Using Real-time Physics to Enhance Game Based Effects

John Mann, Jeff Lyons
Applied Research Associates, Inc.
Orlando, FL
jmann@ara.com, jlyons@ara.com

Julio de la Cruz
RDECOM STTC
Orlando, FL
julio.delacruz@us.army.mil

ABSTRACT

Advanced graphics processors, multi-core processors, and game physics engines have contributed to the rapid growth of games that inch ever closer to replicating real world physical interactions. Vehicles collide with objects and display realistic damage, structures hit by weapons crumble realistically, and trees bend and sway in the wind and snap off when fired at with small arms. The sources of these effects are sophisticated computational physics algorithms available to mainstream game developers through physics engines such as those developed by Havok and Ageia.

In order to simplify the game environments where these effects are used, many assumptions are made about the physical properties and the interactions of objects in these environments. If the game is being used purely for entertainment or if it is being used as a training tool where the realism of the interactions is not critical, these simplifying assumptions are acceptable. However, there are valid reasons to replace simplifying assumptions of the environments and physical responses of objects in the game with more realistic physical models. If the game is being used for experimentation where certain effects can alter the outcome of an experiment, better models may be warranted. Physics-based effects are also valuable for training where accurate weapons effects are important to training requirements.

In this paper, we discuss our research into the implementation of real-time physics models in a game environment. The objectives of this research are to demonstrate the feasibility of using physics-based weapons effects models in a game engine and to develop an approach for optimizing the models for real-time response. The US Army Research & Development Engineering Command Simulation and Training Technology Center (RDECOM –STTC) is currently researching and evaluating these real-time models to support experimentation and training.

ABOUT THE AUTHORS

John Mann leads the Real-time Physics Simulation Group in ARA's Central Florida Division. In his current role, he directs ARA's efforts in the development and integration of physics-based weapons effects models into Army programs. He has been involved with the development and integration of physics-based urban modeling and weapon-target interaction modeling tools for over 11 years. During his tenure at ARA, he has been the principal investigator for several projects, including the Urban and Underground Model Generator (U2MG) project and the Combat Training Center Objective Instrumentation System (CTC-OIS) Area Weapons Effects Simulation (AWES). Mr. Mann has a B.S. in Computer Science from the University of Maryland.

Jeff Lyons is a principal engineer at Applied Research Associates. He has a Bachelor of Science degree in mechanical engineering from Florida State University and a Masters in mechanical engineering from the Massachusetts Institute of Technology. Jeff's experience spans the mechanical engineering and software engineering disciplines. He has experience in structural analysis, CAD design, manufacturing support, and test support. He also has worked extensively in simulation and software design for military training and testing systems.

Julio de la Cruz is the Science and Technology Manager for Synthetic Natural Environment and Simulation Technologies at RDECOM STTC. Mr. de la Cruz has a Bachelor of Science degree in Electrical Engineering from the City College of New York and a Master of Science degree in Industrial Engineering from Texas A&M University. Mr. de la Cruz has over eighteen years experience in the government and private sector. Mr. de la Cruz's previous positions include lead researcher for Army Technology Objectives (ATOs) focused in the core simulation areas of Synthetic Natural Environment and Computer Generated Forces. As the Deputy Division Chief and Manager for Synthetic Natural Environment at the Simulation Technology and Training Center, Research Development and Engineering command (RDECOM-STTC) he oversees research and development of current Database Generation Systems/tools technology and supports the Command to address the defined Army needs in Modeling and Simulation.

Using Real-time Physics to Enhance Game Based Effects

John Mann, Jeff Lyons
Applied Research Associates, Inc.
Orlando, FL
jmann@ara.com, jlyons@ara.com

Julio de la Cruz
RDECOM STTC
Orlanod, FL
julio.delacruz@us.army.mil

INTRODUCTION

In a common counter IED training scenario, a convoy drives through debris filled Iraqi streets when an IED explodes on the lead vehicle (Figure 1). Chaos erupts as insurgents fire upon the convoy vehicles, attempting to add to the confusion and danger. Amidst the chaos, troops are learning to act decisively and resist a natural tendency to panic and react in a way that could endanger them or their fellow Soldiers (Zajac et al, 2005).

Figure 1 Top: Scene from Kuma War, a convoy simulation created for the US Army Combined Arms Support Command. Bottom: A VBS/2 counter-IED training simulation created for the Marine Corps.

These training scenarios are extremely valuable in teaching troops to respond appropriately to an IED attack (Miller, 2008). Until recently, the level of effort to recreate these training scenarios for a virtual environment was high, requiring months of artist time. With the progress that has been made in implementing physics effects, it is now possible to produce many of these effects with much less work on the part of artists, allowing them to focus on developing more realistic models and textures.

This high level of realism provided by physics models provides a very useful capability to the military, where game technologies are increasingly used to prepare Soldiers for more intensive live training exercises or when a rapid deployment schedule does not permit live training. Games are helping junior leaders develop judgment and decision making skills (Johnson, 2008; Wampler et al 2006).

The use of physics models in military simulations is not a new concept. For many years, the military has used finite element analysis and computational fluid dynamics to simulate the complex effects of munitions. What is new is the use of validated physics models in a real-time game environment. The relatively recent phenomena of multi-core processors and high end graphics processing units have made it possible to run these models in real-time.

In this paper, we will discuss a research effort sponsored by the Army RDECOM STTC to leverage validated real-time physics to enhance effects based on game technologies. The objectives of this research are to demonstrate the feasibility of using physics-based weapons effects models in a game engine and to develop an approach for optimizing the models for real-time response.

Effects from an explosion are felt within milliseconds, so to preserve realism, game based explosions should also occur within milliseconds. The term real-time used in this paper means fast enough to be indistinguishable from a live explosion. From the perspective of a user seeing the explosion in the game, real-time effects should not have any perceptible delay. The physics in

real-time physics refer to Newtonian physics, simulating the movement and rotation of objects using variables that include mass, velocity, and friction.

For our research we are evaluating physics engines for use as one component in a comprehensive physics simulation. The physics engine provides methods for simulating the motion of objects and the response of bodies to an applied force. Commercial physics engines include software development kits from Ageia (Ageia, 2007) and Havok (Havok, 2008).

Physics engines do not provide an accurate representation of munition effects and accurate responses of materials to munition events. This is an important point that is easily missed to a casual observer of games. Damage effects represented in most games is generally not representative of effects from real munitions.

The subtle cues of a realistic simulation provide additional information that is important to the overall learning experience. From your own learning experiences, think of all of the subtle cues you have learned after years of driving a car; cues that tell you to speed up to get out of someone's blind spot, slowing down to increase the distance between your car and the car in front of you, knowing when to brake for a yellow light, and merging into traffic at highway speeds. All of these driving techniques can be taught to some extent, but it takes years of driving to perfect them. Applying this concept to Soldier training, more realism provides a greater sense of presence and more cues for learning.

The realism created by validated physics models provides the user of the simulation with a greater sense of presence. This sense of presence can improve the learning outcome if the user feels the consequences of actions in the simulation (Mantovani, 2003). The benefit of realism applies whether the learning is intended to teach decision making skills or for gaining understanding as part of an experiment. We discuss specifics of how our research addresses the I/ITSEC theme of "Learn. Train. Win" in the section titled **Real Time Physics for Training**.

RELATED WORK

The areas of related work can be separated into three primary areas – Army Materiel Systems Analysis Activity (AMSAA) validated vulnerability/lethality algorithms, game based effects, and validated physics models.

AMSAA Algorithms

AMSAA vulnerability/lethality algorithms are documented in the The Physical Model Knowledge Acquisition Documents (AMSAA, 2003). These algorithms were designed to produce effects for munitions against vehicle and personnel targets. The algorithms rely on pre-calculated look-up tables that contain lethal area data for munition/target combinations. By using pre-calculated tables, the AMSAA algorithms are able to produce results very quickly.

The Army uses AMSAA algorithms in several programs, spanning the live, virtual and constructive training domains. One of the first simulations to use the AMSAA algorithms was the Close Combat Tactical Trainer (CCTT), followed by OneSAF. In the live training domains, we have implemented these algorithms for the Army's OneTESS and CTC-OIS programs.

The vulnerability/lethality tables used by the AMSAA algorithms are produced by simulation tools that use high fidelity physics-based algorithms to simulate the velocity and mass of munition fragments and the effect of those fragments on a target. We have used these tools to develop new data tables for IEDs and indirect fire munitions. Our experience in developing and updating the tools for producing this data provided a foundation for the development of real-time physics models.

Game Based Effects

Damage effects in games have evolved from the very simplistic swapping of undamaged and damaged models to keyframe animated damage effects and more recently to complex damage effects produced by physics engines. Model swapping was originally used to avoid the limitations of graphics card performance by sacrificing the realism of animating intermediate damage states. Keyframe animation, while producing intermediate damage states, is limited to the particular effects created by an artist. Unlike model swapping or keyframe animation, physics based equations alone are capable of producing emergent effects that correspond to game play.

Games and other real-time interactive visualization systems commonly utilize an implementation of physics equations that simulate the dynamics of rigid body motion. The most common physics engines, including Havok, PhysX and Bullet (Bullet, 2007) are in fact rigid body dynamics engines. Around a dozen of these

real-time rigid body dynamics engines were available when this paper was written.

Rigid body dynamics allow a game engine to simulate the physical interaction of objects with the game environment. The rigid in "rigid body" refers to the inflexibility of the objects being simulated. A rigid body will respond to an event in the game without changing shape. We can greatly simplify the equations used to simulate physical interactions by ignoring the effects of flexibility. Dynamics refers to the forces applied to the bodies and the responses based on the mass distribution of each interacting body (Hecker, 1996).

We included a study of rigid body dynamics engines under a research effort that began in 2006. For this effort, we developed a proof of concept for a real-time building damage simulation. The simulation modeled structural fracture, building collapse and rubbling effects when the building was hit by a munition (Figure 2). This effort produced a game based visualization based on the OGRE open source rendering engine. We integrated the PhysX Software Development Kit with OGRE to support rigid body dynamics calculations. This research demonstrated the ability to damage arbitrary building structures in real-time using multi-core processors and advanced graphics processing units.

Figure 2 Visualization of rubble and building collapse as shown in the OGRE rendering engine

Validated physics

There is a fundamental difference between physics based effects used purely for visualization and effects intended to simulate real systems. Accurate representation of real systems using physics based effects requires the use of algorithms that are derived from live experiments or high fidelity finite element codes (Mann, et al 2006).

Physics algorithms introduce capabilities that are not possible with low fidelity, pre-calculated table-based approaches. Simulation of complex effects requires variable inputs that change depending on the situation. An example of a complex effect is the simulation of flying glass shards from a blast hitting a window (Figure 3). The effects include the response of a glass window to a blast, the subsequent flyout of glass shards, and injuries resulting from the glass. A table-based approach only captures a limited number of pre-calculated results.

Figure 3 Injuries from glass breakage is one example of an effect that can not be accurately modeled using pre-calculated results

Several of the validated physics models we are using for our research are derived from the Integrated Munitions Effects Assessment (IMEA) tool (Harman, York, 2003). IMEA is used operationally to simulate munition attacks against hardened buildings and bunkers. The models used in this tool have undergone a rigorous validation, verification, and accreditation process where live weapon test results have been compared with IMEA results. The modular architecture of IMEA facilitates reuse of individual models in other simulations.

For over 20 years the Army has relied upon the Modular UNIX-based Vulnerability Estimation Suite (MUVES) (Hanes, 1988) to assess the vulnerability of targets to various weapon systems. MUVES is a comprehensive software package for vulnerability/lethality analysis. Using this tool, it is possible to simulate the flyout of individual fragments from a munition and to model the effects of each

fragment on a vehicle's components. MUVES has been used to generate vulnerability/lethality tables for simulations such as CCTT and OneSAF. These tables define lethal areas for a large number of munition/target pairings, but the high fidelity results MUVES is capable of producing are lost because of limits in the size of the data tables.

REAL TIME PHYSICS FOR TRAINING

Army Research on Benefits of Game Based Training

The Army has strong motivation for spending research dollars on the use of games for training. Live training is very expensive. Live training costs include the training facilities, transporting Soldiers to the facilities, costs for vehicles, training devices, and support personnel. Virtual training in comparison is far less expensive, much more portable, and is capable of providing training in effects that are difficult or impossible to replicate in a live training environment.

The Army RDECOM STTC conducted a recent study to determine the usefulness of multi-player games for asymmetric warfare training. The study found that a multi-player game can prepare troops for more expensive live drills and actual deployment. Games can teach Soldiers judgment, decision making skills, and can help them develop situational awareness (Singer, et al, 2007). The conclusions from this study also echoed the Army Research Institute's conclusion that virtual training enhances the development of cognitive skills (Wampler et al 2006).

One of the deficiencies noted in RDECOM's study was the lack of simulated injuries. Wounds sustained during small unit conflicts can have a significant impact on the actions of Soldiers. The study also noted that Soldiers who participated in the study wanted to see physical effects on terrain, vehicles, and buildings.

Our aim in researching the development of real-time physics-based effects for games is to address some of the deficiencies discussed in the RDECOM study. Physics effects have the potential to revolutionize the way weapons effects are simulated in games. Instead of relying on visual tricks to simulate wounds or damage, physics effects can produce accurate results that correlate to real munitions, methods of employment, and the impact on the environment. This correlation ensures that the effects support the training objectives instead of becoming a distraction.

Current Applications

Many game based training simulations use scripting to control entity behaviors and the environment's response to events. Scripting ensures that objects represented in the environment respond in predictable ways to events. This predictability is critical for training tactics, techniques, and procedures. Any major deviation from the scripted scenario can introduce confusion and negate the lessons that are being taught.

Predictable training scenarios are less beneficial for teaching advanced skills, such as judgment and decision making for asymmetric warfare. In order to advance the simulation beyond a basic level, the environment and entities in the environment must respond with less predictability. Adding un-predictability to entity behavior is usually accomplished either by replacing scripted behaviors with AI or by allowing human operators to control entities. Adding unpredictability to the environment can be accomplished by using physics effects.

There are many current training applications that can benefit from the use of physics effects to simulate unscripted events. The focus of this paper and our current research is on munitions effects, but clearly there are other physics effects that could benefit training applications (e.g. sensor modeling, weather effects, atmospheric effects). Physics-based modeling of munitions effects includes modeling the behavior of a munition, including blast, fragmentation, and incendiary effects and it includes modeling the response of vehicles, structural elements, terrain, and people to munition effects.

Counter IED training simulations could benefit from using physics effects to model different types and sizes of IEDs. IEDs can be constructed of many materials and can vary significantly in their area of effect and method of employment. Suicide bomb vests, vehicle borne IEDs (VBIEDs), daisy chained IEDs, and explosively formed penetrators are a few examples of IED types. Modeling the effects for every possible IED and target combination would require significant effort using pre-calculated tables. Even if all of the combinations of interest were modeled, the effects would be limited by the parameters used to generate the tables (e.g. explosive amount, orientation of IED, target protection levels).

Physics models can represent a large number of IED/target combinations by modeling the physical characteristics of the IED and the physical response of a target. Simulating new IED types and/or target types

requires adding the parameters for the IED and/or target. This involves much less effort than generating new vulnerability/lethality tables. Physics models do not have the size constraints of tables and can simulate an infinite number of parameter combinations.

Medical training simulations can benefit from the use of physics effects. Physics models can simulate different types of wounds resulting from blast, fragmentation, and secondary effects (flying glass, debris flyout). Accurate wound simulation has two major benefits. First, it offers realistic training in treating wounds that are likely to occur, instead of random or scripted wounds. Second, it more accurately simulates the seriousness of a wound and what capabilities a Soldier would actually lose during a conflict. A minor wound may allow a Soldier to continue to fight, while a major wound may require immediate evacuation. The implication is that accurate wound simulation can have a major effect on the outcome of the training.

Using Validated Physics to Enhance Training

There are a few practical ways in which validated physics effects can be used to enhance game based training. First, these effects have the potential to increase the immersive experience of the training. Your brain holds many models of physical interactions that happen in the world around you. The more closely the simulation matches your mental model, the greater the immersive experience. Higher levels of immersion have demonstrated improved learning outcomes (Mantovani, 2003).

The use of validated physics to enhance existing game effects can provide this higher level of immersion by creating effects that more closely match a trainee's mental model of the world.

Second, validated physics effects provide feedback on the use of weapons in a simulation. Weapons employed in the simulation produce damage effects that correlate to real weapons. If the blast and fragmentation effects from the weapon are modeled from validated equations, then the response of humans, vehicles, and buildings will match the real weapon closely enough to reinforce correct employment of the weapon to the trainee. Conversely, validated models will accurately simulate errors in delivery accuracy resulting from incorrect use of a weapon.

Third, validated physics effects facilitate more accuracy in simulating consequences of actions performed in a training scenario. Physics models add realism by accounting for shielding, secondary effects, and accurate modeling of damage (Figure 4). This increased realism allows more accuracy in simulation of wound mitigation through proper cover and concealment, use of protective gear, and evasive action.

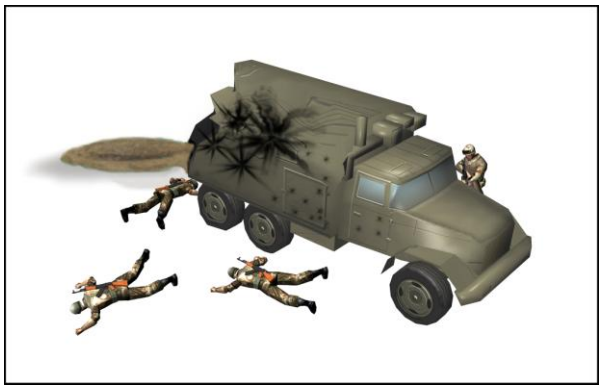

Figure 4 Validated physics effects make it possible to accurately simulate shielding, secondary effects, and damage that correlates to the type of munition and method of employment.

Learn. Train. Win.

The I/ITSEC 2008 theme of "Learn. Train. Win." describes the progression of learning new skills, applying those skills in training, and applying the training to win decisively. Validated physics effects support this progression by helping close the gap between the simulated effects a Soldier experiences in training and the real effects he experiences in a conflict. By supporting these effects from the outset of asymmetric warfare training, the Army is introducing a more direct correlation between the skills being trained and the eventual employment of those skills. The goal of this correlation is to develop a Soldier's intuitive decision making skills before he goes to war instead of making him wait until the actual conflict to experience and learn from these effects.

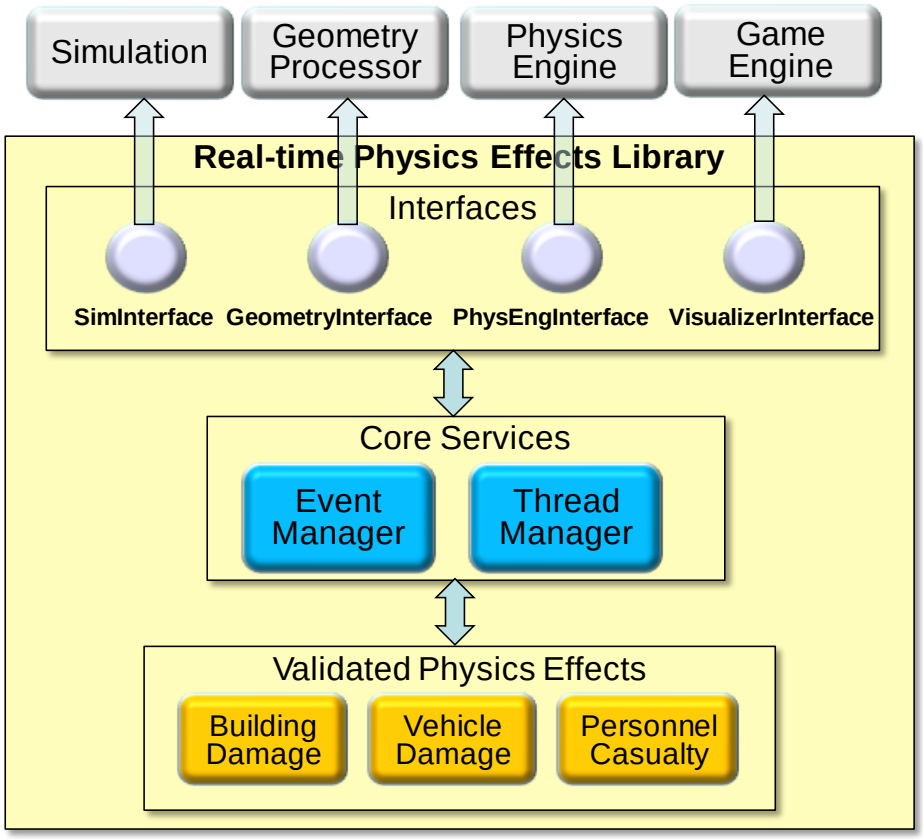

Figure 5 Overview of the RPEL Architecture

CURRENT RESEARCH

This section describes our implementation of a Real-time Physics Effects Library (RPEL). We have implemented this library for two specific use cases: an IED attack on a ground vehicle and on a building. We cover only the first use case in this paper. The RPEL strategy could be applied to model other effects in real-time.

RPEL Design and Architecture

We designed RPEL based on the following objectives:

1. Encapsulate the complexity of validated physics models
2. Create a tunable library that will scale from embedded systems to high performance computing clusters
3. Provide key components for modeling real-time physics effects
4. Provide this capability in a software library that can be integrated into a larger simulation.

RPEL achieves the first objective of encapsulation by providing a set of interfaces that hide the details of the validated physics models. The models contain assumptions on the input data format, units, and data structure. These assumptions vary across the models, sometimes in significant ways.

RPEL simplifies these assumptions by providing a consistency in the data format and units, and by creating the data structure required by each model. Application developers that integrate with this interface do not have to understand how each model works to use them. Instead, developers are presented with documented functions that provide access to model functionality.

RPEL achieves the second objective of tunability by providing access to variable fidelity models. Through the use of interfaces, models of varying fidelity can easily be swapped in and out. This flexibility allows us to use RPEL on various computing platforms, ranging from small and embedded to large and powerful.

RPEL provides key modeling components, the third objective, by providing access to three main components that are necessary for modeling real-time

physics effects. These components are the external interfaces, the core services component, and the validated physics effects models. These components are described in more detail in the following sections.

We achieve the fourth objective by designing RPEL in library form with a simple, consistent application programming interface (API). This API allows developers to integrate RPEL capabilities into their system without having to write additional code to integrate directly with individual models.

Interfaces

RPEL is isolated from specific tools through its external interfaces. These interfaces provide two primary functions. The first function is to provide a consistent entry point to the validated physics models. Without this interface, applications integrating with RPEL would be required to have knowledge of each underlying physics model. The second function of the interfaces is to allow RPEL to call external libraries for supporting functionality. The four interfaces are the simulation interface, the geometry interface, the physics engine interface, and the visualizer interface. These four interfaces are described in the following paragraphs.

RPEL receives simulation events through its simulation interface. Entity movement, entity state changes, firing events, and detonation events are examples of events handled through the simulation interface. A simulation interface might connect through DIS/HLA, or through a simulation specific protocol like OneSAF's Simulation Object Runtime Database (SORD).

In some cases, RPEL requires 3D geometry for targets. It gets this functionality from a geometry processor, through its geometry interface. Physics engines have internal geometric representations; therefore a physics engine could be used as a geometry processor. Other examples of geometry processors are the underlying engines of computer-aided design software.

RPEL leverages the power of 3rd party physics engines for some calculations. It accesses these through its physics interface. These engines model first principles physics. RPEL uses these capabilities to model higher order effects, like a munition detonation or wall rubbling.

One use of RPEL output is effects visualization. RPEL uses 3rd party visualizers through its visualizer interface. As with the other interfaces, the visualizer interface minimizes the impact of integration with multiple visualizers. We are currently using the OGRE open source rendering engine for visualization.

Core Services

Internally, the RPEL framework manages simulation events and module effects calculations. When an event such as a munition detonation is received, RPEL adds it to a queue until resources are available to process the event. If more processors or cores are available, RPEL will distribute events to additional processors or cores.

The advantage of providing these services within the effects library is that a simulation calling RPEL does not have to implement event management and threading to use the physics effects models. The core services reduce the effort required to integrate the library into an existing simulation.

Validated Physics Effects

The lowest layer in the RPEL architecture is the validated physics effects layer. This is where the actual effects of simulation events are calculated. The RPEL graphic (Figure 5) shows three modules that are abstractions of the validated physics models and the supporting code for these models. Developers can add new models by plugging them into this layer. Currently, RPEL has validated physics models for fragment flyout, blast effects, material fracture, and structural collapse.

Data Flow Example

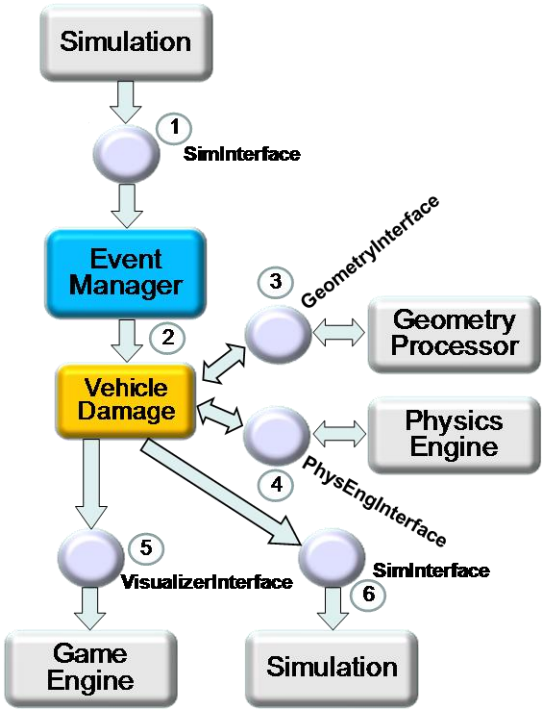

Figure 6 Engagement Data Flow through Interfaces

Figure 6 illustrates how data flows through RPEL for an engagement. The simulation initializes the engagement event. Consider for example an IED detonation near a ground vehicle. RPEL, through its SimInterface, receives notification of this event (1of figure). RPEL internally uses its event manager to handle this event. The event manager sends the event to the vehicle damage module (2 of figure).

The vehicle damage module then uses the geometry processor (through its interface, 3 of figure) to query the vehicle geometry. The vehicle damage module uses the physics engine to raytrace fragments (4 of figure). Once the module completes the analysis, it sends the results to the game engine and the simulation through their respective interfaces (5 and 6 of figure). The game engine renders the damage appropriately, and the simulation receives the vehicle damage state and factors it into the scenario.

Use Case: Vehicle Attacked by IED

Including a description of one of the use cases RPEL implements will aid in understanding how the library works.

Physical Modeling

The first use case modeled in RPEL was an IED attack on a ground vehicle. IEDs can either be placed manually in a game engine or they can be input as events. OneSAF is typically used to generate these events using the OneSAF mission editor to create the IED and vehicle.

Munition fragment effects are modeled based on arena test data. The arena data format defines the velocity, mass, and location of fragments for a particular munition. RPEL generates fragments based on the munition characteristics contained in this data.

Target models include geometric, component fragility, and system fragility characteristics. Model detail can be varied based on the desired accuracy and computing platform power.

Fragments may be flown out in either a uniform grid or as discrete fragments. Both representations are based on arena data characteristics. On a powerful computing platform, discrete individual fragments are modeled. On a less powerful platform, a low resolution grid is used to ensure real-time processing. This will possibly give less accurate results, but will still give damage characteristic of the engagement.

Upon impact, probability of damage is assessed based on component fragility models. Where high accuracy is not necessary, fragility models may be based on intuition and open sources. Classified sources are required when high accuracy is desired.

Also at impact, we must determine if the fragment penetrated, and if so, what percentage of mass penetrated. We used the THOR engineering model for this. This model uses constants based on empirical testing to determine the mass and velocity on exit, given the mass and velocity on impact.

Once all fragments and impacts have been flown out and calculated, a system damage state roll-up is done using the Failure Analysis Lethality Tree (FALT) model. The FALT model defines what kill states will occur based on what components were destroyed. For example, a punctured tire may result in a mobility kill. A destroyed gun would result in a firepower kill.

Visualization

Since RPEL handles higher fidelity damage calculations than pre-calculated models, more realistic and specific damage can be visualized than has historically been possible. There are different strategies that could be employed to visualize effects. For example, given a fragment penetration, the resulting hole could be shown with a texture, or the model could be physically manipulated to include the hole. Tradeoffs must be performed to determine which strategy is more appropriate for the given situation.

Manipulating a physical model is computationally expensive. Visually, a hole shown with a transparent texture is equivalent to an actual hole, so in many cases, texture manipulation is sufficient. In the case of large holes, like those formed by an explosively formed penetrator, the hole could affect subsequent engagements. We have experimented with both geometry and texture manipulation in RPEL and continue to work on approaches that increase realism without sacrificing real-time performance.

Figure 7 shows a RPEL visualization of an IED detonation and traces of the IED fragments as they hit the vehicle. The fragment traces are only shown for illustrative purposes. The vehicle geometry is modified where fragments hit various components. If a fragment has sufficient velocity to exit the vehicle, RPEL creates an exit hole as well.

Figure 7 The top graphic displays an IED detonating in RPEL and visualized in the OGRE rendering engine. The center shows the creation of fragment penetration holes in real-time. The bottom graphic shows the shotlines for each of the fragment penetration holes.

LESSONS LEARNED

We discovered early in the project that there is a large disparity between effects produced by physics engines and the visualization of results from those effects. In order to maintain real-time performance, physics engines use a very crude representation of objects in the environment. The game engine renderer is responsible for translating the effects on the crude representation into a detailed polygonal model. This technique does not support the type of high fidelity effects we are aiming for with RPEL. Our current research involves looking for ways to use higher fidelity representations of objects as input to physics models.

Over the last several years, there has been significant effort invested into the development of high fidelity physics models, but very little effort has been spent on real-time visualization of these effects. As our research into visualization techniques has progressed, we have discovered techniques that permit better correlation of the physics model results with the visualization results. Using a crawl, walk, run methodology, our initial focus is on creating holes in models to simulate fragment penetration. This allows us to understand the tradeoffs of modifying geometry versus using textures to simulate holes. Future research will involve more sophisticated damage to vehicle components and visual effects such as bending metal and glass breakage.

CONCLUSION

In this paper we discussed possible applications for real-time physics effects to support game based training simulations. The idea of using physics effects for game based training is not new. What has been missing from game based training is the validated simulation of effects. The physics effects used in games make assumptions that may be invalid for training. These assumptions are generally made to maintain real-time performance.

We contend in this paper that there are legitimate reasons to use validated physics models to ensure that the game based effects are accurate. Improvements in processing hardware has made it possible to implement these higher fidelity models without sacrificing real-time performance, reducing or eliminating the need for scripting. These improvements make it possible to teach Soldiers more advanced skills such as rapid decision making in the type of asymmetric conflicts that our troops are encountering more and more. Developing these skills early is essential to the "Learn. Train. Win." progression of training.

We also discussed the physics effects library we are developing. Through this research, we are able to develop use cases that demonstrate the utility of physics effects in game environments. We plan to integrate these effects with OneSAF to demonstrate potential uses for validated physics models in a training simulation.

The complexity of our world indicates that there are no limits on the physical interactions that will eventually be modeled in simulations. By necessity, developers will focus their efforts on improving the physical interactions that are most important to the objectives of the simulation.

Our focus in the future will be on accurate modeling of personnel casualties, and secondary effects such as debris flyout from a damaged building. We will continue to find ways to leverage technology improvements, such as using multicore processors to enable faster and more realistic effects.

REFERENCES

Ageia. (2007). Physics, Gameplay and the Physics Processing Unit. Retrieved from http://www.ageia.com/pdf/wp_physics_and_gaming.pdf December 23, 2007.

AMSAA, (2000), *The Compendium of Close Combat Tactical Trainer Algorithms, Data, Data Structures and Generic System Mappings*, Aberdeen Proving Ground, MD: AMSAA.

Bullet. (2007). Bullet Physics Library. Retrieved from http://www.bulletphysics.com/Bullet/ December 26, 2007.

Hanes, P., Murray, K. Gwyn, D.,Polak, H. (1988). An Overview and Status Report of MUVES (Modular Unix-Based Vulnerability Estimation Suite). Army Ballistic Research Lab, Aberdeen Proving Ground, MD.

Harman, W., York, A. Integrated Munitions Effects Assessment: A Weapons Effects and Collateral Effects Assessment Tool. NBC Report, 30-37. (2003, Spring/Summer).

Havok. (2008). Havok Destruction Engine. Retrieved from http://www.havok.com/images/Havok_Destruction_Brief_2008.pdf May 17, 2008.

Hecker, C. (1996). Physics, The Next Frontier. Game Developer.

Johnson, K. (2008). Deployable Sim Training. Training and Simulation Journal.

Mann, J., Fisher, Dr. D., York, Dr. A., Lowndes, Dr. E., Kraus, M. (2006). An Analysis of Engagement Algorithms for Real-time Weapons Effects. Journal of Defense Modeling and Simulation, Vol. 3, Issue 3, July 2006.

Mantovani, F., Castelnuovo, G. (2003). Sense of Presence in Virtual Training: Enhancing Skills Acquisition and Transfer of Knowledge through Learning Experience in Virtual Environments.

Miller, P. Combat Logistics Battalion 22 Employs Deployable Virtual Training Environment to Develop Real-world Marines. Retrieved May 17, 2008, from http://www.dvidshub.net/?script=news/news_show.php&id=12351.

Singer, Dr. M., Long, R., Stahl, J. Kusumoto, L. (2007). Formative Evaluation of a Massively Multi-Player Persistent (MMP) Environment for Asymmetric Warfare Exercises.

Wampler, R., Dyer, J., Livingston, S., Blackenbeckler, N., Centric, J., Dlubac M. (2006). Army Research Institute - Training Lessons Learned and Confirmed From Military Training Research.

Zajac, D., Bissonnette, B.., (2005). Carson, J. The First Army IED Training Methodology. Infantry.