Bridging BIM and Digital Simulations

Oliver Reinke

Introduction

In the modern world, training via digital simulation and instruction is becoming increasingly commonplace, with various uses from emergency preparation to day-to-day work. Training through the internet and various digital means allows more convenient, cost efficient, and effective training. Many of these simulations are delivered through games developed in Unity or other similar platforms due to their more mature community and flexibility of use.

For the purposes of this paper, we will focus specifically on Unity, due to its widespread use, and tools to enhance development in Unity going forward, alongside other applications of this technology. The main item discussed will be the integration of existing Building Information Models (BIM) into Unity scenes using various tools.

Building Information Models

Building Information Models (BIMs) are digital representations of buildings with varying levels of details. Exact standards haven't been set by the US government or other governing bodies, but broadly, large companies have started to mainly use Industry Foundation Classes (IFC) standards for their projects (buildingSMART International, n.d.). Within IFC specifically, there are still different levels of details between different organizations' standards. Some consist of only structural information and detail the shape of a building, whereas others also will include the electrical, plumbing, HVAC, etc. schema as additional layers to the IFC. While the level of detail and precision may vary depending on the organization, generally they will still contain the

level of information needed for most applications related to training/simulation and digital twins. The most modern IFC standard is 4, and these will contain three dimensional assets needed to create simulations of the detailed buildings. Previous standards will work as well, as shown in the Demonstration section, where an IFC 2x3 standard file is used to create a simulation in Unity. IFC 2x3 standard files are much more available due to being in use for quite some time. However, IFC 4 is compatible with the same tools, so switching between these two should not cause issues.

DoE Use cases

Currently, the prime use case for BIMs will be in the training and simulation space. Through the use of BIMs in the development pipeline, we gain efficiency in both cost and realism. By importing these models into our simulations, we model the targeted building with a high level of accuracy and minimal work. Detailed architectural data does not have to be re-examined and re-modeled for every simulation, instead we can simply reuse the work done by the architects themselves. With the digital sites matching the real-world environment, the training itself becomes more effective, as trainees will be able to experience more realistic simulations. This reduces the disconnect from artificial simulations and the reality that those simulations are trying to convey, resulting in a more immersive experience. The use of BIMs to create simulations also opens up the possibility of training employees on buildings that currently do not exist. Considering that a BIM, at least the base structure layers, is made for new buildings before construction begins, this allows us to begin creating simulations before the building itself is completed. Using this, employees can be trained before the facility itself is completed. This would result in a reduced timeline from building construction to its full-swing operation. With

employees being trained ahead of time, crucial facilities can be in operation as soon as finished, instead of taking time to acquaint workers with the new environment.

A prime example of this use case would be training workers for new semiconductor plants being constructed in the United States. In 2022, the CHIPS act was signed, allocated 52.7 billion dollars for semiconductor research and manufacturing development (CHIPS and Science Act of 2022, 2022). This involves the construction of many new facilities designed for the extremely precise work required to develop modern semiconductors. These types of facilities are nearly always running and require high precision, leaving little room for new workers to be trained on real equipment. Through leveraging virtual simulations, workers can be trained asynchronously to the construction and work timeline. Training in this fashion would enable these extremely important and expensive facilities to be ready to produce on day one, utilizing the investment in these facilities immediately and resulting in expedited results.

Similarly to training, these BIMs can be used to bolster the realism of digital twins and make them easier to create. Using BIMs, a detailed model of a building for a digital twin can be obtained and tweaked with little effort. This allows more in-depth analysis of security and safety protocols in advance, letting professionals do their job more effectively. By creating digital twins in an engine like Unity, we can perform real-time emulation of facilities, allowing better monitoring from security to energy efficiency and maintenance.

BIM to Unity methods

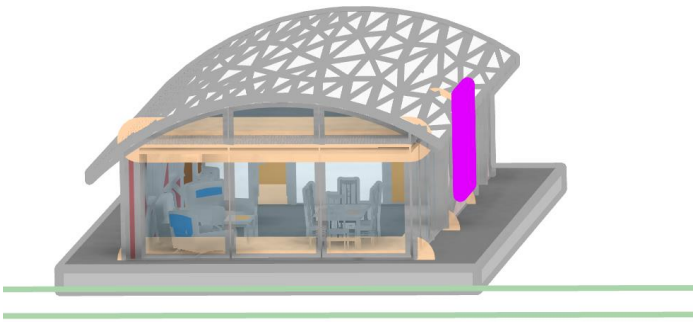
Through research and experimentation, there were two primary pipelines for taking existing BIMs and importing them into Unity. First is the use of Revit. Revit is a very popular and widely used application for development in the BIM space. Transferring models from Revit

to Unity is usually straightforward but offers less direct control to the user. The second approach would be conversion from IFC to Unity objects via open source software, namely IfcOpenShell. This software allows easy conversion from a standard IFC file to a .obj file, which can be directly imported into Unity.

In more detail, Revit is a prime choice for converting models into Unity, as it is already widely in use. Revit was developed by Autodesk and is specifically used to create BIMs. In the software, users can create detailed 3D models of buildings with great detail. Revit allows parametric modeling, which enables a change in one element to affect the elements around it, resulting in dynamic building models. Taking an existing model in Revit and converting it into a Unity object is quite easy as well. To do so, we simply export the structural layer as a .fbx file. Then in Unity we import this .fbx file as an asset. This results in the building being placed as a prefab. From there, we can delete unnecessary elements and tweak the model in Unity itself.

As an alternative to Revit, we can also use IfcOpenShell, an open-source program that, among other things like authoring IFCs, can convert .ifc files into .obj files (IfcOpenShell, n.d.). The main advantage of this method is that the user has total control over the process. The program is open source, allowing alterations to the code and verification of its security. This helps mitigate the risk that comes with passing BIMs of secure facilities to a third-party program. Since we are able to locally run the program and validate its functions in their entirety, we can be far more sure that there are no security threats lying in the program, whereas we cannot assure the same level of security in a program that we do not have this level of access. As with Revit, the conversion process is straightforward. After downloading IfcConvert, all we have to do is drag and drop an IFC file onto the IfcConvert.exe application in file explorer or run the convert

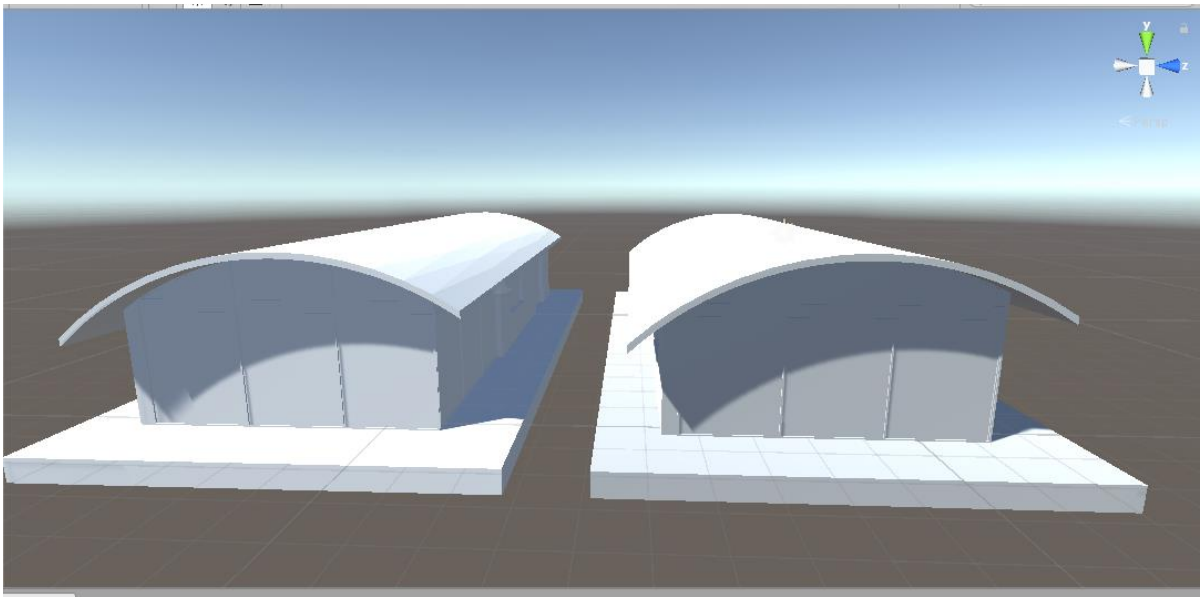
command in our command line. From this, IfcConvert produces a .obj file. This can be directly imported into Unity and function the same as the object produced by Revit.



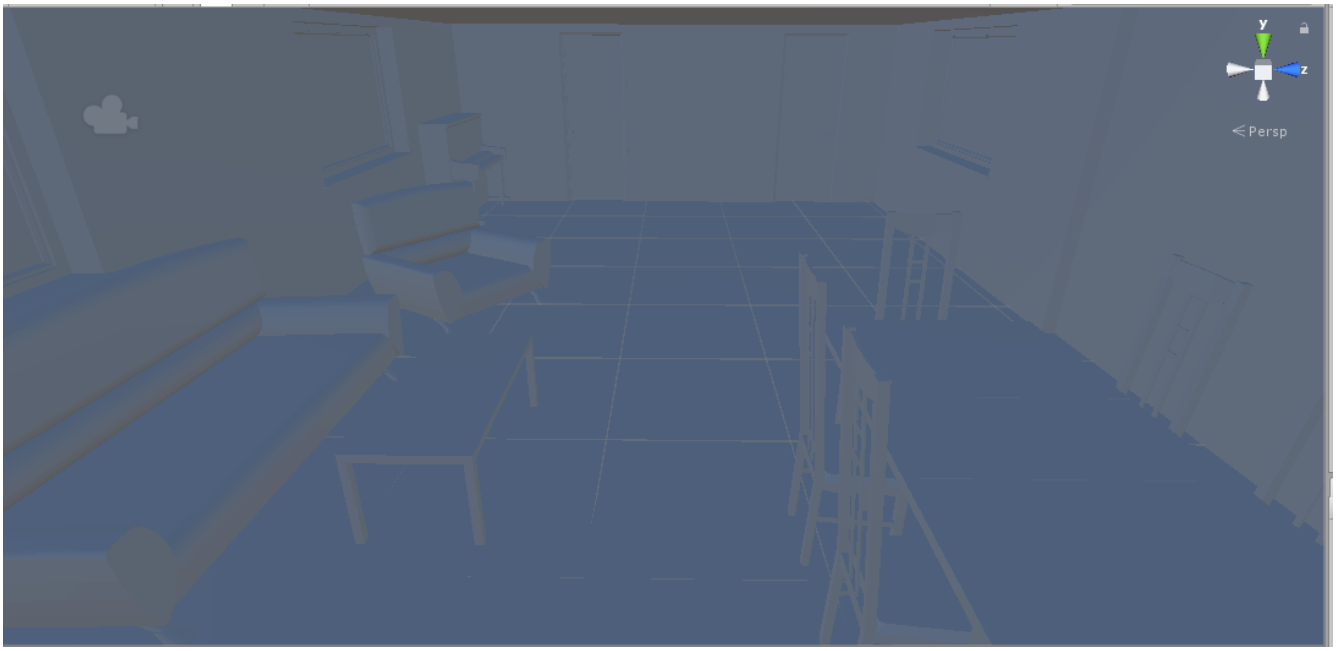
Initial IFC cover image

In my testing, Revit and IfcConvert operated nearly identically. I took an existing IFC 2x3 standard file from github (youshengCode, 2021) and tested using both of the aforementioned pipelines. The initial IFC cover image is pictured to the left.

In Revit, I exported this .ifc file to a .fbx file. For the conversion with IfcOpenShell, I dragged the .ifc file on to the IfcConvert.exe program in file explorer. These were both very quick and painless. After this, I dropped both into a simple Unity 3D template and placed them side by side. After exploring both extensively, there were nearly 0 differences between the two results. Both methods resulted in the same house detailed in the original IFC file, with all furniture items being separate assets that were able to be individually manipulated. The only notable difference was that the IfcConvert method did not preserve asset labels as well. The Revit house preserved the well-labeled assets, with each individual asset having their original labels, making them easier to distinguish, while the IfcConvert house's labels were all erased, with long sequences of seemingly random numbers in their place.



IFC import exteriors. Revit on left, IfcConvert on right.



Ifc import interior.

Either of these two methods are perfectly suitable for use in converting BIMs to Unity scenes. Revit may be preferred in larger IFC where there are several components involved that

will need to be independently edited and tweaked; IfcConvert will not preserve identifying tags for any of these components. However, in cases where a developer does not own Revit or needs to be sure of the application's security, IfcConvert will be more useful, as it is free and open-source, allowing further development to be done on it if required. In the future, we could possibly develop a plugin for Unity to handle IfcConvert inside the application itself, as to not require work outside of Unity. However, IfcConvert is written in C++ and Python, making this difficult, as Unity uses C#.

Lack of Information

Throughout this paper, we have worked under the assumption that we will always have IFC data or some other existing BIM. This, of course, is not always the case. Many older facilities will not have this type of information available. In the circumstances that we do not have an existing BIM, these methods will not be as effective. However, Revit and other authoring tools like IfcOpenShell will still prove useful in these scenarios. These programs are purpose built for architects to model buildings. In most cases, we will still have some sort of 2D blueprint of the building, from which we can create a rough model of a building by importing an image into the authoring software. Inside most authoring software, including Revit, we will have the ability to place model walls on top of the 2D drawing's sketches, and then extrude these walls to our desired height from the simulation. Once a model is created, the same methodology of importing from above can be applied.

In addition, cases where no BIM of the facility is available are likely to decrease in the future. These BIMs are becoming far more common in architecture, especially in larger projects, as they make planning far easier for architects. Alongside this, some government agencies are

pushing for detailed BIMs for their projects, specifically the US General Services Administration (U.S. General Services Administration, n.d.).

Future Work

In the future, there are a few key paths we could pursue to expand our current workflows and findings. First, while both Revit and IfcOpenShell have proven to be very effective at converting BIMs for use in Unity, it would be ideal to integrate these approaches directly into the Unity editor. This way, users could drag and drop an IFC file straight into Unity without needing any external tools or file conversions. One possible route is developing a dedicated Unity plugin that wraps the functionalities of IfcOpenShell, but due to the disparity between Unity's C# environment and IfcOpenShell's C++/Python code, this may require significant effort.

Beyond training simulations, there is the potential to extend this work to fully realized digital twins that incorporate real-time sensor data, or to refine existing IFC data into predictive maintenance models. By combining these models with live data in Unity, we can develop living digital environments, offering real-time monitoring and analytics for critical infrastructure. This would have extensive impacts for facility management, safety, and even energy efficiency, potentially transforming static 3D building models into dynamic assets that continuously reflect real-world conditions.

Conclusion

Overall, importing Building Information Models into Unity offers a highly practical method for rapidly developing detailed simulations and digital twins. By reusing an existing BIM, we save considerable effort that would otherwise be spent on creating building models

from scratch. Revit's pipeline is quite user-friendly and tends to preserve labels effectively, while IfcOpenShell stands out for its flexibility, open-source nature, and auditability. Both conversion methods result in similar final models, enabling realistic simulations of specific sites with minimal manual intervention.

As new facilities are designed and constructed, using these BIM-to-Unity pipelines allows professionals to prepare trainings and test protocols before the buildings are complete. This leads to faster onboarding of new personnel and more seamless day-one operations. Additionally, these same workflows can be extended to digital twins, embedding real-time data into a virtual representation for a range of tasks, like security monitoring, energy optimization, and operational troubleshooting. Moving forward, improvements in model availability, closer integration between BIM tools and Unity, and enhanced security controls will undoubtedly make BIM-based simulations even more robust and indispensable for modern facility planning and management.

References

buildingSMART International. (n.d.). Industry Foundation Classes. buildingSMART. https://ifc43-docs.standards.buildingsmart.org/

National Institute of Building Sciences. (n.d.). Whole Building Design Guide (WBDG). https://www.nibs.org/wbdg

CHIPS and Science Act of 2022, Pub. L. No. 117–167, 136 Stat. 1392 (2022). https://www.congress.gov/117/plaws/publ167/PLAW-117publ167.pdf

IfcOpenShell. (n.d.). IfcOpenShell. https://ifcopenshell.org/

youshengCode. (2021, August 24). Ifc4_SampleHouse.ifc [IFC file]. IfcSampleFiles. https://github.com/youshengCode/IfcSampleFiles/blob/main/Ifc4_SampleHouse.ifc

U.S. General Services Administration. (n.d.). 3D/4D Building Information Modeling. https://www.gsa.gov/real-estate/design-and-construction/3d4d-building-information-modeling