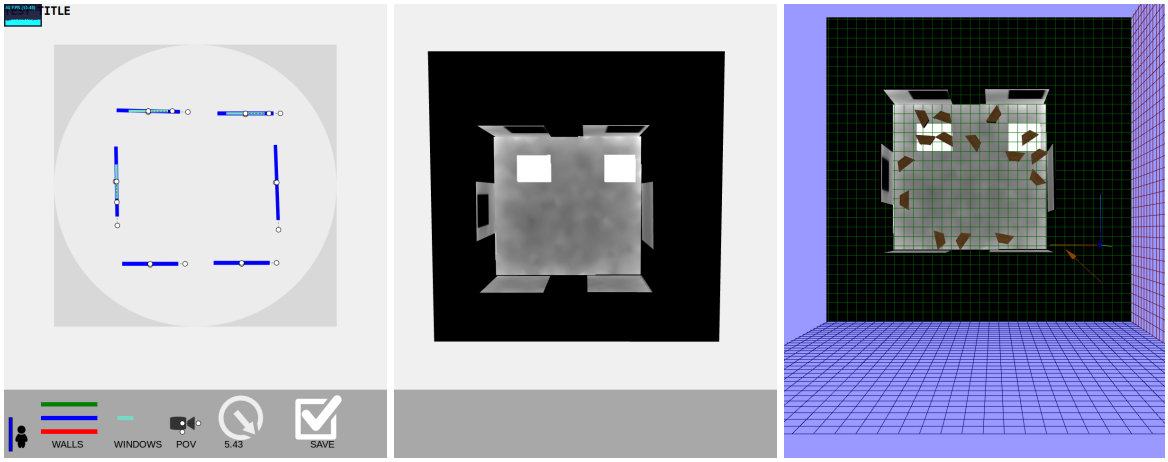
A Framework for Furniture Visualization in WebGL: A User Study Based Extension

Max Espinoza

Rensselaer Polytechnic Institute
espinm2@rpi.edu

Abstract

As an extension of current research done in architectural day-lighting simulation I provide a framework for the visualization furniture in user sketched spaced in WebGL for ease of use and accessibility. We also provide a naive furniture optimization using a simulated annealing approach. Using a texture output for lighting values of a given space, a simplified cost function, and a specified number of furniture items we find an approximate arrangement in an arbitrary shaped room. The framework provided, can be extended to use a Metropolis-Hasting state-search step to solve the optimal furniture arrangement for light, and other ergonomic properties.



Above is shown the pipeline of the online interface. We start with a sketching interface where you can draw wall primitives and then save them into the server, in which daylighting simulation is computed. A visualization is shown on middle image is the resulting scene with textures on a 3D model. On the right is the furniture visualization after running our naive optimization for 20 iterations.

I. Background and Related Works

Research done on the Virtual Heliodon [4] offers a tangible user interface to create and build room layouts and display accurate illumination of natural lighting back onto user created physical sketches through virtual reality augmentation. The process of sketching spaces and displaying daylighting information back on the scene is faster and come with less overhead then traditional methods. Motivation behind this research is to offer architects the ability to consider use of natural lighting earlier in their design phase in order to save energy in lighting cost. Furthermore architects with tangible user interfaces such as the Virtual Heliodon can collaborate and share ideas within a common space and get interactive feedback on changes to physical sketches.

User evaluations on the Virtual Heliodon [3] have been done and have offered us key insight into how the tangible user interface is used, in addition to

possible extensions that would aid users in understanding daylighting in interior spaces. The evaluations discovered that both novice and architects alike didn't have a firm grasps on natural lighting contribution and penetration into a room. The study also revealed that users had difficulties detecting where problem areas with glare would occur. This lead to further research in glare visualization tokens which offered a solution this problem. Also the user studied revealed that adding furniture into the Virtual Heliodon might further provided a better understand of problem areas caused by over illumination. Overall the benefit of the user study is that it give user feedback that can be used to refine and improve research done on this project.

While the previous study offered insight into further extensions and problems with the system, more user feedback is required to gain a better understanding of what direction to take future research. In addition increased feedback offers confirmation that solutions to previous problem have been successfully fixed. Informative feedback requires a planned user study and scheduling participants to come physically interact with the Virtual Heliodon. This limits our number of users who we can test in a given amount of time. However, some parts of the system can be studied in a wide spread manner by offering a simplified online alternative that allows the creation and exploration of room layouts.

For this project our goal is to develop the framework and extension to expand our user studies to a much larger number of participants. We want to make a easy to navigate and use online application that will aid in our collection of user sketches and feedback about daylighting in architectural room designs. Borrowing from previous research done throughout the semester on this online design tool, I extend it to offer furniture visualizations and an optimization.

The project is an exploration into what can be done using the online interface and daylighting information generated by Virtual Heliodon. Just as ARmy [1] was an exploration into the application of the physical system, this is an exploration of what can be done with the daylighting output generated by the Virtual Heliodon.

Also the incorporation of furniture and furniture optimization was done as extra feature that would motivate users to create and edit more wall models based on results generated by key algorithms in the Virtual Heliodon. Similar to Real-time Drawing Assistance Through Crowdsourcing [**?**] application done by Limpaecher at the Mircosoft research group. An iPhone application was used to collect the minimal amount of lines needed to draw a recognizable face, by designing their data collection in the form of a iPhone game they got a large number participants to contribute data for research purposes.

Lastly, it was a learning experience and deviation from OpenGL used in class to what was possible given current online libraries and tools in conductions with WebGL. It would give the research team an idea of continuing this simplified online approach was worthwhile, and offer us an existing extendible framework to continue development.

II. Methods

Our criteria for creation of the successful framework was easy integration into the current online system we provided, extendability to add more features, and fast enough to run on modern hardware. There were other choices considered, and choices that resulted in bottlenecks that made the furniture visualization non-interactive.

I. WebGL

Given that we needed to reach a wider amount of people for this user study there were alternatives to a website considered for development of this furniture optimization. Java was considered for it ability to use OpenGL and portability, as well as C++ which could have much easily been extended since a stable current framework for OpenGL was provided for us in homework assignments. However both of these approaches require additional installation and would discourage users from participating in the user study. Despite choices that offer better performance , WebGL offered a solution that works

in modern web browsers. WebGL can make use of GPUs specialization for rending computer graphics. Modern browsers like Firefox and Google Chrome have become ubiquitous giving us a much wider audience who can participate. In addition, libraries made to simplified usage of WebGL such as Three [2] can increase the speed of development of features to motivate users to participate. Also WebGL like OpenGL allow, can be used to write shader programs that can increase performance. Lastly, being online hosted our website would allow the creation of a database to store all user created floor designs and feedback. There were downsides to working in WebGL however, it generally did not perform with the same frame rates as OpenGL, our unfamiliarity with web development and WebGL, and large bottlenecks in JavaScript programing that needed either optimization or moved server-side.

II. Full WebGL and Javascript Based Appoach

The choice to begun development entirely in Javascript and WebGL was based on visualization demos that used the Three JavaScript library. While Three [2] offered a great set of tools to set up an initial scene, including a room, camera, and ambient lighting, it failed to fully support the set of features within it's documentation. Three was a tool for basic scenes but extending it handle much more complex interactions proved to be a difficult due to lack of documentation.

For example a lot of effort was spent on getting average texture from triangles underneath each desk in the scene. Three library had consistently missing documentation, and functions that were not implemented entirely that caused issues with this WebGL and Javscript based approach. Using Three I managed to raycast from the center of each desk item downwards into the triangles beneath it. However I quickly ran into a problem with how Three handles meshes loaded in from OBJ files. It offered no way to get the textures pixels or iterate through them within WebGL. Instead I got the UV texture coordinates and converted it absolute texture coordinates, created a separate html canvas used that to query pixels in the floor texture.

There was a massive draw back to this approach. Despite running with over 60 fps on high end performance computers, it wouldn't run smoothly on a modern laptops, such as mid-range performance machines such as Thinkpad laptops. This approach as then discarded since it would be a barrier for entry for participants in the user study. Code in Javascript proved to be costly in that it had to run all calculations on user web browsers which created a large bottleneck.

III. Server Side and Texture Based Approach

An alternative considered afterwords was a server side based approach in that we would use a separate program on the server to run the optimization and decide where furniture should go by absolute position relative to the floor texture. Communication between these offline scripts and our visualization in WebGL would occur via state files, which save the position and angle of each item in the scene. Server-side these scripts can create state files that will update the online visualization as to where each desk at specific iterations of an optimization function. Just as in INSERT PAPER TITLE HERE we can reduce the optimization into two dimensional problem, since items are assumed to be supported by the room's floor. Since we are given a room's floor texture in which indoor spaces with lighting are represented with white pixels of varying brightness, and black for spaces with no light and spaces outside the room are shown by pure black pixels on the texture. The texture being a 2D representation of the room was chosen over the 3D representation in that texture look up is fast and provided us the lighting values we could use for finding the interior spaces and where to place our furniture intuitively.

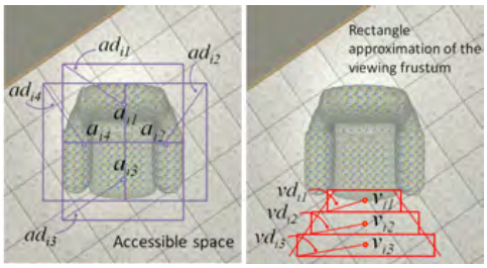
On the other hand using only the floor texture resulted in problems finding a furniture angles relative to the arbitrary space. Since in the texture representation we loose the precise points a wall begins and end, it makes finding the angle difficult through

standard graphics methods. In addition the texture itself is limited in the 512 x 512 pixels used to represent the floor. This leads to sampling issues if I were to use 2D ray cast to find values needed for the optimization and cost function. While in the paper we base this project on they use basic square rooms, we are given user created room of arbitrary shape that can support dividing walls and multiple room designs with use of user created wall files.

Despite setbacks the server-side approach will probably be continued and extended such that the online visualization of furniture arrangements is more then just a visualization offering much more information and manual editing for furniture arrangement.

IV. Framework

The WebGL and server-side framework works through state files, in which we use to store relevant information that needs to be shared between server-side scripts and the online visualization. Currently the framework only supports desk, but finding and loading other OBJ models to work is straight forward and can be easily extended. Manual intervention however is required for the optimization, it is important we know which side of an furniture item's bounding box is considered its backside. The Backside is defined as the side of each furniture item opposite of where person would sit or use that item. Like stated previously our current framework saves the furniture item number requested from the online visualization to state file, that an offline script reads and generated several output state files. Each representing a frame / iteration of a simulated annealing method to find lighting maxima for a specific number of desk.



Left: Bounding boxes that define accessible space. Right: Boxes that define where an item is viewed from.

V. Optimization

Our optimization was based on the simulated annealing algorithm, presented in Make it Home [5] paper, to find best use of lighting in a given space. Plans were made to take into consideration much more then just lighting conditions through a cost function that penalized unergonomic furniture arrangements. I planed to use some of the ergonomic values in presented in the Make it Home paper such as accessible space and viewing frustum. The algorithm considers items based on their position p_i and their Θ_i which represents the angle between a ray from back side of an object and ray to nearest wall. The ability to get ray to nearest well proved to be difficult using only textures and hindered by ability to implement the their cost function, however with use of other data provided in the system this angle can be found.

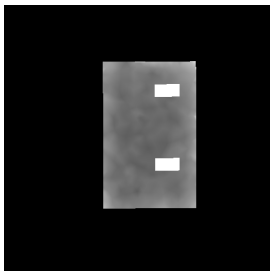
Accessible space is defined by four 2D boxes around an object that define the space needed to navigate around that object. For example in the figure provided accessible space around a sofa would be need to wide enough for a person to fit through without trouble. The cost function for the accessible space is defined below. It punishes items that enter the accessible space of another object. If object i overlaps the accessible space boxes k of object j the cost is calculated as bellow. Where b_i is the diagonal of the bounding box around item i.

$$C_a(\Theta) = \sum_i \sum_j \sum_k max\left[0, 1 - \frac{||p_i - a_{jk}||}{b_i + ad_{jk}}\right]$$

Another element from the Make it Home paper we wanted to implement was the viewing frustum. It is a series of rectangles defined in front of an object that are there to define what space shouldn't be taken up by another object. Like accessibility however with viewing in mind. Such as a monitor has a viewing frustum directly in front of it and should be taken up by another desk. It cost is also calculated identically, however each triangle that makes

up the viewing frustum would have decreasing overall penalty, for example items really close with in the smallest viewing frustum bound would be penalized at more then those that intersect the outer viewing bounds.

$$C_v(\Theta) = \sum_i \sum_j \sum_k max\left[0, 1 - \frac{||p_i - v_{jk}||}{b_i + vd_{jk}}\right]$$



This is a floor texture generated by algorithms in the Virtual Heliodon. We use it as a texture to find interior and exterior spaces.

Daylighting was the most important element I wanted to incorporate into the cost function and was implemented for the live demo. To make the best use of natural lighting within a given space I approximated light received by a surface as the area of their bounding box in 2D. While this doesn't take into account occlusion between objects, it wouldn't effect low laying furniture such as desks, that are typically low to the ground standing only a few feet high assuming desk didn't cluster too closely. I wanted to penalize any desk that were laying in complete darkness by a value large enough value that would discourage any solution that considered such an arrangement. This was a naive greedy solution for handle daylighting and is intended to replaced with a more sophisticated cost function that will take into consideration more then just bounding box of an item and sum of pixels underneath.

The overall cost function is computed as a sum of these ergonomic other cost function, with weights multiplied to choose which of these properties are more important. It is important to note that my implementing only takes into consideration lighting and a bounding box cost that incorporates accessibility by making a bounding box larger on either the sitting side of the desk and side opposite of that one considered the backside.

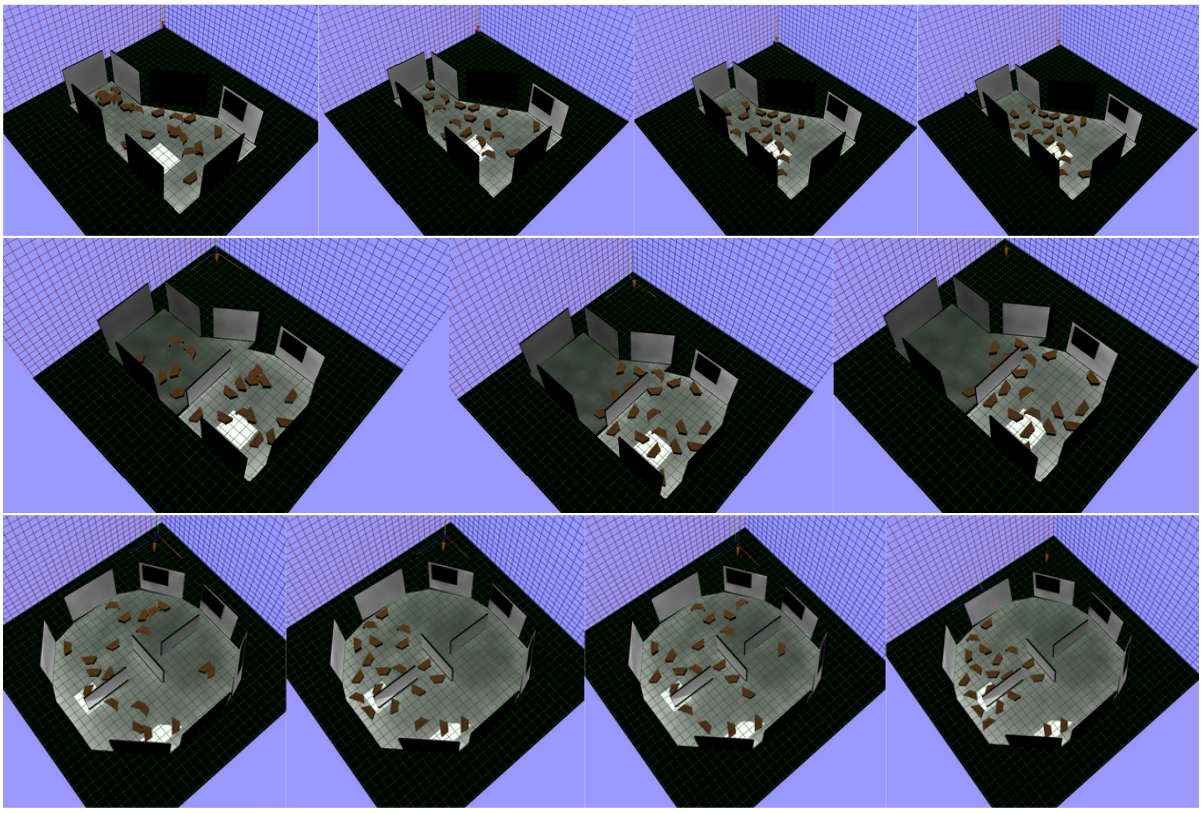
The solution was also implemented as described by the Make it Home paper, the simulated annealing was used to move furniture items between around a given room. This can be best described as moving molecules in a heated medium. As that medium cools the amount the particles move is reduced until the medium loses all heat. In this case we simulate heat by a heat factor, which is set low for our implementation. We then simulate particles moving as the desk moving around within the given space, as long as the desk movement doesn't push it outside the interior region and the cost function of the new arrangement is lower we move that desk. The Make it Home paper describes this same method with the addition of the swaps, in which two items are swapped if lower cost is achieved, however because we only have a single furniture item swaps wouldn't alter the results. We plan implementing swaps after we consider different furniture items.

VI. Challenges

As stated previously there are many challenges in this exploration away from traditional OpenGL programming. With WebGL there is the additional bottleneck of JavaScript programming. Inefficient code can cause large reductions in the frame rates of the visualization. The current implementation has been optimized to run on the RPI Laptop under Google Chrome and Firefox with reasonable frame rates. Previous attempts had such large bottlenecks that made it unusable to those without high end machines.

Besides the challenge of optimization, there was the challenge of working the Three Javascript library in that its documentation as incomplete at times. It also had functions defined that would only work for simple demos and required round about manners to get information about texture coordinates in my first attempt at getting lighting information from the scene. None of these challenges however cannot be overcome with future work and time.

III. Results



Above are results gathered by our optimization.
Top row: Simple case testing if we can keep desk inside arbitrary made spaces. Iterations 0,1,5,19
Middle row: Adding a inner wall to test if desk collide with it during any iteration. Iterations 0,5,19
Bottom row: More complex layout, with many walls, it can be seen desk can phase through walls Iterations 0,1,5,19

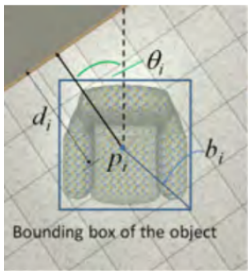
For this project I wanted to test if the current implementation of the framework would be able to first, be able to infer between indoor and outdoor space using the output texture generated for the floor. Second be able handle interior walls, and motion between walls of furniture objects. Lastly we wanted to test the implementation of simulated annealing to see if we could find the light, and if our visualization would show correct sequences to the light source.

In the above image in the first row we show the results of testing out ability to find interior and exterior spaces with furniture objects using texture look ups. We then discount any position that find a specific threshold of black pixels underneath a furniture item. While other parts of the framework will require reworking, the texture look up approach works well and fast for this application.

Above in the second row we make models that have interior walls and see if there is collision between the walls and furniture items placed inside. Because of our texture based look up and way the textures are made in the Virtual Heliodon wall have no floor texture defined underneath them, which means that any desk that get moved into a wall has some black pixels which pass the interior/exterior threshold and considers that item to be outside the given space. In thus making sure no furniture items get moved into walls.

In the last example in the third row we see that when running the simulated annealing algorithm that our furniture items can move between rooms, which can either be desirable if users have no prefer-

ence items populate specific rooms. Because of the way simulated annealing works, the initial heat variable allows items to make large movements, specifically 100 pixel movements in either the x or y direction of a 512x512 pixel floor texture. This wide movement allows desk to move between walls and settle in room with the most light.



This is the angle that was difficult to calculate using only floor textures. Edge detection of walls would have be implemented if only the floor texture was considered. Fortunately we have other data that contains a manner to collect this necessary information.

This is the angle that was difficult to calculate using only floor textures. Edge detection of walls would have be implemented if only the floor texture was considered. Fortunately we have other data that contains a manner to collect this necessary information.

Due to the limitation of not being able to find the angle between the backside of each desk and nearest wall with texture look up alone, the cost function and simulated annealing application will only solve for optimal light, and avoid self collisions. Future work is aimed at a full implementation using the framework provided from this project.

IV. Discussion

As explained before this project was three-fold as an exploration, motivation to get participants, and learning experience. In the results shown, I concluded that exploration justifies further research into a full implementation of this furniture arrangement. We have the framework and tools developed to take into consideration other items and any floor layout generated by users from our previously made interface.

Secondly, I hypothesise that with more furniture models we can create an furniture arrangement application, in which we ask users to design rooms and quantity lighting within the room. This application can be created into a game, with an competitive aspect that will further allow us to get more feed back and data to valid the algorithm behind inferring closed and open spaces, users lack of intuition in architectural daylighting, and eventually the benefit of a quick and easy way to visualize lighting with in an interior space.

Lastly, I can say this project was a learning experience for me, I learned about web development with WebGL and libraries available to help render and load meshes into our user generated scene. In addition to JavaScript optimization and other web development concepts. More importantly I learned how I can use WebGL to accomplish my goal of creating an effective user study that will be usable to anyone with a modern laptop.

V. Future Work

There is a lot of future work to be done, however most of the daunting work setting up the beginnings of this project is complete. Future work to be done is on the full implementation of the furniture optimization, in addition to a user interface that offers more visual information to the user regarding the optimization happing, such as the heat function and display of the current cost at each iteration. More intuitive controls are also needed, the controls made were setup for convince rather then simplicity and intuitiveness.

References

[1] Andrew Dolce, Joshua Nasman, and Barbara Cutler. Army: A study of multi-user interaction in spatially augmented games. 2012.

[2] mrdoob. Three javascript library.

[3] Joshua D. Nasman and Barbara Cutler. Evaluation of a tangible interface for architectural daylighting analysis. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, I3D '12, pages 207–207, New York, NY, USA, 2012. ACM.

[4] Yu Sheng, Theodore C. Yapo, Christopher Young, and Barbara Cutler. Virtual heliodon: Spatially augmented reality for architectural daylighting design. In *VR*, pages 63–70, 2009.

[5] Lap-Fai Yu, Sai-Kit Yeung, Chi-Keung Tang, Demetri Terzopoulos, Tony F. Chan, and Stanley J. Osher. Make it home: Automatic optimization of furniture arrangement. *ACM Trans. Graph.*, 30(4):86:1–86:12, July 2011.