# InfoView3D: A Solution Showing Educational Model on Multi-Touch Surfaces

Qi Ming[1] , Marius Erdt[1] , Chen Kan[1] , Eugene Lee[1] , Gerrit Voß[1] , Wolfgang Müller-Wittig [1]

[1]Fraunhofer IDM@NTU

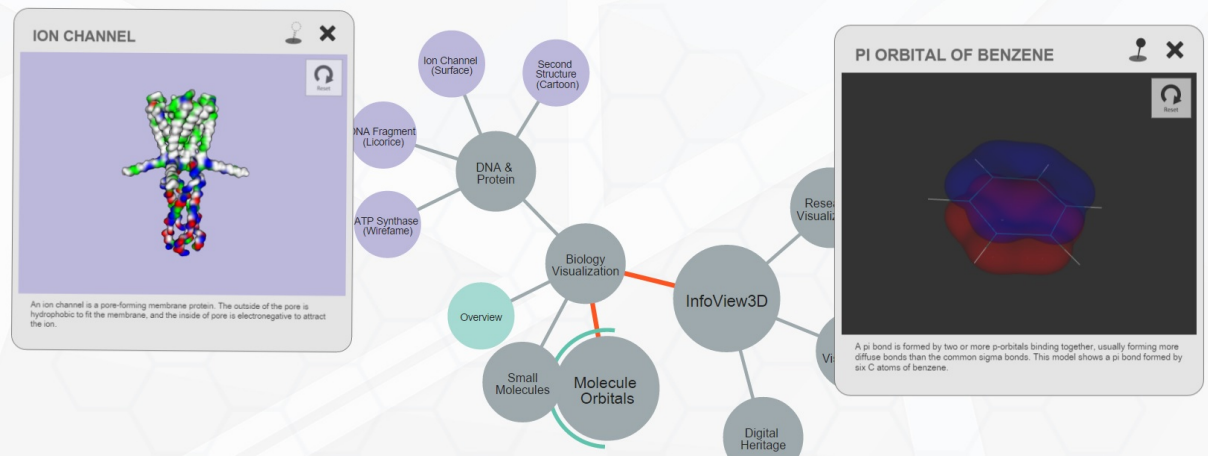


**Figure 1:** *A screen shot of InfoView3D user interface*

**Abstract**

*In this paper, we present a solution for educational presentation which can integrate images, documents, videos, and especially 3D models in a tree based structure. The solution is based on HTML5 and JavaScript, using SVG and X3D techniques and interacting with multi-touch events, therefore supporting multi-touch surfaces that can run a fully-functional browser. In practice, our solution is used for demonstrations both on tablet PCs and a multi-touch video wall.*

Categories and Subject Descriptors (according to ACM CCS): K.3.2 [Computer and Information Science Education]: Computer Science Education—

## 1. Introduction

Presentation software such as Microsoft PowerPoint slides and Keynote slides are radically serial and using an outline feature to show the slide structure. However, we may want to show a clear tree-based structure about the contents of the presentation. Furthermore, in traditional presentation solutions we can not show or interact with 3D models directly but have to open a separate application. Our project, InfoView3D, is meant to present a tree-based information organization and integrate various media, such as images, documents, videos, and especially 3d models. This solution can not only used for presentation, but also for user-exploring demonstrations or info querying applications.

## 2. Overview

Our implementation is a server/browser paradigm with dynamic web UI techniques. The server side organizes all the files and provides HTTP services. Any technique that can ac-



DOI: 10.2312/eged.20151026

cess the file system and process HTTP requests is qualified as the back end. Our implementation uses node.js [3] HTTP server for its convenience and flexibility. Three kinds of files are provided by the server: application functional files such as html, js and css files. These files implement the application UI and features; media files such as images, videos, documents, and 3d models; and dataset files, which are structured JSON files and contain structure information, links to locate media, and also some UI attributes.

Play mode or edit-enable mode is used separetly for presenting the media or editing the dataset and upload media. Users choose the dataset to launch with and the main UI of the play mode is shown in Figure 1. The main structure is a tree based graph. Non leaf nodes will open/close their children when touched/clicked. The leaf nodes are related to certain media and will pop-up a box when touched/clicked. The current node is highlighted by a decorative circle and also the path to the current node is highlighted. All the nodes and links are SVG shapes in a SVG element. These elements are managed by the D3.js [4] library and using force field to make a dynamical and lively layout. When pulling a node, the user just feels like pulling a rubber string. Multiple media boxes can be open at the same time and the boxes can be moved and scaled as well as being controlled by multiple users. This is especially useful when the application is running on a multi-touch video wall. In the edit mode, the user can edit the structure and attributes of the dataset such as create/delete nodes, editing the media and descriptions, and also upload media. Dataset can be built without any existing content/structure as a prerequisite.

## 3. Model Preparation

In our solution, we use the X3D [1] file format to save 3d models and deliver them to the front end. X3D is an XML protocol description of 3d shapes, texture, rendering options, and other attributes. The X3D format works seamlessly with HTML5 and JavaScript without any browser plug-in support. The X3D format is supported by almost all 3d modelling software and model conversion tools. For example, we can export an X3D model form Blender[5] and convert an COLLADA model to an X3D model using X3D-Edit.

In more specialized domains, a 3d model can be created from specified tool chains. For biological models such as a molecule model and a protein model for example, we have a tool chain to get the model from molecule data: First, we get the PDB molecule file from the online PDB database[7] or from other biological software. PDB is a text file that presents small molecules as well as large protein molecules by defining atoms and their positions. Then, VMD[6] visualizes the PDB format with different styles. Visualization parameters can be set and the visualized model can be exported as an X3D file. Also, an orbital molecule model can be created using the following tool chain: First, a molecule is created or edited in ChemOffice and exported as a GAMESS input file. Secondly, the molecule orbitals is computed in GAMESS[10], which is one of the mostly widely accepted tool to compute molecule orbitals. Thirdly, the GAMESS output is visualised in VMD where the orbital and parameters can be set. Afterwards the file can be exported as X3D. Figure 1shows the protein and orbitals models inside the media boxes.

## 4. Model Interaction and rendering

Inside the 3d model media box, a pin down button is used to distinguish between the box moving/zooming or model interacting. When the pin is unpinned, the user's gesture moves or zoomes the box and when the pin is pinned the user's gesture is navigating the model.

The model rendering is done by the browser using x3dom [2] with also external rendering such as OpenSG [8] [9]. Through web sockets communication, control information such as the view matrix are sent to the rendering process. The render window can be in another screen or even on another machine.

## 5. Conclusion and Future Work

Our solution is very flexible and suits many application scenarios. Systems based on this scheme already used for medical school and biology school of Nanyang Technology University. Future work will focus on the integration of more external applications into InfoView3D, so that we can show or control other applications from InfoView3D to make more diverse visualizations and user experience.

## Acknowledgement

This research was done for Fraunhofer IDM@NTU, which is funded by the National Research Foundation (NRF) and managed through the multi-agency Interactive & Digital Media Programme Office (IDMPO).

## References

[1] X3D technique: http://http://www.web3d.org/x3d/ 2

[2] x3dom Project: http://www.x3dom.org/ 2

[3] Node.js Project: http://nodejs.org/. 2

[4] D3.js Project: http://d3js.org/. 2

[5] Blender Project: http://www.blender.org/. 2

[6] VMD Project: http://www.ks.uiuc.edu/Research/vmd/. 2

[7] RCSB Protein Data Bank: http://www.rcsb.org/. 2

[8] OpenSG Project: http://www.opensg.org/. 2

[9] Voß, Gerrit, et al. "A multi-thread safe foundation for scene graphs and its extension to clusters." Proceedings of the Fourth Eurographics Workshop on Parallel Graphics and Visualization. Eurographics Association, 2002. 2

[10] GAMESS Project: http://www.msg.ameslab.gov/gamess/. 2