Asociación Argentina
de Mecánica Computacional

Mecánica Computacional Vol XXVI, pp.962-973
Sergio A. Elaskar, Elvio A. Pilotta, Germán A. Torres (Eds.)
Córdoba, Argentina, Octubre 2007

ALGORITMOS GEOMETRICOS PARA LA SIMULACION DE UN RADAR DE AERONAVE

Juan P.D'Amato[a,c], Cristian García Bauza[a,b], Matias Truchi[a], Marcelo Venere[a,d]

aPLADEMA, Universidad Nacional del Centro, 7000 Tandil, Argentina,
({jpdamato,crgarcia,venerem }@exa.unicen.edu.ar) , http://www.plademа.net
b Comisión de Investigaciones Científicas de la Provincia de Buenos Aires
c Consejo Nacional de Investigaciones Científicas
d CNEA

Keywords: Realidad Virtual, Simulación Computacional, Algoritmos Geométricos, Computación Gráfica.

Abstract. En el presente trabajo se propone un método para simular en tiempo real un radar ubicado en una aeronave. En particular, se exponen los algoritmos implementados para recrear la visualización de una consola radar, dada su posición en un entorno tridimensional con topografía y distintos objetos presentes.
Tanto la topografía como los objetos son descriptos mediante polígonos y el modelado del lóbulo de emisión se realiza discretizando el mismo en un conjunto de rayos, por lo que la imagen se genera evaluando las intersecciones de estos rayos con los polígonos presentes en el escenario. Para ello, se utiliza un algoritmo basado en raycasting, con una cantidad de 150 rayos en el eje vertical y 1080 en la circunferencia, por lo cual el algoritmo debe ser optimizado para lograr trabajar en tiempo real a un ritmo de 15 revoluciones por minuto.
Las optimizaciones realizadas están basadas en simplificación poligonal de los objetos en el escenario y clasificación geométrica para resolver rápidamente las intersecciones.
El modelo recibe además parámetros de entrada como el alcance, la ganancia y el eje de barrido del radar desde la consola; visualizando luego los resultados en la pantalla del puesto alumno, logrando representaciones realistas en tiempos aceptables.

1 INTRODUCCIÓN

Hoy en día el uso de simuladores para entrenar pilotos de todo tipo de vehículos es una práctica corriente y de hecho para cada área existen numerosos productos con distinto grado de realismo (Flight Simulator en aviación, TRANSAS Navi-Trainer en embarcaciones, etc.). Es el esfuerzo en lograr este realismo, tanto en la visual como en los instrumentos y comportamiento del vehículo, lo que determina la calidad de los distintos simuladores. Tanto en embarcaciones como aeronaves, uno de los instrumentos más importantes que debe ser simulado es el radar.

El principio de funcionamiento de un radar (ver Coock et. al (1994) o Meikle (2001)) se basa en la escucha de los ecos generados por la emisión de un corto pulso electromagnético. Este pulso debe ser colimado de forma que los ecos provengan mayoritariamente de la dirección que se está observando, sin embargo los pulsos que se logran en la realidad tienen el aspecto que se muestra en la Figura 1, donde se observa un lóbulo principal con un cierto ancho en la dirección deseada y una serie de lóbulos laterales. La calidad del radar estará asociada a cuan pequeños son los lóbulos laterales y cuan estrecho es el lóbulo principal, ya que los blancos que se encuentren en las zonas alcanzadas por estos lóbulos podrán generar ecos y es claro que aquellos que no provengan de la dirección principal serán interpretados erróneamente.

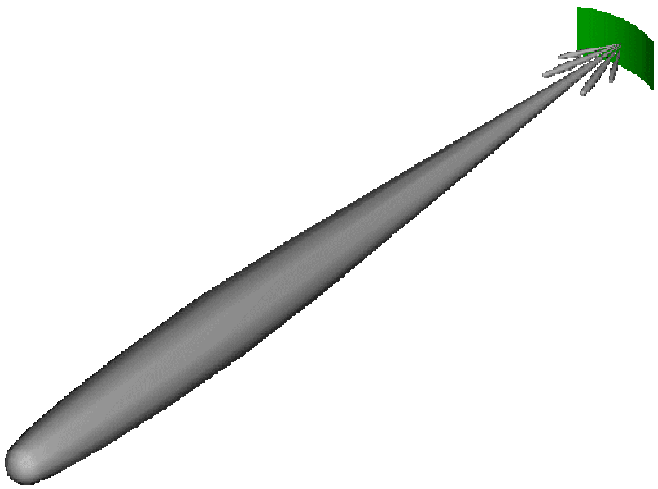

Figura 1: Esquema de un pulso del radar. La zona gris representa la región donde un blanco generará un eco

En el caso de radares en barcos, este pulso es emitido horizontalmente en distintas direcciones, girando la antena a razón de 14 a 20 revoluciones por minuto, con aproximadamente 4000 procesos emisiones-escuchas cada 360 grados (ver Otheguy et. al (2002)). Cuando se trata de radares en aeronaves, el barrido puede tener una orientación o "tilt" (ver Figura 2) y en general se trata de instrumentos más pequeños, con lo cual el ancho de lóbulo es mucho mayor. También es posible operarlo barriendo un determinado ángulo y no realizando giros completos. En general un radar de este tipo realiza 1000 procesos de emisión-escucha cada 360 grados y trabaja entre 10 y 15 revoluciones por minuto.

La discretización natural de este problema puede realizarse considerando al lóbulo como un haz de rayos y a los blancos como un conjunto de polígonos en un espacio tridimensional, tal como se representa en la Figura 3. Un proceso de emisión-escucha se transforma entonces

en encontrar las intersecciones de este haz de rayos con todos los polígonos, graficando luego en la pantalla del radar la primera intersección para cada rayo. El costo computacional para este algoritmo seria $O(N_{pol}.N_{ray})$, donde

N_{pol}: Número de polígonos empleados para describir el escenario y los blancos
N_{ray}: Número de rayos utilizados para discretizar el lóbulo.

Este costo es para cada proceso emisión-escucha. Si se considera un radar trabajando a 15 revoluciones por minuto y con 1000 emisiones por revolución y que una discretización mínima del lóbulo requerirá más de 100 rayos en azimut, la conclusión es que resulta difícil trabajar con escenarios con 104 o 105 polígonos (basados en equipos PC modernos).

En este trabajo explicamos primero como se realiza esta simulación para luego discutir las alternativas analizadas para lograr modelar este problema a un costo computacional aceptable para una aplicación de tiempo real.

2 MODELADO DE LOS DATOS

A diferencia de los radares navales y la mayoría de los terrestres, un radar de aeronave considera también la altura en la que se encuentra; razón por la cual los datos del entorno deben ser necesariamente modelados en el espacio tridimensional.

El principio de funcionamiento de un radar de aeronave consiste en la emisión de pulsos electromagnéticos que conforman un lóbulo o haz de barrido y la escucha del eco correspondiente luego del rebote contra la superficie terrestre y blancos eventuales que se encuentren dentro del sector de detección.

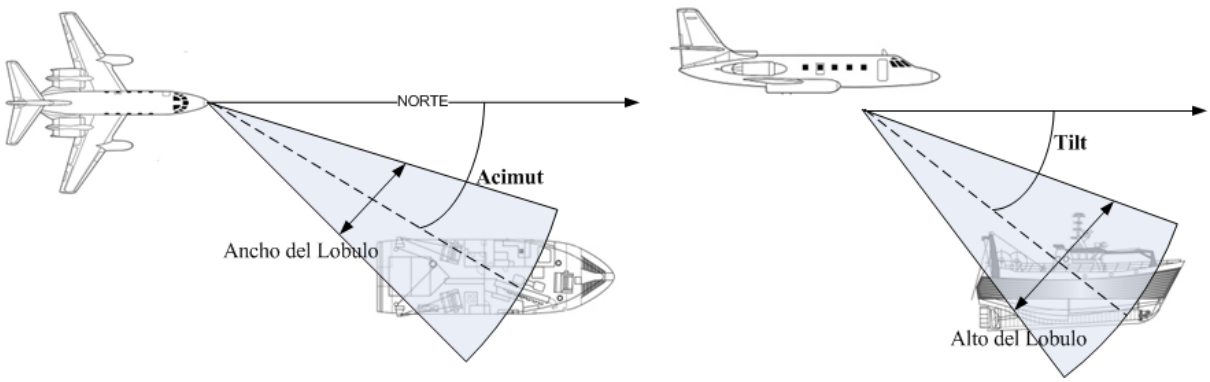

Figura 2: Vista en Planta (Plano XY, Acimut) y Vista Lateral (Plano XZ, Tilt) del lóbulo de un radar aéreo.

Como se muestra en la Figura 2, el haz de barrido gira en dirección de acimut y tiene una inclinación con respecto al horizonte, denominado tilt.

El tiempo que demora el eco en retornar se utiliza para inferir la distancia hasta el objeto que originó el eco. La intensidad del eco retornado es proporcional a las propiedades refractarias del blanco. Si el elemento es muy blando (por.ej: arena) la señal retornada es muy suave, y si es dura (rocas, metales) la señal es más intensa.

El contorno de la costa, al igual que los contactos, son susceptibles de ser detectados por el radar, pero a diferencia de éstos, la costa se considera inmóvil, motivo por el cuál no se

trasmite en la tabla de puntos sino que es representado por un MDE (Modelo Digital de Elevación).

2.1 Representación del Lóbulo

Para simular el comportamiento del haz de barrido, se utiliza un algoritmo simplificado de Ray-Casting. Como veremos en las secciones siguientes, se introdujeron cambios en este algoritmo, principalmente en las estructuras utilizadas y la clasificación geométrica de los objetos del escenario.

El lóbulo es discretizado en el plano XZ (tilt) en un conjunto de rayos, con paso $\Delta\beta$ constante. La Figura 3 muestra el lóbulo y como sería su discretización en rayos.

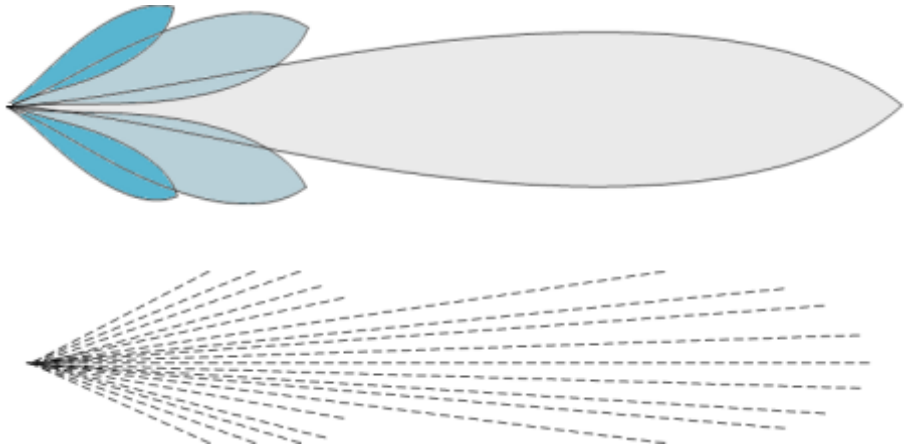

Figura 3: Arriba: Forma del lóbulo. Abajo: discretización utilizada en el algoritmo

Es importante determinar correctamente el paso de la orientación del rayo; si se elige un $\Delta\beta$ muy grande, puede ocurrir que existan polígonos pertenecientes a posibles blancos que nunca sean intersectados por el rayo.

Sin embargo tampoco se puede elegir este ángulo muy pequeño porque equivaldría a mayor tiempo de cálculo al tener que considerar más rayos. Los valores adecuados tanto para este ángulo como para el de giro $\Delta\alpha$ que garantizan no perder blancos detectables se ha calculado de la manera presentada en Cifuentes et al. (2006). La cantidad de rayos resultante para la discretización es de 1080 (tres rayos por grado) en el plano de acimut y de 150 en el plano de tilt.

2.2 Representación de elementos de la escena

Actualmente los MDE son obtenidos de una base de datos del Programa de colaboración USGS EROS/NASA SRTM (*Shuttle Radar Topography Mission*). Básicamente son archivos binarios, con valores de alturas, representando zonas de 1 grado por 1 grado de longitud. La resolución de estos MDE es de 90 metros por píxel. No contienen cabecera, ni archivos adicionales de configuración.

Al utilizar MDE de 1 grado por 1 grado, surgió la necesidad de unir MDEs de regiones para formar los MDE finales utilizados en los ejercicios de entrenamiento. Este problema se solucionó realizando una búsqueda local entre los archivos de los SRTM que concuerden con

los límites de la zona editada, posteriormente se forma el MDE definitivo en base a uniones de sus valores y se recortan los bordes, siguiendo un esquema de edición como el presentado en Vénere et al. (2005). Existe una fase posterior donde se aplican filtros para eliminar ruidos generados en los MDE por la refracción emitida por zonas de agua.

El MDE final es modificado por un algoritmo de simplificación basado en Quadtree presentado en Cifuentes et al. (2004). Esta simplificación es realizada offline previamente al momento de crear un nuevo ejercicio.

Las mallas obtenidas luego de aplicar el algoritmo de simplificación están compuestas por un número de polígonos que oscila entre los 7.000 y 12.000 triángulos. Para representar cada posible blanco en la escena (buques, boyas, submarinos) se utilizan objetos tridimensionales con una cantidad de triángulos que oscila entre los 50 y 100. Una escena total puede estar compuesta de 50.000 triángulos o más por lo que es necesario estructurar estos datos en una jerarquía que permita resolver eficientemente el algoritmo de intersección del rayo con los objetos.

3 ALGORITMO DE SIMULACION DEL RADAR

Como se nombró anteriormente, la técnica implementada es similar a un algoritmo de Ray-Casting o Búsqueda de Rayo. El núcleo de la técnica es el cálculo de intersecciones de elementos del espacio, triángulos en nuestro caso, con rayos salientes desde la posición del radar. Para cada rayo, se almacena el punto de intersección más cercano al radar, mientras que los restantes se omiten. Finalmente los triángulos almacenados se proyectan a la pantalla.

Cada rayo se representa por la siguiente información:

- origen: posición de la antena del vehículo en el espacio.
- ángulo de tilt: componente que indica la dirección del rayo respecto al horizonte del vehículo.
- ángulo de acimut: componente que indica la dirección del ángulo respecto al norte absoluto.

Todo elemento tiene asociado un rango visible (ecuación 1), denominado “horizonte de radar”, el cual indica la máxima distancia desde la que el blanco será alcanzado por el radar y cumple con:

$$HR = factor * \sqrt{Altura_{Emisor} + Altura_{Receptor}} \leq A_R \qquad (1)$$

Donde el factor es constante y conocido para cada tipo de radar simulado y A_R es el rango de alcance del radar. Los elementos que no cumplen esta ecuación son descartados previamente y no son tenidos en cuenta en el algoritmo de intersecciones del raycasting.

Para lograr una representación real del lóbulo, se debe emitir gran cantidad de rayos; teniendo que resolver millones de intersecciones en el espacio. Entonces, se hace indispensable contar con una estructura eficiente para la clasificación y búsqueda de los elementos del escenario.

Esta estructura debe presentar un tiempo de actualización despreciable, dada la dinámica

del escenario, donde todos los elementos, con respecto al radar, se mueven.

3.1 Clasificación basada en Árbol

La primera metodología evaluada para la clasificación poligonal de elementos se basa en subdividir en regiones el espacio de búsqueda del radar. Cada región es dividida a su vez en ocho, mientras se cumpla con un criterio de tolerancia, generando una jerarquía de nodos que conforman un *octree* (árbol octonario), la que almacena y permite obtener aquelllos polígonos coincidentes al recorrido del rayo.

El criterio de tolerancia utilizado es la concentración de información por unidad de volumen (en este caso, cantidad de triángulos). El tiempo computacional de búsqueda de elementos utilizando esta estructura es logarítmico, sumando el tiempo de cálculo de la intersección del rayo con un elemento; siendo el tiempo total de evaluación de un rayo de la forma descripta en la ecuación 2.

$$T_T \; ; \; Log\left(\frac{N}{C_N}\right) + \frac{1}{2} C_N * T_I \tag{2}$$

Donde C_N es el factor de división del Octree, es decir, el número de triángulos que contiene un nodo y T_I es el tiempo de calcular una intersección entre el rayo y un triángulo. En promedio, el rayo será comparado con la mitad de los triángulos que contiene un nodo.

Con la solución del Octree, los primeros resultados obtenidos arrojan tiempos de procesamiento muy altos, no aceptables para la simulación. Por ejemplo, para una malla de 5000 polígonos, con 41 rayos a lo ancho del tilt y 360 de acimut, el tiempo de cálculo fue de 16 segundos, debiendo ser de 4 segundos para cumplir con el requerimiento de 15 revoluciones por minuto.

Adicionalmente, esta estructura presenta una complejidad extra debido a que el tiempo de construcción del árbol es inaceptable. Por estas razones, esta estrategia fue totalmente descartada.

3.2 Clasificación basada en superficie esférica con estructura dinámica

La estrategia propuesta para la clasificación utiliza una segmentación del espacio basada en la proyección de los elementos sobre una superficie esférica, siguiendo el patrón de recorrido que realiza el algoritmo del rayo.

Este algoritmo de clasificación, recorre todos los polígonos de la escena, calcula el volumen mínimo que contiene a cada triángulo (también denominado “bounded box”) y los ángulos máximos y mínimos de tilt y acimut de dicho volumen respecto a la posición del radar, tal como se muestra en la (Figura 4).

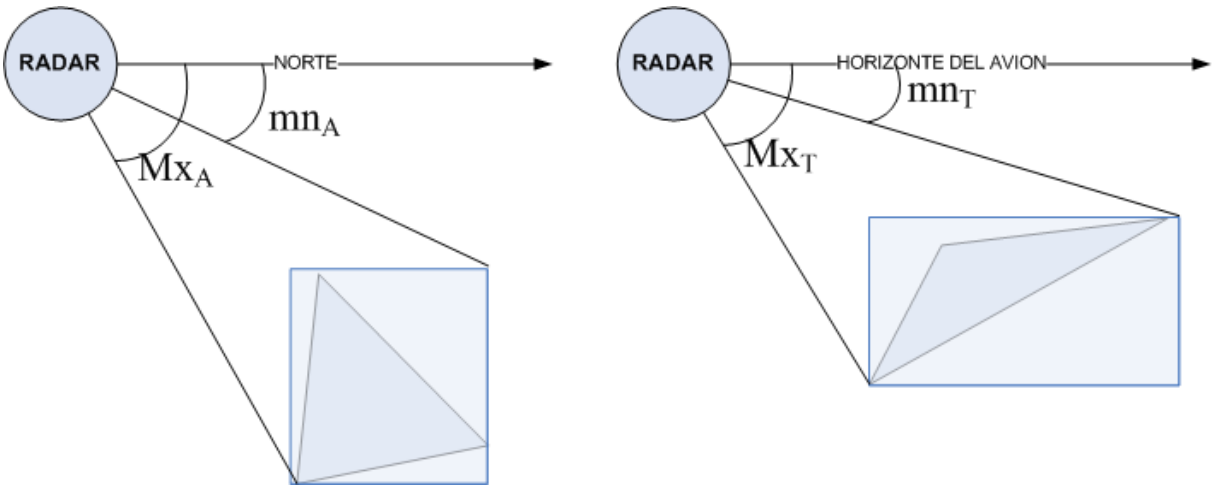

Figura 4: Vista en Planta (Acimut) y Vista Lateral (Tilt) para el Bounded Box de un triángulo.

Por cada triángulo se almacena una referencia al mismo en una estructura de matriz indexada por los valores de acimut y tilt, pudiendo un triángulo abarcar más de una celda. Por ejemplo un triángulo que se encuentra en el rango de tilt 16-18º, y acimut 345º, será almacenado en las celdas [16][345], [17][345] y [18][345]. Cada triángulo es almacenado en la estructura únicamente si cumple la ecuación de horizonte radar (Ver ecuación 1).

Cada celda de la matriz entonces, contiene referencias a aquellos triángulos con los que un rayo de determinado tilt y acimut se superpone (Ver Figura 5).

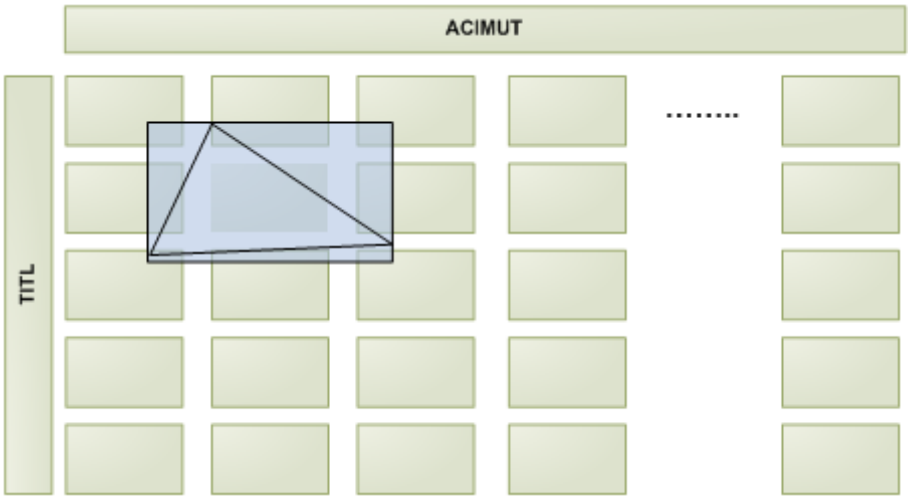

Figura 5: Esquema del Almacenamiento del triángulo

En resumen, los pasos seguidos para la actualización de la estructura son los siguientes:

- Definir la discretización de la matriz.
- Borrar estado actual de la matriz
- Para cada polígono de la escena
  - obtener el bounded box del polígono.
  - calcular el ángulo de azimut mínimo y máximo, ángulo de tilt mínimo y máximo del bounded box con respecto a la posición del radar.

- añadir la referencia del polígono dentro de la matriz en todo el rango de valores de la combinación de los grados de tilt y acimut.

El tiempo de búsqueda en esta nueva estructura, es constante. Cada rayo accede directamente a los elementos de la matriz que se encuentren en su dirección. El tiempo total de evaluación de un rayo ahora está dado por la ecuación 3,

$$T_T \; ; \; K_B + \frac{1}{2}\left(\frac{N}{C_A + C_T} \right) * T_I \tag{3}$$

Donde K_B es el tiempo de búsqueda en la estructura, C_A es la discretización del espacio en Azimut y C_T con respecto al Tilt. La cantidad de celdas en acimut es de 360 (barrido de acimut entre 0º y 360º con paso de 1°), y la de tilt es de 520 (barrido de *tilt* entre -26º y 26º sobre el horizonte con un paso de 0,1°).

Si la distribución de elementos en el espacio es uniforme, la cantidad de triángulos que se encontraran en cada celda es la misma en todas ellas, y el tiempo T_T es prácticamente constante. Esta última deducción puede ser considerada como una virtud de la propuesta presentada. En sistemas de tiempo real, contar con un algoritmo que tenga un costo constante permite predecir su comportamiento en el tiempo y lo hace totalmente estable.

3.3 Clasificación basada en superficie esférica con estructura estática

Una mejora adicional se ha obtenido al convertir la estructura dinámica de la matriz, la cual se borra y recalcula constantemente, en una estructura estática. La idea es no borrar toda la matriz, sino utilizar contadores por cada celda, que indican cuál es el último triángulo activo de la lista.

Supongamos por ejemplo que en el instante T una celda contiene 3 referencias a triángulos, en este caso el contador es igual a 3 (los tres triángulos están activos). En el instante T+1, uno de estos triángulos debe ser borrado porque ya no cae dentro de la celda, en lugar de borrar toda la estructura y luego agregar los triángulos que corresponden, se sobrescribe la información que permanece del tiempo T y se actualiza el contador a 2.

Cada cierto intervalo de tiempo, si la estructura crece en un determinado factor, es borrada completamente para evitar sobrecarga de memoria.

3.4 Esquema de actualización y lectura de la estructura

Como hemos nombrado anteriormente, el algoritmo requiere actualizar el estado de la simulación por cada segundo, para reflejar el movimiento del radar de avión y a su vez, resolver las intersecciones del rayo. Estas operaciones se realizan en dos hilos de ejecución concurrentes.

Para permitir la modificación y lectura simultánea de datos se utiliza un esquema de doble matriz o buffer. Una de las matrices es de solo lectura, para ser accedida por el módulo de cálculo de intersecciones, denominada matriz Activa. Sobre una segunda matriz temporal se realiza la re-clasificación de los polígonos. Al momento de haber finalizado la clasificación,

se realiza un intercambio de matrices, la matriz activa se convierte en la matriz temporal y viceversa. Dada la frecuencia de actualización, este intercambio es transparente al usuario.

3.5 Visualización

Un aspecto importante del radar es la visualización de la información en pantalla. Cada elemento del escenario, tiene asociado información del material que lo compone (metal, agua, terreno, etc.). En caso que el triángulo haya sido intersecado, se lee esta información y se asigna un valor de intensidad de eco de acuerdo al tipo de material leído.

Cada eco recibido por el radar, se proyecta a un punto en pantalla. La intensidad del eco se encuentra asociado a un color. Al momento de visualizar, se realiza una conversión de intensidades mediante interpolación lineal a una tabla de colores definidos para cada tipo de radar. En general, intensidades fuertes, que equivalen a terrenos duros o cascos metálicos de barcos se muestran con tonos rojizos, valores intermedios de intensidad, son azules y valores bajos como el ruido de mar son colores amarillos. Algunos resultados utilizando este esquema pueden verse en la Figura 8

4 EVALUACIÓN DEL ALGORITMO

Para evaluar la confiabilidad de la técnica propuesta, se realizaron un conjunto de comparativas de ambas técnicas sobre escenarios similares, compuestos por un mapa de elevación y un conjunto de vehículos dispersos.

Se tomó el tiempo total de simulación y visualización de una vuelta completa del radar; comparando la estructura de *Octree*, la estructura de clasificación con actualización dinámica y una tercera prueba, de la misma estructura con la mejora de actualización estática.

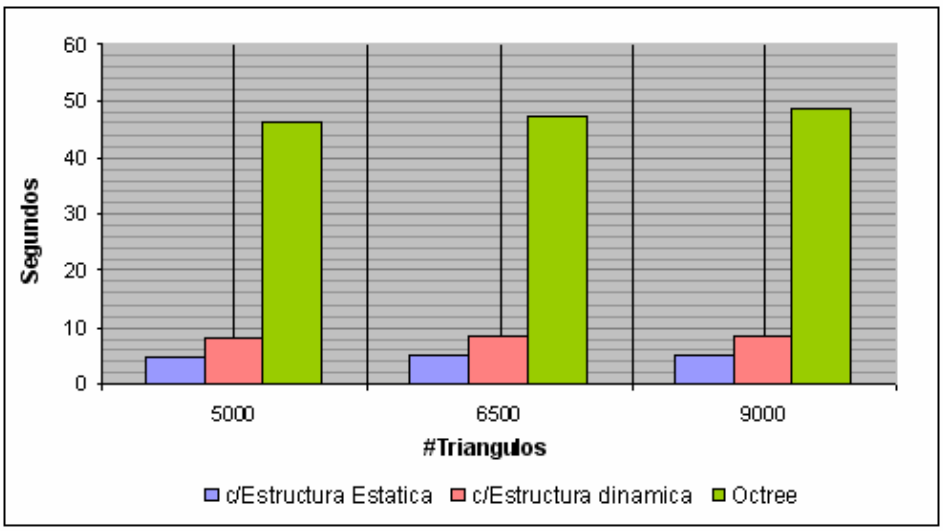

Figura 6: Comparativa de tiempo total entre las tres técnicas

Adicionalmente, se realizaron pruebas exhaustivas sobre la nueva técnica, evaluando escenarios con distintas cantidades de triángulos. Principalmente se realizaron pruebas de stress, trabajando con escenarios de la peor situación posible para la simulación, en donde el radar se encuentra rodeado completamente de terreno (si recordamos, este tipo de radar opera mayormente en mar abierto).

El gráfico mostrado en la Figura 7 resume el tiempo total y el de actualización de la estructura (indicado como Tiempo Locate).

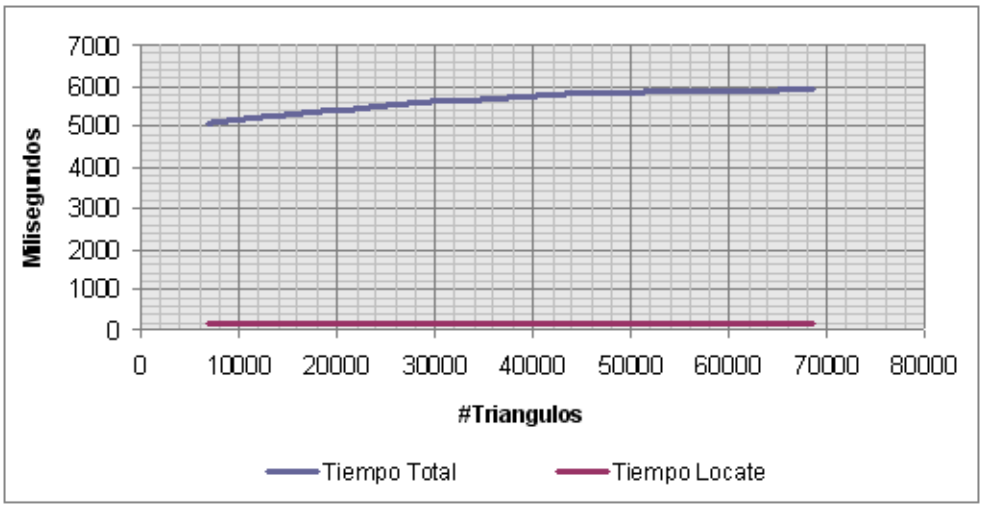

Figura 7: Tiempo de clasificación y tiempo total por número de triángulos

Si analizamos el gráfico, observamos que para la situación planteada (la peor posible a nivel de intersecciones a resolver) el radar trabaja entre 10 y 12 revoluciones por minuto, además el tiempo de clasificación es despreciable. Los resultados se encuentran dentro de lo requerido para la simulación.

5 CONCLUSIONES

El algoritmo propuesto como parte de la implementación del módulo radar ha brindado buenos resultados y ha permitido alcanzar los tiempos especificados. A su vez, las imágenes generadas en tiempo real de escenarios de entrenamiento complejos, son ante la vista de los expertos del área semejantes a las reales. La técnica además ha mostrado ser robusta, y temporalmente precisa al ser ejecutada en procesadores de múltiple núcleo. Como futuras líneas de trabajo, se propone implementar parte del procesamiento directamente en placas gráficas, utilizando los últimos avances de esta área de hardware y extender el modelo de simulación, incorporando nuevas variables como por ejemplo la canalización por temperatura.

6 IMÁGENES OBTENIDAS

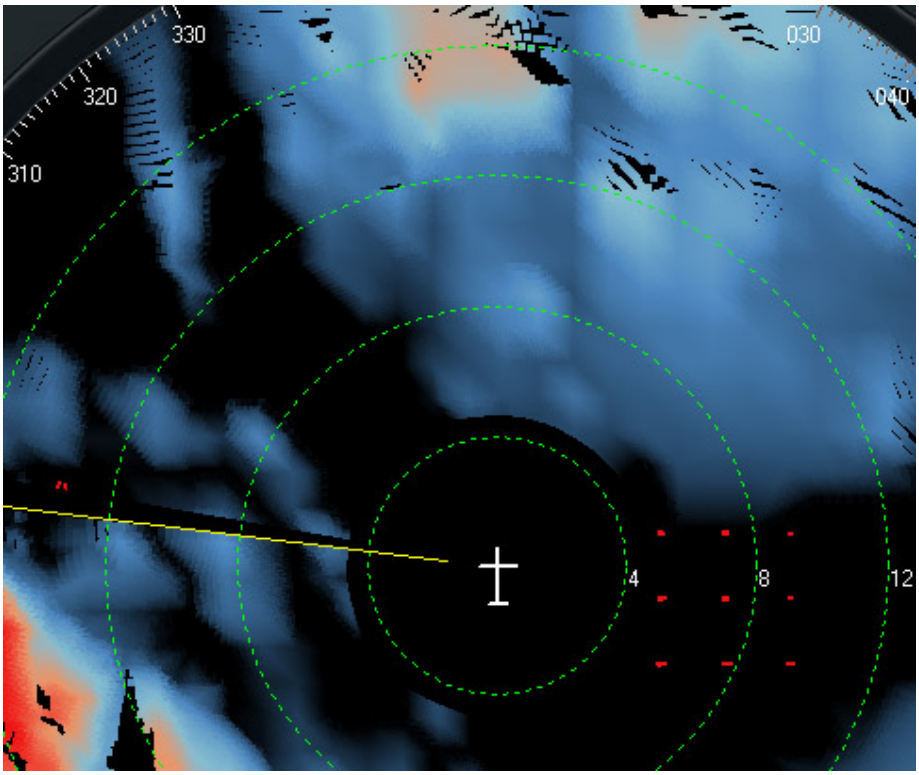

Figura 8: Visualización ampliada de terreno y vehículos (puntos rojos) utilizando el esquema de colores.

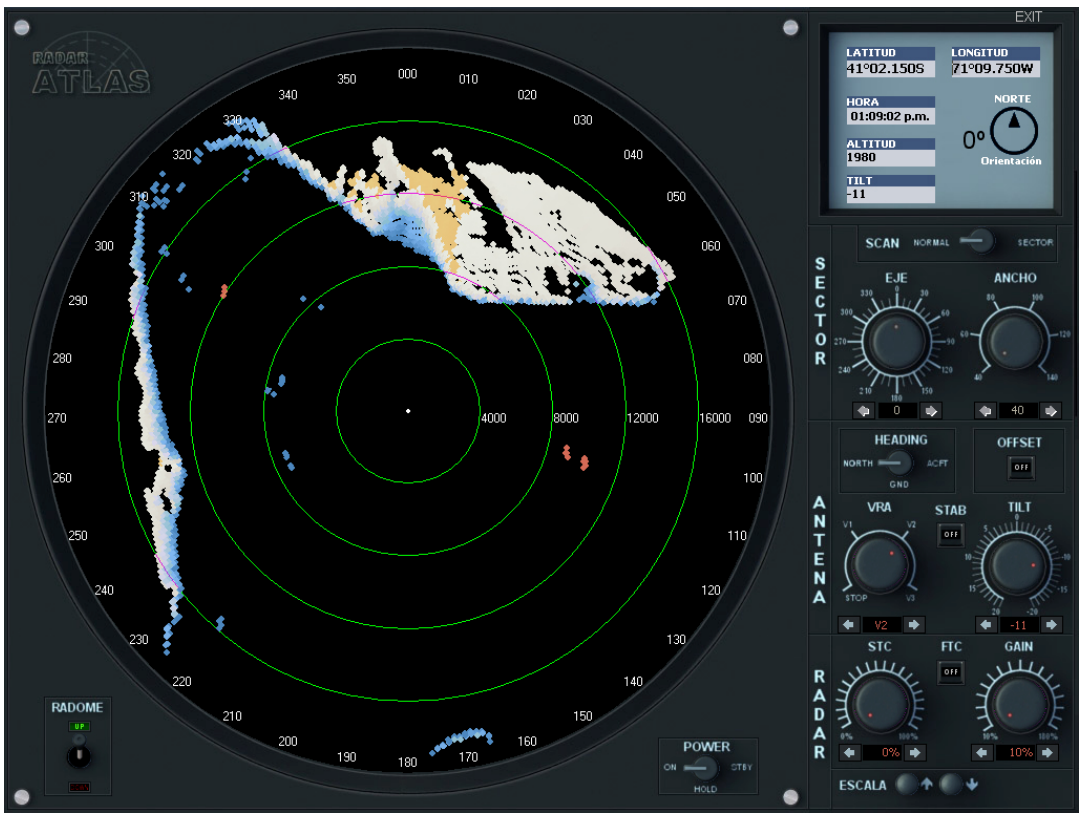

Figura 9: Vista completa del módulo Radar implementado

REFERENCIAS

Cifuentes, M.V., Vénere, M., Clausse, A. "Un algoritmo para la simplificación poligonal de modelos topográficos digitales". *33ª JAIIO, Jornadas Argentinas de Informática e Investigación Operativa.* 2004.

Cifuentes M.V., D' Amato J.P., García Bauza C., Lotito P., Vénere M., Clausse C., "Ray Casting para la Definición de Zonas de Interés en Simplificación Topográfica". *Mecánica Computacional*, XXV: 1177-1186, 2006.

Coock Ch. and M. Bernfeld, "Radar Signals: An Introduction to Theory and Applications", *Artech House Incorporated*, ISBN 0890067333, 1994.

Meikle H., "Modern Radar System", *Artech House Incorporated*, ISBN 1580532942. 2001

Otheguy I., Soriano M., Boroni G., Vénere M., "Simulación en tiempo real de un radar de barrido horizontal". *Mecánica Computacional*, Vol. XXI, pp. 1203-1212, 2002.

Vénere M., Cifuentes M.V., D'Amato J.P., García Bauza C., "Editor de Escenarios para Aplicaciones de Realidad Virtual". *VI Simposio Argentino de Tecnología en Computación*, 2005.

Vénere M., Boroni G., "Un simulador distribuido para entrenamiento de operarios". *VIII Congreso Argentino de Ciencias de la Computación, CACIC*, 2002.