The Application of COTS Physics Engines to CGF Vehicle Dynamics

Jon M. Williams, Eytan Pollak, Steven R. Schwalm
L3 communications
Orlando, Florida
jmwilliams@link.com, epollak@link.com, srschwalm@link.com

ABSTRACT

Computer Generated Force (CGF) systems continue to benefit from the ever increasing computational power offered by modern computing hardware. These computational resources are often used to provide higher entity counts as opposed to improving entity fidelity. This is particularly true of vehicle dynamics which, with some exceptions, continues to lag far behind the fidelity of manned simulators. With CGF-simulated ("constructive") entities being asked to participate at close range with manned ("virtual") simulators in scenarios such as urban operations and formation flight, the fidelity of CGF vehicle dynamics becomes important.

This paper discusses the use of a commercial-off-the-shelf physics engine to add high fidelity vehicle dynamics to CGF-simulated entities without impacting entity count. Specifically the NVIDIA PhysX physics engine was applied to both the Army OTBSAF and Air Force XCITE Computer Generated Force systems. Vehicle dynamics was modeled on basic vehicle and environmental parameters such as mass, inertia, engine power, torque, lift, drag, thrust, etc. Dynamics were implemented and evaluated for wheeled and tracked vehicles, life forms, rotary wing and fixed wing aircraft. Physics augmentation of vehicle dynamics was demonstrated with a single physics engine simultaneously supporting multiple, dissimilar, CGF systems providing smooth, physically correct, and consistent dynamics for all vehicles in all of the CGF systems.

ABOUT THE AUTHORS

Jon Williams is employed by L3 Communications as a Principal Systems Engineer. He has 25 years of experience in simulation, including 10 years in rotorcraft engineering development simulation and 15 years of modeling experience with Computer Generated Force systems.

Steve Schwalm is employed by L3 Communications as a Senior Systems Architect. He has 26 years of experience in training simulation, including Image Generator development, 10 years in Command Staff Trainer development and 20 years experience with Distributed Simulation systems.

Eytan Pollak received his B.Sc. and M.Sc. from Technion-I.I.T (Haifa, Israel) and his Ph.D. from Purdue University, West Lafayette, IN. He has thirty years experience in managing research and development programs, he holds several patents and has published papers in Control Systems, Robotics, Distributed Flight and Ground Simulations/Simulators, Embedded Systems, and Cyber Physical Systems. He is a member of the Training Transformation Collaboration Advisory Group (TTCAG) for the warfighter program. He is currently a research professor at the UCF School of Electrical Engineering and Computer Science, and he is also the Director of Strategic Technologies at L-3 Communications Link Simulation & Training..

The Application of COTS Physics Engines to CGF Vehicle Dynamics

Jon M. Williams, Eytan Pollak, Steven R. Schwalm
L3 communications
Orlando, Florida
jmwilliams@link.com, epollak@link.com, srschwalm@link.com

Introduction

Simulation interoperability is not just about the communication of bits back and forth across a network. The simulation models must also interoperate with each other to provide a fair fight between entities participating. This is extremely important in a virtual training situation where the trainee is provided a visual representation of the simulated entity to interact with.

In this perspective, the CGF entity is no longer a constructive entity but rather a virtual entity that just happens to be generated within a constructive type system. CGF systems must then generate entities with enough fidelity to mimic the capabilities and dynamics of a manned simulator.

With the increases in computational resources at a lower cost, the capability to provide this higher fidelity is available. The technology in multi-core CPUs and GPUs provide this leap in an affordable package.

L-3 began to look at how we could apply these computational capabilities to CGF systems. Our customers have spent large sums of money developing validated behavior models, sensor performance and weapons deployment characteristics. One area we found that they could use enhancement was in the physical movement of entities. So instead of building a new CGF we looked at methods to improve existing CGF systems.

Integration with the SAF System

L-3 selected the AVCATT version of OTBSAF to integrate with this enhanced physical movement capability. OTBSAF is the precursor to ONESAF and provides a good basis for verifying this approach. Other CGF systems like OneSAF or XCITE can be enhanced in a similar way. Figure 1 shows the connection between OTBSAF and what is now called the Physics Based Environment Generator (PBEG)[5].

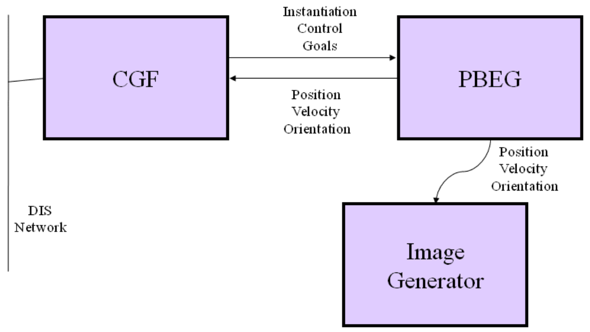

Figure 1. System Configuration

As can be seen from the diagram, PBEG is loosely coupled to the CGF system. This was done for several reasons. The first reason of which was the use of the NVIDIA PhysX SDK[1], discussed below, which only ran in Microsoft Windows and OTBSAF which ran in Linux. The loosely coupled capability also allows us to choose among multiple CGF systems.

To obtain the loose coupling, the SAF/CGF system sends routes to the PBEG system from which goal points can be determined. The goal points are deterministically calculated each frame. The controllers discussed below then try to obtain the goal point based on the individual entities performance. To minimize the communication between the systems, OTBSAF was updated to plan an intended path that could be passed once versus incrementally. If the CGFs goal changes, then the system just sends the new goal and the PBEG adapts.

PhysX Terrain

The PhysX SDK provides a means for defining objects within the environment. This capability was utilized to define an urban terrain database. A process was written to convert from L-3's common database format to the internal format of the PhysX SDK. This allowed the PhysX database to have a one to one polygon correlation with the visual rendering system. Figure 2 shows an example of the correlated PhysX terrain. The buildings were defined as static objects within the

PhysX environment. The same source was then used to generate a correlated CTDB for OTBSAF.

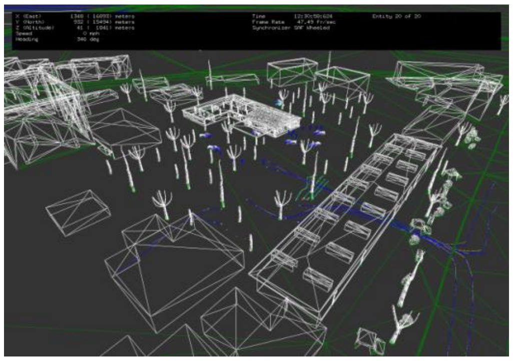

Figure 2. PhysX Terrain Database Example

Additional work has been done to provide a dynamic database capability. Destructible buildings were modeled within the PBEG environment and damaged based on detonations provided from the external simulations. The initial models were based on the visual polygon representation. The polygons were fractured into smaller pieces which were then impacted by the detonation. This produced an adequate effect but some of the polygons would eject themselves into space thus making the detonations unrealistic. By fixing the polygons to a simple frame within the building, adding physical properties to and between the polygons most of these anomalies were corrected.

General Entity Modeling

The SDK also allows the modeling of dynamic objects within the PhysX environment as per Figure 3. The system provides a set of primitive objects or allows the user to define more complex objects. Each object has physical characteristics such as mass, inertia and coefficients of friction. Additionally, joints or connections can be defined between objects to compose more complex objects with each joint having its own characteristics such as damping and spring coefficients.

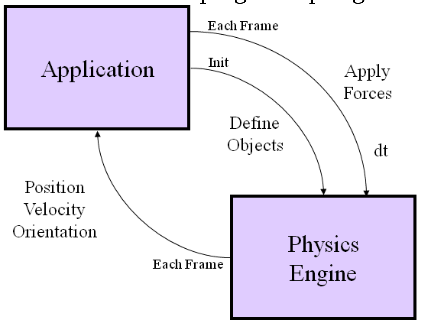

Figure 3. PhysX Control Loop

Once a vehicle is defined within the PhysX environment, forces can be applied to move it through the environment. The positions and attitudes of the vehicles/objects are computed by the PhysX system. Different forces or torques are applied to each object from the controlling application prior to each iteration of the system.

Vehicle Performance Modeling

Within many CGF systems today the entities are moved through the environment as point masses by the basic equations of Force equals Mass times Acceleration. The vehicle dynamics often take into account other characteristics but it still comes down to moving around the point mass. In PBEG the vehicle dynamics are modeled as a rigid body based on a series of components that make up the vehicle. Each component then has variable parameters such as mass, inertia, engine power, torque, lift, drag, and thrust. Forces are applied to these components individually and the physics environment accumulates the forces to provide the proper movement. Later areas of this paper discuss the different vehicle classes we investigated.

The performance data for the vehicles was derived from known vehicle data. For ground vehicles we used the existing CGF data and augmented it with data from the Internet. For helicopters and fixed wing entities we utilized the vehicle dash 10 data. The model is data driven and thus can be easily adjusted for specific vehicles.

Modeling the Driver

To successfully separate the physical models from the high level behaviors, the low level "driving" functions also needed to be moved into the PBEG system. This provided an interface that allowed OTBSAF to pass routes from which goal points could be calculated by PBEG for the physical models to try to achieve. The vehicle moves to obtain the goal based on it's defined physical capabilities as well as the environment conditions. The driver function also performs obstacle avoidance to avoid other objects within the simulated environment.

Since the driver behaviors run in the PBEG environment, they execute at a much faster rate than was previously provided within OTBSAF. Where before, the CGF driver behaviors may have run a 2 to 3 times a second, they now run at between 30Hz and 60Hz. This higher update rate provides better movement control and avoidance of objects within the environment.

Control Loop

The goal of the system was to loosely couple the parent CGF with the PBEG environment. To minimize the communication, the PBEG system needed to implement a control system for each entity. This was complicated by the need for the PhysX environment to be supplied with forces that should be applied to the entity rather than a simple position and orientation. To solve this, PBEG implements a control loop system for each entity. Figure 4 shows how the control fits into the system. The Control takes the goal that has been provided by the CGF system and determines where the entity should be at any given time.

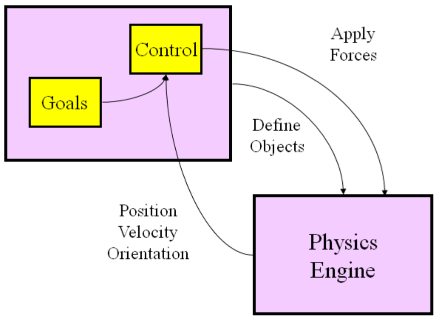

Figure 4 PBEG Control Mechanism

The control then adjusts the entity forces and torques to obtain that goal. The forces and torques that can be generated are limited by the specific capabilities of each individual vehicle. If the control mechanism determines it can't achieve the specified goal, it notifies the CGF system to take corrective action.

Groups of entities provide a different control problem. As the number of entities groups increases so does the CGF operator workload as he tries to control them. To address this issue, L-3 in conjunction with UCF implemented a cooperative control mechanism to allow the entities to work as a unit but still interact as individuals.[2,3,4]

Wheeled Entity Control

We began the ground vehicle by first modeling the wheels and by applying torques to the wheels to pull the vehicle through the environment. Through the use of the physics environment the traction provided is based on the friction between the wheels and the ground.

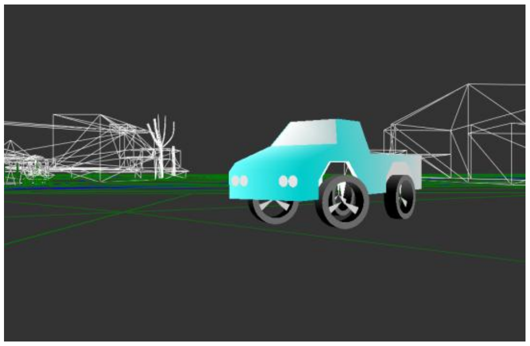

Figure 5. Wheeled Vehicle

This approach allows us to model different terrain surfaces and provide the appropriate affects. Figure 5 provides a graphical representation of the vehicle model within the physics engine.
To make the entity turn, the steering of the wheels had to be modeled. With this method the entity is pulled through the turns and thus if the entity turns too fast the vehicle can slide through the turn.

To increase the fidelity of the ground vehicle, the suspension of the vehicle was modeled. Figure 6 shows the basic configuration which includes a spring and shock absorber. With the addition of the suspension, the wheeled vehicle hull also dynamically moves. These dynamics provide the pitch of the vehicle when it accelerates and brakes while rocking when it goes around turns.

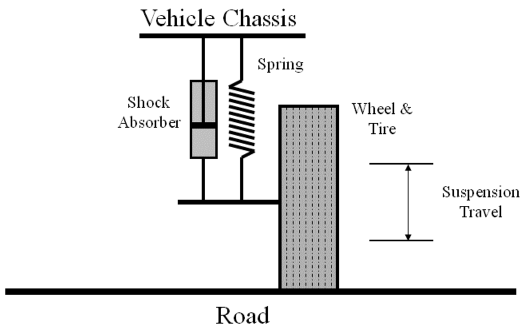

Figure 6. Vehicle Suspension

Tracked Entity Control

Tracked entities follow the basic principles of the wheeled vehicles. The number of wheels was increased to support all of the road wheels of the vehicles. The control was changed to support the different mode types of steering that are possible with a tracked vehicle.

Human Entity Control

The human entities required a large number of instances provide crowds. To enable this, the entities were not modeled with individual limbs. The first model built the human as if he were riding a unicycle with a single wheel which was then controlled. As the system was scaled to a larger number of entities, the wheel calculation within the PhysX system required too many resources. The human entity was then changed to be an upright capsule representation that was moved through the environment. This increased the performance since the capsule is a PhysX primitive that is hardware accelerated.

Rotary Wing Control

Next we moved onto rotary wing control as shown in Figure 7. This modeling added the complexity of flight. The rotary wing control models the force from the main rotor to lift the aircraft. To provide an accurate model that would provide the proper visual effects the control also had to model the torque from the main rotor spinning plus the tail rotor which is used to counter act this torque. Closed loop control for each of these forces is provided to generate the movement models. As discussed above, the model uses the published aircraft performance data to model the capabilities of the helicopter.

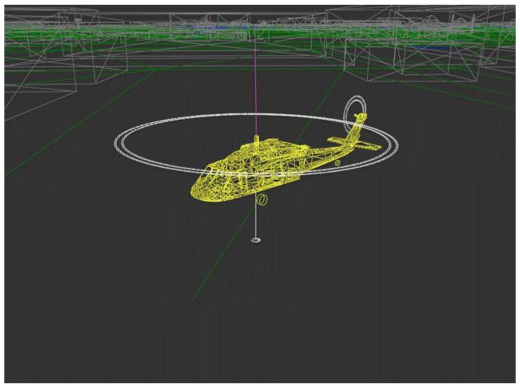

Figure 7. Helicopter Model

Integration of XCITE

With the modeling of fixed wing entities, OTBSAF only had minimal fixed wing behaviors so we integrated XCITE with the PBEG system. Figure 8 show the system configuration with both OTBSAF and XCITE utilizing the same PBEG system.

With the integration of XCITE, the instantiation, control, and behaviors remained within XCITE but the physical movement of the entities was transferred to PBEG control. PBEG then passed the entity locations and orientations back to XCITE just as was done with OTBSAF.

This provided a common environment for all of the entities to interact within. With the low level driver functions placed within PBEG the entities avoided each other without any interactions from the controlling CGF systems.

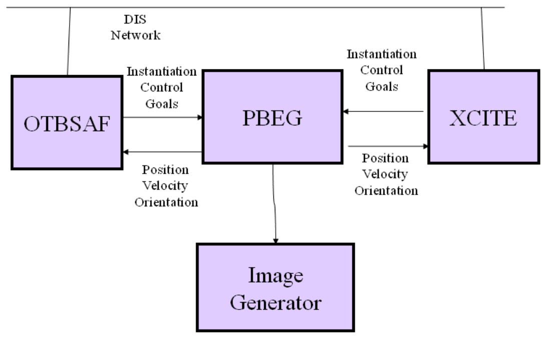

Figure 8. Hybrid CGFs with Physics Environment

Fixed Wing Control

With a fixed wing aircraft the four basic forces are weight (mass x gravity), lift, forward thrust and drag. These forces are each controlled based on the aircraft performance specifications except for weight which is computed by the PhysX engine. If aircraft just flew straight and level, these would be fairly easy calculations. The control issue is that the affect of the forces vary based on the aircraft angle of attack. This adds induced and parasitic drag to the flight calculations increasing the complexity. These impacts were added by coupling the control loops into a more complex control system.

Additional work was performed on the control of fixed wing formations.[4] This allows the vehicles to work as a unit and provide for the high fidelity formation flight that pilots can perform. These controls utilize robot control theory that was developed for the control of UAV systems.

The takeoff and landing of the aircraft also needed to be provided. This required providing control for the aircraft on the ground and during take-off by taking into account the runway length, rotation speed, and gross weight in accordance with the maximum thrust and take of acceleration.

Performance Improvements

The specific CPU performance impacts within each CGF system have been hard to measure. The main performance increase comes from the extraction of the entity movement out of the CGF into the PBEG system. This decoupling of the entity movement returns the processing that is normally done back to the CGF for execution of the behaviors. The issue is, within the OTBSAF the modeling of the entity movement was minimal to begin with and thus it may only return 5% to 10% back to the CGF system after the transfer of the positions is taken into account. We have not yet done any analysis on the XCITE system but it could have a greater impact due the dynamics requirements of the aircraft it models.

The real performance impact of this method comes from the fidelity of the movement that can be provided to the training audience. The movement dynamics provide the subtle nuisances that immerse the trainee into a believable training exercise. The ground vehicles provided no longer ice skate across the terrain but dynamically interact with it depending on the speed of the entity and the roughness of the terrain.

Un-intended Benefits

During testing, scenarios were built for the OTBSAF with PBEG based on scenarios that were used for AVCATT. These scenarios included some dense urban areas with the entities negotiating down streets and between buildings. Previously on AVCATT the definition of the scenario in this type environment was a very tedious task that took around a week to test and get a usable scenario. This was mainly due to the OTBSAF entities not being able to avoid each other efficiently and thus having to correct and re-correct which eventually lead to a traffic jam at some intersection.

With the inclusion of the PBEG system this scenario setup time was cut. By moving the collision avoidance to the PBEG system and increasing its iteration rate the entities could make decisions quicker based on current data.

Conclusion

By applying commercial game technology to existing training simulation components L-3 Communications increased the fidelity of existing CGF systems while maintaining the entity count supported by the parent CGF system. Through the use of control techniques we were able to adapt the commercial technology and enhance our training capabilities. Separate control instances are provided for different vehicle classes to support the unique aspects of each type of vehicle. These data driven control classes are populated with vehicle data derived from real world measured performance specifications. Additionally, we showed that multiple existing CGF systems can interact in this common augmented physical environment with this increased fidelity.

ACKNOWLEDGEMENTS

We would like to thank the L-3 Technologies team for working diligently to implement the PBEG infrustracture and modeling algorithms and integrating it with the CGF systems. Additionally, thanks to the NVIDIA development team for providing the toolkit and hardware acceleration to make the modeling possible in real time.

REFERENCES

1. NVIDIA Corporation, (2008). *NVIDIA Physx SDK 2.8*
2. J.Chunyu, Z.Qu, E.Pollak, M.Falash, "A New Reactive Target-tracking Control with Obstacle Avoidance in a Dynamic Environment", 2009 American Control Conference, St. Louis, Missouri, USA.
3. J.Chunyu, Z.Qu, E.Pollak, M.Falash, "A New Reactive Target-tracking Control with Obstacle Avoidance in a Dynamic Environment", 2009 American Control Conference, St. Louis, Missouri, USA.
4. H.Yuan, V.Gottesman, M.Falash, Z.Qu, E.Pollak, and J.Chunyu, ``Cooperative Formation Flying in Autonomous Unmanned Air Systems with Application to Training," the 7th International Conference on Cooperative Control and Optimization, University of Florida, Gainesville, Florida, January 31 to February 2, 2007.

5. Falash; Mark ; et al. "Distributed Physics Based Training System and Methods" US Patent Application 20090099824.