Gamification and Visualization of Sensor Data Analysis in Research Buildings

Jackson Stone
University of Tennessee at Chattanooga
Chattanooga, Tennessee
Jacksonastone@gmail.com

Jibonananda Sanyal, Charles Castello,
Joshua New
Oak Ridge National Laboratory
Oak Ridge, Tennessee

ABSTRACT

The use of video game elements in non-gaming systems, or "gamification", has the potential to transform data analysis. Our study focused on creating a web-based videogame that models two physical test buildings, each of which contains hundreds of sensors. After the application renders the models, the "player" can walk through the environments and interact with the virtual representations of the sensors inside. Rather than querying a database with textual commands and spreadsheets of data, the user can (virtually) walk up to a sensor and view its data graphically. The recent progress in gaming hardware, game design, and data processing can be leveraged for data science. The net result is a more engaging and intuitive platform for data analysis that has the potential to provide additional insight for complex experiments.

ABOUT THE AUTHORS

Jackson Stone was a student intern at Oak Ridge National Laboratory through the SULI program. While working at Oak Ridge he designed and developed the 3D application described in this paper. He's currently a computer science major at the University of Tennessee at Chattanooga working as a designer for Torch.

Jibonananda Sanyal is a staff scientist at the Oak Ridge National Laboratory. His current research focus includes massively parallel and scalable energy simulations on high-performance computing systems and subsequent big-data analysis, with a strong emphasis on advanced experimental design and uncertainty evaluation. He currently leads projects focused on sensor data management, provenance, and simulation engine development, as well contributes to several key projects focused on achieving the Department of Energy goals of significantly reducing our energy and carbon footprint.

Charles Castello is a Senior Electrical Engineer with Pacific Architects and Engineers at the NASA Houston facility. He is an expert in Energy Management and Control Systems with an emphasis in data analysis and sensor data validation.

Joshua New joined Oak Ridge National Laboratory's Building Technologies Research & Integration Center (BTRIC) in 2009 after completing his PhD in Computer Science from The University of Tennessee. He currently serves as subprogram manager for software tools involving websites, web services, databases, building simulation, supercomputing, and artificial intelligence algorithms for big data mining. He has lead more than 35 competitively-awarded projects involving software integration with large companies, simulation validation, theoretical advances in machine learning algorithms, and optimization techniques. He is an active member of ASHRAE and has over 65 peer-reviewed publications.

Gamification and Visualization of Sensor Data Analysis in Research Buildings

Jackson Stone
University of Tennessee at Chattanooga
Chattanooga, Tennessee
Jacksonastone@gmail.com

Jibonananda Sanyal, Charles Castello,
Joshua New
Oak Ridge National Laboratory
Oak Ridge, Tennessee

ABSTRACT

The use of video game elements in non-gaming systems, or "gamification", has the potential to transform data analysis. Our study focused on creating a web-based videogame that models two physical test buildings, each of which contains hundreds of sensors. After the application renders the models, the "player" can walk through the environments and interact with the virtual representations of the sensors inside. Rather than querying a database with textual commands and spreadsheets of data, the user can (virtually) walk up to a sensor and view its data graphically. The recent progress in gaming hardware, game design, and data processing can be leveraged for data science. The net result is a more engaging and intuitive platform for data analysis that has the potential to provide additional insight for complex experiments.

INTRODUCTION

Gamification, being the use of video game elements in non-gaming systems to improve user experience, is increasingly used as part of the educational process as well as in business. In business, gamification is used to improve employee performance and customer satisfaction by giving immediate feedback. Recent studies have shown that online customer interactions may increase 30-40% (Palmer, et al, 2012). It's also being used by many employers to train their employees more effectively, resulting in a decrease in the turnover rate (Rozenfeld, 2014). The research firm Gartner, projects that 70% of 2,000 global organizations depend on gamified applications for employee performance, health care, marketing, and training with 50% of corporate innovation being gamified (Gartner, 2011).

Oak Ridge National Laboratory's Building Technologies Research and Integration Center (BTRIC) has two reconfigurable commercial buildings deployed on Flexible Research Platforms (FRPs) (Hughes and Patrick, 2012) as illustrated in Figures 1 and 2. The first building is a single-story metal warehouse building referred to as FRP1. It was constructed in coordination with the Metal Building Manufacturer's Association to represent the most prevalent warehouse building in the U.S. building stock. The second is a two-story medium office building called FRP2. It was constructed in coordination with DOE's Building's Hub, a 10-county region around the Philadelphia Naval yard, to represent the most typical, late-70s era office building in that area. These buildings are equipped with emulated occupancy to simulate thermostat control, lighting, equipment usage, and human emulators emitting heat and moisture according to a set schedule in line with industry-standard benchmarks. A large number of sensors measure a number of variables such as temperature, relative humidity, power draw from miscellaneous plug loads, HVAC efficiencies, as well as heat flux across doors, windows, and walls. The two commercial buildings comprising the FRPs stream data at a 30-second resolution for a total of 1,071 channels for both buildings. The FRPs are also equipped with a weather station to record current conditions. The sensor data is collected using data-loggers from Campbell Scientific and are saved to a shared network location as well as a database for later analysis. A number of scientists conduct experiments in the FRPs, run simulations, and perform analysis with the data collected.

This paper describes how the gamification of this sensor data is utilized to enhance data analysis through grounding the data in a virtual 3D space. In this way, a three-dimensional spatial component is added to the data analysis process allowing a greater amount of interactivity on the part of the user by allowing them to navigate this 3D space and associate data values to physical locations in the facility space. This allows for immediate association of any values, anomalous or otherwise, to physical components in the FRPs.

Our objective was to create an analysis application that leverages the benefits of 3D game-like interactions while enabling analysis. Such an application benefits both the technical user, who needs to draw correlations between certain channels of data as well as the non-technical user, who requires less detailed knowledge of the types of processes within the FRPs. The application gives each class of users a visual metaphor to understand and explore the data rather than long tables of data whose meaning is not always readily apparent. It provides an enticing, less taxing environment for the researcher, while having an easy to learn, unimposing, and informative interface for the non-researcher. The breadth and complexity of the many data types being measured within the FRPs, combined with the spatial nature of sensor placement provide an ideal set-up for testing and utilizing a gamification approach for its analysis.

Unity 3D, a game development engine, was used for the development of the 3D game environment. The rest of the paper discusses the technical process of creating the gamified analysis tool involving database integration, creation of 3D renderings of the buildings, placing sensors in their corresponding virtual locations, streaming data in real time, creating an interface for interacting with sensors, securing proprietary data, and integrating it into an existing website. Conclusions and future work are then discussed.

(a) (b)

Figure 1. (a) Flexible Research Platform #1 – interior of the single-story Warehouse; (b) FRP #2 – Exterior of the two-story, medium office building

GAME ENVIRONMENT DESIGN

Overview of Game Creation

The process of creating the analysis game consists of several components. First, there was a need to create working virtual representations of the FRPs within Unity. Second, we needed to design a way for sensors to be generated and placed dynamically within the Unity application to account for changes along with an interface that would allow a user to interact with them intuitively. Third, each generated sensor's data had to be retrieved while maintaining the security of proprietary data in the database. Fourth, an analysis tool had to be implemented within the game to allow selection and retrieval to juxtaposition multiple channels of data for the user. Finally, the application was envisioned

to be integrated into an existing website for the Flexible Research Platforms as a new tool for analyzing experimental data.

Virtual models

One of the first hurdles was creating the 3D models that worked within Unity. Each facility presented its own challenges and therefore each required a different approach for creation.

For the warehouse, no prior 3D model existed so Unity's asset store was used to purchase a model similar to the existing building. It was scaled to fit the warehouse's dimensions (Figure 3) which substantially increased the model quality and saved us dozens of man hours. The model we used was not a perfect match to the original FRP1, but this was acceptable considering that the intent of the game was to give a sense of what was occurring in the FRPs. The model selected was close enough for immediate identification of the different components.

Figure 3. FRP1 to scale warehouse model that was purchased from Unity's asset store, with added lighting and air conditioning ducts.

In the Unity design environment, adding lighting is a simple dragging operation of light objects into the model where they are needed, and tweaking the brightness till it is realistic. Though the process was relatively simple, it required manual calibration to get something that matched qualitatively to the real-world experience. For collision and clipping control, Unity's built-in functionality was used with the imported model. However, the automated process was not always consistent so manual effort was required to double-check and edit such that everything behaved in the way that was physically realistic.

A 3D model of the two-story office was available as a Revit file created by other researchers at the center. This model was imported into Unity as illustrated in Figure 4. However, the planes comprising the walls of the model overlapped, causing the rendering engine to render choppy textures and creating disconcerting visual artifacts from multiple walls overlapping and rendered through one another. The building had two levels but the model did not have a stair case or doors. It also did not have any colliders. In Unity, a collider is a component of game objects that prevents them from intersecting by performing collision detection on every frame. Without these in place, a player would fall through the floor of the model when Unity's physics engine simulated gravity. Also, for the videogame environment, lighting objects need to be added for anything to be visible. Revit files either do not encode or do not retain these when imported into Unity, so they had to be added manually (Figure 5).

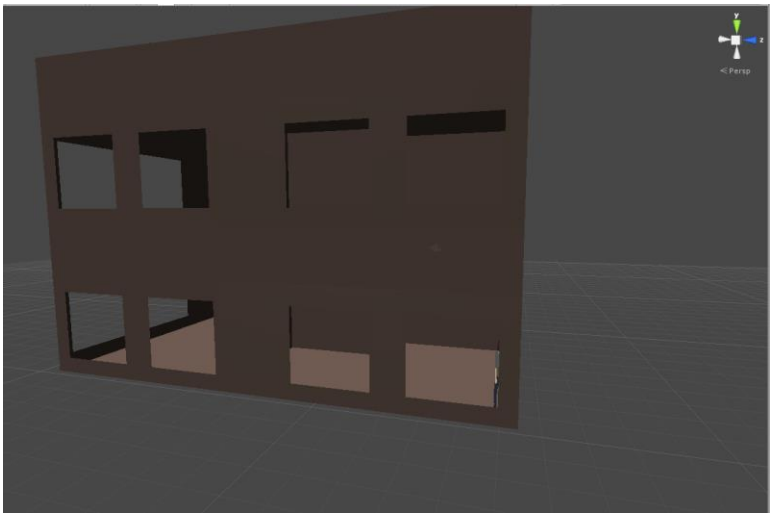

Figure 4: Revit file for FRP2 after import and a light object added. Due to format differences and file conversion losses, the walls are flat planes, have no textures applied to them, and have no colliders for physically-realistic interaction.

Figure 5. The final FRP2 model includes several aesthetic effects including wall thickness, texture, environmental lighting, and effects such as grass and a sky box surrounding the building.

In the instance of FRP2, a decision was made to give the player the ability to walk through interior walls by not giving them colliders and having the player jump up to get to the next floor, rather than taking the stairs. This allowed easier navigation within the virtual environment. When placing plane colliders in Unity, one can determine which side triggers a collision and prevents intersection. This allowed us to create one-way boundaries. This was done with the flooring of FRP2 in such a way that the ground kept players from falling through, while the ceiling did not keep them from going through. The jump height of the player was tweaked (see Figure 6) to allow them to jump from level to level. Not only did this save researchers time at getting to another point of interest, but it allows physically unrealistic environmental interaction which can be simultaneously efficient and pleasantly rewarding. Rather than rotating a 3D model to see it from all angles, they can walk or fly around it while easily moving from level to level.

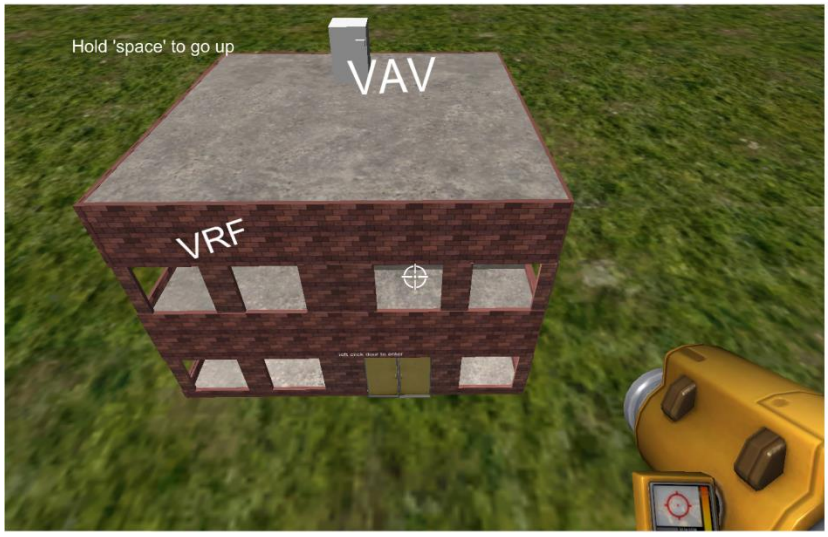

Figure 6. A player flying above the virtual FRP2

Once the models were created, we added a plane with a grass texture and a sky box along the perimeter. After that, the Unity's premade first-person character controller was dropped into the scene and we were able to walk through our buildings. This gave us the typical computer game control options for mouse and keyboard-based turning and directional movement.

SENSORS IN THE VIRUAL ENVIRONMENT

Sensor Placement

Adding sensors and their locations to the virtual models was a critical step. This presented two major challenges. The first was to dynamically place sensors within the model so that our model could change as the physical facility changed, especially in relation to security level changes for different sensors. The second was finding an intuitive way for the user to interact with the diverse types of sensors throughout the building.

In the FRPs sensors are frequently replaced, added, or removed. This becomes increasingly common as more sensors become uncalibrated, fail, drift, or otherwise report undesirable data. Also, it is a living laboratory where experimental envelopes and equipment are frequently changed. Due to the temporal nature of the FRPs' composition, we elected to dynamically allocate virtual sensors based on data about the real physical sensors to mitigate the need to manually change properties in the Unity Editor every time something changed.

To accomplish this dynamic generation of sensors we used a database to store physically measured x, y, and z values for each sensor's location and a corresponding column ID to the original database which stores the sensor data. As sensors are moved, added, and removed, only the database storing the sensor location and ID requires updating.

A user selects the table whose sensors they want to view and the time period of data they want to query through a front page within the game engine which was tied to the main database. Once selected, the web application game would use Unity's WWWForm syntax to post to a PHP script we had stored on the same database that contained the x, y, z, and the column ID of the sensor's data in the database. The PHP script would then call the appropriate MySQL query, and output the result in an HTML page. The game then read the HTML page, storing the coordinates and IDs, and created sensor objects in the corresponding locations (Figure 7 and 8).

Some sensors record data from proprietary equipment and are covered through Non-Disclosure Agreements, so password protected login and proper user-based access controls is used. This feature filters the list of sensors

allowing only the permitted ones for a user to be fetched from the server. The previously mentioned PHP script contained a login SESSION variable that contained a string indicating security clearance level. It would then cross-reference their security level with that of each sensor. If the user's security level was greater than or equal to what was required, then the x, y, z and column name would be returned. Otherwise, it would skip that sensor and not output its information, preventing the game from rendering it.

Figure 7: Sensor layouts for a database table

Figure 8: Sensor layout for a table with fewer sensors

Sensor Interface

Between the two FRPs there are over a thousand sensors embedded in the walls, ceiling, air vents, air-conditioning units, and circuitry. Many are imperceptible; they often overlap, are packaged into the same envelopes, and are sometimes integrated into the circuitry itself. The diversity of sensor setups led to a complication that had to overcome when deciding how best the player could access the data of these sensors and how to visualize them in the game.

Sensors are rendered as separate white cubes in the 3D models. Once the game had read the x, y, and z coordinates, it would create a sensor game object, the white cube, in the corresponding locations in the model. Once this was done the next step was determining how players would select a desired sensor. Because the mouse was used to turn the player's head, when a mouse click was implemented to select a sensor, it often caused a slight movement causing the sensor to move away from under the cursor. The 3D nature of the environment made this more acute the further the sensor was located from the player. To alleviate this, a first-person shooter game affordance was implemented giving the player a gun that they could aim and shoot at sensors. This not only gave an easier to use interface but also gave it a more stimulating experience within the FRP models.

While this worked for most sensors, a problem arose when many sensors were within close physical proximity. Many sensors are embedded within the building envelope, overlapping, or within an HVAC unit, so a system needed to be put into place to allow a user to select sensors from a grouping. Instead of implementing an additional level of complexity with database assignment of groups for each sensor, a minimum distance was set between sensors where two sensors would automatically be assigned a group, envelope, or equipment if they were within the minimum distance to one another. If a sensor that was part of an envelope that was part of an east-facing wall, there would be a drop down menu that would allow a player to see all other sensors grouped into the same envelope. From there, they could select any one the sensors in that envelope. So the gun became a means of selecting both individual sensors or a grouping of sensors.

Database Interaction

Once an operational and intuitive interface was constructed, we connected the sensor IDs to the actual sensor data using a combination of the Unity specific C# library, PHP, HTML and MySQL (see Figure 9). The data setup required us to run our game came from two different servers. One contained x, y, z locations, descriptions, security levels, and column IDs that were identical in both servers. The second server contained the data that streamed in from the sensors themselves, as well as all other metadata, such as units of measurement, sensor health, name of the building in which it was located, and a matching column ID. Security requirements compelled us to use two servers. In the discussion below, we'll refer to the database that contains the sensor data as S, and the second server that contains the information necessary to place sensors in the application as A.

The system we used for data retrieval was largely based on how game designers create high score boards in Unity-based games. Unity has two classes, WWW and WWWForm, which allow a programmer to post data to a PHP script stored on a server while in the application. The PHP scripts on our servers receive this post, and based on the information in the post, then call a corresponding SQL query to get the desired data. It then outputs the data as an HTML page. The WWW and WWWForm classes can then be used to read data from the output HTML page.

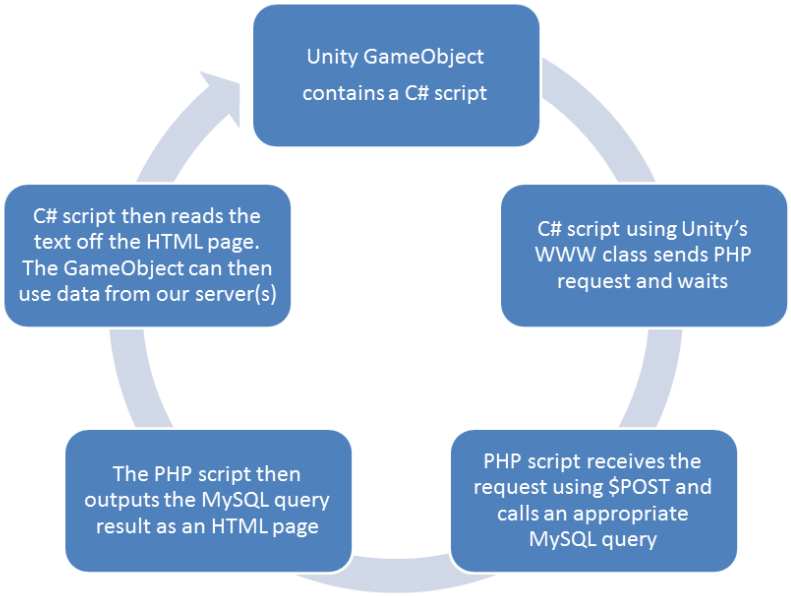

Figure 9. A system diagram of the interaction feedback loop for our application.

Our game goes through five stages when starting up in order to place sensors dynamically in the models, the last four of which utilize the above technique. The first is logging in to verify the user has the appropriate credentials to view the data. The second is getting a list of all the tables in database S so that the user may pick which one they want to view. Once they have selected their table, the third step is to get a list of all Column IDs in that table from S. The fourth step is using the retrieved IDs to find all the appropriate rows in A, retrieving all needed location information. It is at this stage that the PHP scripts check security levels, making sure no proprietary sensor data will be retrieved if the user doesn't have the clearance. If the security level of the user is less than that of the sensor, no information of the sensor is sent to the HTML page that is scanned by the game. The last step is then pulling the sensor data from the selected table for the timespan for each sensor.

The application then creates all dynamically placed sensors throughout the models, all of which contain the data from their physical counterparts. The sensors are also put into their groups.

INTERACTIVE ANALYSIS AND VISUALIZATION

Unity, being a development engine most commonly used to produce videogames, was not created with data analysis in mind. For example, Unity has no basic, out-of-the-box graphing functionality. There were add-ons on Unity's asset store that could have covered basic graphing situations, but the unconventional, multi-server database setup proved challenging for these solutions. Instead, we elected to create our own graphing utility in Unity from the ground up.

The application required the capability to compare data sets on the same graph in order to preform basic analysis. To accomplish this, an interface was designed that allows the user to dynamically create custom groups and add or remove sensors from a group. To compare two or more sensors to each other, the user could add them to the same group. If they then selected that group, all the sensors in that group would have their retrieved data plotted, each sensor having a unique color (see Figure 10) by sending proper directives to Unity's particle engine.

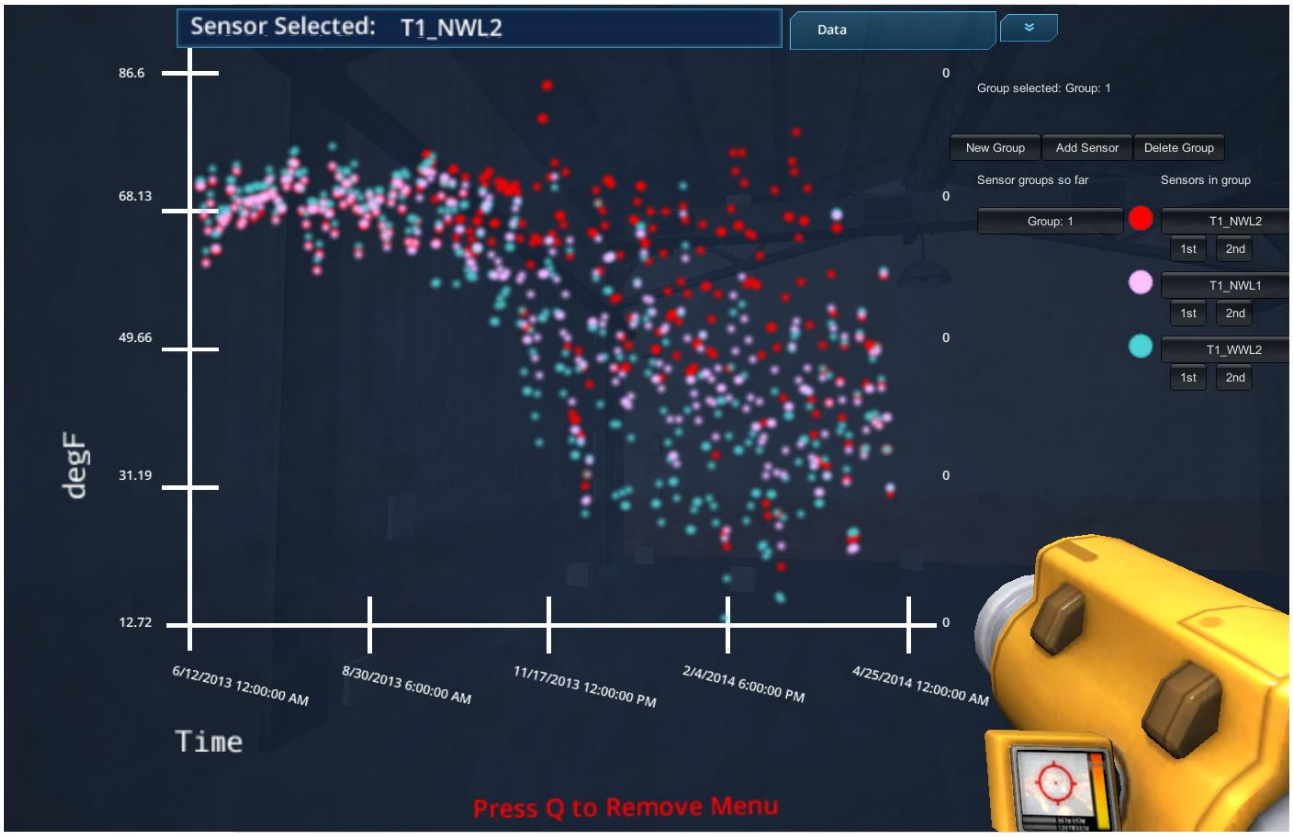

Figure 10: Groups of individually-colored sensors can be graphically displayed in-game.

In database S, there are over two dozen different data types, so there was a need to handle multiple combinations of axis types. In addition, some of the data fields are unitless, so there became a need to label and manually assign which sensors' data appeared on the axes. Another issue was that many sensors shared a single data stream with another column indicating which sensor was sending data at a given time. The graph generated had to be dependent on multiple sets of data. To deal with all of these, we needed a high level of customizability with our graphing utility which we were able to achieve by leveraging Unity's built in particle system.

This approach proved a simple and adequate solution to the problem but also suffers from some specific limitations. In Unity, particles are not optimized for graphical use. Each particle contains a host of variables that are commonly

utilized in videogame settings such as particle momentum, rotation, linear drag, angular drag, shade, color, speed, etc., that are not meant for creating basic points on static graphs. The large demand made by large numbers of particles began to lag an average strength desktop computer when rendering beyond approximately 10,000 points on a graph. However, when under this value, the game operates smoothly and 10,000 points is often enough resolution for most needs. The main startup page was created to inform the user of this limitation and guide them into selecting a manageable set of data for exploration.

SOFTWARE INTEGRATION

The entire Unity interface was embedded into a previously developed application named ProvDMS (Zachary 2014), a provenance application for tracking data use. This provided a common authentication mechanism as well as retrieval of the user's access privileges to display sensor data.

Upon login, the player typically steps through the game as follows: the player is presented with a list of all tables and two scroll bars, one for selecting the start date and end date for data of interest. Below that is an estimation of the number of data points in between is used to inform the player whether they may experience lag. Once finished, they click "GO" (Figure 11).

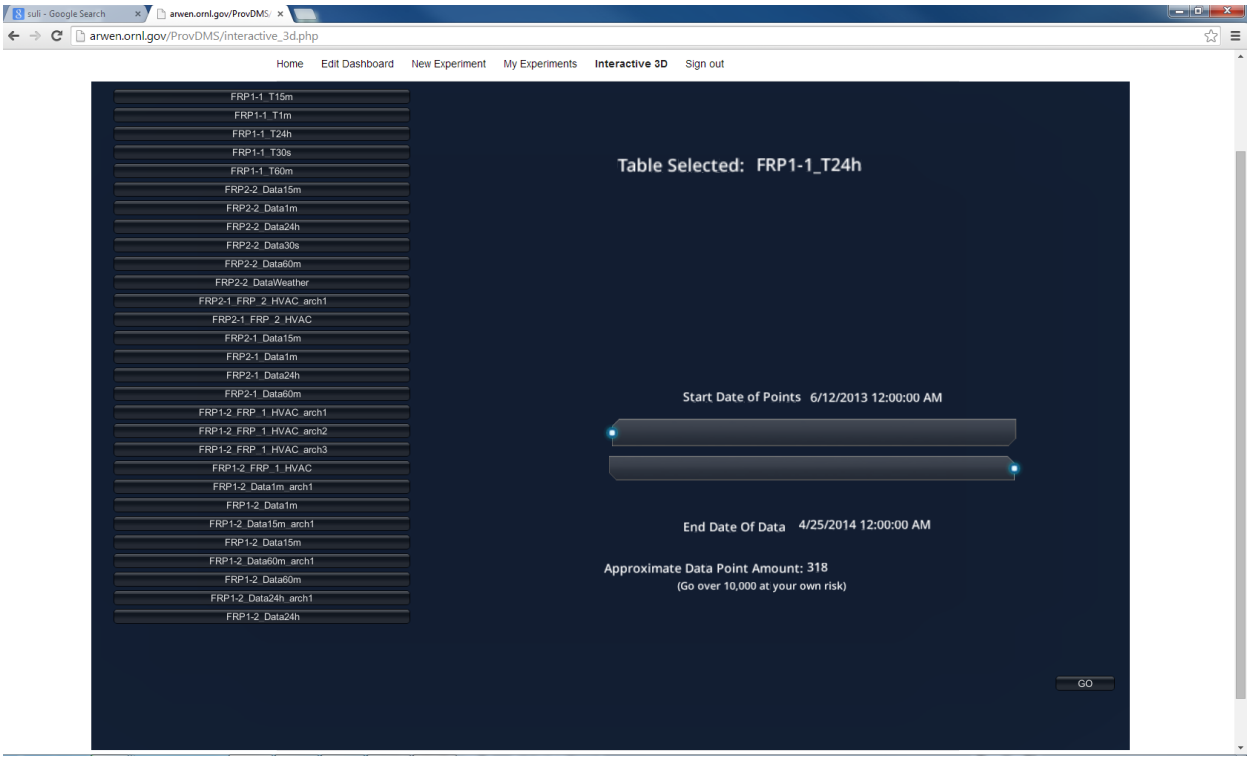

Figure 11. A screenshot of the main menu of our application inside a web browser.

Based on the table selected, either the FRP1 or FRP2 model is rendered, with the appropriate sensors in place. Certain sensors only send data to certain tables, so based on which table is selected, different sensor layouts may render. The player can then walk through the facility with traditional W, A, S, D controls, jump with the space bar, turn their head and aim with the mouse, and shoot their gun by clicking. When a sensor is hit, a plot of its data is generated as a heads-up display. From there, the user can exit to select another sensor, add the selected sensor to a group, view all other sensors in the same envelope, or view an image of the physical sensor if one is on our server.

The end result is an intuitive and informative application that can be utilized for data analysis in a way that incorporates the spatial relations between sensors, so rather than only comparing sensor to sensor, the analyst can compare location to location.

CONCLUSION

This application illustrates a method for leveraging gains made in game design techniques for the purpose of scientific analysis. The project required approximately 500 man hours by a sophomore undergraduate, but includes the time spent learning the software from scratch, decoding the building data to construct models, and populating a second database with measurements of physical sensor locations.

This more enticing and user-friendly experience is immediately approachable and has the potential to raise the bar of what analysis tools can do. A non-technical user, such as a potential investor who may only need a general picture of what the project accomplishes, can immediately step in with minimal training and navigate the 3D analysis tool, and actually see the data stream in, and where it is streaming from. This provides a shallower learning curve to navigate the massive database as well as providing a more engaging environment. Technical users on the other hand can see where a malfunctioning sensor is located in a building and communicate that to a repair team. They can also recognize anomalies in data that are spatial in nature, such as poor insulation in a section of one of the FRPs. For this reason, any data, that may have relevant spatial factors about where the data is coming from would benefit from this type of 3D analysis tool because it is able to concisely communicate that information without adding another set of numbers to the display.

Through unconventional projects like these the more main stream analysis techniques can field-test proof of features worth adopting. Not only does it make analysis a much more welcoming and inclusive, but it's more fun, keeping users at the application longer. This type of gamification has the potential to spur new interfaces and new methods of communication between the technical and non-technical, lessening the communication gap by making the application feel more approachable.

One of the most obvious advantages to the approach demonstrated is the non-technical requirements for operating the system. It does not require a user to have memorized SQL syntax or to be overwhelmed with screens full of data. Instead they click on a table, drag a couple sliders, then move and shoot like in any other first-person computer game. With 97% of American youth playing video games, analysis tools can help transition toward long-term educational benefits and the potential to spark a child's interest in analysis. Gamified educational applications can function as a bridge between the technical professional and a non-technical audience. Often the two must work together, but due to different skill sets there is frequent miscommunication. A way to assist in communication is to create an application that they can both use by lowering the learning curve, but maintaining utility. Gamified tools provide a possible solution. With today's large amounts of data, visualization is a necessary approach to understand our data. Visualization, with gamification can provide a newer, more enticing packaging and present that data in a way that is consistently more engaging and productive.

ACKNOWLEDGEMENTS

This manuscript has been authored by UT-Battelle, LLC under Contract No. DE-AC05-00OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).

REFERENCES

Gartner, "Gartner says by 2015, more than 50 percent of organizations that manage innovation processes will gamify those processes" [Online], Available: http://www.gartner.com/newsroom/id/1629214, April 12, 2011.

Hughes, Patrick, "Light Commercial Building Flexible Research Platforms", ORNL internal report ORNL/TM-2012/143, 2012, 96 pages. Available: http://btric.ornl.gov/publications/ORNL%20Report_Light%20Commercial%20Building%20Flexible%20Research%20Platforms.pdf

M. Rozenfeld. (2014, May 7). *Gaming in the Workplace, How mixing work and play can motivate employees*[Online]. Available: http://theinstitute.ieee.org/technology-focus/technology-topic/gaming-in-the-workplace

Palmer, Doug, Lunceford, Steve, and Patton, Aaron J, "The Engagement Economy: How gamification is reshaping businesses", 2012. Available: http://www.deloitte.com/view/en_US/us/Insights/Browse-by-Content-Type/deloitte-review/c7cee86d96498310VgnVCM1000001956f00aRCRD.htm

Zachary Hensley, Jibonananda Sanyal, and Joshua New. 2014. Provenance in sensor data management. Commun. ACM 57, 2 (February 2014), 55-62. DOI=10.1145/2556647.2556657 http://doi.acm.org/10.1145/2556647.2556657