On-Site BIM-Enabled Augmented Reality for Construction

Adam Chernick[1(✉)], Christopher Morse[3(✉)], Steve London[1], Tim Li[1], David Ménard[2], John Cerone[1], and Gregg Pasquarelli[1]

[1] SHoP Architects, New York, NY, USA
ajc@shoparc.com
[2] Unity Technologies, San Francisco, CA, USA
[3] Assembly OSM, New York, NY, USA
cwm@assemblyosm.com

Abstract. We describe a prototype system for communicating building information and models directly to on-site general contractors and subcontractors. The system, developed by SHoP Architects, consists of a workflow of pre-processing information within Revit, post-processing information outside of Revit, combining data flows inside of a custom application built on top of Unity Reflect, and delivering the information through a mobile application on site with an intuitive user interface. This system incorporates augmented reality in combination with a dashboard of documentation views categorized by building element.

Keywords: Augmented reality (AR) · BIM · On-site construction

1 Introduction

1.1 Motivation

The process of constructing a building is complex. A typical deliverable for the architect is a set of construction documents that outline design intent, which can easily consist of upwards of 2,000 pages.

Flipping pages and cross-referencing drawings in order to gain understanding of design intent can be inefficient and confusing, especially within the ever-changing context of a construction site. In 2D drawings, it is inherently difficult to understand three-dimensional depth and how new construction connects to the as-built or context. Navigating these documents on site is unintuitive and can be spatially misleading, leading to errors and costly change orders. "It is becoming increasingly necessary to develop new ways to leverage our project data to better manage the complexity of our projects and allow the many stakeholders to make better more informed decisions" [1].

At the same time, new technologies are emerging that create opportunities to change the way this information is delivered and consumed on site. Mobile devices such as smartphones and tablets are increasingly pervasive and provide more computational power than ever before. Pre-existing cloud computing and data streaming infrastructure

P. F. Yuan et al. (Eds.): CDRF 2020, *Proceedings of the 2020 DigitalFUTURES*, pp. 46–56, 2021.
https://doi.org/10.1007/978-981-33-4400-6_5

is making data connections to these devices faster and more seamless. Camera improvements and image processing algorithms allow for mobile devices to continually scan their physical locations and analyze the results with increasing precision. The combination of these advances creates the ability to use augmented reality (AR) on site and in real time as construction proceeds. While still in their infancy, studies of the uses of new technology methods within construction have already shown productivity gains of 14 to 15% and cost reductions of 4 to 6% [2].

1.2 Related Work

There are many products on the market that look to solve related issues. Procore and Fieldwire are similar solutions in that they focus heavily on the management aspect of the construction process. Their solutions have robust tools for handling submittals, RFIs, financials, and schedules of a project. Bluebeam is a session-based PDF markup tool largely used for construction documents, with some functionality built around navigating a drawing set efficiently. Bluebeam has found a way to link sheet number callouts within a drawing as a sort of hyperlink to the next PDF drawing, reducing the time it takes to navigate a drawing set. Their collaboration tools also let teams jump into a live "Google doc"-like session that can be edited simultaneously. InsiteVR provides an immersive multi-user VR solution for construction and project design coordination. Insite's tools let users jump into 3D models before they are built to overlay multiple trade models to find conflicts and track issues. These solutions address similar information gaps as our solution, but in fundamentally different ways and within different scopes.

Other related work within the field of AR has also advanced in recent years. In their Rocky Vault Pavilion, Sun and Zheng describe a hybrid fabrication paradigm for onsite free-form construction using Unity3D and immersive AR (specifically, the Hololens) [3]. Fologram has also been used with the Hololens to assist in complex assemblies, and it's on-site capabilities have been demonstrated in the bending of steel and wood [4, 5]. Unlike these solutions we are not focusing on construction management as a whole, nor documentation viewing, but on a specific process for efficiently obtaining project information and spatial understanding.

1.3 Our Solution

At SHoP, we have developed a prototype application for mobile devices that addresses these issues. By combining interactive AR on site with a digital set of linked drawings and models, we aim to enhance the ability of construction workers and contractors to more quickly and comprehensively understand design intent within the direct physical environment of the existing construction. They are able to easily access all the information they need within the actual building context. The BIM model can be used as a central repository of information to which external tools for analysis, interaction, and representation can be linked [6].

In this paper we discuss two primary features. First is the ability to access relevant two-dimensional drawings from the traditional drawing set. Rather than jumping from sheet to sheet and view to view based on reference numbers, the user instead is able to

access all drawings that relate to a specific building element of interest by selecting that element.

The second feature is the ability to overlay models digitally onto the physical space of a construction site using AR. This feature requires alignment of digital and physical spaces, interaction with the digital models to display the appropriate geometry of interest at any given time, and well-optimized digital models that maintain a high level of visual fidelity and information accuracy while also being computationally efficient to display on low-powered mobile devices.

In order to implement this application, we use the Unity 3D real-time engine for development and Unity Reflect for data import from Revit into our application. For the first round of development, we use a static model, but we anticipate taking further advantage of Unity Reflect in order to establish a live connection as part of our future work. Additionally, we implement custom data preparation workflows in order to create additional metadata relating to the building elements as well as process the existing drawings from Revit into formats usable within our application.

2 AR Application

Our solution enables an on-site user to obtain associated drawings and details about a specific building component in one click. AR accomplishes this in an immersive way, in a 1:1 scale that gives the user spatial understanding within context. Additionally, we note the importance of providing a simple user interface and intuitive user experience, as well as offline capabilities to account for challenges inherent at construction sites. The fundamental differences between other solutions and ours can be broken into two key pieces: contextual model overlay using AR, and model interaction as a query system.

2.1 Model Overlay Using Augmented Reality

By using AR to overlay our BIM model on site, the user can see conflicts with the contextual, or as-built, environment that would not have been possible to see using typical methods. The one-to-one scale enables instant spatial understanding that is difficult and time consuming to distill from the typical drawing set itself.

3D model localization is the process of "pinning" a virtual 3D model at a specific position, scale, and rotation within the physical world. Our application uses image recognition for localization; however, a single image recognition solution does not work within the context of a construction site (Fig. 1). As construction sites can be very large and complex, a single image target will allow the 3D model overlay to "drift" as the user moves around the site, causing inaccuracy between the digital and physical environments. We therefore create a solution that re-localizes at different positions around the construction site. Multiple unique image targets are set at specific intervals around the site which work to negate drift between the digital model overlay and physical environment, as well as reset the model's origin to the new target position.

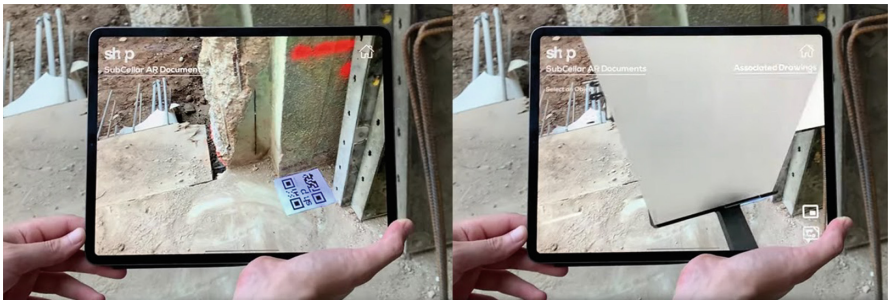

Fig. 1. Image recognition system for localization of digital model overlay

2.2 Model Interaction as Query System

The established connection between model and drawing data enables the 3D model to become an interactive query system to obtain relevant information. This is a fundamental shift from flipping physical pages or searching through a PDF. By embedding metadata into the model, we are able to list and give access to associated information related to a specific selected element. This is in contrast to navigating a traditional construction document set, which is typically a sequence-based approach for locating and referencing applicable drawings and details.

In a typical construction document system, to understand and build a small section of a simple partition wall, the navigation sequence can take many steps, requiring a subcontractor to jump back and forth between many different sheets, each of which contains only some of the information required and requires references to other sheets for a full understanding. Our system is different: The user selects the building component in question from a 3D view and instantly gets a list of all associated drawings and information (Fig. 2).

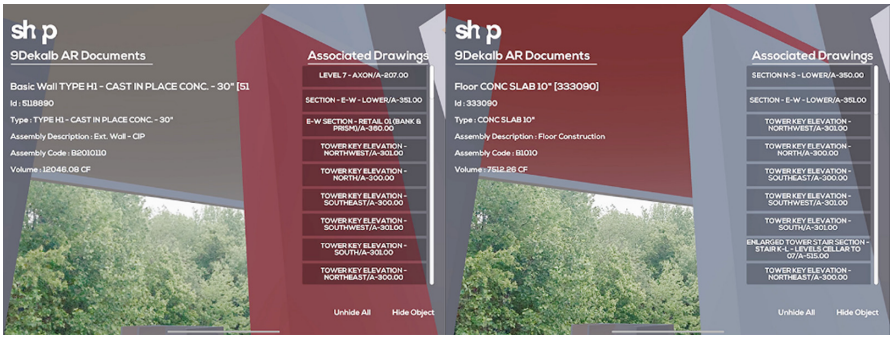

Fig. 2. A screenshot showing the information and associated views of a selected model element

2.3 Abstraction of Drawings

"Sheets" within a construction document set are based entirely on their physical namesake: a printable sheet of paper. Typical sheet sizes include ARCH D 24" × 36" and ARCH E 36" × 48". PDFs are still based on what can fit in those predefined areas. We propose to maintain the idea of "sheets" as a collection of sub-drawings, but also to expand it beyond its traditional page boundaries. Our system does not include sheets but will in future development feature a view dashboard that performs the same function of presenting multiple unique yet related drawing views.

In the current version of the application, these unique drawing views are presented independently and individually to the user. These unique views have associated metadata which contains their view name, view number, and associated sheet name and sheet number, to be used in the future development of our view dashboard system. This dashboard will lean on its roots in current typical sheet systems to allow easy adoption.

2.4 Additional Features

Additional features that have been developed to a proof-of-concept level include augmented versions of typical drawing elements. These drawing elements include augmented section cut marks and callouts. The iconography of these augmented elements mimic their 2D drawing counterparts to facilitate adoption and comprehension for end users (Fig. 3). This allows users who are accustomed to a certain way of working to easily transition to this new system. The section cut marker bears the same view and sheet number as the section cut drawing within the construction document set; in this sense, it is a literal extension of the drawing set (Fig. 4).

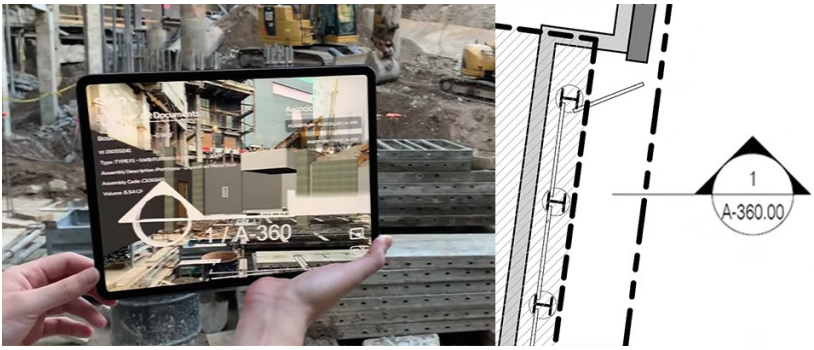

Fig. 3. Left, a screenshot of an interactive section cut callout in AR. Right, the counterpart call-out on within the traditional drawing set.

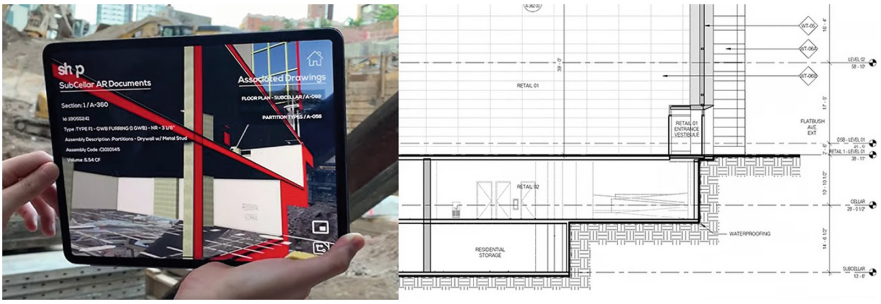

Fig. 4. Left, a screenshot of the section cut interactive BIM model in AR. Right, the counterpart section cut drawing within the traditional drawing set

3 Data Pipeline

In order to provide direct and intuitive access to the appropriate drawings, we developed a workflow to extract necessary information from the Revit model and bring it into our application (Fig. 5).

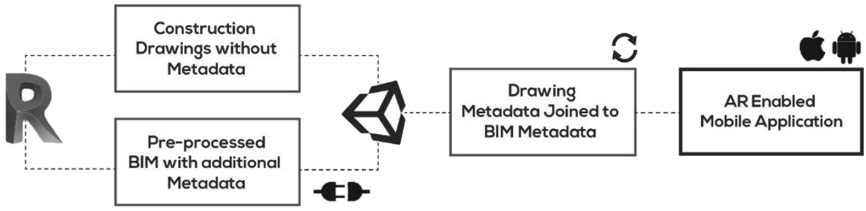

Fig. 5. Data flow diagram from the Revit model to the final mobile application.

3.1 BIM Pre-processing, Custom Parameter Creation and Population

The AR application requires a relatively simple data set. For each element selected by the user, the application requires a list of every Revit View that the element is visible in; This information is not readily populated in Revit. Although each View object contains an array of each element within it, the elements themselves are blind to the Views in which they occur.

We perform a brute force operation as a pre-processing step to generate an association of elements and views and save the results for later use in the application. We iterate through every View, then iterate again through each View's elements, ultimately using the Revit API to generate a custom 'Views Associated' parameter populated on each element containing the list of views that element is referenced in. For large models with many Views, this method can be extremely time intensive and not viable for a live application.

In order to respect Revit as an authorship tool and the origin of our data, rather than using an external file or data format, the list of Views is populated directly into a custom

Family Parameter embedded in each element. This embedded parameter also allows us to access the information after importing the model into our application through Unity Reflect. One downside to this strategy is that the resulting parameter cannot contain a list of information, so the view IDs are concentrated into a single string, then re-parsed once in Unity, requiring additional processing time. Although Revit doesn't currently maintain this data set itself, we think it should, and this approach can serve as a placeholder until it does.

3.2 Construction Document Export and Metadata Post-processing

3.2.1 Construction Document Export

Without the physical limitations of construction documents (CDs), the concept of paper sheets can be expanded to accommodate faster and more intuitive navigation. Our proposed solution entails a dashboard of views dynamically displayed as the user selects an element in the overlaid 3D model. The dynamic grouping of views replaces the role of static CD sheets, and as a result, the individual views defined in Revit become the basic unit of a set of construction documents rather than full sheets.

The collection of views is exported from Revit as high-resolution PNG images and are named according to Revit's default naming convention of the family and type of the view and the view name. In order to provide access to these views, the application associates the View ID generated in the metadata processing step with the file names generated here. Revit offers a limited ability to customize the prefix and suffix to the file name and does not include other useful identifiers such as ViewID.

To provide the information needed to connect elements to the resulting views, as well as to maintain the option for conventional sheet number identification, we use the view schedule functionality of Revit and pull together useful attributes of the views, including view ID, view name, associated sheet number, sheet name, and family and type into a single lookup table. This table is exported as a CSV file, which can then be parsed into our application, and the appropriate fields can be recombined to generate the corresponding file name for each View ID.

3.2.2 Metadata Post-processing

To reduce runtime loading and processing of data, the CSV file is pre-processed in the Unity editor using an Editor Script to generate a hierarchy representing sheets and views with associated metadata components. We found this structure to be useful for organization while developing the application, as well as for maintaining relationships between views that can be utilized in addition to the element-based view sorting described above. Additionally, we populate a dictionary using the View ID as a key in order to efficiently search for the relevant information after selecting an element.

The large number of high-resolution images presents performance challenges for the use of Unity with low-powered mobile devices. A system is developed to load and unload images at runtime. In order to reduce development time for a proof of concept application, we first attempted to use the Resources system built into Unity to allow for runtime loading of data, but found that this system generated uncompressed image storage which was then too large for the application to handle. As a workaround, we

currently utilize Unity's Sprite Atlas feature instead. We have found this system to work well to organize and reference images, without running into file size issues when building the application.

Alternative systems were investigated that would allow files to be stored remotely in cloud storage and downloaded to the application when needed. Such systems would allow the overall file size to remain at a minimum, while also allowing for individual runtime loading of images. This could potentially be done with built-in systems such as Asset Bundles or Addressables, or developed independently to allow for direct and automated delivery of images generated from Revit without requiring pre-processing in Unity. Such methods, however, would require near-constant internet connectivity on the job site. For both this reason and to reduce development time, remote storage of views is not currently implemented. This is an area of ongoing refinement and development, and we expect that a combination of strategies will ultimately be required for optimal performance.

4 Unity Reflect

Our solution leverages Unity Reflect in order to translate Revit data into a real-time context. Since Revit is a parametric tool, its data needs to be tessellated into geometry for a real-time engine to consume. This process would typically be done by exporting the model to a geometry format like FBX or OBJ, but there are three main issues with this workflow. First, the exported data loses all the BIM information associated with it in the process. Second, when going from parametric models to geometry, the resulting geometry is massive and not adapted for real-time rendering. Third, every change in the source model would need a fresh export.

Reflect allows for all the data to be dynamically generated, while maintaining the metadata attached to the geometry. In Revit's case, this metadata is the BIM information added to the model. The connection with the source data is also maintained so that any upstream changes to the Revit model are transferred to the game engine. Reflect also handles the complexity of the generated geometry by simplifying the resulting meshes and merging them together when appropriate.

Our solution leverages this connection to import data into the Unity Editor. The incoming metadata is then accessible through the Unity engine, both for the metadata post-processing step as mentioned above, and for displaying it at runtime to inform the end user. Reflect also handles the complexity of the data coming into a real-time engine so that no additional work is needed after the import process.

5 Results

5.1 Case Study/User Testing

We have tested this on site at the supertall tower designed by SHoP currently under construction at 9 Dekalb Avenue in Brooklyn, NY. Our internal construction administration team, as well as the general contractor and owner, have tested the application within a limited scope and have expressed that it could positively impact their work efficiency.

The project director for the general contractor and owner JDS Development, Michael Jones, outlined the difficulties of communication on a construction site. He and his team noted that they believe this application could prevent expensive mistakes on site. Michael and his team mentioned a few improvements that could increase the usability of the application, including the option to drop the opacity of the 3D model overlay to see the underlying context site, as well as the ability to add comments at specific positions within the digital overlay.

5.2 Future Development

The system described in this paper is an initial proof of concept application. There are many areas of future development planned based on initial development roadmaps, as well as user feedback from testing the existing application on site.

We have identified several potential bottlenecks that delay the import of information from Revit into our system. This includes pre-processing information to establish relevant references, as well as producing and processing the 2D drawings. We are looking into implementing workflows to automate these processes.

We note that the generation of lists connecting Revit Views with elements is not a particularly novel problem to solve. Having elements that contain references to their View contexts is something that we believe could be implemented as a native feature of Revit. One reason why it may not already exist is that Revit was ultimately designed to produce drawing sets. The idea that individual model elements may be interactive on site is a relatively new concept, so this contextual awareness was unnecessary. It is worth noting that during our development, we discovered that the Autodesk Forge API does contain a "__viewable_in__" property associated with each element which, as the name suggests, lists Views where each element occurs [7]. A future effort may take advantage of this API by uploading our model to BIM360, letting it do the processing, and then accessing this parameter via the Forge API.

Another area of future development is connecting the Unity Reflect system to a live Revit model to allow automatic updates for the end user. This would allow the transference of dynamic views in context, while keeping the complexity of the model abstracted away, only delivering to the end user what is needed in their current context. While the Reflect system already handles the information transfer well, the future work in this area is more focused on legal questions around how to validate and approve changes within this non-traditional method of information delivery.

One of the largest areas of improvement is the user interface and user experience. While we continue to work on the dynamically generated view dashboard, we will maintain the ability of our dynamic system to tailor the grouping and sorting of this set of information to specific users. There is the potential of integrating sorting algorithms such as Brin and Page's PageRank Citation Ranking based on the number of elements referencing each view, or the number of times all users view them [8]. This dashboard system will break the rigid drawing set structure into something that can be expanded upon, re-organized, and integrated with other information. Instead of pages collected by type and level of detail, we can present drawing information deliberately grouped together to answer questions as they are asked.

The goal of this project is to facilitate information delivery on-site to the people who need it. Without an intuitive user interface and seamless user experience, we run the risk of losing the advantages of real time communication because of a hard-to-use application. This level of development requires rigorous user testing to establish how people expect to use such a system, and uncover what features are most useful.

5.3 Connections

In addition to the specific application described in this paper, we see larger-scale potential to change the ways building designs are delivered, communicated, and built. The prevalence of devices and software that can visualize three-dimensional geometry in increasingly immersive and interactive environments suggests a fundamental shift away from static two-dimensional drawings sets and towards connected digital models augmented by metadata. Such a shift allows for more direct communication between designers and contractors and has the potential to cut out the middle stage of converting models into two-dimensional projections that lose information.

The amount of data that can be generated and delivered will continue to increase, and the filtering and selective display of this data will thus become increasingly important. Methods of data visualization and user interaction will help communicate this data in effective ways to produce better outcomes.

With this in mind, we expect that some of the lessons learned from this effort involving data interoperability and user experience testing will be increasingly useful for other bespoke applications for future projects and future scenarios.

References

1. Aparicia, G., Kontovourkis, O.: Sustainable computational workflows. In: 6th eCAADe Regional International Workshop Proceedings (2018)
2. Koeleman, J., Ribeirinho, M., Rockhill, D., Sjödin, E., Strube, G.: Decoding Digital Transformation in Construction. 20 August 2019. [Online]. Available https://www.mckinsey.com/industries/capital-projects-and-infrastructure/our-insights/decoding-digital-transformation-in-construction. (Accessed 05 2020)
3. Sun, C., Zheng, Z.: Rocky vault pavilion: a free-form building process with high onsite flexibility and acceptable accumulative error. In: Yuan, P.F., Xie, Y.M.M., Yao, J., Yan, C. (eds.) CDRF 2019, pp. 27–36. Springer, Singapore (2020). https://doi.org/10.1007/978-981-13-8153-9_3
4. Jahn, G., Newnham, C., Beanland, M.: Making in mixed reality. holographic design, fabrication, assembly and analysis of woven steel structures. In: Proceedings of the 38th Annual Conference of the Association for Computer Aided Design in Architecture (ACADIA), pp. 88–97 (2018)
5. Jahn, G., Wit, A.J., Pazzi, J.: [BENT]. In: Proceedings of the 39th Annual Conference of the Association for Computer Aided Design in Architecture (ACADIA), pp. 438–447 (2019)
6. Abdelmohsen, S.: Genres of communication interfaces in bim-enabled architectural practice. In: 6th International Conference Proceedings of the Arab Society for Computer Aided Architectural Design (ASCAAD), pp. 81–91 (2012)
7. Goncalves, G.: Navigating Between 2D Views. 21 June 2017. [Online]. Available https://forge.autodesk.com/blog/navigating-between-2d-views. (Accessed May 202)
8. Page, L., Brin, S.: The PageRank Citation Ranking: Bringing Order to the Web. (1998). [Online]