DSC North America 2003 Proceedings, Dearborn, Michigan, October 8-10, 2003 (ISSN 1546-5071).

Terrain Validation and Enhancements for a Virtual Proving Ground

presented at the

Driving Simulation Conference – North America

October 8 – 10, 2003

Dr. David Lamb
lambd@tacom.army.mil
586-574-5209

Dr. Alexander Reid
reida@tacom.army.mil
586-753-2212

Nancy Truong
truongn@tacom.army.mil
586-574-8633

John Weller
wellerj@tacom.army.mil
586-574-8633

U.S. Army RDECOM-TARDEC
National Automotive Center
MS: AMSTA-TR-N #157
6501 E. 11 Mile Rd.
Warren, MI 48397-5000

DSC North America 2003 Proceedings, Dearborn, Michigan, October 8-10, 2003 (ISSN 1546-5071).

ABSTRACT

Engineers and scientists from the Ground Vehicle Simulation Laboratory (GVSL) located at the U.S. Army Tank-Automotive Research, Development and Engineering Center's (TARDEC) National Automotive Center (NAC) have validated a virtual graphical terrain for use in the real-time warfighter/hardware-in-the-loop motion base simulators. This was accomplished by comparing and analyzing the profile data acquired from the virtual environment of Aberdeen Proving Ground's (APG) Churchville B course with the real data collected over the actual course. To obtain the data from the virtual terrain, complex mathematical equations developed by scientists at GVSL were utilized. The MATLAB analysis tool was used to analyze the data and help verify the terrain. The paper will discuss the processes that we incorporated to validate the database, new techniques being developed to improve our validation and methodologies to give the virtual terrain higher frequency terrain characteristics. Verification of the virtual terrain is important since engineers need to confirm that the profile of the terrain that they are driving the Ride Motion Simulator over corresponds to the real terrain.

INTRODUCTION

The GVSL at RDECOM-TARDEC NAC has a six degree-of-freedom hexapod, man-rated, motion based simulator, the Ride Motion Simulator (RMS). This simulator is utilized to perform real-time warfighter/hardware-in-the-loop simulation and is capable of reproducing the ride of most ground vehicles over a wide variety of terrains. To replicate the authenticity of a real vehicle traversing a terrain, visual rendering of a scene and motion dynamics are needed to enhance the driver's perception of being in a realistic environment while driving the RMS. To achieve this realism, engineers and scientists from GVSL teamed with engineers from the Army Corps of Engineers ERDC (Engineering Research & Development Center) to form a Science and Technology Objective (STO). This STO, called "High Fidelity Ground Platform and Terrain Mechanics Modeling," is intended to create methodologies for better ground vehicle modeling and simulation. The STO researchers partnered with the University of Iowa to generate high-resolution visual databases for the Evans and Sutherland ESIG®-HD/3000 and Harmony® Image Generators. Thus, a visual model of Aberdeen Proving Ground's (APG) Churchville terrain was built to satisfy this requirement. The STO, through ERDC, is developing realistic terramechanics modeling for inclusion with this and other terrains.

The visual database in conjunction with the RMS will be used for engineering design trade-off studies such as comparison of head mounted display (HMD) vs. flat panel, vehicle performance over a terrain, and human factors such as driving performance under stress.

VALIDATION

Engineers and scientists need to determine whether a simulation model, in this case, the visual representation of APG's Churchville Proving Ground being rendered by the image generator and the corresponding terrain database used by the dynamic model, are accurate representations of the actual visual and roughness characteristics that are seen and felt while driving on the real Churchville. If the model is inaccurate, questionable or unreliable, then any further research and development being done by using the model will be inconclusive and invalid. To prove the effectiveness and credibility of a model, a validation must be done.

According to Department of Defense Instruction (DoDI) 5000.61, validation is defined as the process of determining the degree to which a model and its associated data provide an accurate representation of the real world from the perspective of the intended uses of the model. In simple terms, a person doing a validation should ask, "did I build the right model?". To validate that Churchville was built correctly, profilometer data was used.

PROFILOMETER

Actual Profilometer

A physical profilometer measures and captures elevation data at equally spaced points along a path to determine the roughness of the terrain. To measure the profile of Churchville, APG's Aberdeen Test Center (ATC), located at Aberdeen, Maryland, used a profilometer. The profilometer consists of a wagon type trailer that has 4 wheels with 2 axles (which is towed by a vehicle), a data acquisition sub-system and a mobile mini-computer base data analysis system. The trailer has a gyroscope with a potentiometer to measure pitch and roll angles, and an ultrasonic device, which consists of four magnets mounted on the rear wheel and a pickup coil mounted on the wheel frame, to measure the vertical distance between the frame and the terrain. As a magnet passes the coil, a pulse is generated, hence providing the data to be sampled at equal increments as the vehicle travels over the terrain surface. The data is then

stored in the microprocessor-base data acquisition sub-system which processes it to give the x and y coordinates of the terrain profile. To determine x and y, the pitch and roll angles are used:

$$y_R = ?\sin(?)dl + w/2\sin(F) + U_L$$
$$y_L = ?\sin(?)dl + w/2\sin(F) + U_R$$
$$x_L = x_R = ?\cos(?)dl$$

where $?$ is the pitch angle and F is the roll angle, dl is the increment of travel along the surface, w is the width of the road, $_R$ and $_L$ are the right and left sides, and U_L and U_R are the distances measured by the ultrasonic sensors for the left and right sides of the track. Therefore, the profilometer used by ATC outputs measurements along two tracks which are set at a certain distance width apart instead of measurements along the center of the road.

Virtual Profilometer

The virtual profilometer is a software tool designed to operate over a virtual terrain (as stored in a terrain database) the same way a real profilometer would operate over actual terrain. It can also capture the surface normal, soil type, and other data stored in the database for each of these points. The virtual profilometer will capture elevation data at equally spaced points along a path. The virtual profilometer can also replicate the physical profilometer since it is able to take measurements along two tracks which are set a certain width apart. If two tracks are desired, and a track width is specified, the virtual profilometer will offset the terrain queries to the left and right by half the track width, and report out two profiles. The path being followed is assumed to be the centerline of the vehicle. This offsetting is accomplished by maintaining a current direction for the path, so the directions of 'left' and 'right' can be computed. If two tracks are not desired, the virtual profilometer only reports data for the path itself (the centerline), with no offset.

There are several inputs to the virtual profilometer. A primary input is the path to be traveled by the profilometer. Basically, the virtual profilometer acts as if it were a trailer being towed behind a vehicle. It does not independently decide on a path, rather it 'follows' a path generated by another source (as a trailer follows a vehicle across the terrain). Some other process is needed to generate the path.

Input paths for the virtual profilometer are usually generated by another simulation of a vehicle on the terrain. This simulation acts as a vehicle pulling the virtual profilometer behind in post-processing. However, generally these paths do not consist of equally spaced points, but rather they are time series with a constant time (but not space) delta. One of the main jobs of the virtual profilometer is to generate an equivalent path consisting only of equally spaced points. This is the main algorithm that lies in the heart of the virtual profilometer.

The core of the virtual profilometer is an algorithm for calculating equally spaced points along a trajectory. Generally, the trajectory is given by a sequence of points which are not originally equally spaced. This is because the points were generated by a simulation, and the points are given at a constant time delta, but with a platform speed that varies along the path, resulting in points which are sometimes close together spatially, and other times farther apart. In the extreme case, the simulated platform generating the trajectory might actually pause and stand at one spot for a while, before resuming travel, which would result in a section of the sequence being stationary. The profile to be computed will have a constant space delta, regardless of the speed of the platform.

Once the constant time delta trajectory provided by a simulation is converted into a constant space delta path, then the virtual profilometer is ready to begin terrain queries to the database. Thus, the file name for the terrain database to be queried must be input. After the path has been determined, another key input is the distance to use at the constant space delta.

This algorithm is based on the arc length calculation:

$$s = \int_{t_0}^{t_1} \sqrt{\left(\frac{\partial x}{\partial t}\right)^2 + \left(\frac{\partial y}{\partial t}\right)^2 + \left(\frac{\partial z}{\partial t}\right)^2}\, dt\,.$$

The arc length formula is then coupled with an algorithm to interpolate between input points,

$$\left(ax_1 + (1-a)x_2, ay_1 + (1-a)y_2, az_1 + (1-a)z_2\right)$$

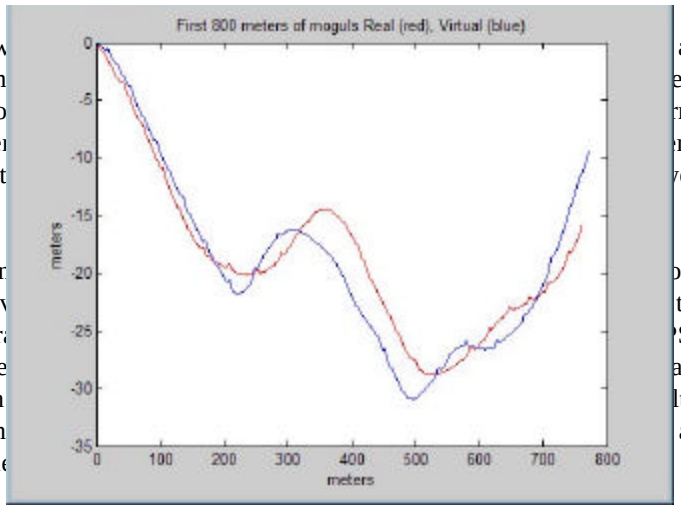
where $a = \dfrac{s - s_1}{s_2 - s_1}$ represents the interpolation factor between two points (x_1, y_1, z_1) at arc length s_1 and (x_2, y_2, z_2) at arc length s_2. This is all combined with an algorithm to converge quickly on a point that will define the end of a pre-established arc

length. Several subroutines are used, including an arc length calculator, and an interpolation routine. The method by which these are put together allows for the generation of the constant space delta path that corresponds with the input path.

In addition to calculating the arc length, the virtual profilometer also contains a rather simple algorithm to generate Power Spectral Densities (PSDs). This is not a sophisticated method, but rather it is based on the Fast Fourier Transform (FFT). The simple algorithm does neither windowing nor averaging, and it is prone to inaccuracies such as noise and leakage. If better results for PSDs are required, such as using windowing and other techniques to reduce errors, the user should take the raw profile and feed it into their own PSD algorithm. However, for a quick-and-dirty estimate of the PSD of the terrain, the profilometer can be used.

Besides outputting a quick estimate of the PSD, there are other output options to select from. One is a straight profile, just giving elevation, z, in terms of distance down the path, s. Another is to give the path traveled, including x, y, z points, at equally spaced intervals.

A problem was encountered when the algorithm was first written. It was originally assumed that the virtual profilometer was being pulled along the path at some speed that was bounded away from zero, i.e. that the trajectory had a minimum speed greater than zero for the entire run. An initial "guess" for the next point is used to start the convergence algorithm. When the speed dropped, a division by zero resulted. This caused faulty answers that eventually resulted in a new algorithm, one that makes no assumptions about the speed of the trajectory. The current algorithm can handle trajectories that slow, or even stop, during the run defining the path that the virtual profilometer is to follow. It has since been established that real, physical profilometers do not have problems when pulled at very low speeds, or even if stopped for a time during a run. Therefore, the current version better replicates the behavior of the real thing.



A second problem was that the file formats, especially for output, w and each person had his/her own preferred output format. The program ed a format different from the last use. To resolve the various output fo rmat selections and the user selects the output he/she desires from a me r of times the program needed to be changed and recompiled, while st ere handled the same way, as their formats varied.

The program has evolved while it was in development, and also sir only going to produce a path of equally spaced points as described abov that actual profiles may be output. Later, the ability to output two tr SD calculation was added. Most recently, it has been modified to handle arch Division (ERD) format as well as new input formats and terrain lts. Eventually, all major formats for input files, output files, and terrain and any new features that are desired by the user community can be adde

The result of the virtual profilometer program was a digital profile of the virtual terrain that was used on this project. The data obtained from the virtual profilometer is used to analyze and validate the virtual Churchville.

VALIDATION PROCESS AND RESULTS

To implement the validation process, MATLAB® was used to plot and compare the real profilometer data obtained from APG with the data acquired through the use of a virtual profilometer. To acquire the data for the virtual terrain, a person sat in the RMS a ch a he d to o it v he e

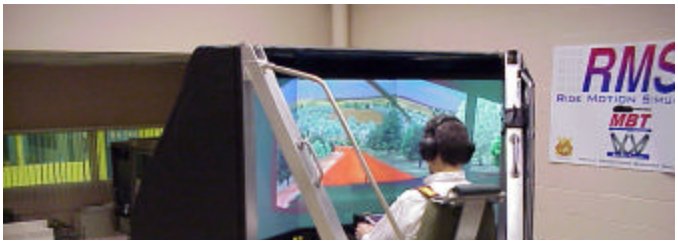


FIGURE 1 RMS in real-time.

T as n ta

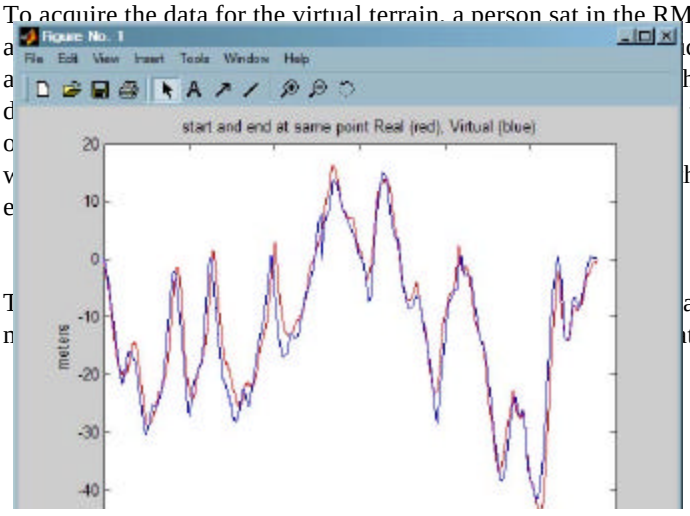


FIGURE 2 Real and virtual profiles.

obtained was at different sampling interval; hence, linear interpolation techniques were applied to give equal interval outputs. After interpolation, the plot of the two curves were adjusted to start and stop at the same point on the course at identical elevations, and traverse the course in the same direction and for the same distance. Figure 2 shows the profilometer data for both the real and virtual terrain starting and ending at the same points. The graph shows that the profile from the virtual proving ground correlates well with large terrain changes when comparing it to the profile data from the real proving ground since both profiles have the same shape and length. Furthermore, a profile from a terrain is unique, so this validates that a driver using the RMS to traverse the virtual Churchville course B is indeed driving on the same course as a driver on the real Churchville course B at APG. This validation was very beneficial because when the GVSL team first evaluated and validated Churchville B using the virtual and real profilometer data, they noticed that the virtual terrain was half the size of the real one. Hence, numerous future headaches were avoided due to this discovery.

To further evaluate the profile of the terrain, the data was partitioned into long and short-wavelengths to identify where the moguls, or bumps, are located. The moguls were plotted (Figure 3) to determine if the moguls were in the virtual terrain and the slight bumps in the plot represent that they are. In addition, Figure 3 illustrates that upon closer inspection, there is a difference between the real and virtual terrain.

As can be seen from Figure 3, the virtual data has sharper troughs and peaks than the real data. The sharper shapes that are observed in the virtual profile are due to the effects of sampling from the real world terrain and the use of flat polygonal surfaces to model the terrain. Sampling causes a predictable loss of data and using polygonal surfaces can sometimes create artifacts such as edges which are not present in the real world. From partitioning the data into long and short wavelengths or detrending (removing long wavelengths), values for the root-mean-square (rms) were calculated. In the detrended terrain area, which includes wavelengths of 18m or less, the calculated rms values for the virtual terrain is 4.57cm (1.8in) and for the real terrain is 4.95cm (1.95in).

FIGURE 3 Moguls for real and virtual.

To further evaluate the data, a power spectral density was done on both the virtual and real profilometer data. From looking at the graph (Figure 4), there is a noticeable difference between them. The power that is contained in the virtual terrain is much lower than the real terrain. Again, this discrepancy is due to how the polygons in the virtual terrain were constructed. To solve this problem, Non-Uniform Rational B-Splines (NURBS) can be used to increase the high frequency area.

THE NEED FOR HIGH-RESOLUTION TERRAIN DATA

In order for the GVSL to conduct engineering-level, man-in-the-loop simulations using the RMS, the terrain skin that the simulation uses needs to be improved. The real-time dynamic model is a high-fidelity, multi-body, mathematical model representing the vehicle's dynamics. The Ride Motion Simulator is capable of re-creating six degree-of-freedom motions with a bandwidth approaching 40 Hz. Thus, the limiting factor in creating a truly high-resolution, engineering-level simulation is the terrain skin that is used.

The terrain skin that the real-time dynamic model traverses (which we will hereafter refer to as the real-time terrain) is typically derived from the virtual terrain that the image generator is using (we will hereafter refer to this as the visual terrain). There are pros and cons to this method. The pro is that the terrain that the vehicle is traversing and that the dynamic model is using as inputs to the suspension system is the same one that the occupants are viewing. This provides the proper correlated visual/motion queues, which helps to reduce simulator sickness. The con is that an image generator can only display a limited amount of polygons in a scene. This leads to the terrain being broken up into polygons of a fairly large size, typically from 30m to 90m in size or larger. Thus, a cross-country terrain is reduced to 30m resolution, thereby removing all higher frequency ground content that would excite the vehicle during the traversal. Unfortunately for the dynamic model, this is equivalent to driving over a flat world and in no way can properly represent a cross country terrain. In order to properly recreate the effects of off-road simulations, a higher fidelity terrain database must be used as an input to the dynamic model.

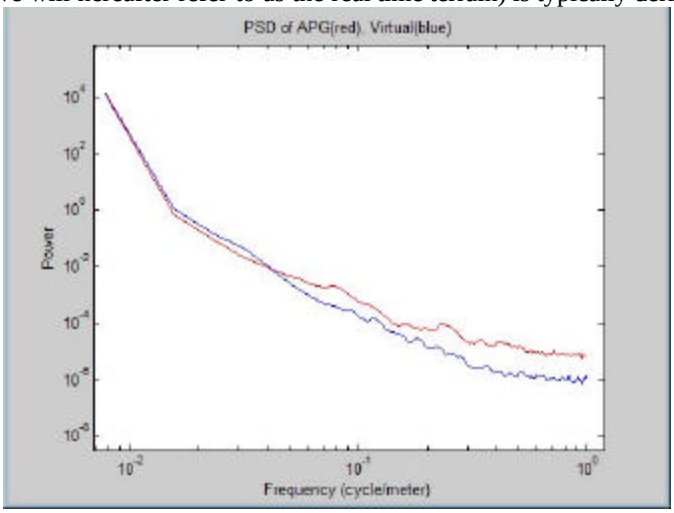


FIGURE 4 PSD of the terrain.

However, if the real-time terrain is vastly different from the visual

terrain that the image generator will render, simulator sickness may occur during a motion-based simulation. This will happen because the test subject is viewing one terrain while the dynamic model is traversing a slightly different terrain. Hence, the challenge in increasing the resolution of the real-time terrain is to keep it correlated to the visual terrain.

One solution to this problem is to develop a separate, high-resolution terrain database for use as an input to the dynamic model but yet have it be correlated to the image generator's visual terrain database. This high-resolution terrain database would consist of two parts: the original image generator's terrain skin and a separate piece which consists of high-frequency, low-amplitude terrain which is simply added to the image generator's terrain skin.

DEVELOPMENT OF A CORRELATED TERRAIN SURFACE

This section describes the development of a new methodology for creating a high-resolution surface which can be stored with as little information as possible, and which can be summed to (overlaid upon) a lower-resolution surface derived from a computer image generator's database.

To create the high-resolution surface, one must know a few details about the type of terrain they wish to create. The minimum and maximum frequency content must be known. This relates to the wavelengths of the terrain surface one would wish to create. For example, the virtual surface developer may wish to create a surface with bumps/hills with wavelengths between 0.5 meters and 5 meters. The developer must also have an idea of the rms (root-mean-square, a measure of the size of the bumps) value for the terrain and an idea of the frequency distribution of the waves in the surface. Does one want more higher frequency content than lower frequency content? More lower than higher? Or a uniform distribution? Figure 5 depicts the effects on a PSD plot of a surface by varying the fractal dimension of a NURBS surface.

Spectral fractal geometry (fractional Brownian motion) was used to create a high-resolution surface in the frequency domain, which was then transformed into the spatial domain using a two-dimensional inverse FFT (Fast Fourier Transform). Non-Uniform Rational B-Splines (NURBS) are used to interpolate this surface.

Once the minimum and maximum frequency content of the surface patch is determined, the control points describing the NURBS surface patch are then extracted so the spatial representation contains the desired frequency characteristics.

The current intent of the GVSL is to develop a library of representative surface patches which are overlaid upon the low-resolution terrain surface in real-time. This results in a geometrically smooth, high-resolution surface which is correlated to the lower resolution surface from the image generator. While a driver is navigating the image generator's database, the mathematical model representing the vehicle is receiving terrain input at a continuous resolution from the correlated high-resolution NURBS database. This unique methodology permits one to create an infinite resolution database storing a small, finite set of data points which describe the NURBS surface patches.

The development and implementation of this technology will provide a ground simulator with a high-fidelity terrain surface with user selectable frequency content, rms and frequency shaping (through the fractal dimension coefficient) for use with a real-time dynamic model which is correlated with an image generator's database. This will provide the motion simulator with realistic terrain inputs, thus creating a more realistic simulation.

A problem arises using any method where the motion-based simulator will be traversing a different database than the occupant is seeing. That is one of simulator sickness. When the occupant sees the terrain that he is traveling over, he is expecting to feel certain vibrations as he traverses it. If he is feeling the vibrations from a different (or modified in this case) terrain, simulator sickness may set in. This happens when his brain tells him from experience to feel one thing, but the simulator does something else. Obviously, a way to impart this new terrain information visually to the user is needed.

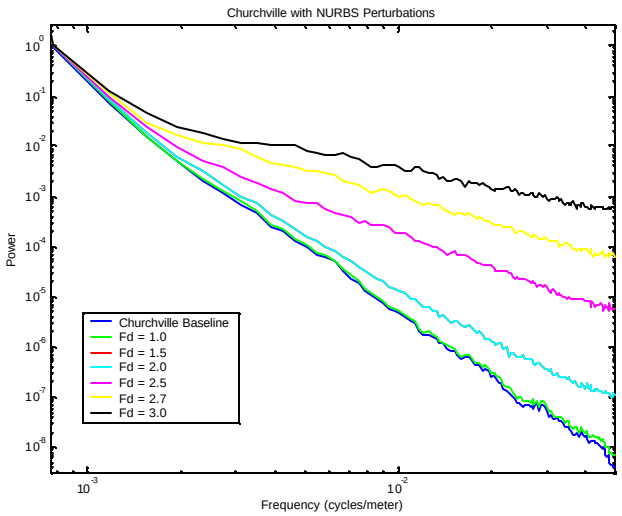


FIGURE 5 Effects on PSD of varying fractal dimension.

The reason an Evans & Sutherland Harmony® image generator is being used is because it provides the ability to manipulate the surface normals on the rendered polygons. This will create variable shading throughout each individual polygon instead of one constant shade for each polygon because of the way that the light source reflects back at different angles to the user's eye off of the polygonal surface. This technique is called bump-map texturing and lends the illusion of being able to render more image content than a typical image generator.

Figures 6 and 7 depict an image of the U.S. Army's Churchville B Proving Ground without bump-mapped textures and the high-resolution surface (Figure 6) and with (Figure 7).

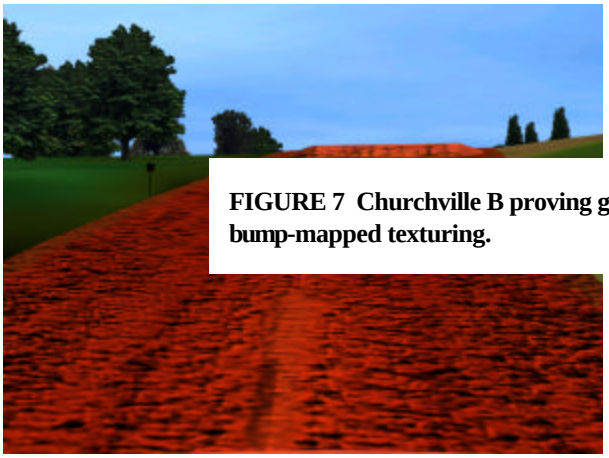


FIGURE 7 Churchville B proving ground with bump-mapped texturing.

DETAILED METHODOLOGY

While only a brief overview of the necessary steps needed to create a high-resolution terrain has been presented here, a detailed version can be found in [3].

TERRA-MECHANICS AND DEFORMABLE TERRAIN

In addition to using bump-mapped texturing, terra-mechanics will also be incorporated into the terrain. Terra-mechanics involves using physics-based analytic models to compute the 'in-plane' forces which the terrain (soil) will exert on a vehicle driving over it. It is hoped that these terrain models will compute the longitudinal and lateral forces on each tire patch in a realistic way. Also, the pressure exerted by the vehicle on the ground should deform the ground in a way that alters the geometry and the force response of the soil. That way, if a location is ever encountered again during the simulation, the second pass will respond to compressed soil both geometrically and in force response. Terra-mechanics can be divided up into two categories: what the vehicle model needs and what the terrain model needs. The vehicle model needs to have terrain elevation under each tire node (the contact patch of the tire to the terrain is broken up into smaller pieces called nodes) and it also needs lateral and longitudinal force information at each tire node. The terrain model needs to know the normal forces (pressures) which cause the terrain deformation and it needs to know where the tire is so it can get the correct type of soil for that location and also so that it can properly deform the correct location of the terrain.

Accomplishing all of this requires that the terrain query from the vehicle be more complicated than it previously needed to be. In the GVSL, we have chosen to break the terrain query into two separate calls. Each call is made by the vehicle model into the terrain model each time step, and the tire is expected to do some work between the two calls. We shall describe both calls, and the work done by both the vehicle and the terrain in response to both functions.

The first call the vehicle model makes on the terrain is a geometry query. The vehicle decides on a list of points on the tire which might be in contact with the ground, and queries the terrain for each point to determine what the ground elevation and upper unit normal is for that location. This allows the vehicle to decide if ground contact was actually made or not, and also to make a computation of the

normal force the vehicle exerts on the ground at each point where contact is made. The function is written to allow for an arbitrary sized list of points which may be queried with a single function call, and the returns are made in a list with the same size as used in the call, and the same ordering. This cuts down on overhead, allowing for a single function call to process many points.

After returning from this call, the vehicle can create a new list of points (usually shorter than the list used in the first (geometry) call) consisting of points where contact with the terrain is actually made. For each point in this new list, the vehicle can compute a pressure exerted on the ground at that location. The pressure will be given as a normal force exerted uniformly over a patch of known area centered on that point. For simplicity, we assume that all patches will be the same size throughout the simulation. However, an alternative would be to have the vehicle compute a patch each time this computation is made, and communicate the area of the patch to the terrain along with the normal force.

The second call the vehicle model makes on the terrain is the terra-mechanics query (also called the force query). The vehicle passes a list of points in contact with the ground, along with the normal force applied by the vehicle to the ground at each point. These normal forces really represent pressures applied to the ground in a patch around each point.

The terrain should use these pressures and the soil-properties at each point to compute the way the soil deforms at that location. These soil deformations will be stored (as described later) and used to alter the geometry if this location is ever visited again during the simulation.

Also, the vehicle needs to provide "slip" velocities to the terrain for each point in the second query. The terrain model should use these slip velocities, the soil properties at that location and the pressure to compute the 'in-plane' forces (i.e. lateral and longitudinal forces) the terrain applies to the vehicle at that point. In the future, we hope to have physics-based analytic models of soil to compute the 'in-plane' forces based on soil properties. Presently, this part of the terrain model is done heuristically from experimental data. We hope that eventually the terrain model will alter the soil strength at a location to account for the compaction due to pressure applied from earlier time steps. That capability is not available now, but we are building into the infrastructure a way to store the needed data so that it will be there when the modeling capability can use it. Like the first call, described above, this function is written to allow for an arbitrary sized list of points which may be queried with a single function call, and the returns are made in a list with the same size as used in the call, and the same ordering.

DEFORMABLE TERRAIN MEMORY STORAGE

Terra-mechanics requires a methodology of having the deformable terrain remember where the vehicle has driven and making modifications to the traversed portions of the terrain. As mentioned above, terra-mechanics can be divided up into two categories: what the vehicle model needs and what the terrain model needs. The vehicle model needs to have terrain elevation under each tire node (the contact patch of the tire to the terrain is broken up into smaller pieces called nodes) and it also needs lateral and longitudinal force information at each tire node. The terrain model needs to know the normal forces (pressures) which cause the terrain deformation and it needs to know where the tire is so it can properly deform the correct location.

It is readily apparent that we must somehow have the terrain remember the locations where the tire nodes have deformed it. However, to provide an acceptable deformation model, the terrain must be broken down into patches measuring approximately 1cm x 1cm (1 sq. cm). This terrain division leads to a daunting memory problem if we use a brute force method of simply subdividing the terrain parcel into sq. cm grid cells. Even using a small 1 km x 1 km parcel, we would have 10 billion sq. cm cells. Thus, brute force is an unacceptable way to mange the terrain memory.

An alternative technique that the GVSL is utilizing is to break the terrain up into 1 sq. meter grid cells. Each sq. meter cell can then be further subdivided into sq. cm cells, where the terrain deformation is actually stored (see Figure 8). The parent class, Grid, is defined as follows:

```
class Grid
{
    public:
        Grid(void);
        ~Grid(void);
        bool    Init(float xSize, float ySize, float xllcorner, float yllcorner);
        bool    SetDeltaZ(float x, float y, float newDeltaZ);
        bool    AddDeltaZ(float x, float y, float newDeltaZ);
        float   GetDeltaZ(float x, float y);
        bool    Destroy(void);
    private:
        mCell **mCells;
```

```
        int xSize;
        int ySize;
        int xllcorner;
        int yllcorner;
}
```

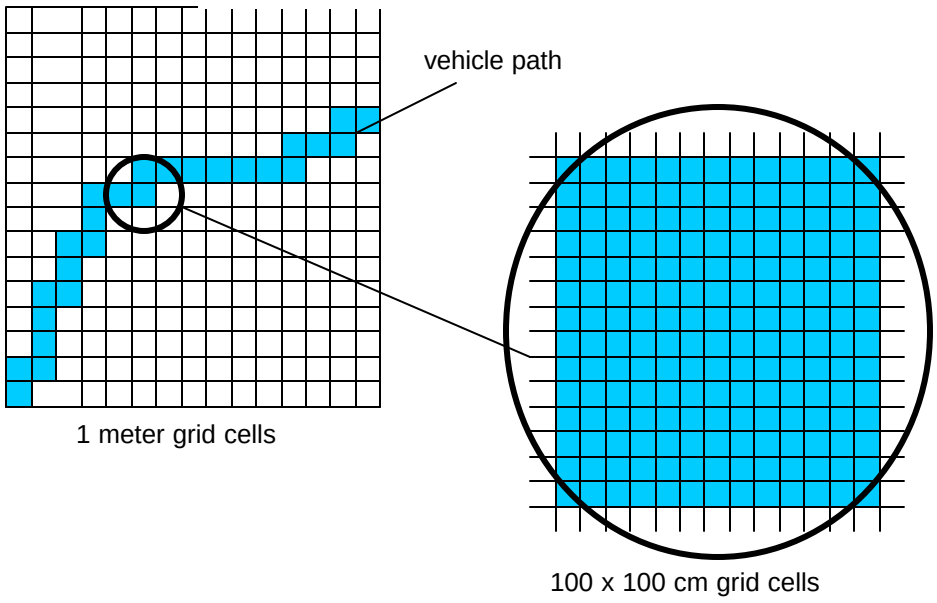
The Init(…) function defines the size of the gaming area along with coordinates which assign the lower-left corner. Functions are included to set and retrieve deltaZ (the terrain deformation) along with a function to increment the value of deltaZ. There is a pointer, mCells, which points to a two-dimensional array of 1 meter x 1 meter cells. This array, which covers the gaming surface, is created during the Init call.

The class, mCell, is defined as:

```
class mCell
{
    public:
        mCell(void);
        ~mCell(void);

        bool
    InitHiResGrid(void);
        void
    SetDeltaZ(float x, float y, float
newDeltaZ);
        void
    AddDeltaZ(float x, float y, float
newDeltaZ);
        float
    GetDeltaZ(float x, float y);
        bool
    Destroy(void);
    private:
        cmCell    **cmCells;
};
```

FIGURE 8 Terra-mechanical grid cell



where the standard set, retrieve and increment deltaZ functions exist, along with a function InitHiResGrid and a pointer to a two-dimensional array of 1 cm x 1 cm cells. Initially, the pointer is set to NULL in the constructor. As a vehicle enters this 1 sq. meter cell, a 100 x 100 array of cm cells is created. Thus, we only create terrain memory down to the 1 cm cell size when we have entered a 1 sq. meter cell. The cmCell class is defined as:

```
class cmCell
{
    public:
        cmCell(void);
        void      SetDeltaZ(float);
        void      AddDeltaZ(float);
        float     GetDeltaZ(void);
    private:
        float deltaZ;
};
```

It is in the cmCell structure that the deltaZ values actually exist for each 1 sq. cm cell.

Experimental runs using a 3.05 GHz, Pentium IV PC running Windows XP and the code developed in Microsoft Visual C++ yielded results of approximately 3.4 mSec to create each one cm cell and write an initial deltaZ value to it. This would be similar to the first run over virgin terrain. When the vehicle crosses (or follows) a previous path, the time to write the updated deltaZ drops to approximately

2.9 mSec. This drop in time is due to the fact that the memory to store each cmCell only has to be created once. These experimental results should be taken as just that. The authors feel that these times can (and should) be further decreased by optimizing the code.

CONCLUSION

As the Army transforms itself to meet the needs of its soldiers through the Future Combat Systems initiative, researchers and engineers are enhancing their capabilities by studying vehicles, terrains and the associated human interactions in a virtual environment. They are utilizing modeling and simulation tools to improve vehicle, terrain and human simulation capabilities. However, for the research and development to be valid, the model must be valid. Scientists and engineers from the GVSL validated that at low frequencies, the virtual Churchville B does indeed correlate to the real Churchville B at APG. With the addition of terramechanics to the simulation, off-road simulation of a large group of vehicles will properly reflect the damage to the terrain surface that is caused and how that effects the movement of the group. The GVSL team, along with its STO research partners, will continue working on creating, improving and validating other virtual proving grounds and off-road technologies to create diverse visual and terrain databases to implement in their research.

REFERENCES

1. Department of Defense Instruction (DoDI) 5000.61: DoD Modeling and Simulation (M&S) Verification, Validation, Accreditation (VVA), April 1996

2. “Determining Test Course Severity by Measuring the Course Profile”, Aberdeen Test Center (ATC) White Paper

3. Reid, A.A. (1999). “Correlated Parametric Terrain Surface Development using Non-Uniform Rational B-Splines and Spectral Fractal Geometry.” Ph.D. Dissertation, Oakland University, Rochester, MI.

4. “Vehicle Test Course Severity”, Test Operation Procedure 1-1-010, April 1987

5.Website http://www.tacom.army.mil/tardec/nac/teams/mbt/overview.htm