sincronización, el nodo común

actualiza ciertos parámetros iniciales, tales como la inclinación inicial y la distancia inicial, y vuelve a calcular el retardo y reformula la regresión lineal para obtener el desplazamiento del reloj final y del offset.

En Mobi-Sync [7], se muestra la simulación de 10 súper nodos y 30 nodos ordinarios, donde el nodo ordinario sólo puede enviar información y los súper nodos son los encargados de llevar esta información a las boyas de superficie. Tanto los súper nodos y los nodos ordinarios son desplegados al azar en una región de 100 x 100 x 100 [m^3]. Se escoge un nodo ordinario al azar como el nodo que se desea sincronizar con respecto a un súper nodo vecino. Podemos decir que Mobi-Sync es un buen algoritmo que estima con gran precisión los largos retardos de propagación dinámicos, pero como está basado en la correlación espacial, sólo funciona para redes densas.

2.7. ROCS [8]

Este algoritmo de sincronización de reloj robusto y oportunista (por sus siglas en ingles Robust, Opportunistic Clock Synchronization) calcula el desplazamiento entre dos relojes de módems (nodos). En un principio, cada nodo contiene un reloj con una deriva casi constante. Cada nodo es capaz de registrar el tiempo exacto de enviar y recibir mensajes. Además, cada nodo determina con precisión la velocidad relativa al nodo transmisor, por lo general a través del desplazamiento Doppler de los paquetes entrantes. Para realizar la sincronización, este algoritmo consta de tres fases importantes, que son:

1. Transmisión de estampas de tiempo
2. Asociación de las estampas de tiempo
3. Regresión lineal

Cuando un nodo transmite o recibe un paquete del nodo fuente, se registra la marca de tiempo correspondiente, una para la transmisión y otra para la recepción. Cada nodo codifica su propia dirección y las marcas de tiempo previamente registradas en el mensaje a enviar. En cada marca de tiempo de recepción, se incluye la dirección de origen correspondiente. Se utiliza un parámetro de tamaño codificado para limitar el número de bytes disponibles de

la codificación de las marcas de tiempo. Las marcas de tiempo de transmisión se codifican, y el espacio restante se llena con la mayor cantidad de marcas de tiempo de recepción. La codificación relativa reduce aún más la huella de la comunicación de la codificación de las marcas de tiempo de transmisión. La propiedad de la marca de lapso establece la diferencia máxima entre la primera y la última marca de tiempo más reciente.

A diferencia de otros trabajos, esta investigación fue desarrollada desde la creación del módem para hacer pruebas (en Toscana, Italia), la región donde se realizaron los experimentos fue dentro de 7 x 7 [m^2] con una profundidad que va desde 40 a 60 [m] (noreste y suroeste respectivamente).

2.8. Descripción general

De acuerdo con el trabajo relacionado, se puede decir que en ningún algoritmo de sincronización han implementado estudios con un simulador basado en redes submarinas. En los algoritmos investigados descritos en las secciones anteriores, se observa que la mayoría son simulados con de herramientas matemáticas (Matlab) o, incluso, de manera experimental. Pero lo que no pudo ser hallado, fue el uso de algún tipo de simulador para UWSN.

Capítulo 3

Protocolos de Sincronización de tiempo

3.1. Tipos de protocolos de sincronización

Existen varios protocolos de sincronización en redes ad-hoc y de redes de sensores inalámbricos que se han desarrollado en los últimos años. Sin embargo, estas investigaciones no se pueden aplicar directamente al medio acuático debido a las diferencias que hay entre el agua y el aire (ancho de banda, potencia limitada en el agua, y las condiciones ambientales bajo el agua). Por esta razón, existen desventajas con respecto a la aplicación de las redes terrestres en un medio marino.

Para entender de una mejor manera el medio marino, primero se hace un estudio de los protocolos de encaminamiento, que se dividen en tres categorías [17]: los protocolos proactivos, reactivos y geográficos.

3.1.1. Protocolos Proactivos

Los protocolos proactivos intentan reducir la suma de los retardos generados en cada mensaje, cuando se han descubierto las tablas de encaminamiento, se guarda la información y se actualizan las tablas en todo momento. Este tipo de protocolos provoca una sobrecarga de señalización para establecer rutas una sola vez y cada vez que se modifica la topología de red, debido a la movilidad y la información, tiene que propagarse a todos los nodos de la red. De esta manera, cada nodo es capaz de establecer una ruta hacia cualquier nodo de la red. Por esta razón, los protocolos proactivos no son una buena opción para redes submarinas.

3.1.2. Protocolos Reactivos

Un nodo cualquiera inicia el proceso de descubrimiento de ruta (sólo en el caso de requerirla). Cuando se establece la ruta a seguir, esta se mantiene hasta que ya no se desea. Estos protocolos son adecuados cuando hay un ambiente dinámico, pero exhiben una mayor latencia, además de requerir inundación iniciada por el nodo fuente para el descubrimiento de nuevas rutas. Por lo tanto, los protocolos proactivos como los reactivos mantienen una sobrecarga de señalización excesiva debido a su dependencia de las inundaciones. Los protocolos reactivos se consideran inadecuados para las redes submarinas debido a su alta latencia causada al formar las rutas. Considerando que las UWSN tienen un problema grave con altas latencias, los protocolos reactivos pueden aumentar más la latencia en las redes submarinas. La mayoría de los protocolos reactivos están basados en enlaces simétricos (misma velocidad de transmisión como de la recepción), pero estos, no son adecuados para el medio ambiente submarino.

3.1.3. Protocolos de encaminamiento geográfico

Este tipo de protocolos establecen sus rutas de fuente-destino aprovechando la información de su ubicación. Es decir, cada nodo selecciona su próximo salto con base en la posición de sus vecinos y del nodo destino. La localización generalmente requiere una estricta sincronización entre los nodos, que es difícil de conseguir bajo el agua, esto se debe al retardo de propagación variable.

3.2. Protocolos MAC para UWSN

La subcapa MAC de la capa de enlace de datos y es la encargada de controlar el acceso al medio y la responsable de transmitir los paquetes. La fragmentación de paquetes, tasa de transmisión, control de flujo, control de potencia y funciones relacionadas con la gestión de baterías, son algunas de sus funciones de la capa de enlace de datos.

Para la sincronización de UWSN, un protocolo geográfico puede ser el adecuado. En este trabajo se realiza una investigación de los protocolos de sincronización, donde se encuentran protocolos basados en el encaminamiento oportunista (OR) [11]. Los protocolos OR se basan en un paradigma prometedor que selecciona al siguiente nodo pasarela. El nodo pasarela es

el encargado de recibir el paquete transmitido por un nodo fuente y reexpedirlo a otro nodo hasta que la información llegue a su nodo destino. Así, cada candidato que recibe el paquete lo puede continuar enviando. Se utiliza un nodo de retransmisión dinámica para volver a enviar el paquete, la fiabilidad de transmisión y el rendimiento de la red aumentan.

Sin embargo, también existen otros protocolos que no son geográficos, como R-MAC [12] y Broadcast [16].

3.2.1. R-MAC

El protocolo R-MAC se centra en evitar las colisiones de las tramas en las UWSN, además de ser un protocolo MAC basado en reservas, para lograr una eficiencia energética y equidad. R-MAC no usa el intercambio de mensajes "solicitud para enviar (RTS)/limpiar para enviar (CTS)" en colisiones de paquetes de datos. En su lugar, se programa la transmisión de paquetes de control y los paquetes de datos en el nodo fuente y en el nodo destino para evitar colisiones de paquetes.

R-MAC es un protocolo que soporta la equidad entre nodos y adopta una nueva técnica de solicitud de repetición automática (ARQ). ARQ es un protocolo de control de errores en la transmisión de datos. Este proceso se repite hasta que el paquete esté libre de error. El reconocimiento de la señal se realiza por un barrido de frecuencias combinado con los algoritmos de propagación.

El protocolo R-MAC se divide en tres fases: detección de latencia, periodo de comunicación y funcionamiento periódico. Las dos primeras fases son utilizadas para sincronizar los nodos de la red y la tercera fase es para operaciones de escucha/duerme. En la primera fase, un nodo detecta la propagación de todos sus vecinos. En la fase de periodo de comunicación, cada nodo selecciona aleatoriamente su propio estado de escucha/duerme y lo transmite a sus vecinos. Si hay datos, los transmite en la fase de funcionamiento periódico, es decir, cuando los nodos están escuchando y reciben datos, los retransmiten a sus demás vecinos y el nodo que retransmitió vuelve a dormir.

3.2.2. Broadcast

Cuando un nodo tiene paquetes para enviar, primero detecta el canal. Si el canal está libre, transmite los paquetes. De lo contrario, no lo envía. Cuando el receptor recibe un paquete no necesita enviar un ACK de vuelta al remitente. Este protocolo es simple, pero eficaz en redes de bajo tráfico. Además, este protocolo aprovecha todas las ventajas de la naturaleza de transmisión del canal acústico submarino y es adecuado para los protocolos de encaminamiento geográficos.

3.3. Protocolos de encaminamiento

Las posiciones geográficas de los nodos se usan en esta categoría de protocolos para las redes de sensores submarinas, específicamente, los protocolos de encaminamiento oportunistas, con el fin de seleccionar candidatos y tomar la decisión de reenviar los paquetes.

A continuación, se describen dos protocolos de encaminamiento oportunistas geográficos confiables como: VBF [9] y HH-VBF [10].

3.3.1. Reenvío Basado en Vectores (VBF)

Este protocolo basado en el encaminamiento oportunista geográfico no requiere de la información de estado de sus vecinos. Como se ha mencionado anteriormente, las comunicaciones acústicas sufren de un ancho de banda limitado, para no ver afectado de este problema al protocolo, VBF envía los paquetes a lo largo de las rutas redundantes para no generar pérdidas. Las posiciones del nodo fuente, el destino y el nodo pasarela están incluidas en el encabezado del paquete de datos que se trasmiten utilizando VBF. Con las posiciones del nodo fuente y el nodo destino se crea un *pipe* entre ambos nodos y a través de este *pipe* se reenvían los paquetes con ayuda de los nodos que se encuentren dentro de él (nodos pasarela) como se muestra en la Figura 3.1(a). Entonces, se puede decir que VBF es un protocolo de encaminamiento basado en el nodo fuente y el conocimiento de la posición del nodo destino.

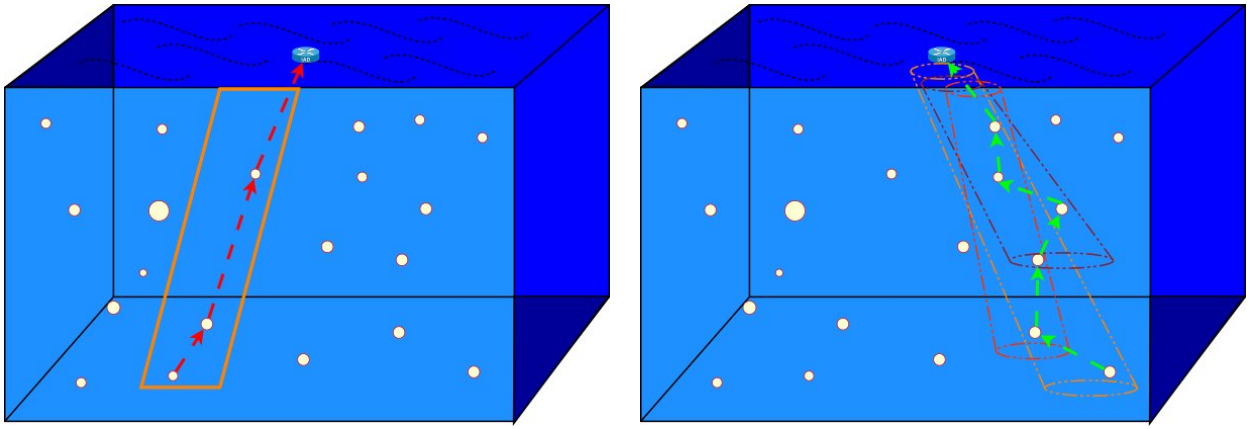
Cuando un nodo pasarela recibe un paquete, comprueba si está lo suficientemente cerca de la línea entre el nodo fuente y el nodo destino. Si un nodo se encuentra lo suficientemente cerca del *pipe*, incluye su posición en el encabezado del paquete como el promotor y lo transmite. Si

un nodo recibe el paquete, pero no se encuentra dentro del *pipe* que existe entre el nodo fuente y destino, el nodo pasarela simplemente descarta el paquete recibido. Si existen varios nodos dentro de este *pipe*, el resultado será que todos los nodos que reciban el paquete lo volverán a transmitir, provocando transmisiones duplicadas. El protocolo VBF consiste en un algoritmo de auto adaptación para reducir el número de veces de transmisiones entre los nodos pasarelas.

Cuando un nodo recibe un paquete, determina si está dentro del *pipe* entre el nodo fuente y el nodo destino para saber si es o no considerado como un promotor potencial. Si el nodo es un candidato potencial, espera un periodo de tiempo determinado por su factor de deseabilidad. Es el valor que muestra la proximidad del nodo al promotor anterior y el vector entre el nodo fuente y el destino. Cuanto más deseable sea el nodo, menos tiempo tendrá que esperar. Durante el tiempo de espera, el nodo escucha el medio para saber cuántos nodos están reenviando el mismo paquete que el nodo actual. Cuando el tiempo de espera termina, el nodo transmite su paquete. El enfoque de temporizador utilizado en VBF es similar a la coordinación basada en temporizador con las siguientes diferencias. El nodo de origen en VBF no especifica el conjunto de candidatos, sino que especifica el *pipe* entre el origen y destino. En contraste con el OR tradicional, dónde sólo un nodo debe reenviar el paquete, en VBF más de un candidato puede reenviar el paquete hacia el destino. Por lo tanto, la ventaja de VBF es que es escalable al número de nodos en las redes de sensores submarinas. Además, varios candidatos reenvían el paquete sobre caminos diferentes, la robustez del protocolo contra la pérdida de paquetes puede aumentar.

3.3.2. Reenvío basado en vectores de extremo a extremo (HH-VBF)

La consideración de un *pipe* entre el nodo fuente y el destino en VBF, da como resultado una baja entrega de paquetes en redes dispersas. VBF de extremo a extremo (HH-VBF) es una mejora de VBF. Por lo que crea un *pipe* en cada salto, a medida que los paquetes avanzan hacia el nodo destino. A diferencia de VBF que sólo crea un *pipe* entre el nodo fuente y el nodo destino, HH-VBF considera un *pipe* por cada salto, hasta llegar al nodo destino como se ve en la Figura 3.1(b). En otras palabras, cada nodo pasarela que se encuentre dentro del *pipe* forma otro *pipe* hacía el nodo destino. De tal manera de que en cada *pipe* generado se formen nuevos conjuntos de nodos candidatos para transmitir el paquete. Sin embargo, en VBF esto no sucede



(a) Reenvío Basado en Vectores (VBF)

(b) Reenvío Basado en Vectores de extremo a extremo (HH-VBF)

Figura 3.1: Protocolos de encaminamiento basado en vectores.

ya que en él sólo se forma el conjunto de candidatos una sola vez en todo el trayecto desde el nodo fuente y el nodo destino. Con esto nos damos cuenta de que en una red con pocos nodos hay una mayor probabilidad de entregar el paquete al nodo destino a través de nodos en cada *pipe* que se va generando con cada candidato.

Sin embargo, las transmisiones duplicadas siguen siendo un problema debido al método de coordinación utilizado en HH-VBF. Es decir, hay más de un candidato a reenviar el paquete. Estas transmisiones pueden causar colisiones de paquetes y desperdicio en el ancho de banda, que es un parámetro crítico en las redes de sensores submarinas.

VBF y HH-VBF comparten el problema de las regiones vacías de la comunicación. Por lo tanto, si un paquete se entrega a un nodo que ya no puede reenviarlo, el paquete se descarta porque no hay otros nodos en una proximidad cercana al destino.

Capítulo 4

Simuladores para UWSN

4.1. OMNet++

OMNeT++ [13] es un simulador modular de eventos discretos de redes orientado a objetos. Es una arquitectura genérica que ha sido y puede ser utilizada para varios tipos de problemas:

1. Modelado de redes de comunicación cableadas e inalámbricas
2. Modelado de protocolos
3. Modelado de colas de espera en redes
4. Modelado de multiprocesadores y otros sistemas de hardware distribuido
5. Validación de arquitecturas de hardware
6. Evaluación de aspectos de rendimiento de sistemas de software complejos

En general, puede ser utilizado para el modelado y la simulación de cualquier sistema donde el enfoque de eventos discretos es adecuado y puede ser convenientemente mapeado en entidades que se comunican mediante el intercambio de mensajes. En sí mismo, OMNeT++ no es un simulador de nada en concreto, sino que proporciona la infraestructura y herramientas para la escritura de simulaciones. Uno de los ingredientes fundamentales de esta infraestructura es una arquitectura de componentes para los modelos de simulación. Los modelos se ensamblan a partir de componentes reutilizables llamados módulos. Bien escritos son realmente módulos reutilizables y se pueden combinar de varias maneras como bloques de LEGO.

Los módulos se comunican con mensajes que pueden contener datos arbitrarios, además, atributos habituales como una marca de tiempo. Los módulos simples suelen enviar mensajes a través de puertas, pero también es posible enviarlos directamente a sus módulos de destino. Las puertas son las interfaces de entrada y salida de los módulos. Los mensajes se envían a través de puertas de salida y llegan a través de puertas de entrada. Una puerta de entrada y una puerta de salida pueden conectarse mediante una conexión. Las conexiones se crean dentro de un solo nivel de jerarquía de módulos. Dentro de un módulo compuesto, pueden conectarse puertas correspondientes de dos submódulos o una puerta de un submódulo y una puerta del módulo compuesto. Las conexiones que abarcan niveles de jerarquía no están permitidas, ya que impedirían la reutilización del modelo. Debido a la estructura jerárquica del modelo, los mensajes típicamente viajan a través de una cadena de conexiones, comenzando y llegando en módulos simples. Los módulos compuestos actúan como "cajas de cartón" en el modelo para transmitir transparentemente mensajes entre su reino interior y el mundo exterior. Parámetros tales como retraso de propagación, velocidad de datos y tasa de errores de bits, se pueden asignar a las conexiones. También se pueden definir tipos de conexión con propiedades específicas (denominadas canales) y reutilizarlas en varios lugares. Los módulos pueden tener parámetros, que se utilizan principalmente para pasar los datos de configuración a módulos simples y ayudar a definir la topología del modelo, además de:

1. Tomar valores de cadena, numéricos o booleanos
2. Pueden actuar de forma transparente como fuentes de números aleatorios, con las distribuciones reales proporcionadas de la configuración del modelo
3. Los módulos compuestos pueden transmitir parámetros o expresiones de parámetros a sus submódulos

OMNeT++ proporciona herramientas eficientes para que el usuario pueda describir la estructura del sistema actual. Algunas de las características principales son las siguientes:

1. Módulos jerárquicamente anidados
2. Los módulos son instancias de tipos de módulos
3. Los módulos se comunican con mensajes a través de canales

4. Parámetros flexibles del módulo

5. Lenguaje de descripción de topología

A pesar de que OMNeT + + puede tener buenos resultados con respecto a los bloques que maneja, no tiene los bloques necesarios para simular un medio marino o hacer una simulación en tres dimensiones. Las problemáticas a las que se enfrentan las redes submarinas no son resueltas en su totalidad por OMNeT + +.

4.2. UWSim

UWSim es una herramienta de software que ayuda a la visualización y simulación de misiones robóticas bajo el agua [14]. El software visualiza el escenario virtual submarino que puede configurarse utilizando el software de modelado estándar. Los vehículos submarinos controlables, los bloques de superficie y los manipuladores robóticos, así como la simulación de sensores, pueden acceder a la escena externamente a través de las interfaces de red. Esto con la idea de integrar fácilmente la herramienta de simulación de bucle (HIL). UWSim es un software de código abierto para la comunidad de robótica submarina, donde predominan los simuladores comerciales orientados a la formación de un vehículo con control remoto (ROV).

La experimentación con robots submarinos es normalmente muy difícil debido al gran número de recursos requeridos. Con el fin de facilitar el desarrollo de robots submarinos, es de gran importancia crear simuladores adecuados para:

- Los sistemas antes de su despliegue.
- Supervisar una tarea submarina real donde el supervisor no tiene una visión directa del sistema.

UWSim es un software de simulación submarina de código abierto diseñado con base en los siguientes objetivos principales:

1. Ser fácilmente integrable con las arquitecturas de control existentes. Los algoritmos de control son externos al simulador que en muchos casos sólo funciona como una visualización de la salida calculada por programas externos. El software está dirigido principal-

mente a investigadores y desarrolladores de robótica submarina, aunque también podrían implementarse escenarios especiales para la formación de pilotos de ROV.

2. Ser general, modular y fácilmente extensible. Los nuevos robots se pueden incluir fácilmente en los archivos de descripción XML. También se proporciona soporte para widgets, lo que permite mostrar información útil en la parte superior de la escena.

3. Incluir soporte para manipuladores subacuáticos, permitiendo simular misiones de intervención bajo el agua. Se pueden crear y controlar cadenas cinemáticas.

4. Ser visualmente realista y permitir al usuario la configuración de parámetros importantes como el color del agua, la visibilidad, las partículas flotantes, etc.

UWSim está implementado en C++ y hace uso de las bibliotecas de gráfico de escena abierta (OSG) y osgOcean. OSG es una interfaz de programación de aplicaciones de gráficos 3D de código abierto. Las herramientas están escritas en C++ usando OpenGL y se ejecuta en una variedad de sistemas operativos (Microsoft Windows, Mac OS X, Linux, Irix, Solaris, FreeBSD y Android). Por otro lado, osgOcean es otro proyecto de código abierto que implementa renderizado subacuático realista y usa OSG.

UWSim utiliza las bibliotecas antes mencionadas y agrega funcionalidad adicional para agregar fácilmente robots submarinos, simular sensores y hacer la interfaz con programas de control externo a través del sistema operativo de robot (ROS).

4.2.1. Configuración del entorno

La geometría 3D de UWSim se puede configurar fácilmente con software de modelado de terceros como Blender, 3D Studio Max, etc. La escena básica puede ser modelada libremente, incluyendo materiales y texturas. La escena resultante tiene la posibilidad de ser cargada en el simulador siempre y cuando sea exportada a cualquiera de los formatos que OSG pueda leer (.ive, .wrl, .3ds, etc.).

Además de la estructura básica, los elementos adicionales se pueden agregar dinámicamente, se modifican y se quitan del programa principal. OSG representa el escenario virtual con un escenario gráfico, donde los nodos pueden ser accesibles y gestionados.

La escena completa puede ser descrita por el usuario con un archivo XML. En la Tabla 4.1 se pueden ver las etiquetas que se pueden utilizar y sus funciones.

4.2.2. Soporte de múltiples robots

Un vehículo predeterminado de UWSim se compone de un modelo 3D que se puede posicionar en la escena estableciendo seis grados de libertad (DOF). También, se incluye soporte para cadenas cinemáticas (necesarias para manipuladores). Los robots se crean con un archivo XML de acuerdo con el formato de descripción de robot unificado (URDF), que puede incluir información cinemática, dinámica y visual.

4.2.3. Sensores simulados

En la versión actual de UWSim se pueden simular cuatro sensores diferentes. Por defecto, cualquier objeto del vehículo incluye sensores de localización que proporcionan su posición de 6 DOF (x, y, z, roll, pitch, yaw) en la escena. El ruido gaussiano se puede añadir, por ejemplo, para simular incertidumbres en las lecturas del sensor. Los manipuladores subacuáticos también incluyen sensores de posición simulados en los ángulos de unión.

Las cámaras virtuales se pueden agregar a los vehículos, proporcionando imágenes virtuales del entorno que pueden utilizarse para desarrollar algoritmos de visión. Estas cámaras pueden ser iniciadas desde parámetros intrínsecos, permitiendo de esta manera la utilización de cámaras virtuales con las mismas propiedades visuales de sus contrapartes reales.

Finalmente, también se incluyen sensores virtuales que miden obstáculos a distancia a lo largo de direcciones predefinidas (simulando sensores de alcance). Muchos de estos sensores se pueden agregar a los vehículos especificando los límites del rango.

4.2.4. Interfaces de red

Los diferentes sensores y actuadores de robots pueden ser conectados con software externo a través de la red.

UWSim incluye una interfaz para su integración con el Robot Operating System (ROS), que es un conjunto de bibliotecas y herramientas. Estas ayudan a los desarrolladores de software a crear aplicaciones robóticas. ROS es un sistema distribuido en el que diferentes nodos pueden

Bloque	Función
<oceanState>	Permite la configuración de los parámetros del océano como dirección y velocidad del viento (afecta a la cantidad de ondas), color subacuático, visibilidad y factores de atenuación.
<simParams>	Permite al usuario desactivar los efectos de visualización, establecer la resolución de la ventana y establecer un marco mundial, que es un desplazamiento con respecto al predeterminado.
<camera>	Establece los parámetros principales de la cámara. La cámara principal es la que observa la escena y la hace a la ventana principal. El usuario puede configurar el modo de movimiento de la cámara (cámara libre frente a la cámara) y otros parámetros como el campo de visión, la relación de aspecto y los planos de recorte. También es posible ajustar los parámetros de la cámara desde la matriz de calibración intrínseca.
<vehicle>	Pueden incluir robots submarinos. El usuario tiene que especificar un archivo de descripción del robot en URDF; valores de unión por defecto, en caso de que el robot contenga articulaciones; la postura del robot en la escena y los sensores del vehículo si es necesario.
<object>	Permite incluir otros modelos 3D en la escena de recursos de archivos existentes en cualquiera de los formatos 3D soportados por OSG.
<rosInterfaces>	Permiten asociar interfaces ROS a determinados objetos. Estos proporcionan información de sensores a software externo (pose de objetos, imágenes de cámaras virtuales, valores conjuntos, etc.) y también reciben referencias externas usadas para actualizar la pose de objetos y robots en la escena.

Tabla 4.1: Bloques de XML.

ejecutarse en diferentes computadoras y principalmente comunicarse a través de "temas" mediante mecanismos de publicación/suscripción. La interfaz ROS puede comunicarse con el resto de la arquitectura con las instalaciones de comunicación ROS estándar. Esto permite validar perfectamente los métodos de control desarrollados en ROS, ya sea en UWSim o en los robots reales, siempre que proporcionen la misma interfaz.

A través de las interfaces ROS, es posible acceder/actualizar a cualquier posición o velocidad del vehículo, mover las articulaciones del brazo y acceder a los datos generados por los sensores virtuales.

4.3. Aqua-Sim basado en NS-2.30

NS-2 [15], es un simulador que no puede hacer frente fácilmente a entornos de redes submarinas. En especial con los canales acústicos submarinos que presentan largos retardos de propagación y alta atenuación. NS-2 no puede aplicar modelos de atenuación que existen para los canales terrestres. Además, los largos retardos de propagación hacen que los comportamientos de colisión sean muy diferentes a los de las redes terrestres. Esto requiere una forma diferente de simular las colisiones en redes de sensores submarinas. Por otro lado, el despliegue tridimensional no está soportado en el paquete inalámbrico CMU de NS-2. También se desean algunos protocolos MAC y de encaminamiento básicos, adaptados a entornos de redes submarinas, para la evaluación de cualquier protocolo avanzado por medio de simulaciones. Sin embargo, NS-2 no cuenta con este tipo de simulaciones, pero Aqua-Sim [16] es un módulo que se agrega a NS-2 y que permite extenderlo a redes submarinas de sensores inalámbricos. En NS-2, Aqua-Sim está en paralelo con el paquete de simulación inalámbrica CMU. Aqua-Sim es independiente del paquete de "wirelessimulation″ y no se ve afectado por ningún cambio en el paquete de "wireless″. Por otro lado, cualquier cambio en el código de Aqua-Sim es independiente de sí mismo y no tiene ningún impacto con otros paquetes de NS-2. De esta manera, Aqua-Sim puede evolucionar de manera independiente.

Aqua-Sim sigue el estilo de diseño orientado a objetos de NS-2, y todas las entidades de red se implementan como clases en C++. Actualmente, Aqua-Sim está organizado en cuatro carpetas, uui_common, uw_mac, uui_routing y uw_tcl. Los códigos que simulan los nodos de sensores subacuáticos y el tráfico se agrupan en la carpeta uw_common. Los códigos que

simulan los canales acústicos y los protocolos MAC se organizan en la carpeta de uw_mac. La carpeta uw_routing contiene todos los protocolos de encaminamiento. La carpeta uw_tcl incluye todos los ejemplos de script Otcl para validar Aqua-Sim.
En general, Aqua-Sim tiene tres tipos de clases.

1. **Clases de entidad de red:** este tipo de clases representan entidades de red concretas.

2. **Clases de interfaz pura:** este tipo de clases son puramente virtuales y no se pueden crear instancias en absoluto. Sin embargo, especifican interfaces comunes y sirven como clases base para otros.

3. **Clases de funciones comunes:** este tipo de clases proporcionan algunas funciones comunes a otras clases y pueden incluirse en cualquier clase de Aqua-Sim. Aunque estas clases no tienen instancias en el nodo de red correspondiente, son muy importantes para Aqua-Sim.

Todas las entidades de red están representadas por sus objetos correspondientes e interactúan entre sí de la misma manera que se especifica en la pila de protocolos. Por ejemplo, si la entidad del protocolo MAC en el nodo "A" desea transmitir un paquete de control al nodo "B", pasará el paquete directamente a su objeto "UnderwaterPhy" que es el representante de su capa física. Este paquete se pasará entonces al único objeto "UnderwaterChannel", que pondrá el paquete en una cola utilizando el concepto FIFO y calculará el retardo y el offset de transmisión en cada nodo receptor. Este paquete entonces se pasa al objeto "UnderwaterPhy" en el nodo "B". Finalmente, alcanzará el objeto de capa MAC en el nodo "B"

4.3.1. Capa de Red

Es indispensable conocer cómo se opera en la capa MAC, pero para un algoritmo de sincronización se debe trabajar especialmente en la capa de encaminamiento. Por esto, Aqua-Sim tiene a todas las clases relacionadas con la capa de encaminamiento que se incluyen en la carpeta de uw_routing. La implementación de todos los protocolos de encaminamiento sigue una estructura estándar de los protocolos existentes en NS-2. Los parámetros para los protocolos se pueden ajustar a través del script Tcl. Con el fin de soportar las características avanzadas de los protocolos de encaminamiento en redes de sensores submarinos, Aqua-Sim proporciona

interfaces estándar en casi todos los protocolos de la capa de red, por ejemplo, los protocolos de encaminamiento geográfico basados en posiciones de nodo pueden obtener fácilmente la información de ubicación desde la interfaz del objeto "Submarino". Así, Aqua-Sim proporciona una buena plataforma para desarrollar protocolos avanzados de encaminamiento.

Actualmente, en Aqua-Sim se implementan tres protocolos de encaminamiento que incluyen:

1. **Reenvío Basado en Vectores (VBF):** es un protocolo de encaminamiento geográfico (descrito anteriormente).

2. **Encaminamiento Basado en Profundidad (DBR):** es un protocolo de encaminamiento basado en un algoritmo codicioso de reenvío. Utiliza la información de profundidad de nodos de sensores submarinos para enviar paquetes desde un nodo de origen hacía nodos desplegados en la superficie del agua. Los paquetes siguen la regla de reducir la profundidad de los nodos de reenvío en cada paso de direccionamiento hacia la superficie del agua. En DBR, una vez que se recibe un paquete, un nodo primero obtiene la información de profundidad del nodo de salto anterior que se ha registrado en el paquete. El nodo receptor compara su propia profundidad con la profundidad del nodo de salto anterior. Si la propia profundidad del nodo receptor es menor que el salto anterior, es decir, el nodo receptor está más cerca de la superficie del agua, se considerará como un candidato cualificado para reenviar el paquete. De lo contrario, simplemente lo descarta.

3. **QELAR:** es un protocolo de encaminamiento adaptativo, eficiente en energía y basado en el aprendizaje de refuerzo. Al extraer la información necesaria del entorno subacuático en tiempo de ejecución, los agentes de aprendizaje son capaces de tomar decisiones óptimas de encaminamiento para reducir el recuento de saltos a nodos que se encuentren en una mayor profundidad del escenario simulado, así como para hacer que la energía residual se distribuya uniformemente y por lo tanto la vida de toda la red se prolonga en gran medida. Al mismo tiempo, el protocolo se puede configurar fácilmente para equilibrar el consumo de energía y la distribución de energía residual para hacerse más flexible.

4.4. Aqua-Sim basado en NS-3

Aqua-Sim es un simulador de redes submarinas que soporta una gran cantidad de protocolos y características. Originalmente desarrollado sobre la base de NS-2, Aqua-Sim puede simular de manera efectiva colisiones, atenuación de la señal y de paquetes en (UWSN). Por otra parte, Aqua-Sim admite la distribución tridimensional. El último trabajo de Aqua-Sim consiste en volver a escribir el código y portar Aqua-Sim en NS-3 para mejorar temas actuales de simulación tales como fugas de memoria y simplicidad de uso.

4.5. Simulador seleccionado

De acuerdo con la descripción de los simuladores, se opta por utilizar Aqua-Sim que es un simulador que utiliza todas las ventajas, desarrollo y además sigue el estilo de diseño orientado a objetos de NS-2.30, por esto puede integrar fácilmente los códigos existentes de NS-2.30. Las principales características para elegir este simulador son las siguientes: En primer lugar, la comunicación acústica es el método comúnmente aceptado para entornos subacuáticos, mientras que la velocidad de propagación de la señal acústica en el agua es muy lenta (aproximadamente 1500 m/s), significativamente diferente de la de la señal de radio; en segundo lugar, el modelo de atenuación de la señal acústica es drásticamente diferente del de la señal de radio y, por lo tanto, deben incorporarse modelos de canales acústicos; en tercer lugar, las redes de sensores subacuáticos suelen desplegarse en un espacio tridimensional, mientras que estos simuladores generalmente sólo admiten el despliegue bidimensional.

Capítulo 5

Evaluación

La evaluación de desempeño de los protocolos MAC y de encaminamiento a través de un simulador de eventos discretos, necesita de un conjunto de medidas de desempeño que muestre los efectos de cada esquema de encaminamiento inducido a la red. Por este motivo, se ha elegido utilizar el simulador Aqua-Sim, basado en NS-2 el cual realiza simulaciones de UWSN necesarias para generar trazas de tiempo. El protocolo de encaminamiento que se utiliza es HH-VBF implementado con los protocolos R-MAC y Broadcast de la capa de MAC.

Como se mencionó en la sección 3 2, los protocolos R-MAC y Broadcast de la capa MAC, se encargan de propagar las tramas a todos los nodos vecinos. R-MAC transmite el paquete hasta que el nodo receptor lo reciba libre de errores. Broadcast detecta que el canal de comunicación esté vacío para transmitir los paquetes. En las secciones posteriores, se describe el comportamiento de los protocolos MAC antes mencionados. Para estas diferencias entre protocolos de la capa MAC se simula el mismo escenario en ambos casos.

5.1. Escenarios de simulación

Uno de los elementos más importantes para este tema de investigación es la simulación de la red utilizando los protocolos de encaminamiento y MAC mencionados en la sección anterior. Con ayuda del simulador Aqua-Sim basado en NS-2 se crea una UWSN con 40 nodos, en un volumen de 100 x 100 x 100 [m^3], para tener un escenario de simulación como se muestra en la Figura 5.1. El canal de comunicación que se utiliza es un canal submarino mientras que en la capa MAC se utiliza un sistema de broadcast y R-MAC respectivamente. Es importante mencionar que

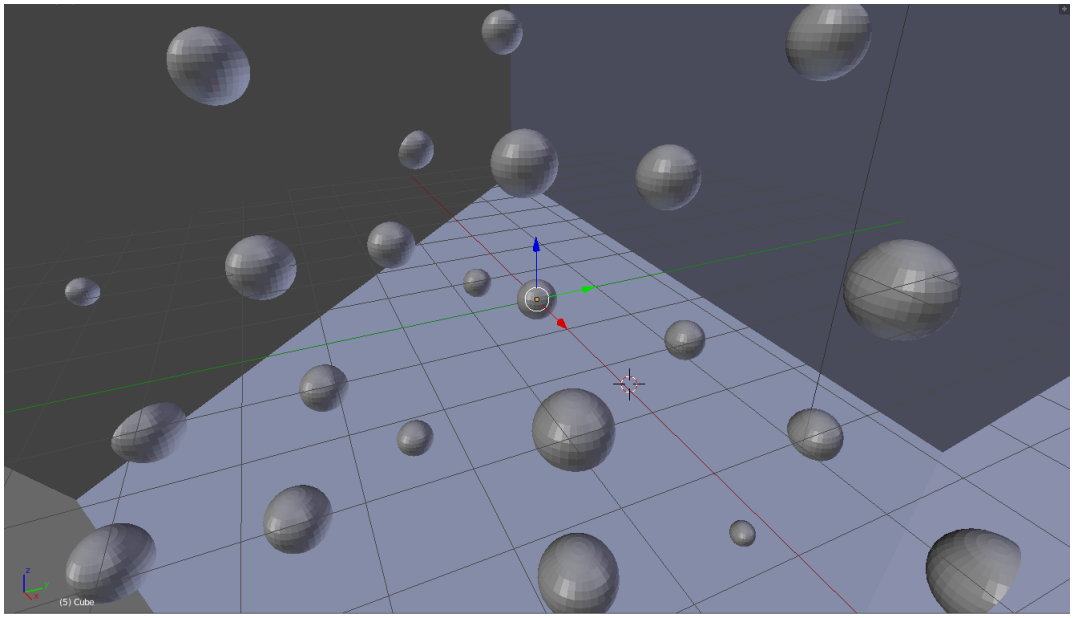


Figura 5.1: Escenario de simulación.

cada ejecución es de 100 simulaciones con una duración de 1500 segundos, puesto que en cada simulación los nodos se comportan de diferente manera. Por otro lado, el valor de la energía para cada nodo se considera infinito porque en este trabajo sólo nos estamos enfocando en la sincronización por señalización. La característica establecida de encaminamiento en Aqua-Sim utilizada en este estudio ha sido "Vectorbasedforward" que trabaja con Reenvío Basado en Vectores. Como se menciona en la subsección 3.3.2, los canales acústicos en este protocolo se utilizan como el método de comunicación. Por último, se obtiene un tipo de traza nueva, utilizando la función "use – newtrace" la cual facilita la lectura y obtención de los datos de tiempo requeridos.

5.2. Modelo de movilidad

El modelo de movilidad es un punto importante debido a la naturaleza de aguas marítimas. Gran parte del agua de mares y océanos no está quieta, además, tiene diversos tipos de corrientes. El primer movimiento del mar que llama la atención es el de las olas. Las olas son generadas por el viento. Mientras más fuerte sople el viento, más grandes son las olas. Las olas no desplazan agua, pero cuando llegan a costas, la amplitud del movimiento es mayor al volumen del agua y esta adopta un movimiento cilíndrico que desequilibra la masa del agua ocasionando la ruptura de

la ola. Determinados movimientos sísmicos marinos son capaces de provocar ondas que generan olas muy fuertes (tsunamis).

Por eso, la mayoría de los nodos sensores (excepto boyas de superficie) tienen una movilidad baja o media, dependiendo de la corriente de agua y otras actividades submarinas. Los objetos submarinos se mueven entre $3 - 6$ [km/hr] en una condición típica. El protocolo de encaminamiento HH-VBF utiliza una aproximación del modelo de movilidad entre $1 - 3$ [m/s].

Por otro lado, se puede especificar el modelo de movilidad de los nodos de la siguiente manera: el nodo selecciona aleatoriamente una velocidad entre "max_speed" y "min_speed" y se mueve en una dirección aleatoria, su posición se actualiza cada "position_update_interval" segundos. Con esto se puede generar el movimiento que se desee, pero para este trabajo se utiliza el modelo de movilidad del protocolo HH-VBF.

Capítulo 6

Resultados

Una vez realizado el escenario de simulación se obtienen las trazas generadas por el simulador, las cuales se filtran para obtener los datos de tiempo. Cuando se tienen los filtros de trazas de tiempo, se realiza un análisis de regresión lineal. El modelo de regresión lineal [20], se utiliza como una forma de estimar una variable Y de otra variable X dada la información sobre la asociación entre X y Y. Utilizando el criterio de mínimos cuadrados o regresión de mínimos cuadrados ordinarios en la regresión lineal, se calcula el peso de X y los valores reales de Y para minimizar la suma de los cuadrados residuales. Los residuales representan la diferencia entre la estimación de un modelo de Y y los valores reales de Y observados en un conjunto de datos.

En este capítulo se toma a X como el tiempo del nodo al que vamos a hacer referencia, de igual manera, se tomará a Y como el número de nodo. La regresión lineal se puede calcular a partir de covarianzas, estas no suelen ser interpretadas, pero son útiles para calcular los coeficientes de regresión y las correlaciones. En la ecuación (6.1), se define la desviación del tiempo (t_i) como:

$$t_i = T_i - \bar{T} \tag{6.1}$$

donde t_i es el instante del tiempo i menos la media total del tiempo. De la misma manera, en la ecuación 6.2 se obtiene la desviación del nodo (n_i)

$$n_i = N_i - \bar{N}. \tag{6.2}$$

Por último, se hace el producto de t_i con n_i. Contando con estos valores se puede generar una matriz en Matlab con la finalidad de tener estos datos en un solo lugar, ya que estos valores

Tabla 6.1: Valores para la regresión y modelo lineal.

tiempo	nodo	Desviación de T	Desviación de N
T_1	N_1	t_1	n_1
T_2	N_2	t_2	n_2
$\vdots$	$\vdots$	$\vdots$	$\vdots$
T_m	N_m	t_m	n_m

son los que se utilizarán para obtener la regresión lineal.

Esta información es la primera parte para poder realizar la regresión lineal que se obtiene de la siguiente manera:

$$T = b_0 + b_1 N \tag{6.3}$$

Donde b_0 es el offset de nuestra función lineal y b_1 es la pendiente de la regresión lineal, para determinar los valores de b_0 y b_1. Por [20], se sabe que la covarianza es la media del producto de T y N. La covarianza $Cov(TN)$ se denota por:

$$Cov(TN) = \frac{\sum_{i=1}^{k} t_i n_i}{k} \tag{6.4}$$

También, se sabe que la covarianza de cualquier variable con sí misma es la varianza de la variable. La varianza de T se denota por $Var(T)$ como:

$$Var(T) = \frac{\sum_{i=1}^{k} t_i t_i}{k} = \sum_{i=1}^{k} \frac{t_i^2}{k} \tag{6.5}$$

Las covarianzas pueden definir el coeficiente de regresión b_1 de la siguiente manera:

$$b_1 = \frac{Cov(TN)}{Var(T)} \tag{6.6}$$

Una vez que se tiene el valor de b_1, se puede saber el valor de b_0 de la siguiente manera:

$$b_0 = \bar{T} - b_1 \bar{N} \tag{6.7}$$

De los valores encontrados en las ecuaciones 6.6 y 6.7, se tienen las constantes del offset y la pendiente para la regresión lineal. De esta manera, se tiene una aproximación de los valores b_0 y b_1 que son el offset y la pendiente respectivamente. Se hacen dos aproximaciones de la

Tabla 6.2: Aproximación de la regresión lineal.

	Primera		Segunda	
	b_1	b_0	b_1	b_0
Broadcast	0.0238	4.3648	1	$-1.38x10^{-13}$
R-MAC	0.0267	-1.1503	1	$-2.45x10^{-13}$

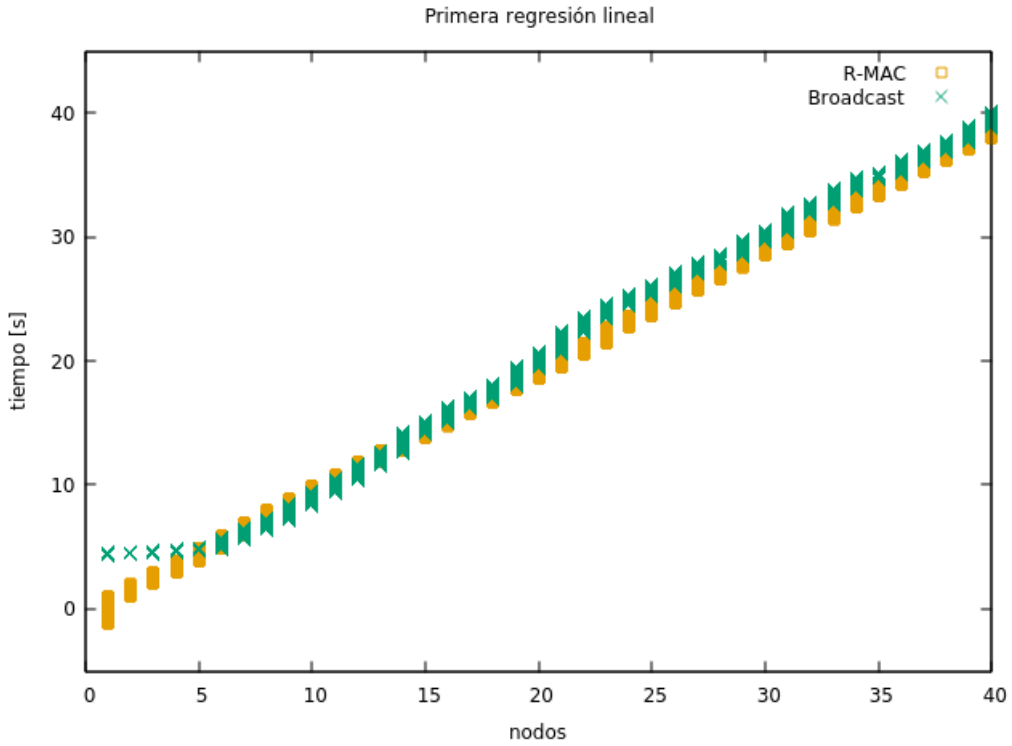
regresión lineal para observar la mejoría que hay entre una o dos regresiones como se muestran en la Tabla 6.2.

Con las constantes de la primera y segunda regresión lineal se sustituyen las variables generadas originalmente por las trazas del simulador, como se muestran en las Figuras 6.1(a) y 6.1(b). Con la primera regresión lineal se puede observar el comportamiento de los protocolos MAC (Broadcast y R-MAC). En los primeros nodos, el protocolo Broadcast tiene un offset mayor al de R-MAC, sin embargo, cuando aumentan los nodos, los tiempos de ambos protocolos llega a coincidir.

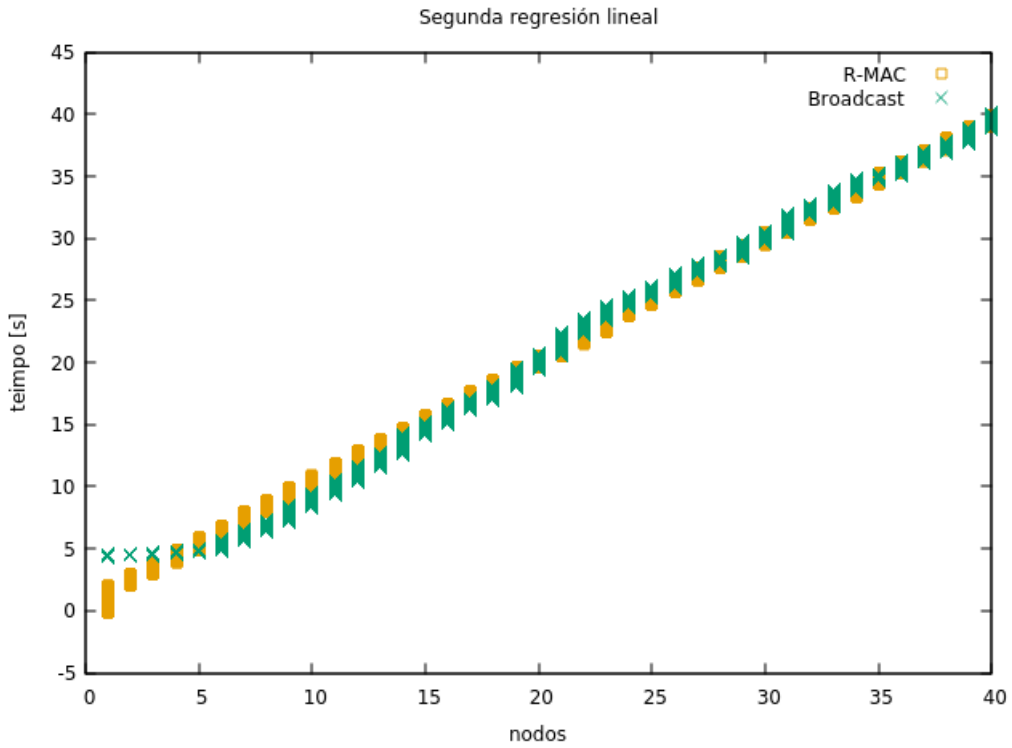
En la segunda regresión lineal, se tiene el mismo comportamiento que en la primera con los primeros nodos, pero cuando estos se aumentan, el comportamiento del protocolo Broadcast tiende ser más rápido que R-MAC. De acuerdo con los datos de la Tabla 6.1, esto corresponde a los datos aproximados, debido que los valores del offset y de la pendiente tienden a ser muy parecidos, en especial para la segunda regresión lineal.

La mejora de la segunda regresión lineal en cada protocolo simulado puede ser casi igual, sin embargo, existe una ligera diferencia entre cada offset y la pendiente de las regresiones. Por este motivo, la diferencia entre la primera regresión y la segunda parecen ser iguales como se ve en las Figuras 6.2(a) y 6.2(b). Para R-MAC se observa en la Figura 6.2(b) que en la segunda regresión lineal el offset de tiempo aumenta, pero el comportamiento sigue siendo el mismo. También, se observa que el protocolo R-MAC tiende a tener una distribución uniforme en todos sus nodos, a diferencia de Broadcast que su distribución varía mucho en cada nodo.

Con las aproximaciones obtenidas, se realiza la calibración que nos ayuda a predecir el comportamiento de la sincronización de un número n de nodos. Para esto, se utilizan dos puntos aproximados (obtenidos de la segunda regresión lineal), que ayudan a calcular la pendiente (b_1) de la calibración por medio de 6.3 despejando a b_0.

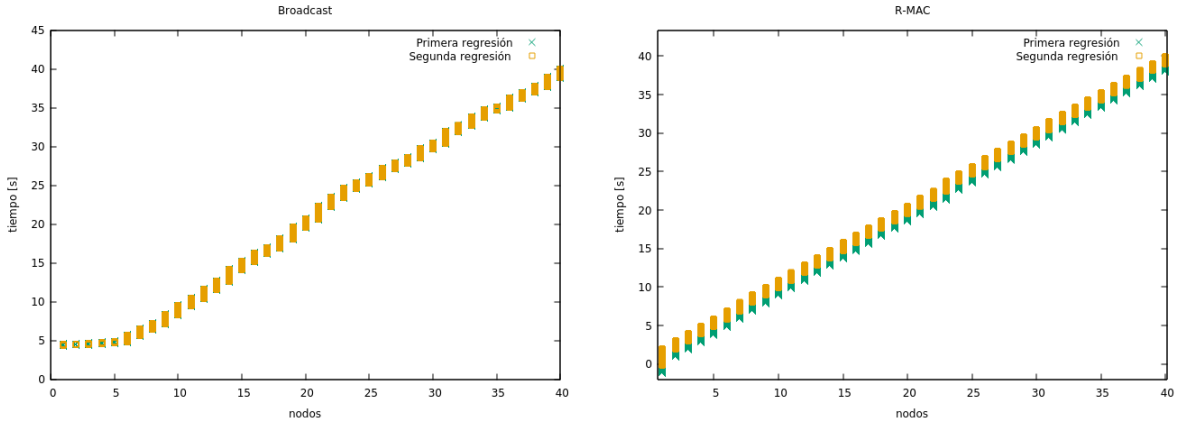


(a) Primera regresión lineal



(b) Segunda regresión lineal

Figura 6.1: Regresión lineal.



(a) Broadcast (b) R-MAC

Figura 6.2: Primera vs Segunda regresión lineal.

Tabla 6.3: Calibración para la sincronización de tiempo en UWSN.

	Tiempo máximo	Tiempo mínimo	Pendiente	Offset
Broadcast	40.0094	4.4200	1.0958	−0.0958
R-MAC	39.9111	−0.1232	0.9741	0.0258

Ahora, se tienen las constantes de calibración expresadas en la Tabla 6.3 y se observa que la diferencia del valor de la pendiente y del offset entre cada protocolo es prácticamente la misma. De hecho, al hacer un redondeo en los valores se tendría que la pendiente es igual a 1 y el offset es igual a 0. En otras palabras, el tiempo de sincronización es igual al número de nodo. Pero al realizar la gráfica tomando los valores de la calibración para la sincronización de tiempo, se observa en la Figura 6.3 que las líneas de calibración no crecen de la misma manera.

De acuerdo a la Figura 6.3 se puede notar que R-MAC tiene una pendiente menor a Broadcast con 40 nodos, sin embargo, de acuerdo a la naturaleza del mar, el movimiento de los nodos puede formar una zona densa de nodos, por este motivo se hace una simulación de red más densa, ahora con 120 nodos en el mismo volumen de 1000m x 1000m x 1000m. Para la primera regresión lineal se obtienen los datos mostrados en la Figura 6.4(a) en donde se puede observar que R-MAC hace una mejor sincronización a diferencia de Broadcast, pero sólo para antes de 45 nodos, por otro lado, cuando se tienen más de 45 nodos aproximadamente, Broadcast mejora el tiempo de sincronización en comparación a R-MAC. Un efecto importante que se nota en

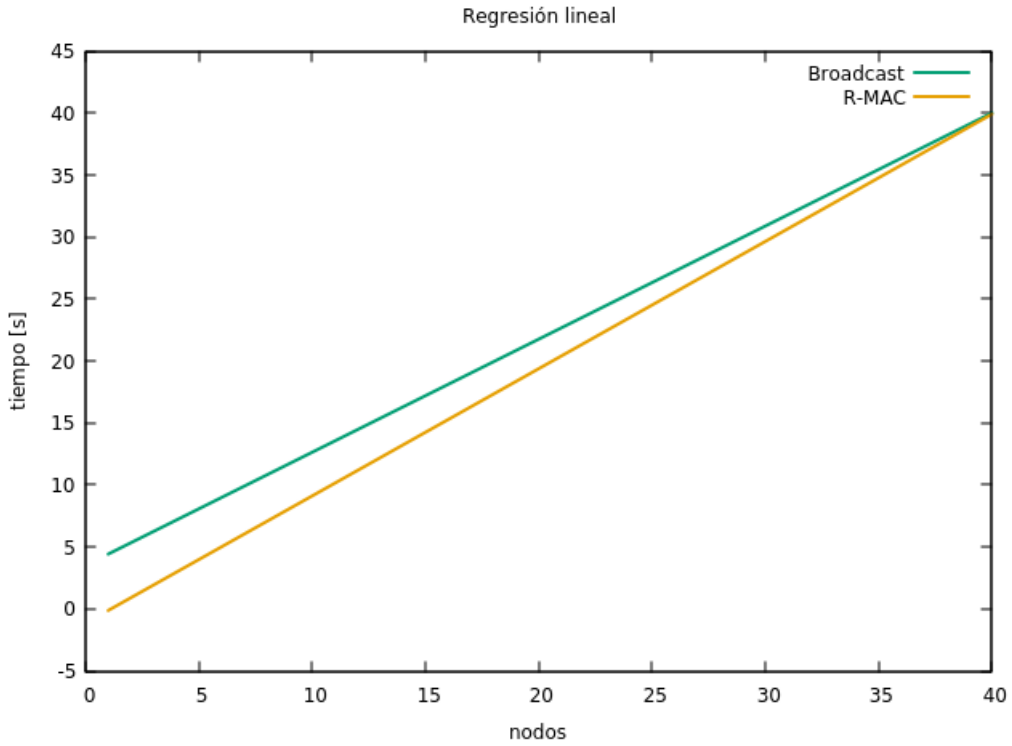
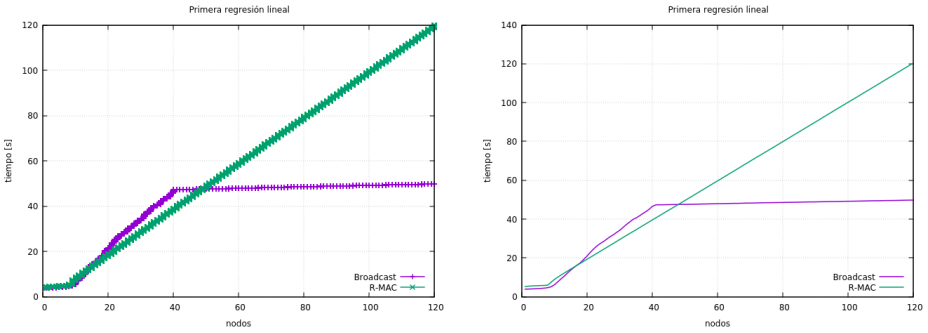


Figura 6.3: Calibración de la regresión lineal.

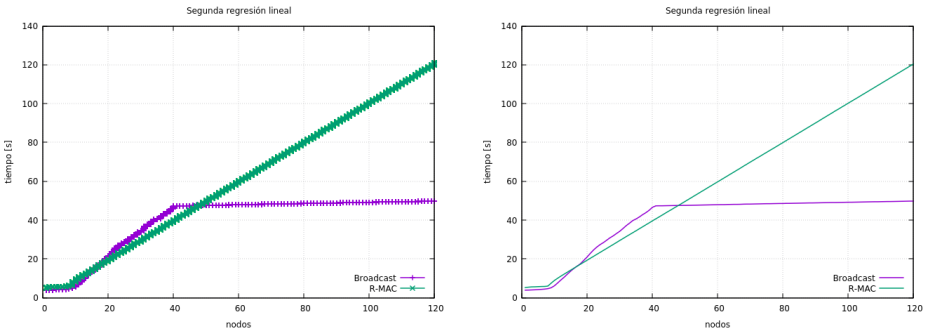
Broadcast es que a partir de 40 nodos su pendiente tiende ser a cero.

Para la segunda simulación con 120 nodos también se tienen dos regresiones lineales y con esto se puede decir que R-MAC va creciendo con una pendiente aproximadamente de 1 nodo/s, por otro lado, Broadcast primero tiene un crecimiento de pendiente exponencial, al incrementar los nodos está pendiente empieza a disminuir hasta quedar con una pendiente aproximadamente de cero. En 6.5(c) se observa que el rango de error en cada nodo puede variar entre ± 1[s] lo cual incrementa el nivel de confianza de acuerdo de los datos obtenidos en la investigación.

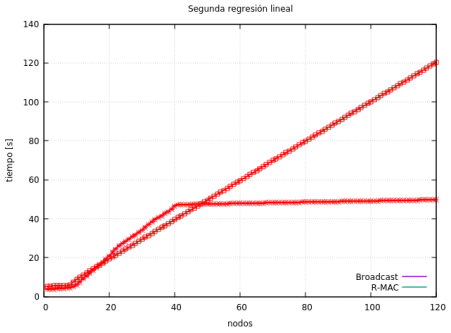


(a) Datos filtrados (b) Calibración

Figura 6.4: Primera regresión lineal con 120 nodos.



(a) Datos filtrados (b) Calibración



(c) Error

Figura 6.5: Segunda regresión lineal con 120 nodos.

Capítulo 7

Conclusiones

En este trabajo se ha realizado un análisis de los protocolos de encaminamiento y MAC para UWSN, con el propósito de realizar la sincronización de tiempo en una red de nodos sensores submarinos. De acuerdo con la literatura encontrada, existen difíciles condiciones ambientales bajo el agua. Por esta razón, surge la necesidad de protocolos que nos sirvan en las UWSN. Actualmente, existen diversos trabajos de investigación en la sincronización de tiempo en redes submarinas de sensores inalámbricos, para esto se encontraron algunos de algoritmos como los citados en [2–8]. En [2], se utiliza un protocolo en encaminamiento geográfico que ayuda a la sincronización y adicionalmente a la ubicación de nodos. Para este trabajo sólo nos basamos en la Fase II de este protocolo que es la sincronización por medio de la estimación del retardo de la propagación, regresión lineal y la actualización del retardo de la propagación.

Se buscaron algunos protocolos de encaminamiento y MAC para UWSN. En los protocolos de encaminamiento, de acuerdo con la literatura consultada, hay una cierta consideración con los protocolos de encaminamiento Oportunistas Geográficos. Por este motivo se decidió trabajar con HH-VBF. Este protocolo tiene un buen funcionamiento tanto para redes dispersas como en redes densas; además de utilizar la redundancia de la ruta generada que debe estar dentro del *pipe* generado entre cada nodo fuente (pasarela) y el destino.

De acuerdo con la investigación, se utilizaron dos protocolos MAC para hacer el estudio de R-MAC y Broadcast. R-MAC es un protocolo que apoya la equidad entre nodos y evita las colisiones de paquetes de datos, pero es un protocolo diseñado para redes no densas. Por otro lado, Broadcast es un protocolo simple y eficiente. Además, aprovecha todas las ventajas de la naturaleza en un canal acústico submarino. Este tipo de protocolos MAC es adecuado para

protocolos de encaminamiento oportunistas geográficos.

Existen varios trabajos para UWSN, sin embargo, no se encontró que algún trabajo haya utilizado un simulador de eventos discretos para este tipo de redes. Entre los simuladores encontrados, Aqua-Sim cuenta con unos varios protocolos implementados, tanto en capa MAC, como en la capa de encaminamiento. En cambio, en este trabajo se realizan dos campañas de simulación. En ambas, se utiliza a HH-VBF en la capa de encaminamiento, pero en la capa MAC se utilizan dos protocolos: Broadcast y R-MAC, para hacer la evaluación y determinar qué protocolo es recomendable utilizar.

Al contar con las trazas generadas por el simulador se filtraron los datos de tiempo de retardo. Se utilizó (Matlab) para realizar los cálculos estadísticos de la regresión lineal. Para generar las gráficas fue necesario contar con datos suficientes para que los datos sean confiables. En el caso de Broadcast se tienen 169005 datos generados. Y con R-MAC se generaron 11289 datos.

De acuerdo con la Figura 6.1(b) se puede decir que el comportamiento entre ambos protocolos MAC es el mismo en la mayor parte del tiempo para cualquier nodo. Sin embargo, cuando se hace la calibración como en la Figura 6.3, se puede observar que la pendiente de Broadcast es menor a la del protocolo R-MAC, aunque en los primeros nodos R-MAC se comporta con un menor retardo de propagación para cada nodo. Al ir incrementando los nodos, es más rápido Broadcast. Por lo tanto, para redes dispersas conviene tener un protocolo R-MAC, pero si la red es densa, lo mejor es implementar un protocolo Broadcast.

Al ver una gráfica con mayor número de nodos se puede concluir que entre R-MAC y Broadcast el tiempo de sincronización con pocos nodos puede ser prácticamente igual, pero al ir incrementando, Broadcast empieza a crecer de manera exponencial y R-MAC crece de manera lineal, esto sucede hasta un cierto número de nodos. En la Figura 6.5(b) se puede observar que Broadcast disminuye su pendiente a partir de 40 nodos, después de tener una pendiente con crecimiento exponencial, la pendiente cambia su crecimiento de manera lineal y el crecimiento es aproximadamente de cero, esto se debe a que Broadcast en un principio comienza a inundar la red con mensajes de señalización para tener una comunicación con todos los nodos, al conocer la ubicación y la ruta de cada nodo con Broadcast, posteriormente Broadcast tarda menos tiempo para la sincronización. Sin embargo, R-MAC en todo momento tiene un crecimiento lineal y el número de nodos que tenga va a ser aproximadamente el mismo número de segundo que va a

tardar en sincronizar toda la red.

7.1. Trabajo a futuro

Como un trabajo a futuro queda una vertiente para UWSN densas y dispersas, se propone seguir con esta investigación y realizar un algoritmo de sincronización el cual tenga un protocolo MAC que adapte el protocolo R-MAC para redes dispersas y para redes densas adapte el protocolo Broadcast. Así mismo, adaptar el algoritmo de sincronización en un simulador de software libre (Aqua-Sim en NS-2 o NS-3).

Bibliografía

[1] X. Zhang, J.-H. Cui, S. Das, M. Gerla, and M. Chitre, "Underwater wireless communications and networks: theory and application: Part 1 [Guest Editorial]," IEEE Commun. Mag., vol. 53, no. 11, pp. 40–41, Nov. 2015.

[2] A. A. Syed and J. Heidemann, "Time Synchronization for High Latency Acoustic Networks," in Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications, 2006, pp. 1–12.

[3] N. Chirdchoo, W.-S. Soh, and K. C. Chua, "MU-Sync," in Proceedings of the third ACM international workshop on Wireless network testbeds, experimental evaluation and characterization - WuWNeT '08, 2008, p. 35.

[4] F. Lu, D. Mirza, and C. Schurgers, "D-sync," in Proceedings of the Fifth ACM International Workshop on UnderWater Networks - WUWNet '10, 2010, pp. 1–8.

[5] J. Liu, Z. Wang, M. Zuba, Z. Peng, J.-H. Cui, and S. Zhou, "JSL: Joint time synchronization and localization design with stratification compensation in mobile underwater sensor networks," in 2012 9th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2012, pp. 317–325.

[6] D. Zennaro, B. Tomasi, L. Vangelista, and M. Zorzi, "Light-Sync: A low overhead synchronization algorithm for underwater acoustic networks," in 2012 Oceans - Yeosu, 2012, pp. 1–7.

[7] J. Liu, Z. Zhou, Z. Peng, J.-H. Cui, M. Zuba, and L. Fiondella, "Mobi-Sync: Efficient Time Synchronization for Mobile Underwater Sensor Networks," IEEE Trans. Parallel Distrib. Syst., vol. 24, no. 2, pp. 406–416, Feb. 2013.

[8] A. Vermeij and A. Munafo, "A Robust, Opportunistic Clock Synchronization Algorithm for Ad Hoc Underwater Acoustic Networks," IEEE J. Ocean. Eng., vol. 40, no. 4, pp. 841–852, Oct. 2015.

[9] P. Xie, J.-H. Cui, and L. Lao, "VBF: Vector-Based Forwarding Protocol for Underwater Sensor Networks," 2006, pp. 1216–1221.

[10] P. Xie, Z. Zhou, N. Nicolaou, A. See, J.-H. Cui, and Z. Shi, "Efficient Vector-Based Forwarding for Underwater Sensor Networks," EURASIP J. Wirel. Commun. Netw., vol. 2010, no. 1, p. 195910, 2010.

[11] A. Darehshoorzadeh and A. Boukerche, "Underwater sensor networks: A new challenge for opportunistic routing protocols," IEEE Commun. Mag., vol. 53, no. 11, pp. 98–107, Nov. 2015.

[12] P. Xie and J.-H. Cui, "R-MAC: An Energy-Efficient MAC Protocol for Underwater Sensor Networks," in International Conference on Wireless Algorithms, Systems and Applications (WASA 2007), 2007, pp. 187–198.

[13] Objective modular network testbed in c++ - omnet++. http://www.omnetpp.org/, Enero 2017.

[14] Sanz, PJ, .ᴬn open source tool for simulation and supervision of underwater intervention missions", 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2577-2582, 7-12 Oct. 2012

[15] The Network Simulator - ns-2. Available in: https://www.isi.edu/nsnam/ns/. Consulted on: January, 2017.

[16] Y. F., S. Z. Peng Xie, Zhong Zhou, Zheng Peng, Hai Yan, Tiansi Hu, Jun-Hong Cui, Zhijie Shi, "Aqua-Sim: An NS-2 based simulator for underwater sensor networks," IEEE J. Ocean. Eng., 2009.

[17] T. M. Ian F. Akyildiz, Dario Pompili, "Underwater acoustic sensor networks: research challenges," Ad Hoc Networks, vol. 3, no. 3, pp. 257–279, 2005.

[18] S. Climent, A. Sanchez, J. Capella, N. Meratnia, and J. Serrano, "Underwater Acoustic Wireless Sensor Networks: Advances and Future Trends in Physical, MAC and Routing Layers," Sensors, vol. 14, no. 1, pp. 795–833, Jan. 2014.

[19] J. Heidemann, M. Stojanovic, and M. Zorzi, "Underwater sensor networks: applications, advances and challenges," Philos. Trans. R. Soc. A Math. Phys. Eng. Sci., vol. 370, no. 1958, pp. 158–175, Jan. 2012.

[20] R. B. Darlington and A. F. Hayes, "The Simple Regression Model," in Regression Analysis and Linear Models, D. A. Kenny, Ed. New York: London, 2016, pp. 23-31.

Apéndices

Apéndice A

Parámetros de simulación

Para generar la simulación en Aqua-Sim se declaran los valores: del canal, tipo de propagación, capa MAC, entre otros.

Parámetros de simulación para R-MAC

```
set opt(chan)  Channel/UnderwaterChannel
set opt(prop)  Propagation/UnderwaterPropagation
set opt(netif)  Phy/UnderwaterPhy
set opt(mac)  Mac/UnderwaterMac/RMac
set opt(ifq)  Queue/DropTail/PriQueue
```

Los parámetros del escenario de la red que se quiere simular se determinan por el número de nodos que se tienen, tiempo de simulación y las dimensiones donde se va a trabajar

Parámetros de simulación en R-MAC

```
set opt(ifqlen)                50 ;# paquetes maximos en ifq
set opt(nn)                   40 ;# numero de nodos
set opt(layers)                          1
set opt(x)                   100 ;# dimension de X
set opt(y)                          100  ;# dimension de Y
set opt(z)                               100 ;# dimension en Z
set opt(stop)                  1500 ;# tiempo de la simulacion
```

De acuerdo con los parámetros antes mencionados, se generan los 40 nodos en una distri-

bución aleatoria de la dimensión determinada. Para ejecutar la simulación se realizó un script para generar 100 simulaciones consecutivamente y al terminar entregue un archivo en formato ".tr".

Comando para ejecutar la simulación

```
#!/bin/bash
# -*- ENCODING: UTF-8 -*-
for (( i = 1; i < 101; i++ )); do
 ns-aqua  ruta_de_la_simulacion/nombre.tcl
done
exit 1
```

Realizando 100 simulaciones para poder obtener un vector promedio de todos los tiempos que se tarda cada nodo en enviar un paquete. Las trazas obtenidas con el simulador NS-2 se realizaron con el comando **"$ns_ use-newtrace"**. Las trazas obtenidas en formato ".tr" son de la siguiente manera.

Traza.tr

```
vectorbased
M 0.0 nn 40 x 100 y 100 z 100
M 0.0 prop Propagation/UnderwaterPropagation ant Antenna/OmniAntenna
s -t 0.015119549 -Hs 15 -Hd -1 -Ni 15 -Nx 38.72 -Ny 68.67 -Nz 60.12 -Ne
   10000.000000 -Nl MAC -Nw --- -Ma 0 -Md 0 -Ms 0 -Mt 0 -Is 0.0 -Id 0.0
   -It UnderwaterRmac -Il 40 -If 0 -Ii 0 -Iv 0
s -t 0.037951005 -Hs 36 -Hd -1 -Ni 36 -Nx 45.01 -Ny 48.64 -Nz 68.60 -Ne
   9999.997802 -Nl MAC -Nw --- -Ma 0 -Md 0 -Ms 0 -Mt 0 -Is 0.0 -Id 0.0
   -It UnderwaterRmac -Il 40 -If 0 -Ii 0 -Iv 0
s -t 0.038371615 -Hs 38 -Hd -1 -Ni 38 -Nx 27.57 -Ny 22.57 -Nz 29.50 -Ne
   10000.000000 -Nl MAC -Nw --- -Ma 0 -Md 0 -Ms 0 -Mt 0 -Is 0.0 -Id 0.0
   -It UnderwaterRmac -Il 40 -If 0 -Ii 0 -Iv 0
s -t 0.052737033 -Hs 39 -Hd -1 -Ni 39 -Nx 26.10 -Ny 93.81 -Nz 52.42 -Ne
   9999.997758 -Nl MAC -Nw --- -Ma 0 -Md 0 -Ms 0 -Mt 0 -Is 0.0 -Id 0.0
   -It UnderwaterRmac -Il 40 -If 0 -Ii 0 -Iv 0
```

Apéndice B

Descripción de trazas

Posteriormente de generar las trazas, se realizó un filtro de las trazas respecto a los valores que nos interesa para esta investigación. Es importante mencionar la manera en la que nuestras trazas se clasificaron:

- La primera columna muestra el Tipo de Evento
  - s = Enviar (send)
  - r = Recibir (receive)
  - d = Soltar (drop)
  - f = Adelante (forward)
- En la segunda columna tenemos a "t" lo que nos dice que la tercera columna se tiene el tiempo que tardó en llegar la información al nodo destino
- En la cuarta columna tenemos la información "-Hs", significa que en la columna cinco está el nodo fuente
- Para la sexta columna ahora tenemos "-Hd" y esto quiere decir que para la séptima columna tendremos el nodo destino (boya de superficie)

Para la realización de la filtración de datos se tomó en cuenta la primera columna y que tuviera una **r** que son todos los paquetes recibidos por el nodo receptor, posteriormente la información tomada de las trazas fue "-Hs" y el tiempo que se encuentra en la tercera columna.

Apéndice C

Regresión lineal

Para realizar los cálculos estadísticos necesarios de la regresión lineal, se utiliza Matlab. En este se realizan todos los cálculos correspondientes y sólo se utiliza un archivo de entrada (con el nombre de "filtro.txt"), en él se encuentran todos los datos necesarios (número de nodo y tiempo) de forma filtrada de los archivos ".tr" como se muestra a continuación.

Regresión lineal

```
1  %% Se acomoda la matriz 'filtro' de manera ascendente respecto a los
       nodos
2  for h=1:40
3      repet(h,1)=numel(find(reline==h));%obtenemos una matriz en la cual
          estan el numero de repeticiones de cada nodo
4  end
5
6  %% Se realiza el promedio de cada nodo transmitido
7  for q=1:40
8      count=count+repet(q,1);
9      init=count-repet(q,1)+1;
10     reline(init:count,3)=sum(reline(init:count,1))/repet(q,1);%En esta
          funcion se obtiene el promedio del tiempo de cada nodo
11     prom(q,1)=(sum(reline(init:count,1)))/(repet(q,1));
12 end
13
```

```
14 %% Se obtiene la media de los nodos y del tiempo
15 media_x=sum(reline(1:m,1))/m;
16 media_y=sum(reline(1:m,2))/m;
17
18 %% Se crea una matriz para poder programar los datos de la regresion
      lineal
19 for i=1:m
20     rline(i,1)=reline(i,1);%tiempo de transmision
21     rline(i,2)=reline(i,2);%numero del nodo
22     rline(i,3)=reline(i,1)-media_x;%Desviacion de X
23     rline(i,4)=reline(i,2)-media_y;%Desviacion de Y
24     rline(i,5)=rline(i,3)*rline(i,4);%Producto cruzado
25 end
26
27 %% Se programan todos los datos necesarios para la regresion lineal
28 co_var=sum(rline(1:m,5))/m;
29 var_x=sum(rline(1:m,3).^2)/m;
30 var_y=sum(rline(1:m,4).^2)/m;
31 s_x=sqrt(var_x);
32 s_y=sqrt(var_y);
33 r_xy=co_var/(s_x*s_y);
34 b1=co_var/var_x;
35 b0=media_y-(b1*media_x);
36
37 %% Se realiza la regresion lineal
38 for q=1:m
39     salida(q,2)=b0+(b1*reline(q,1));
40     salida(q,1)=reline(q,2);
41 end
```

Casa abierta al tiempo
UNIVERSIDAD AUTÓNOMA METROPOLITANA

ACTA DE EXAMEN DE GRADO

No. 00066
Matrícula: 2153805328

SINCRONIZACIÓN DE TIEMPO EN REDES SUBMARINAS DE SENSORES INALÁMBRICOS

En la Ciudad de México, se presentaron a las 10:30 horas del día 22 del mes de noviembre del año 2017 en la Unidad Iztapalapa de la Universidad Autónoma Metropolitana, los suscritos miembros del jurado:

DR. ENRIQUE STEVENS NAVARRO
DR. VICTOR MANUEL RAMOS RAMOS
DRA. GRACIELA ROMAN ALONSO

Bajo la Presidencia del primero y con carácter de Secretaria la última, se reunieron para proceder al Examen de Grado cuya denominación aparece al margen, para la obtención del grado de:

MAESTRO EN CIENCIAS (CIENCIAS Y TECNOLOGIAS DE LA INFORMACION)

DE: SERGIO ARTURO ELIZALDE HERNANDEZ

y de acuerdo con el artículo 78 fracción III del Reglamento de Estudios Superiores de la Universidad Autónoma Metropolitana, los miembros del jurado resolvieron:

APROBAR

Acto continuo, el presidente del jurado comunicó al interesado el resultado de la evaluación y, en caso aprobatorio, le fue tomada la protesta.

SERGIO ARTURO ELIZALDE HERNANDEZ
ALUMNO

REVISÓ
LIC. JULIO CESAR DE LARA ISASSI
DIRECTOR DE SISTEMAS ESCOLARES

DIRECTOR DE LA DIVISIÓN DE CBI
DR. JOSE GILBERTO CORDOBA HERRERA

PRESIDENTE
DR. ENRIQUE STEVENS NAVARRO

VOCAL
DR. VICTOR MANUEL RAMOS RAMOS

SECRETARIA
DRA. GRACIELA ROMAN ALONSO