Physics-Based Real-Time Dynamic Generation of Temporal Energy Maps for Virtual Training in Realistic IR Sensors Simulators

Harleen J. Lappano
Cornerstone Software Solutions, Inc.
Orlando, FL
hlappano@cssflorida.net

Dr. Joseph T. Kider Jr.
Institute of Simulation and Training
Orlando, FL
jkider@ist.ucf.edu

Mark Faulk
Cornerstone Software Solutions, Inc.
Orlando, FL
mfaulk@cssflorida.net

ABSTRACT

The evolution of effective and efficient synthetic sensor views of virtual environments to plan and execute training activities has lagged behind warfighter needs for *Command, Control, Communications, Computers* (C4) *Intelligence, Surveillance and Reconnaissance* (ISR). We present a novel real-time system to render realistic visible and infrared (IR) surveillance and reconnaissance training scenarios. This simulation accurately generates dynamic temporal energy maps during runtime to produce imagery for various sensors and effects to create a realistic visualization experience for training. The temporal energy maps are mappings that represent the amount of thermal energy accrued over time for a given pixel on any surface form. Our system utilizes automatic material classification and a spectral material database to facilitate realistic synthetic environments under different sensor spectral band requirements. This work describes our approach to generate convincing material mapped terrain and objects. We present runtime algorithms that store temporal spectrum energy in a physically-based system to simulate temporal thermal behavior throughout the day accurately. This approach is capable of accurately rendering large high-resolution geospatial imagery. We describe our work to demonstrate the feasibility of creating real-time temporal radiative heat transfer in a commodity game engine that provides essential detail to phenomenological sensor signatures and effects. The resulting prototype delivers robust training capabilities with a detail level that allows for better training and mission planning than traditional techniques. Additionally, synthetic sensor data can be generated from this approach and can be used to train machine learning algorithms that require large datasets where imagery is difficult to acquire.

ABOUT THE AUTHORS

Harleen J. Lappano is a Senior Software Engineer at Cornerstone Software Solutions, Inc. in Orlando, Florida. Ms. Lappano has over 12 years of experience in software development in Modeling, Simulation, and Training for the DoD. Such expertise includes Artificial Intelligence, Serious Games, Computer Generated Forces (CGF), and Semi-Automated Forces (SAF). She has led Cornerstone's Air Force STTR Phase II work and delivery: Development Lightmap Rendering Technology to Advance Infrared Simulation Capabilities, aka Temporal Energy Map Generation.

Dr. Joseph T. Kider Jr. is an Assistant Professor at the Institute for Simulation and Training (IST) at the University of Central Florida (UCF) in Orlando, Florida. Dr. Kider has over ten years of research and numerical simulation experience spanning light transport problems, such as physically-based illumination and material appearance.

Mark Faulk is a Systems Engineer at Cornerstone Software Solutions, Inc. in Orlando, Florida. He is currently the Lead Systems Engineer for the Army Synthetic Environment (SE) Core program. Mark has over 30 years of experience in extensive systems development and more than 20 years of experience with simulation and training systems architectures, systems engineering, and terrain generation.

Physics-Based Real-Time Dynamic Generation of Temporal Energy Maps for Virtual Training in Realistic IR Sensors Simulators

Harleen J. Lappano
Cornerstone Software Solutions, Inc.
Orlando, FL
hlappano@cssflorida.net

Dr. Joseph T. Kider Jr.
Institute of Simulation and Training
Orlando, FL
jkider@ist.ucf.edu

Mark Faulk
Cornerstone Software Solutions, Inc.
Orlando, FL
mfaulk@cssflorida.net

INTRODUCTION

It would be beneficial to the *Modeling, Simulation, and Training* community to require high-fidelity, physically-based, and non-visual rendering to provide effective and efficient ways to simulate physical and environmental data that allows training in real-time. As reported in Modern Integrated Warfare: *How to Close the Concurrency Gap in Military Simulation Training* (2020, June 3), there exists a capability gap between the real world and the simulator, and it hinders the effectiveness of a training tool because it does not reflect accurately on the actual platform. An examination of virtual environments demonstrates that high-fidelity trainers facilitate faster skill acquisition, higher overall performance levels, and quicker training stabilization (Champney, Carroll, Stanney, & Cohn, 2017). Simulators require the right amount of fidelity that depends on the knowledge, skills, and abilities (KSA) being trained (Galanis, Stephens, & Temby, 2017). Negative training can transfer to users if the stimulus and response are different (Galanis, Stephens, & Temby, 2017); therefore training simulators should utilize the correct fidelity to help reduce negative-impact training. The definition of physically-based modeling is modeling that incorporates physical characteristics into models, allowing numerical simulation of their behavior (Barzel & Barr, 2013). The *Modeling, Simulation, and Training* community could utilize physically-based sensor models to simulate higher fidelity physical and environmental stimuli that trainers need for a positive transfer of training.

In the description of the solicitation topic for our awarded *Air Force Small Business Technology Transfer* (STTR) program, it states that "a critical shortcoming is the inability to render realistic infrared representations in real-time" (SBIR.gov, 2016). This is a prime example of the need of the Modeling, Simulation, and Training community requiring a higher fidelity system for training needs for *Command, Control, Communications, Computers* (C4) *Intelligence, Surveillance and Reconnaissance* (ISR). Our research represents the complexities addressed within two phases of this STTR award and the advances made within Phase II (SBIR.gov, 2018). This work leverages the U.S. Army's *Synthetic Environment Core* (SE Core) program's *Synthetic Imagery Processing Toolkit* (SIPT) for the creation of synthetically generated, "artifact-free" visible and material mapped imagery. Our work was to auto-generate *Temporal Energy Maps* (TEMs), leverage a leading commercial game engine to support high-fidelity infrared (IR) sensor views, and help enable rapid response to requirements of future flight. The TEMs are mappings representing the amount of thermal energy accrued over time for a given pixel on any surface form. During Phase I, emphasis was given to traditional lightmaps that incorporated pre-baked TEMs, which only allowed static background IR views. This approach provided limited moving objects, moving model interactions, and only simple scenarios that could not provide actual training to warfighters (Kider Jr., Faulk, Moore, Barriga, & Holt, 2018). In Phase II, the team delivered a fully functional prototype to generate and utilize synthetic imagery, material maps, and dynamic run-time TEM generation to render high fidelity real-time infrared (IR) sensor views. The prototype provides the capability to rapidly generate lightmap-based models to enhance infrared capabilities in game engines/image generators to support C4ISR personnel training.

Runtime Sensor View Challenges

Modeling and simulating non-visual spectrum such as infrared (IR) can be very complex to generate for high-fidelity training. Infrared is electromagnetic radiation (EMR) with wavelengths longer than visible light. The

wavelengths viewed in a visible light spectrum show an increase in temperature from blue to red (Butcher, 2016). The appearance of objects to infrared sensors is known as an infrared signature. As infrared light, IR is often used concurrently with infrared signature to show apparent temperature differences at the sensor. Infrared signatures depend on many factors, including the object's size, temperature, the reflection of external sources from the object's surface, and emissivity (Chopra, 2020). Simulating and rendering to produce realistic IR involves many interactions with matter. In line with Budzier and Gerlack (2011), "if radiation transcends a body, it can be absorbed, reflected, or transmitted". These interactions build complicated formulas to form an IR signature by modeling the thermodynamics of heat transfer onto objects. In older systems, the computing capabilities required to perform higher fidelity IR were computationally expensive and sometimes non-feasible. Those systems and current ones generate static textures for models and terrain that best represent an approximation for non-visual spectrum viewing. Using static textures poses several limitations in modeling and simulating the non-visual spectrum. For non-visible rendering, having static non-realistic renderings can affect the transfer of training if the KSA requires a dynamic runtime rendering as it does in our STTR award topic (SBIR.gov, 2016). To accurately represent the non-visual components in a simulated environment, this research examines IR to detect infrared signatures and variations within a visible simulated environment.

Game engines help create this convergence of high-fidelity simulators known as serious games that can potentially increase the cognitive learning requirements with a high degree of effectiveness (Petridis, et al., 2012). These state-of-the-art game engines allow real-time realistic and dynamic global visible spectrum illumination using various physically-based techniques from lightmaps, virtual point lights, cone tracing, instant radiosity, and deferred rendering. They provide a dynamic computation of lights that change shadows and reflections based on material properties. Game engines focus only on three visible light wavelengths (red, green, blue), a transparency value (alpha), and ignoring or greatly simplifying IR and other non-visible spectra. Using physically-based modeling and rendering can help achieve this high level of realism for non-visual rendering. The physically-based models combine many components, including mathematics, numerical analysis, mechanical engineering, computational mechanics, and physics (Barzel & Barr, 2013). This research utilizes automatic material classification and a spectral material database with 80 material properties to facilitate realistic synthetic environments under different sensor spectral band requirements.

Source Data Availability and Quality

As reported by Jean (2011), Creating training simulation databases is a labor-intensive process. For example, "a typical database that covers a 5-kilometer-by-5-kilometer area of the world requires several "man-months" of effort" (Jean, 2011). In an article in Modern Integrated Warfare: *Beyond the Hype of AI for Training: Looking at Today's Challenges* (2019, April 30), there is a lack of data access, but also a competition of resources and competency. The availability of supporting data for material-based sensor simulation has lingered. Material encoded imagery and video are limited to available real-world collected geographic areas, resolution, and prevalence. The limits to these datasets increase the lack of image fidelity and user-driven variability. Creating "*what-if"* training scenarios is then limited to basic simulator techniques such as the runtime placement of the 3D model. These limitations may strain the *Modeling, Simulation, and Training* community and can impact the trainees' KSA.

The increased availability of authentic aerial imagery has provided a valuable resource for simulation and training developers. There are, however, several challenges with this approach, including the expensive step of removing unwanted artifacts such as cloud cover, shadows, and seasonal effects such as snow. Procedurally generating imagery products with correlated material maps allows creating a proper, training-capable synthetic environment, free of undesired artifacts from authentic aerial imagery. This cleaned imagery is the starting point for our prototype and enables us to generate training scenarios during various times of the day.

In addition, statically generated textures representing IR are also limited in quality and availability. One case is the inability to create thermal signatures and thermal shadow imprints dynamically. Another case challenge is that the static textures will require multiple datasets for different times, seasons, or weather conditions. Having multiple datasets also creates a strain in the production rate to deliver to training systems. The post-processing production pipeline generates these textures with limited resources, and providing the new datasets can take a while.

Overview of the Technical Approach

Our research has developed a methodology that helps the *Modeling, Simulation, and Training* community to provide higher-fidelity physics-based real-time non-visual rendering such as IR. We put together with careful thought a generation of TEMs created dynamically throughout the simulation. These TEMs are textures that model the thermal energy of an object or surface at a specific time from heat sources like the sun. Our STTR Phase II prototype utilized Epic's *Unreal Engine 4* (Epic Games, 2021) to harvest its physically-based rendering power for lightmaps, post-processing, and physically-based materials. The *Unreal Engine*'s codebase is open-source, allowing custom engine changes to adapt complex illumination effects. Figure 1 shows our high-level architectural pipeline process.

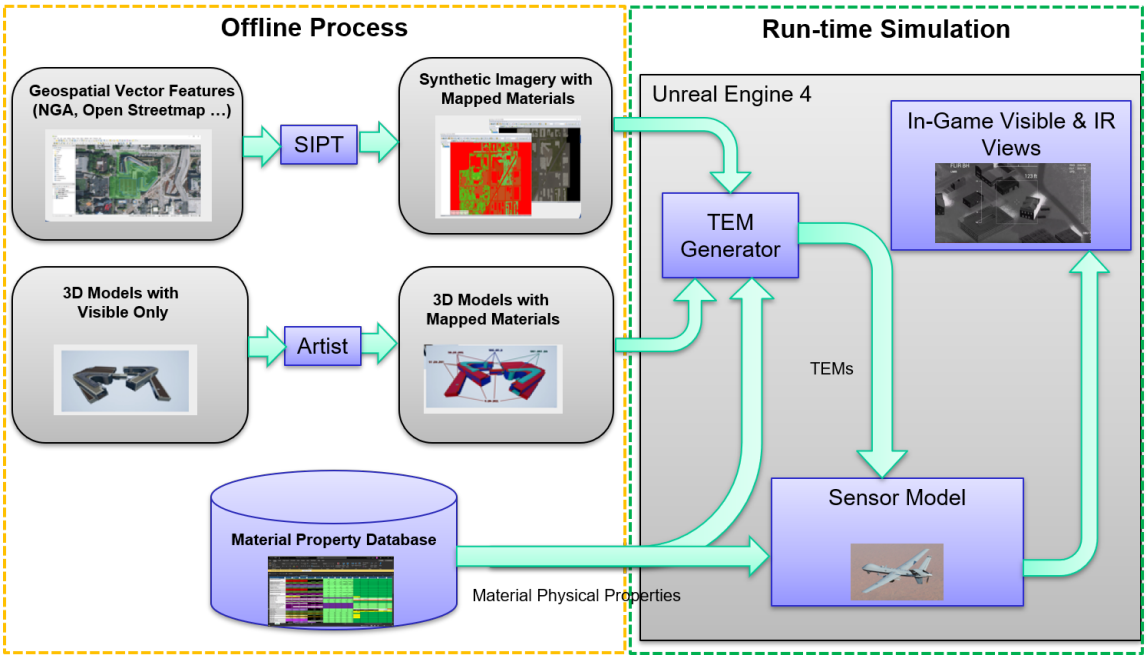

Figure 1. High-Level Architectural Pipeline Process

Two parts make up the *high-level architectural pipeline*: the *Offline Process* and the *Run-Time Simulation*. The *Offline Process* consists of pre-exercise preparations to feed into the *Run-Time Simulation* to generate TEMs for a sensor model to produce non-visual rendering. The SIPT generates synthetic imagery and mapped materials from *Geospatial Vector Features* (GVF) datasets. The *National Geospatial-Intelligence Agency* (NGA) and *OpenStreetMap* are some providers for GVF datasets. The SIPT does not process 3D Models, so 3D artists can manually generate the mapped materials for these 3D models in place of the SIPT. The mapped materials are masked textures used in the *Run-Time Simulation* to identify the type of material at a given pixel.

Another part of the *Offline Process* is the *Material Property Database* which links these specified materials via a defined index and allows the *TEM Generator* and the *Sensor Model* to identify and read the material properties. During the *Run-Time Simulation*, the *TEM Generator* generates dynamic *temporal energy maps* based on the time of day and solar radiation transferred to each object. The *Sensor Model* can read these TEMs that provide the radiating energy needed for the calculations for IR. The following sections will explain in further detail the methodology of our research and prototype.

METHODOLOGY

Synthetic Material Masks

One of the critical parts for generating *temporal energy maps* is identifying materials. Each material produces a different IR signature behavior, and a modeling IR system needs to differentiate between material types. We recognized in computer graphics that using textures for masking purposes is a common technique and we could use this ability to identify materials within the GPU shader code. We have identified technologies that produce material masks from GIS vector feature data during our Phase I research, such as SE Core's SIPT software.

The U.S. Army SE Core program developed SIPT to generate imagery from GIS vector feature and elevation datasets (Toth, Ramos, Hale, & Kehr, 2016). It essentially takes attributed lines and polygons representing the geographic features, such as roads, bridges, grasslands, forests, and water, and automatically "paints" imagery with texture brushes to create color or black & white imagery. It can also "paint" a correlated material map, similar to the output from image material classification.

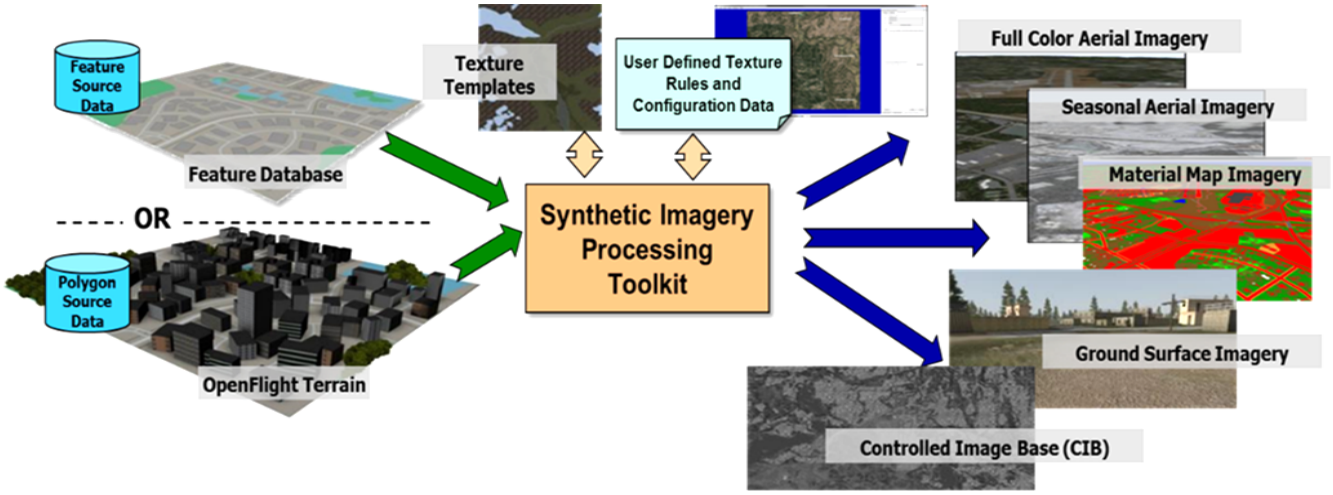

Figure 2. Synthetic Imagery Generation Processing Flow

For our prototype, we utilize the SIPT to generate aerial imagery and a correlated material map. End-user advantages to incorporating SIPT include:

- High-quality per-pixel material mapped imagery with no dependency for pre-flying the area to collect imagery.
- Provides the capability to generate image resolutions to match training needs, including an ultra-high resolution to support sensor zoom.
- Generated imagery is free of artifacts such as clouds, time of day shadows, and vehicles, leaving the runtime free to develop as desired for the training exercise.
- The end-user team can modify the GIS data, such as adding a new building based on new intel and "what-if" scenario training and re-generating a specific imagery patch.

The outputs generated by SIPT used in our prototype are full-color aerial imagery and the material mapped imagery. The material mapped imageries are the important textures needed to create TEMs. These material maps or masks are used by the TEM Generator and sensor model to identify the type of material within a given pixel.

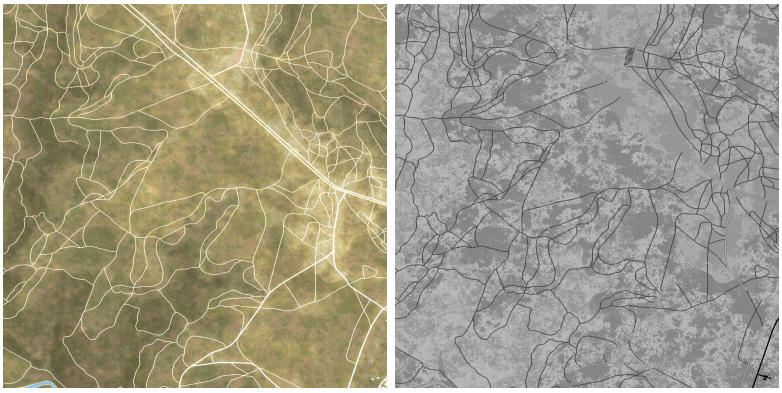

Figure 3. Example of a terrain tile SIPT output. Synthetic imagery on left and material mask map right. (Brightness adjusted for viewing purposes)

The SIPT software is only able to produce terrain-related features. Adding an artist in the loop was required to create a close representation of material mappings for 3D models. Since 3D models use textures and UV mappings similarly to terrains, adding a material map with the same UV positions is feasible. An artist would essentially copy the original out-the-view (visible spectrum) texture used and repaint to identify the different physical materials that the 3D model may contain. Figure 4 and Figure 5 are examples of material maps created by a 3D artist.

Figure 4. Example of a 3D building model. The left image is the 3D view, the middle image is the visible texture, and the right image is the material mapped texture. (Brightness adjusted for viewing purposes)

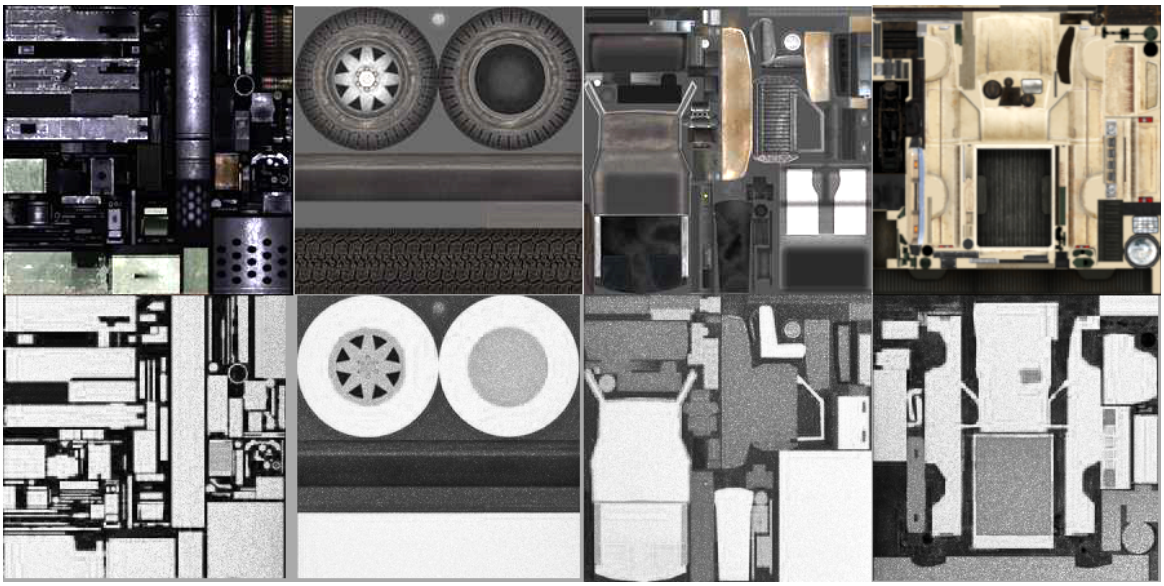

Figure 5. Example of 3D moving model textures. The visible textures are the above images, and the material mapped textures are the below images. (Brightness adjusted for viewing purposes)

Material Properties and Material Database

Physically-based modeling for visible and IR sensors requires materials with accurate optical properties for proper electromagnetic radiation interactions to produce simulations with high-quality realism. As the illumination changes, objects and material appearance and cumulative TEMs also change. Materials also respond differently to different spectrums of light. An intuitive example would be a car with a shiny metallic surface. The shiny metallic surface would specularly reflect the most visible light and absorb IR spectrums capturing more energy. Creating a plausible material database with their spectral properties is a must to create an accurate physically-based model of these sensors for practical training. With the current game engine technologies, such as *Unreal Engine 4*, these engines provide a mechanism to model this electromagnetic radiation within the engines' physically-based material system.

In our efforts during our Phase II STTR, we built out a *Material Property Database* in *SQLite* (SQLite Consortium, 2021) format to store all material spectral properties for a sample of 80 materials. The *SQLite* format provides the system RAM saving, fast queries, and modifying data with ease. As an initial basis for our material properties, we acquired the NAVAIR *Portable Source Initiative* (NPSI) Standard for *Material Properties Reference Database* (MPRD). We performed the best mapping of data coverage for other material properties. The *Material Property Database* contains a material index value that maps the properties to a specific material type to access its properties. There are two ways to represent the material index: a single value per pixel for single-channel material masked textures or three values per pixel for multi-channel RGB material masked textures. We decided to use single-channel (greyscale) material index masks since the SIPT output defaults to greyscale for our Phase II efforts.

Even with the ability to run fast queries using *SQLite* or equivalent database technology, we still needed to improve the reading capabilities when generating TEMs using the rendering pipeline. The most efficient way of doing this was to take the subset of materials required within the game level from the database and create a material texture. Within *Unreal Engine 4*, we can load this texture into Unreal's physically-based materials so that shaders will access this subset material database during the GPU processing of TEMs.

Modeling of Thermal Energy for Simulation

The modeling of physics-based thermal energy for simulation can be broken down into three main parts: *Capturing Diurnal Temperature Changes*, *Generating Temporal Energy Maps,* and *Modeling an IR Sensor*. Generating the *Temporal Energy Maps* is dependent on the *Capturing of Diurnal Temperature Changes*, and the *IR Sensor Model* is dependent on the *Temporal Energy Maps*. Our research and development for our STTR work describe these parts below.

Capturing Diurnal Cycle Temperature Changes

Throughout the day, solar radiation produced by the sun emits different irradiance levels where the irradiance is at its density peak around noon. As the solar radiation passes through the atmosphere and clouds, the radiation becomes filtered and scattered. The irradiance that reaches the surface is called global horizontal irradiance, which is the total amount of sunlight on a horizontal surface (Vignola, Michalsky, & Stoffel, 2017). On the surface, the surface materials can absorb heat via the thermodynamics of radiating heat transfer. Each material has physical properties that determine the amount of radiating energy absorbed, reflected, and transmitted from an energy source such as the sun. These physical properties can be unique for each material; therefore, each material has a different radiating energy curvature throughout the day. Just as these materials absorb energy, they also emit energy. The IR sensors show this infrared radiating energy in the materials; this can help sensor operators identify types of materials or spot inconsistencies in the environment.

According to Gregory (2018), A Lightmap is a texture map that stores precalculated lighting on a per-pixel basis. (section 11.1.3.3 Modeling Light Sources par. 2). *Lightmaps* in game engines are needed to help simulate the heat transfers for energy sources such as the sun. The *Lightmaps* determine what objects or parts of objects are absorbing the heat from the sun. At the same time, *Lightmaps* also identify areas that are not in direct absorption of the heat source, thereby cooling surface areas by the radiating heat loss. The cooling factor is essential to simulate shadow signatures cast by terrain or objects such as buildings and vehicles. One option in *Unreal Engine* is to build these *Lightmaps* statically during a pre-build process. However, the static generated *Lightmaps* do not contain any dynamic shadow information. One way to get the dynamic lightmap information is to read the *Light Attenuation* texture produced in the *Forward Rendering Pipeline* (Looman, 2021). However, this method no longer functions in later versions of *Unreal Engine* after 4.19. Another way is to generate

Lightmaps manually using *Scene Capture* components within the *Unreal Engine*. The *Scene Capture* component's position is at the location of the heat source. The *Scene Capturer* will take a *Scene Depth* using an orthographic view snapshot which produces a *Normalize Depth Map*. This depth map and some calculations identify where light collides with a surface object.

Further measures calculate the daily cycle temperature changes for each material and surface area with the lightmap information. Thermodynamic calculations can be used in conjunction with *Lightmaps* to determine how much addition or reduction of radiating energy accrued per measurement frame. The accumulated result is a *Temporal Energy Map*, which stores the amount of energy per pixel.

Generating Temporal Energy Maps

The main component that drives the modeling of the thermal energy for simulations is the TEM. The TEMs provide flexibility by way of having spectrum radiation rendered out. One part that makes the TEMs flexible is that the TEMs represent the amount of energy stored over time for a particular surface. This representation is generic enough to plug in different sensor models that render out the non-visual spectrum differently. Also, by generating these TEMs dynamically throughout the *Run-time Simulation*, the TEMs can produce high-fidelity "*what-if*" scenarios that the standard static versions cannot.

An example is rendering out vehicle IR signature shadows after their locations have changed. These vehicle IR signatures are prime examples of some IR signatures a UAV sensor operator may detect on a mission. Another aspect that the TEMs provide flexibility is that they can simulate the environment affecting IR modeling. Some examples that this can include are weather, atmospheric conditions, and explosions.

Our Phase II work developed a baking process that generates these TEMs for the sensor model to digest. The primary model used for generating TEMs is the thermodynamics of radiating heat transfer discussed above in the *Capturing Diurnal Temperature Changes* section. This model calculates every point on the surface of an object and returns the temperature changes over the time given. The model solution involved utilizing shaders or *Unreal Engine*'s physically-based material editor to process the diurnal temperature changes and the *temporal energy map* calculations within the GPU. The *Material Property Database* is available in the shaders as an encoded texture and has the material properties needed for the calculations of the TEMs. Figure 6 below illustrates the pipeline for generating the TEMs.

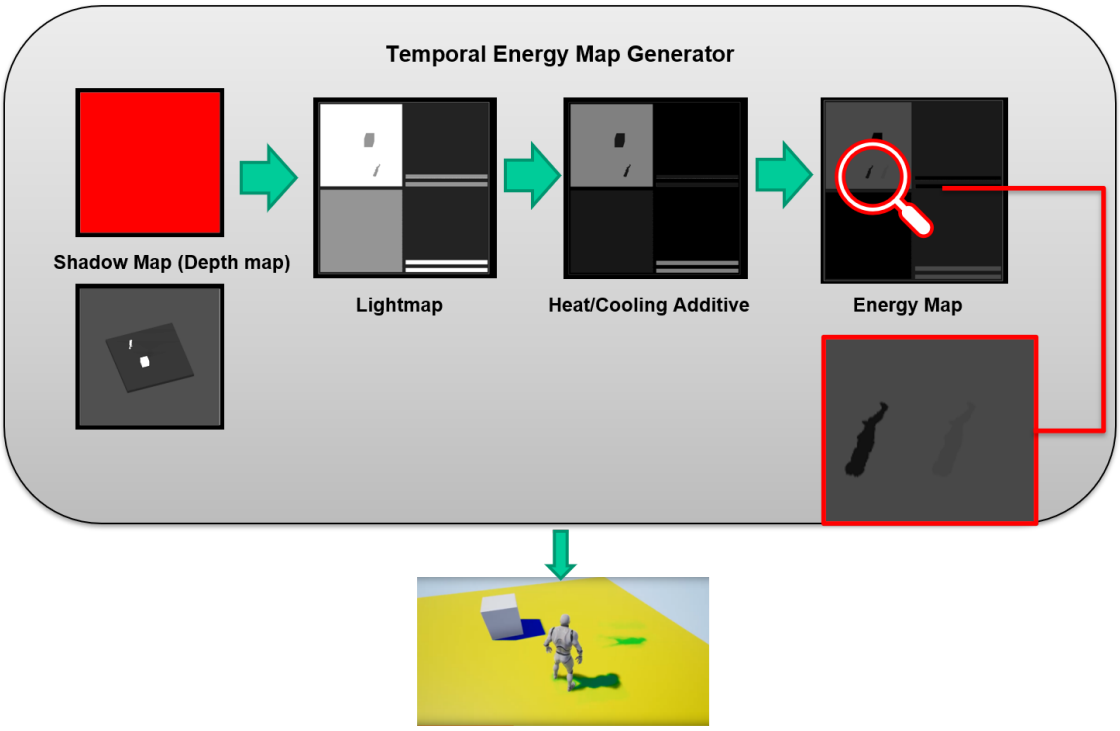

Figure 6. Temporal Energy Map Pipeline Generation

As described earlier, the *Shadow Map* or *Scene Depth Map* is used to create dynamic *Lightmaps*. The thermodynamics of radiating heat transfer calculates the delta heat and cooling, adding to the final energy map output. This calculation is represented in Figure 6 as the "*Heat/Cooling Additive*" block. The TEMs store the *Lightmap*, heat and cooling additive, and the radiating energy within each texture channel, RGB (Red, Green, Blue). Keeping the *Lightmap* in the texture provides an ability to compare a *Lightmap* from a previous frame, providing the knowledge of applied heat during the rendering frames. Also, keeping the heat/cooling additive information can represent a multiple frame pass when areas with no lighting change; it will save some rendering processing time by skipping a new *Lightmap* creation. With the *Unreal Engine*'s *Physically-based Rendering* API, we can overwrite the TEM texture at the end of the pipeline generation.

Modeling an IR Sensor

With the post generation of the TEMs, the *Sensor Model* can be written in shader code and use TEMs to produce the specific IR sensor model; such models include *Night Vision Goggles*, *Forward-Looking Infrared* (FLIR), and *Infrared Search and Track* (IRST). The TEMs provide the *Sensor Model* as an input of the absorbed thermal energy within the material object. The *Sensor Model* also has access to the *Material Property Database* via the encoded texture. This access to the *Material Property Database* allows the sensor models to add additional material sensor-specific properties. With the inputs to the *Sensor Model*, the model can calculate the emittance of radiating temperature from a material object. In addition, the *Sensor Model* can calculate the different IR wavelength spectrums such as NWIR, SWIR, MWIR, and LWIR for a specific sensor. Figure 7 shows an example of a sensor model that uses the TEM and *Material Property Database* for the sensor model formula.

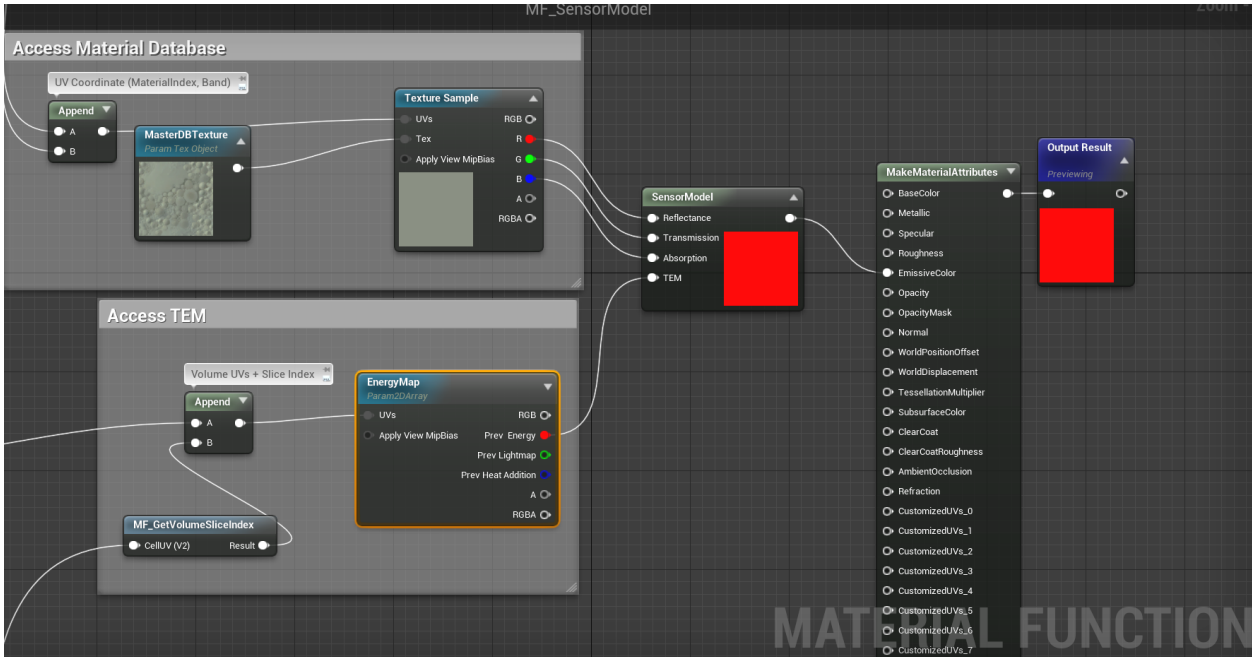

Figure 7. Example of a Sensor Model shader using Unreal Engine's Physically-Based Material Editor.

CONCLUSIONS

Our Phase II efforts proved to be a successful concept for modeling radiating heat transfers from the sun and utilizing that information for IR sensor models. The TEMs provide higher fidelity through their dynamic generation process. These dynamic physics-based TEMs are incredibly beneficial for trainers that need this level of detail. The prototype was limited to radiating heat transfers. However, TEMs could also represent energy transfers from conduction and convection since the TEMs constitute the total thermal energy stored within the objects. The TEMs and *Material Property Database* provide valuable information for any sensor model to compute the amount of radiating energy emitted from an object.

Lessons Learned

As our development progressed and reached the final stages of Phase II, we evaluated our prototype and discovered parts that need improvement. The two main lessons we have learned during this work are memory and performance. These items are as expected with any development for simulating high-fidelity. The following lends more details on these two lessons.

Memory
Concerning memory, our prototype consumed lots of RAM and VRAM. A typical gaming laptop tends to have a minimum of 16GB of RAM and a graphics card with 6GB of VRAM. We quickly consumed both RAM and VRAM when simulating a 5x5 mile terrain scenario in developing our prototype. Below are some reasons for this high memory usage.

The terrain adds quite a bit of complexity to memory. Terrains are called *Landscape Actors* in *Unreal Engine 4*. These *Landscape Actors* also contain at least one *Landscape Component*. The *Landscape Actor* can include many *Landscape Components* in a $n \times n$grid format. The *Landscapes* also have many different resolution settings and can be found in *Unreal's Landscape Technical Guide* (Epic Games, 2021). For each *Landscape Component*, our prototype also tied a $n \times n$ grid format of textures of TEMs. All TEM textures will have a pixel resolution for the quality of IR signatures needed. Depending on the settings, there can be many TEMs that represent a small area of landscape. Also, not to forget the other TEMs for every object in the world that requires IR signature behavior.

Each TEM texture has four channels: Red, Green, Blue, and Alpha (RGBA). The four-channel float texture takes up a load of memory. The red channel contains the actual energy map data, while the green and blue channels have lightmap data and the heat/cooling additive data. Unfortunately, the alphas channel is unused and is a waste of memory; but needed for the render pipeline executions within *Unreal Engine*. The lightmap data and heat/cooling additive data could be stored off in disk memory and loaded in on the next cycle of TEM baking to help reduce the memory used by 4x.

Performance Factors
The other lesson learned is in respect to the performance of baking out the TEMs. Baking out the TEMs can impact the rendering performance if there are many TEMs in the queue for baking. The additional work for generating a *Lightmap* adds an entire draw call to that pipeline. Our work added this *Lightmap* to the pipeline because the version of *Unreal Engine 4* could not access the *Light Attenuation* texture within a *Forward Rendering Pipeline*. If generating these dynamic *Lightmaps* can somehow be added to the engine's pipeline and accessed, this will save the baking process by an extra draw call for all TEMs. Also, with an increasing number of objects that require IR signatures, the number of TEMs increases. So, part of additional work and research is to harvest cloud computing discussed in the next section.

Future Work

As our efforts lead into Phase III development, our future work for this technology improves performance and increases fidelity. We also understand the importance of supporting legacy systems to help move into the transition of future technology. The following sections cover more details of *Legacy Support*, *Cloud Computing*, and *Higher Fidelity Material Properties*.

Legacy Support
The dynamic run-time creation of TEMs does not fit the architecture scope of legacy systems. These legacy systems use static textures for their IR image rendering. However, this technology can create those static textures to support legacy devices with some alterations. The TEM technology would capture a snapshot to generate a static texture in IR for these legacy devices by running simulations from dawn to the point in time of the exercise scenario. These snapshots would be part of the *Offline Process* instead of the *Run-Time Simulation* described in Figure 1 above. Having the TEMs generated in the *Offline Process* would limit the fidelity and not provide as much of the dynamics of heat transfers in real-time, which is the tradeoff of supporting legacy devices.

Cloud Computing
Cloud computing can help TEMs perform well with larger terrains and many objects with the performance factors mentioned above. The main idea would be to harvest cloud computing technology to offload the baking out of TEMs and provide a runtime client to receive a stream of these recently generated TEMs during run-time. This offload will also allow for baking out areas that are not within the range of scope of the player but still qualify for a record state of energy until needed in a

distributed environment. As with level-streaming in gaming technology, this process would be similar to adding a cloud that hosts the TEMs as a streaming service.

Higher Fidelity Material Properties

Our Phase II base framework allows for expansion to produce higher fidelity models through the material properties. Using a spreadsheet or SQL workbench as an access point to add more materials and their correlated properties made the process more straightforward. Part of the expansion for higher fidelity with these properties is via color attributes, where a specific color may affect how the material properties behave. For example, the color black absorbs and emits more heat on materials than the color white. Also, surface textures such as smoothness and roughness affect the emissivity constants. Part of a possible expansion would be to incorporate these attributes to create a higher fidelity. Another addition that we prepared for was to add more band wavelength customizations. The current framework contains average values based on different wavelengths related to IR. However, adding the entire spectroscopic IR for each material provides higher fidelity that allows a user to dial into a wavelength without modifying the database. In our prototype, we kept the door open for such an implementation to take place.

ACKNOWLEDGEMENTS

The NVIDIA Corporation donated the GPUs used for this research. The U.S. Army PEO STRI SE Core program for sharing their procedural generated imagery tool through an approved distribution agreement. This material is based upon work supported by the Department of Defense (Airforce) under STTR Grant FA8650-17-P-6859. Any opinions, findings, and conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Department of Defense or Air Force.

REFERENCES

Barzel, R., & Barr, A. H. (2013). Physically-Based Modeling for Computer Graphics. Morgan Kaufmann.

Budzier, H., & Gerlach, G. (2011). *Thermal Infrared Sensors: Theory, Optimisation and Practice*. Chichester, West Sussex, U.K: John wiley & Sons, Ltd. doi:10.1002/9780470976913

Butcher, G. (2016). Tour of the Electromagnetic Spectrum. National Aeronautics and Space Administration.

Champney, R., Carroll, M., Stanney, K., & Cohn, J. (2017). An Examination of Virtual Environment Training Fidelity on Training Effectiveness. International Journal of Learning Technology, 42-62. doi:10.1504/IJLT.2017.083997

Chopra, K. N., & Walia, R. (2020). A Short Technical Note on the IR Signatures Studies and Designing Aspects of the IR Technology Devices for Defense Aircraft. *Journal of Aeronautics & Aerospace Engineering Vol. 9 Iss. 1 No: 220*.

Epic Games. (2021). Retrieved from Unreal Engine: https://www.unrealengine.com/

Epic Games. (2021). *Landscape Technical Guide*. Retrieved 12 2020, from Unreal Engine: https://docs.unrealengine.com/en-US/Engine/Landscape/TechnicalGuide/index.html

Galanis, G., Stephens, A., & Temby, P. (2017). What is Transfer of Training, and What Does it Have to Do with Simulators? In *Fundamental Issues in Defense Training and Simulation* (pp. 307-319). CRC Press.

Gregory, J. (2019). *Game Engine Architecture (3rd Edition)*. Boca Raton, FL: CRC Press, Taylor & Francis Group.

Jean, G. V. (2011, September 1). *Real-World Imagery Sought For Military Training.* Retrieved from National Defense Magazine: https://www.nationaldefensemagazine.org/articles/2011/9/1/2011september-realworld-imagery-sought-for-military-training

Kider Jr., J. T., Faulk, M., Moore, R., Barriga, J., & Holt, J. (2018). Temporal IR Energy Maps for Synthetic Virtual Training. *Interservice/Industry Training, Simulation, and Education Conference (I/ITSEC)*.

Looman, T. (2021). *Disney's Dr. Facilier shadow effect in Unreal Engine 4*. Retrieved from Tom Looman: Unreal Engine Tutorials and Game Development Blog: https://www.tomlooman.com/disneyfaciliershadow/comment-page-1/

ModernIntegratedWarfare.com. (2019, April 30). *Beyond the Hype of AI for Training: Looking at Today's Challenges.* Retrieved from Modern Integrated Warfare: https://www.modernintegratedwarfare.com/military-training/training-effectiveness/beyond-hype-ai-for-training-looking-todays-challenges/

ModernIntegratedWarfare.com. (2020, June 3). *How to Close the Concurrency Gap in Military Simulation Training*. Retrieved from Modern Integrated Warfare:

https://www.modernintegratedwarfare.com/military-training/training-effectiveness/how-to-close-the-concurrency-gap-in-military-simulation-training/

Petridis, P., Dunwell, I., Panzoli, D., Arnab, S., Protopsaltis, A., Hendrix, M., & de Freitas, S. (2012). Game engines selection framework for high-fidelity serious applications. *International Journal of Interactive Worlds*, Vol. 2012, Article ID 418638. doi:10.5171/2012.418638

SBIR.gov. (2016, November 30). *Development lightmap rendering technology to advance infrared simulation capabilities for training applications*. Retrieved 2021, from SBIR-STTR: https://www.sbir.gov/node/1208303

SBIR.gov. (2018). *Award Details: Development lightmap rendering technology to advance infrared simulation capabilities for training applications.* Retrieved 2021, from SBIR-STTR: https://www.sbir.gov/sbirsearch/detail/1586739

SQLite Consortium. (2021). *SQLite*. Retrieved 12 2020, from https://www.sqlite.org/

Toth, R., Ramos, P., Hale, J., & Kehr, T. (2016). Aerial Imagery Unraveled. *Interservice/Industry Training, Simulation, and Education Conference (I/ITSEC)*.

Vignola, F., Michalsky, J., & Stoffel, T. (2017). Solar and Infrared Radiation Measurements. CRC Press.