LOD and Texture Mapping for Real-Time Radar Ground Map Simulation

Radu Visina, Jameson Bergin, David Kirk, Peter Skangos
ISL Inc.
San Diego, CA
rvisina@islinc.com, jbergin@islinc.com, dkirk@islinc.com, pskangos@islinc.com

ABSTRACT

Modern radar simulation techniques can be used to generate synthetic, yet highly realistic, radar images such as Synthetic Aperture (SAR) and Real Beam (RB) ground maps. In general, when simulating the effect of an external channel, a numerical electromagnetic solver should have 3D models of the scene it is simulated to observe. The creation of these scene models, as well as the rendering of radar images (using ray tracing and/or rasterization), parallels the developments and techniques of modern computer graphics (CG) systems. While CG techniques have mostly focused on attempting to simulate optical cameras, the physics of RF microwaves, as well as the basic radar fact that information from one dimension of radar images comes from range, means that simulation of a radar RF receiver requires some adjustment when applying CG techniques. For example, in wide-area ground map radar images, features such as buildings and ground vehicles can be seen as strong returns from many angles, but it is computationally impractical to use highly detailed mesh models for every scatterer in a scene. This paper reviews and adapts the modern CG techniques of Level of Detail (LOD) and mipmapped texture mapping to solve the problem of increasing resource requirements when the physical area of interest increases in the domain of radar ground map or ground clutter simulation. The improvements that are achieved with these techniques are of immense value for mission planning, radar scope interpretation training, and for the generation of synthetically generated training data for machine learning systems.

ABOUT THE AUTHORS

Radu Visina (PhD EE University of Connecticut) is a research engineer at ISL and has been actively developing theory and practical solutions for tracking of maneuvering targets using radar, real-time radar clutter simulation, and knowledge aided radar tracking in 3D environments. In publications, Dr. Visina has advanced the theory and practical methods of tracking maneuvering targets, including highly-maneuvering/goal-oriented targets, the track-to-track fusion of maneuvering target state estimates, and advances in real-time radar clutter simulation.

Jameson Bergin (MSEE, University of New Hampshire) Mr. Bergin is a Senior Vice President and Principal Engineer with ISL, Inc. where he is currently the General Manager of ISL's Research, Development, and Engineering Solutions (RDES) Division. He has over 25 years of experience in RF systems, signal processing, and physics-based modeling and simulation. Mr. Bergin is one of the principal developers of ISL's RFView® commercial M&S software and has published numerous conference and journal articles in the areas of radar and electronic warfare. He holds 7 US Patents and is a Senior Member of the IEEE.

David Kirk (BS, Physics, Texas A&M University, MS, Physics, The Pennsylvania State University) is a Principal Engineer with ISL and is a Senior Member of IEEE. Mr. Kirk has over 30 years experience in systems analysis and modeling of sensor systems, specializing in the areas of exploitation of full motion video, synthetic aperture radar, automatic target recognition in both radar and hyperspectral imagery, ground moving target indication radar, ground target tracking, and navigation/geolocation accuracy. He is currently working on high-fidelity hardware in the loop radar simulation at ISL.

Peter Skangos is a Cold War Warrior Rocket Scientist/Angel Investor/Competitive Rower/Dad focused on solving challenging problems through innovation and creative perspectives. He currently serves as the Chief Commercialization Officer for Information Systems Laboratories (ISL). Mr. Skangos also currently serves as Chairman/President of the Interagency Seminar Group (IASG) located at Brookings Institute.

LOD and Texture Mapping for Real-Time Radar Ground Map Simulation

Radu Visina, Jameson Bergin, David Kirk, Peter Skangos
ISL Inc.
San Diego, CA
rvisina@islinc.com, jbergin@islinc.com, dkirk@islinc.com, pskangos@islinc.com

INTRODUCTION

Radar simulation is important for applications including virtual radar toolset training (e.g. scope interpretation), mission planning, EW training, algorithm validation, hardware validation, and virtual machine learning training data. Physically accurate approximations of ground map or clutter are important in radar simulation because relative received power can determine the existence or nonexistence of targets of interest. Specular returns (especially those from dihedral or trihedral geometries) can come from objects that are smaller than one range/cross-range data bin, and this makes radar cross-section (RCS) a useful concept for characterizing targets in radar data. Similarly, when we sample a polygon mesh scene with ray tracing or rasterization, area scattering properties of surfaces may be smaller than the computationally feasible pixel size or average spatial sampling frequency.

Since ray tracing and rasterization are both sampling operations, the simple naïve solution is to increase the number of pixels and/or the number of rays per pixel to achieve a higher spatial sampling frequency. For computationally efficient implementations, especially real time applications, complexity should not be proportional to the physical area sampled. A relatively small area synthetic aperture radar (SAR) ground map may be modeled accurately using detailed meshes of buildings, trees, cars, etc., but a wide area real beam ground map (RBGM) should also highlight important, aggregate terrain features such as urban centers, forests, and active highways. It is desirable that these synthetic wide area maps be computed without the proportional growth in the time to compute the radar image resulting from the increased physical land area of interest.

In computer graphics literature, texture maps (specifically normal maps and height maps) encode 3D information as approximations to more complex faceted, polygonal geometry. When 3D features, such as an entire reflecting object, are smaller than a single pixel, and each pixel/ray samples the scene at one location, then sampling the object will result in very inaccurate reflectivity since only a few points on the object are sampled. In many simulation engines, such small objects may be culled from vision completely (Lengyel, 2012). However, in radar simulation, it is common to include point targets described by a simple RCS model, usually as a function of the aspect orientation, radar frequency, and wave polarization. Since radar relies on detecting reflections from point-like targets, reflective objects should not be culled, but rather an RCS model should be used in their place (Richards, 2014).

For extended targets and ground clutter, when rasterization or ray tracing sample a point on a triangle, that point must be representative of the entire pixel area for performing numerical integration to accurately quantify the power contributed by that sample. At low ground grazing angles, radar is very sensitive to returns from artificial structures such as buildings, towers, and vehicles, but modeling every single contributing reflector becomes computationally impractical. Using an RCS function defined at every point on the terrain surface seems like a practical possibility, but RCS cannot be defined at a point since it must integrate over a nonzero area.

Alternately, the authors propose the use the differential quantity *RCS per unit surface area*, which is sometimes known as the dimensionless quantity *area reflectivity*, as a function of aspect, frequency, and polarization. In parallel, computer graphics techniques often incorporate *light maps* for real-time rendering, where all the effects of static lights on static scene objects are precomputed and stored as textures that are remapped to mesh vertices. Although this is typically not done for dynamic lighting scenarios (except for a moving viewer (Green, 2007; Öztürk & Akyüz, 2020)) it is still possible to achieve this for a monostatic radar simulation because there is only one "light" (the transmitter antenna), and that light is always co-located with the "viewer" (receiving antenna). This simplifying fact enables the creation of multiple lightmap textures, each corresponding to different aspect angles, carrier frequencies, and polarizations for real-time synthetic radar data generation.

Once the area to be covered is large enough, it is expected that fine details of dihedrals, trihedrals, and shadows will be lost, but their aggregate contributions should still be visible on the large-area ground maps. It would be impractical to model all of the mesh details of the ground features, but if they are "baked" into the ground texture at relevant aspect angles, frequencies, and polarizations, then aggregate detail such as urban centers, forests, and highways should be preserved.

PROBLEM STATEMENT

As a radar scene of interest is extended over a larger land area, constituent mesh triangles become smaller relative to the sampling area of the pixels, unless more pixels are proportionally added or more rays are traced. To prevent the explosion of complexity, the CG techniques of *level of detail* (LOD), *texture maps*, and *mip mapping* can be employed.

LOD, though sometimes encompassing the entire process of reducing scene detail, is used in this paper to refer to the relative number of facets (triangles) that a scene is composed of. Texture maps provide surface information at higher spatial sampling frequencies than vertices by projecting 2D surface information on the 3D surface of the mesh, and when properly mip mapped, are capable of aggregating surface detail into maps of lower spatial sampling frequency to be used when objects are far from the viewer or when screen resolution (or rays per pixel) are reduced. The term *mip* comes from the Latin acronym "multum in parvo" meaning "much in little" (Williams, 1983). The required modifications of these concepts to properly create radar ground maps is presented. Ground maps generated in real-time with these approximate methods should be physically accurate, especially regarding the conservation of power and the accurate relative reflectivity of physical features, and this goal is also achieved using the methods in this paper.

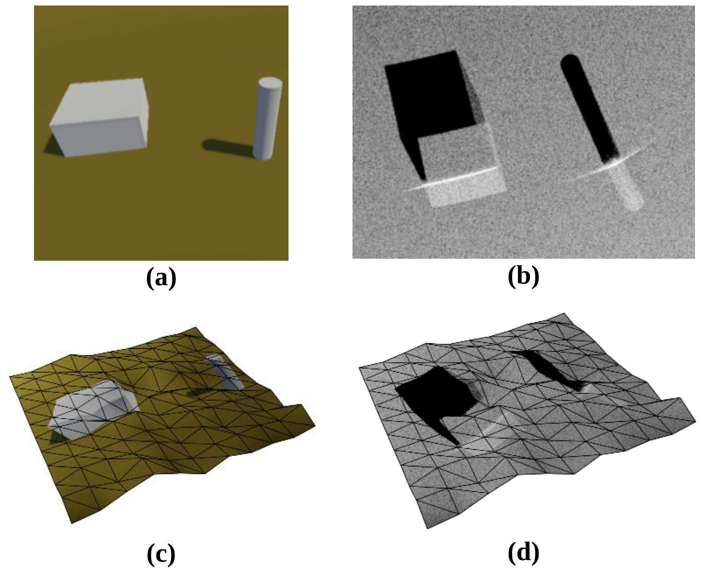

Figure 1: Simple scene showing common radar image phenomena that should be reproduced. (a) – optical view. Shadows created by external light south-east of the scene. (b) – Ideal radar ground map of scene in (a) with radar located south-east of the scene. Note the presence of shadows that extend along radar viewing direction, overlay (object features lay over the terrain in front of them), and specular multibounce intensity at dihedral corners. (c) – optical lightmap example and (d) – radar lightmap example, overlayed on a triangularized heightmap.

Figure 1a shows a simple scene consisting of a ground plane, a rectangular prism, and a cylinder. Figure 1b shows how the relative power of an ideal radar channel or "ground map" might appear on a decibel intensity plot – the phenomena of shadowing, layover, and dihedral responses are shown here. It is important to recreate these effects when using a coarser scene model. Figures 1c and 1d show what a conventional optical color map and what a texture map fitted for radar use would appear like when projected onto a triangularized heightmap mesh. The presented technique achieves the construction of texture maps for fast synthetic wide area radar ground map generation.

RADAR SIMULATION USING RASTERIZATION OR RAY TRACING

Background

The references (Hossain et al., 2020) and (Visina et al., 2021) describe algorithms for simulating radar receiver buffer data utilizing the familiar CG techniques of ray tracing and rasterization. It is shown there that to accurately recreate a radar ground map, where at least one axis of the image represents a time-based range coordinate, a weighted histogram algorithm should be used to *bin* data sampled by rasterization and/or ray tracing into respective azimuth-range bins (or Doppler-range bins in some cases). This technique reproduces many significant radar ground map features, such as shadows, layover, speckle, and Doppler cross-range shifts, as opposed to methods involving geometric transformations on world vertices (Peinecke et al., 2008), (Balz et al., 2015). As this technique is used at runtime to produce visually convincing radar ground maps, it is also used during baking. In both cases, it ensures that all features that fall into the same azimuth-range bin are summed together to contribute to the final bin value.

World (i.e. scene) information is part of the input to the radar data generator. In most CG applications (especially real-time performant simulations), a scene is composed of meshes that are defined by vertices, triangles, and texture maps. Vertices contain primarily 3D coordinates but may also contain additional properties. The most applicable second property is U-V coordinates, which are mapped normalized coordinates of a square or rectangular texture. The U coordinate spans the horizontal axis of the texture and the V coordinate spans the vertical axis, and by convention, both span from 0 to 1 between the axis limits of the texture. Each vertex can map to a texture's U-V coordinates (and potentially to additional textures), and this can be used to define fine resolution properties of any triangle surface by mapping points on the surface of the mesh geometry to points on the 2D texture. This is not adequate for a radar simulation since it is desirable to simulate broad angle effects such as dihedral reflectors and shadows. In general, it is desired to have an approximating surface reflectivity function $\sigma_0(\theta,\phi, f,P)$ that is highly customizable to allow for potentially multiple modes (i.e. maximas), and multilinear interpolation between points on a fixed multidimensional grid of reflectivity samples accomplishes this. Examples of multimodal $\sigma_0(\theta,\phi, f,P)$ can be found at the locations of box shaped reflectors in the scene whose dimensions become smaller than the spatial sampling resolution of rasterization or ray tracing – such an object will be highly reflective from 5 different viewing angles. A single conventional normal map is not sufficient to capture this multimodal phenomenon.

Traditionally, in CG, the most common texture maps are RGB (red, green, blue) color and normal vector maps. The RGB maps defines color intensity, while the normal map defines the normal vector of the surface at a sampling point. These properties can be encoded into the vertex attributes as well, but texture maps provide the opportunity for detail that is finer than the vertex spacing. Additional shading variation is provided by changes with the viewer position, which takes advantage of the normal map to compute a pixel's color in real-time. However, only an intensity map and a normal map is not adequate for radar terrain images because strong reflections can come from multiple look angles, as in the case of dihedrals and trihedrals, and this varies with polarization and carrier frequency. The pattern of shadowing that buildings, trees, and other structures create in the ground map image should also be reproduced in the texture map information. The design of multiple-input, function-approximating "lightmaps" for radar-viewed ground terrains will be covered in detail in the next sections.

Within a meshed environment, ray tracing samples the scene by simulating one or more rays traced through every pixel and determining whether there are any ray/triangle intersections, evaluating the intersection points, and sorting the hits to, typically, find the closest hit. Rays traced through the scene are allowed to reflect and refract with relatively simple algorithms. When many rays are repeatedly traced through a scene by randomizing the ray directions within the pixel, and by randomizing the reflected/refracted rays utilizing physics-derived importance sampling, the techniques have been termed *path tracing* (Caulfield, 2022; Jensen et al., 2003). These methods effectively perform the Monte Carlo numerical evaluation of the integral rendering equation. Modern Graphics Processing Units (GPUs), such as Nvidia RTX, have built-in hardware accelerated ray tracing features.

Rasterization has been a broadly applied technique, and has been accelerated by GPU parallel processors, for many decades now. Within the scope of this paper, sampling by rasterization can be concisely interpreted as a single bounce approximation of the ray tracing technique described above using a single ray per pixel. The mechanism is very different from ray tracing, however, and is typically much faster. Every triangle within the virtual camera's view frustum is projected onto the viewing screen through a perspective projection and is sampled on the equally spaced grid of screen pixels. The triangle pixel samples whose depth coordinate *z* is closest to the camera is sorted to be the "closest hit" for that particular pixel and its output makes it to the screen (or to a writable output texture contained within the GPU memory) – this technique is sometimes referred to as *z-buffering*.

It is nearly impossible to create a multi-bounce electromagnetic simulation using rasterization alone, and so it is highly recommended to use path tracing during the baking process to highlight multibounce effects such as at dihedrals, trihedrals, and to make use of the spatial anti-aliasing and special coherence that sampling with multiple rays per pixel provides. However, in real time, wide area ground maps utilizing the reflectivity texture map technique introduced here do not need to simulate multibounce effects since the terrain no longer contains specular reflector geometry descriptions such as building faces and vehicle frames – the ground is assumed smooth, while having surface reflectivity that is a numerically-evaluated multidimensional function – therefore, rasterization can be effectively used at LOD indices greater than zero for substantial resource efficiency.

GPUs and graphics APIs such as DirectX perform linear interpolation of texture and triangle sampling by default. For example, sampling a scene with ray tracing produces the intersection points of rays and triangles. Continuous-valued vertex attributes such as position, radiance, normal vector, and U-V coordinates can be linearly interpolated using the barycentric coordinates of the intersection point. Likewise, textures with U-V coordinates are typically sampled with hardware accelerated bilinear or trilinear interpolation. GPUs and graphics APIs provide the ability to automatically generate mip maps of textures and storing them in the GPU RAM for recalling in shader programs. Mip maps are progressively smaller versions of the original texture, where each subsequent texture in the series contains half the width and height in terms of texel count (i.e. four times smaller than the previous texture), and are evaluated by averaging four pixels into one at each increasing LOD index. In the current application, this is a convenient feature: once the continuous valued reflectivity texture map is numerically generated using the procedure described in this section, versions of this cover texture, designed to be used for progressively coarser ground map data requirements, are automatically generated at application load time and the correct sampling level is automatically used in rasterization. Mip maps increase the memory requirements for textures by a maximum factor of 1/3 but provide enhanced sampling performance when the spatial sampling frequency decreases (Lengyel, 2012). So, in most cases, for optimal real-time performance with visually accurate detail, the mip map level that is sampled will increase with the ground area of interest.

SOME MATHEMATICAL BACKGROUND

This section briefly introduces the concepts of radiance and radar cross-section as a means of justifying the physical consistency of the data stored in area reflectivity texture maps. Radiance, not radiant flux (power), is chosen in CG as the measure of interest when it comes to defining pixel brightness because flux cannot be defined along a ray, and ray tracing is a very powerful algorithmic tool for assessing the visibility of surfaces in a 3D scene. The radiance at the receiver is the second partial derivative of the radiant flux P_r *intersecting the effective area of the antenna aperture* with respect to the differential solid angle of arrival $\partial\Omega$ and the differential effective area of the receiving antenna ∂A_r:

$$C_r = \frac{\partial^2 P_r}{\partial\Omega\partial A_r} \tag{1}$$

In graphics, by having equal pixel areas on screen, and having nearly equal pixel solid angles looking into the scene, then simulated power will always be proportional to the pixel brightness, and the constant of proportionality is approximately equal for all pixels.

In monostatic radar literature, the reflective capability of objects is measured by their RCS. With r the distance of the antenna to the object (i.e. the range), RCS can be defined from empirical measurements as

$$\sigma \triangleq \frac{16\pi^2 r^4}{P_t}\frac{\partial P_r}{\partial A_r} = \frac{16\pi^2 r^4}{A_a}\frac{P_r}{P_t} \tag{2}$$

and typically varies with the azimuth and elevation angles of the target relative to the antenna (θ,ϕ), the electromagnetic carrier frequency f_c, and the polarization state. The dimension of RCS is area (SI units of squared meters). A_g is the total antenna surface area and P_t is the transmitted power.

For area reflectors such as terrain cover, the differential quantity of *RCS per unit ground area*, also known as *area reflectivity*,

$$\sigma_0 \triangleq \frac{\partial \sigma}{\partial A_g} = \frac{16\pi^2 r^4 \left(n \cdot l\right)}{P_t} \frac{\partial P_r}{\partial A_c \partial A_r} = \frac{16\pi^2 r^2 \left(n \cdot l\right)}{P_t} \frac{\partial P_r}{\partial \Omega \partial A_r} = \frac{16\pi^2 r^2 \left(n \cdot l\right)}{P_t} C_r \tag{3}$$

is defined and this unitless quantity can be conveniently described at every point on a surface. $\left(n \cdot l\right)$ is the dot product of the unit normal vector to a locally smooth surface n and the unit "light" direction vector l. Eq. (3) is derived by considering that a differential ground area of interest is proportional to a differential cross-section area oriented in the direction of the antenna ∂A_c as

$$\partial A_g = \frac{\partial A_c}{n \cdot l} = \frac{r^2 \partial \Omega}{n \cdot l}. \tag{4}$$

Although the actual surface to be sampled originally must include the surfaces of every individual entity in the scene within the area of interest, a reduction in computing resources is made by approximating the terrain surface as a relatively smooth one, with geometry described only by a triangularized heightmap grid, and without considering the triangles that form terrain features such as trees, buildings, vehicles, etc. Instead, our team designed a method to encode an array of radar-specific texture maps that can be numerically evaluated offline in a process similar to lightmap baking in CG. To codify the LOD measure according to conventional language as in the Unity engine, an LOD index of zero represents the highest level of detail in terms of vertex count, cover texture resolution, physics based reflectivity models, and high resolution sampling utilizing multibounce, importance sampled path tracing. Progressively greater LOD indices indicate progressively lower detail information and the utilization of mip mapped area reflectivity texture maps. Due to the "one light" model of monostatic radar, and because acceptable generation times for radar ground maps are on the order of seconds, real-time path tracing for small area ground maps is a feasible prospect. Nvidia's recent push for developers to incorporate path tracing in RTX-enabled games is evidence of this feasibility (Caulfield, 2022), as are the examples shown in this paper.

While the technique described here might seem similar to normal mapping and heightmapping techniques in CG, some distinctions are made for monostatic radar simulation because of the interest in recreating the broad angle response of certain terrain features for visual identification. Dihedral edges and trihedral corners are prevalent in human-made structures because of architectural right angles and they reflect a large amount of energy back at the radar receiver even when they are illuminated from different angles. Layover and shadows are caused by the inherent azimuth-range coordinate system of radar measurements. When no data is returned at a certain range because of terrain self-blockage (i.e. no data exists for a time segment of the received waveform), a black shadow region usually appears behind the blocking surface's response. These effects are recreated at the coarser terrain resolutions and lower pixel count texture maps provided at LOD indices greater than zero. In this paper, area reflectivity cover textures are created, whose texels represent the area reflectivity at sampled points on this surface, viewed from different angles and with different frequencies and polarizations. They are numerically evaluated through simulation (i.e. baked) using importance-sampled path tracing. They can then be recalled at runtime, and the final output is computed using multilinear interpolation between samples of the multidimensional reflectivity texture maps to create visually convincing radar ground maps of large areas.

BAKING AREA REFLECTIVITY TEXTURE MAPS

A flowchart of the area reflectivity texture map generation process is shown in Figure 2. The baking of scene data utilizes established importance sampled path tracing techniques (Caulfield, 2022; Jensen et al., 2003) and, although real-time performance is not required, the process benefits from an efficient path tracing algorithm. The generation begins with the sampling of the scene by an importance-sampling path tracer. A virtual "camera" is placed in the scene of interest, and a view frustum is established as in (Visina et al., 2021). If a cross section rectangle intersects the frustum parallel to its base, then this rectangle can be considered the view plane of the scene perspective sampling, and can be divided into $N_w \times N_h$ pixels. N_r rays are traced through each pixel at randomly perturbed directions uniformly distributed within the pixels. A maximum of N_b bounces are traced for each ray, and the reflected ray direction for each ray is randomized according to importance sampling. Material properties are manifested in the BRDF $b_x\left(\omega_i, \omega_0\right)$, and popular BRDF models such as the Cook-Torrence model with polarimetric Fresnel factors are appropriate for use in specular RF scattering models (Lengyel, 2012) and are used here. Diffuse models may be used as well, and some of these models have been documented for RF systems in (Billingsley, 2002) and (Ulaby et al., 2019) for landcovers including forests, desert, ocean, and shrubbery. However, these models are a very coarse view of the landcover information and are discrete valued due to relying on a material classification when creating texture cover maps (or when assigning the classification to meshes, vertices, or triangles). The discrete space of material classifications does not easily fit within a multilinear interpolation methodology when needing to create mip map textures of decreasing size; however, area reflectivity, which is no more than a normalized transformation of received radiance, is continuous valued and can be naturally interpolated.

The baking process first starts by setting all of the terrain objects on the area of interest, including the material cover texture, buildings, trees, etc. on a flat ground plane. It should be flat because there is no need to encode the smooth terrain height variation into the texture map – the terrain geometric level of detail will be automatically lowered for wide areas of interest, but will still be part of the rendered scene. Vertex U-V maps on the decreased density heightmaps are adjusted accordingly to sample the correct locations of the smaller textures.

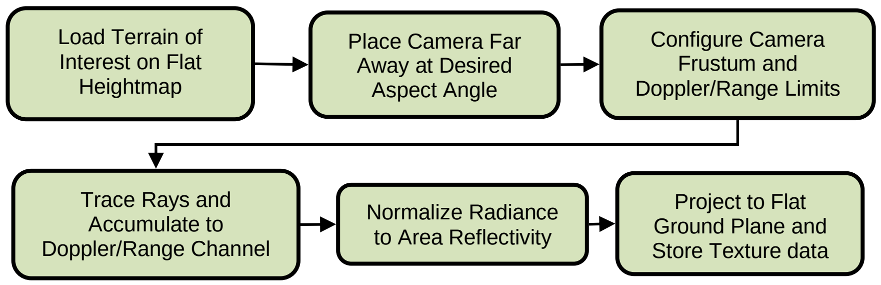
Load Terrain of Interest on Flat Heightmap
Place Camera Far Away at Desired Aspect Angle
Configure Camera Frustum and Doppler/Range Limits
Trace Rays and Accumulate to Doppler/Range Channel
Normalize Radiance to Area Reflectivity
Project to Flat Ground Plane and Store Texture data

Figure 2: Flowchart of area reflectivity texture map generation process.

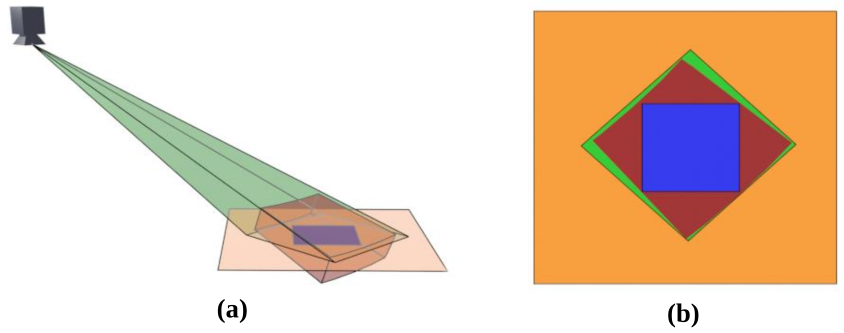

Figure 3: Scene setup for area reflectivity covermap generation for one patch at one aspect angle. (a) - 3D view with transparencies (b) - top-down projected view. Color codes: terrain (orange), the area of interest of the current capture (blue), the radar polar section (red), and the camera frustum (green). Note how the camera points down and the frustum's projection is a rotated square.

We place antenna at a very long range away from the area of interest so that the azimuth and elevation angle of the antenna to all of the points in the area of interest is approximately equal. It was explained in (Visina et al., 2021) that to faithfully simulate the phenomenon of constant radar range resolution, the camera view frustum should be properly specified such that each pixel represents nearly the same ground surface area; and, the relative level of detail (i.e. number of vertices and texture texels to be rendered per ground area) does not decrease with the range from the viewer. This contrasts with most graphics applications that simulate perspective optical cameras, where it is desirable to increase LOD index within the scene with increasing range. Instead, for radar, the LOD should be adjusted based on the expected value of the number of rays per ground plane triangle, and the viewing plane is parallel to the ground plane so that the $y = 0$ ground plane has constant depth relative to the camera's orientation. It is recommended to maintain an average of 16 or more pixels (rays) per triangle – this ensures that triangles will be visually reproduced, vertex attributes can be properly smoothed by barycentric interpolation when sampled over the surfaces of triangles, and it increases the probability that visible triangles will be sampled at least once.

Figure 3 shows a 3D view and a 2D top-down view of the scene setup for generating an area reflectivity texture map for a single patch in the center (colored blue). The radar is located north-east of the area of interest. Note that, logically, the blue patch (the current area of interest) must be contained within the radar's polar slice (red), which is defined by range and azimuth limits, and this polar slice must be contained within the camera's frustum (green). Because of the camera's downward pose and the frustum being shifted using an oblique frustum/lens shift operation, the frustum's projection on the nearly flat terrain is a rotated square and this ensures that pixel projections on the terrain are all approximately the same area.

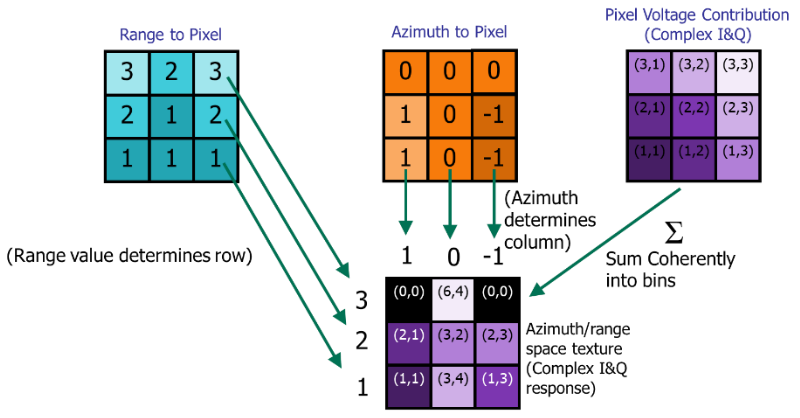

Figure 4: The weighted histogram algorithm to create radar receiver data. Output bins (bottom) are composed of the sum of complex amplitude samples (top right).

Radar receiver data is typically complex valued, and simulated echo waveform values from the scene must be summed into a set of range-azimuth or range-Doppler bins (i.e. a 2D texture with two real values per texel). Figure 4 shows an overview of the weighted histogram algorithm that describes how range and azimuth information for each ray can be used to find the location in the final data where to add the contribution. This algorithm is used at baking as well as at runtime.

The elevation angles sampled span angles greater than zero and less than 90 degrees, and the azimuth angles span angles around the area of interest. The number of frequencies and polarization states sampled will be application specific, but it should be noted that since there are only 4 possible polarization states (VV, HH, VH, HV), variation with polarization state can be encoded into the 4 channels of every texture as the GPU's architecture maximizes available RAM when all 4 channels of textures are used (RGBA textures contain red, green, blue, and alpha channels).

For every closest hit ray-triangle intersection, including those from multibounce reflection rays, the reflectivity contribution must be summed into appropriate azimuth-range polar coordinate bins – see Figure 4 for an overview of the process. The goal is to rearrange camera sampling data into range/azimuth bins similar to radar receiver buffers. Like the RB and SAR ground mapping projections, the resulting azimuth-range samples are then re-projected back down to the ground plane in Cartesian coordinates. Using this technique, every point on the flat ground plane now contains a measure of its area reflectivity as would be seen by a radar system in varying conditions. In the real-time end user radar data generation process, in-phase and quadrature (I&Q) data must be generated to simulate radar receiver buffer data with phase information and to reproduce phenomena such as speckle. Coherent summation of complex values is used in the binning process to model in-phase and out-of-phase phenomena.

Square patches of ground, for example 1 square kilometer patches, can be processed one at a time until the entire simulated terrain has had its reflectivity texture maps generated. The amount of time required to generate these texture maps is proportional to the number of samples within each dimension considered (angles, frequencies, and polarizations).

RECALLING REFLECTIVITY TEXTURE MAPS IN REAL-TIME

As mentioned previously, once the triangles in the scene are, on average, smaller than around 16 pixels after perspective projection, it is not advisable to continue having so many of them in the scene. The level of detail of the terrain heightmap triangles should be accordingly coarser, and the surface detail (material cover texture map, buildings, trees, vehicles, etc.) should be abandoned and the baked reflectivity textures described above should be projected on to the terrain and sampled in a pixel shader or in a single bounce ray tracing shader. At every sampled point on the terrain, its x and z ground plane coordinates determine which maps should be recalled and the u

and v coordinates of the cover texture to be sampled. The sampled value will be the unitless area reflectivity $\sigma_0(\theta, \phi, f, P)$ obtained from the proper mip mapped texture in the array and trilinear interpolation is used to interpolate between the fixed azimuth, elevation, and frequency grid points. Within the shader program, this quantity is then converted to radiance using

$$C_r = \frac{P_t}{16\pi^2 r^2 \left(n \cdot l\right)} \sigma_0 \qquad (5)$$

To produce the desired I&Q receiver buffer data, the radiance must be converted to the square root amplitude domain and phase must be assigned. A simple but effective model for phase information is uniformly distributed phase at each sample, with amplitude scaling perturbation according to a Reyleigh distribution (Richards, 2014).

As in the baking process, the final steps in the final I&Q data output is to bin the data to azimuth-range or Doppler-range bins, then to reproject the magnitude of the I&Q data (or power, or relative power in dB – user choice) back onto a ground mesh for final presentation.

EXAMPLES

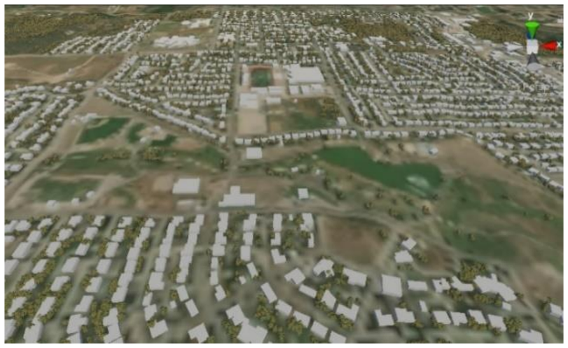

Figure 5: Optical view of example scene, showing an RGB satellite image texture map (unused in radar), buildings, and trees.

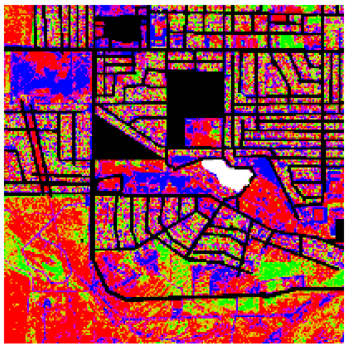

Figure 6: Material classification texture map for example scene. Black = concrete, green = trees, red = grass, white = water, and blue = dirt.

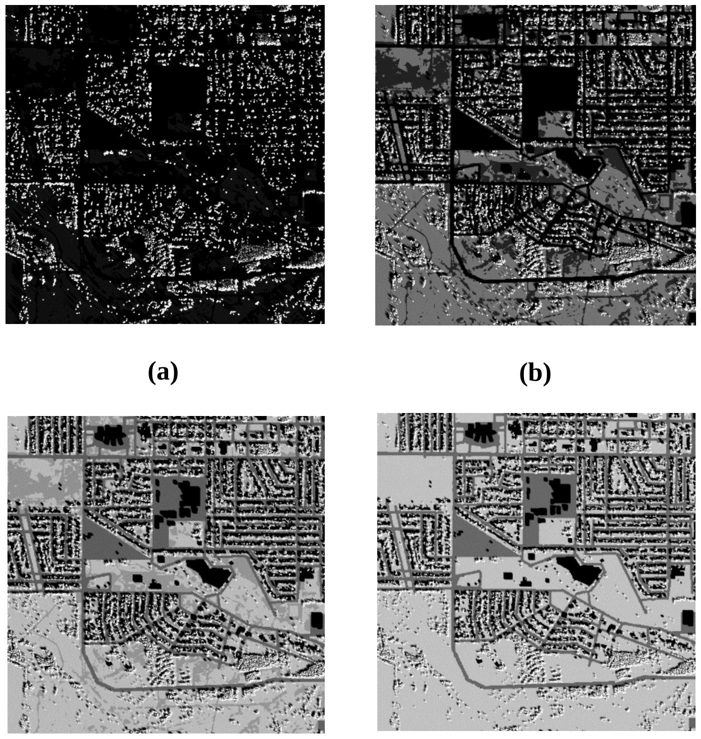

Figure 7: Example area reflectivity texture maps from single azimuth angle (antenna located diagonally southeast of the scene). (a) 5 degrees elevation, (b) 13 degrees elevation, (c) 21 degrees elevation, (d) 29 degrees elevation. Notice that the terrain heightmap is flattened for this baking process.

Examples of the area reflectivity texture map method are presented here. A scene composed of a terrain heightmap, a material classification texture map, simple building structures, and trees is shown in Figure 5. The terrain surface materials are modeled as diffuse according to (Billingsley, 2002; Ulaby et al., 2019) – see Figure 6 for material classifications. The building surfaces, tree leaves, and tree bark are modeled as Cook-Torrence specular materials with roughness factors of 0.1, 0.08, and 0.6, respectively. The conductivity parameters are 10^5, 10^2, and 5×10^{-3} Siemens, respectively. Relative permittivity parameters are 10^5, 50, and 200, respectively. The land area of interest is 2 km by 2 km: the heightmap contains 513x513 vertices and the material classification map has 4096x4096 texels. There are 70,928 trees with 2257 vertices each, and 2082 buildings with an average of 30 vertices each. This results in a scene with approximately 160 million vertices. $N_r = 100$ and $N_b = 8$.

For each look of the scene in the area reflectivity map generation process, it takes approximately 5 seconds to generate using an NVIDIA RTX A3000 GPU. A collection of 8 azimuth angles around the scene at the elevation angles 5, 13, 21, 29, and 37 degrees are taken from 1000 km away, resulting in the generation of 32 1024x1024, 4 channel floating point textures (537 MB). Since this is for demonstration purposes, only a single 10 GHz frequency was captured. After removing the buildings and trees, the number of vertices in the scene is reduced to 263,000 and only rasterization is used to generate the final image – consequently, the rendering time is reduced to the order of milliseconds. See Figure **7** for the baked area reflectivity lightmaps and Figure 8 for final ground map outputs comparing the full detail rendering with the rendering using the dynamic area reflectivity texture maps. The final images are taken from a 20 km slant range to the center of the image at 8.6 degree elevation and 130 degrees azimuth left of north, using a sum of all polarization contributions. The final ground map outputs are colored green to resemble cathode ray tube scope displays and a simple low pass filter is applied to blur the image, which accentuates strong reflective returns. Both images span the same 50 decibel range from black to white. It can be visually observed that there is very strong correlation between the output of the two methods. Since the area reflectivity texture maps require relatively few vertices and triangles (around 0.16% of the number of vertices and triangles of the full resolution scene are used in the example), and since multibounce ray tracing can be replaced with rasterization or single-bounce traces when using these texture maps, it becomes clear that they are a strong solution to the wide-area radar ground map data explosion problem that this paper first described.

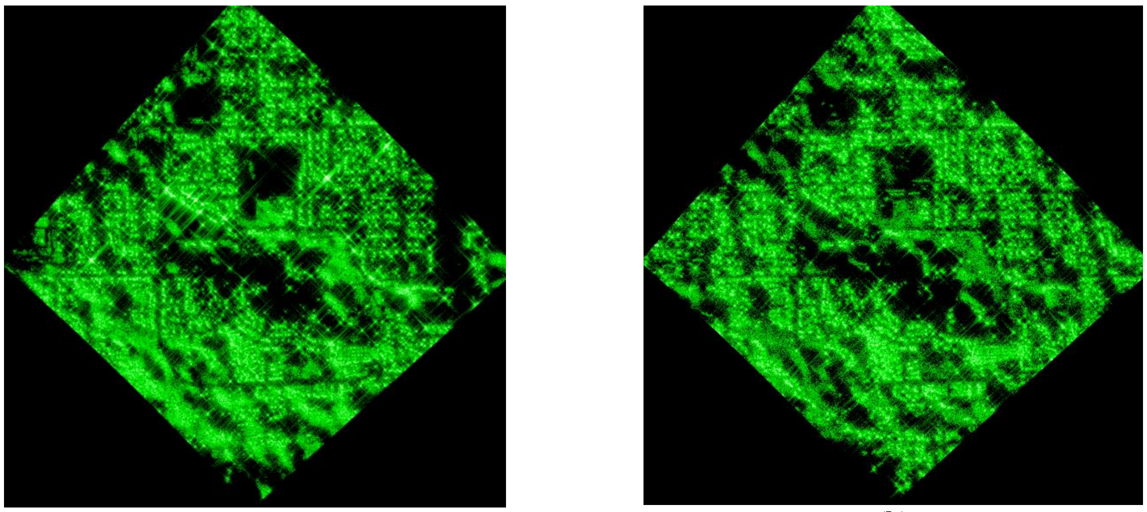
(a) **(b)**

Figure 8: Radar ground map final outputs (50 dB scale): (a) – Full quality path-traced output, (b) – Generated with rasterization and area reflectivity texture map. Post-process low pass image filtering is used to accentuate bright areas. Notice that although the reproduction reduces the accuracy of the exact image slightly, overall terrain patterns, especially those from building and terrain self-shadows, are highly correlated with the full quality rendering.

CONCLUSIONS

This paper presents a novel and straightforward technique to solve the problem of data and processing growth requirements when generating visually convincing synthetic radar ground map data of wide areas of interest. Although it is desirable to show detailed ground features, such as strong reflections and shadow patterns from urban centers, it is not desirable to represent the geometry of every contributing object such as buildings, trees, and vehicles since their size is probably smaller than the spatial sampling resolution of the underlying rasterization or ray tracing schemes. With a summarizing reduction in the scene information that is input to the radar data generator, visually convincing data can be generated without the need to increase the number of rays or pixels proportionally to the area of interest. A numerical solution to this problem was presented here. The new method bakes the area reflectivity of the terrain offline, using binned angle-range data, assuming all terrain features such as its material cover classification map, buildings, trees, and other objects are infinitesimally tall above the terrain. This method is similar to normal mapping, heightmapping, and dynamic lightmapping techniques employed in CG for similar purposes but is tailored to the needs of monostatic radar simulation. By baking versions of the area reflectivity map on a fixed grid of aspect angles and carrier frequencies, linear interpolation can be used within shader programs to compute the reflectivity of a ground point in real time without having to load the memory and computational resources further as the area of interest increases. This enables multimodal reflectivity functions to be encoded into the texture maps that effectively approximate the clutter patterns of dihedral reflectors, shadows, and overlay effects. A data packing strategy for encoding the variation with polarization state in texture maps using all 4 channels of RGBA textures was presented. Example texture maps and final radar ground map results were generated using a full detail scene and using the efficient area reflectivity texture map technique introduced here, and these were shown in the example results. The mathematics and examples show that, at the expense of the memory required to maintain the area reflectivity texture maps, radar ground maps generated with the dynamic area reflectivity texture map are highly correlated to their high-fidelity, full detail, path-traced simulation counterparts but are computed in only fractions of the time.

REFERENCES

Balz, T., Hammer, H., & Auer, S. (2015). Potentials and limitations of SAR image simulators – A comparative study of three simulation approaches. *ISPRS Journal of Photogrammetry and Remote Sensing*, *101*, 102-109.

Billingsley, J. B. (2002). *Low-angle Radar Land Clutter: Measurements and Empirical Models*. IET.

Caulfield, B. (2022, March 23, 2022). What Is Path Tracing? *NVIDIA Blog*. https://blogs.nvidia.com/blog/2022/03/23/what-is-path-tracing

Green, C. (2007). *Efficient self-shadowed radiosity normal mapping* ACM SIGGRAPH 2007 courses, San Diego, California. https://doi.org/10.1145/1281500.1281664

Hossain, S., Willis, A. R., & Godwin, J. (2020). Hardware-accelerated SAR simulation with NVIDIA-RTX technology. Algorithms for Synthetic Aperture Radar Imagery XXVII,

Jensen, H., Lecturers, J., Arvo, Dutre, P., Keller, A., Owen, A., Pharr, M., & Shirley, P. (2003). *Monte Carlo Ray Tracing Siggraph 2003 Course 44*.

Lengyel, E. (2012). *Mathematics for 3D Game Programming and Computer Graphics* (Third ed.). Course Technologt PTR.

Öztürk, B., & Akyüz, A. O. (2020). Realistic Lighting for Interactive Applications Using Semi-Dynamic Light Maps. *The Computer Games Journal*, *9*(4), 421-452. https://doi.org/10.1007/s40869-020-00116-2

Peinecke, N., Döhler, H.-U., & Korn, B. (2008). Simulation of Imaging Radar Using Graphics Hardware Acceleration. Proceedings of SPIE - The International Society for Optical Engineering,

Richards, M. A. (2014). *Fundamentals of Radar Signal Processing*. McGraw-Hill Education.

Ulaby, F. T., Dobson, M. C., & Álvarez-Pérez, J. L. (2019). *Handbook of Radar Scattering Statistics for Terrain*. Artech House.

Visina, R., Smith, C., Kirk, D., & Carter, G. (2021). *Advanced Real-Time Radar Simulation: Maintaining Range Resolution for Large-Area Ground Maps* I/ITSEC 2021, Orlando, FL.

Williams, L. (1983). Pyramidal Parametrics. *Computer Graphics*, *17*(3), 1-11.