GPU Ray Tracing-based Method for Real-time ISAR simulation

Taieb Lamine Ben Cheikh
École Polytechnique de Montréal
Montréal, Canada
taieb.lamine-ibnecheikh@polymtl.ca

Pascal Guillemette
CAE Inc.
Montréal, Canada
pascal.guillemette@cae.com

ABSTRACT

The Inverse Synthetic Aperture Radar (ISAR) technique serves to identify and/or classify targets such as aircraft, ships and ground vehicles. In the context of radar operator training, ISAR simulation is a complex CPU-intensive process since it needs to compute and update a target's range-Doppler signature for constantly varying aspect angles. The main challenge is to generate a reliable ISAR image in real-time. To do so, current simulators employ several approximations. These commonly consist of (1) computing only direct reflections and tuning manually some reflection features by editing the 3D models, (2) coarsening the target mesh to reduce the number of intersected polygons or (3) undersampling the integration time by ignoring intermediate aspect angles.

In this work, we present a solution where we implement a modified visual ray tracing method as an analogy to simulate the radar wave scattering. The objective is to improve real-time simulation of the ISAR imagery for the purpose of radar operator training. The method presented here has the ability to compute multiple reflections. Consequently, intense flashes produced by corner reflectors are naturally depicted. The method also allows using complex 3D models directly without offline preprocessing or manual modifications of the models. Real time is achieved by implementing a ray tracing-based algorithm written in CUDA running on GPU. This takes advantage of the massive number of parallel threads that can run on current GPUs. Moreover, the general-purpose programming model supported by CUDA offers a more flexible implementation with more appropriate data structures.

The proposed solution can generate realistic ISAR images of a ship at sea including effects of multiple reflections on a mid-range graphics card. Since this solution does not require preprocessing or manual alteration of the 3D models to add scatterers, it makes training on realistic ISAR simulation more accessible.

ABOUT THE AUTHORS

Taieb Lamine, Ben Cheikh is a Postdoctoral fellow at École Polytechnique de Montréal. Currently, he is working on a MITACS project on the acceleration of CAE Radar simulator on modern high performance computers. He received his Ph.D. degree in Computer Engineering from École Polytechnique de Montréal in 2015. He has formerly received his B.Eng. and M.Sc. degrees in Electrical Engineering from École Nationale d'Ingénieurs de Sfax respectively in 2005 and 2006.

Pascal Guillemette. Graduated in Physics at the Université du Québec à Trois-Rivières in 1997. He obtained a M.Sc. degree in Meteorology at McGill University (Montréal) in 2000. Since then he joined CAE Inc. where he currently works as a Subject Matter Expert in radar simulation.

GPU Ray Tracing-based Method for Real-time ISAR simulation

Taieb Lamine Ben Cheikh
École Polytechnique de Montréal
Montréal, Canada
taieb.lamine-ibnecheikh@polymtl.ca

Pascal Guillemette
CAE Inc.
Montréal, Canada
pascal.guillemette@cae.com

INTRODUCTION

Synthetic Aperture Radar (SAR) is a technique where the motion of the radar platform (airborne or spaceborne) is used to collect samples of an area of interest from different viewpoints. Post-processing of these samples results in a high-resolution image as if it had been captured by a single large antenna. Inverse Synthetic Aperture Radar (ISAR) is somewhat of a reciprocal technique, where the different samples are collected as the target is moving or rotating. The operational purpose of ISAR is to produce a high-resolution image in order to attempt target identification (Wehner, 1995). For example, this mode can be used by ground-based radar stations to categorize an unknown aircraft intruding into a given airspace or by a maritime surveillance aircraft for ship identification.

Realistic computer-based ISAR simulations are helpful for training radar operators (Bair and Greaves, 1987) however, the modeling is complex and computationally intensive, which tends to limit the fidelity of real-time simulations.

In ISAR simulation we can identify two main challenges:

1) Determination of visible/invisible parts of the modeled target: a classical hidden-surface determination problem common in 3D computer graphics that has many possible implementations (e.g.: z-buffering, Warnock algorithm, ray tracing, etc. such as described in Foley et al., 1996). This step represents a large proportion of the computational cost.

2) Multiple reflections: can be simulated by tracing the path followed by the radar waves as they bounce on the different surfaces. Since each part of the target structure can scatter energy in all directions including onto other parts, this becomes a recursive problem.

Graphical Processor Units (GPUs) have been used with success to accelerate target Radar Cross Sections (RCS) prediction (Gao et al., 2010), which has elements in common with ISAR simulation. For SAR simulation, ray tracing and rasterization have been studied, revealing the same dilemma between realism and real-time (Hammer et al., 2008). Another solution for ISAR is to perform preliminary analysis of the 3D models, like the manual identification (tagging) of structures to assign them a predefined signature (Cochin et al., 2015) but this requires time and expertise.

In this paper, we propose an approach which improves the performance of ISAR simulation measured in terms of rendering frame rate. It also provides a high level of fidelity by simulating multi-path reflections of radar waves and the related side-effects that a radar operator would observe on a real ISAR system. The ultimate goal is to increase the fidelity of the ISAR simulation to improve radar operator training.

In the following sections, we will present more background information about ISAR, with attention to range-Doppler imaging on ships and the ray tracing method. Then we will describe the proposed approach, its implementation and discuss the results. We will conclude with other potential applications of this method.

BACKGROUND

This section gives an overview of ISAR principles and the ray tracing method used as a base to conduct the ISAR simulation. Comprehensive descriptions of SAR and ISAR can be found in textbooks such as Skolnik, 1990 and Wehner, 1995.

ISAR in general

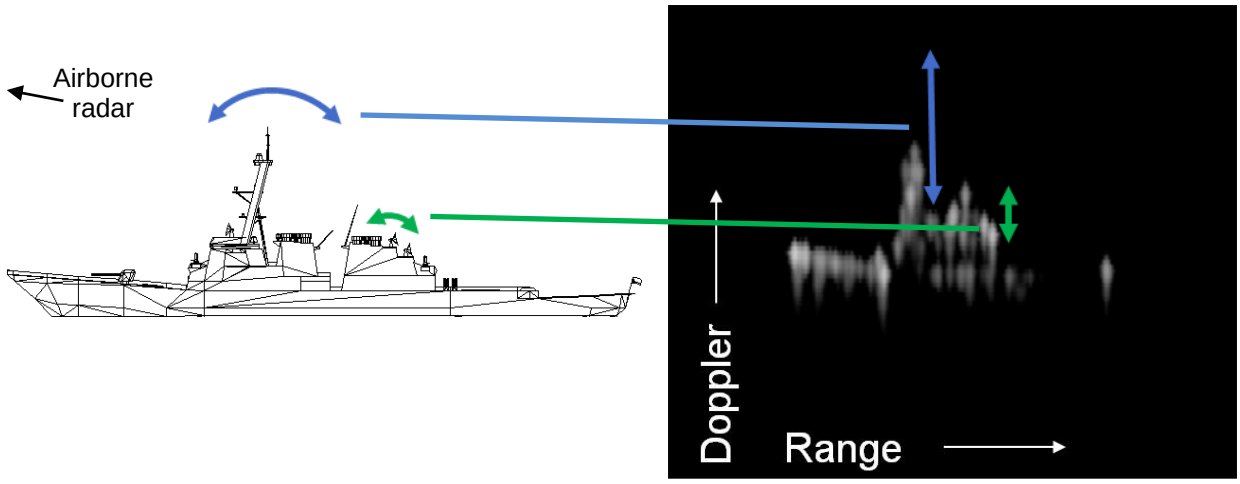
There are two common forms of ISAR display. One of them is a projection of the reflectivity on a 2D plane. Conventionally, one of the axes is the range away from the radar while the other one is the perpendicular distance (the cross-range) in the horizontal plane. The color/brightness of the pixels represents the intensity of the echoes. It is essentially a 2D map of the most reflective surfaces presented from above the target (even if the data is collected from a side).

The other type of display shows the intensity plotted on a grid of range and Doppler shift. Even though it may not seem very intuitive, for ship targets it has the advantage to show, at times, a silhouette of a ship which is very similar to a visual profile.

This paper focuses on the range-Doppler display. but the present work could be generalized to include the range/cross-range imaging as well. Furthermore, we will focus on the case of an airborne radar used to identify a ship on the surface of the ocean.

ISAR on a ship

In the case of a ship on the ocean, the action of the waves causes the ship to oscillate about its roll, pitch and yaw axes. As a result, different parts of its structure will have different relative velocities with respect to the radar. As illustrated in Figure 1(a), the highest point of a mast will have a greater motion amplitude, hence a greater velocity, than the structures closer to the center of oscillation. The relative velocity with respect to the radar can be measured as a Doppler shift. By plotting radar echoes in the form of Doppler shift as a function of range, one can visualize something that is very similar to a height vs. range profile as in Figure 1(b). In ISAR mode, the antenna is pointing at the target and the radar can take images in sequence, showing the operator a more or less fluid live animation of the ship in motion. In optimal conditions, this allows possible identification of the ship type by simply recognizing the silhouette or by measuring various dimensions such as the length between the masts, etc.



(a) (b)

Figure 1 – Example of ship motion and its correspondence on a range-Doppler display: (a) ship wireframe model; (b) sketch of a range-Doppler display where brightness corresponds to strength of echoes.

As an example, consider a stationary ship with sinusoidal oscillations in the pitch axis only. At one point, the sinusoidal motion reaches a maximum and the rotation about this axis stops. All of the elements of the ship have no relative motion, hence no velocity. Therefore, the range-Doppler signature collapses to zero Doppler. As the ship starts to rock in the other direction, the Doppler velocities start to grow in the opposite direction. This results in an alternating upright/inverted image.

Particular aspects of ISAR images

Multi-path reflections will produce additional echoes displayed at different ranges than the primary reflection because paths are longer. This can create some ghosting on the display.

Perpendicular surfaces, such as between a wall and the deck can create a dihedral corner reflector, which has the property of reflecting all incident energy back in the direction from which it came. This will make these surfaces appear more reflective to the radar. This is even more important when three surfaces are mutually perpendicular (trihedral corner reflector).

Large rotating antennas will add echoes of fluctuating intensity due to their orientation constantly changing. As it rotates, the reflecting surface will have one side going away from and one side going towards the radar. This makes the echoes spread vertically along the Doppler axis.

Ray tracing

Ray tracing is a rendering technique that fires rays of light from the camera backwards through the viewing plane and into the scene. Once in the scene, the ray tests for intersections against a set of visible 3D models. When the ray hits a surface, the material properties of that surface are calculated using a given shading model. If the material has specular reflectivity, a new ray is traced in the reflection direction to test for reflecting objects and return their illuminated color to the first surface. If the reflected ray fails to hit any surface of the scene, the background color is returned. Each time a ray hits an object it computes the corresponding shading model, which requires illumination calculations. If ray-traced shadows are required, a ray is fired in the direction of the light source. This ray tests for geometry intersections to determine if the surface is in shadow, partially illuminated, or fully illuminated.

Ray tracing can be used to simulate the propagation of a wide range of waves including light (Livatino, 2007), radio waves (Yun and Iskander, 2015) and sonar waves (Hovem, 2011). While ray tracing simulates realistic propagation aspects, it demands considerable computational power. In the recent years with GPUs offering higher computation power (massive parallel architecture) and more programming flexibility (e.g. the CUDA programming model), ray tracing is becoming more efficient in simulating complex radiation models in less time.

ISAR SIMULATION PRINCIPLES

Our objective is to improve real-time simulation of the dynamic ISAR image (Figure 1(b)) for the purpose of radar operator training.

Training context

In a research context, one tries to predict the exact strength of the signal reflected by a target in the real world, with accurate physics, diffraction effects, etc. (Chen and Martorella, 2014). In this case, precision is more important than performance. On the other hand, for interactive training, we do not need to simulate all the intermediate steps of the ISAR processing that the operator will not perceive (e.g.: generating the exact waveform, its modulation and processing by the numerous stages of radar electronics). We aim is to simulate the macroscopic variables that the operator will see, i.e. Doppler shift and echo intensity of the different parts of the target, as realistically as possible.

Simulation geometry

We will consider the case of an isolated ship, which will allow us to ignore echoes from the coastline. We will also assume that the ship is uniformly illuminated by the radar beam. This is reasonable in the case where the target is offshore and not too close to the radar.

The general idea is to sum the energy reflected for a given range bin (Figure 2) and Doppler bin. Each of these bins will become a pixel where the total energy will be converted into color/grey intensities.

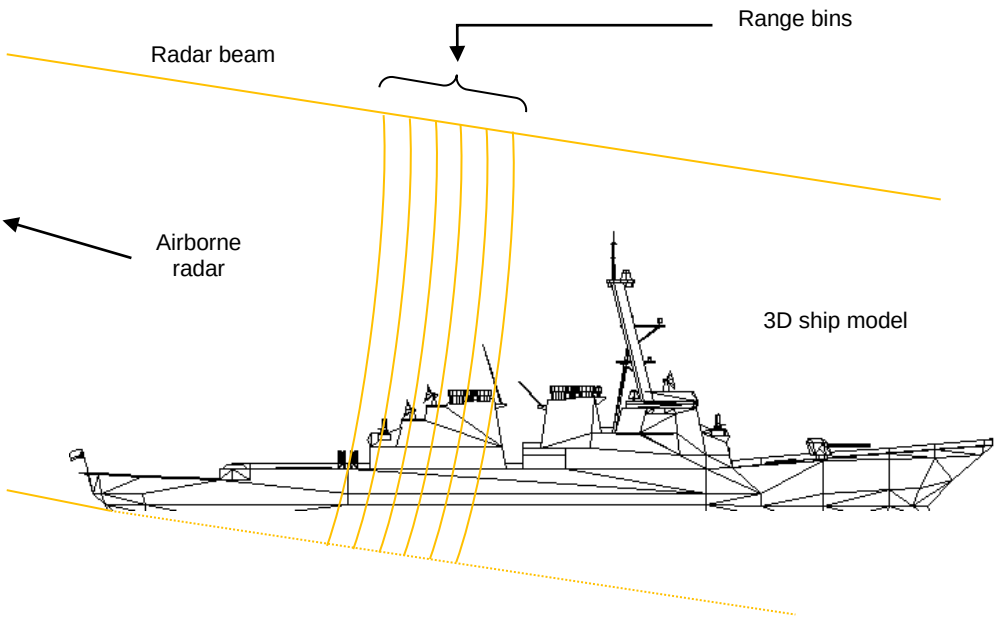


Figure 2 – Sampling of the 3D ship model in range

In other words, we must sample the 3D ship model and, for each element, determine if it is visible and how much energy it reflects, determine when the reflected energy will return and what Doppler shift will be caused.

From a given vantage point, a portion of the structure of the ship can be hidden by another part in front of it. Hidden parts will not be illuminated by the radar and therefore will not contribute to the returned energy. The power reflected by a visible part is computed based on the incident power, the (modeled) nature of the material and the angle of incidence of the radar waves relative to the surface.

Finally, for each visible element, we need to compute its velocity component towards the radar (radial velocity). Equivalently, we can compute the Doppler shift resulting of the reflection of the radar waves based on the equation proposed in (Wehner, 1995):

$$f_D = \frac{2f}{c}\left[(\vec{\omega}\times\vec{r}) \cdot \hat{R}\right] \quad (1)$$

where $\vec{r}$ is the position of the element relative to the center of rotation, $\vec{\omega}$ is the ship's angular velocity vector which can be a function of time, $\hat{R}$ is the unit vector in the direction of the radar, f is the radar frequency and c is the speed of light.

METHODOLOGY

We model the propagation and reflection of the radar waves as if they were rays of light. By doing this, we can leverage a ray tracing engine to solve the hidden-surface problem and perform other calculations. Furthermore, these computations can be offloaded to a GPU, which is highly optimized for parallel computation and ray tracing.

The model: ray tracing applied to ISAR simulation

The target (ship) model is loaded in the scene, i.e. the virtual space that we will render. It is composed of polygons with different materials. Each material can be parameterized to have different reflection properties.

In the 3D rendering terminology, we use the light source to model the energy sent by the radar and the camera to measure the returns. This light source/camera system is analogous to the radar antenna being able to transmit and receive. It naturally follows that a visual rendering of the target model under this lighting condition and viewing angle will be similar to the "illumination" of the target by the radar.

The main difference between a visual rendering (or camera view) and radar imagery is that the first one produces a 2D image where the depth is inferred by our brain from the perspective perceived in the image. A conventional radar image, normally detects intensity versus range since the radar sends a pulse and samples the returned echo for discrete elements of time. Basically, to obtain information useful for a radar simulation, one of the tasks is to extract the depth (or range) from the 2D image. This is possible because for each pixel (for each ray) we can query the rendering engine for the travelled distance.

Based on the ray tracing procedure to calculate the ray depth, the reflection directions and coordinates of the hit points, we compute both the range and the Doppler shift of the illuminated element.

The Algorithm

We build our ISAR image generation model on top of a ray tracing engine named OptiX API (Parker, 2010), a generic framework that supports ray tracing pipeline customizations. While OptiX is used to implement the ISAR simulator, our approach could be implemented on different ray tracing engines. To model the radar wave propagation, we implement the following customization:

- the radar's transmission is modeled as a directional light source;
- the light source is positioned at a large distance from the illuminated scene;
- the radar's reception is done by the camera, positioned close to the illuminated scene in order to capture details at a high resolution;
- the camera is orthographic; i.e.: the rays assigned to each pixel are normal to the rendering plane and will be initially launched in the scene parallel with each other.

The orthographic projection is used to avoid incorrect perspective effects introduced by positioning the camera close to the scene. This is a good approximation since, in the case of a real radar far from the target, the rays are nearly parallel and perspective effects are minimal.

For each pixel that we render from the camera's point of view, we propagate a ray perpendicular to the rendering plane and we determine if it intercepts any surface of the rendered ship model. If so, we compute its Doppler shift with Equation 1 and how much light this element of surface receives from the light source (applying coefficients for specular and diffuse reflection). We continue to compute the path of the ray by simulating all possible reflections depending on the surface reflectivity. The reflected energy is computed with the Phong shading model which will be discussed in the next section.

In a conventional ray tracing model, the collected intensity of a ray is an accumulation of intensities each time it encounters an object. However, in our case, we compute the Doppler velocity, the range and the reflectivity of each hit point along the ray’s path. And rather than accumulating the intensities of every hit point, we save separately their contributions, which are then mapped on different ISAR image pixel coordinates (different range bins and Doppler velocity bins).

We perform repeatedly the above steps until there are no more reflections (Figure 3). The final step is to convert this energy into a pixel brightness level for display.

An example of the implemented model is represented in Figure 4. It is a reflector composed of three surfaces of any material. The incident ray generated by the radar is represented with black arrows. Each received ray is represented with different color where, the red arrow represents the diffuse reflected ray by the surface (1), the green arrow represents the diffuse reflected ray of the surface (2), the blue arrow represents the diffuse reflected ray of the surface (3) and the magenta arrow represents the specular reflected ray by all the surfaces. Our simulation model allows computing all the reflections and the respective range and Doppler shift for every hit point.

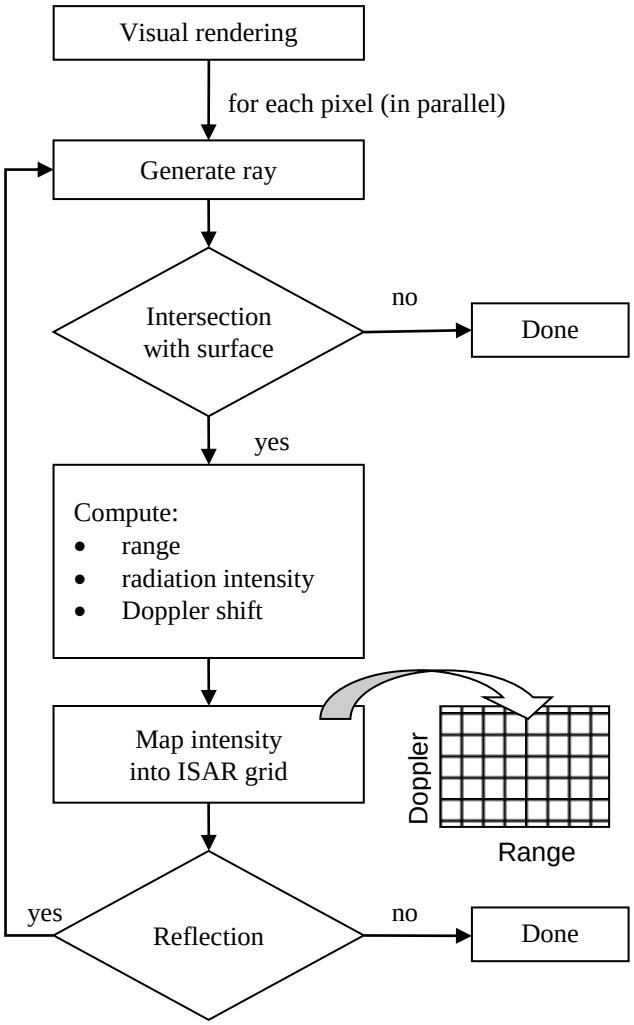


Figure 3 – Multiple-reflection algorithm for ISAR

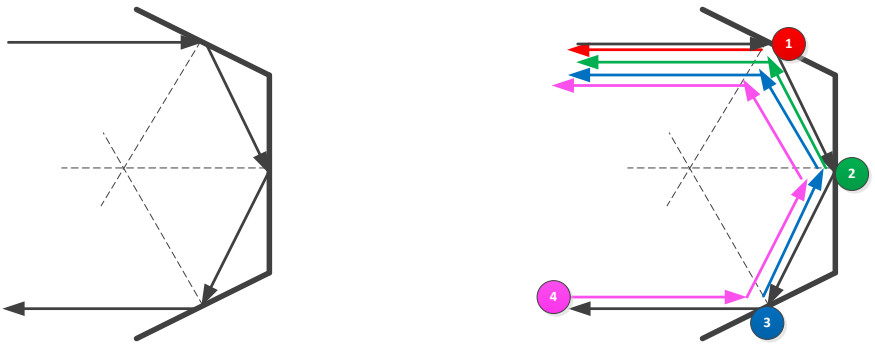


Figure 4 – Multiple-reflection simulation

Additional effects can be added for realism such as noise, hysteresis (i.e. a running time window for averaging the images), etc. The results presented in this paper do not include these effects as we wish to focus on the potential of ray tracing customization.

Ray Propagation Model

In our case, we use a radiance model that is based on the Phong shading model (see Figure 5).

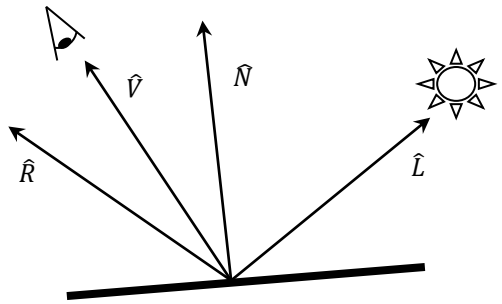


Figure 5 – Phong shading model

The Phong shading model (Phong, 1973) is described by this equation:

$$I_p = k_a i_a + \sum_{m \in lights}\left(k_d\left(\hat{L}_m \cdot \hat{N}\right)i_{m,d} + k_s\left(\hat{R}_m \cdot \hat{V}\right)^{\alpha} i_{m.s}\right) \quad (2)$$

where:

I_p is the radiance of a surface point;
m is the light source from the set of directional light sources $lights$;
i_a is the ambient lighting;
i_d and i_s are the diffuse and specular component of the light source;
$\hat{L}$ is the direction vector from the point on the surface toward each light source;
$\hat{N}$ is the normal at this point on the surface;
$\hat{R}$ is the direction that a perfectly reflected ray of light would take from this point on the surface;
$\hat{V}$ is the direction pointing towards the viewer (such as a virtual camera);
k_s, k_d and k_a are the coefficients for specular, diffuse and ambient light reflection;
α is the shininess constant for this material.

Our radiance model uses essentially two radiance parameters: (1) k_d: the diffuse radiance coefficient and (2) k_s: the specular radiance coefficient. The ambient coefficient is assumed to be zero because ambient sources of radiation at the radar frequency are considered negligible for this exercise; the radar is the only source of illumination.

EXPERIMENTS AND RESULTS

We used ship models composed of triangles. The models can be given any orientation and can rotate about the roll, pitch and yaw axes to reproduce the conditions of the ship in the virtual world. To illustrate the simulated features, we will first consider a simple destroyer model composed of 1625 triangles and subject to pitch motion only.

To simplify the interpretation, all the surfaces were made of identical material which had arbitrary diffuse reflection coefficient k_d = 0.2 and specular reflection coefficient k_s = 0.8. The objective was to make the surfaces behave a bit like mirrors for specular reflection while still allowing them to diffuse some energy in all directions. Figure 6 shows the ISAR simulation results when a single reflection only is allowed. Figure 6(a) shows the intermediate result or "visual" rendering of the destroyer from the radar point of view. We can mention that surfaces hidden behind structures are not visible (occulted) and light intensity depends on the orientation of the surfaces with respect to the camera (radar). In the ISAR display, this translates in into brighter areas and areas of shadows i.e.: where no radar energy could reach a surface such as in front of the bridge (Figure 6(b)). The orientation changes because this is a range-Doppler display where range increases to the right.

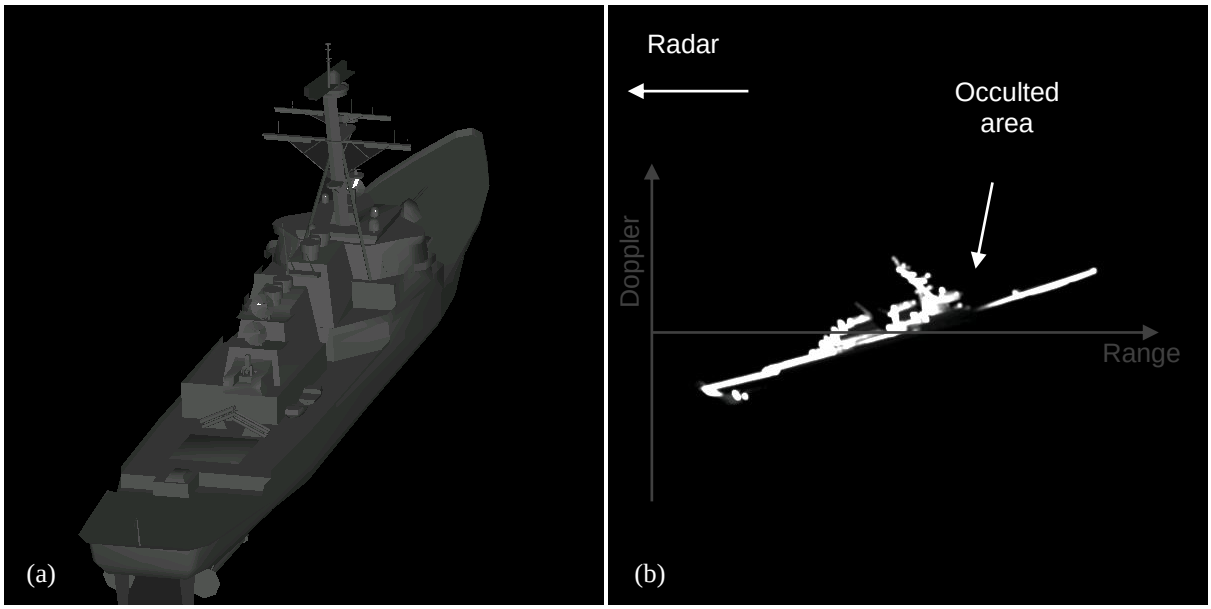


Figure 6 – ISAR simulation using a single reflection: (a) "visual" rendering as seen from the radar's point of view, (b) corresponding range-Doppler display.

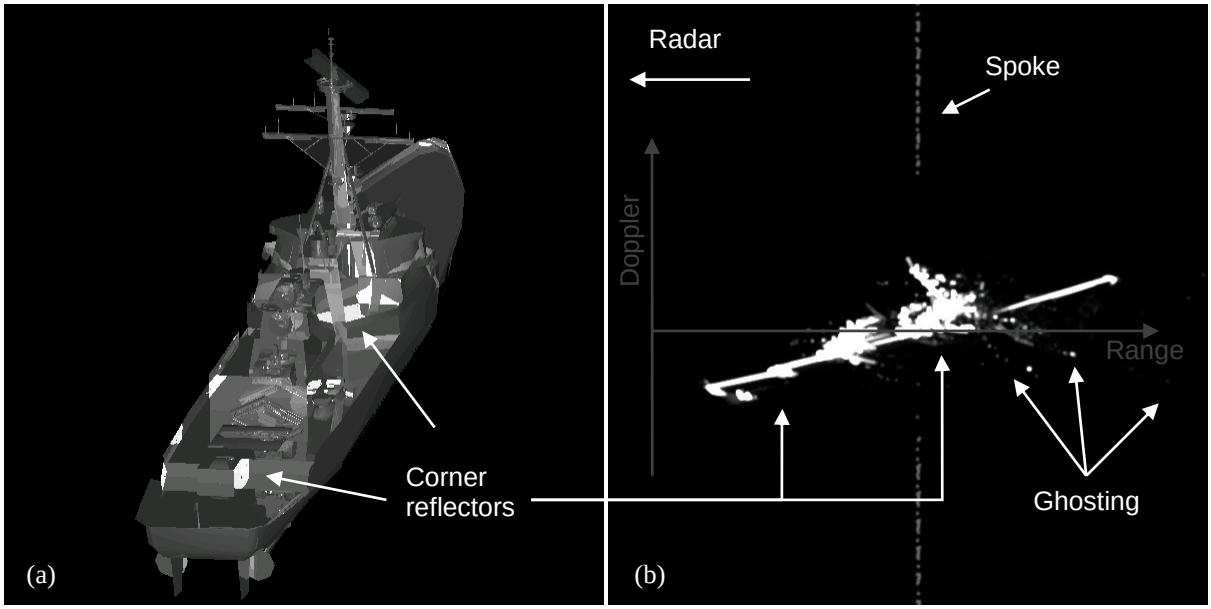


Figure 7 – ISAR simulation using multiple reflections: (a) "visual" rendering as seen from the radar's point of view, (b) corresponding range-Doppler display including spoke caused by rotating antenna.

In Figure 7, multiple reflections were enabled. In Figure 7(a) we can see the brightening of the surfaces forming a corner reflector. We can also see the reflection of the masts and other structure on the deck. The corresponding range-Doppler image in Figure 7(b) shows brighter spots caused by corner reflectors in areas near the structures. We can also see "ghost" echoes outside the envelope of the ship. These are the result of multiple reflections, causing the path of the ray to lengthen and hence, to appear at a further range. Figure 7(b) also features a vertical spoke (periodic in the real-time simulation) caused by the rotating antenna located at the top of the mast. For the sake of this experiment, the antenna was made highly reflective and rotating at a high speed which results in a spreading across a large Doppler interval.

Performance

The simulation model is implemented using NVIDIA OptiX API and was also tested on different 3D models such as a cruise ship, and an aircraft carrier (Figure 8). For the latter, made of 27,463 triangles, the frame rate was recorded for three window sizes (varying ray number) and up to 6 calculated reflections. The model was allowed to oscillate about roll, pitch and yaw axes.

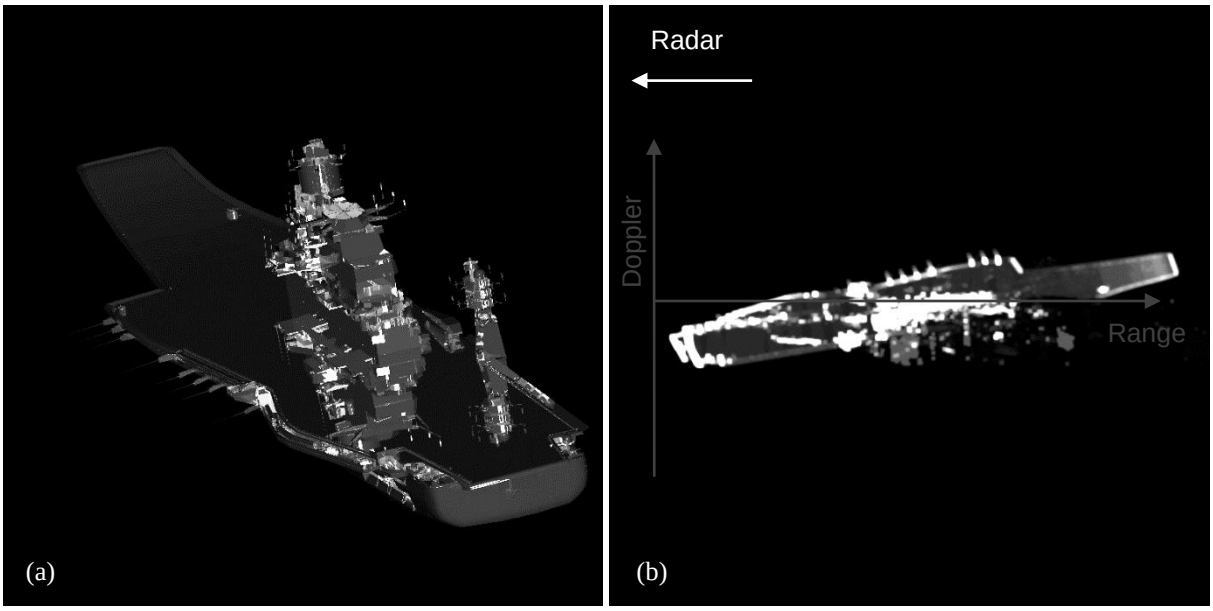


Figure 8 – ISAR simulation for an aircraft carrier with 27 463 triangles: (a) "visual" rendering as seen from the radar's point of view, (b) corresponding range-Doppler display. Notice the reflection of the forward and aft bridge islands on the deck and their associated ghost effects on the ISAR image.

Once loaded in the GPU memory, the number of triangles in the model does not have a measurable impact on the frame rate. On the other hand, the frame rate decreases as the size of the window or the maximum number of reflections increase, as shown in Figure 9.

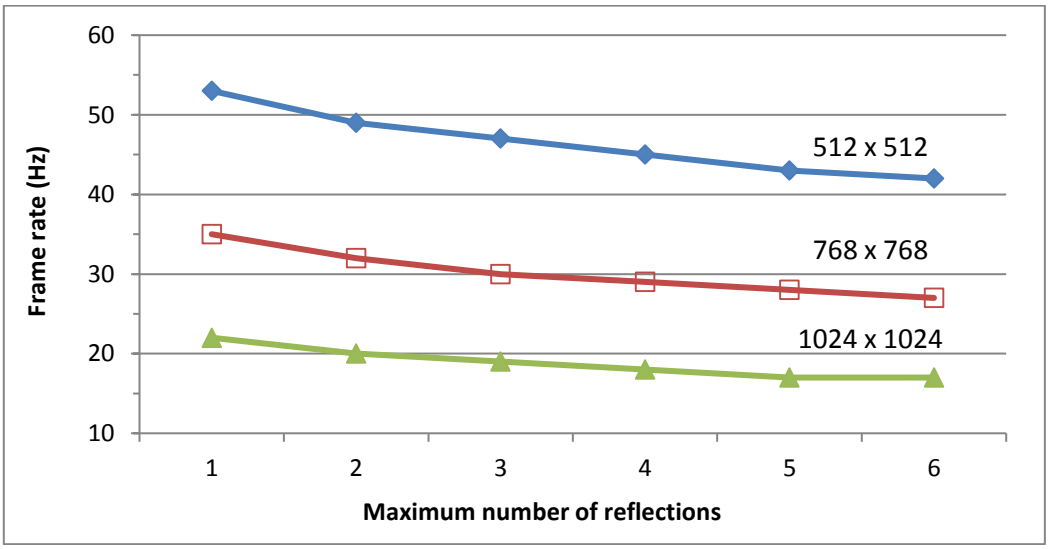


Figure 9 – Frame rate of the ISAR simulation as a function of the maximum number of reflections for different sizes of the "visual" image. Model used: aircraft carrier (27 463 triangles).

CONCLUSION

This work demonstrates that ISAR images can be simulated efficiently with a high degree of realism by exploiting analogies with visible light and visual rendering principles. Indeed, ray tracing can determine if a target's surface is visible or occulted by another surface. Then the amount of radar energy reflected by a surface can be computed based on its nature and orientation using the Phong shading model. Finally, the brightening caused by corner reflectors and the ghosting effects will result from the calculation of multiple reflections (with no need for offline analysis or preprocessing of the polygonal target model). By summing the reflected power contributions sorted by distance and relative velocity intervals, we obtain the range-Doppler display of a target.

These tasks can be performed on a GPU, which is optimized for these kinds of calculations. This saves computation time on the CPU. These savings can be used for other simulated radar functions which, on real radars, continue in the background while the radar is in ISAR mode. A simple example would be the dead-reckoning (i.e. extrapolation) of the positions of targets previously tracked by the radar in surveillance mode.

While the proposed method simulates in real time some interesting ISAR effects, it does not simulate:

- defocusing which happens when the antenna is not properly trained on the target;
- noise, which was not simulated but could be simply added to the power array;
- surface waves and diffraction on the target, since the Phong shading model was used.

However, some of these artifacts will be explored and integrated in the future.

The primary objective was the simulation of range-Doppler imaging, but this approach could be extended to simulate range/cross-range displays. It could also be used to estimate the fluctuation of a target's overall radar cross-section (RCS) for a surveillance radar simulation. This could be useful when no RCS data is available for a simulated target.

REFERENCES

Bair, G. L. and Greaves, W.C. (1987), Low-Cost Digital Radar Generator for Comprehensive Radar Simulation, in *Proceedings of the Interservice/Industry Training, Simulation and Education Conference (I/ITSEC)*, Orlando,FL.

Chen, V.C. and Martorella, M. (2014), *Inverse Synthetic Aperture Radar Imaging: Principles, Algorithms, and Applications*, Edison, NJ.: SciTech Publishing.

Cochin, C., Louvigne, J.C. and Houssay, J. (2015), Raw Data Generation of Maritime Scenes Using MOCEM V4 and PHYS-IQ V1, in *Piers Proceedings*, Prague, Czech Republic.

Foley, J.D., van Dam, A., Feiner, S.K. and Hughes, J.F. (1996), *Computer Graphics: Principles and Practice*, Cornell University: Addison-Wesley.

Gao, P.C., Tao, Y.B. and Lin, H. (2010), Fast RCS prediction using multiresolution shooting and bouncing ray method on the GPU, *Progress In Electromagnetics Research, vol. 107*, pp. 187-202.

Hammer, H., Balz, T., Cadario, E., Soergel, U., Thönnessen, U. and Stilla, U. (2008), Comparison of SAR simulation concepts for the analysis of high-resolution SAR data, in *Proceedings of the 7th European Conference on SAR (EUSAR 2008)*, Friedrichshafen, Germany

Hovem, J. M. (2011), *Ray trace modeling of underwater sound propagation - documentation and use of the PlaneRay model*, SINTEF Report A21539, ISBN 978- 82-14-04997-8.

Livatino, S. (2007). Photorealistic VR games?, in *17th International Conference on Artificial Reality and Telexistence, (ICAT 2007)*, pp. 292–293.

Parker, S.G., Bigler, J., Dietrich, A., Friedrich, H., Hoberock, J., Luebke, D., McAllister, D., McGuire, M., Morley, K., Robison A. and Stich, M. (2010), OptiX: A general purpose ray tracing engine, *ACM Transaction Graphics, 29(4)*, pp. 1–13.

Phong, B. T. (1973), *Illumination for Computer-Generated Images*, Salt Lake City: Department of Computer Science, University of Utah.

Skolnik, M. I. (1990) – editor in chief, *Radar Handbook, 2nd ed.*, Boston, MA. : McGraw-Hill.

Wehner, D. R. (1995), *High-resolution radar, 2nd ed.*, Norwood, MA.: Artech House, Inc.

Yun, Z. and Iskander, M. F. (2015), *Ray tracing for radio propagation modeling: principles and applications*, IEEE Access, vol. 3, pp. 1089- 1100.