SAR SIMULATION FOR LARGE SCENES BY RAY TRACING TECHNIQUE BASED ON GPU

Tingting Liu, Kaizhi Wang and Xingzhao Liu

Department of Electronic Engineering, Shanghai Jiao Tong University
Shanghai, China
liutt@sjtu.edu.cn

ABSTRACT

This paper uses ray tracing technique to simulate a complete imaging process of Synthetic Aperture Radar (SAR). The simulator can collect the raw data and generate accurate shadows of the object. The radiation pattern is defined by a spotlight to have more realistic imaging effect. Considering the high performance computing of modern graphics processing units (GPU), we transfer the ray tracing algorithm from CPU to GPU. Thus, the simulator can work on a large 3D scene with relatively high efficiency.

Index Terms --- Ray Tracing, SAR simulation, 3D scene, Shadow, Graphics Processing Units (GPU)

1. INTRODUCTION

SAR imaging possesses the outstanding merits and is widely used in many fields. With the new very high resolution SAR sensors onboard the TerraSAR-X and COSMO-SkyMed satellites with spatial resolution down to 1m, radar imaging simulators has become increasingly popular [1].Basically, there are two main kinds of simulator: the SAR image simulator and the SAR raw data simulator. The object of the former one is the image without and intermediate products [2]. We focus at the latter one, which can get back raw data and be used to generate SAR image.

There exist several SAR simulation tools such as SE-RAY-EM [3], MOCEM-LT [4], which are based on ray tracing but have no real-time capability [5]. S. Auer using POV Ray, simulated reflection maps which is similar to real TerraSAR-X images [6,7]. These approaches focus on geometrical quality of the simulation and indicate the benefits of ray tracing. But, there is always a trade-off between the calculation time and the richness of details. The speed of CPU cannot satisfy the requirement of efficiency.

Supported by National 863 Plan: 2011AA120401

With the development of graphics processing units (GPU), modern graphic cards support user-supplied programs and process the 3D hardware acceleration ability. GPUs are currently very powerful platforms, provided for tens or hundreds of cores with acceptable clock frequencies (500-600MHz) [8]. The GPU performance has been increasing at a rate of 2.5 to 3.0 × annually, compared with CPUs of 1.41 × annual performance growth rate [9]. Thus, it is reasonable to transfer the calculation process from CPU to GPU.

This paper focuses on the whole process of the simulation, including radar beam projection, SAR raw data collection and image formation. We use a ray tracing engine OPTIX to deal with real-time SAR simulation for large 3D scenes. Section 2 describes each part of the simulator in detail. In Section 3, we introduce the realization method of the simulator. Section 4 gives the simulation results. A conclusion is drawn in section 5.

2. DESCRIPTION OF THE SIMULATOR

The core approach of this simulator is ray tracing technique. In order to simulate image that is as close to the real one as possible, we make an effort in radar beam and reflection phenomena.

2.1. Radar Beam Projection

In real SAR imaging, the received signal strength is decided by azimuth beam direction. Since most SAR antennas have no weight in the azimuth plane, its single pass pattern can be approximated to a sinc function [10].

$$p_a(\theta) \approx \operatorname{sinc}\left(\frac{0.886\theta}{\beta_{bw}}\right) \qquad (1)$$

Where, θ is the angle between the ray and the slant range plane, β_{bw} is the azimuth beam width.

To describe the two-way propagation of radar energy, the received signal strength can be described as the square of $p_a(\theta)$, and often be presented as the function of azimuth time η.

$$\omega_a(\eta) = p_a^2\{\theta(\eta)\} \tag{2}$$

The radiation pattern is defined by a spotlight to have more realistic imaging effect, defining the power of each ray of different directions.

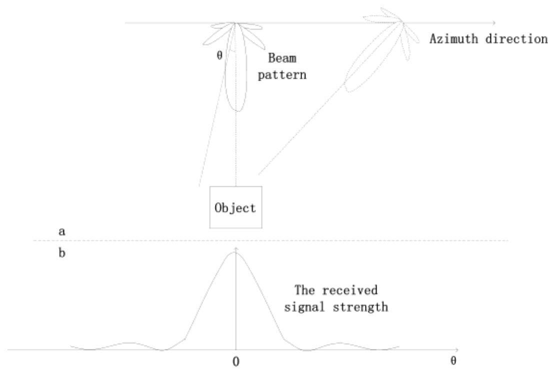

Fig.1. SAR simulation model

In figure 1(a), a radar beam is emitting from the right over the object. Figure 1(b) shows the current signal strength received by the object. In this way, the simulator can also reflect the side lobe of the radiation pattern.

2.2. SAR Raw Data Collection

Ray tracing is then used to simulate SAR imaging, it can render a relatively realistic shading effect. Figure 2 shows the side-looking geometry of SAR and Phong illumination model on the surface of the object.

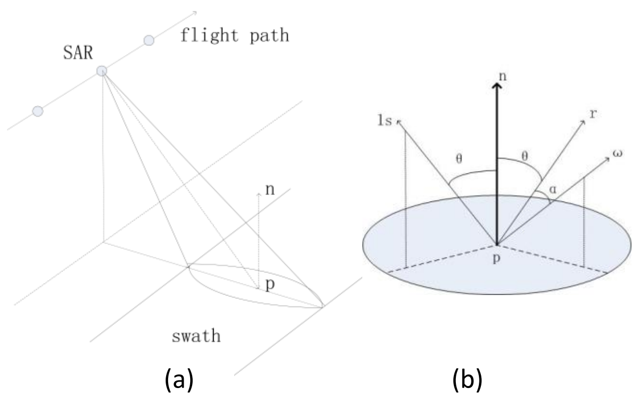

Fig.2. **(a)** The side-looking geometry of SAR **(b)** Phong Illumination Model

According to Phong illumination model [11] shown in figure 2(b), the light-intensity of point p perceived by the position ω can be presented as

$$L_p = L_c \times k_d \times \cos\theta + L_c \times k_s \times \cos^e \propto + k_a \times L_{am} + L_r \tag{3}$$

$$L_c = L \times \text{attenuation} \tag{4}$$

Where, θ is the incident angle, $\propto$ is the angle between the direction of the reflected light and the line of ω, e is the Phong index. K_d (k_s) is the diffuse (specular) reflection value, which is grabbed from the properties of the surface material. Other properties like normal vectors will be calculated by the simulator. This formula can be used to calculate the final result of the point p.

We define a payload to store the raw data, including the azimuth information and the energy received by the sensor. Intersection returns will be recorded in the payload.

2.3. Image Formation

For rendering the results to an image, ray tracing is a reciprocal process, where the ray is launched from the imaging panel. Since the viewpoint is not fixed with the sensor, a ray is traced from observer's viewpoint to the pixel. However, this is more about an internal mechanism of the computer graphics.

The overall idea of ray tracing is to shoot rays towards the target judging whether the scene is illuminated or not is. If a ray does not intersect any object in its way, the pixel's value is the background color of the scene. Conversely, shadows, diffusion and reflection will be calculated. After calculating the intersection results between the hit point and the radar, we can determine the shading method of the hit point.

During the SAR imaging process, an object will occlude a portion of itself and the background. No returns are received from the object's radar shadow [12]. The shadow region is devoid of speckle and therefore can provide a much more robust representation of the observed object [13]. Thus, it is valuable to explore the shadow in SAR simulation.

Since ray tracing is a recursive process, a reflection ray might send other reflection rays, shadow rays, and so forth. The bounce level will be sent up to 100, which should be adequate for nearly any scene. Any reflection ray that exceeds this threshold will be terminated. Thus, the simulator can generate an accurate shadow effect.

To observe the shadow effect, we use a PPM file to save the imaging results from a fixed viewpoint. This approach is very useful when a series of radar beam is sent by the vehicle in flight, so the returns processed to the same pixel can be added coherently.

3. SIMULATION METHOD

3.1. 3D Models

The models are defined by the vertices and vertices' index, which constructs the surface of the model with numerous triangles. We also define the surface parameters, including the ambient reflectivity, diffusion reflectivity and specular reflectivity. In SAR imaging, the ambient light is set to zero. Combined with normalized surface normal, these properties will be loaded to GPU to calculate the shading results.

For large scene simulation, it is reflected as the scale and the complexity of the scene objects. With regard to ray tracing algorithm, the most obvious difference is the amount of calculation.

A method to reduce the amount of intersection calculation is space-subdivision approach. It tries to locate the objects within a set of separated space cells; each cell contains a limited number of objects. When a ray hits a given cell, it is tested against the objects within the cell for any possible intersection. Otherwise, the calculation will not be carried.

3.2. Graphic Tools

The NVIDIA OPTIX ray tracing engine is a programmable system designed for NVIDIA GPUs and other highly parallel architectures [14]. The combination of user programs and hardcoded OPTIX kernel code forms the ray tracing pipeline, which is outlined in figure 3.

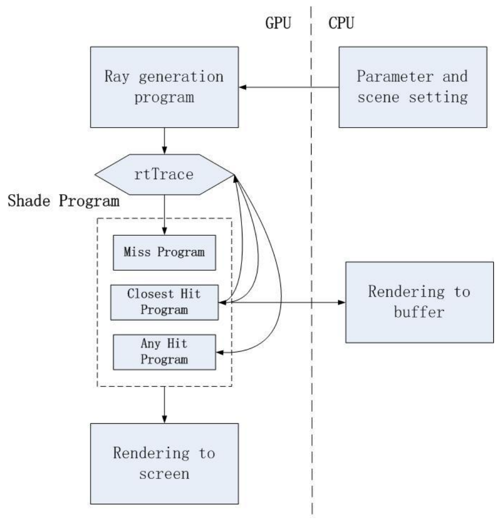

Fig.3. Ray tracing architecture

We use three programs to classify the ray intersection conditions. Miss program is invoked when a ray intersects no object. Closest hit program is mainly used for principle rays to shade the surface. Any hit program is to implement shadow ray casting and eliminate redundant traversal computations. These three programs constitute the main body of the ray tracing algorithm.

4. RESULTS OF THE SIMULATOR

4.1. Shadow Synthesis

We make the sensor move in a certain track and shoot a series of radar beam, which is also depicted in figure 2. While the radar's position is changing, the shadow will transform accordingly. Then, synthesize these PPM files.

To have an obvious imaging effect, we use a simple tetrahedroid as the target and simulate the spotlight SAR. The sensor shoots 1000 radar beam along its track, and the azimuth angle changes from 45° to 135°. Figure 4(a) shows the 3 status of the changing shadow. After adding the generated PPM files coherently, we can get the final shadowed area in figure 4(b). Figure 5(a) shows the status at zero Doppler time by a fixed shooting direction along the same path, figure 5(b) is the synthetic shadow.

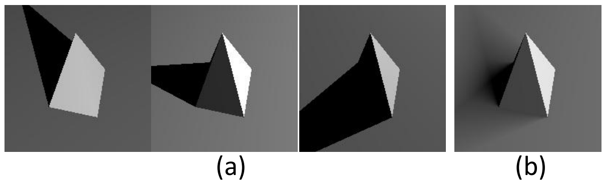

Fig.4. Shadow of the object

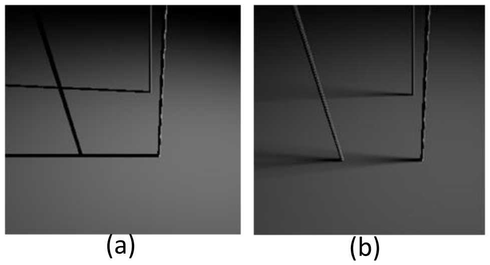

Fig.5. Synthetic shadow of the sticks

In figure 4 and 5, one area always cannot be detected along the track. These results are feasible for real-time SAR simulation.

4.2. Simulation for 3D model

Large and complex 3D models are used to construct the large scene for the SAR simulator. The angular aperture is 20°, the slant range is 8400m, and the resolution is about

3km. Figure 5 shows the ray tracing result of an aircraft carrier. The object contains 251,328 vertices and 482,600 triangles.

In figure 5, we can see the side lobe of the radiation pattern, which will decrease the capability of radar to detect weak targets in multi-target environment.

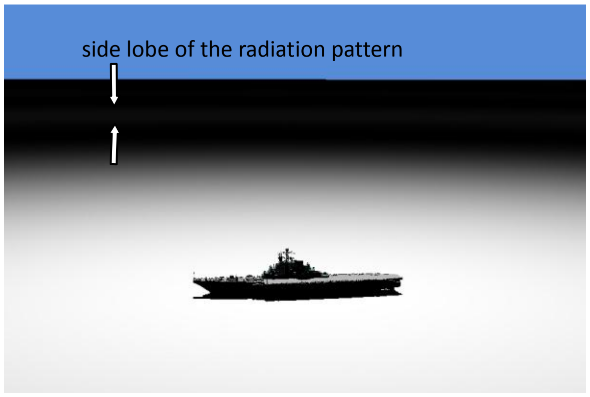

Fig.5. Simulation for an aircraft carrier

Time cost is shown in the table 1. Compile kernel means to create a final computation kernel from the programs. Besides the time to load geometry, total time to carry out ray tracing algorithm is about 2.3s in a NVIDIA GTX 570 graphic card.

Table.1. Time cost of the simulator

Time to load geometry	17.432s
Time to compile kernel	1.526s
Time to build AS	3.824s

5. CONCLUSION

In this paper, ray tracing algorithm is used to simulate SAR imaging based on GPU. The results indicate that ray tracing method has a good performance on shadow rendering and possesses many user-defined properties. Combining the multi kernels of GPU, this method improves the imaging effect and efficiency.

Despite the application in SAR simulation, the simulator can be also applied to many other situations. One aspect now being studied is ghost imaging.

The next step for this simulator is to compare the imaging result with the one based on electromagnetic calculation.

6. REFERENCE

[1] D. Brunner, G. Lemoine, H. Greidanus, L.Bruzzone, “Radar Imaging Simulation for Urban Structures”, IEEE Geoscience and Remote Sensing Letters, Vol.8, pp.68-72, 2011.

[2] Yuan Lu, Kaizhi Wang, Xingzhao Liu, and Wenxian Yu, “A GPU based real-time SAR simulation for complex scenes,” Radar Conference - Surveillance for a Safer World, Bordeaux, pp.1-4, 2009.

[3] H. J. Mametsa et al., “Imaging Radar Simulation in Pealistic Environment Using Shooting and Bouncing Rays Technique”, Proceedings of SPIE 4543, SAR Image Analysis, Modeling and Techniques IV, Toulouse, 2002.

[4] C. Cochin et al., “MOCEM-An ’all in one’ tool to simulate SAR image”, Proceedings of EUSAR, Friedrichshafen, Vol.1, pp.225-228, 2008.

[5] H. Hammer, K. Schulz, “SAR-Simulation of Large Urban Scenes Using an Extended Ray Tracing Approach”, Publications of Urban Remote Sensing Event, Munich, pp.289-292, 2011.

[6] S. Auer, S. Hinz, R. Bamler, “Ray-Tracing Simulation Techniques for Understanding High-Resolution SAR Images”, IEEE Transactions on Geoscience and Remote Sensing, Vol.48, pp.1445-1456, 2010.

[7] S. Auer, Xiaoxiang Zhu, S. Hinz, R. Balmer, “3D Analysis of Scattering Effects Based on Ray Tracing Techniques”, Publications of IGARSS, Cape Town, pp.17-20, 2009.

[8] S. Ciolo, G. Ordeix, E. Fernández, M. Pedemonte, P. Ezzatti, “Improving the Performance of a Ray Tracing Algorithm Using a GPU”, Proceedings of the conference of the Chilean Computer Science Society, Antofagasta, pp.11-20, 2010

[9] P. Trancoso and M. Charalambous, “Exploring graphics processor performance for general purpose applications,” Proceedings of 8th Euromicro Conference on Digital System Design (DSD), pp.306–313, 2005.

[10] Ian G. Cumming, Frank H. Wong, Digital Processing of Synthetic Aperture Radar Data: Algorithms and Implementation, Publishing house of electronics industry, Beijing, pp.91, 2007.

[11] Bui Tuong Phong, “Illumination for computer generated pictures”, Communication of ACM, Vol.18, pp.311-317, 1975.

[12] S. Papson, “Classification via the Shadow Region in SAR Imagery”, IEEE Trans. on Aerospace and Electronic Systems, Vol.48, pp.969-980, 2012.

[13] M. Jahangir, D. Blacknell, et al, “Extracting information from shadows in SAR imagery”, Publications of ICMV, Islamabad, pp.107-112, 2007.

[14] S. Parker, J. Bigler, A. Dietrich, et al, “Optix:A general purpose ray tracing engine”, ACM Transactions on Graphics, 2010.