Open Dynamics Engine (ODE)

opende.sourceforge.net

Tyler Streeter 2003

Topics to be Covered

- What is ODE?
- How ODE works
- Demos
- Getting Started with ODE
- Applications to Games
- More Demos
- Helpful Resources

What is ODE?

- Real-time rigid body dynamics simulation
- Integrated collision detection (OPCODE)
- Free, open source software library
- Runs on Unix, Windows, OS X
- Cool

How ODE Works

- Each object usually has a:
  - Dynamics component (body) - mass, forces
  - Collision component (geom) - physical shape
  - Visual component - appearance
- Use joints to connect objects
  - Hinge, ball-and-socket, slider, universal, etc.
- Use forces/torques, not objects' positions or orientations
- Collision detection creates "contact" joints

Demos

- Chains
- Stacks of blocks with flying spheres
- More later…

Getting Started with ODE

- Build ODE, OPCODE libs
- Link these libs into your application
- #include <ode/ode.h> somewhere
- Start coding…

Note: Code is in C++ with a C-style API

```
dBodySetPosition(dBodyID, x, y, z);
```

Instead of:

```
Body.SetPosition(x,y,z);
```

Syntax

- dWorld - a dynamics world.
- dSpace - a collision space.
- dBody - a rigid body.
- dGeom - geometry (for collision).
- dJoint - a joint
- dJointGroup - a group of joints.
- dReal, dVector3, dVector4, dMatrix3, dMatrix4, dQuaternion – commonly used data types

Typical ODE Application Sequence 1/6

1. Create a dynamics world

```
dWorldID dWorldCreate();
dWorldSetGravity(dWorldID, dReal x, dReal y, dReal z);
// adjust global ERP and CFM parameters – see documentation
```

2. Create a collision world

```
dSpaceID dSimpleSpaceCreate(dSpaceID space);
```

Typical ODE Application Sequence 2/6

3. Create a joint group for the contact joints

```
dJointGroupID dJointGroup(int max_size);
// max_size is deprecated and should be set to 0
```

4. Create bodies in the dynamics world

```
dBodyID dBodyCreate(dWorldID);
```

5. Set the position, mass of bodies

```
void dBodySetPosition(dBodyID, dReal x, dReal y, dReal z);
void dBodySetMass(dBodyID, const dMass *mass);
```

Typical ODE Application Sequence 3/6

6. Create joints in the dynamics world

```
dJointID dJointCreateHinge(dWorldID, dJointGroupID);
// dJointGroupID is normally 0.  This is used to insert contact
// joints (for collision detection) directly into a global joint group.
```

7. Create collision geoms

```
dGeomID dCreateSphere(dSpaceID space, dReal radius);
void dGeomSetBody(dGeomID, dBodyID); // join geom with body
void dSpaceAdd(dSpaceID, dGeomID); // put geom into a space
// Hierarchies of spaces…
```

Typical ODE Application Sequence 4/6

8. Attach joints to bodies

```
void dJointAttach(dJointID, dBodyID body1, dBodyID, body1);
```

9. Set joint parameters

```
void dJointSetHingeAnchor (dJointID, dReal x, dReal y, dReal z);
void dJointSetHingeAxis (dJointID, dReal x, dReal y, dReal z);
```

Typical ODE Application Sequence 5/6

- 10. Loop
  - Apply forces to bodies when necessary

    void dBodyAddForce(dBodyID, x, y, z);
  - Check for collisions

    This creates contact joints where collisions occur and adds these joints to the contact joint group.
  - Take a simulation step

    void dWorldStep (dWorldID, dReal stepsize); // more accurate

    Or, a faster, iterative way…

    dWorldStepFast1(dWorldID, dReal stepsize, int maxiterations);

Typical ODE Application Sequence 6/6

- 10. Loop (continued)
  - Remove all joints from the contact group
  - Redraw scene bodies' positions/orientations
- 11. Destroy the dynamics and collision worlds

```
void dSpaceDestroy(dSpaceID);
void dWorldDestroy(dWorldID);
void dJointGroupDestroy(dJointGroupID);
void dCloseODE(); // deallocate stuff that doesn't get
                  // destroyed elsewhere
```

Note About TriMeshes

- Triangular meshes have mixed reviews
- Right now, no support for trimesh/trimesh collision detection
- …But this situation can be avoided

Applications to Games

- Basic collision detection – invisible sphere around player
- Breakable joints
- Explosions
- Sports
- Realistic Machinery – catapults
- Vehicles – boats, cars, planes, etc.
- People

More Demos

- A simple person
- Simple person on a ledge
- Random neural net person
- Energetic random neural net person
- Person caught between ledges
- Evolved behaviors
- Links to demos on the ODE website

Helpful Resources

- Zip file of sample code, these slides, typical simulation sequence doc

  www.vrac.iastate.edu/~streeter – bottom of page

- opende.sourceforge.net

  Great documentation, Wiki/CVS links, mailing list archives