

# Interactive Virtual-Reality Driven Learning Framework for Engineering and Science Education

Ehren Biglari<sup>1,3</sup> and Yusheng Feng<sup>1,2</sup>

<sup>1</sup>Center for Simulation, Visualization and Real-Time Prediction

<sup>2</sup>Department of Mechanical Engineering

<sup>3</sup>Department of Computer Science

The University of Texas at San Antonio

## Abstract

The University of Texas at San Antonio has established a cutting edge Advanced Visualization Lab (<http://vislab.utsa.edu>) that consists of a large-scale high-resolution VizWall powered by a hybrid CPU/GPU LINUX cluster with 6,000 processors, and various latest visualization technology including large screen 3D stereoscopic display, immersive 3D Oculus Rift virtual reality device, Google Glass, multi-touch visual control, Lean Motion gesture control, haptic control devices, Kinect enabled model control as well as laser scanner and 3D printers. It provides us a comprehensive visualization facility capable of building a simulation and visualization pipeline from start to finish.

In this paper, we first present a general framework on how to develop and integrate some of the visualization devices into the large-scale (14.5 x 6 ft<sup>2</sup>) high-resolution (98 Mpixels) VizWall for interactive learning. Then, we discuss a few application examples that are built on this platform include classical engineering beam deformation, biomedical virtual surgery simulation, artistic application with Kinect to track a dancer so that a live video mask can be interactively controlled by the dancer's movements, as well as a football kicking simulation system to provide interactive visual-based feedback to both players and coaches with model-based coaching techniques for performance improvement and injury mitigation. With rapid advancement of today's technology, it can make learning of various subjects of engineering and science more interesting and effective by designing interactive and visually stimulating curricula.

## 1. Introduction

The visualization framework we developed is based on the open source Processing programming language. Processing is a set of libraries (<http://processing.org>) that extend the Java language built for electronic arts, new media art, and visual design communities to teach the fundamentals of computer programming in a visual context. The Processing language aides in the rapid development of visualizations while providing an environment that is easy to learn. We chose this environment to make the development of custom scientific visualizations easy for researchers, to minimize the time spent on visualization development. In addition, the platform also has the added benefit of being suitable for real-time rendering to any platform running Java. Furthermore, integrating visualizations with the Massive Pixel Environment (MPE) library developed by the Texas Advanced Computing Center at the University of Texas at Austin we can display and interact with visualizations on the large-scale high-resolution VizWall.

In addition to Processing and the MPE library, we have added our own framework for programmable data-driven visualizations, integration with the Oculus Rift Virtual Reality headset, and integration with the motion and orientation controller the Sixense Razer Hydra. We will discuss each of these components and how they are integrated into a single immersive visualization environment.

## **2. Programmable Data-Driven Visualization and Human-Computer Interaction**

We have developed a programmable data-driven visualization framework - Immersive Data Visualizer (IDV) based on the Processing programming language, which consists of eight components: 1) a Wavefront OBJ loader, 2) an XML data file loader, 3) a 3D force-directed graph algorithm, 4) a rendering module, 5) an animation module for time series visualization, 6) a Oculus Rift VR headset integration module, 7) a Sixense Hydra Razor controller integration, and 8) a VizWall integration module.

The Processing programming language permits rapid development of scientific visualization due to its simplicity and its ability to provide a programming environment supporting interactive 2D and 3D output and OpenGL applications. The Processing development can be done via the Processing IDE or via the creation of PApplet in a Java IDE such as Eclipse for Java. We choose the latter because it provides additional auto-complete tools that can assist researchers with rapid development. The framework we developed with the Processing language can either be used as a programmable template for scientific visualizations or as a programmable data-driven platform for scientific visualizations.

Although Processing has many tools for 3D visualization, one of the core elements needed for 3D visualizations, a built-in Wavefront OBJ 3D model loader, was severely lacking in its capabilities. Rather than using the built-in loader, we extended the add-on Wavefront OBJ loader maintained by SAITO at UCLA and Matt Ditton. To fix surface normal for 3D models we added methods to recalculate smooth normal, which can be applied to scientific visualizations of surface normals. In addition, we fixed code that loads texture and normal indices, added shininess material properties, and added code to provide render-to-texture capabilities. This provided us with a fully functional OBJ loader for scientific visualizations.

The next component we developed was the XML data file loader, which is designed to load research data with little or no modifications. The XML file can reference other files including: OBJ files, CSV point clouds files, CSV links files, and topological XML data files (see Appendix A). The CSV link files specify offsets into the point clouds, allowing researches to indicate relationships between data points. The CSV point cloud files can be 2D or 3D point clouds, with additional per-point color, scale, rotation, and vector information. The 2D point cloud files require additional XML information that specifies how the 2D should be projected into 3D, either on a sphere, or a cylinder. The points in the 2D or 3D point clouds can then be replaced by OBJ files by referencing the OBJ file. The topological data file can be in either CSV or XML file formats.

The XML data file loader uses built-in Processing XML file reading libraries to read the XML file information. Once all of the information has been read from the XML file, the data loader loads the referenced OBJ files, CSV files, and XML files. TimeFrame objects are created for each time frame of data. Each time frame consists of a set of data points. Each data point is an object that can contain multiple x,y, and z, along with multiple vectors, colors, points, rotations, and scaling parameters, all user-defined in the main XML file.

The 3D force-directed graph algorithm uses the Fruchterman-Rheingold algorithm along with octree partitioning of world space from the Toxic library for fast node distance calculation. The topological data points are individually added into the force-directed graph as it expands. Optionally, the force specified in the XML file can be increased or decreased. Each node can also be labeled with additional data provided in the topological data file. This tool provides researchers with the ability to visualize topological data sets. Some examples of such data could include structural topology, temporal hierarchies such as evolutionary taxonomy, or C++ class diagrams.

The force-directed graph algorithm also has several parameters that may be tuned including: attraction, repulsion, iterations, minimum length, minimum temperature, and force scale. Increasing the attraction parameter tends to force nodes closer together. Increasing the repulsion parameter tends to force nodes further apart. Increasing iterations tends to stabilize the node configuration more, unless the minimum temperature is already reached. Increasing the minimum temperature tends to lead to less stable configurations, but faster convergence time.

The rendering module is designed for fast rendering of data. In order to optimize the rendering loop we pre-render 3D models to textures and use sprites to render many copies of the object, one per data point. In addition, we use textures to pre-render any per-node text data that may need to be rendered. While rendering the points we also render any vector data associated with the points. Each time step is used as a frame in an animation where either linear or spherical linear interpolation is used. In addition, to rendering to a monitor, we have the option to render to the Oculus Rift.

## **2.1 Oculus Rift Integration**

The Oculus Rift is Virtual Reality headset with orientation tracking that creates an immersive virtual environment for the wearer that is being developed by Oculus VR. The Oculus Rift “dev kit” version currently has a resolution of 1280x800, however, the final commercial product is expected to have a resolution of 1080p. Oculus VR provides the Oculus SDK, a C++ library for Oculus Rift integration. Using the Java Native Interface we created a Java library that can be used for integration with the Oculus Rift. On a thread separate from the main thread the current yaw, pitch, and roll rotations of the headset are read every 50 milliseconds and the data is passed to the main control module. From there, the render loop is yaw, pitch, and roll are modified. Using OpenGL shaders the inverse lens geometry of the rift is mimicked on screen. When the user wears the Oculus Rift headset they see an undistorted 3D perspective. As the user turns their head the 3D environment rotates in the appropriate direction, giving the illusion of an immersive 3D environment.

## **2.2 Razer Hydra Controller Integration**

The Sixense Razer Hydra motion and orientation controller has two controllers, each with six degree of freedom position and rotation tracking, trigger and bumper buttons, along with a small thumb controlled joystick on each controller. The Razer Hydra motion and orientation controller is ideal for 3D spatial navigation or object manipulation in a virtual world. Similar to the Oculus, the C++ Sixense SDK was used to interface with the Razer Hydra. The Java Native Interface was used to produce a Java library to interface with the Sixense Razer Hydra controller. The library monitors the x,y, and z position, as well as rotation of both the left and right controllers. In addition, button pushes are monitored. On a thread separate from the main thread, the Razer Hydra controller is polled every 50 milliseconds and data is passed to the main control module. The control module uses this information, in conjunction with information from the Oculus Rift to calculate 3D position and rotation for rendering. The Razer Hydra allows users to fly through the 3D environment and select data points for further inspection.

## **2.3 VizWall Integration**

The large-scale (14.5ft x 6 ft) high-resolution VizWall is composed of 24 monitors each running at a resolution of 2560x1600. Horizontal pairs of monitors are controlled by single workstation, and an additional workstation as the head controlling node, for a total of 13 workstations controlling the VizWall. In order to display a Visualization we use Massive Pixel Environment (MPE), a Processing software library developed by the Texas Advanced Computing Center (TACC). MPE was designed to synchronize multiple instances of Processing across multiple workstations controlling an array of monitors. As long as the Processing applications are deterministic they will stay synchronized. In addition to synchronizing the applications, each the OpenGL camera position is controlled to tile a single 3D image across the VizWall. With minor modifications to the MPE library the VizWall can be used to display

## **3. Visualization for Education**

There are numerous potential applications for the IDV with VizWall integration. Although each of these applications has their own visualizations tools, the IDV serves as a single platform capable of performing various tasks across a diverse array of subjects and areas of study allowing interdisciplinary interaction in a real-time immersive programmable data-driven environment. Here we discuss a few examples for Engineering and Science Applications, and then demonstrate its capability for broader applications such as in Music and Arts.

### **3.1 Engineering and Science Applications**

Scientific visualizations are an integral part of science and engineering education. In the learning process, visualization as a cognitive skill plays a central role in navigating different modes of representation. In addition, visualization plays a role in dissemination and acceptance of knowledge acting as a bridge between scientific theory and the world-as-experienced. Visualizations can also act as idealizations allowing one to make connections between imaged and observed reality [1]. Thus, a single integrated tool for science and engineering visualizations, such as the IDV, that brings together immersive visualization with ease of data-driven

visualizations and the power of high-level programmatic visualizations, provides a powerful educational tool for teachers and students.

The data-driven platform allows for rapid development of visualizations with engineering and science applications. When applied to software engineering, the data-driven visualizations may be used to display a 3D UML diagram. Visualizations developed by 3D UML modelling tools such as X3D-UML or other 3D UML modelling tools can be exported to Wavefront OBJ and displayed in an immersive environment with the IDV. Alternatively, UML diagram images from any UML tool can be mapped to the surface of a cylinder in the virtual world using built-in data-driven XML, or the visualization can be converted to textures. Several studies have shown that algorithm visualization increases test performance and student engagement [2-4].

In addition to applications in software engineering, the IDV has applications for biochemistry visualization. The past couple of decades have seen an explosion in knowledge and data. Conveying the core concepts involved with this knowledge is an important central goal in biochemistry education. In order to convey this knowledge effectively visualization is an essential cognitive skill that is enhanced by scientific visualizations [5, 6]. Existing computer and web-based resources currently make use of visualizations for enzyme-substrate binding, protein folding, and the dynamic structure of a cell membrane [7]. Many of these visualization can be quickly and easily generated with PyMol molecular visualization software and exported to Wavefront OBJ keyframes and displayed with the IDV, making these visualizations accessible on the VizWall and the Oculus Rift, with the Sixense Razor Hydra controller to grab and manipulate molecules in the virtual world or to navigate around to virtual world [8]. In addition to displaying 3D chemical models and animations, the IDV can display semantic networks applicable to biochemistry [7]. Semantic networks can generated with Scalable Vector Graphics (SVG) format and converted to OBJs with a topological CSV describing semantic relationships. If a particular connection layout is preferred, a static link CSV file can be used to generate the visualization.

There are also many applications for chemistry visualizations. Similar to biochemistry, the cognitive skill of visualization in chemistry is complemented by the scientific visualization of 3D chemical bonds and ionic structures, along with semantic relationships [9-11]. In chemistry, existing tools such as WebMO and PyMol can be used to generated visualization that can be easily exported and displayed with the IDV [8, 12]. Existing immersive environments for chemistry visualization also exist, such as the CAVE<sup>TM</sup> environment, have been tried with some success in improving student comprehension [13]. However, such systems have drawbacks including the cost, the relatively large amount of space needed, and the potential for viewing occlusion during collaboration. With the VizWall and MPE synchronization, it is possible to synchronize multiple Oculus Rift headsets with the VizWall. For a classroom, one student can use an Oculus Rift Virtual Reality headset and “pilot” the visualization, while other students view the exact same visualization on the VizWall in real-time and direct the “pilot”.

Several applications for visualization exist in the geosciences as well. The GeoWall was originally developed in 2001 as a single small scale low resolution (1080p) stereoscopic 3D visualization wall for the geoscience research and education [14]. Later iterations of the GeoWall in 2003 used an array of monitors, similar to the VizWall, but with only a total

resolution of 8,000x3,600. When rendering 3D images on the GeoWall each eye sees a resolution of 4,000x3600 or about (14.4 Mpixels). The later 2003 GeoWall works with polarized glasses to provide a 3D stereoscopic image. Since the construction of the original GeoWall in 2001 over 500 GeoWalls have been built.

The GeoWall uses several different software components including Immersaview to display 3D animations with in OpenInventor and VRML formats, Partiview to display particles, and ArcGIS commercial mapping software. The drawback to these software packages is that they are not programmable, they require separate software packages, and they do not allow natural spatial navigation with head movement or the total immersion of the Oculus Rift. However, the data for each these software packages can be easily exported to Wavefront OBJ or other formats using Blender 3D modelling software to convert the models to OBJ keyframe animations. In addition, raw data may be loaded and superimposed on maps exported from ArcGIS. Particle systems can be loaded from the CSV file and programmed in the high-level language of Processing to move in relation to loaded contour maps from ArcGIS. Using maps exported from ArcGIS, a variety of dynamic geoscience visualizations can be created and displayed on the large-scale high-resolution VizWall, while an Oculus Rift user navigates in the 3D world with the Sixsense Hydra Razor and Oculus Rift directional navigation.

In addition to geoscience visualizations, the IDV has applications for medical visualizations. Existing 3D medical visualizations have been developed that help students gain a better understand of the differences between normal and abnormal function in the human body [15, 16]. Using models obtained from a 3D model library the Oculus Rift visualization was generated that allow the user to remove a kidney (see Fig. 1).

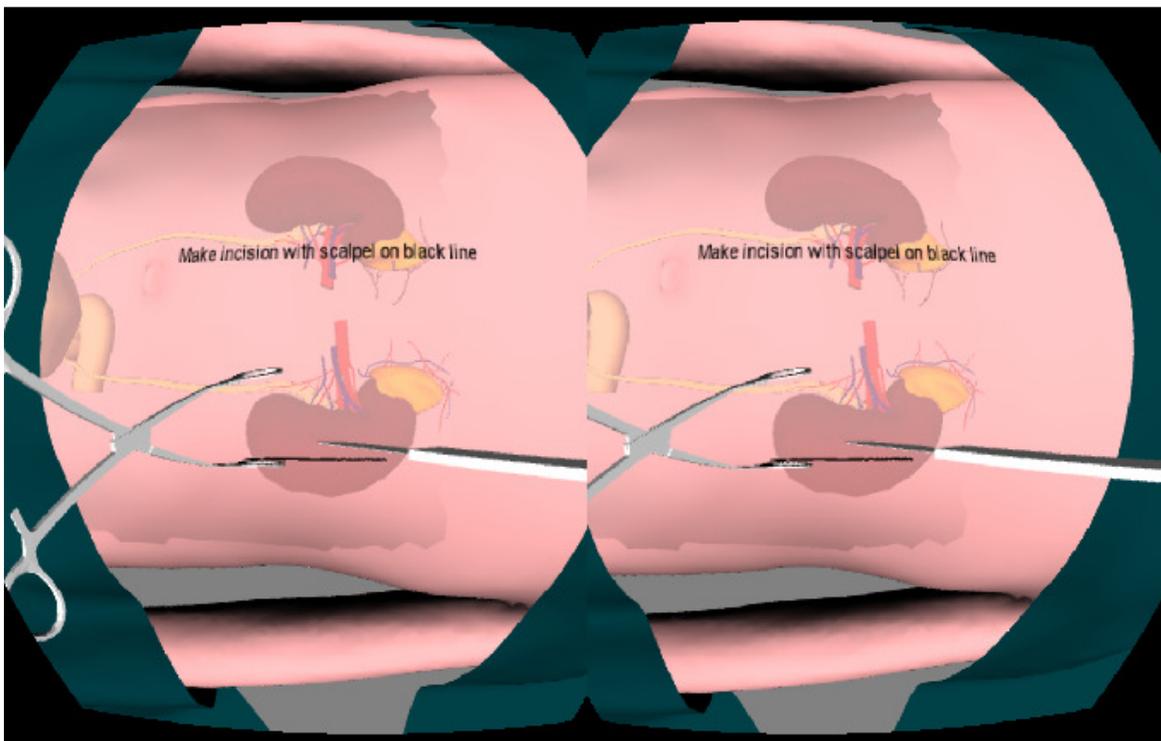


Fig. 1: A simple medical simulation created with IDV.

The simulation requires minimal programming to develop. Only a few lines of Processing code are necessary to control cutting the incision and grabbing the kidney.

The IDV can also be used for finite-element simulation and visualization in engineering applications. In fact, real-time finite element simulations can be programmed in Processing and displayed on the VizWall and Oculus Rift. Alternatively, Matlab can be used to generate finite-element simulations and the data can be saved as CSV point cloud file, and an optional CSV link file. The visualization can then, optionally, be programmatically manipulated and combined with other external data loaded by the XML file.

### **3.2 Music and Arts Applications**

In addition to visualization for science and engineering the IDV can be applied to visualizations for the arts and music. Meaningful and rich visualization can be created with the IDV by incorporating existing Processing libraries for the Xbox 360 or Windows Kinect sensor. The Kinect contain a depth sensor that can detect depths between 0.8 meters and 4 meters. In addition, to the Kinect library, Processing also includes the Minim audio library. Here we discuss application of the IDV to real-time dance visualization generation, and real-time visualization of music both in an immersive 3D environment in the Oculus Rift and synchronized with VizWall.

Existing platforms, such as Dance.Draw use wireless mice to track position of hands for real-time dance visualization generation. With the Kinect, rather than tracking hand position, we can now track motion of the entire body. In addition, Processing has easily accessible libraries for reading the Kinect. Thus, reading Kinect sensor data and creating dynamic programmatic data-driven dance visualizations becomes an easy task with the IDV. For example, one could load a terrain, and use the Processing language built-in particle system library to create fireworks that synchronize with the motions of the dancers. Alternatively, one could load a dataset as the basis for a particle system, and apply gravitational attraction to the limbs of the dancer, such that particles loosely follow the motions of the dancer (see Fig.2).

In addition to reading data from the Kinect, the Processing language can read and play data from an audio file. The Minim library allows users to take a Fast Fourier Transform of the data to create equalizer like visualizations. However, unlike standard 2D equalizer visualization, in the immersive environment of the Oculus Rift, the equalizer visualization can surround the user with rich visual experience. As the user turns, the image on the VizWall is synchronized and VizWall viewers see a high-resolution real-time rendering that is synchronized to the music in the background.

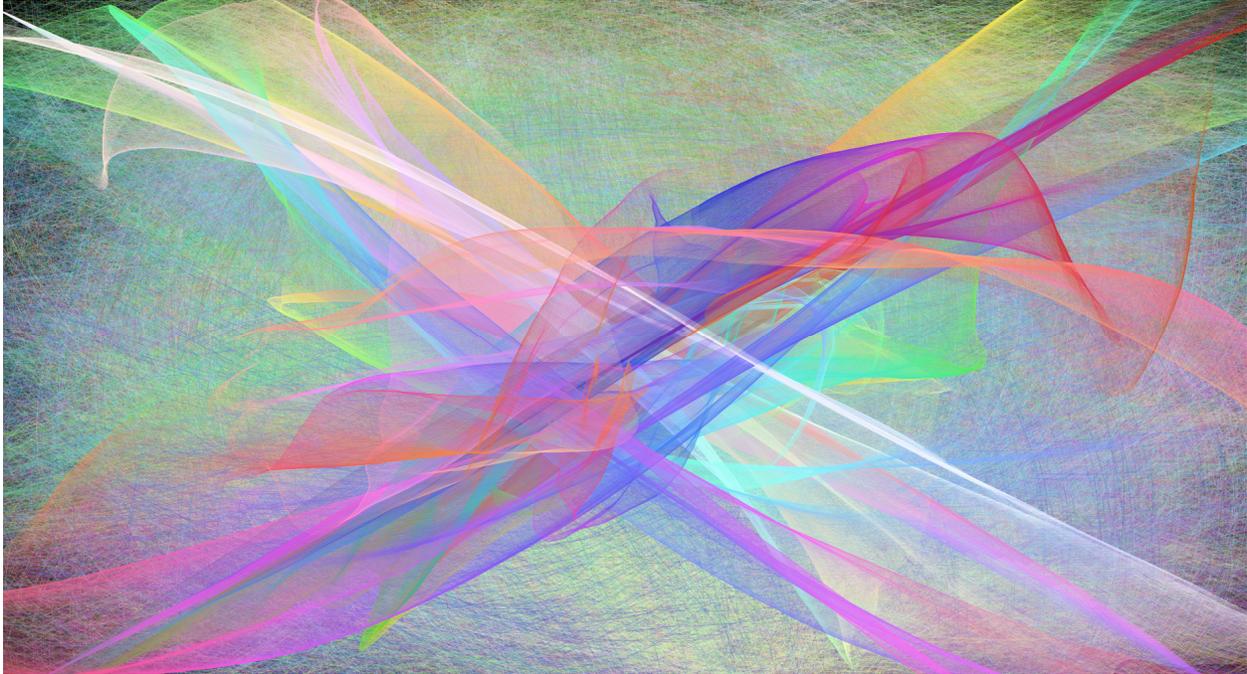


Fig. 2: An example of a Processing visualization that can be applied to dance.

### 3.3 Educational Impact

As teaching and learning tools, the advanced visualization technology has already provided successful applications in a few classroom examples, although more systematic implementation and evaluation have yet to be seen.

The most straightforward application of the high-resolution VizWall (Fig. 3) is for students to present their results on titled screens to compare, for example, various design options by zooming in details. It provides not only a global picture of the system but also fine details that single screen cannot provide.



Fig. 3: The high-resolution VizWall permits both global view as well high-definition details.

For topics in elementary engineering education, we have implemented a haptic-enabled cantilever beam so that students can “feel” the force and torque while observing the deformation and stress field.

In biomedical engineering applications, as mentioned previously, the haptic-enabled virtual reality model can provide digital representation of a patient for medical students to be trained in basic surgeries as well as emergency situations.

One of the most interesting implementations to students is a football kicking simulator, in which students have designed visualization tools to study football dynamics and body biomechanics []. This project has provided an interdisciplinary platform for students to mechanics, electronics, wireless communication, and computer science topics in the context of interactive visualization.

Finally, for music and arts, advanced visualization has provided unparalleled capability to perform concerts with vibrant visual effects, or make sculpture using virtual knife to cut before printing out their art work by a 3D printer, or as mentioned before, trace master dancer’s motion to teach novice to improve their movements.

#### **4. Conclusions**

The interactive visualization has many applications ranging from science and engineering to music and arts. A comprehensive framework including the Oculus Rift Virtual Reality headset provides an immersive 3D visualization environment for guiding and controlling simulations that other viewers can provide vivid images and dynamic videos on the VizWall as an effective education tool. In addition, the data-driven environment makes loading data from a variety of sources easy and intuitive. The data can contain reference to 3D models, animations, CSV point clouds, CSV link clouds, and CSV/XML topological data. Point clouds can contain addition metadata such as vectors, colors, scales, rotation, and OBJ model name or number. Data-driven visualization can be expanded with Fruchterman-Rheingold force-directed graph algorithm, and parameters of that expansion can be changed in the data-driven XML file. Furthermore, in addition to data-driven visualization, after the data is loaded, the user can easily program visualizations that dynamically modify point locations and links based.

As for development work, we will focus on the integration of the Matlab console directly with the IDV to provide a powerful mathematical framework for IDV visualizations. In addition, for arts and music applications, future work will focus on integration with Google Glass wearable computer with optical head-mounted display. Google Glass can provide the partially immersive environments that dancers and artists can use in real-time as a light-weight alternative to the Oculus Rift. Moreover, we will explore how the interactive visualization can be integrated in classroom setting as well as online learning.

#### **Acknowledgements**

This work is supported in part by the National Science Foundation grant (NSF/HRD-CREST#0932339), which is greatly appreciated and acknowledged.

## References

- [1] Gilbert, J.K., "Visualization: A metacognitive skill in science and science education", *Visualization in science education*, Springer, p. 9-27, 2005.
- [2] Hundhausen, C.D., S.A. Douglas, and J.T. Stasko, "A meta-study of algorithm visualization effectiveness," *Journal of Visual Languages & Computing*, 13(3):259-290, 2002.
- [3] Grissom, S., M.F. McNally, and T. Naps, "Algorithm visualization in CS education: comparing levels of student engagement", *Proceedings of the 2003 ACM symposium on Software visualization*, ACM, p. 87-94, 2003.
- [4] Naps, T.L., et al., "Exploring the role of visualization and engagement in computer science education", *ACM SIGCSE Bulletin*, ACM, 35(2):131-152, 2002.
- [5] Small, M. and M. Morton, "Research in College Science Teaching: Spatial Visualization Training Improves Performance in Organic Chemistry," *Journal of College Science Teaching*, 13(1):41-43, 1983.
- [6] Schönborn, K.J. and T.R. Anderson, "The importance of visual literacy in the education of biochemists\*," *Biochemistry and Molecular Biology Education*, 34(2):94-102, 2006.
- [7] Schönborn, K.J. and T.R. Anderson, "Bridging the educational research - teaching practice gap," *Biochemistry and Molecular Biology Education*, 38(5):347-354, 2010.
- [8] DeLano, W.L., "The PyMOL molecular graphics system," (2002).
- [9] Wu, H. and P. Shah, "Exploring visuospatial thinking in chemistry learning," *Science Education*, 88(3):465-492, 2004.
- [10] Baker, S. and L. Talley, "The relationship of visualization skills to achievements in freshman chemistry," *Journal of Chemical Education*, 49(11):775-776, 1972.
- [11] Wu, H.K., J.S. Krajcik, and E. Soloway, "Promoting understanding of chemical representations: Students' use of a visualization tool in the classroom," *Journal of research in science teaching*, 38(7):821-842, 2001.
- [12] Schmidt, J. and W. Polik, "WebMO, Version 12.0," WebMO, Holland, MI, (2012),
- [13] Limniou, M., D. Roberts, and N. Papadopoulos, "Full immersive virtual environment CAVE<sup>TM</sup> in chemistry education," *Computers & Education*, 51(2):583-593, 2008.
- [14] Johnson, A., et al., "GeoWall: stereoscopic visualization for geoscience research and education," *Computer Graphics and Applications, IEEE*, 26(6):10-14, 2006.

- [15] Silén, C., et al., "*Advanced 3D visualization in student-centred medical education*," *Medical teacher*, 30(5):e115-e124, 2008.
- [16] Warrick, P.A. and W.R.J. Funnell, "*A VRML-based anatomical visualization tool for medical education*," *Information Technology in Biomedicine, IEEE Transactions on Information Technology In Biomedicine*, 2(2):55-61, 1998.
- [17] Stout, A., E. Biglari, J. Kantor, C. Meyers, A. Schaeffbauer, and Y. Feng, "Quantitative training of American football kickers using real-time sensing and prediction," *Proceeding of International Workshop on Computer Science in Sports (IWCSS)*, p. 255 – 261, 2013.