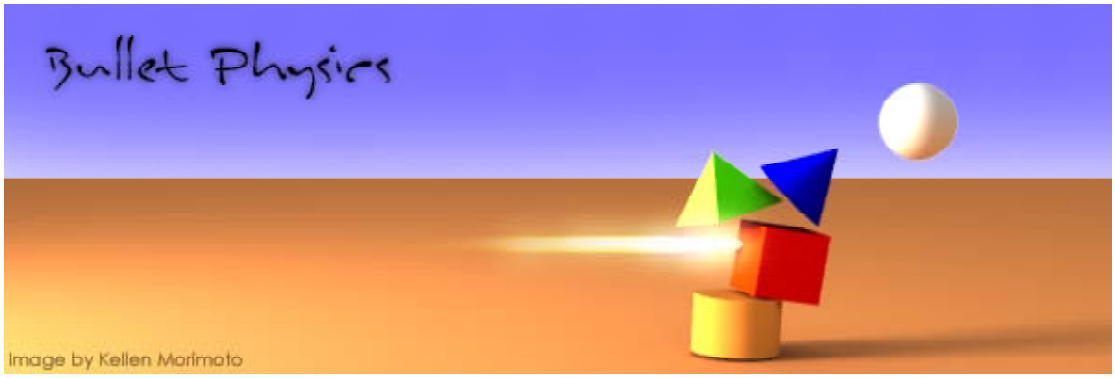

Bullet Continuous Collision Detection and Physics User Manual

Bullet version: 1.1, **Thursday, 23 February 2006**

Introduction

Bullet Physics is an open source collision detection and physics library, related tools, demos, applications and a forum. It is created as a skunk project by Erwin Coumans, ex-Havok employee, free for commercial use. Primary location of the software and the forum is http://www.continuousphysics.com/Bullet/phpBB2/index.php.

Bullet is mainly a Collision Detection library. It features both discrete and continuous collision detection. Discrete collision detection provides closest points, normal and shortest distance, as well as persistent multi-contact management.

Continuous Collision Detection calculates the time of impact between moving Objects. Several implementations are provided in Bullet. They differ on the allowed motion and on the type of algorithm. Solutions for both pure linear motion (sweep) as well as combined linear and angular motion are available. An iterative ccd solution (Coumans [1]) as well as a closed form algebraic solution can be used (Redon [2]).

On top of this, Bullet provides its own Dynamics library, and also bindings to ODE.

Furthermore Bullet Physics is being used in Blender 2.4x Game Engine. See www.blender.org or read the Bullet_Blender Manual. And last but not least, Bullet physics provides a Physics abstraction layer / interface, that will be integrated with Collada Physics (http://www.collada.org).

Main Collision Detection Features

- Generic and extendible Collision Shape support: Sphere, Box, Cylinder, Capsule, Cone, Convex Object, Minkowski Sum, Convex Compounds, Static Triangle Meshes using Bounding Volume Hierarchies.
- Discrete: Closest Point/Normal/Distance, Penetration depth
- Continuous: time of impact to prevent tunnelling
- Persistent Contact Manifold to manage multiple contact points
- Island (De-)Activation (Sleeping)
- Broadphase and Midphase culling optimizations
- Generic Physics Interface. Bindings with Ode, Havok, Novodex, Ipion, Mathengine, Dynamo etc. have been successfully done using this Interface.
- Raycast support for all shapes

Main Physics Features

- Impulse based constraint solver and PGS solver based on ODE quickstep
- Vehicle support
- Point to point constraint

Future Plans

- Increased Performance, bugfixes, improved continuous collision detection
- More constraints (6DOF), limits, motors
- Parallel Dispatch of Collision Detection and Constraint Islands
- Character controller (allows moving a player proxy with sliding movement)

Quickstart

There are several ways to use Bullet:

Collision detection:

- You can just use for performing collision detection. In that case, you only need the files in the Bullet/Bullet folder.
- There is no dependency on the BulletDynamics project or the high level physics interface.
- Recommended is looking at the files in the Bullet/Demos folder.

Collision and Physics Library

- If you want both Collision Detection and Physics, its best to use the CcdPhysicsEnvironment implementation.
- Requires both Bullet/Bullet and Bullet/BulletDynamics projects
- See Bullet/Extras/PhysicsInterface
- Also see Bullet/Demos/CcdPhysicsDemo

Bullet ODE binding

- Provides GJK alternative to Opcode
- Optional compatibility with Ode and Opcode colliders
- See Bullet/Extras/ode/ode/test/test_BulletGjk.cpp
- See Bullet/Extras/BulletOde

Abstraction layer, physics comparison, research

- See Bullet/Extras/PhysicsInterface
- Comparison with Solid 3.5 See Bullet/Extras/ExtraSolid3.5
- http://www.continuousphysics.com/Bullet/phpBB2/index.php.

Blender game engine physics

- Please visit www.blender.org
- See Bullet Blender section on the Physics forum: http://www.continuousphysics.com/Bullet/phpBB2/viewforum.php?f=11

Description of Demos

CollisionInterfaceDemo

The Collision Interface Demo shows the use of CollisionWorld, CollisionObject as main interface. It can be used with external Physics engine or use Bullet Dynamics / ODE. See Bullet/Extras/ode/VC6/Samples/ MakeAllTests.sln

CcdPhysicsDemo

The CcdPhysicsDemo is the most high level demo. It uses Collision Detection, Dynamics and the high level physics interface.
The demo in a nutshell:

Create a CcdPhysicsEnvironment object. For each Rigidbody, create a CcdPhysicsController , and add this to the CcdPhysicsEnvironment. Then every frame, you need to update the CcdPhysicsEnvironment by calling the Proceed method, passing the elapsed time.

To construct a CcdPhysicsController, you need to provide:

- a CollisionShape, like a Box, Sphere, Cone, Convex Hull or Triangle Mesh
- Mass and Material properties like friction and restitution
- a link to your Graphics object by deriving from MotionState

Bullet will update the position of every Graphics Objects, by simulating the Rigidbodies, performing Collision Detection, and synchronizing the transform by calling your provided MotionState. There is performance functionality like auto deactivation for non moving objects, and debugging feedback that can be linked to your renderer. See Debugging section for this.

ConvexHullDistance

Good starting point to directly use GJK and closest point calculation.

Raytracer

Implements a basic raytracer that uses the continuous collision detection query.

GjkConvexCastDemo, ContinuousConvexCollisionDemo

Basic time of impact demo, for pure linear motion (GjkConvexCastDemo). ContinuousConvexCollisionDemo show Ccd with combined linear and angular motion.

ConcaveDemo

Shows the usage of static triangle mesh. An Aabb tree is automatically build and used to speed up the collision query.

CollisionDemo, SimplexDemo

CollisionDemo shows some degenerate case for Bullet GJK implementation. SimplexDemo demonstrates the internal workings of GJK subdistance algorithm.

Download Bullet SDK Collision Detection and Physics

Visit http://bullet.sourceforge.net for demos, downloads, and a generic physics forum.

Latest Bullet Version from CVS:

```
cvs -z3 -d:pserver:anonymous@cvs.sourceforge.net:/cvsroot/bullet co -
P bullet
```

Bullet Doxygen Documentation On-line

http://www.continuousphysics.com/Bullet/BulletFull/index.html

Please note that the doxygen docs might sometimes be out-of-date.

It is best to generate the documentation from the sources: Download doxygen, and run 'doxygen' in the Bullet folder. There is a Doxyfile provided, which recursively traverses all sources. For best result also install graphviz for the graphical class diagram and graphical collaboration diagram.