Surf Zone Modeling for an EFV Trainer for the USMC

Lawrence M. Lachman  
MultiGen-Paradigm  
Richardson, Texas  
lawrence.lachman@multigen.com

ABSTRACT

Simulating open water is very common in realtime simulations, where it is done with varying degrees of fidelity. However, when typical ocean simulations get close to the shore they experience problems such as: water undulating through the ground, wave activity that does not account for depth, no cues for breaking waves, no dampening of the water at the beach, and no surf.

Ocean water simulation is a computationally expensive process and a difficult problem for applications that require visually plausible three-dimensional effects at high frame rates. Modeling shallow water differs significantly from deep water because the wave interactions with the seabed that give the surface its shape are immensely complex. As a result, simulations that have attempted to model shallow water at realtime frame rates have been limited to simulating one long crested wave in a small body of water, and relying on preprocessing of wave surface profiles offline.

Recently, MultiGen-Paradigm, in conjunction with HART Technologies, developed the training simulation for the Expeditionary Fighting Vehicle for the USMC, under a contract from General Dynamics Amphibious Systems. A surf zone simulation is critical for appropriately training the Marines because the EFV must travel directly from ships at sea to inland objectives.

The background for the training system, the full set of surf zone requirements, and our creative solution that solves all of the problems discussed above, are presented. Specifically, our solution features a seamless transition from deep water to shallow water, transition at shoreline where water tapers to zero height at zero depth, visual cues that the wave has broken or spilled into the trough of the wave, near vertical wave fronts that result as a function of the bottom depth, controls for slope of the littoral zone, and sand bars with a shallower depth than the surrounding area.

ABOUT THE AUTHOR

Lawrence Lachman is a senior software engineer at MultiGen-Paradigm. He joined MultiGen-Paradigm in 1995 and has been the architect and project lead for Vega Prime Marine, a realtime ocean simulation module for Vega Prime, since 2002. His research has been primarily focused on simulating ocean waves, computer graphics, shader technology, and CPU optimization, and has led to the delivery of multiple first-to-market technologies. Prior to 2002 he was a charter member of the team that designed and developed FlightIG, and was one of the principal architects of the image generator.

Lawrence began his career with IBM in 1987. In eight years at IBM he designed and developed innovative software solutions in a product-oriented, R&D environment, and was awarded eight patents in the areas of multimedia, software engineering, computer security, and graphical user interface enhancement.

Lawrence received a B.A. in Computer Sciences from The University of Texas at Austin in 1991, where he completed graduate work in computer graphics that included advanced work in ray tracing.
Surf Zone Modeling for an EFV Trainer for the USMC

Lawrence M. Lachman
MultiGen-Paradigm
Richardson, Texas
lawrence.lachman@multigen.com

INTRODUCTION

Development of the modeling and simulation of the surf zone for the EFV Training System for the USMC was recently completed. A surf zone simulator was critical to the success of the training system because the EFV – the newest USMC amphibious vehicle – can be launched at sea, from an amphibious transport dock ship beyond the horizon, and is able to transport Marine infantry units to shore (and then return to sea). Once ashore, EFVs are capable of participating in the full spectrum of ground combat operations.

The research and development for modeling the surf zone began in May 2005. The impetus for the research and development was two-fold: an ambitious set of requirements for surf zone simulation that far exceeded any previous capabilities in the realtime, maritime simulation industry, and aggressive deadlines for delivery of the surf zone to the USMC.

BACKGROUND

In June 1996, under contract from the United States Marine Corps, General Dynamics Amphibious Systems (GDAMS) began development and construction of the Expeditionary Fighting Vehicle (EFV), formerly known as the Advanced Amphibious Assault Vehicle (AAAV). The first prototype was rolled out three years later.

In June 2004, GDAMS was awarded a contract for the development of training systems for the EFV, with a targeted completion date of December 2006. The contract called for them to design, develop, fabricate, test, install and integrate the EFV training systems.

GDAMS subcontracted with HART Technologies to design, develop, integrate and test the EFV training system software baseline, and selected MultiGen-Paradigm to supply the visual system for the EFV training system. However, GDAMS faced aggressive deadlines in 2005 for delivery of the modeling and simulation of the surf zone that would not be met because no existing realtime, maritime simulation software provided the required feature set. As a result, MultiGen-Paradigm was contracted to accelerate the development of modeling and simulating the surf zone. Research and development commenced in May 2005 with the goal to transform the most significant risk item to a success in just several months. That goal resulted in an extraordinarily focused development that produced the critical set of features that are described in this paper.

THE EFV

Along with the MV-22 Osprey and Landing Craft Air Cushion (LCAC), the EFV is an integral component of the amphibious triad required to execute the Marine Corps’ “over the horizon” strategy for ocean based assaults and naval ship protection. The EFV allows immediate, high-speed maneuvering of Marine infantry units as they emerge from ships located over the visual horizon (25 miles and beyond).

Figure 1. USMC Expeditionary Fighting Vehicle with the bow extended into the hydroplaning position.

Two platform variants were developed: the personnel variant, and a command and control variant.
The baseline EFV Personnel variant is an armored, tracked, amphibious combat vehicle that provides motion stabilized fire power while moving. It is armed with a 30 mm cannon and a 7.62 mm machine gun and is intended to transport 17–18 combat-equipped Marines and a three-man crew (driver, gunner, vehicle commander).

The EFV Command variant was designed to transport a commander and battle staff. It provides them with mobile capabilities in command and control electronics, seven computer workstations, and intelligence, but lacks the 30 mm cannon.

The salient operational capability of these vehicles for this discussion is that they were designed to rapidly transport troops and/or cargo from ship-to-shore from over-the-horizon in less than one hour.

The EFV is capable of high water speeds (20–25 knots) and normal amphibious characteristics (e.g., seaworthiness) for ship-to-shore activities. It is also capable of speeds up to 45 mph on land, and possesses land mobility characteristics comparable to the contemporary main battlefield tank.

A smooth transition from water to cross-country movement has always been a difficult and dangerous task for amphibious vehicles. The EFV design solves this problem by the automatic transfer of power from the high-speed water jets to the vehicle tracks.

The EFV features a unique combination of offensive firepower, armor, and nuclear, biological, and chemical collective protection, significantly enhancing battle space dominance by Marine forces. Its high-speed mobility on land and sea represent major breakthroughs in the ability of Naval and Marine Corps expeditionary forces to exploit the intervening sea and land terrain in order to achieve the element of surprise, rapidly penetrate weak points in the enemy’s littoral defenses, and seize operational objectives.

TECHNICAL REQUIREMENTS

The key requirements for modeling the surf zone, identified in the requirements document for the trainer [1] were:

1. Display three-dimensional surf waves, defined in wave heights and periods, to support the operational training environment.
2. Transition seamlessly from deep water to shallow water.
3. Transition at shoreline; water tapers to zero height at zero depth.
4. Visualize almost vertical wave fronts that result as a function of the bottom depth.
5. Provide visual cues that the wave has broken or spilled into the trough of the wave.
6. Allow query of the dominant wave direction and the group phase velocity in order to account for the force of drift on the vehicle.
7. In lieu of providing bathymetry, control the slope of the littoral zone.
8. Within the littoral zone, provide the ability to specify a width and length of a rectangular ridge simulating a sand bar with a shallower depth than the surrounding area.
9. Render a fully synchronized five-channel simulation utilizing five computers for contiguous rendering across multiple displays: Four out-the-window channels and one driver thermal viewer channel, which is a FLIR (Forward Looking Infrared), monochrome channel.

While GDAMS desired to model the surf zone off the coast of Camp Pendleton, CA (home of the Marine Corps base and the 1st Marine Expeditionary Force), they did not wish to restrict the training system to the actual geo-specific bathymetry of Camp Pendleton’s very shallow littoral zone; thus, the desire to control the slope of the littoral zone.

The minimum ocean dimensions called for a three kilometer coastline and an ocean that extended (out to sea) to the visual horizon (i.e., 25 nautical miles) in order to support over the horizon navigation. Because the eyepoint was to be no higher than the actual EFV Driver Station and/or EFV Turret Station, the maximum visual range was four kilometers.

TECHNICAL APPROACH

A technical approach for the development of the modeling of the surf zone was formulated in order to address the key requirements as outlined above. While this ambitious set of requirements set the bar for ocean scene simulation higher than any previous capabilities in the industry, the key requirement which drove the design at all levels was realtime performance.

The primary elements of this technical approach included:

- Based on the ambitious schedule, build on the foundation of an existing realtime, maritime simulation software application that already...
provided the functionality and capabilities required for simulating infinitely deep ocean water, as well as constructing and rendering the visual scene from the perspective of one or more observers. Vega Prime Marine was chosen because it satisfied those requirements.

- Design a flexible solution that would be able to model realistic shorelines (regardless of whether they have a shallow sloped or a steep sloped bottom), require no preprocessing of the subsurface terrain in order to achieve the realtime performance target, and simulate large bodies of water.

Wave Model for Ocean Waves

There are many models for ocean behavior, with each having multiple inherent assumptions and initial conditions, such that no single approach can be used to generate accurate, realistic results in varying environmental conditions. This is inherent in the nature of the ocean.

Our solution employs an oceanographic statistics-based surface wave model for ocean waves. This model allows control of sea state, including distributions of direction, height, wavelength, and alignment with the wind. The algorithm is explained in detail in Tessendorf's "Simulating Ocean Water" [6]. This approach synthesizes and evolves a patch (mesh) of ocean waves from a Fast Fourier Transform (FFT) precept, using a regular grid with user-controllable size and resolution, and yields a periodic height field that can be tiled seamlessly over a larger domain. It starts in the frequency domain, using the Phillips spectrum, and then transforms a Fourier domain description of the ocean water to the spatial domain via the inverse FFT.

The general equation for a sinusoidal wave is:

\[ \zeta(\mathbf{X}) = A \cos(\mathbf{kX} - \omega t) \]  

where

- \( \zeta(\mathbf{X}) \) is the wave elevation at geographic location \( \mathbf{X} \)
- \( A \) is the amplitude
- \( \mathbf{k} \) is the wavenumber
- \( \mathbf{X} \) is the geographic location \( (x \cos\theta + y \sin\theta) \)
- \( \omega \) is the angular frequency in radians per second
- \( t \) is the elapsed time since the beginning of the simulation, and \( \omega t \) is the time variation.

Works by Mastin [2] and Stewart [5] show that the concept of a Fourier series can be expanded to include series that represent surfaces \( \zeta(\mathbf{x}, \mathbf{y}) \), where any surface can be represented as an infinite series of sine and cosine functions oriented in all possible directions. Wavelengths and wave frequencies are related through the dispersion relationship, and accordingly, the sea surface can also be represented as an infinite sum of sine and cosine functions of different frequencies moving in all directions.

Thus, the Fourier series is more than just a mathematical expression; it states that the sea surface is truly composed of sine waves, each one propagating according to equation (1).

Tessendorf [6] then shows that the most efficient way to implement the evolution is in the Fourier domain, which, for this application, ensured that realtime performance requirements would be met.

Surf Model for Ocean Waves

One of the more elusive phenomena in realtime visual simulations has been the realistic simulation of shallow water behavior. Physics-based wave models work reasonably well when modeling infinitely deep water, but these procedural models break down when attempting to model shallow water waves because they are so computationally expensive. This is especially true when attempting to model large bodies of water, as the numerical model must be evaluated at each vertex, as it is dependent upon time and space.

This solution not only solves the surface wave equations that can handle propagation of waves in a shallow water area, but also models the somewhat short crested character of the shallow water waves at realtime frame rates.

While the wave height amplitudes are computed using the Phillips spectrum (equation 40 in [6]), our solution extends the Phillips spectrum [7]. The nature of the extension is to:

- Parameterize the power-law distribution of wave heights and the alignment exponent, which controls the alignment of the waves to the direction of the wind.
- Add travel to suppress waves moving away from the wind direction.
• Account for the bottom depth.

For realistic wave behavior, we model the following phenomena as waves approach the shore:

• The amplitude of the waves increases, while the spacing between waves gets shorter. Both are controlled by factors in the wave spectrum.
• The waves tend to reorient themselves to align with the shoreline, a result of the variation of the speed of the waves with respect to depth.
• The waves in shallower water move slower than the waves in the deep water because of the depth-altered dispersion relationship.
• As a wave begins to break, foaming white water forms just ahead of the crest of the wave. After the wave has broken, foam is left on the water surface.

The magnitude of each of these behaviors depends on the details of the shape and depth of the previously defined ocean floor. A realtime simulation package will never be an extremely accurate simulation of these phenomena. The wave interactions with the subsurface terrain that give the surface its shape are immensely complex and have to be approximated and simplified for realtime software. Consequently, our solution produces each of these behaviors to some degree.

Shallow water modeling requires a much denser mesh (than does infinitely deep water) to capture the phenomena discussed above, leading to a significantly increased draw time (requiring that realtime performance tradeoffs be considered).

The Littoral Zone

The littoral zone is the part of the ocean closest to the shore; from the shoreline to approximately 200 meters out into the water and is divided into three zones:

• The supralittoral zone
• The intertidal zone
• The sublittoral zone.

The supralittoral zone is only underwater during unusually high tides or during storms. It starts at the high tide line and goes toward dry land. The intertidal zone is between the high tide and low tide lines. The sublittoral zone extends from the low tide line out to about 200 meters.

Our solution models the littoral zone as it occurs in nature, allowing it to be divided up into all three zones.

Figure 2. Friction between the water and the seafloor in the littoral zone produce shallow water waves.

Specifying the slope of the littoral zone leads to the automatic generation of bathymetry that corresponds to the provided slope. Each zone may have a different slope.

THE SURF ZONE FRAMEWORK

The surf zone framework consists of four main components: an ocean, a wave generator, breaking waves, and sandbars. In order for our solution to run at realtime frame rates, the framework:

• Employs Fourier synthesis to generate oceans, which enables meshes to be periodic and ocean cells to be tiled seamlessly over a larger domain.
• Exploits mesh periodicity in the deep water region and in the littoral zone.
• Constructs the ocean asynchronously, in a local coordinate frame that is independent of the orientation of the shoreline.
• Renders the ocean using range-based level of detail and adaptive triangle meshes.

The Ocean

Our discussion begins with the terminology used in the context of simulating ocean water.

• An ocean is a single level (of detail) rectangular grid of cells. The ocean is divided into columns and rows (the grid structure), which define the individual cells of the grid (figure 3a).
• A cell is a subdivision of the grid structure. Each cell has a unique column and row number (address). A cell is independent of the geometry inside of it; it simply identifies the boundaries and
address of a location within the grid structure (figure 3b).

- A mesh is the geometry that occupies the cell of a grid structure. A mesh is a collection of uniformly spaced, discrete points arranged in a roughly rectangular pattern (figure 3c). The points of the mesh correspond to multi-component data elements (e.g., height (1D), horizontal position (2D), and normal (3D)). While the horizontal dimensions (extents) of a mesh are constant throughout an ocean grid, each mesh may have a different resolution (level of detail) that is a function of range from an observer (figure 3d).

- A mesh point is the smallest indivisible element of a mesh (figure 3e); while four mesh points comprise a mesh facet (figure 3f).

Figure 3. Topology of the ocean. (a) An ocean with a grid structure of 2x3 cells. (b) Cell (1, 0) of the grid structure representing a physical region. (c) A mesh with a resolution of 8x8, co-located in cell (1, 0). (d) Two meshes with different resolutions; cell (1, 0) with a mesh resolution of 8x8, and cell (2, 1) with a mesh resolution of 4x4. (e) A mesh point. (f) A mesh facet.

The ocean can be translated and oriented to align with a beach shoreline model. Once rotated, its x and y axes no longer align with the west-east and north-south axes. This allows the ocean to be aligned with any arbitrarily oriented, yet fairly straight shoreline.

The ocean geometry is generated in an asynchronous thread, and the underlying geometry is unique per observer. This enables the cell's level of detail and optics (i.e., the reflection, refraction, and propagation of light) to be unique for each observer.

Each ocean cell is aware of the level of detail of its underlying geometry and that of its two neighbors (to the east and to the north). Thus, each ocean cell is rendered, potentially, in a different level of polygonal detail each frame. If an east and/or north neighbor cell exists, then the right and/or top of the mesh is stitched correctly to correspond to the resolution of the corresponding neighbor.

The Wave Generator

An ocean instance contains a wave generator. The wave generator acts as the interface to the various controls and data generators for basic, choppy, and shallow water waves. It provides the user complete control of sea state, including distributions of direction, height, wavelength, and alignment with the wind, and is capable of modeling the 13 sea states described by the Beaufort scale, or the nine sea states described by the Spectral Ocean Wave Model (SOWM) [3].

The wave generator manages two regions: the littoral zone and the deep water region. It employs Fourier synthesis to generate its oceans, which enables meshes
to be periodic and ocean cells to be tiled seamlessly over a larger domain.

This realtime solution exploits mesh periodicity in the deep water region by modeling one ocean cell and then tiling it horizontally and vertically throughout the entire region. In the littoral zone, because each zone may have a different slope, bathymetry is automatically generated for one ocean cell to correspond to each zone’s slope. The accurately modeled shallow water surface of that cell is then tiled (horizontally) across the entire zone.

Zones are comprised of one or more shallow water regions. A region’s physical width is a multiple of the wave generator's cell extent in the x dimension (its physical width).

It is the wave generator’s responsibility to:

- Seamlessly continue shallow waves into infinitely deep water where no depth profile exists.
- Clamp the water’s elevation (in the wave troughs) to the depth of the seabed (thereby preventing the water from going underground).
- Taper the wave height at the shore.

Once the waves have fallen to zero amplitude (when the bottom depth has reached zero) the wave patterns then exhibit the behavior of a cusp surf, where the cusp surf sporadically washes farther up inland in various places and then dissipates / retreats back into the ocean.

Along the coastline the opacity of the water is modulated as a function of depth, generally more transparent in the shallowest depths. Then it is modulated based on the viewing angle, so that the water is more transparent when viewed straight on than when viewed at a glancing angle. As a result, there is not a distinct line between where the ocean ends and the beach begins, increasing the realism.

**Breaking Waves**

A breaking wave is caused by the water particles at the top of the wave traveling at a much higher velocity than those at the bottom of the wave, eventually curling and toppling over (as there is no volume below to support these particles).

A challenging task in developing this framework was determining when and where a wave was breaking and how to best generate the visual cues associated with it.

The wave generator manages a set of breaking waves that provide visual cues that a wave has broken or spilled into the trough of the wave. Peak detection and breaking is accurately identified by using the minimum eigenvalue [6] and a user-controllable peak threshold factor.

The visual cues include spray dynamics, water motion, turbulence, and foam in the trough of the wave, and are simulated via a particle system animation. Vertex and fragment shaders are used to render the white caps, foam, and backwash, as well as view dependent, realistic optics and shading that account for the actual depth of the water (including the semi-transparent water for shallow and clear water areas along the shoreline). Figures 5 and 6 show examples of these visual cues.

The particle emission process is performed in three stages.

- In the first stage, the ocean surface (within the littoral zone) is examined to determine where each
wave has broken, and thus, from where particles should be emitted.

- In the second stage, emitted particles are initialized with their position set to be relative to their location on the crest of the wave, and their velocity set to the group phase velocity of the wave.

- In the third stage, the particle dynamics process takes over, using the forces of the initial velocity, gravity, and wind, to move and evolve each particle.

Once a particle has been emitted, it no longer interacts with the ocean surface, which (as it is for infinitely deep water) continues to be modeled as a contiguous surface.

Our realtime solution does not attempt to reproduce the physics of wave breaking and the energy loss associated with it. Consequently, it does not differentiate between the various breaker surfs (spilling, plunging, or surging). However, the breaking wave kinematics do account for the wave's height, group velocity, and underlying water depth, which do influence the breaking wave’s growth and decay with time.

The distribution of breaking waves is user-controlled, as is the radius of the area of interest (from the observer) in which breaking waves are simulated.

This approach allows the user to customize the breaking wave effects by supplying a particle system effect template (instead of using the default breaking wave effect template). This solution also allows the user to customize the appearance or kinematics of individual breaking waves – as they are detected – that have broken somewhere on the ocean's surface.

The following wave generator properties strongly influence the size and shape of the breaking wave, and whether the breaking wave is even detected: peak detection, sea state, modal period, choppy waves factor, significant wave height, and surface wind speed.

Sandbars

Any region of the littoral zone can contain a sandbar. When a slope is specified, bathymetry is generated for one region of the littoral zone (equal to the physical dimensions of one ocean cell), the water surface is accurately modeled, and that surface patch is then tiled (repeated) horizontally along the shoreline. As a result, the sandbar, which has become an inherent part of the bottom topography, is repeated as well throughout this region if the region spans more than one ocean cell.

![Figure 7. The sandbar is defined within the context of region bathymetry, which is per cell. So if a region’s width is more than one cell in width the sandbar will be repeated.](image)

Within a zone, because each region has the same bottom topography and slope, the only benefit of defining multiple regions is to have a sandbar in one region, but not others.

Sandbars are centered within a cell, and are defined by their width, length, depth along top and distance from the shoreline. Outside of the width and length, the sandbar slopes down over a precomputed distance enabling the wave interactions with the seabed to gradually account for existence of the sandbar and alter the shape of the water’s surface accordingly.

![Figure 8. (a) How the bottom topography of the depth profile is modified. The depth along the top is measured from the tide height of the ocean. (b) The horizontal width in the x dimension, the horizontal length in the y dimension, and the distance from the shore.](image)

Wherever sandbars are present, the wave crests peak as the waves roll over the bar. The water depth over the sandbar and the wave height determine whether or not breaking takes place on or near the bar. If the water depth over the bar is more than twice the significant breaker height, nearly all waves will pass over the bar without breaking but the crest will peak up distinctly. If the depth is between one and two times the breaker
height, waves will break near the bar, some on the bar itself and others on the shoreward side. With water depth less than the breaker height all waves will break on the seaward side.

Shallow sandbars can seriously impede or stop most landing craft, but have no effect on EFVs. However, EVF unit leaders need to consider the effect of surf beat when sandbars are present. Surf beat is the periodic rise and fall in coastal water levels caused by two or more wave trains arriving at a sandbar or the shoreline. The rising and lowering action can cause the EFV to bang down hard on a sandbar.

Database Considerations and Beach Modeling Guidelines

When the Camp Pendleton 3D beach model was created to go along with the ocean created by our solution, certain rules had to be followed to ensure that the beach / ocean transition looked appealing and that there were no visual anomalies (with respect to the beach / ocean interaction).

Beach modeling can be reduced to a two dimensional problem by thinking in terms of a side view. Figure 9 is an example of this view. It contains three line segments each of which are important to the topic of beach modeling.

Figure 9. This 2D representation of a shoreline can be extruded for the entire length of the coast.

Segment A is called the Dry Land Zone, B is the Transition Zone, and C is the Ramp Zone. The most important part of each zone is the slope. Typically A and C are sloped slightly downwards from left to right while B is level.

The ocean is oriented such that the water near the shore is just slightly above the land, as shown in figure 10.

Figure 10. In the real world the water is right on top of the land, of course, but in 3D simulation z-fighting occurs if the water and land polygons are too close. So the water and land are separated by a few centimeters.

The water will be level as it gets closer to the land, so the land should be level as it meets the water – this is segment B, the Transition Zone.

The ocean, including the cusp surf, should come to a stop just before it reaches the end of the Transition Zone. The cusp surf sporadically washes farther up inland in various places and then dissipates / retreats back into the ocean. Since the land slopes upwards inland, if the ocean proceeds too far (that is, into the Dry Land Zone, segment A) there will be a distinct visible line between water and land (as a portion of the water will now be under the land). When the ocean stops just short of the Dry Land Zone, the water, which fades into transparency, will seamlessly blend into land giving the correct visual cue. The Transition Zone needs to be of appropriate length (several meters); otherwise an observer (viewing the ocean from land) may be able to see a “hole” in the database. This occurs when the observer can see the gap between the water and land (where the water is semi-transparent) and can be overcome by lengthening the Transition Zone (out towards the sea).

The Dry Land Zone is sloped down from the main land mass to the ocean to mimic nature, in which the sea shore is indeed sloped downwards to the water. This zone is normally sloped by only a few degrees, but can be more aggressively sloped if needed.

The Ramp Zone, segment C, is optional. It provides ground polygons, with which a seafaring entity can interact (or the runtime can intersect or collide). For this application, it was essential, since the EFV can travel on land and in water, and needs to follow the terrain as it moves from the land into the ocean until its buoyancy allows it to float. Prior to that point, the vehicle follows the land, which, in nature, slopes downwards into the ocean. If a Ramp Zone is modeled, the slope should be at least as great as the largest planned slope of the littoral zone. If the slope of the Ramp Zone is not steep enough, the troughs of the waves will pass below the Ramp and the sea floor will be visible (the solution would be to increase the Ramp Zone slope).

The Ramp Zone of the Camp Pendleton database was designed to accommodate a maximum (very steep) slope of -0.5, which translates to a depth of 50 meters at 100 meters from the shoreline.

While training exercises are planned in five different configurations (very shallow, shallow, medium, steep, and very steep), the Ramp Zone of the Camp Pendleton database was designed to accommodate a maximum (very steep) slope of -0.5, which translates to a depth of 50 meters at 100 meters from the shoreline.
Once the 2D concepts of shoreline creation are understood, the user can easily extend them into three dimensions. For this application, the 2D shape was extruded to cover the entire length of the shoreline. After the shore was extruded into three dimensions, variations were made to the polygonal beach to make it more realistic. However, since the realtime ocean transitions at the end with the height of the water tapered to zero height at zero depth, the Transition Range area still needed to be level and fairly straight throughout the length of the shoreline – otherwise there would have been holes and/or hard edges where the land and water meet.

While this approach requires the shoreline to be fairly straight, it places no restrictions upon its orientation (i.e., the entire shoreline can be oriented along any heading). For example, the Camp Pendleton shoreline runs from roughly 32.3° North, 117.5° West to 33.3° North, 117.4° West. In a Cartesian coordinate system, this orientation translates to a 125.518° angle (counter-clockwise) from the x-axis. This orientation needed to be preserved because the training simulation features full terrain correlation with SAF (Semi-Automated Forces) tactical operations and training.

The surf zone ocean can be oriented to match the orientation of the shoreline, allowing a realistic surf zone to be created (figure 11).

In the end, a realistic 3D beach model was created, with correctly sloped zones that interact appropriately with the realtime ocean model to generate a viable surf zone simulation.

**CONCLUSION**

In this paper we presented the background for the EFV training system, the full set of surf zone requirements, and our solution.

The challenge of this work was to meet an ambitious set of requirements for surf zone simulation that far exceeded any previous capabilities in the realtime, maritime simulation industry, and do so under intense deadline pressure. This resulted in a focused effort that produced the set of features critical to realistic surf zone simulation.

The following factors were paramount in enabling our solution to run at realtime frame rates:

- We employ Fourier synthesis to generate our oceans, which enables meshes to be periodic and ocean cells to be tiled seamlessly over a larger domain.
- We exploit mesh periodicity in the deep water region and in the littoral zone.
- We construct the ocean asynchronously, in a local coordinate frame that is independent of the orientation of the shoreline.
- We render the ocean using range-based level of detail and adaptive triangle meshes.

While this solution automatically generates bathymetry that corresponds to the provided slope, this could be extended to allow users to supply a bottom depth profile for the varying bottom of the ocean over one or more regions of the littoral zone.

Our solution was implemented using C++, Vega Prime, VSG, OpenGL, and Cg on a dual Intel Xeon processor (2.8 GHz) workstation and nVIDIA GeForce 7800 GTX graphics hardware. Our frame rate was consistently very close to 60 Hz, and even when rendering the most complex ocean scenes never fell below 30 Hz.
ACKNOWLEDGEMENTS

The author would like to thank Eric Hirschorn, Anthony Hinton, Kevin Harris, Brett Chladny, Dan Oller, David Dao, and Raymond Casady, of MultiGen-Paradigm, for their invaluable contributions to this project. Dan Oller also contributed to this paper. Special thanks to Doug Price of MultiGen-Paradigm, Dave Lincoln of GDAMS, and (Bird Dog) Anne Woolstenhulme of Rockwell Collins, for their thorough review of the paper. Vega Prime Marine’s native modeling of the wave environment was created using WaveTools V2, developed by Finelight Visual Technology.

Finally, thanks to our industry partner HART Technologies. HART Technologies was responsible for the development of the EFV’s true six degree of freedom hydrodynamic and vehicle response model (which includes displacement and planing characteristics), as well as the construction of the Camp Pendleton terrain and beach database.

REFERENCES


