Simulating Hand Interaction in a Virtual Environment with Open Dynamics Engine and CyberGlove

By

Nick Nunley

A senior thesis submitted in partial fulfillment of
the requirements for the degree of

Bachelor of Science
With Departmental Honors

Computer Science & Engineering

University of Washington

June 2009

Presentation of work given on _____June 12, 2009_____

Thesis and presentation approved by ___Miro Enev_____

Date _June 16th, 2009_____

Simulating Hand Interaction in a Virtual Environment with Open Dynamics Engine and CyberGlove

Nick Nunley

Honors Thesis

Department of Computer Science and Engineering

University of Washington

Abstract

The CyberGlove and Open Dynamics Engine (ODE) are two technologies that can be used to simulate hand interaction in a virtual environment. ODE is an open source physics simulation framework and CyberGlove is a data glove that provides precise finger and palm orientation measurements. The combination of these technologies would produce a tool that could be used to collect measurements in the virtual environment to obtain a better understanding of the techniques of hand control during object interaction. This paper discusses the best approach to implement the integration of a virtual hand into a simulated rigid body physics environment with ODE and CyberGlove. The steps and explanation of implementation are explained, and then the results are provided with accompanying discussion on the uses, limitations, and possible next steps.

1. Introduction

Open Dynamics Engine (ODE) and the CyberGlove are both instruments used within the general field of digital data collection and virtual simulation for realistic physical situations. The CyberGlove is a human-wearable data glove that is capable of obtaining precise finger joint angle measurements as well associated data such as palm and wrist flexion. Data gathered with the CyberGlove is often exported for use in related software applications. ODE is a physics simulation library that provides a framework for emulating real-life physical interactions.

The goal of this project was to integrate the real-time data collection capabilities of the CyberGlove into a virtual environment simulated with ODE. Objects within this simulated environment would be manipulable through a virtual hand object controlled via the human-operated CyberGlove. This environment would then be a suitable place to conduct experiments and collect data on hand interactions. For example, this could be used to observe the physical forces exerted when an object is picked up.

The motivation for this project is that scientists currently do not have a good understanding of the strategies and techniques of the brain during object manipulation. This tool would provide a sandbox to enable a dynamic interaction environment to collect precise contact and collision force data to recognize important events and actions that underlie dexterous manipulation.

This paper explores the approaches to best connect the two technologies together in order to achieve the desired functionality and in doing so assesses the viability of using this tool, as well as discussing potential next steps to improve upon this project.

2. Background

2.1 ODE

ODE is "a free, industrial quality library for simulating articulated rigid body dynamics [1]." ODE was designed to provide an interactive, stable, real-time environment to simulate rigid body object interactions. It includes support for primitive geometric shapes as well as more complicated triangle meshes, and has a built-in collision detection engine. The physics simulation library is written in C++ but the provided interface to the code is in C.

Within an ODE environment objects are usually represented internally as a rigid body attached to a geometric object. A rigid body contains object state such as position, velocity, and mass, while the geometry provides information on its shape to interact with other objects through the collision detection engine. Objects can be connected through joints to affect and constrain interactions between them.

Using ODE in a program typically involves:

- Creating an initial world configuration and populating it with objects and joints
- Entering into a simulation loop where:
    - State-modifying input can be received from external sources
    - Collision detection between objects and the environment is performed
    - The simulation state is advanced one time step based on collision data, object and environment state, etc.
    - The new environment and object positions can be extracted and displayed visually

2.2 CyberGlove

The Immersion CyberGlove is a data glove able to capture measurements from joint angle sensors. The data is available to arbitrary software applications through a C++ device connection API provided in the VirtualHand SDK. With this driver and API it is possible to read angle measurements directly from the CyberGlove from within a program. Figure 1 shows the 22-sensor CyberGlove model similar to the one used in this project.

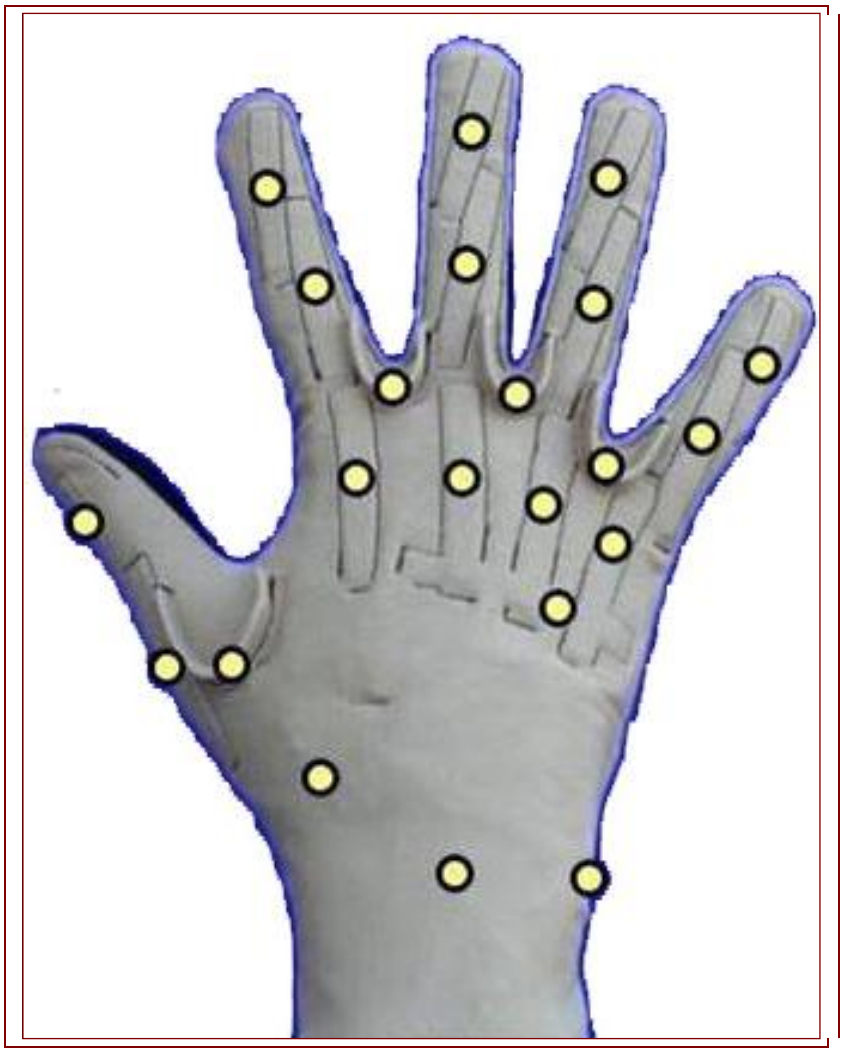

Figure 1. CyberGlove with sensor locations.

3. Method

This section describes some of the approaches to best integrate the CyberGlove with an ODE simulated virtual environment in order to obtain the desired functionality. Getting the CyberGlove working with ODE is not as simple of a process as it may seem. There were many unique requirements to consider. For example, because the state of the virtual hand should only reflect the data from the glove, and not react in interactions with other objects, the hand and finger positions are constantly corrected.

Environment setup

The environment setup followed the typical formula for initializing an ODE simulated program. The virtual world is initialized with standard parameters and the hand and other objects are placed in the environment. Object dimensions and

location can be specified with a text file. Figure 2 shows the syntax for the lines of the file.

```
sphere xPos yPos zPos radius
box xPos yPos zPos length width height
cylinder xpos ypos zpos length radius
```

Figure 2.

Hand Representation

The hand is modeled using basic geometric shapes - the palm is represented as a flat block and finger segments are represented as capped cylinders.

The fingers and thumb are connected to the palm through joints. The mid-finger joints are implemented with hinge joints (see Figure 4). The joint connecting the lower finger segment to the palm is a universal joint (Figure 5). Universal joints are like a ball and socket joints in that they allow for abduction ("sideways") movements, but restricts the additional degree of rotational freedom.

The finger- and palm-specific data are stored in corresponding data structures. Figure 3 shows the structure used to contain all the data needed to represent the fingers and thumb. Pointers to the contained body and geometry objects are present, in addition to pointers to the joints used to connect the finger segments and palm together. The most recently read CyberGlove angle measurements are stored here as well. Finally, a structure to store object interaction data is contained here. The palm data is represented with a similar structure although it lacks joint information and contains position and orientation information.

```
struct Finger {
      dBodyID bodyUpper, bodyLower;
      dGeomID geomUpper, geomLower;
      dJointID hingeUpper, universalLower;
      dReal angleUpper, angleLower1, angleLower2;
      ContactFeedback upperFeedbacks, lowerFeedbacks;
};
```

Figure 3.

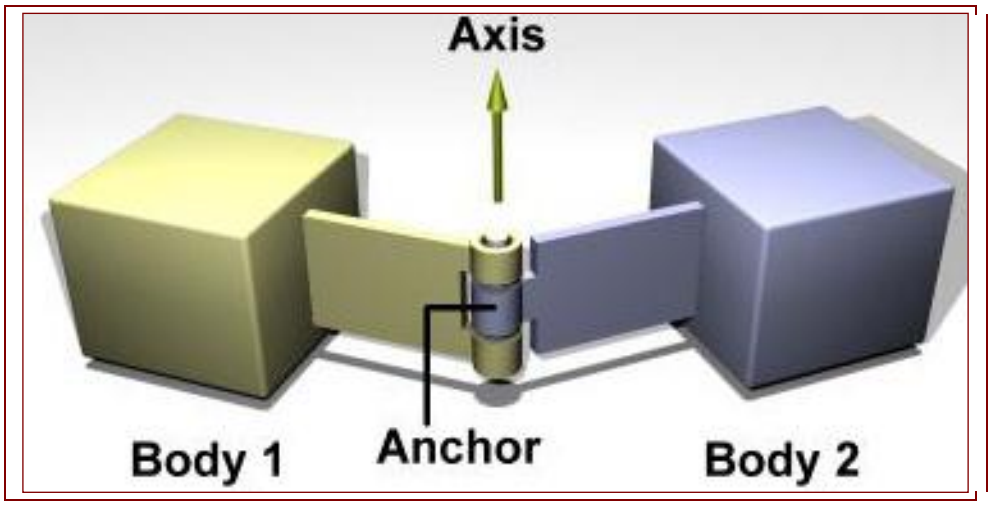

Figure 4.

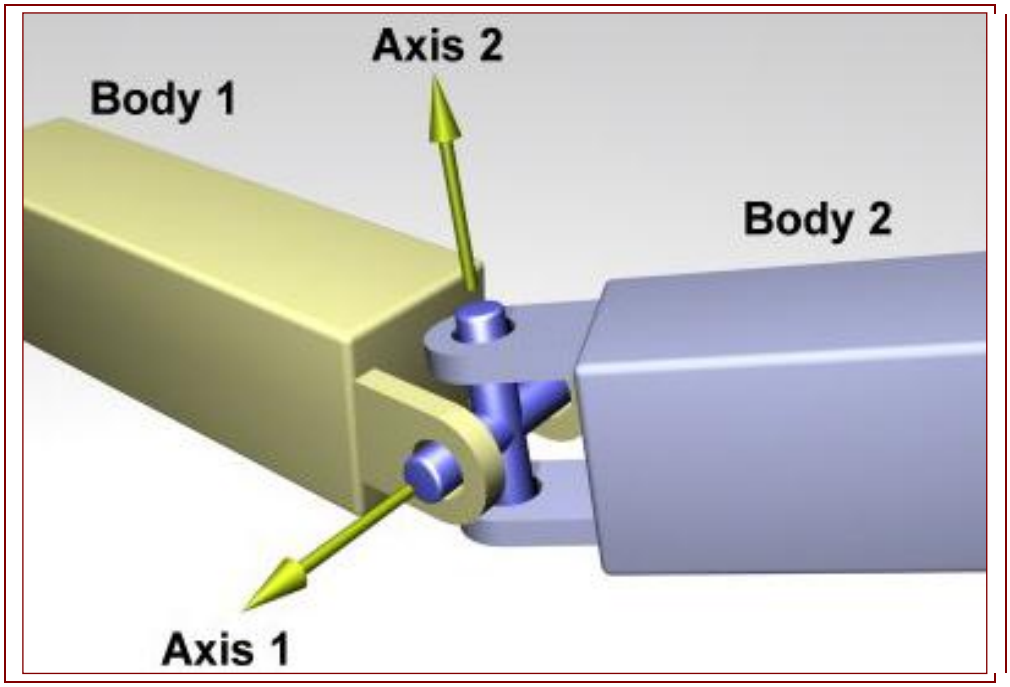

Figure 5.

Connection to CyberGlove

Collection of the joint angle data from the CyberGlove device is available through the API provided with the VirtualHand SDK. In the early stages of program initialization a connection is opened to the device where upon data can be read in subsequent simulation time steps. In this program, after the data is read from the glove it is transferred to and stored in the data structures for the associated hand part.

```
gloveDict = vhtIOConn::getDefault( vhtIOConn::glove );
glove = new vhtCyberGlove(gloveDict);

...

glove->update();
```

Figure 6.

Simulation Loop

Like world initialization the simulation loop in this program is representative of a typical ODE simulation loop. Collision detection is performed and the environment state is advanced one time step. During each simulation loop angle data is also read in from the glove. The data from the glove is used to determine how much to adjust the joints angles. The hand and objects are redrawn. Figure 7 shows an image capture of the simulation.

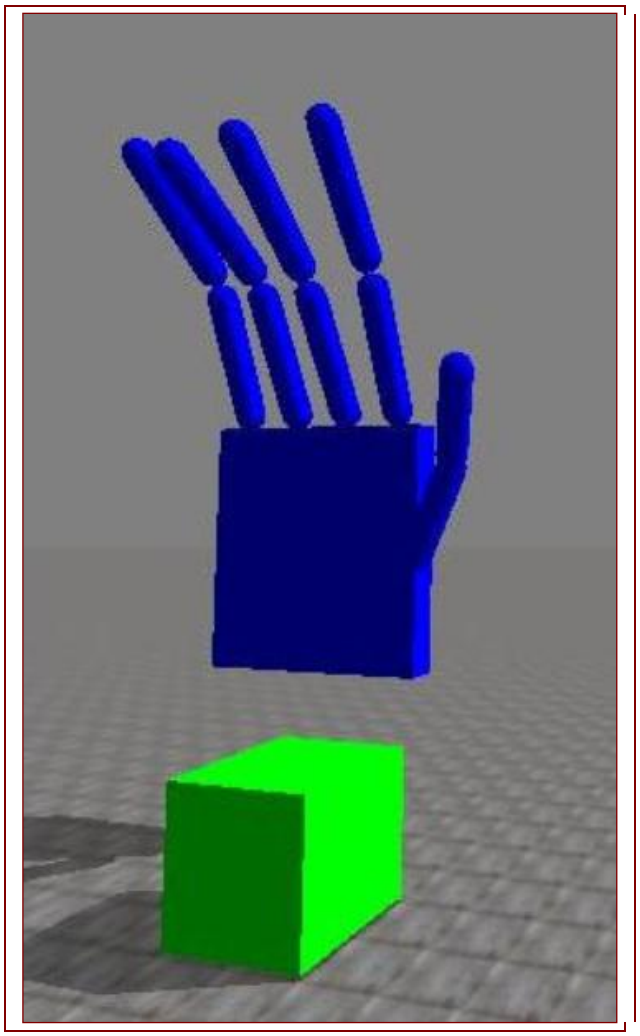

Figure 7.

Contact Data

Each finger and palm object stores a data structure holding the collision information from the most recent interaction with an object. These are updated every time step and should be read out if the data is intended to be saved. Interactions between parts of the hand are ignored.

```
struct ContactFeedback {
      dJointFeedback feedbacks[MAX_CONTACTS];
      int numContacts;
};
```

Figure 8.

Joint motors

Each time step the joints are adjusted to match the readings from the CyberGlove. This is accomplished with a servomechanism that applies a variable force through an angular motor based on the necessary correction value. The maximum force as well as the gain can be adjusted to tune the operation of the movement.

```
dReal truePosition = dJointGetHingeAngle(currentFinger->hingeUpper);
dReal desiredPosition = (currentFinger->angleUpper + calibration[i][0])
      * calibration[i][1] * DTOR;
dReal error = truePosition - desiredPosition;
dReal desiredVelocity = -error * gain;
dJointSetHingeParam(currentFinger->hingeUpper, dParamFMax, maxForce);
dJointSetHingeParam(currentFinger->hingeUpper, dParamVel,
      desiredVelocity);
```

Figure 9.

4. Results

The virtual hand exhibits mostly realistic hand movement and the ability for basic interaction with other objects. With just the standard settings for the rigid body collision detection provided with ODE, though, the hand is not currently able to perform complicated tasks like grasping an object. The hand size and mass as well as object specifics have not yet been adjusted to emulate precise realistic scenarios.

5. Discussion

These results are expected given the configuration of the standard rigid body simulation parameters. With modifications to the collision detection parameters a more realistic grasping mechanism should be accomplishable. ODE also supports the ability to easily integrate with alternative collision detection libraries if the provided ODE collision engine provides insufficient customizability to achieve the ideal configuration.

Currently this program uses a rudimentary method for displaying a graphical representation of the simulation environment. This feature could be replaced with a more advanced display library without much difficulty.

A 3-dimensional position and orientation sensor would also be a useful addition to this project. Gathering and integrating this data in a manner similar to the CyberGlove data collection would allow for a more powerful and intuitive control scheme for the virtual hand.

The implementation successfully overcomes the major difficulties in integrating the CyberGlove sensor data into a realistic physical simulation environment. For simple object interactions the virtual hand affects the simulated environment in a realistic manner. With the additions or expansions to the features described above the tool should be even more effective in obtaining useful information on the techniques of

dexterous object manipulation.

6. Conclusion

The integration of the CyberGlove into an ODE simulated virtual environment was described and evaluated. The results were discussed and improvements to the tool were proposed. Using and improving on the results of this project should allow for an environment where research and experiments can be conducted to better understand human hand control during object manipulation.

References

[1] ODE Manual. http://opende.sourceforge.net/wiki/index.php/Manual , 2009.