A PLATFORM FOR MULTILAYERED DOCUMENTATION OF CULTURAL HERITAGE

M. Radanovic [*], K. Khoshelham, C. Fraser

Department of Infrastructure Engineering, University of Melbourne, Parkville VIC 3010, Australia – (m.radanovic, k.khoshelham, c.fraser)@unimelb.edu.au

KEY WORDS: Heritage BIM, Spatial dataset integration, Visual fidelity, Semantic richness, Heritage building documentation, Game engine, Point Cloud, Polygonal Mesh.

ABSTRACT:

This paper presents a platform for multilayered documentation of cultural heritage, inspired by the current lack of a heritage BIM approach capable of creating models with both high geometric accuracy and high semantic richness. The platform is developed in the Unity game engine. It comprises several integrated and interconnected layers or datasets that can include data of different types, such as a point cloud, textured polygonal mesh, parametric information model and images, both 2D images and 360° panoramas. We present an overview of the platform concept, the benefits of the proposed multilayered representation and the details on the implementation and integration of datasets. Also, we present some of the innovative functions made possible by this integration, such as point cloud or mesh cutting and preforming measurements on 2D images and 360° panoramas. We perform and present the results of a preliminary analysis of platform functions, which indicates that the platform can be used for accurate measurement and retrieval of 3D coordinates.

1. INTRODUCTION

Technological developments of the past decade have made capturing the geometry of heritage buildings and creating detailed reality-based models more accessible. But, there has recently been an increased demand for models that offer additional information along with the geometry, so these can be used for life-cycle management of heritage buildings. An important recent concept for life-cycle management is Building Information Modelling (BIM). A building information model represents a building by elements that contain geometric attributes, but also employing functional, semantic and topological information (Volk et al., 2014).

BIM has recently been extensively applied to heritage buildings. The high semantic richness of heritage building information models can facilitate day-to-day management, support restoration, and help perform complex analyses and simulations, like facade weathering, structural analysis or valuation of assets (Radanovic et al., 2020). However, creating visually detailed parametric models of often unique and complex heritage buildings is challenging (Baik, 2017). The scan-to-BIM process is highly manual, often relying on automated object recognition from point clouds or shared libraries of historical elements. Most importantly, parametric models are not suitable for applications that require visual fidelity and high geometric accuracy, such as documentation, restoration, exhibitions or virtual tours. For these purposes, highly detailed reality-based polygonal mesh or point cloud models are preferred, but these on the other hand lack the semantics. Ideally, a model composed of geometric elements and semantic attributes would have both high geometric accuracy and high semantic richness. Unfortunately, there is always a trade-off between these aspects and no heritage BIM approach currently offers both.

This paper presents a platform being developed to address this gap using an innovative multilayered concept, where the model is comprised of several integrated and interconnected layers or datasets with data of different types. Preliminary results from analysis of platform functions and layer integration will also be highlighted. Although in an advanced state, the platform is still a work in progress with more functions to be added and analyses to be performed, these being listed in the final section.

Several dedicated heritage BIM platforms have been proposed in the literature. BIM3DSG is an innovative heritage BIM platform presented by Fassi et al. (2015), which was used to create a model of the main spire of Milan Cathedral. BIM3DSG models are stored in online 3D repositories and can be visualised with common web browsers. The system shows very good performance and great flexibility, as it can store previously segmented models of multiple formats, such as point clouds, polygonal meshes or parametric models. Models are stored in a PostgreSQL database and follow a custom hierarchy, and custom data and information can be attached to the elements.

NUBES is another platform for cultural heritage developed by De Luca et al. (2011). Like BIM3DSG, it is a web-based information system that supports point cloud, mesh or parametric models. The interactive 3D scene is connected to a MySQL database containing various heterogeneous data and a semantic descriptor model for the multi-field analysis of the heritage building. The platform offers several specialised tools, for example, a tool for structuring and displaying temporal changes.

The main limitation of existing platforms is in how they manage additional information. Both BIM3DSG and NUBES propose novel ways to describe building elements and to organise and store semantic and other information. In our view, there is no need to reinvent a way to store and share information about buildings because BIM already exists as an internationally accepted paradigm in AEC. As such, it comes with a

[*] Corresponding author.

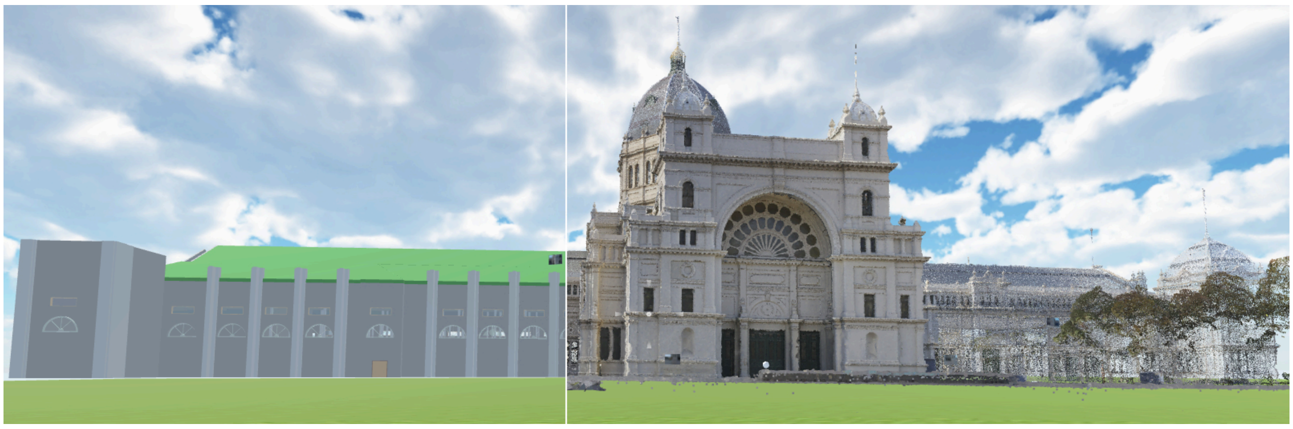


Figure 1. The multilayered model of the Royal Exhibition Building in Melbourne. The figure merges two screenshots made in the developed platform, parametric model on the left and polygonal mesh and point cloud on the right.

standardised structure, standard exchange formats and a wide variety of modelling and other supporting software. Hence, our platform uses BIM to store information and make use of these benefits. This also means that the models don't need to be specifically made for the platform in order to use it, as is the case with the existing platforms. Also, the multilayered representation brings several additional benefits which are discussed in the next section.

2. PLATFORM CONCEPT AND DATASETS

We propose a novel heritage BIM concept that enables the creation of models with both rich semantics and high geometric accuracy. These models have a novel layered structure, with several interconnected layers or datasets comprising a parametric information model, textured polygonal mesh, point cloud and images (both 2D images and 360° panoramas), as indicated by Fig. 1. The different datasets are not only independent individual visualisations but form an integrated system with the ability to exchange information and interact between the layers. For example, if a point cloud of a building is used in the front for visualisation, interacting with a building element such as a column will retrieve all the information about that column stored within a hidden underlying parametric information model. This integration between the layers enables several innovative functionalities, such as mesh and point cloud cutting or precise 3D measurements on images, as presented in Section 3. The concept is based on a platform under development, currently in a pre-alpha stage, which means it is a fully working prototype with all the most necessary functions and content now working. Planned work needed to reach the alpha stage is outlined in the final section. The platform is being developed in a game engine, mainly because game engines have high rendering capabilities and are optimised for visualisation of large 3D models often associated with heritage buildings. After initial consideration, Unity 2019.4 by Unity technologies was chosen for the development. Unity is free for personal use and companies that do not generate large revenue and it is one of the most widely used game engines in the world, with useful built-in tools and wide community support. Importantly, it is also a cross-platform game engine that supports building games and applications for all major target platforms.

The main idea behind the layered concept is to utilise each dataset to its fullest extent, in the way it is intended to be used. So, instead of trying to segment meshes and point clouds and attach information to them in an unstandardized database, they are used here only for purposes related to visualisation. Similarly, instead of trying to make detailed parametric models of high visual fidelity, they are kept visually simple and are used here only as containers for information in the background.

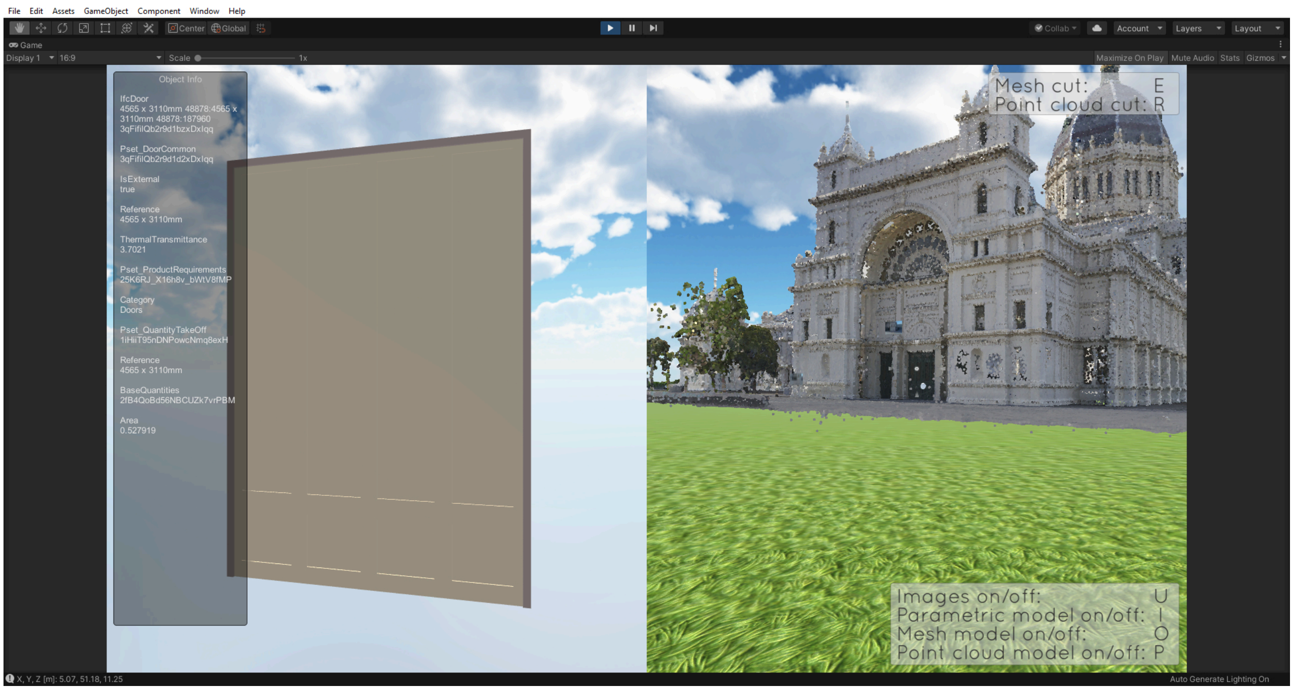
2.1 Benefits of multilayered representation

There are multiple benefits of the proposed layered concept, but perhaps the most important one is that the integrated model can be used both in applications that require high geometric accuracy and in applications that require semantic richness. In other words, it combines the strengths and advantages of reality-based and parametric modelling approaches. It does not rely on shared libraries of historical elements nor does it require point cloud or mesh segmentation algorithms. Moreover, as the parametric model is visually simplified, this concept facilitates and shortens the scan-to-BIM process, which has been recognised as a major bottleneck of heritage BIM by several authors, e.g. Dore and Murphy (2017). It should be noted that, even when simplified, the parametric model can be used to produce engineering drawings such as 2D plans, elevations, and sections, which is an important and often needed functionality of heritage BIM (Murphy et al., 2009).

Another benefit is that storing information within a building information model enables its sharing using standard exchange formats, the dominant one being IFC (Volk et al., 2014), which enables the use of external specialised software for modelling or performing different analysis and simulations. Furthermore, the platform combines often disjointed spatial datasets into a coherent and integrated system, thus increasing their usability. For example, point clouds are often created from images that are afterwards seldom used or not used at all. Similarly, parametric models of existing buildings are usually created from point clouds which are afterwards scarcely used. This raw data is both expensive to produce and holds valuable information, so making efficient use of it is an additional benefit of the proposed platform.

2.2 Dataset implementation

The imperative for implementation of different 3D datasets into the same system is to ensure they are related to the same spatial reference system. If not, the discrepancies in translation, rotation or scale will either make the datasets unsuitable for integration (if large enough), or result in errors caused by misalignment, such as errors in retrieving the data between the datasets (if small enough not to be detected initially). Spatial datasets are usually gathered by land surveying experts that make sure the datasets are properly georeferenced, most



Object Info

IfcDoor
4565 x 3110mm 48878:4565 x 3110mm 48878:187960
3qFifilQb2r9d1bzxDxIqq

Pset_DoorCommon
3qFifilQb2r9d1d2xDxIqq

IsExternal
true

Reference
4565 x 3110mm

ThermalTransmittance
3.7021

Pset_ProductRequirements
25K6RJ_X16h8v_bWtV8fMP

Category
Doors

Pset_QuantityTakeOff
1iHiiT95nDNPowcNmq8exH

Reference
4565 x 3110mm

BaseQuantities
2fB4QoBd56NBCUZk7vrPBM

Area
0.527919

Figure 2. Split screen after the inspection of a building element, showcasing the integration of the front and the back layers in the platform. Information from the IFC file is listed on the left, while movement and inspection remains functional on the right.

commonly in the WGS84 reference coordinate system used by GPS, or in different national or local reference systems, based on the location of the model and user preference. Even if no specific reference system is used, the local 3D Cartesian coordinate system of the point cloud can be used as a reference for the remaining two 3D datasets. The polygonal mesh is created from the point cloud and retains its coordinate system. The parametric model is again created from the point cloud which is used as a kind of a scaffold in parametric modelling software, where building elements are placed on top of it in the scan-to-BIM process. By setting the coordinate system in the modelling software to be aligned with the point cloud, the resulting parametric model retains its coordinate system. It is crucial to make sure the datasets are not translated, rotated or scaled at any point after their creation, because this will lead to problems with their integration.

Coordinates in Unity are represented with floating-point numeric types, so each coordinate has a memory allocation of 4 bytes which limits its precision to around 7 digits. This is not enough to represent spatial data in either a world or a national coordinate system, assuming millimetre accuracy, which is a common issue faced by software dealing with the representation of 3D spatial datasets. However, this can be easily circumvented by applying a global shift to the model's coordinates, which truncates them into a manageable range of 7 digits.

The implementation of images and their integration with 3D datasets rely on camera centre positions and orientations determined by photogrammetry. Different software pieces use different algorithms for this task, most commonly the SIFT algorithm for feature detection and descriptor generation, bundle adjustment for camera pose determination, and dense image matching for point cloud generation. The output point cloud serves as a connection between the adjoining images and the 3D space. If this photogrammetric point cloud is georeferenced or aligned with the laser-scanned point cloud in the photogrammetry software, the adjoining images are also positioned in space and aligned with the rest of the datasets. The photogrammetric point cloud created in this way can be used as one of the datasets, but if there is an existing point cloud created with laser scanning techniques it does not have to be, in which case it serves only to tie the images to the 3D datasets.

Only multi-image photogrammetry datasets can be imported and aligned with the rest of the datasets as there is currently no way, short of pose determination via manual measurement of assigned object control points, to align a single image of a building or a building element. However, this does not mean that a photogrammetric survey of the whole building must be undertaken to be able to import the images. A minimum of two images, but preferably more, with a sufficient overlap are needed. These are used to create a point cloud of the represented part of the building, and this partial point cloud is then georeferenced or aligned with the laser scanner point cloud using control points. The partial point cloud aligns adjoining images so their position and orientation can be used for the integration.

It is important to keep in mind that game engines can only visualise polygonal meshes and don't have natural support for point clouds or volumetric solids representation, as used by parametric models stored in IFC format (Khoshelham et al., 2018), so point clouds and parametric models need first to be converted to meshes. This specialisation and heavy optimisation for meshes is the main appeal of using game engines to visualise spatial data.

Unity does not natively support point clouds as it interprets points as non-dimensional, hence invisible, so they must be converted into a readable mesh-based format. Point clouds are visualised using Potree, a free and open-source WebGL based rendering system for large point clouds created by Schütz (2016), which Fraiss (2017) imported into Unity and made the package open source. The conversion of a point cloud from LAS (LASer) or LAZ (compressed LAS) format to Potree octree structure is made by an open-source Potree converter 1.6 (Schütz, 2016) and can be performed at runtime. Thus, the point cloud is visualised by attaching a small piece of mesh to each of the points. The size and the shape can be adjusted, and the piece is always oriented towards the viewer. But, perhaps the most important benefit of using Potree is the optimisation. A point budget can be set, which defines the maximum number of points being displayed at any specific location, so even very large point clouds can be visualised on limited-capacity hardware.

Similarly, the geometry of a parametric model in IFC4 or IFC2x3 format must be converted into a mesh-based geometric representation readable by Unity. A command-line application IfcConvert 0.6.0 from IfcOpenShell (Krijnen and Beetz, 2020), which is an open-source IFC toolkit and geometry engine, is used to convert both the geometry and non-visual information from IFC while retaining their GUIDs (Globally Unique Identifiers). The geometry is converted into a Wavefront OBJ mesh, which is natively supported by Unity, and a free C# package from Unity Asset Store is used to import the mesh at runtime[1]. The information is converted into an XML file, which is parsed using a set of custom written C# scripts[2] that also extract and organise IFC Property and Quantity sets. Furthermore, these scripts attach the data to the geometry of building elements using IFC GUIDs and they organise the flat list of entities into a hierarchy commonly used by IFC viewers. An example of an imported parametric model can be seen in Fig. 1, while Fig. 2 shows an inspected building element along with the attached information.

3. PLATFORM STRUCTURE AND FUNCTIONS

The model can be navigated in 1st person view, with a crosshair in the middle used for selection and the rest of functions accessible via keyboard button commands. In the case of the Royal Exhibition Building (REB) used as an example in this paper, the model is split into exterior and interior, but this depends on the size and the specifics of a building (for example, some models may not have the interior at all, while others may have multiple storeys of the interior). Exterior and interior are represented by two Unity scenes and the user can traverse between them by interacting with the orb located on the doors of the building, but the scenes can also be shown at the same time if required. The visibility of different layers or datasets can be toggled by pressing the corresponding keyboard button, per the instructions on the UI that can be seen in the bottom right of Fig. 2. Toggling the visibility of images on or off toggles the visibility of planes and spheres that serve as containers for 2D images and panoramas shown in Fig 3.

3.1 Accessing the background IFC data

Everything present within the Unity scene is a game object, which is defined as a base class for all entities in the scene. Game objects contain components, such as the Transform component that defines its position, orientation and scale; a Mesh Renderer component that deals with the rendering of the game object on the screen; a Mesh Collider component that deals with the physical collisions, and so on, depending on the needs and purposes of the specific game object. When the visibility of any of the 3D datasets in the platform, namely point cloud, polygonal mesh or parametric model, is toggled on or off, only the renderer components of the corresponding game objects are turned on or off. In other words, the datasets are still present as game objects within the scene and, while invisible, can still be interacted with. This functionality allows for innovative integration of 3D datasets. A semantically rich but visually simplified parametric model can be put in the back, or more precisely made invisible but still accessible, while a visually detailed dataset with high visual fidelity and geometric accuracy, such as a mesh or a point cloud, can be put in the front for visualisation.

Fig. 2 illustrates the integration of the front and back layers or datasets. Here, a point cloud of the REB exterior is used in the front for visualisation. Clicking on a building element of interest on the point cloud identifies the corresponding IFC building element that holds all the information related to this element. Interaction with the parametric model is possible because, although it has its Mesh Renderer component turned off, the Mesh Collider component is turned on and still occupies the exact position of the object in space. Upon inspection of the building element, a vertical split-screen is brought up, with the element in question shown on the left-hand side along with all its data stored within the IFC file. The right-hand side remains a functional 1st person view, where the user can traverse the model and inspect another element. The split-screen can be turned off to return to a full-screen 1st person view, or a mesh or point cloud cutting can be performed by pressing the corresponding keyboard buttons listed in the top right corner of the user interface.

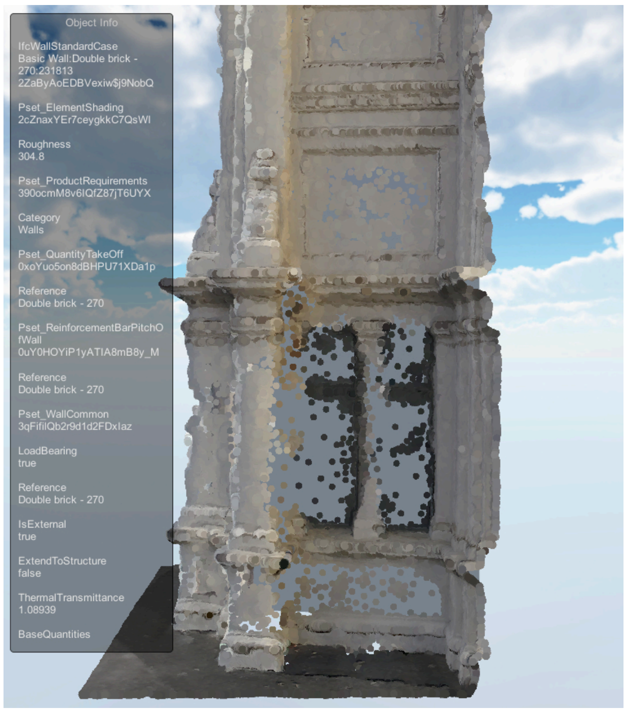
How do we know if the correct building element is retrieved? As mentioned before, the 3D datasets are neither automatically nor manually aligned with each other, they simply retain the placement they had in their respective reference systems and it is up to the user to ensure they share a common system. So, the precision of integration of the parametric model with the rest of the datasets depends solely on how precisely the building elements were aligned with the point cloud in the BIM modelling software during the scan-to-BIM process. If there was a discrepancy in the placement of a building element, this discrepancy is carried over to the platform and can lead to incorrect element retrieval along the edges between the elements.

3.2 Mesh and point cloud cutting

When a part of a building is changed, updating the mesh or the point cloud model usually involves capturing a whole set of data. An alternative is to segment a mesh or a point cloud into building elements. However, this is known to be a challenging task, which in spite of numerous recent improvements, often struggles when dealing with basic elements in cluttered or complex environments. It is still widely agreed to be both in the early stages of development and a largely manual process (Dore and Murphy, 2017; Thomson and Boehm, 2015). The alignment of 3D datasets in the developed platform enables an innovative approach to mesh and point cloud segmentation and cutting. Instead of performing a challenging segmentation, the fact that the parametric model is constructed out of individual building elements is used. In other words, there is no need to segment the mesh or a point cloud when the parametric model is already segmented. When they are aligned, parametric building elements are used to isolate corresponding parts of a mesh or a point cloud, which can be used to segment them and only replace the changed segments with new data. This alleviates the need for expensive and labour intensive capture over the whole building.

[1] Runtime OBJ Importer, free Unity asset. https://assetstore.unity.com/packages/tools/modeling/runtime-obj-importer-49547 (5/1/2021).

[2] Getting BIM data into Unity. http://cad-3d.blogspot.com/2018/09/getting-bim-data-into-unity-part-9.html (5/1/2021).


Object Info

IfcWallStandardCase
Basic Wall:Double brick - 270:231813
2ZaByAoEDBVexiw$j9NobQ

Pset_ElementShading
2cZnaxYEr7ceygkkC7QsWI

Roughness
304.8

Pset_ProductRequirements
390ocmM8v6IQfZ87jT6UYX

Category
Walls

Pset_QuantityTakeOff
0xoYuo5on8dBHPU71XDa1p

Reference
Double brick - 270

Pset_ReinforcementBarPitchOfWall
0uY0HOYiP1yATIA8mB8y_M

Reference
Double brick - 270

Pset_WallCommon
3qFifilQb2r9d1d2FDxlaz

LoadBearing
true

Reference
Double brick - 270

IsExternal
true

ExtendToStructure
false

ThermalTransmittance
1.08939

BaseQuantities

Figure 3. The point cloud of a wall building element after the point cloud cutting function is performed. Only the left-hand side of the split-screen is shown.

After the inspection of a building element in the developed platform, as shown in Fig. 2, mesh or point cloud cutting can be performed by pressing the corresponding keyboard buttons listed in the top right corner. The result of one such cutting is presented in Fig 3, where a point cloud is cut based on the inspected wall element. The cutting is done with a custom written C# script. A modifiable buffer, by default 20 cm, is added around the bounding box of the inspected building element, after which all the points of a point cloud within this space are found, returned and visualised on the left-hand side of the user interface, as shown in Fig. 3. The process for mesh cutting is similar, with the addition of mesh faces being identified along with the points. The precision of the cut segment again depends on the precision with which the parametric model was placed during the scan-to-BIM process, where misalignment can result in wrong parts of the point cloud or mesh being cut. Also important is the size of the buffer, as a large buffer will encapsulate surrounding building elements.

3.3 Image integration and measurements

As mentioned in the previous section, the integration of images within 3D datasets is based on the camera centre positions and orientations determined using photogrammetry software. A plane containing a 2D image of the exterior of the REB and a sphere containing a 360° panorama image of the interior are shown in Fig. 4, floating in space in the place they were recorded from. The containers can be turned on or off like any of the remaining layers or datasets of the multilayered model. However, the information about position and orientation only is not enough, and to fully integrate 2D images the FOV (field of view) of the camera used to take the images must be replicated by the camera in Unity. In that way, the image of the virtual world of the scene in Unity that is projected onto the screen exactly replicates the image of the real world captured by the camera. The integration and alignment of an image with 3D datasets is illustrated in Fig 5, where only the bottom half of an image is shown while the top half shows the virtual world behind it, aligned with the image, containing a point cloud.

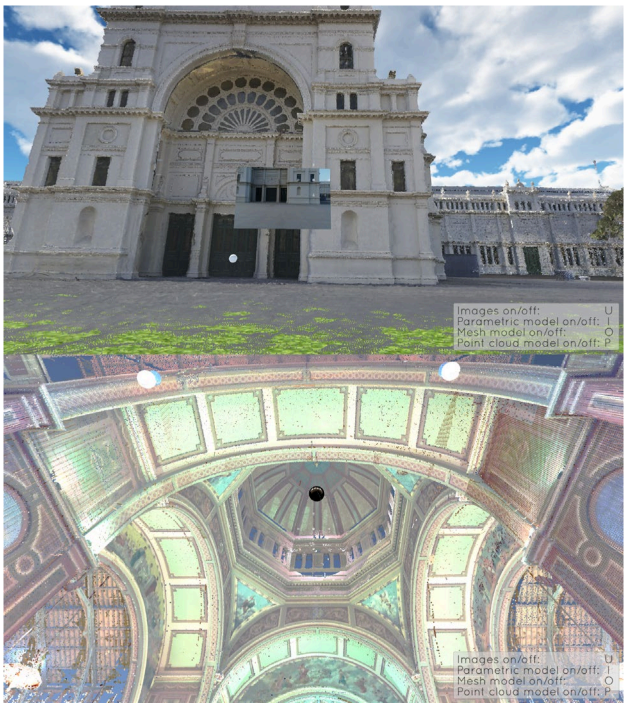

Images on/off: U
Parametric model on/off: I
Mesh model on/off: O
Point cloud model on/off: P
Images on/off: U
Parametric model on/off: I
Mesh model on/off: O
Point cloud model on/off: P

Figure 4. Images positioned in space based on where they were taken from. Plane with a photograph on top, sphere with a 360° panorama on the bottom.

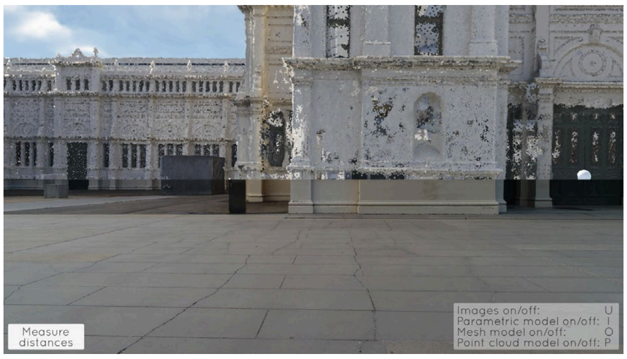
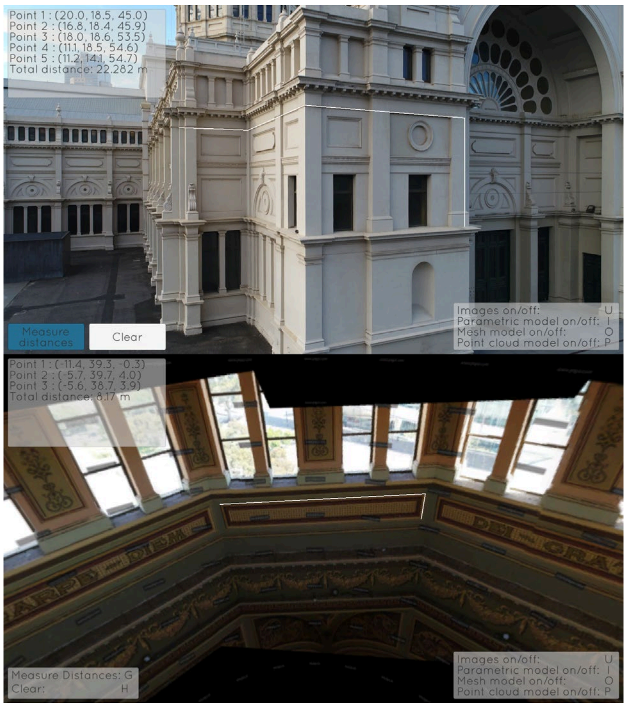


Figure 5. The integration of an image with a point cloud. Only the bottom half of the image is shown to better illustrate the integration and the 3D space behind it.

When a 2D image container is interacted with in the multilayered model, as indicated in Fig. 4, a full-screen view of that image is displayed. The image is interactive and clicking anywhere on the building returns the 3D coordinates of the corresponding point in the model's reference system. Without it being apparent, the user is not interacting with the image but with the underlying 3D dataset aligned under it, and the 3D coordinates returned are the coordinates of the intersection of a polygonal mesh with the ray cast in the direction of a click. In other words, the image can be thought of as a sort of a curtain over the 3D world, with which the user in fact interacts. In addition to retrieval of coordinates, this integration can be used for the precise measurement of distances, as shown in Fig. 6. Clicking a button in the bottom left corner brings up the panel in the top left, where the coordinates of the points clicked on are listed along with the total distance between these points. The points and the line that connects them are drawn on top of the image.

The integration of 360° panoramas is similarly performed. Upon interaction with the sphere containing a panorama, as shown on the bottom part of Fig. 4, the camera is transferred at the centre of the sphere, which uses a special custom written shader to render the panorama on the inside of the sphere. As with the 2D images, 3D coordinates can be retrieved by clicking on a panorama and the distances can be measured. These are again based on the intersection of a ray, cast from the centre of the sphere in the direction of a click, with the polygonal mesh model, invisible behind the sphere's surface.


```
Point 1 : (20.0, 18.5, 45.0)
Point 2 : (16.8, 18.4, 45.9)
Point 3 : (18.0, 18.6, 53.5)
Point 4 : (11.1, 18.5, 54.6)
Point 5 : (11.2, 14.1, 54.7)
Total distance: 22.282 m
```

```
Point 1 : (-11.4, 39.3, -0.3)
Point 2 : (-5.7, 39.7, 4.0)
Point 3 : (-5.6, 38.7, 3.9)
Total distance: 8.17 m
```

Figure 6. Distance measurements performed on images, 2D image on top and on a 360° panorama on the bottom.

4. PRELIMINARY ANALYSIS

The preliminary analysis tests the accuracy of 3D coordinates retrieved with the developed platform. The analysis is based on the comparison of coordinates of points retrieved within the platform with the coordinates of the same points retrieved with the popular Cloud Compare 2.12 (2020) software system.

When the multilayered model is traversed with the point cloud turned on for visualisation, clicking anywhere on a building retrieves the corresponding 3D coordinates. However, the retrieved coordinates are not from the point cloud as points are non-dimensional in Unity, but they are the coordinates of the intersection with the aligned polygonal mesh. The mesh has the Mesh Renderer turned off and the Mesh Collider turned on, or in other words, it is invisible but still present and interactive in the scene, like the parametric model described in the previous section. So, no matter which of the layers is turned on as the front layer, the retrieved 3D coordinates are always from the intersection with the underlying mesh layer. In the case of the REB model, both the mesh and the point cloud are photogrammetric.

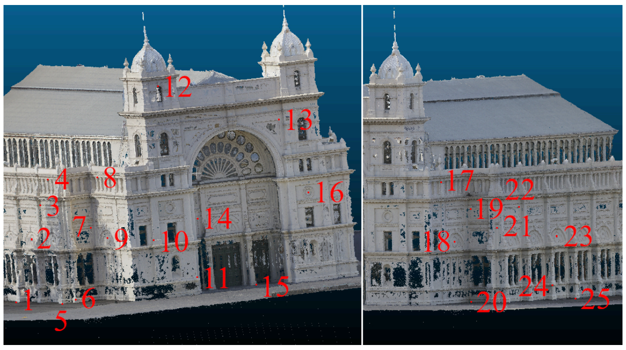


Figure 7. The distribution of 25 points used for the comparison of point cloud coordinates.

The comparison of point cloud 3D coordinates from the platform to coordinates from Cloud Compare can show the accuracy of the coordinates retrieved with the platform and the reliability of the 3D coordinates retrieval function. In this case, the coordinates from Cloud Compare can be considered ground truth, and the difference between them and the coordinates retrieved with the developed platform are errors, labelled *e*. Readings of coordinates are rounded to centimetres. Errors are calculated for 25 points evenly distributed along a part of the REB model, as shown in Fig 7. Basic statistical indicators of errors *e* along each of the coordinate axes are given in Table 1. It can be seen that errors are of relatively small magnitude, with the MAE (Mean Absolute Error) below 3 cm and RMSE (Root Mean Square Error) below 5 cm. There are no significant differences between the values of indicators for different axes. We assume that errors are caused by two main factors. First, although in this case both the mesh and the point cloud are photogrammetric, the mesh never perfectly fits the point cloud, and the discrepancy depends on the number of faces in the mesh and the meshing process. Second, the platform currently has no zoom-in option, which certainly led to errors caused by misaligning the crosshairs and the points of interest.

Indicator [cm]	X	Y	Z
Min. *e*	-8.0	-11.0	-5.0
Max. *e*	13.0	5.0	8.0
MAE	3.0	2.2	2.4
RMSE	4.5	3.2	3.4

Table 1. Basic statistical indicators for point cloud coordinate errors.

Furthermore, the achieved accuracy indirectly indicates a good alignment and integration of point cloud and mesh layers, as the point cloud coordinates retrieved by the platform in fact come from the intersection with the underlying mesh. It should be noted that, if the point cloud and the mesh come from different sources, for example, a laser scanner point cloud and a photogrammetric mesh, a discrepancy between them would cause the 3D coordinates retrieved by the platform to be unreliable.

5. CONCLUSION AND FUTURE WORK

In this paper, we presented a novel heritage BIM platform inspired by the current lack of an adequate heritage BIM approach offering both high geometric accuracy and high semantic richness. While still under active development, the platform has already achieved most of its design goals. The platform successfully integrates several types of spatial datasets into an interconnected multilayered model that offers high visual fidelity in the front, with the parametric information model in IFC format in the back. We have presented details of the integration of datasets and described the innovative functions made possible by the integration.

We performed the preliminary analysis of platform functions. Based on the results of analysis of point cloud coordinates, it can be concluded that the measurement and retrieval of point cloud and mesh coordinates within the platform is accurate. This also indicates that the integration of a point cloud and the mesh is correct. The accuracy of measurements performed on 2D images and 360° panoramas, which would also indicate the alignment accuracy of images and 3D models, is to be analysed in future work.

An important aspect of any 3D modelling platform is performance, as the visualisation and processing of large and detailed 3D models are known to be computationally expensive. The platform currently runs the REB model at 30 frames per second on a mid-range laptop without a dedicated GPU. This is enabled by various processing acceleration techniques, such as the maximum number of points of a point cloud to be visualised being an adjustable parameter, or the frustum culling that makes sure only the visible objects in front of the camera are rendered. However, as future work, it would be beneficial to undertake a full performance analysis using different hardware and more complex models.

As mentioned, the platform is a fully functional prototype in a pre-alpha stage of development. To reach the alpha stage, marked by the presence of all essential functions and general test-readiness, several additional functions and polishing are planned: the addition of a top-down camera view as an alternative to the existing 1st person view, a function to annotate the model with text or images, the addition of the main menu, and an overhaul of the user interface. There are also several optional additions and modifications: a function for visualisation of temporal changes to the building, expansion of the distance measurement function on images to include area measurements, and restructuring the code and scripts so the platform can be posted in an open-source repository.

The alignment of a point cloud and a parametric information model of a building inside of a game engine opens up an interesting opportunity for a novel camera pose estimation method in Augmented Reality (AR). Most current AR applications use various camera (image) based methods for camera pose estimation, either marker-based or markerless, which often perform poorly in varying light conditions and outdoors, and are generally constrained to a smaller area or several fixed locations. Recently, we have witnessed considerable enhancements in real-time automated point cloud matching due to developments in Machine Learning algorithms, mainly for autonomous driving applications. These developments might potentially be utilised for an innovative camera pose estimation method based on the alignment of the point cloud gathered by a wearable AR head-mounted display with the existing point cloud of the building. Alignment of the real world and the virtual world would also position the parametric building information model, resulting in a real-world building information model.

ACKNOWLEDGEMENTS

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors. The first author acknowledges the financial support from the University of Melbourne through Melbourne Research Scholarship.

REFERENCES

Baik, A., 2017. From point cloud to Jeddah Heritage BIM Nasif Historical House – case study. *Digital Applications in Archaeology and Cultural Heritage*, 4, 1–18. https://doi.org/10.1016/j.daach.2017.02.001.

CloudCompare (version 2.12) (2020). GPL software. http://www.cloudcompare.org/.

De Luca, L., Busayarat, C., Stefani, C., Véron, P., Florenzano, M., 2011. A semantic-based platform for the digital analysis of architectural heritage. *Computers & Graphics*, 35, 227–241. https://doi.org/10.1016/j.cag.2010.11.009.

Dore, C., Murphy, M., 2017. Current State of the Art Historic Building Information Modelling. *Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci.*, XLII-2/W5, 185-192. https://doi.org/10.5194/ispr-archives-XLII-2-W5-185-2017.

Fassi, F., Achille, C., Mandelli, A., Rechichi, F., Parri, S., 2015. A New Idea of Bim System for Visualization, Web Sharing and Using Huge Complex 3d Models for Facility Management. *Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci.*, XL-5/W4, 359–366. https://doi.org/10.5194/isprsarchives-XL-5-W4-359-2015.

Fraiss, S. M., 2017. Rendering Large Point Clouds in Unity. Bachelor's Thesis. Vienna University of Technology, Vienna. https://github.com/SFraissTU/BA_PointCloud (7/1/2021).

Khoshelham, K., Tran, H., Díaz-Vilariño, L., Peter, M., Kang, Z., Acharya, D., 2018. An evaluation framework for benchmarking indoor modelling methods. *Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci.*, XLII–4, 297–302. https://doi.org/10.5194/isprs-archives-XLII-4-297-2018.

Krijnen, T., Beetz, J., 2020. An efficient binary storage format for IFC building models using HDF5 hierarchical data format. *Automation in Construction*, 113, 103134. https://doi.org/10.1016/j.autcon.2020.103134. http://ifcopenshell.org/ifcconvert (6/1/2021).

Murphy, M., McGovern, E., Pavia, S., 2009. Historic building information modelling (HBIM). *Structural Survey*, 27 (4), 311-327. https://doi.org/10.1108/02630800910985108.

Radanovic, M., Khoshelham, K., Fraser, C., 2020. Geometric accuracy and semantic richness in heritage BIM: A review. *Digital Applications in Archaeology and Cultural Heritage*, 19, e00166. https://doi.org/10.1016/j.daach.2020.e00166.

Schütz, M., 2016. Potree: Rendering large point clouds in web browsers. Master's Thesis. Vienna University of Technology, Vienna. https://github.com/potree/PotreeConverter (7/1/21).

Thomson, C., Boehm, J., 2015. Automatic Geometry Generation from Point Clouds for BIM. *Remote Sensing*, 7, 11753–11775. https://doi.org/10.3390/rs70911753.

Volk, R., Stengel, J., Schultmann, F., 2014. Building Information Modeling (BIM) for existing buildings - Literature review and future needs. *Automation in Construction*, 38, 109–127. https://doi.org/10.1016/j.autcon.2013.10.023.