Visualizing and Simulating on Whole World Terrains

Peter Swan
VT MAK
Cambridge, MA USA
pswan@mak.com

ABSTRACT

Traditionally, depending on the application, simulations and visualizations have used different and often incompatible methods to represent terrain.

The Holy Grail for visualization is perhaps the realistic representation of a whole world synthetic environment, supporting simulations from boots on the ground to strategic level campaigns, without-the-window, sensor and big picture views displayed on the desktop and in VR headsets, simulated instruments and immersive domes.

This paper discusses:

- The advantages and limitations of different terrain generation techniques that are often used to meet specific simulation requirements.
- New techniques and technology that can achieve the desired goal of a ubiquitous, multi-purpose visualization system.
- How source GIS data is loaded, stored and served up; how the earth's surface and man-made objects are modeled; and how high resolution detail can be procedurally added to the scene on the fly.
- How different terrain techniques can be mixed to support different use cases and how correlation can be achieved across disparate simulations, even with dynamic terrains.

I'll conclude by providing specific examples and applications that have been delivered using this new technology.

ABOUT THE AUTHOR

Peter Swan is Business Development Executive at VT MAK, a company of VT Systems Inc. that develops software to link, simulate, and visualize the virtual world.

Mr. Swan has 35 years of experience in the modeling and simulation industry in various technical and managerial roles. He has specialized in the development, sales, and product management of modeling and simulation, networking, and synthetic environment software products, in the UK, Canada, and the US.

Mr. Swan has held several volunteer positions in modeling and simulation related organizations including Secretary of the Simulation Interoperability Standards Organization (SISO) Conference Committee and Board of Directors; Secretary, Vice Chairman and Chairman of the Executive Committee of the National Training Systems Association (NTSA); Member of the Board of Directors of NDIA; and member of the OBW Strategic Committee and NTSA M&S Awards Committee.

Visualizing and Simulating on Whole World Terrains

Peter Swan
VT MAK
Cambridge, MA USA
pswan@mak.com

INTRODUCTION

The US Army is moving towards a single common shareable geospatial representation of the world. (1)

This paper starts by explaining why traditional terrain techniques do not appear to meet the Army's requirements, and goes on to explain new techniques that are intended to meet customer needs for running simulations on a whole world terrain.

TRADITIONAL TERRAIN TECHNIQUES

Traditional terrain creation techniques typically involve taking source data, modifying it in a terrain tool, adding hand modeled elements, and optimizing it for the target applications. This takes considerable time and effort, and rapidly becomes untenable when the intent is to create, store, and serve up a whole-world terrain database. Areas of concern for terrain generation that are not addressed by traditional methods include:

- Correlation amongst simulations. The distributed simulation terrain database correlation problem has been recognized for over 20 years as part of the 'fair fight" (2) (3) conundrum. The introduction of gaming into military training has to an extent exacerbated the situation, since game engines usually have proprietary terrain formats and were never designed to be part of a larger simulation environment. How do we ensure correlation across disparate LVC systems?
- Correlation with command and control systems. Wouldn't it be great if the simulation system used the same terrain (i.e. GIS data) as the operational system it is attached to?
- Redundancy. Many military commanders over the years have lamented how many times they have had to pay for the same database. Maybe the reasons were justified – varied geographic extents; incompatible terrain database formats; and differing levels of detail, resolution and fidelity – but many redundancies were simply due to a lack of centralized, accessible (and well publicized) terrain sources. (4) How can we build it once and use it many times?
- Accessibility. Traditionally, terrain databases are stored locally with the image generator or simulation application. Prior to running a large distributed exercise, they have to be copied across to multiple machines, at multiple sites, which takes a lot of time and creates the very real possibility of errors and inconsistencies creeping in. How can we make terrain readily available where and when needed, especially with the trend of moving simulations into a public or private cloud environment?
- Cost. Terrain and database development can be extremely costly and time consuming and is often manually intensive. Simulator manufacturers often have to build and maintain multiple databases in a single simulator – ownship, avionics, computer generated forces, radars, EO/IR sensors, out-the-window, … - a costly exercise. How do we reduce the cost of terrain database generation?
- Scalability. Terrain databases are usually built with a compromise between fidelity and size. How do we enable the vision of using the same simulations and content for training soldiers with boots on the ground This paper by no means proposes solutions to all of these problems, but certainly describes techniques and technology that are a significant step in the right direction.

NEW TECHNIQUES AND TECHNOLOGY

Given the nature of military training exercises, supporting simulations generally use or display geospatial data in some form. This could be as 2D maps, as a 3D visualization of the environment, or as the source data used for analyzing the terrain for vehicle movement, simulating sensors, communications, or other military systems, or enabling semi-automated forces behaviors. Most simulations rely on local copies of this terrain data. As the areas of interest for each exercise expand and high-fidelity geospatial data becomes more readily available, the requirements to store this data has grown exponentially. Although disk drive capacity also continues to become larger and less expensive, the process of copying and re-copying data is time-consuming and prone to errors.

The optimal solution is to store a single reference dataset and for each application to access that data when needed. The Open Geospatial Consortium (OGC) has published a number of web mapping specifications to enable transmission of geospatial data across a networked environment that can be leveraged to support this approach (5). These standards can be used to provide the foundation to deliver the imagery, elevation, feature data, and 3D models to support real-time applications, both for visualization and simulation. The data can be provided on-demand, avoiding the need to copy large terrain databases to each trainee PC, instructor station, or image generator, and can be controlled in a central location helping manage possible terrain correlation and update issues.

Many of us routinely use Google Earth, Bing Maps, or the many other web mapping applications to view locations, calculate driving directions, or discover other information about a particular location. Using these available standards rather than *reinventing the wheel*, we can use the same approaches to support real-time simulations by creating and publishing a central whole-world terrain.

However, generating whole-world terrain databases on-the-fly requires support for some specific concepts not often found in traditional simulation systems. For example:

- Coordinates, images, features, etc. are specified in a round-earth (geodetic or geocentric) system even when dealing with relatively small play boxes. Round-earth coordinates are typically not natively supported by game engines that were built for small artist-created worlds. If you see a vendor describe the "maximum terrain size" that they can support (instead of saying they can support whole-earth terrains), that's an indication that they will have a difficult time natively supporting these global terrains.

- Elevation, imagery, and features are specified in separate layers that can be independently loaded and managed, while many traditional IGs rely on pre-compiled databases where images and textures are tightly coupled with the geometry in proprietary run-time formats. The system must support direct import of source layers such as DTED, GeoTIFF and shapefiles; as well as streaming independent layers through open standards such as Web Map Tile Service (WMTS) and Web Feature Service (WFS).

- Run-time publishing into in-memory representations that are efficient for each tool's purpose (for example, a scene graph optimized for fast rendering in the IG, and a collision graph optimized for fast intersections in the CGF).

- Choosing run-time representations for features by mapping feature codes to 3D models or procedurally-generated elements.

- Run-time "modifications" to the terrain skin based on the application of various terrain features (e.g. "cutting" roads into the underlying mesh, which may be represented at different resolutions depending on proximity to the eyepoint). Many run-time systems expect the "cutting-in" logic to take place in an offline terrain tool, so do not have the capability to modify an elevation layer at run-time – much less to do it in a way that achieves correlation between the simulation engine (CGF) and IG.

- Cache data and interim results of run-time publishing the first time an area of the terrain is visited, in order to improve load-time in subsequent runs.

For many systems, adding support for on-the-fly terrains isn't just a matter of reading a new file format – it requires re-architecting fundamental aspects of the terrain system to match the modern concepts that an on-the-fly database relies upon. Such an architecture is shown in Figure 1, and described below.

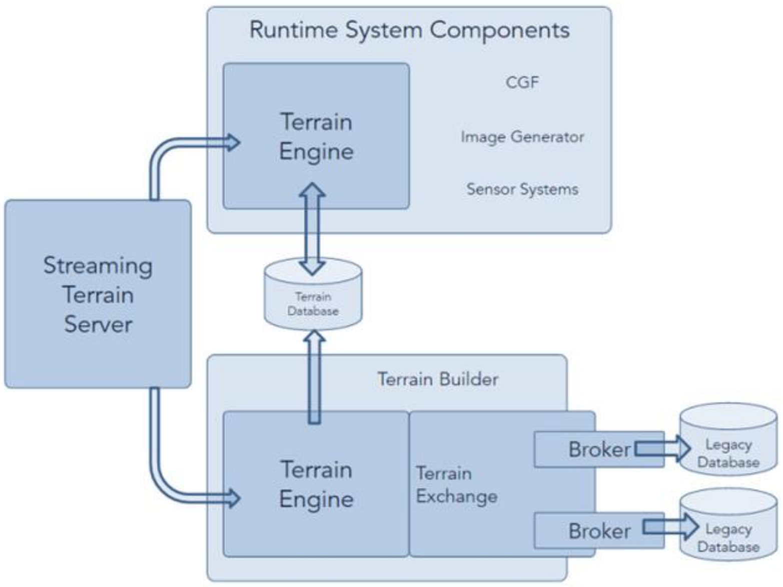

Figure 1 – Whole World Terrain System Architecture

The Streaming Terrain Server stores the whole world terrain and provides a correlated representation of the Earth, based on the WGS-84 ellipsoid, to simulation software, image generators, immersive trainers, and even remote viewers. To maximize compatibility, the Server must combine the use of streaming data services, standardized by the Open Geospatial Consortium, with widely accepted Modeling & Simulation industry/government standards to deliver a terrain solution that can create extremely high detail anywhere on Earth. The central tenet of the design is to defer the expensive content creation process to as late in the data pipeline as practical. This approach allows the essential geospatial descriptions of the Earth to be as small as possible, minimizing storage and distribution costs.

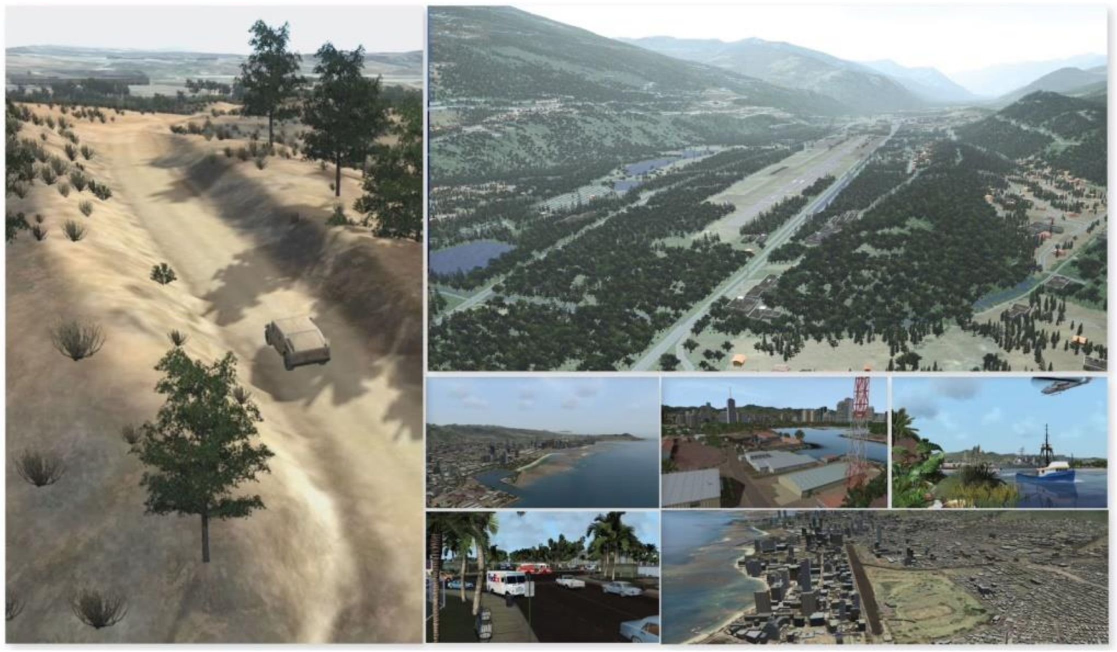

Figure 2 - Procedurally generated terrain from geodata streamed by a Streaming Terrain Server with cut in site models

At the core of the architecture is a terrain engine – a common set of terrain libraries that are embedded into the Image Generator (IG), the Computer Generated forces (CGF) application, and the virtual simulations. The Terrain Engine creates a 3D virtual terrain from geographic data and 3D models. It's the core of the whole world terrain system. It uses data from terrain databases or the Streaming Terrain Server as the source and procedurally generates super-dense virtual Earth terrain for Modeling, Simulation, and Training applications.

The Terrain Engine should have the ability to:

- Page in individual source data layers (elevation, imagery, and features) from a local disk, or stream them using open standards from a terrain server;
- Stitch in high-fidelity insets or 3D site models at run-time;
- Augment the source data with procedurally-generated detail;
- Store and present the resulting 3D terrain in multiple run-time representations that are optimized for high-performance rendering, collisions/intersections, and search (e.g. path planning or road-following).

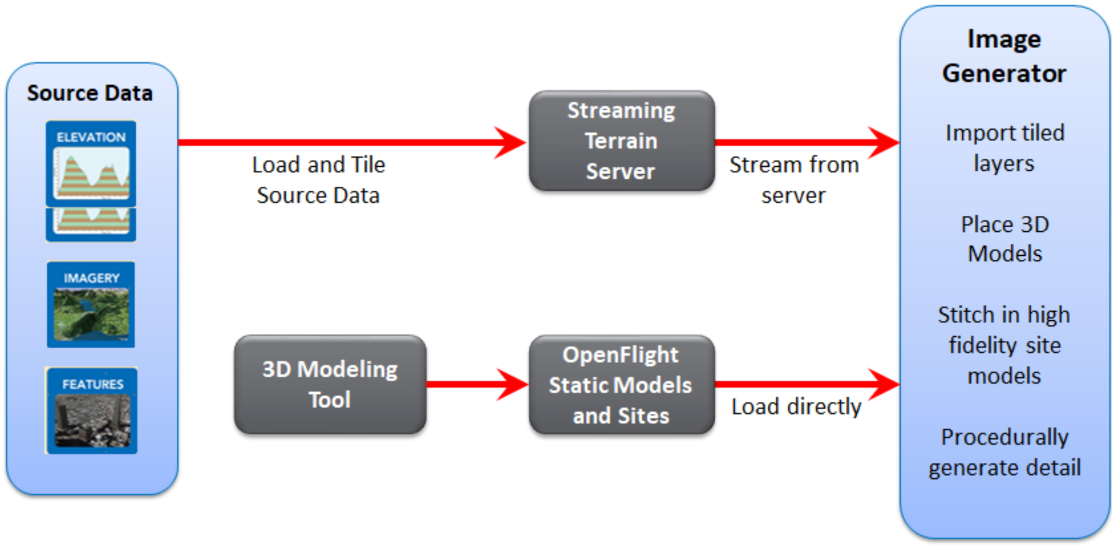

Figure 3 - Terrain Data Flow

By using the same terrain engine within all of the components, the architecture ensures perfectly consistent and correlated run-time representations of the terrain across all virtual simulators, as well as the CGF and IG.

Streaming source layers from the Streaming Terrain Server

Applications using the Terrain Engine can directly import source layers using formats such as DTED, GeoTiff, and shapefiles. However, for large data sets it makes more sense to use a streaming Terrain Server to centrally manage, tile, and stream data to the various client applications using open standards such as WMTS and WFS (5). A server can store large quantities of terrain data which can be accessed from client applications. To enable more efficient access, the server should tile the data into bite-sized chunks, but also be able to generate tiles requested by clients "on demand."

To create the finished, rendered scene for both the simulation and visualization applications, as raster and feature tiles are loaded or streamed into each client application, the Terrain Engine must:

- pass heightfields and imagery tiles directly to the GPU for rendering;
- generate triangles for a high-performance collision graph;
- place models for trees and buildings based on point feature locations;
- automatically extrude and generate textured building geometry based on areal feature footprints and attributes (heights, types, etc.);
- cut-in textured roads and rivers based on linear features; and
- procedurally generate geo-specific airports (including runways, taxiways, lighting, and signs) from source data.

The client-server approach minimizes storage requirements at each simulator, and keeps the client applications in synch as updates to the terrain are applied. The Streaming Terrain Server could be hosted locally within a simulation facility, remotely on a network, or on a private or public cloud.

Supplementing source data with high-fidelity insets

The process of automatically generating a 3D scene from streaming tiled elevation, imagery, and feature data can yield a fairly high-fidelity terrain over very large areas. However, we find that modelers often want the flexibility to author or hand-edit very-high-fidelity site models for particular areas of interest. This is particularly true for urban areas with curbs, ramps, holes, alleys, tunnels, overpasses, clutter, and other elements that are difficult to represent with regular grids of elevation and imagery data and 3D models placed on top.

Our customers typically use tools like 3ds Max® and Creator to author such sites in OpenFlight. Therefore, the Terrain Engine should have the ability to load multiple OpenFlight site models and seamlessly "stitch" them into the surrounding terrain skin grid – while retaining full correlation.

Adding procedural detail

Since a. high resolution imagery is not always available and b. flat terrain with few 3D features is not particularly realistic, a valuable feature of a Terrain Engine would be the ability to procedurally add detail to the terrain based on land-use raster data or areal features (like other layers, land use data can be tiled and served by the Streaming Terrain Server). Based on land use data the engine should be able to:

- Use "splatting" techniques to generate synthetic imagery that is geo-specific at the macro level, but geo-typical at the micro-level (e.g. apply a high-resolution grass texture to grassy areas, and high-resolution rock and snow textures to mountainous areas). As the eyepoint gets closer to the terrain, these geo-typical textures are fractally applied at higher resolutions, and are blended with the source imagery. This technique allows creation of effective imagery resolutions of 1mm per texel without the need for terabytes of expensive source imagery.

- Procedurally generate vegetation (dense trees and grass that are appropriate for the desired biome type and geographic location) in land areas that are designated as forest or fields. Hundreds of thousands of trees could be in the scene by employing advanced techniques such as generating simple tree geometry directly on the GPU using tessellation and geometry shaders.

Putting it all together

The combination of whole-earth streaming tiled source data, high-fidelity OpenFlight insets, and procedural detail allows both virtual and constructive applications to efficiently support terrains that can be used for human-character-scale simulation, vehicle driving in both urban areas and off-road, and for aircraft and UAVs - all at the same time. This approach keeps data in source form as long in the process as is practical, and generally avoids time-consuming terrain database "compilation" steps. It also allows deployment of simulators with very light data loads that fetch increased-resolution terrain only where needed.

RUN-TIME CORRELATION

Certain terrain enhancement techniques described previously may well break correlation with other systems and violate the 'fair fight", since they modify the terrain database (3). For example, procedurally generating and placing trees may create line of sight and mobility inconsistencies unless the same algorithm is used everywhere. This doesn't prevent use of visual enhancement techniques that don't impact the fair-fight, but until standards are developed for procedural terrain generation, the issue will remain.

The same holds true for dynamic terrain – something that MAK is attempting to resolve. MAK has developed, a dynamic terrain server that calculates damage caused to 3D objects in the environment (such as buildings and bridges), tracks the state of "switchable" objects such as doors and lights. To ensure interoperability and correlation between all simulations, we have developed an open specification for publishing the terrain changes over the DIS/HLA network using extensions to the HLA RPR FOM and DIS PDUs. The dynamic terrain changes are received from the network, and can be applied in a consistent and correlated way across visual systems, simulation collision graphs, and path-planning networks using a set of common algorithms. Thus, all stations - whether CGF, IG, or Instructor - will see the exact same representation of the synthetic environment at all times.

The dynamic terrain server processes events and commands that can cause changes to the terrain, and publishes these changes to the network as persistent objects via a custom DIS PDU or HLA object class. Normally a single server will manage and publish all dynamic terrain changes for the entire simulation, but it is possible to configure different servers to control the dynamic terrain for different regions of the terrain. User applications can also process these dynamic terrain changes for their own purposes, or send control messages to the server to request changes to the terrain.

Each published dynamic terrain object includes one or multiple terrain geometries that define the areas of the terrain that are being changed. For each terrain geometry, a list of key/value pairs are specified. These key/value pairs are ASCII strings that define the type of change and the new state. This could, for example, identify a new state for switch nodes in that area of the terrain, or tell the simulation how to programmatically change the terrain. The exact values keys and values used will depend on the terrain and how the server is configured.

Our intent is to move towards an open standard for dynamic terrain.

USE CASE

Operation Blended Warrior

For the last three years, VT MAK has participated in the Interservice/Industry Training, Simulation and Education Conference (I/ITSEC) Operation Blended Warrior (OBW) (6). OBW is a planned multi-year exercise to demonstrate and explore the potential for Live, Virtual, and Constructive (LVC) capabilities to maximize training, education, and testing for the defense and security sectors. In 2016, the Terrain Working Group decided to focus on identifying common source data from which Image Generator (IG) and simulation vendors could produce their final run-time products. The primary area of interest (AOI) was Southern CA, mostly focused around Camp Pendleton. A number of publicly-available US Geological Survey (USGS) datasets were identified including 1m and 1ft imagery and 10m elevation, which would serve as the basis for the terrain skin. As discussions proceeded to refine the one or two specific Military Operations on Urban Terrain (MOUT) sites that would be used to demonstrate the ground operations, it was discovered that USGS had also released 1m digital elevation models (DEMs) and source LIDAR data for much of the AOI including all of Camp Pendleton. The difference between the 10m and 1m elevation is significant, especially for ground combat operations, as shown in Figure 4.

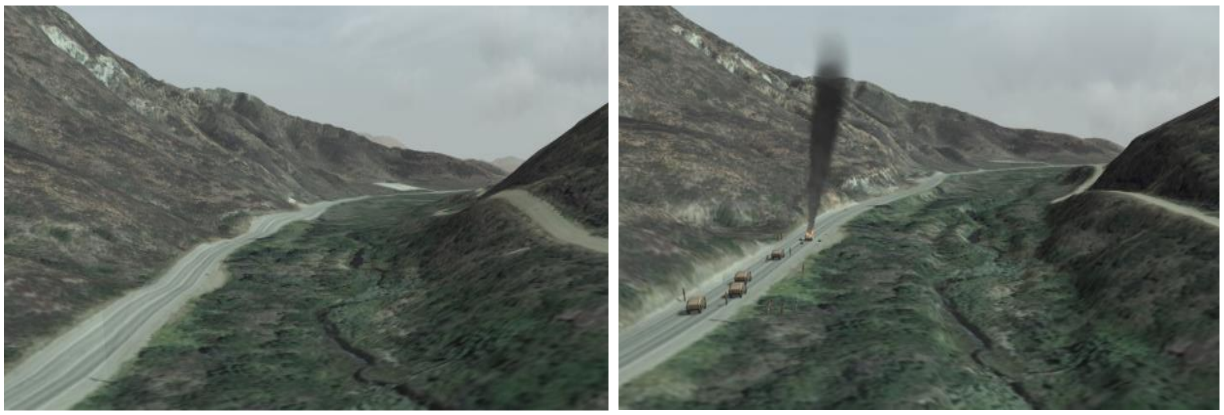

Figure 4 - A comparison of rendered 10m elevation (left) to 1m elevation (right), draped with the same 1ft imagery as displayed in VR-Vantage.

Despite being very late in the exercise preparation process, the decision was made to use the 1m elevation data for the Camp Pendleton area. Participating system integrators had to reprocess their terrains in order to use the new data. In many cases, the integrator could not or chose not to use that level of fidelity, resulting in correlation issues in the final exercise. However, because VT MAK used the streaming terrain process, the new data was uploaded to the VR-TheWorld Server, where it was composited with the existing 10m elevation data for the AOI, and 30m global data along with 10m and 90m bathymetry data. VR-Forces simulations were rerun and the terrain visualized with VR-Vantage with little effort. Figure 5 shows the results of the data streamed from VR-TheWorld.

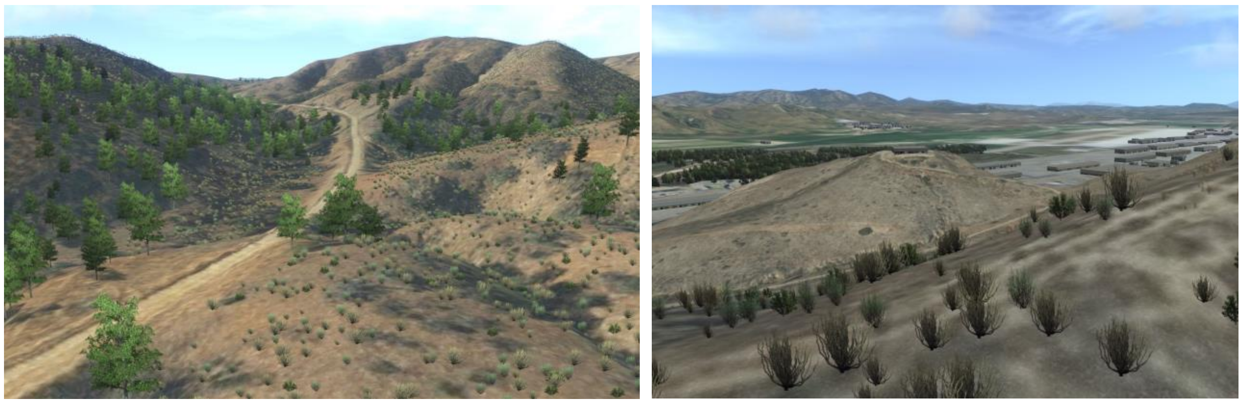

Figure 5 - VR-Vantage screenshots showing 1m elevation, 1ft imagery, and procedural vegetation in Camp Pendleton.

For 2017, the US Army was the primary sponsor, so there was a desire to make use of SE Core terrain data. The decision was made to have some vignettes reuse the previous CA terrain and expand to have other vignettes use the SE Core Northwest US (NWUS) terrain data. A subset of the NWUS for downtown Seattle, referred to as the Emerald City (EC) dataset, was made available to all participants. MAK was able to incorporate the EC 5m elevation with the existing US 10m data and add the EC procedural image along with 6” USGS HRO imagery and 1m NAIP on the VR-TheWorld server. Many EC Feature layers were also uploaded to the VR-TheWorld server, including Roads, Water areas, Tree and Shrub points, Building areas, Sidewalks, Flagpoles, Cranes, and Towers. Some layers were used as part of the VR-Forces CGF terrain reasoning, while others were procedurally rendered using a mix of SE Core-provided 3D models and existing MAK assets. These were merged with existing Open Street Map features and vegetation procedurally rendered from the National Land Cover Database (NLCD).

The resulting terrain provided a dense urban area to support the specific ground vignette, but also provided seamless incorporation with the existing terrain areas so that supporting aircraft and UAVs did not fall off the edge of the terrain. Figure 6 and Figure 7 show the results of the EC data streamed from VR-TheWorld.

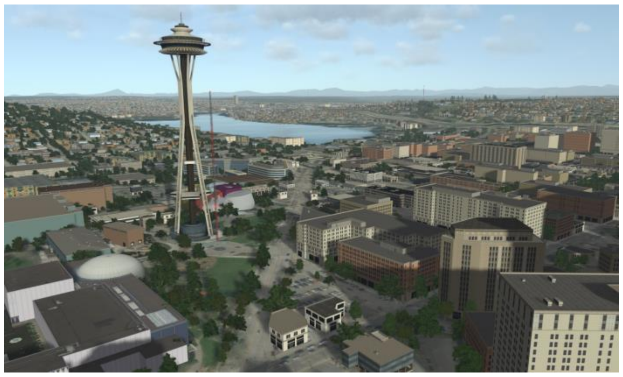
Figure 6 - VR-Vantage screenshot of Emerald City rendered feature data for ground vignette focus area.

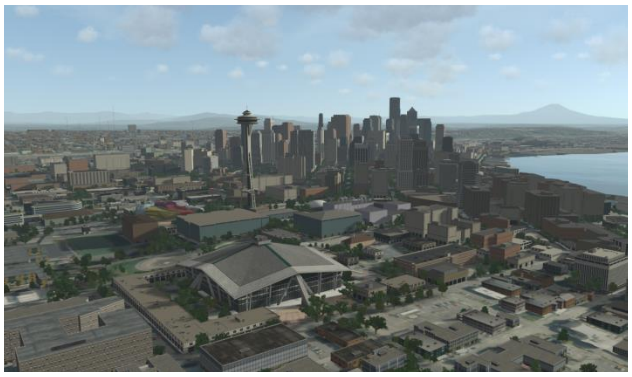
Figure 7 - VR-Vantage screenshot of Emerald City features from SE Core and surrounding area including Mt Rainier in the distance.

The final result was a whole earth terrain, with varying levels of fidelity and feature density, incorporating a number of disparate datasets.

One issue that was encountered was the mechanism that SE Core used to create geo-typical buildings. These buildings were algorithmically generated and then exported as OpenFlight models. The algorithms used produced rich, high-detailed buildings. The EC building feature data referenced these OpenFlight models to place and orient them in the proper location. However, the process of loading thousands of individual model files, each having four LODs, with many textures (which did not use efficient techniques like a texture atlas) resulted in long load times and lower performance as compared to the VR-Vantage procedurally generated buildings. Since performance was good enough, the EC buildings were used rather than having better performance but noticeable differences in the look of the buildings. What would have solved this was if the procedural algorithms to generate the models were also shared along with the data used to generate them. Feature attributes like building height, type, footprints, construction details, or other attributes are typically shared, but what is lacking is the ability to share procedural algorithms and building blocks. If these had been available, we could have procedurally generated the same buildings from the feature source rather than loading the individual models. Although this was achievable for the few thousand model files used, this would be impossible for much larger areas.

CONCLUSION

MAK has been supporting terrain agility in our simulation and visualization products for several years. New technologies are now emerging that will enable scalable, multi-domain simulations on whole world terrains. These technologies go hand-in-hand with initiatives to offer modeling and simulation as a service in public and private clouds. However, we feel that work is required to develop standards to cover procedurally generated and dynamic terrains, ensuring correlation and a fair fight across all simulation and visualizations.

BIBLIOGRAPHY

1. Synthetic Training Environment Statement of Need, STE Industry Day 9/18-19/2017.

2. *Interoperability issues for terrain databases in distributed interactive simulation.* **Schiavone, Guy A., Russell, Nelson S. and Hardis, Kenneth C.** s.l. : SPIE, 1995. Proc. SPIE 10280, Distributed Interactive Simulation Systems for Simulation and Training in the Aerospace Environment: A Critical Review, 1028008 (19 April 1995).

3. *How to ensure Fair Fight in LVC Simulations.* **Robert Siegfried, et al.** 2011. RTO-MP-MSG-087.

4. *Modeling and Simulation Terrain Database Management.* **Martin, Grant, et al., et al.** 2005.

5. Open Geospatial Consortium Standards. [Online] http://www.opengeospatial.org/docs/is.

6. *Washington, We Have a Problem: The Foundation for Live Virtual Constructive (LVC) Exercises Requires Fixing!* **John M. Kent Gritton, David M. Kotick, Gary R. Fraas, Courtney Dean.** 2016.