# Journal of Virtual Worlds Research

jvwresearch.org ISSN: 1941-8477

## The Metaverse Assembled

Volume 2, Number 5

# Volume 2, Number 5
# The Metaverse Assembled
# April 2010

The Journal of Virtual Worlds Research
is owned and published by the
Virtual Worlds Institute, Inc.
Austin, Texas, USA

Journal of Virtual Worlds Research
jvwresearch.org ISSN: 1941-8477

**Volume 2, Number 5**

**The Metaverse Assembled**

**May 2010**

# ExtSim: A Flexible Data Mapping and Synchronization Middleware for Scientific Visualization in Virtual Worlds

By Johan Berntsson, Norman Lin, Zoltan Dezso
3Di Inc, Japan

## Abstract

*In this paper we present a general-purpose middleware, called ExtSim, that allows OpenSim to communicate with external simulation software and to synchronize the in-world representation of the simulator state. We briefly present two projects in ScienceSim where ExtSim has been used: Galaxsee which is an interactive real-time N-body simulation, and a protein folding demonstration, before discussing the merits and problems with the current approach. The main limitation is that we have been limited to a third-party viewer only, and a fixed server-client protocol, but now we can present our work on a new viewer, called 3Di Viewer "Rei," which opens new possibilities by enhancing both performance and richness of the visualization and it is suitable for scientific computing. Finally, we discuss some ideas we are currently studying for future work.*

Keywords: scientific visualization; simulation; OpenSim; ExtSim

# ExtSim: A Flexible Data Mapping and Synchronization Middleware for Scientific Visualization in Virtual Worlds

By Johan Berntsson, Norman Lin, Zoltan Dezso
3Di Inc, Japan

OpenSim (The OpenSim Project) is an open source virtual world server, which in its standard configuration implements the Second Life server protocol from Linden Labs. The software allows users to build content and interact with each other in virtual worlds, using Second Life compatible client software. OpenSim is implemented in C# and runs-on .NET or mono, making it available both on Unix and on Windows platforms. OpenSim implements regions ("sims") that can be linked in grids, either on the same server, or across several servers. The user can then move between regions, or even between grids, by teleporting in the same way as in Second Life.

Although the main usage until now has been social networking, there is interest in using the technology for other applications. This has led to the development of distributions based on OpenSim, such as ScienceSim which is designed to fill the needs of scientific computing and visualization.

OpenSim provides new opportunities to implement, display and manipulate simulations not present in many of the virtual reality systems discussed in the literature to-date. The main difference is that OpenSim allows collaboration, in that many avatars share the same space and are able to talk to each other (either by voice chat or messaging), look at objects and videos together, and move around using realistic physics to prevent people from occupying the same space. The contents of the world can be created either directly by the avatars while running the application, or by importing existing models from other 3D design software. The objects in the world can be controlled by scripts and various visual effects, such as different particle system algorithms which can be used to display physical phenomena.

## Scientific Computing in OpenSim

There has been a keen interest in virtual reality to display and manipulate computer simulations and advance general scientific understanding. OpenSim, with its focus on social interaction and collaboration provides new opportunities not featured in previous systems. Consequently, there is an interest in applying OpenSim to new application areas, such as astrophysics (Djorgovski et. al., Farr et al.). One recent development is ScienceSim (The ScienceSim Project), which is an OpenSim distribution customized to meet the needs of scientific computing. ScienceSim is a part of the 3D Internet initiative, which is a Thrust Area at SC09 (The 3D Internet Thrust Area).

Scientific computing applications generally require a lot of computing power (Pentland), and a typical simulation will be too heavy to run on the OpenSim servers. One solution to this problem is to run the heavy physics simulation on an external physics server, and to represent the calculation with in-world objects whose attributes are calculated using values from the physics simulation. The state of the OpenSim and external physics servers must be synchronized, which could be done either by extending the scripting languages to read and write data from the external server, or by defining a mapping between in-world variables and those of the external physics server. The latter method is more flexible.

In the next section, we present a general-purpose middleware, called ExtSim (The ExtSim Project) that allows OpenSim to communicate with external simulation software and to synchronize the in-world representation of the simulator state. We also discuss two projects in ScienceSim where ExtSim has been used, Galaxsee (Gibbon, Murphy, and Peck, 2009) which is an interactive real-time n-body simulation, and a protein folding demonstration which is also supplied as an example application with the ExtSim source code distribution. The final part of the paper discusses the limitations of the current approach and possible direction for future work aimed at enhancing the utility of OpenSim for scientific computing.

## The ExtSim Middleware

ExtSim (**Ext**ernal **Sim**ulator Bridge) is an OpenSim plug-in, which has access to the internal state of the OpenSim server. ExtSim polls data from an external server and creates and modifies virtual objects on the OpenSim server (see Figure 1 below). The server then sends object updates to any attached client viewer, which renders the objects on the screen. The objects created by ExtSim are under full control of the external server, and not affected by the internal OpenSim physics engine.

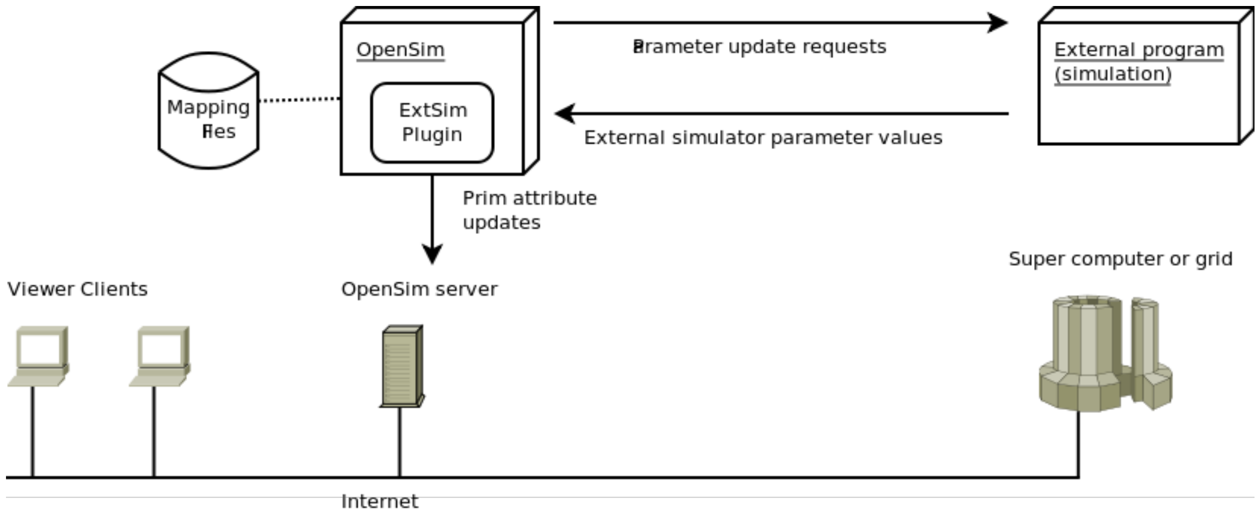


**Figure 1.** The communication links between OpenSim, ExtSim and external simulators.

ExtSim currently uses XmlRpc to achieve a platform-independent data transfer from and to any cluster resource that has an active connection to the Internet and provides external server stubs in several languages, including C, Python, and Clojure. The protocol is defined by data files that specify the update frequency and the location of the external server: the mapping of OpenSim object is attributed to variables in the external simulation. Each of these mappings can control either the position, size, or color of the OpenSim object. In addition, events from OpenSim (for example, pressing a button) are sent to the external server, allowing a two-way communication between the servers.

Object updates on the OpenSim server cause update messages to be sent to each attached client viewer. When the simulation includes many objects, this can be an expensive operation that limits scalability. To reduce this load, ExtSim puts an upper the number of internal OpenSim updates caused by new data from the external simulation by keeping track of both the current actual position of the object and the new position received from the external process. For each object, a weighted displacement is calculated using this formula:

$$ageBias = 1 + \frac{currentTime - lastUpdateTime}{updateBias}$$

$$weightedDisplacement = distance(actualPosition, newPosition) * ageBias$$

Thus, the update bias parameter determines how much bias is given to old parameter updates that have not yet been updated in the OpenSim server.

The object with the largest weighted displacements will now be updated, until a maximum number of objects have been moved. The age bias promotes older objects so that smaller object updates will eventually show up on the client screen, even in the presence of objects with frequent large displacements.

## ExtSim Applications

In this section, we briefly present two applications that use ExtSim to demonstrate the use of virtual world technology to display computer cluster calculation results.

### *Protein Folding*

Protein folding is an example of the computationally heavy applications that typically cannot run in parallel with OpenSim on a server. As a demonstration of ExtSim, the ExtSim project includes a demo server, which publishes a data stream from a protein folding experiment. ExtSim reads data from the server and shows the position for each atom in the molecule in an OpenSim region. The demo data stream is short, and a longer demonstration can be seen on the "3Di ExtSim Demo" region in ScienceSim (see Figure 2). The data is visualized in OpenSim as a cloud of spheres, representing atoms in the molecule. When ExtSim receives new position data for an atom, its corresponding sphere is moved in the OpenSim scene. OpenSim will then automatically update all connected viewers.

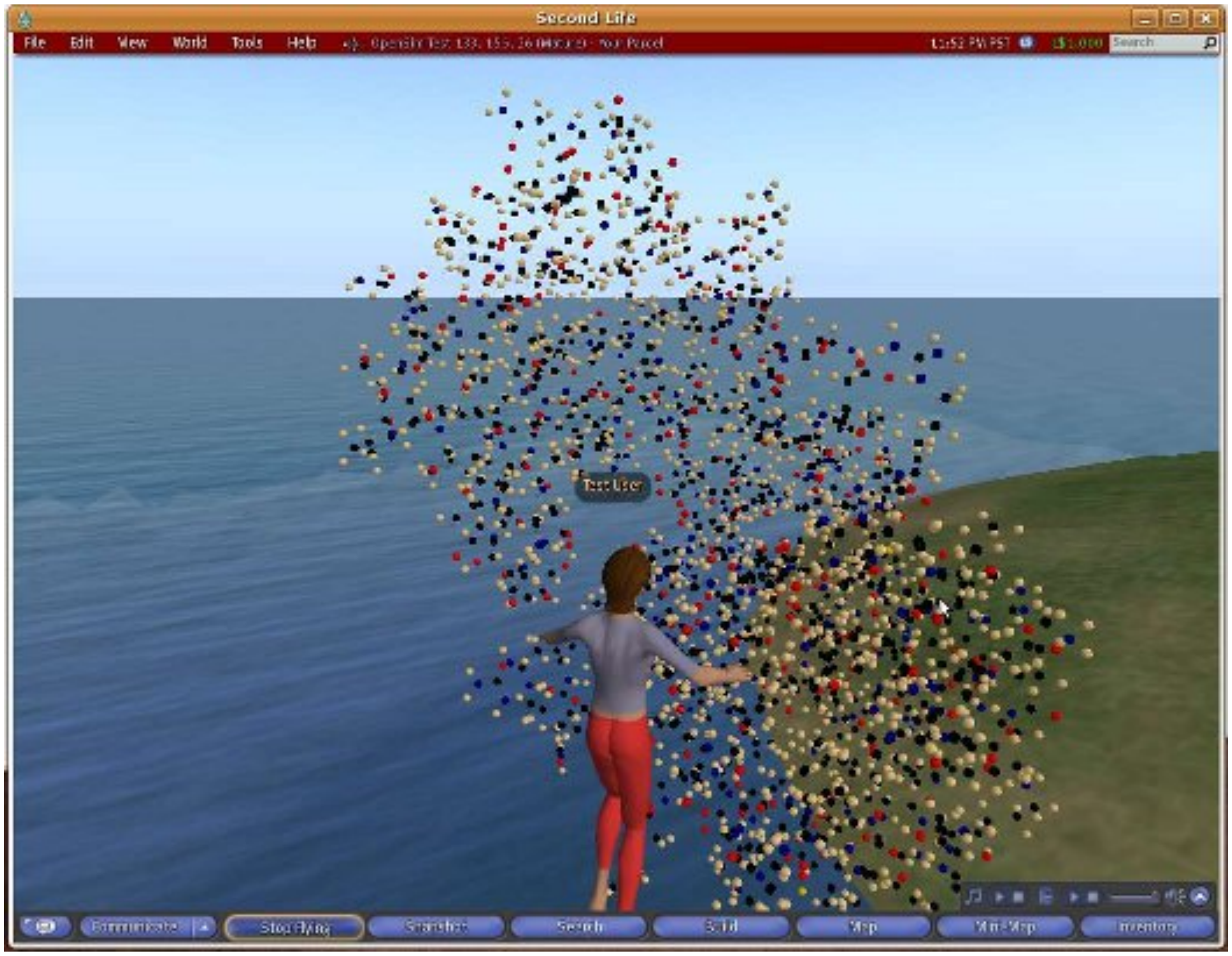


**Figure 2.** An OpenSim user observing a protein folding experiment.

### *Galaxsee*

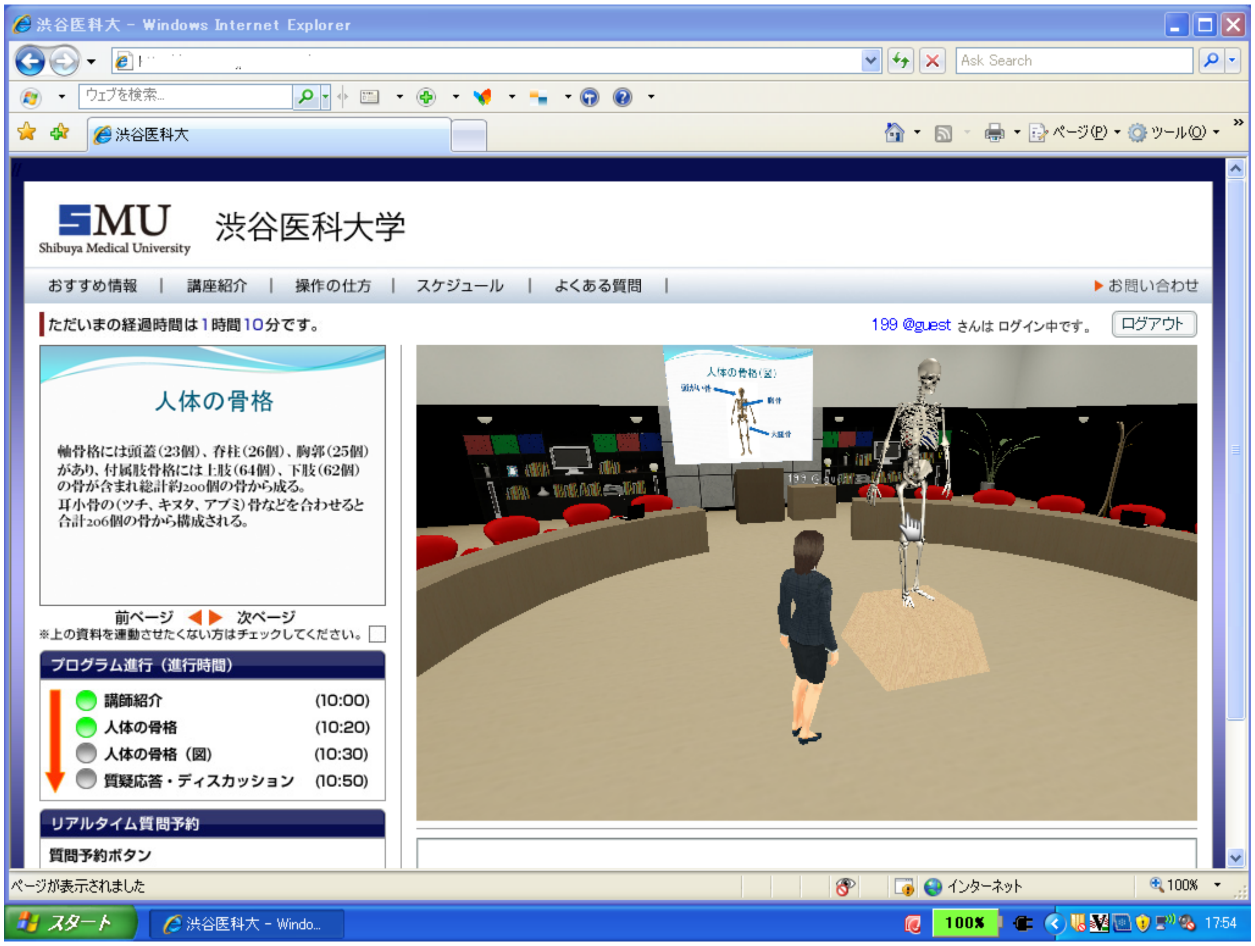
Galaxsee is a demonstration of the mechanisms needed to connect real-time simulation (in this case an N-Body problem) into a collaborative virtual environment. The output is displayed as spheres, representing stars in a spiral galaxy, in a ScienceSim region, which is otherwise dark. A user can control various aspects of the simulation, and examine the model from different vantage points, possibly in collaboration with other users. The project uses ExtSim for updating in-world contents, using data from an external computer cluster on which the simulation is run.

## Problems and Future Work

Unfortunately, technology restricted what we were able to achieve, since we had to work only on the server side. Things like atom bonds had to be represented as prims, creating quite a bit of overhead for simple things like drawing a line between two spheres, and we were not able to scale to more than 1,000 atoms. Scalability problems were also reported in the Galaxsee project (Gibbon, Murphy, and Peck 2009). Furthermore, the current implementation requires active polling of data from the external source, which leads to an inefficient usage of bandwidth.

### *The Need for New Viewer Software*

Although the Second Life viewer software is widely used by OpenSim users, one problem is that OpenSim developers cannot make changes or improvements to this viewer, due to software license incompatibility between the OpenSim project and the Second Life viewer software. Inability to alter the client side software thus limits OpenSim scientific visualization solutions to using the existing viewer facilities; new viewer capabilities or new network protocols cannot be added to the Second Life viewer by OpenSim developers.



ただいまの経過時間は1時間10分です。

199 @guest さんは ログイン中です。

人体の骨格

軸骨格には頭蓋(23個)、脊柱(26個)、胸郭(25個)があり、付属肢骨格には上肢(64個)、下肢(62個)の骨が含まれ総計約200個の骨から成る。
耳小骨の(ツチ、キヌタ、アブミ)骨などを合わせると合計206個の骨から構成される。

※上の資料を連動させたくない方はチェックしてください。

プログラム進行(進行時間)

| | |
|---|---|
| 講師紹介 | (10:00) |
| 人体の骨格 | (10:20) |
| 人体の骨格(図) | (10:30) |
| 質疑応答・ディスカッション | (10:50) |

**Figure 3.** The Rei viewer, embedded in a Web page showing a virtual medical seminar room.

Recognizing this need, we have recently developed and open-sourced a browser-based viewer called 3Di Viewer “Rei” (The “Rei” Viewer Project), which is compatible with OpenSim. Figure 3 shows the Rei viewer embedded in an html page.

With the new viewer, we are planning to extend the protocol to allow line nodes and meshes, in addition to the normal prim-based content used by Second Life. This will help us to increase scalability in the protein-folding problem, since we will not need heavyweight prims[1] to represent atom bonds. However, Rei also extends the current capabilities of the standard Second Life viewer to a new viewer in a way that allows new ideas in 3D data visualization, which we want to investigate in future work.

### *HUD Display and Control*

With a more advanced viewer, we will be able to enhance the rendered 3D image with other information. For instance, we will be able to add a transparent surface graph showing things like temperature or pressure levels directly on top of the 3D image and to move the graph together with the rendered scene. We can also add effects, such as an arrow showing the flow of steam, water, or even people's movements on top of the image.

The Rei viewer could support Head-Up Displays (HUD) that can be used to display data, such as trend curves, graphs, data tables, or even animations and videos. The display can also contain control elements, such as buttons and sliders, which can activate scripts that control the simulation. One HUD use case is to access data that is normally hidden. For example, imagine that we have a room showing the nuclear reactor hall. By clicking on a pipe or a valve, we can pass the simulator parameters for this object and appropriate control elements. A valve could show its current position (e.g. 37% open) and buttons to open, close and stop the valve motor. A pipe could show a trend curve that displays pressure, temperature, and vapor levels. Various buttons can be used to change the time range, or displayed variables.

### *3D Browsing*

Since each avatar only sees his or her immediate neighborhood, we may need a way to move about in a larger 3D space, as well as keeping track of other participating avatars. The Second Life viewer has a map mode that either shows a global overview of all areas or an outline of the current area the avatar is standing on, but the global information is quite limited.

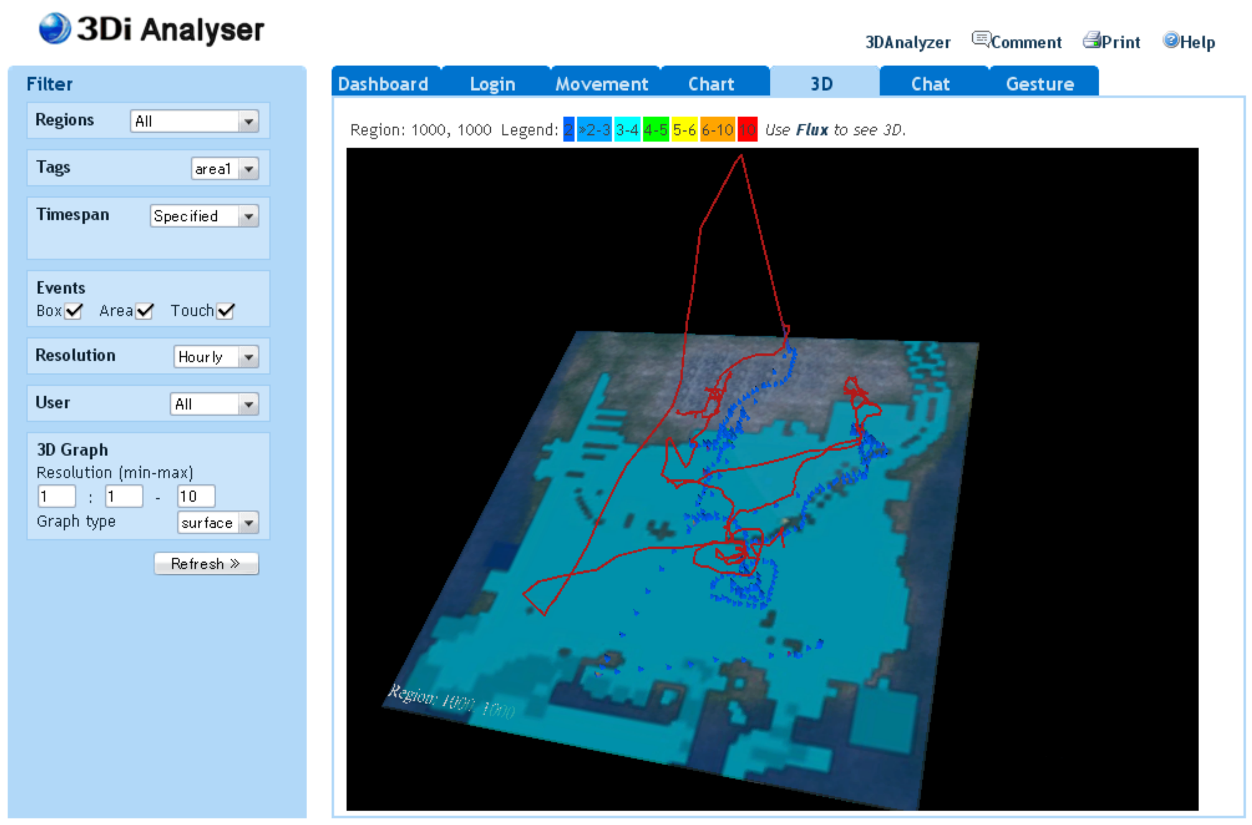
[1] The word prim is a shortened version of "primitive" and refers to the basic 3D polygonal shapes used to build objects in OpenSim.

We think that a promising area for future research is experimenting with ways of adding external video streams and external pictures to the 3D world. Using VNC links, external applications or desktops display can be rendered on a surface in the 3D world. This can be used to let the avatar interact with external software, or share information by sharing their desktop contents with other avatars. Moreover, we can create an HUD display of several views of the environment through virtual in-world video stream from fixed positions. This allows the user to interact with the current surroundings while still seeing what is happening further away. A natural extension would be to allow the user to teleport between different areas of the environment by simply clicking on the area in the HUD that shows the area he/she wants to visit.

### *Collaborative Tasks*

The multi-user nature of OpenSim environments makes is possible to study how humans interact with other people and with the simulated world. In addition, the social aspect makes OpenSim a valuable tool for cooperative investigation and presentation of scientific data. At the same time, the collaborative aspect could also be a valuable tool for man-machine studies. For example, OpenSim makes it easy to make an environment where several people can meet and cooperate to carry out tasks. An application in the nuclear industry could be to model a control room, in which typically 4-5 operators with different specialties monitor and control the plant together. Since the model can be very close to reality, operators can easily work in the simulation as they would in a real plant without special training. However, the physics model may differ and the setup can be used in research and training when considering new or modified hardware in a plant.

Another application area is crowd control, e.g. monitoring how people move around during normal operations or in emergencies. Are there bottlenecks? Are people able to access areas of interest in time? We are working on solutions where such data can be stored in log files and later analyzed in programs that can summarize information over various categories, such as a given time period, only in a given area, and so on; as well as to show the actions of single individuals (see Figure 4).



Regions: All
Tags: area1
Timespan: Specified
Events: Box ✓ Area ✓ Touch ✓
Resolution: Hourly
User: All
3D Graph
Resolution (min-max): 1 : 1 - 10
Graph type: surface

Region: 1000, 1000 Legend: 2, >2-3, 3-4, 4-5, 5-6, 6-10, 10

**Figure 4.** Data analyzing tool, showing recorded avatar movements in 3D space. The view is animated and can be rotated freely.

## Conclusion

OpenSim has unique strengths compared to traditional virtual reality and data display systems, and the previous sections briefly describe ideas on how OpenSim may be used in Scientific Computing. In this paper, we have presented a new middleware for OpenSim called ExtSim, which has been used by several project groups and we have discussed the advantages and limitations of the current approach. We have identified that good performance requires source code modifications to the viewer software, and presented work done on a new open source viewer project, which could enable more efficient future implementations.

## Bibliography

Djorgovski, S. G., Hut, P., McMillan, S., Vesperini, E., Knop, R., Farr, W. M., et al. 2009b. *Exploring the Use of Virtual Worlds as a Scientific Research Platform: The Meta-Institute for Computational Astrophysics*. In J. Sablatnig (Ed.), Proceedings of the FaVE 2009 Meeting. Springer Verlag.

Farr, W.M, Hut, P., Ames, J., Johnson, A. An Experiment in Using Virtual Worlds for Scientific Visualization of Self-Gravitating Systems. *Journal of Virtual Worlds Research*, (Vol. 2, number 3).

Gibbon, A.F., Murphy T. Peck C.. (2009, September). *Science Visualization, Collaboration, and Education through Virtual Worlds*. Poster, Intel Developer Forum.

Pentland, A. *Computational Complexity Versus Virtual Worlds*. ACM SIGGRAPH Computer Graphics: (Vol. 24, issue 2, pp. 185–192).

The 3D Internet Thrust Area. *Super Computing conference 2009 Thrust Area*. Retrieved from http://sc09.supercomputing.org/?pg=thrustareas.html.

The 3Di Viewer “Rei” Project. (2009). Retrieved from http://3di-rei.org/ .

The External Simulator Bridge Project. Retrieved from http://forge.opensimulator.org/gf/project/extsim/.

The OpenSim Project. Retrieved from http://opensimulator.org/.

The ScienceSim Project. Retrieved from http://sciencesim.com/.