Interactive Facade Detail Design Reviews with the VR Scope Box

Christopher Morse
SHoP Architects

Foteinos Soulos
SHoP Architects

ABSTRACT

We present the development of the VR Scope Box as an example of the potential for Virtual Reality to enhance the design process as an interactive medium. The opportunities afforded designers by virtual environments should not be limited to simple immersive visualization. The VR scope box is shown to able to visualize details in 3-dimensional space at a 1:1 scale with accurate material representations. This visualization is not restricted to a single typical example detail, but rather allows for the dynamic exploration of the entire facade system. At the same time, the building exterior as a whole is also visible, to allow for a simultaneous understanding of the connections and the consequences of those details on the building as a whole. Additionally, we discuss the importance of user experience on the usability and adoption of new tools within architectural design reviews and the advantages of developing such tools in-house.
INTRODUCTION
Over the past several years, virtual reality has emerged as an increasingly valuable tool for architectural visualization. It is now commonplace in many firms to walk through design models in VR. A number of applications such as Enscape, Prospect, and Holodeck aim to provide a quick and seamless solution for designers to experience their digital models in an immersive environment. While clearly valuable to the design process, as general visualizations they are limited in their functionality.

The opportunities afforded designers by virtual environments should not be limited to simple immersive visualization. Unlike mobile VR headsets, room-scale systems such as the HTC Vive, Oculus Rift, or Windows Mixed Reality allow users to move through the space more naturally. Tracked hand controllers provide a method of input with which the designer can move, manipulate, or change the contents of the experience. Such interactive visualizations can enhance the ability to communicate complex data and empower designers to explore designs more fully.

In this paper, we will present the development of the VR Scope Box as one example of the potential for this interactive visual medium to enhance the design process. In contrast to the software mentioned above, this tool was developed in-house without any intent to distribute outside the office. The bespoke development of VR Scope Box allows us to proceed more rapidly as it is not necessary to make the tool as robust and bug-free as distributed software would require. It also allows us to uniquely tailor the experience to each new project or building.

VR Scope Box
The architectural detail is a prime area in which to explore the potential of interactive visualization with VR. Facades, building envelopes, and structure are often initially designed by different people, but all must come together at a point of connection. This connection point must negotiate these different systems in complex ways. They must work in multiple dimensions and require careful consideration of the layering of materials. They consist of small parts, typically hidden from view in the final building, yet can have a large influence in the final patterning, rhythm, and material qualities of the facade as a whole. Despite these characteristics, details have remained largely constrained to 2-dimensional drawings for design reviews.

With the VR Scope Box tool, the goal was to be able to visualize details in 3-dimensional space at a 1:1 scale with accurate material representations (Figure 1). This visualization should not be restricted to a single typical example detail, but rather allow for the dynamic exploration of the entire facade system. At the same time, the building exterior as a whole should also be visible to allow for a simultaneous understanding of the connections and the consequences of those details on the building. Finally, the tool should allow for multiple design options to be displayed and compared.

BACKGROUND
Software including Enscape, Prospect, and Holodeck provide designers with ways to experience their models in virtual reality in a mostly static way. Work by Arnowitz et al. discussed the potential for creative authoring of digital models within a virtual environment with their vSpline project (Arnowitz et al. 2017). Such environments are now seen in software such as Gravity Sketch, Tilt Brush, or Blocks and have been experimented with for use in architectural design (Barczik 2018). These are generalist applications, and while they do allow the user to import custom models, they do not provide any ability to customize the interactions with the models produced.

Heydarian et al. demonstrate that spatial immersive visual environments produce similar presence as physical environments on users, demonstrating the potential for VR visualizations to inform and communicate design alternatives (Heydarian et al. 2014). Al Bondakji et al. look at data visualization within VR for helping urban planners access and understand complex data sets by presenting multi-dimensional data in a spatial environment (Al Bondakji et al. 2018). Others show the potential for VR experiences to illuminate the understanding of space and the importance of precision and fidelity through point cloud capture of a spatial environment and subsequent VR experience of the data (Nagy et al. 2018). The importance of user-centered design is shown in an example of a museum installation that allowed users to share an experience of urban data sets through a combination of VR point clouds and projection mapping, and emphasize the need for responsive user-driven design tools (Kreitemeyer et al. 2017).

Developing custom tools has in the past been too difficult and technical a process for most architecture firms to be able to afford the necessary time and employees with appropriate expertise. In recent years, game engines have grown in prevalence alongside immersive visualization hardware and have been shown to be well suited for visualizing architecture in general (Boekyns 2011). Black and Forwood demonstrate the use of Unity3d as a tool for communication and analysis of facade systems in particular (Black and Forwood 2017).
METHODS

The key functionality of this tool is the ability to dynamically place a scope box on a full-scale facade in VR. The full-scale facade remains in place, while a copy of the selected section is shown in a separate location, which allows the designer to view the full complete model at the same time as the call-out sectioned scope box, both at a scale of 1 to 1.

The tool is developed using Unreal Engine with the Datasmith pipeline available at the time of writing as an open beta program that provides an easy method for importing geometry from design software, in this case Rhino, into the engine’s editor. Below we discuss three main areas of focus that contribute to the successful implementation of this tool: technical methods for producing the sectioned scope box within VR with real-time performance, workflow strategies for importing new models quickly, and user experience considerations to allow for designers to intuitively use the tool.

Scope Box Implementation

The primary feature of the scope box tool is the ability to see a sectioned version of an imported model and to be able to set the area of focus dynamically. We look at three potential solutions for creating a ‘clipping cube’. The first method uses a custom material shader to calculate the pixels outside of the specified cube and render them invisible. The second method uses Unreal Engine’s built-in procedural mesh slicing function to cut the meshes along the cube. The third method integrates a third-party solid geometry library into Unreal Engine.

The first option is implemented through the use of a simple custom shader component within Unreal Engine’s material editor. A set of material parameters—containing the center and extents of the box as vectors—is updated as the box is moved. Along with each pixel’s world location, these vectors are used within the shader to calculate whether that pixel is inside or outside of the box. The result is used as an opacity mask to render the pixels outside the box invisible (Figure 2). This strategy allows the calculations to be run on the GPU and results in high performance. The downside is that this strategy is unable to produce new pixels along the cut surface, as it is only operating on a pixel-by-pixel basis, and the result is that cut solid meshes are seen as hollow.

The second strategy uses the engine’s built-in function that slices meshes. As this calculation is performed on the CPU rather than the GPU, it requires significantly more calculation time than the shader method. Additionally, the built-in function can only operate on a single plane. To create a sectioned cube we, therefore, had to perform this operation...
six times causing an even larger performance hit on the frame rate. In order to increase the performance, we first use bounding volumes to determine which meshes intersect the cube, and only perform the slicing operations on those meshes. While this method works well for simple test cases such as slicing a simple cube, we find we are unable to maintain an adequate frame rate for virtual reality in a fully detailed model. Additionally, we find the implementation to be imperfect in all cases and would occasionally produce clearly incorrect mesh sections (Figure 3).

As both methods are imperfect, we decide on a combination of the two. While the scope box is in motion, the custom clipping material successfully hides geometry outside the box. Once the box is no longer moving we then calculate the mesh slices to generate the caps. Attempting to calculate all slices at once can still cause a noticeable one-time pause, however we found this to be an acceptable condition for the results. Future work will look into performing the slicing asynchronously or over multiple threads in order to avoid any noticeable pause to the user.

A third option using a third party constructive solid geometry library to perform calculations was researched, but for time considerations it has not yet been possible to implement. It is possible, but certainly not guaranteed, that such a library would be more performant than Unreal’s slicing algorithm. This route would take more time to complete and would be more difficult to maintain over time. As the results from the combination of the first two methods are adequate for our use case, we do not anticipate going this route for this particular tool. We wish to note, however, that such integration would provide opportunities for even more advanced tools beyond the scope of this paper and remain an important area of interest.

In addition to the primary feature, we also implement a secondary feature of exploding the sectioned geometry to simulate an exploded-axon 2D drawing. This allows the designers visual access to parts that may be occluded within a detail assembly (Figure 4). In contrast to a static drawing that can be carefully curated for the particular area chosen for investigation, the VR Scope Box tool is dynamic, and we cannot predict what the contents of the box will be when the user decides to explode it. We, therefore, use a procedural solution of first calculating the centroid of each mesh within the box and then moving that mesh outward radially from the center of the box proportionally to the centroid’s distance.

Keeping in mind the shader discussed above, which culls pixels outside the section box, simply moving the meshes radially outward on the CPU would move them outside of the box and thus render them invisible. This requires us to instead add a vertex offset on the shader, allowing the GPU to first calculate the visible pixels and then apply the vertex offset so that the resulting mesh segments would still be visible after being moved outside the box.
material shaders are not useable. We implement an editor script ("blutility") in Unreal to replace the imported materials with our custom material and to enable CPU access on the imported meshes, which will allow them to be included in mesh intersection calculations by Unreal. A second editor script is optionally run to search for design options by layer name and generate the user interface for the different options (Figure 5).

At the time of writing this paper, the Datasmith import process is only available in the editor and not at runtime in standalone executable applications. This means that in addition to the importing and adjusting settings, the application must also be compiled and packaged for each new model. While the runtime performance of the model would be improved by using pre-calculated static lighting, this would increase the build time for each new model. The decision to build the lighting is made on a case by case basis depending on the amount of time available.

Additional optimizations are also possible. The static full-scale facade that is not being sliced can be significantly optimized by joining all meshes that use the same material, which significantly reduces draw calls. For especially large and complicated models, it is also necessary to work with designers on modeling and texture strategies they can use to improve the import process and reduce the number of polygons that will result. We implement a workflow where designers separate geometry layers in the design software according to the material. We then use a script to mesh and join all geometry by layer, resulting in a single mesh per material significantly reducing the load on the CPU at runtime.

**User Experience**

A third area of focus during development was the user experience of the tool. In order to be adopted into a design review, the experience needs to be as simple as possible while still providing the necessary level of interaction to enhance the understanding of the design options. By working with design teams, we continue to iterate through different features and interactions.

The primary feature is the ability to dynamically move the scope box in real-time along the facade, and so this has been the focus of most of the user experience development efforts. The primary method uses a ray-cast pointer from the user’s hand controller to the facade, which allows the user to ‘point-and-click’ at a position on the facade, or alternatively move the pointer continuously along the facade (Figure 6). Because the user often stands a significant distance away from the facade, this method can be difficult...
to control carefully since a small motion in the hand would result in a large motion on the facade. Additionally, the software is only able to move the scope box along the surface of the facade even though many of the important details lie within the depth.

To overcome these problems we add a second method to provide fine-tuned control of the position of the scope box and to allow the user to move the box into the depth of the model. In this second method, the user puts their hand into the box, and can then move the section model by grabbing and dragging it from within the section box (Figure 7). Finally, the relative position of the section itself needs to be adjustable to allow the user to look at the details from the top, side, or bottom. For this, the user can grab the edge of the box to move it to any position within the virtual space.

RESULTS

The primary result of this project is a custom VR tool that has been integrated into design reviews of facade details. By streamlining the workflow and developing scripts to automate repetitive work, we are able to reduce the deployment time to 30 minutes from receiving a Rhino file from the designer to creating a packaged executable file to run on any VR-capable computer.

The short deployment time is essential to the limited time provided to the designers between 3D modeling production to client or internal presentation. With the help of this tool, the viewer experiences the design from a very close distance, concurrently understanding how the proposed detail design affects the overall architectural reading of the project. By being able to view options using this platform, design teams have reported anecdotally the ability to reach a decision faster because of the enhanced ability to understand the design holistically.

With regards to user experience, we anticipate that this tool would not be used frequently enough and long enough by the same users for them to develop a solid understanding of a complicated user interface. Instead the goal is to make the interface as simple and as intuitive as possible. To this end, we maintain two guiding principles: there should be no distinction between the left-hand controller and right-hand controller, and there should only be one button to use for the basic functionality.

The action of the button is, therefore, entirely contextual, changing depending on what the user is pointing at and the location of their controller. Through a process of user testing and iteration, we settle on four simple functions. First, by pointing at the facade model, the user can move the scope box to that point on the model. Second, by pointing at the ground, the user can teleport themselves to that location. Third, if the hand controller intersects with the scope box, they will be able to grab the box and move its relative position to provide a better view from the top, sides, or below. Finally, by reaching into the scope box, the user can move the section model within the scope box itself, providing more detailed control over the location of the box.
These self-imposed restrictions force us to be very deliberate in the features we choose to implement, resulting in increased focus on a small number of features. Through presenting the tool to new users periodically throughout development, we observed that such restrictions resulted in fewer questions and perceived confusion about the tool, as well as an observed decrease in the amount of time needed to perform the basic functions of the tool. We propose that the potential functional limitations of a single button are outweighed by the observed increase in ease of use by users who do not need to keep track of which hand they are using or which button they are pressing. We additionally note the advantage of such a system for improved accessibility of the tool for users who may not be able to use both hands or reach every button.

CONCLUSION
Future Work
Some areas of future work have been mentioned above, such as improving the performance of sectioning calculations through multi-threading or incorporating third party solid geometry libraries.

From a workflow point of view, we hope to continue to decrease the turnover time from model to VR by implementing runtime loading of models, which will allow designers to use the software directly without the need to repackage the application through Unreal. Toward this goal, we have begun integrating the OpenNURBS library that will allow for the reading and writing of Rhino files. We have additionally made some initial efforts toward implementing a Speckle client that will allow for real-time data flow between CAD software and our custom VR tools. We fully expect, however, that as more designers become developers and use tools like Unity and Unreal Engine more extensively, it is likely that these companies will implement better runtime import functionality.

Additional features that we intend to implement include a system of annotation within virtual reality and a multi-designer experience so that more than one person can be present in the same virtual environment. These features would facilitate discussion and exploration of the design options. While this tool has successfully moved beyond a proof-of-concept level, it nevertheless is not intended for general distribution. This allows the development team to focus their time on features that can improve the design process while not needing to solve every potential edge case or fix every single error.

A final important finding from this development is the importance of the user experience and simplifying the
interaction to produce the most informative immersion with intuitive controls. A 2D printout of a detail does not contain the same information as an immersive virtual reality interactive experience, but anybody with a pen can draw on it without thinking about how to use a pen, and a group of designers can gather around without needing to put on a headset or worry about cords. As we develop these tools, a key focus of UX design will be to enable the most useful features while maintaining an intuitive experience. We anticipate the solution to be a system of controls that allows experienced users to optionally enable more advanced toolset without losing the initial simplicity for first-time users.

Connections
As virtual reality and augmented reality become more common in the architectural design, it will become increasingly necessary to understand these technologies with a critical view of what advantages they bring to the process. Realtime visualization technologies and the increased influence of software development in architecture are leading to increased custom tool development, where software development can be done at the local level with a focus on a single tool or exploration rather than the full software ecosystem of companies like Autodesk.

The VR Scope Box is an example of a tool that is difficult to produce at a scale that would allow it to be developed for general consumers. The use case is too narrow, and the interactions too specific to a unique project or building. For this particular instance, the development of the tool may not have saved time overall; however, these tools are becoming easier and quicker to develop. In the same way that a custom Grasshopper script is both particular to a single project and also contains parts that can be reused across different projects, in-house software development has the potential to be written in a way that makes functionality easily reusable across different custom applications. Such tools not only will be more general than Grasshopper because they will not be restricted to the Rhino environment, but also more specific in the functionality they offer.

IMAGE CREDITS
All drawings and images by the authors.

REFERENCES


Christopher Morse is an Associate of Interactive Visualization at SHoP Architects. His research focuses on the integration of interactive visualization technologies such as VR and AR into all phases of the design process. He is collaborating with the Department of Architecture at Cornell University to use immersive VR tools as an integral part of an urban design studio. He received a Master’s degree in architecture from Cornell University and holds additional degrees in mathematics education and physics.

Foteinos Soulos is a Senior Associate at SHoP Architects. Foteinos has been the facade lead on several big-scale projects including Pier 17 at Lower Manhattan. Currently, he is managing “9 Dekalb,” a 77-floor residential tower in Brooklyn currently under construction. He is a Mentzelopoulos Foundation Scholar and holds a Master’s degree in advanced architectural design from Columbia University and a Diploma in Architecture and Engineering from University of Patras.