Utilizing Parametric Computational Modeling to Generate Masonry Wall System to Facilitate Robotic Task Creation

Austin D. McClymonds;[1] Somayeh Asadi, Ph.D.;[2]
and Robert M. Leicht, Ph.D.[3]

[1]Dept. of Architectural Engineering, The Pennsylvania State University, Email: adm5535@psu.edu
[2] Associate Professor, Dept. of Architectural Engineering, The Pennsylvania State Email: sxa51@psu.edu
[3]Associate Professor, Dept. of Architectural Engineering, The Pennsylvania State Email: rml@psu.edu

ABSTRACT
Building information modeling (BIM) technology in construction has become increasingly prevalent in recent years, and the integration of robotics is seen as a natural step to improve the efficiency and accuracy of the construction process. To increase the level of development (LOD) of a BIM model to support a construction robot, parametric modeling can be used to create highly detailed models by supplementing and defining the geometric and physical properties of the construction elements, such as the components' size, shape, and material parameters, which can be used as inputs for designing robotic tasks. Component information and data are stored as parameters and extractable from the BIM model, allowing a robot to perform highly precise and repeatable tasks. This study develops a framework for implementing computational parametric modeling for masonry wall systems modeled in the BIM with Dynamo as the computational modeler. This study tested six wall configurations constructed of 8" x 8" x 16" concrete masonry units (CMUs). Dynamo successfully interpreted most wall geometries and placed most full-sized CMUs into the correct design locations. Errors occurred when placing partial-sized CMUs, typically shown at wall intersections, revealing a need for further script refinement. The study shows the careful planning and considerations needed for implementing computational modeling to generate model content and identifies areas for future work.

KEYWORDS
Computational Parametric Modeling; Information Exchange; Robotic, Construction, Building Information Modeling

1.0 Introduction
The construction industry has rapidly adopted Building Information Modeling (BIM) technology in recent years, and it has proven to enhance collaboration, reduce errors, and save time and money throughout the project's lifecycle (Borrmann et al., 2018). As BIM technology continues to evolve, integrating robotics has been identified as a natural progression changing the dynamic of project information requirements. Notably, not all systems in the BIM model contain the desired level of development (LOD) to facilitate robotic construction (Ren & Zhang, 2021). Thus, third-party applications are being developed to fill this gap that integrates with BIM authoring software, such as Autodesk Revit (Karimi et al., 2021).
Additionally, computational modeling uses computer processing and algorithms to define and manipulate objects' geometric and physical properties within a BIM model (Ramage, 2022). One computational modeler is Dynamo, commonly used in conjunction with Autodesk Revit in the architecture, engineering, and construction (AEC) industries. Dynamo is a visual programming

platform that allows users to create custom scripts that automate tasks and manipulate data in Revit. This enables users to generate complex geometries quickly, perform parametric studies, and analyze data to inform design decisions. For this research, Dynamo is employed to increase the LOD of wall systems to add additional model information. Dynamo allows the creation of elements, and import/export models, among different applications that can be used to increase the LOD (Monteiro, 2016). The target LOD for wall systems is 400, based on the guidelines established by BIMFourm (Bedrick et al., 2020). Level 400 provides information about the model's construction and installation of elements (Banfi, 2016).

This research paper uses computational modeling to modify a parametric model developed in Autodesk Revit, increasing the masonry wall system's LOD and allowing location and material parameters to be extracted to support the creation of tasks for robotic construction. The methodology presented in this study outlines the developmental steps for the Dynamo script and provides insight into testing and validating the process. The study highlights the importance of careful planning and considerations for generating model content when implementing parametric modeling. This paper is intended to guide practitioners and researchers interested in implementing computational modeling in robotic construction and identifies potential areas for future work and expansion.

2.0 Background Literature

Integrating robotics in BIM has been seen as a natural step to improve the efficiency and accuracy of the construction process. This literature review explores current research and practice on using BIM with computational modeling to facilitate robotics in construction. Parametric modeling provides the capability to implement design changes and interpret model geometry (Debney et al., 2022), providing a viable method to supplement the model with additional model content, such as concrete masonry units (CMUs) or light gauge metal framing. However, material content libraries must be established to facilitate, which allows elements and geometry to be imported into the parametric model. Material content libraries typically contain physical properties, geometry, manufacturer, and other information that impacts the building process (Kim & Chin, 2016; Sharif & Gentry, 2015). Thus, what is contained in the library relies heavily on the use case subjected to the computational modeling (Venkatraj & Dixit, 2022). Notably, more complex geometry, such as partial-sized elements, require different methods for implementation into the construction project instead of importing standardized units from the library (Sharif et al., 2015).

Currently, research is being conducted to supplement model information with computational modeling. For example, (Follini et al., 2021) utilized algorithms that allowed for the extraction and modification of BIM element parameters to facilitate the robotic navigation of construction tasks based on geographic and semantic information. Research also exists for extracting information from a BIM model to support robotic construction. For example, some studies combined construction schedules with model elements to create construction tasks, such as the order in which a robot could paint walls (Kim et al., 2021). Additionally, Dynamo was used to create topological data in IFC files to aid in robot data collection and navigation (Karimi et al., 2021). Another study extracted geometric data from a customized BIM file and exported it following the contour crafting method of 3D printing structures (Davtalab et al., 2018). To summarize, BIM, computational modeling, and robotics in construction can significantly improve project outcomes and productivity. Further research should focus on developing integrated systems seamlessly combining BIM with computational modeling to facilitate robotics in construction.

3.0 Process for Parametric Wall Detail Generation

The development of the computation modeling script was built on the previous studies discussed and applied to different wall configurations developed in Revit to test its capabilities and determine its limitations. The script was developed in Dynamo to increase the LOD to level 400 of masonry wall construction modeled in Autodesk Revit. It should be noted that the primary focus of this research is the generation of model content in the BIM model that supports the future of robotic construction by providing a method to supplement information for robotic construction between the first and second phases of the system architecture (McClymonds et al., 2023). The robotic platform utilized for the information requirements for this research is the Clearpath Husky A200 utilizing the Robot Operating System (ROS) Melodic (Stephans et al., 2022).

The methodology includes four main steps: analyzing project requirements, developing system architecture, implementing data management procedures, and testing and validating the implementation of the system. The first step involves analyzing project requirements to identify the project's goals, scope, and limitations. The second step focuses on developing the system architecture by defining the workflows and procedures necessary to create the parametric models. This includes selecting the appropriate BIM software and creating templates for the wall systems. The parametric models must be designed to support robotic tasks, so the geometry and physical properties of the construction elements, such as the components' size, shape, and material parameters, must be accurately defined. The third step involves implementing data management procedures to ensure the model content is accurate, consistent, and easily accessible to construction robots, establishing naming conventions, and creating material content libraries. The final step, the focus of this study, tests and validates the script for masonry wall construction and presents the process associated with the Dynamo script.

3.1 Model Setup

To begin the model setup, walls modeled in Revit must be imported into Dynamo, where parameters associated with the geometry and materials are utilized. All walls are selected and converted into elements allowing parameter extraction. The wall construction materials are extracted using a naming convention based on the standardized formats for construction materials, such as 8" × 8" × 16" CMU providing a means to import a Revit Family file with the same name. The script then setups up the means allowing for parameters to be extracted from the walls based on the geometry in the model, which includes references edges for the wall models and dimensions of the wall material. Additionally, parameters are created for generic models in later phases of the script, allowing values for length, width, and height to be set for partial-sized elements in the model. Partial-sized elements are being created individually to reduce complexity in the script and family models.

3.2 Model Development

The first step for model development is to extract the reference edges from the model geometry, a line around the wall's perimeter. The one extracted here is for the wall's structural material exterior edge; the structural material here is CMU. A filter obtains the bottom reference edge for each wall section. After the reference edge is known, its start location is marked based on how the wall was modeled, resulting in the start location being placed at its initial model location. The reference edges and start locations are depicted in Figure 1.

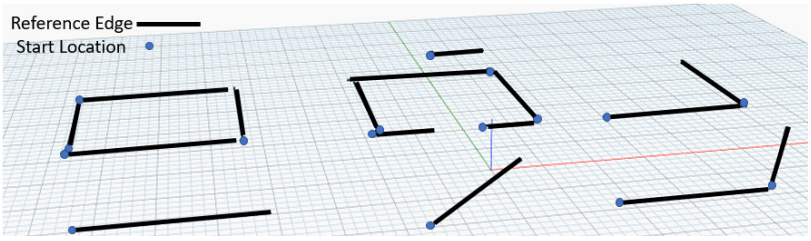

Figure 1: Reference Edges for Generating Model Elements

One issue that had to be addressed at this stage was that there were more reference edges than walls due to reference edges around openings. The model has 16 reference edges and 14 walls where all data is stored as lists, which vary in size. To overcome this challenge, additional lists were created for the wall and materials parameters to equalize list sizes to provide the required parameters for the additional reference edges. After the start location of the reference line is marked, locations for the first and second-course blocks are determined based on the size of the element being put into place. For the first course of the wall, coordinates are marked every 16" from the start point of the reference edge till the edge of the wall. The last point is removed from the list as it would allow an additional CMU to be generated past the geometric constraints of the wall.

For the second course of the wall, geometric offsets are applied to the reference edges based on the first course. The first offset value is applied to the points generated for the first row based on the bond type of the wall system. Here an offset factor of 50% correlates to a running bond; however, an offset of 0% would be applied for a stack bond. This provided adjustments to the reference edge of the second course to reflect the running pattern by adjusting its start and end locations. An additional offset was applied to the points equal to the element's height, 8" in this case. Was the reference edge is offset to the correct location, points are placed every 16" following the same procedure as the first course. Once the coordinates were determined for the first and second courses, these coordinates were translated to additional rows in the project providing the locations for all full-sized elements in the system. Figure 2 shows the location of each element modeled as a dot in the image based on lines determined from the points located on the second and first rows of the wall up to the top constraint of the wall spacing the blocks based on their height.

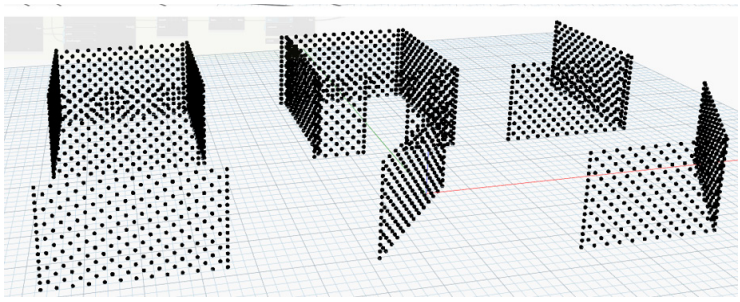

Figure 2: Generation of all Coordinates for Full-sized CMUs

Once all points for full-sized units have been generated, partial-sized units are then created in the model. Gaps are measured at the start and end of the reference lines to the edge of the nearest element providing the dimension for the length of the partial CMU. The height and depth of the block are assumed to be equal to the surrounding CMUs for the blocks providing the locations for each corner of the partial-sized CMU. From there, a cuboid is generated and converted into a solid

model to represent the partial CMU, which is converted into a generic model and named based on the same naming convention utilized for the full-sized CMUs. Figure 3 provides insight into the partial-sized CMUs and shows their locations as solid cubes.

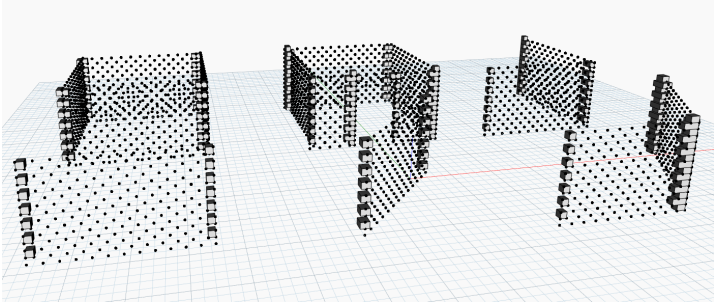

Figure 3: Generation of Partial-Sized CMUs.

The last step for the model development is to place and orientate the components in the model. Full-sized components are placed at coordinates identified and rotated based on the directional vector extracted from the wall converted into degrees. Elements are realigned to their coordinates to ensure the blocks are placed in the correct design locations. Without this step, slight variations in block placement were observed, such as blocks being misaligned from the edge of the wall. The additional alignment reduced the misalignment. Figure 4 shows the results of generating blocks for each wall system.

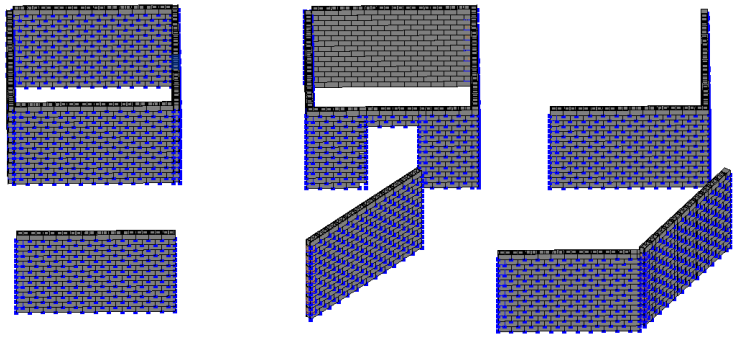

Figure 4: All CMUs Generated by the Dynamo Script

3.3 Information Extraction

The information extracted from the model is based entirely on requirements determined to facilitate robotic construction. These requirements were identified in previous studies and served as the basis (McClymonds et al., 2022; Stephans et al., 2022). Information to be extracted includes coordinates of each element (Full-sized and partial CMU), their orientation, type of element (8" x 8" x 16" CMU), and component identification (ID). To accomplish this, an additional script was created in Dynamo, where all elements generated in the model were gathered. From there, information is gathered for each element in imperial and metric units. It is then formatted into a .CSV file that can be imported to the robot. Figure 5 depicts the script developed for the information exchange process with four overarching steps: gathering elements and parameters, extracting information, unit conversion, and exporting data.

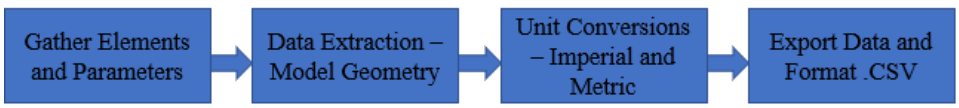

Figure 5: Dynamo Information Exchange Process Model

4.0 Results

Six wall configurations were tested utilizing the Dynamo script detailed above. These wall configurations are based on simple geometry that could be expected on a construction project, starting with a simple wall going in one dimension and progressing to rectangular rooms integrating door openings. Various wall configurations were tested because of the geometric differences that impact the process. Initially, the script was developed only for a straight wall, but efforts were made to incorporate additional configurations. Future testing iterations will include curved walls, different-sized blocks, and different materials. The method was tested on wall configurations consisting of 8" × 8" × 16" CMUs and 10 feet in height. CMUs are modeled with their nominal sizing and are dry set in place. Grout/mortar measurements are incorporated into the overall size of the CMU. Figure 6 provides details and justification for each wall configuration used in testing.

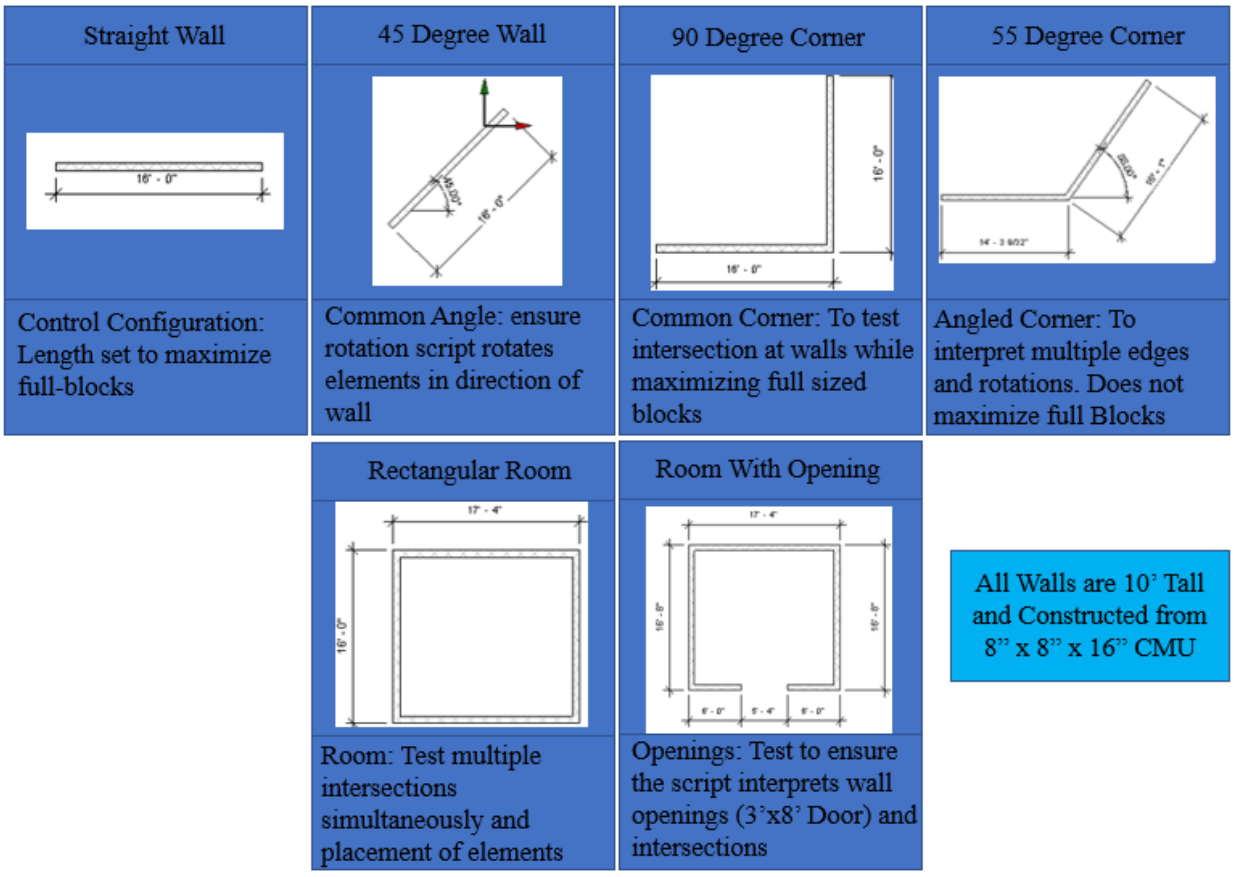

Figure 6: Tested Wall Configurations and Justifications

The results for the Straight Wall, 45 Degree Wall, 55 Degree Wall, and 90 Degree Wall are shown in Table 1. A positive difference means more CMUs were expected to generate, while a negative value means more CMUs were generated than excepted.

Table 1: Straight Wall vs. 45 Degree Wall vs. 55 Degree Wall vs. 90 Degree Wall

Wall Configuration	Expected Full CMUs	Expected Partial CMUs	Generated Full CMUs	Generated Partial CMUs	Full Difference (1-(Gen./Expect))	Partial Difference (1-(Gen./Expect))
Straight Wall	173	14	173	14	0%	0%
45 Degree Angle	173	14	173	14	0%	0%
55 Degree Corner	338	36	301	45	11%	-25%
90 Degree Corner	345	30	331	36	4%	-20%

Figure 7 shows the results generating the CMUs. For the straight wall and 45-degree wall, all blocks are placed into the expected locations, and the correct number of blocks are placed. A standard error is highlighted in the 90-degree corner in red, where two 8" x8" x16" CMUs were generated. This error is repeated at each intersection and was expected based on methods of interpreting reference edges. Another error occurred for the 90-degree corner for where 8" x8" x16" CMUs were placed instead of full-sized ones highlighted in yellow. The 55-degree corner had two locations overlapping partial and full-sized units highlighted in yellow. Additionally, the script generated a 19" long CMU instead of dividing it into partial and complete sizes, highlighted in blue.

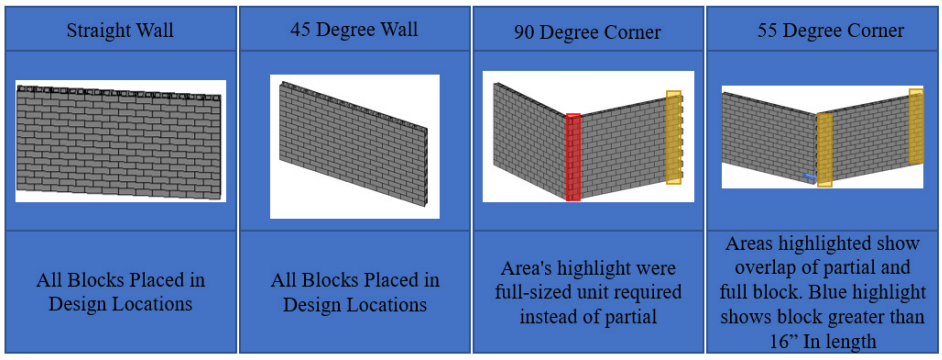

Figure 7: Straight Wall vs. 45 Degree Wall vs. 55 Degree Wall vs. 90 Degree Wall Results

After testing the four previous wall configurations, the rooms were tested. The results are shown in Table 2. A positive difference means more CMUs were expected to generate, while a negative value means more CMUs were generated than excepted.

Table 2: Rectangular Room Vs. Room with Opening

Wall Configuration	Expected Full CMUs	Expected Partial CMUs	Generated Full CMUs	Generated Partial CMUs	Full Difference (1-(Gen./Expect))	Partial Difference (1-(Gen./Expect))
Rectangular Room	722	56	692	70	4%	-25%
Room with Opening	668	75	642	78	4%	-4%

Figure 8 presents images of both rectangular rooms tested in this study. First, both rooms had the same error identified for the partial blocks at the corner where two 8" x 8" x 8" CMUs were

placed next to each other due to methods used to interpret reference edges. Additionally, both had an occurrence where partial CMUs were not placed at one of the corners (shown in red for the rectangular room and yellow for the room with an opening). Additionally, partial blocks did not generate to the left of the opening.

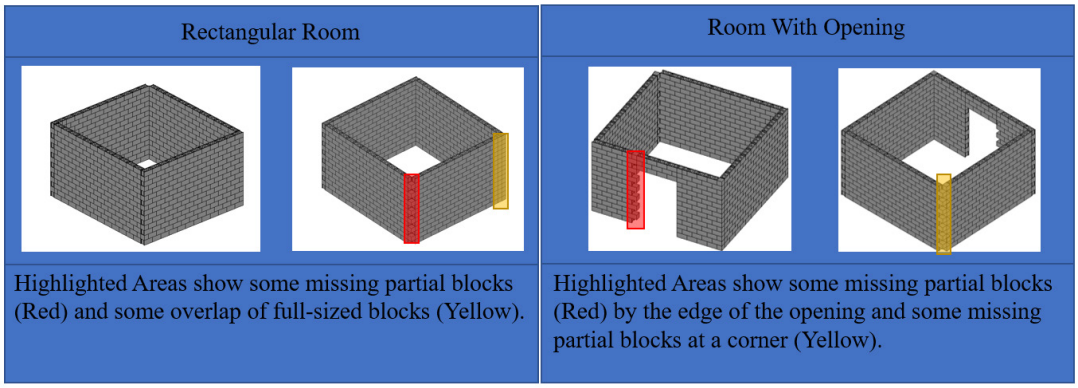

Figure 8: Rectangular Room Vs. Rectangular Room with Opening

Overall, the script generated 2,312 full-sized elements and 257 partial elements. These values were extracted directly from the Revit model. The expected values were 2,419 full-sized CMUs and 225 partial CMUs. Considering all wall configurations and block types (full and partial), 97% of the required blocks were placed into the correct location if considering the overall number of generated blocks (2,555) compared to the expected amount (2,637). Additionally, generating the model content took around 20 minutes, greatly influenced by the computer's hardware and redundancy in the script. All walls had the elements generated simultaneously.

5.0 Discussion

Numerous elements were placed based on the geometric and materials parameters set in Dynamo. The elements generated by the script must be verified to ensure they are placed in the correct design location. First, the elements were placed on top of the wall system to ensure that the blocks mirrored the locations displayed by the wall texture. After a visual inspection to ensure the blocks were in the correct design locations, the amount of full-sized and partial elements extracted is extracted from Revit, then compared to the expected number of units for each wall configuration. To determine the expected number of CMUs for each wall configuration, we verified the texture of the walls to ensure the blocks represent the correct size CMUs and are then aligned with methods used for placing the elements. Once the texture is appropriately aligned, the first two rows are counted and multiplied by the repetition of that row in the wall configuration. Since each wall reached a height of 10 ft, the first course was multiplied by a factor of eight, and the second was multiplied by a factor of seven. To determine the difference between generated and expected, the generated was divided by the expected, subtracted from one, and converted to a percentage.

There are noticeable differences between the generated elements and expected values; however, some variation was expected during the implementation, contributing to the configuration of the reference lines. Depending on how the model was initially modeled and orientated could change the starting location for the reference line and cause variation in the placement of the elements. Additionally, the script would occasionally misinterpret the relationship at corners between the reference lines resulting in partial blocks being placed instead of full-sized or the script neglecting to place a block. This was also noticed on the left of openings where partial elements should have

been placed; however, a cuboid was successfully placed at those locations but not converted into a generic model except for one element generated at the top. Therefore, with additional optimization and refinement, the success rate of the script would increase. Additional testing is required for blocks of other sizes and bond types.

6.0 Conclusion

This study sought to implement computational modeling for generating the components of masonry wall construction to aid in developing tasks for robotic construction. The components supplement the model with the information required for robotic construction. This study utilized Dynamo as the computational modeling platform to increase the LOD of wall structures for masonry construction in Autodesk Revit. Six different masonry wall configurations were designed and tested in this study, of which the straight and 45-degree wall configurations generated all full and partial-sized CMUs in the correct design locations. For the other wall configurations, errors occurred in generating partial-sized units, typically at corners and wall intersections. Some errors were expected, such as two 8" x 8" x 8" CMUs being generated side by side at intersections. A possible solution to this would be the addition of a node in Dynamo that combines two partial-sized CMUs if they are placed next to one another or be replaced by a full-sized one if the two partial units sum to be full-sized. Despite errors, the script successfully placed most units in the correct locations and orientations based on the geometric constraints of the wall. While it did not place all the elements required by design, it does provide proof of a concept, and refinements could prove to be a viable method for increasing the LOD for wall systems.

It requires an expansion for other materials. This process used here is not without its limitations. First, the script focused on masonry wall systems; alternations would have to be made for other wall materials. However, the methods explored here could be applied to other use cases, such as metal framing. Additionally, this method relies on material content libraries that are easily accessible and integrable into the parametric modeling process for full-sized elements. Revit families for each full-sized unit should be available in the material content library, and the naming schema of wall materials should correlate to it so the correct material is loaded into the project. Future work for this study will include adding the ability for more wall systems in the script and scheduling information to generate robotic tasks directly from Dynamo. However, despite these challenges computational model proves to be a viable method for increasing the LOD of a BIM model and could facilitate task creation for robotic construction in the future.

7.0 References

Banfi, F. (2016). Building Information Modelling – A Novel Parametric Modeling Approach Based on 3D Surveys of Historic Architecture. In M. Ioannides, E. Fink, A. Moropoulou, M. Hagedorn-Saupe, A. Fresa, G. Liestøl, V. Rajcic, & P. Grussenmeyer (Eds.), *Digital Heritage. Progress in Cultural Heritage: Documentation, Preservation, and Protection* (pp. 116–127). Springer International Publishing. https://doi.org/10.1007/978-3-319-48496-9_10

Bedrick, J., Ikerd, W., & Reinhardt, J. (2020). *Level of Development Specification – BIM Forum*. https://bimforum.org/resource/level-of-development-specification/

Borrmann, A., König, M., Koch, C., & Beetz, J. (2018). Building Information Modeling: Why? What? How? In A. Borrmann, M. König, C. Koch, & J. Beetz (Eds.), *Building Information Modeling: Technology Foundations and Industry Practice* (pp. 1–24). Springer International Publishing. https://doi.org/10.1007/978-3-319-92862-3_1

Davtalab, O., Kazemian, A., & Khoshnevis, B. (2018). Perspectives on a BIM-integrated software platform for robotic construction through Contour Crafting. *Automation in Construction*, *89*, 13–23. https://doi.org/10.1016/j.autcon.2018.01.006

Debney, P., Jeffries, P., Christodoulou, A., Titulaer, R., Shepherd, P., Lineham, S., Lazenby, H., Melville, S., & Solly, J. (2022). Advanced Applications in Computational Design. In M. Bolpagni, R. Gavina, & D. Ribeiro (Eds.), *Industry 4.0 for the Built Environment: Methodologies, Technologies, and Skills* (pp. 77–102). Springer International Publishing. https://doi.org/10.1007/978-3-030-82430-3_4

Follini, C., Magnago, V., Freitag, K., Terzer, M., Marcher, C., Riedl, M., Giusti, A., & Matt, D. T. (2021). BIM-Integrated Collaborative Robotics for Application in Building Construction and Maintenance. *Robotics*, *10*(1), Article 1. https://doi.org/10.3390/robotics10010002

Karimi, S., Iordanova, I., & St-Onge, D. (2021). An ontology-based approach to data exchanges for robot navigation on construction sites. *ArXiv Preprint ArXiv:2104.10239*.

Kim, B., & Chin, S. (2016). Parametric Library Components for BIM-based Curtain Wall Design Automation Module. *ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction*, *33*, 1–6. https://www.proquest.com/docview/1823081919/abstract/4918451B7C114B2CPQ/1

McClymonds, A., Asadi, S., Wagner, A., & Leicht, R. M. (2022). *Information Exchange for Supporting BIM to Robotic Construction*. 839–848. https://doi.org/10.1061/9780784483961.088

McClymonds, A., Leicht, R., & Asadi, S. (2023). System Architecture for Supporting BIM to Robotic Construction Integration. In S. Walbridge, M. Nik-Bakht, K. T. W. Ng, M. Shome, M. S. Alam, A. el Damatty, & G. Lovegrove (Eds.), *Proceedings of the Canadian Society of Civil Engineering Annual Conference 2021* (pp. 225–236). Springer Nature. https://doi.org/10.1007/978-981-19-0968-9_18

Monteiro, A. (2016). *VISUAL PROGRAMMING LANGUAGE FOR CREATING BIM MODELS WITH LEVEL OF DEVELOPMENT 400*.

Ramage, M. (2022, April 21). *What Is Computational Design?* https://constructible.trimble.com/construction-industry/what-is-computational-design

Ren, R., & Zhang, J. (2021). A New Framework to Address BIM Interoperability in the AEC Domain from Technical and Process Dimensions. *Advances in Civil Engineering*, *2021*, e8824613. https://doi.org/10.1155/2021/8824613

Sharif, S., & Gentry, R. (2015). *BIM for Masonry: Development of BIM Plugins for the Masonry Unit Database*.

Sharif, S., Gentry, R., Eastman, C., & Elder, J. (2015). *Masonry Unit Database Development for BIM-Masonry*.

Stephans, T., McClymonds, A., Leicht, R., & Wagner, A. R. (2022). Automated material selection based on detected construction progress: 39th International Symposium on Automation and Robotics in Construction, ISARC 2022. *Proceedings of the 39th International Symposium on Automation and Robotics in Construction, ISARC 2022*, 406–413.

Venkatraj, V., & Dixit, M. K. (2022). Challenges in implementing data-driven approaches for building life cycle energy assessment: A review. *Renewable and Sustainable Energy Reviews*, *160*, 112327. https://doi.org/10.1016/j.rser.2022.112327