Realistic Geo-Specific Feature Densities in Real-Time Synthetic Environments

Brett Chladny
Renaissance Sciences Corp.
Chandler, AZ
bchladny@rscusa.com

John Femiani
Arizona State University
Tempe, AZ
john.femiane@asu.edu

Benito Graniela
NAWCTSD
Orlando, FL
benito.graniela@navy.mil

ABSTRACT

Creating a high fidelity virtual environment for flight simulators can be both expensive and labor intensive. Aircraft simulators rely heavily on geo-specific imagery to provide the trainee with a high fidelity virtual environment which is rich in two-dimensional (2D) visual cues. It is well known that the addition of three-dimensional (3D) models to this synthetic environment enhances visual cues that enable the perception of depth and motion. Alignment of 3D features with the underlying imagery is crucial to avoid visual distractions, especially at low altitudes. Constraints in hardware performance and financial budget limit the amount and quality of 3D features that can be included in the virtual environment. This paper presents an innovative automated process that leverages commonly available geospatial data sources and graphics processing unit (GPU) technologies to enhance visual cues. The design goal is to create a framework that is largely automated and able to process a wide variety of geospatial datasets including imagery, material classification, digital elevation and vector data. The result of our work is twofold. First, a generic framework for extraction and processing height and feature information from commonly available geospatial data was developed. Secondly, a run-time rendering component that is independent of the image generator was developed. This run-time component is designed to be a drop-in module that uses the power of modern commercial off-the-shelf GPUs and OpenGL 4.0 to achieve consistent rendering performance, regardless of how many features are visible in the scene. Furthermore, it is expected that this run-time technique can be used to enhance feature content in existing terrain databases and bypass current image generator system bandwidth limitations. This novel work flow and rendering approach has the potential to raise the bar for high complexity and high fidelity virtual environments for real-time training simulators while lowering overall database acquisition cost.

ABOUT THE AUTHORS

Brett Chladny has a Masters degree in Computer Science from the University of Missouri – Columbia. He began his career in 1998 at Silicon Graphics where his primary responsibility was creating custom applications for their customer base. In 2002, he took a position with MultiGen-Paradigm where he became the technical lead for the F-16 NVG and IR program at Luke AFB. In 2006, Brett began working for Renaissance Sciences on the R&D team to develop methods to improve the physical accuracy of sensor and out the window visual simulations. Brett is currently the Principal Engineer for an ongoing Navy Phase II SBIR project. He is also the project manager and lead engineer for the Deployable Sensor Scene Simulation System (DS4) application, Interservice Common Sensor Model (ICSM) software module and the IG (Image Generator) component of NAVAIR's next generation Reduced Oxygen Breathing Device (ROBD) and Carrier Approach Landing Fidelity (CALF) projects.

John Femiani is an Assistant professor at Arizona State University in the Department of Engineering where he is co-director of the Image and 3D Data Exploitation and Analysis (I3DEA) lab, and is also an organizer for the Health Engineering Applications Lab (HEAL). Dr. Femiani has professional experience in visualization. He has patents on handwriting segmentation and is CTO of VProctor, Inc. Research and teaching interests include Computer graphics, Computer Aided Geometric Design, and Computer Vision.

Benito Graniela is a Visual Display System Engineer at the Naval Air Warfare Center Training System Division Visual and Sensor Simulation Branch. Dr. Graniela has professional experience in virtual and constructive synthetic environments, visualization, modeling, simulation and visual systems. Dr. Graniela holds a patent for a virtual environment capture system, a PhD in Modeling and Simulation and a Masters in Computer Engineering from the University of Central Florida.

Realistic Geo-Specific Feature Densities in Real-Time Synthetic Environments

Brett Chladny
Renaissance Sciences Corporation
Chandler, AZ
bchladny@rscusa.com

John Femiani
Arizona State University
Tempe, AZ
john.femiane@asu.edu

Benito Graniela
NAWCTSD
Orlando, FL
benito.graniela@navy.mil

INTRODUCTION

In high quality aircraft simulators, a combination of hardware, image generation software, and geo-specific terrain is used to generate a visual scene or synthetic environment that provides the necessary visual cues for training. McIntyre and Roberts (1996) suggest that a number of factors including level of detail, field of view, resolution, ambient light levels, flicker, smoothness of apparent motion, contrast, brightness, color, accommodation, and lack of stereopsis influence the simulator's ability to represent visual cues adequately. Aircraft simulators utilize environmental models that provide an approximation of the real world but due to a number of factors, which include acquisition costs and processing limitation, it is not possible to provide all of the visual cues found in the real world. Humans have a variety of perception mechanisms which can provide information to compensate for the weakness or absence of some cues by enhancing the effects of others (McIntyre & Roberts, 1996). This mechanism helps the human visual system boost visual cues in natural scenes which lack strong cues.

Visual cues in the simulator are generated using a combination of systems which include display, image generator, and terrain database components. Visual cues allow pilots to identify terrain features, determine rate and direction of movement, rate of closure, orientation, distance from obstacles and height above terrain (McIntyre & Roberts, 1996). The pilot transforms the perceived motion in the optical frame of reference into relative motion in the internal frame of reference and applies feedback regulation to minimize errors between commanded and perceived motion. Hart (1991) identified three levels of vehicle control: inner loop; intermittent maneuvers; and outer loop. The inner loop control is responsible for control that counteracts momentary effects of disturbances on vehicle position. Intermittent maneuvers are used to avoid obstacles and outer loop control is used to maintain a planned route. In this paper, we will concentrate on the visual cues that provide support for the inner loop control of air vehicles. At low altitudes, vehicle control is primarily based on visual cues (Entzinger, 2009; Padfield, Lee, & Bradley, 2003; Hart & Battiste, 1991). Using inner loop control, pilots regulate speed, heading and altitude to maintain a safe speed and adequate clearance to obstacles, while continuing to head in the intended direction. Pilots estimate their speed by integrating dynamic visual cues from the visual scene. To estimate velocity from dynamic optical flow of features in the scene, pilots first estimate their altitude and then use this information to determine their apparent speed. In a similar fashion, pilots estimate and maintain vertical, lateral trajectories, and clearances from obstacles by monitoring the environment.

An aircraft simulator environment needs to provide enough optical stimulation so the pilot can make sense of motion from the optical flow of the surfaces in the visual scene. Optical flow is generated from the relative motion between objects in the scene and the viewpoint and is attributed to the differential angular velocities of points and edges in the visual field. Paterson et al (2004) and Gibson (1950) claim that one of the most important visual cues associated with heading is motion parallax. Motion parallax is associated with the optical flow that is produced when the observer moves through the environment. Motion parallax refers to the difference between images in the retina produced by the relative motion of objects at different distances. These patterns of relative motion are used for tasks such as directing locomotion, controlling posture balance, and steering vehicles. In experiments with pilots, Patterson et al (2004) concluded that the primary carrier for horizontal parallax information was vertical tree boundaries (i.e. vertical edges). In this sense, the presence of edge parallax cues improves heading control. The presence of trees dynamically occluding the background texture elements enhanced edge parallax information. Martin and Rinalduci (1983) also reported that pilots would intentionally fly a course that will place tree tops close to the aircraft with the purpose of calibrating their flight path. The strategy followed by pilots is to place trees in and near the fovea visual field to control level flight at low altitudes. This indicates that the presence of trees and other vertical 3D features such as buildings increases aircraft simulator low level flight fidelity as it relates to visual cues and in particular to motion parallax.

It is well known that creating a high fidelity virtual environment for aircraft simulators can be both expensive and labor intensive. The visual and sensor terrain database (VSTDB) is responsible for the 3D description of the terrain surface and cultural features. Current VSTDB development techniques include manual and semi-automatic processes executed in batch and or real-time (Pafford, 2011). Common to all these techniques is a careful balance between training needs, extents, source data resolution, image generation performance, and modeling. These parameters must be weighed against each other in order to generate a VSTDB that provides a virtual environment with the necessary visual cues in a timely and cost effective way.

The training mission drives the requirements for the gaming area and source data resolution while image generation performance typically limits the level of fidelity provided by the rendered scene. Geo-specific terrain elevation, imagery, and cultural features are incorporated into the VSTDB to provide the necessary visual cues for aircraft simulators. Modern aircraft simulators rely heavily on the geo-specific imagery in VSTDBs to provide the trainee with a high fidelity geo-specific virtual environment. In this virtual environment, natural and manmade features are typically added (manually or procedurally) as 3D models to the terrain and imagery to increase the level of detail, add realism, and provide enhanced visual cues.

Identifying which features are required in the VSTDB to meet specific training objectives is not a trivial task. Current methods are primarily driven by input from subject matter experts (i.e. pilots), IG performance, and cost. A widely used technique for controlling modeling cost is to concentrate the level of detail on areas of interest. In aircraft simulators, these areas of interest normally include airfields, landing zones, confined area landing sites, and terrain following routes. The visual cues around these areas of interest normally include landmarks, obstacles, navigational aids, lights and other natural and manmade features considered of interest by the subject matter experts. Other geo-typical components are normally included procedurally and limited by the IG performance. Since imagery is widely used, it is necessary that man-made features and vegetation be aligned with the underlying imagery in order to avoid distracting artifacts. The placement and alignment of buildings and vegetation in and around these areas can be time consuming for large areas and therefore is limited in scope. Limitations in feature content can affect the optical stimulation needed by the pilot for the accurate perception of motion.

The main goal of this research is to substantially increase the number of imagery-aligned geo-specific features in visual simulations. Achieving this goal will both simplify the development of the VSTDB and increase the amount of visual cues provided to the trainee. However, a rendering technique alone would not result in a viable solution for adding high quality features and clutter to the scene. Data beyond what is typically available in VSTDBs is necessary to support the rendering of features and clutter across the entire scene. Furthermore, this data should be stored in an IG independent format that can be easily paged as the observer moves through the scene. Lastly, using this data at run-time should not require radical restructuring of the software that renders the scene.

One of the most desirable aspects of image generation for aircraft visual simulation is consistent rendering performance. For this reason, an ideal rendering solution should maintain consistent rendering performance regardless of the feature and clutter density of a given location. Additionally, the ideal rendering solution should be capable of being abstracted into a Software Development Kit (SDK) so that it can be integrated into a wide variety of IGs with minimal effort. Furthermore, a rendering solution that could not be easily integrated into current rending software architectures would be of limited use.

This paper presents an innovative and automated pipeline approach that leverages commonly available geospatial data sources and GPU technologies to render realistic numbers of 3D features and clutter. The proposed framework generates appearance and geospatial data in a manner that is both statistically likely and consistent with the existing geospatial data or other Geographical Information Systems (GIS) information. At a minimum, height and feature information is generated. This data is stored in a raster format so that it can be easily paged by the runtime using the same approaches that already exist to page out-the-window imagery. Modern rendering techniques, such as bump maps, parallax maps, and displacement maps, allow for the efficient rendering of more complex surface detail. These techniques can provide the appearance of small extruded features on an object when it is viewed from off-nadir viewpoints. The runtime solution presented in this paper uses the generated raster data to present the appearance of realistic numbers of features while maintaining consistent rendering performance.

PREVIOUS WORK

Any solution to our stated objective requires both a data generation and a rendering solution. The following methods address the challenges associated with generating and displaying very large numbers of 3D features over large area terrains.

Google and Apple have both used airplanes equipped with high resolution cameras to capture texture-mapped 3D digital surface models over select cities. Both companies solve the problem of generating 3D models of urban environments by collecting multiple viewpoints of the same locations. Google provides a '45 degree' image feature for a number of cities, and this additional data is processed in order to generate texture mapped 3D models. Apple acquired a company called C3 Technology and has been using 3D maps reconstructed from their data using stereophotogrammetry. Similar data appears to be used by Nokia (HERE, 2013). These commercial solutions require multiple viewpoints collected from a site in order to reconstruct the area in 3D. Most of the world is covered by lower resolution imagery captured from a small number of viewpoints, and is not suitable for the high resolution stereophotogrammetry approach. These systems do offer an advantage over the proposed framework because they can represent geo-specific building facades. However, the proposed framework is capable of generating realistic 3D features given monoscopic nadir imagery and other vector data that is already available for much of the world. Additionally, the proposed pipeline fits nicely into current visual database creation and rendering work flows.

Another common approach to generate data is procedural modeling (Müller, Wonka, Haegler, Ulmer, & Van Gool, 2006; Wonka, Wimmer, Sillion, & Ribarsky, 2003). The approach has been commercialized by the company Esri into their product as 'CityEngine'. Rather than model individual buildings, the procedural approach uses stochastic grammars in order to produce random buildings. The grammars can be tuned so that the buildings generated are statistically similar to buildings in a certain area. Procedural models can be constrained to generate buildings within a given parcel, so that the results are consistent with GIS data. However, these approaches produce random buildings, and do not ensure that the geometry is aligned with existing imagery.

New technology has been developed in recent years by Diamond Visionics Corporation and others related to just-in-time database creation (Pafford, 2011). This approach uses refined source data as input to the run-time and constructs the database dynamically as the observer moves through the scene. As a result, the fully constructed large area databases neither exist on disk nor in memory at any one point in time. The in-memory database, which is rendered each frame, typically consists of geometry that represents the surface of the terrain and geometry that represents features like buildings and trees that are placed on the terrain. Much like our proposed solution, this approach significantly reduces overall database acquisition costs. However, this approach has the same inconsistent rendering performance of more traditional image generators.

Significant work has been done to add surface detail to flat polygons in visual simulations. Most of this work has been focused on adding detail to objects close to the observer, such as adding the appearance of rocks protruding from the surface of a wall or wrinkles on a character's face. As the GPU has improved over the years, so have the techniques used for rendering surface details. Two of the components that have stayed relatively constant until recently are that information about the surface is stored in a 2D texture that is applied to the geometry of the object and that some form of ray casting algorithm is used to render the object.

Bump mapping was the first technique developed to visualize surface variations across flat geometry. The texture used for this technique contains high resolution normal information. A fragment shader applied to the geometry uses this normal information to compute per-pixel lighting. This technique is able to depict the high frequency shading variations that should be present across the non-smooth surface. The next evolutionary step is a technique called parallax mapping (Tomomichi, et al., 2001). This technique adds surface height information to the texture that contains the normal information. The run-time technique used for this technique casts a ray from the observer to a location on the surface of the geometry. The technique then walks along this ray, looking to see where this ray first intersects with the surface height information. A final pixel value is then computed using the lighting and out-the-window texture information from this new location. This technique provides both per-pixel lighting and appropriate parallax results. Several variations on this basic technique have been developed over the past several years, but they all contain the same limitation. The surface of the object is not truly displaced, it only appears that way. Some of the various resulting issues are aliasing artifacts, depth buffer errors, and silhouette limitations.

Another rendering technique is called displacement mapping. This technique deforms the geometry of an object to include the variations in its surface. The data required for this technique is typically the same as for parallax mapping. Until recently, this technique was not practical for real-time applications due to the polygon density that it requires. However, modern GPUs are able to re-tessellate or sub-divide the geometry they are sent from the CPU very efficiently. They also provide the ability to modify the location of the resulting vertices based on texture information. This technique has been shown to be quite effective at rendering bare earth terrain surfaces (Cantlay, 2011). The rendering technique presented in this paper is a variation of this technique which renders both natural and manmade features as part of the terrain surface. To the authors' knowledge, this rendering technique has not been used in this way before.

APPROACH

The proposed approach is a largely automated pipeline that goes from refined source as input to a run-time visualization of feature rich terrains as the output. This paper describes a framework that consists of three independent applications with the connection between them being implemented through the exchange of files in pre-existing open formats. When used together as shown in Figure 1, they make up a pipeline that goes from refined source data ingest on the left to visualization of a feature rich scene in an IG style application on the right.

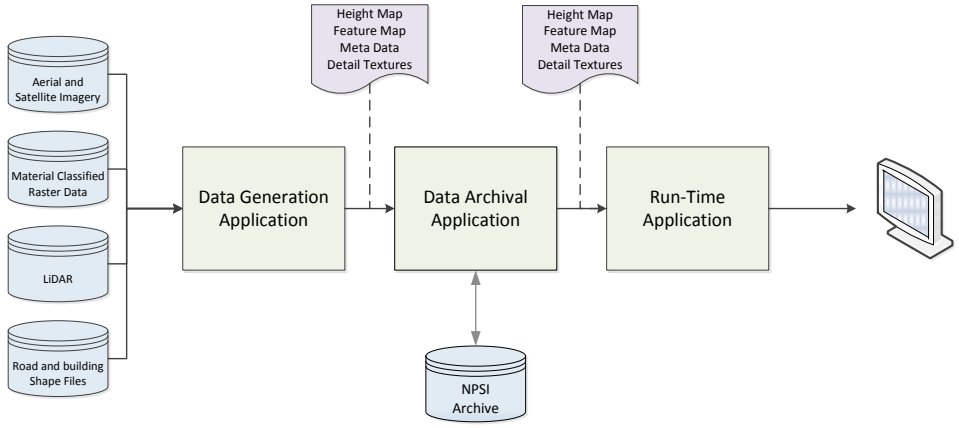


Figure 1: Displacement Mapping Pipeline Solution

Two key concepts of the proposed approach are the notions of *plausible* visual results and *appearance* of features. For the purposes of this paper, we define *plausible* visual results as rendered images that are free from distracting inconsistencies. As a converse example, the depiction of trees and buildings on a lake would NOT create a plausible visual result. Additionally, we define the *appearance* of features to mean the visual representation of features that are void of explicit geometry and textures that uniquely define said features. As an example, rather than model every tree in a forest, displace and deform the terrain's surface to capture the appearance of the trees in the forest.

Data Generation

The pipeline begins with an application that consumes multiple types of input data to generate raster files that contain both height and feature information. One of the goals of this application is to support a wide variety of input data types. At a minimum, these include out-the-window imagery, material classified data, road vector data, and digital surface model data derived from stereo pair imagery or LIDAR data. Different pieces of knowledge about the features on the terrain are gained from each type of data. For example, a vector data file that specifies where roads are located is used to prevent buildings or trees from being placed too near or too far from those locations. More sophisticated rules based on probabilities are also used. For example, houses are commonly near roads, but not typically right up against or in the middle of them. If available, height information extracted from stereo pair imagery or LIDAR data is used to determine the correct height of identified objects and indicates where other non-identified objects might be located. If LIDAR is unavailable, geo-typical heights are generated randomly.

A key requirement for the data generator is that it produces features that are aligned with existing out the window imagery to avoid adding visual disparity. There are many features visible in aerial imagery (either directly or through their shadows) that may exhibit depth and motion parallax. The most common features are trees, rooftops, parked cars, and fences. Each of these present challenges to the data generator, which must be able to operate with only OTW imagery in some cases. Trees are forgiving features because most viewers cannot judge the exact number or location of groups of trees, and the irregular boundaries of trees can disguise errors in feature extraction. Buildings are much more challenging because building footprints have a very constrained (often polygonal or rectilinear) structure and irregular (blobby or jagged) artifacts would be extremely distracting. Furthermore buildings are often camouflaged by the color of the ground, especially in desert environments. Additionally, rooftops can be occluded by shadows that they cast onto themselves and also by nearby trees.

One approach to avoid artifacts in buildings is to use a restricted model that is only capable of representing possible building shapes. Since it is required that features align with the OTW imagery, the model generation must be guided and constrained by that imagery. One such model is illustrated in Figure 2; this model is designed to represent suburban houses with rectilinear footprints but is also capable of representing many commercial buildings. Buildings are represented as the union of a set of mutually orthogonal 'blocks' with hipped or flat roofs. The shape of each block is described using parameters *length* (ℓ), *width* (w), *base-height* (h_b), *roof-height* (h_r), and *gable-inset* (g). Using a model with only a few parameters controlling shape prevents the data generator from producing buildings with unusual footprints that could distract a pilot. A simple model also enables random sampling of building attributes that are not visible from imagery or other GIS sources. For instance, the building roof shapes in Figure 2d are generated with and selected by random sampling to produce a convincing amount of variety in a scene. As in procedural modeling systems, the distributions that are sampled can be controlled in order to produce the variety of building shapes that are typical in a geographic area.

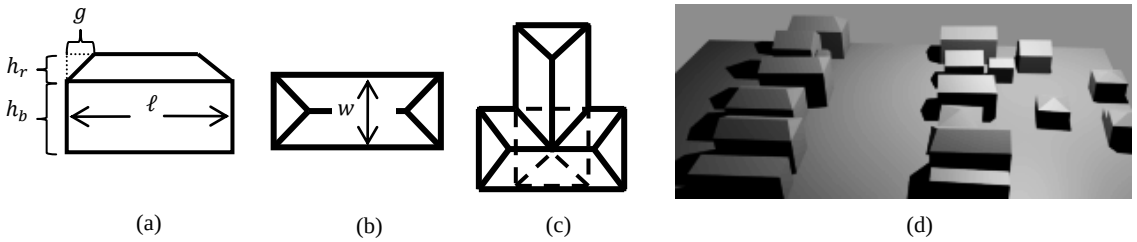


Figure 2: Block Based Building Model
(a) A Ground-Level View (b) Nadir View (c) Multiple Overlapping Blocks (d) 3D Visualization of Multiple Buildings

In the proposed framework, models such as the one depicted in Figure 2 are used to generate a height map that is suitable for displacement mapping. The generation of the height map is illustrated in Figure 3; at each output pixel only one height (the height of the tallest object) is recorded.

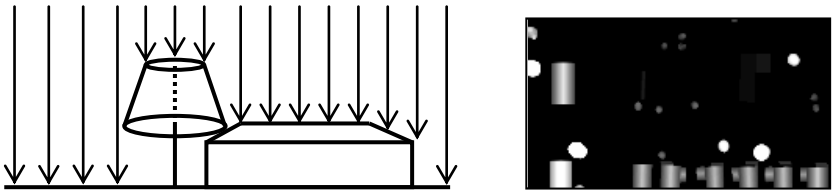


Figure 3: Height Map Generation
(left) Topmost Fragments Stored in Height Map (right) Heights Encoded in Grayscale Texture Map

The framework we propose can be used in conjunction with existing terrain databases that include geo-specific polygonal models. This is accomplished, in part, by allowing the user to input a mask that prevents features from being placed in certain locations on the terrain. The user creates this mask by identifying where all pre-existing features, like landmark buildings, are located in their pre-existing database. In theory, this could be accomplished by stitching together multiple orthographic on-nadir renderings of the scene without the terrain. If this mask is not

provided, height and feature information will likely be generated for the same location as objects that already exist in the database. As a result, two representations of the same feature would likely intersect with each other in undesirable ways.

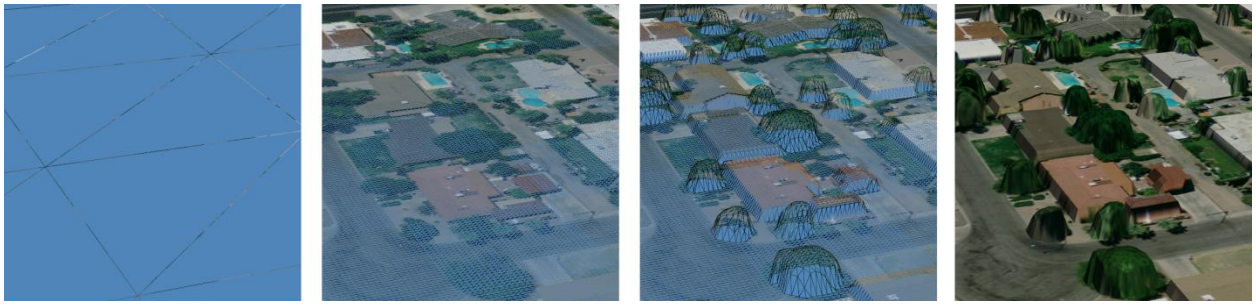
Data Archival and Interchange Application

Many geospatial source data archival and interchange formats exist across the DoD. Data archival and interchange formats enable reuse and sharing of refined and sometimes intercorrelated geospatial data sets (Nichols, 2005). For the purposes of this paper, the NPSI (NAVAIR Portable Source Initiative) archive is addressed. The proposed pipeline incorporates an application that facilitates easily committal and extraction of data, that was previously generated by the first phase of the pipeline, to and from the NPSI archive. This data is in a geospatial source data form which is independent of the run-time format. The output of the data generation application is written to raster formats supported by NPSI plus additional metadata describing the feature content.

Run-time Application

The last application in the pipeline is a run-time application that uses displacement map data and an innovative rendering technique to display geo-specific high-complexity terrain surface regions in a virtual environment that is similar to those in aircraft simulators used by the DoD. It is common that the limiting factor in the number of features that can be rendered on the terrain is dependent on the performance of the hardware that is being used to render the scene. Graphics hardware performance improvements have allowed for an increase in feature densities. However, the fidelity and density of features visible in a real-time simulation does not typically match the number of features that are discernible in the real world. The goal of this application is to provide the *appearance* of realistic numbers of features on COTS graphics hardware while sustaining real-time (60 Hz) frame rates.

Displacement mapping techniques are commonly used in video games to represent small surface details, such as the bumps on a rock wall or wrinkles on a character's face. These techniques have also been employed to create terrain surfaces at run-time based on elevation data. The selected rendering approach is an adaption of these techniques that is ideally suited for rendering realistic numbers of features as part of a pre-existing terrain surface. This approach is significantly different than traditional methods of adding clutter and features to the scene which place additional geometry and textures into the scene for each object that is to be visualized. The basic technique is illustrated in Figure 4 below. The original coarse terrain surface, shown on the far left, is re-tessellated on the GPU to produce the dense mesh show on the left. The vertices of this mesh are then displaced, as shown on the right, based on the height information contained in the displacement map. When the resulting displaced mesh is rendered, see the far right image, the *appearance* of the features that had been captured in the displacement map is revealed.



(a) (b) (c) (d)

Figure 4: Displacement Mapping Technique
(a) Original Mesh (b) Re-tessellated Mesh (c) Manipulated Mesh (d) Rendered Result

PRELIMINARY RESULTS

The results discussed in this paper are based on the current state of the proof of concept applications developed for the data creation and run-time components of the pipeline shown in Figure 1. Indentified limitations, lessons learned, and future directions are also discussed

Displacement Map Data Generation

A prototype application, called Displacement Map Data Generator (DMDG), has been developed to implement the data generation component of the pipeline. The DMDG application is ideally suited for ingesting different types of geospatial data and generating displacement maps with minimal user intervention. It also provides an easy to use interface that facilitates loading many different types of data and enables users to select a region for height and feature map generation. DMDG looks similar to other desktop GIS applications, but allows users to provide input that is used to improve the quality of the resulting maps. DMDG is the front-end to the algorithms that have been developed to produce the height and feature information that is needed. The DMDG application produces output in the standard raster file format GeoTIFF. At a minimum, it generates a height map that contains the vertical information for features that protrude from the bare earth surface. Examples of these features would be houses, buildings, trees, and bushes. Additionally, DMDG can generate feature maps. These maps capture feature information for all vertical objects that were identified in the input data. The run-time application can use this data to process each part of the terrain appropriately. Lastly, DMDG was built in a way that makes it easy to incorporate new processing algorithms and enables the reuse of these algorithms in other COTS GIS software packages.

Displacement Map Reference Implementation

The Displacement Map Reference Implementation (DMRI) prototype application utilizes the data generated by the DMDG in order to render an environment that is rich in motion parallax cues. DMRI enables the user to freely fly over a large section of terrain and continually page high resolution imagery and height data from disk to the GPU, based on the observer's location. DMRI leverages pre-built polygon based terrain databases in order to replicate the rendering techniques commonly used by DoD image generators. This run-time application provides the ability to evaluate the performance and training benefits of using displacement mapping in aircraft simulators.

Image Quality

The images generated by DMRI provide visually plausible results when the observer is more than 1000 feet above the terrain (See Figure 5 below).

Figure 5: Run-time with Observer at 1000'

One of the key metrics in determining the viability of a rendering technique like this is to determine how close the observer can get to the terrain before the features no longer maintain a plausible appearance. Figure 6 shows a series of images with the observer at different elevations above the terrain. The image on the right shows that the current implementation begins to show significant visual artifacts at 250 feet. More specifically, the tree depicted in the lower left corner of that image is not recognizable as a tree, and therefore cannot be considered a *plausible* visual result. This known limitation is one that will be addressed as DMRI's rendering implementation is improved.

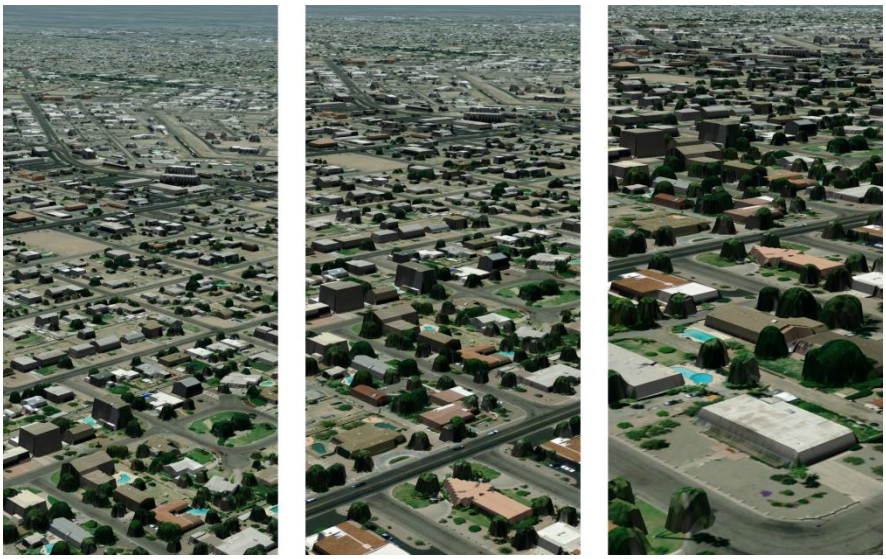


Figure 6: DMRI with Observer at 750' (left), 500' (center), and 250'(right)

Enhanced Visual Cues

There are several potential benefits of the rendering method described in this paper beyond the consistent rendering performance that is achieved regardless of feature densities. This technique can be used to improve feature contrast, lighting, and terrain texture detail. Metrics to assess image quality should allow for the objective determination of detail vs. human vision performance. At this point, it is not clear what metrics will be used, but, contrast, resolution and optical flow are some of the options that may be explored.

The authors expect that the amount of perceived contrast between foreground features and the background imagery in the rendered scene will increase. Furthermore, it is expected that the quality of this perceived contrast will be proportional to the quality of the texture on the terrain and the number of features on that terrain. By deforming the terrain at the location of all identified features, this rendering technique is able to create vertical edges in the scene that enhance what the authors call local contrast. The occlusion created by trees and buildings break up the uniformity of the terrain imagery and generates local contrast that augments motion parallax and optical flow. Figure 7 shows how significantly this technique can alter the appearance of the rendered terrain. The right image in Figure 7 shows a considerable increase in vertical edges, which may result in increased local contrast, edge parallax, motion parallax, and potentially an increase in optical flow. On future efforts, the authors plan to develop metrics that quantify the amount of local contrast that results from various rendering approaches.

It is sometimes the case that the lighting effects of an object in the scene can create more of an impact on the scene then the object itself. A good example of this is a green tree in a grass covered field. The color of the tree is close to the color of the grass, but the shading of the tree and the shadow it casts can introduce substantial contrast differences in the rendered scene. The identified rendering technique is ideally suited for computing these lighting effects by leveraging the height information contained in the displacement map. Figure 8 shows that the final output (left image) is the result of multiplying the lighting component (center image) by the out-the-window component (right image). It should be pointed out that the lighting component in the center image only includes shading and that shadow calculations will be added in the future.

Lastly, the amount of detail perceived in a scene is commonly limited by the resolution of the out-the-window texture that is applied to the terrain. An additional benefit of the DMDG application is that it can generate a feature map that can be used by the run-time. This map contains the feature type (an index into a set list of supported types) for each texel in the displacement map. This also means there is at least one feature type index value available on the GPU for every out-the-window texel. This data enables the run-time to selectively augment or blend geo-typical detail textures into the geo-specific out-the-window textures. The authors plan to investigate this concept and if results are encouraging, this will open the possibility for not only the addition of texture detail on indentified 3D features, but for the augmentation of the entire terrain texture.

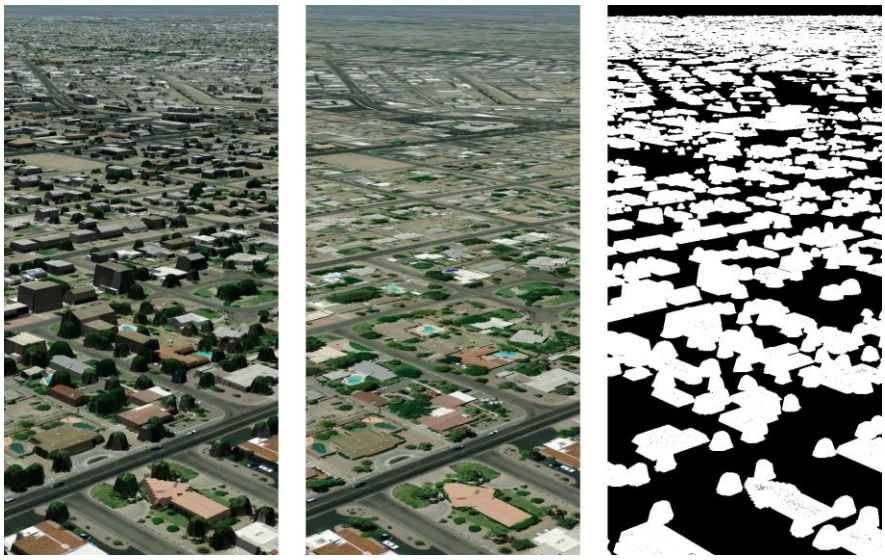


Figure 7: Improved Local Contrast Using Displacement Mapping
(left) Technique Enabled (center) Technique Disabled (right) Affected Pixels

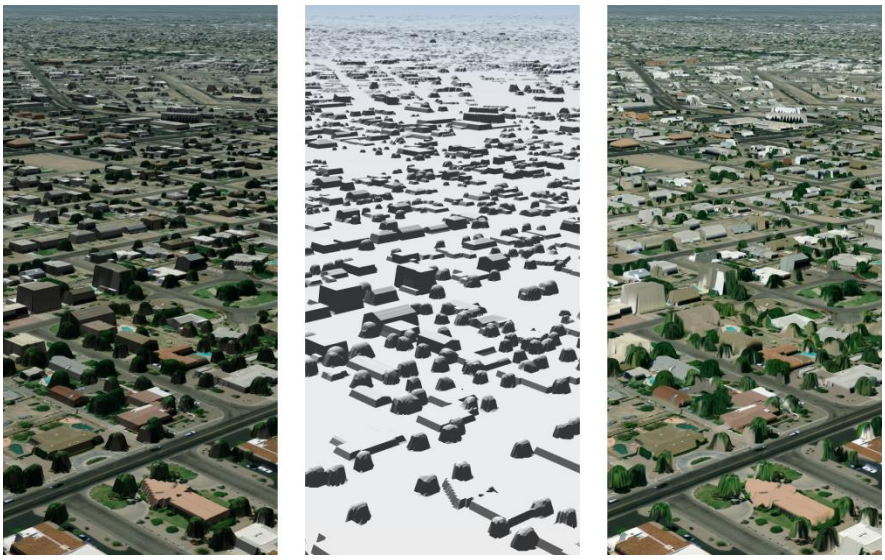


Figure 8: Shading
(left) Final Scene (center) Lighting Component (right) Out-the-Window Imagery

FUTURE WORK

The authors plan to focus on four areas in the future. First, they will increase the quality of their feature extraction techniques. Second, they will work on reducing the height above the terrain at which the run-time is able to render plausible visual results. Third, metrics will be developed to quantify the benefits of the proposed pipeline approach.

Lastly, the authors would like to create stand-alone software libraries based on the work presented in this paper. Their goal is to facilitate the incorporation of the data generation and rendering components of their pipeline into mainstream GIS and IG applications.

CONCLUSION

In the real world, a large number of redundant visual cues provide the human vision system with important physiological and psychological perceptual information. In the ideal scenario, aircraft simulators would provide a high fidelity set of visual cues to trainees. However, due to various limiting factors, today's aircraft simulators provide pilots with a subset of all possible real world visual cues. The authors set out to find a solution for adding realistic geo-specific feature densities to visual databases that can be rendered at real-time frame rates. A prototype application was created that reliably extracts *plausible* features based on various types of input data. This application is able to process very large data sets in a manageable amount of time. Furthermore, a runtime prototype was implemented that demonstrates how the combination of imagery and displacement maps can be used to provide the *appearance* of realistic feature densities at real-time frame rates. The combination of these two applications show that the proposed framework is capable of generating enhanced feature data and visual results in ways that are compatible with typical DoD database creation workflows, image generation software, and aircraft simulator architectures. It is expected that the increase in feature density will provide enhanced visual cues that will lead to enhanced local scene contrast, motion parallax, and optical flow. These enhanced visual cues have the potential to provide a higher fidelity environment in future aircraft simulators.

ACKNOWLEDGEMENTS

Thank you to the NAVAIR SBIR program for providing the funding for this research (topic N102-116 Geospecific Displacement Maps for Real Time, Stereoscopic Training Simulation). However, the views presented in this paper are those of the authors and do not necessarily represent the views of DoD or its Components.

REFERENCES

Cantlay, I. (2011, January). DirectX Terrain Tessellation. Retrieved from https://developer.nvidia.com/sites/default/files/akamai/gamedev/files/sdk/11/TerrainTessellation_WhitePaper.pdf

Entzinger, J. O. (2009). The role of binocular cues in human pilot landing control. *Proceedings of AIAC13.* Melbourne.

Gibson, J. J. (1950). The Perception of Visual Surfaces. *The American Journal of Psychology , 63* (3), 367-384.

Hart, S. G., & Battiste, V. (1991). The use of visual cues for vehicle control and navigation. *Visually Guided Control of Movement , 3118*, 7.

HERE. (2013, 6 6). Retrieved 6 6, 2013, from HERE: http://here.com

Martin, E. L., & Rinalducci, E. J. (1983). *Low-level flight simulation: Vertical cues.* No. AFHRL-TR-83-17. AIR FORCE HUMAN RESOURCES LAB BROOKS AFB TX.

McIntyre, H. M., & Roberts, M. E. (1996). Simulated Visual Scenes-Which are the Critical Cues? *Thomson Training and Simulation Ltd, Flight Simulation: Where are the Challenges? p(SEE N 97-10546 01-09) .*

Müller, P., Wonka, P., Haegler, S., Ulmer, A., & Van Gool, L. (2006). Procedural modeling of buildings. *ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH 2006 , 25* (3), 614 - 623.

Nichols, W. K. (2005). The NASMP Portable Source Initiative Shifting the Contracting and Acquisition Paradigm. *I/ITSEC .*

Padfield, G. D., Lee, D. N., & Bradley, R. (2003). How do helicopter pilots know when to stop, turn or pull up?(Developing guidelines for vision aids). *Journal of the American Helicopter Society , 48* (2), 108-119.

Pafford, J. (2011). Are simulation specific formats becoming obsolete? *Proceedings of IMAGE 2011.* Scottsdales, AZ.

Patterson, R., Akhtar, S. C., Geri, G. A., Morgan, W., Pierce, B. J., Dyre, B. P., et al. (2004). *Terrain texture and 3-D object cues in the control of heading in simulated flight.* WASHINGTON STATE UNIV PULLMAN: DTIC Document.

Tomomichi, K., Toshiyuki, T., Masahiko, I., Naoki, K., Yasuyuki, Y., Taro, M., et al. (2001). Detailed Shape Representation with Parallax Mapping. *ICAT 2001.* Tokyo, Japan.

Wonka, P., Wimmer, M., Sillion, F., & Ribarsky, W. (2003). Instant architecture. *ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH 2003* , 669-677.