Automating the Integration of 4D Models in Game Engines for a Virtual Reality-Based Construction Simulation

Simon Bourlon and Conrad Boton(✉)

École de technologie supérieure, 1100, rue Notre-Dame Ouest, Montreal, Canada
simon.bourlon.1@ens.etsmtl.ca, conrad.boton@etsmtl.ca

Abstract. This article is part of the first steps of a more global work in progress. It aims at automating the integration of 4D simulations in virtual reality environments, to easily generate 4D simulation alternatives in such environments. The final goal is to support collaborative constructability analysis sessions based on intuitive simulations of the construction process using BIM models. The results show that while it is possible to create 4D models in a game engine directly from a BIM model and an existing schedule, several improvements still need to be made in order to achieve the desired situation. Similar to most workflows presented in the literature, the one used in this work is not free of limitations. The evaluation of the application shows that most of the problems encountered come from the imperfect transfer of the model from the BIM software to the game engine.

AQ1

Keywords: 4D simulation · Virtual Reality · Game engines · Building Information Modeling

1 Introduction

The Building Information Modeling (BIM) approach is increasingly being adapted in the architecture, engineering, construction and operation industry [1, 2]. 4D simulation is one of the BIM uses recognized by professionals as having the greatest benefit [3]. It consists in associating the objects of a three-dimensional model with the corresponding activities stemming from a work schedule, in order to simulate the construction process over time [4]. Several benefits may arise from the use of 4D simulation, including Analysis of constructability and activity scheduling alternatives [5, 6], communication of ideas, facilitating collaborative work and education [7–10], logistics, site development and excavation [11–13]. Despite these perceived benefits, the adoption and implementation rates of 4D simulation remain low [3]. The main reasons for this low penetration are the lack of adaptation of visualization methods and interaction techniques proposed by the current tools [4, 14].

Recent research works have proposed using Virtual Reality (VR) to overcome some of the limitations of current 4D simulation approaches, and to provide more intuitive and efficient simulation environments for practitioners [14]. 'Virtual Reality' can be referred to as "electronic simulations of environments experienced via head-mounted

Y. Luo (Ed.): CDVE 2019, LNCS 11792, pp. 1–10, 2019.
https://doi.org/10.1007/978-3-030-30949-7_14

eye goggles and wired clothing enabling the end user to interact in realistic three-dimensional situations" [15]. In this context, four elements have to be considered and combined: the virtual world, the immersion, the sensory feedback, and the interactivity [16].

To be able to successfully conduct a 4D simulation in VR environments, many challenges have to be addressed [14]. One of these challenges is related to the preparation and the transfer of the 4D simulation in VR environments. While the transfer of 3D models in VR environment is very well mastered and some tools are more and more proposed to transfer existing 4D models, the resulting simulation can be used only for visualization purposes as-is. However, to be helpful in most of the practitioners' usages including constructability analysis, the 4D simulation should be editable and the users should be able to automatically generate different alternatives of the simulation.

The work presented in this article aims at automating the integration of 4D models in game engines for a virtual reality-based multi-purpose construction simulation. The rest of the paper is organized in four main sections. The first section presents a literature review on Virtual Reality in architecture, engineering and construction, and the transfer of 4D simulations in VR environments. The second section reports the process of development of an application for the automated integration of 4D models in virtual reality environments. The third section proposes a discussion and the improvements track and the fourth section concludes the article.

2 Related Works

2.1 Virtual Reality in Architecture, Engineering and Construction

Using Virtual Reality in construction is not really new. In 1996, the use of VR applications in the construction industry has been investigated by Bouchlaghem et al. [17]. The investigation showed the usefulness of VR technologies both in design (for landscaping, to assess fire risk, to design lighting and interior design, etc.) and in construction (to evaluate construction scenarios, to plan and to monitor the construction processes, and to plan and layout the construction site) [17]. Berg and Vance, in 2017 [18], proposed an evaluation of the use of VR as a decision-making tool in engineering-focused businesses. The results showed that VR is now "mature, stable, and, most importantly, usable" [18]. Moreover, Virtual Reality has been presented by Issa [19] as a solution for integration in the construction industry, since "a virtual reality platform supported by knowledge-based database systems can become the main interface to construction information for every specialty throughout the construction (life) cycle of the project" [19]. The practitioners in construction generally use VR "on large complex projects and on small projects with design reuse" [20]. The use it on large complex projects, to visualize and to understand the engineering problems, with the aim to reduce uncertainty and risk [20]. In the other hand, they tend to use it on small projects with cheaper web-based equipment for more focused usages [20]. Woksepp and Olofsson [21] also assessed the applicability of VR on large construction projects. They showed that VR was useful, but need to be associated with a sufficient workflow in order to minimize waste and to improve the final result [21].

A quantitative evaluation of immersive VR systems capabilities in helping to understand virtual mockups has been proposed by Paes et al. [22]. They showed "an overall better spatial perception of the virtual model when using the immersive environment" [22] and how VR environments can "benefit current design practices by improving professionals' understanding of the spatial arrangement of the virtual model" [22]. Goulding et al. [23] proposed a prototype of an interactive VR environment, The aim is to support the learning of the challenges related to offsite construction. With the proposed system, users can identify some of the "unforeseen problems often caused by professionals' decisions, faulty work, and health and safety issues" [23].

2.2 Transferring 4D Simulations in VR Environments

Transferring 4D simulations in VR environments consists of preparing the data in an appropriate format for their suitable use in the VR systems. Usually, one uses game engines to prepare the file in the required format. Multiple game engines are available, including Unity, Quest 3D, CryEngine, Blender, Unreal, Torque 3D, etc. and it is possible to compare them regarding their functionalities and features [24, 25]. A model export workflow from Revit to Unity 3D has been proposed by the Penn State CIC Research Group [26]. It suggested to prepare the model in Revit and to export it in FBX format. Then, this FBX file should be imported and optimized in Autodesk 3DS Max. One can also use a paid script can also be used to convert Revit textures into standard material used by 3DS Max and readable by virtual reality software. Ultimately, the model can be imported as a new asset in Unity 3D.

While this is interesting, it is not adapted to the transfer of 4D simulations, as both 3D geometry and sequencing information are contained in 4D models. Broadbent [27] presented a method based on Autodesk authoring tools for the transfer of 4D models in game engines. The method proposed to export the geometry directly from Revit to 3DS Max by the use of a plugin based on the DWFx format. In the other hand, the temporal information is exported by using an XML, directly from Navisworks to 3DS Max, using a Timeline exporter plugin. It is then possible, according to Broadbent [27], to export the 4D model from 3DS Max to Unity based on FBX and Timeline files. Note that the steps of the method are only poorly described and do not provide sufficient information about how the timeline is actually transferred.

More recently, Boton [14] explored a similar workflow to transfer a 4D simulation in a VR environment. The result was not very conclusive since "all the objects start their path at 0, 0, 0 despite starting in their original locations in 3DS Max".

3 Development of an Application for the Automated Integration of 4D Models in Virtual Reality Environments

3.1 Data Preparation

The first step is to import the data that will create the 4D simulation. In the literature review, we saw that the import of the 3D model from Revit should be done in FBX format. This format allows the model to be used on Unity 3D. In order to keep the texture rendering on Unity 3D, an intermediate step was needed. It was necessary to first import the Revit model into 3DSMax and then export it in FBX format. This system made it possible not to lose visual information. However, the problem was that the tree of the elements was too complicated to manage by the code. The subobjects of the model were classified in subgroups. The code would have needed a lot of loops to recover the elements one by one. To retrieve an element in Unity from its name, it is necessary to find the "parent" GameObject and after searching among the "child" GameObjects the one who has this name. If we consider our model in full at step 0 of the tree, step 1 is then the differentiation of the stages for example, then step 2 lists the types of object present on this stage (example: ripe, doors, stairs …), this up to the level N where we find the elements named exactly as in the planning of construction. Only this would require at the code level additional iterations to at least know the value of the N scale where to look for and then go through each loop by associating the elements.

We then tested to export our model in FBX format from Naviswork. The representation of the tree was then much easier to manage. Sub-objects are not classified in subgroups. The GameObjects "child" are directly at level 1, in the case where we consider step 0 at the level of the main GameObject. The retrieval of the elements by the code was then simplified, we did not need different paths loops in search of the elements corresponding to the elements of the imported construction schedule. However, it should be noted that the rendering of textures is not optimal. The schedule is imported into Unity from an Excel file, in .csv format. Note that Unity does not take into account special characters. This requires changing one by one these elements in the text. Otherwise, it is possible to use a modification on Excel that automatically replaces all these special characters.

3.2 Automating the Generation of 4D Simulations in Unity 3D

Codes that can be used by Unity must be written in C# language. For the application, it was made the choice that all the calculations, all the functions necessary for the creation of the 4D simulation are written in the same code. Thus, the user has all the elements in a single code file from which, he can access any coding changes that may be necessary. This choice has imposed a rigour in the organization of writing to simplify reading. In this part, we present each element that constitutes this code. The chosen organization, following the constraints of this type of language, is represented in Fig. 1.

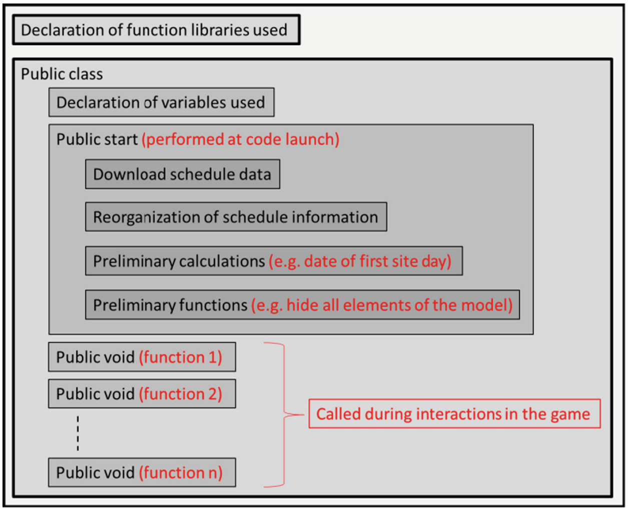

Declaration of function libraries used
Public class
Declaration of variables used
Public start (performed at code launch)
Download schedule data
Reorganization of schedule information
Preliminary calculations (e.g. date of first site day)
Preliminary functions (e.g. hide all elements of the model)
Public void (function 1)
Public void (function 2)
Public void (function n)
Called during interactions in the game

Fig. 1. Organization of the code of the application.

The first step performed at the launch of the game is the download of data related to the construction schedule. The information is to be retrieved in the "Resources" folder by means of a text file named "Planning". Then this information is reorganized in a format suitable for the following manipulations. This reorganization process is detailed in Fig. 2. The text file is transformed into a list consisting of a string list.

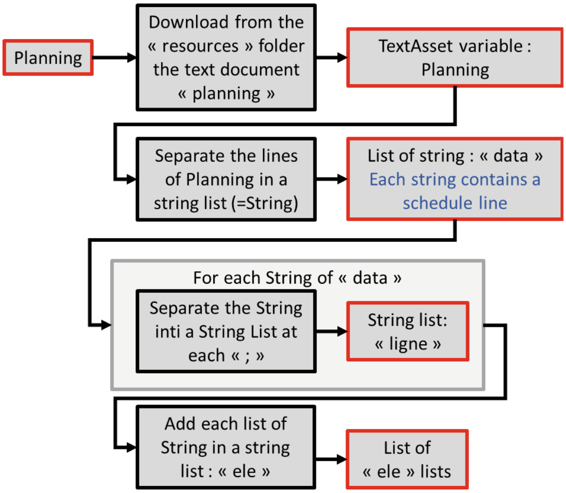

Planning
Download from the « resources » folder the text document « planning »
TextAsset variable : Planning
Separate the lines of Planning in a string list (=String)
List of string : « data »
Each string contains a schedule line
For each String of « data »
Separate the String inti a String List at each « ; »
String list: « ligne »
Add each list of String in a string list : « ele »
List of « ele » lists

Fig. 2. Path of the program to transform the text file Planning in the list "ele".

From this list “ele”, several initial calculations are possible. A first manipulation allows to create two lists “deb” and “end” respectively containing the dates of beginning of construction and the dates of end of construction of the tasks in the same order as that of the tasks in the list “ele”. Then, it is possible to determine the start date of the construction and the end date of the construction. The duration of the project is then deduced. Two actions are allowed thanks to this information. The date displayed in the application is then updated on the first day of construction. As well as the length of the scale of the cursor is initialized on the duration of the building site. For example, if the difference between the first day of construction and the last day is 100 days then the cursor will have 100 possible values. Figure 3 summarizes the principles underlying this part of the program.

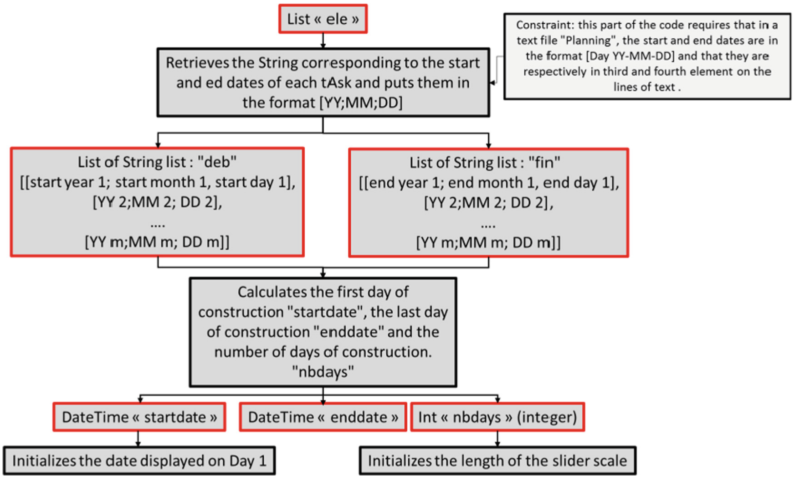
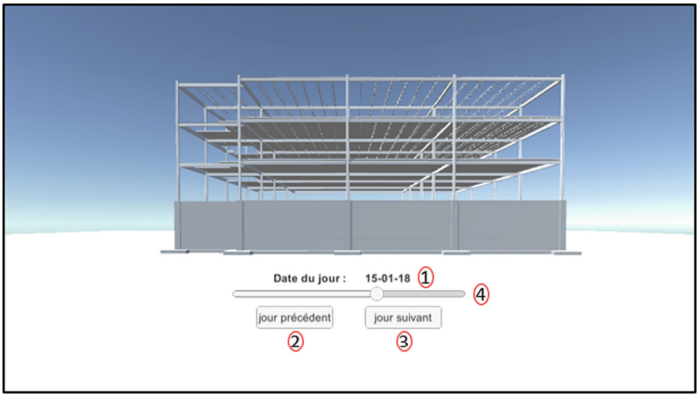

List « ele »
Constraint: this part of the code requires that in a text file "Planning", the start and end dates are in the format [Day YY-MM-DD] and that they are respectively in third and fourth element on the lines of text .
Retrieves the String corresponding to the start and ed dates of each tAsk and puts them in the format [YY;MM;DD]
List of String list : "deb" [[start year 1; start month 1, start day 1], [YY 2;MM 2; DD 2], ... [YY m;MM m; DD m]]
List of String list : "fin" [[end year 1; end month 1, end day 1], [YY 2;MM 2; DD 2], ... [YY m;MM m; DD m]]
Calculates the first day of construction "startdate", the last day of construction "enddate" and the number of days of construction. "nbdays"
DateTime « startdate »
DateTime « enddate »
Int « nbdays » (integer)
Initializes the date displayed on Day 1
Initializes the length of the slider scale

Fig. 3. Program flow from the initial calculations.

Finally, when launching the game, the 3D model is connected to the schedule and hidden as the game starts on day 1 of the site, so no element is supposed to be displayed.

3.3 The User Interface

The goal is to make an interface as simple as possible. Thus, in addition to the main window displaying the 4D model, the interface of the application is composed of four elements (Fig. 4). The first is a text, it informs the user of the date of the day of the site displayed by the application (1). At game launch, this date is initialized on the first day of the first task built. At each change on the time scale, this date is updated. Then the interface has two button type interactors. The first one displays the text “Previous day” (2) and the second displays “Next day” (3). They make it possible to modify the date of the day by making use of the previously developed interaction functions. The last element of the interface is a slider (4). This is a cursor that can be translated on the time scale. Its minimum value is the first day of construction and its maximum value on the last day of construction.



Date du jour : 15-01-18

Fig. 4. User interface of the application (in French)

4 Discussion and Improvements Track

Thanks to the first-person simulation control module, the application is adapted to most virtual reality systems using Unity 3D as a game engine. This is the case of the virtual reality system named Valyz, developed by the company Realyz. The application was tested and evaluated in the Valyz environment, using multiple 4D models.

4.1 Analysis of Use and Criticisms

Several 4D models have been tested in the Valyz system. The functionalities developed in the application are present and active. The movements of the camera follow the directions given by the controller of the system. The 3D glasses can simulate the movement of the head to orient the image.

It is possible to select the construction period to display as expected in the application. However, the interface developed with the buttons, the cursor, and the display of the date is not projected. It is present on the computer screen that is connected to the Valyz system, but it is not present in the virtual environment projected on the wall. It is possible, however, for the user to interact with the buttons from the computer and this affects the displayed model. It would have been preferable for the interface to be displayed in the virtual environment so that the user can point the joystick to the interact button and manage the temporality. The problem is that in the Valyz system support software, only three-dimensional elements are displayed in the virtual environment. One of the solutions provided by computer scientists of the company marketing this system is to change the representation of the interface. In the application, the interface is created from a 2D canvas connected to the camera. The idea would be to replace each element with three-dimensional objects, for example, the buttons would be replaced by cubes also connected to the camera. The user could then interact with the object by pointing his controller. These changes require that the code be modified and that new functions be developed accordingly.

Regarding the modification of the simulation and the use of alternative schedules, the solutions proposed previously tackle the problem that the Unity application is no longer connected to the .exe file used by the software that allows to control the Valyz tool. Therefore, it is necessary to perform a new export of the file from Unity with each change, which is not optimal.

4.2 Improvements Track

A number of comments and suggestions for improvement arise from the evaluation, namely improvements in data imports, the addition of additional features related to BIM 4D, and the constraints of Unity.

Improvements in Data Imports: At the 3D model level, the FBX format is the most suitable for use in Unity. Only we saw that there was a choice to make between rendering the displayed texture and the structure of the object tree. It would be interesting to continue the search for an export system from Autodesk software in FBX format that can satisfy both a good rendering state of the model display and a structure at the same time. The tree that can be used by the developed code.

In the same way, when importing the schedule, the text file is too formally constrained. For example, the code developed assumes that the original Excel file has the following form (Column 1: Task Name; Column 2: Duration of tasks; Column 3: Start date of the tasks; and Column 4: End date of tasks).

This means that this type of file must follow a standard that is not necessarily adopted in the workplace. Only, we can notice that the code is easily modifiable if the workbook at the origin is not exactly adapted to the code. The improvement track would be to make the code more automatic, for example by first performing a function that detects in which column the useful information is located.

Regarding the link between the list of 3D elements and the scheduled task list, it could be improved by adding the use of a standard. Some, such as the Uniformat II standard, may be able to link the data more easily. It would be possible to import also in the planning a column con-holding this kind of code. For the moment, no solution has been found for this standardized code to appear in the name of the elements of the 3D model. For the editable aspect of the simulation without having to leave the game to reload a new schedule, a track of improvements would be to load in the file "Resources" several versions of the planning of construction and connected to the interface as many buttons. Thus, simply by activating a schedule by clicking on it in the interface, the application directly calculates the new simulation and therefore modifies accordingly the display of the elements constructed or not.

Additional Features Related to the 4D BIM and Constraints of Unity: One of the main features that could not be added to the application is the differentiation of the build state of each task. In our case, from the moment a task has a start date prior to today's date in the simulation, it is displayed. Otherwise, it is hidden. The idea would be to be able to have several states possible: built elements - elements in construction - elements not built. One could for example display in a normal way the constructed elements, in green the elements under construction and with a transparency effect the elements that remain to build. This was not possible because Unity 3D considers that each item from the 3D model import does not have a feature to change its appearance.

When using Unity 3D, several problems were encountered. Especially at the level of the text file corresponding to the schedule. Many characters are not allowed by the software when reading the document, such as accents or "-" that may be in the name of the tasks. It may be possible then to create an additional function that takes care of replacing or removing these characters, keeping in mind that they must still correspond to the names of the elements of the 3D model present in the tree.

5 Conclusion and Future Works

Thanks to the evolution of computer tools, it is possible to simulate and to anticipate many problems that arise on construction sites. This is all the more remarkable since the modeling software makes it possible to integrate the construction schedule. It is important for the workers to visualize the 3D model in an equivalent way. The research project reported in this paper aimed to automate the integration of 4D simulations in virtual reality environments, to easily generate 4D simulation alternatives in such environments. The work shows that it is possible to create 4D models in a game engine, directly from a BIM model and an existing schedule. However, several improvements still need to be made in order to achieve the desired situation. Future work will focus on correcting these limitations, before going further in the study of 4D models in virtual reality environments.

References

1. Sacks, R., Eastman, C., Lee, G., Teicholz, P.: BIM Handbook: A Guide to Building Information Modeling for Owners, Designers, Engineers, Contractors, and Facility Managers. Wiley, Hoboken (2018)
2. Boton, C., Forgues, D.: Practices and processes in BIM projects: an exploratory case study. In: Advances in Civil Engineering (2018)
3. Kreider, R., Messner, J., Dubler, C.: Determining the frequency and impact of applying BIM for different purposes on building projects. In: Proceedings of 6th International Conference on Innovation in Architecture, Engineering and Construction (AEC), pp. 1–10 (2010)
4. Boton, C., Kubicki, S., Halin, G.: Designing adapted visualization for collaborative 4D applications. Autom. Constr. **36**, 152–167 (2013)
5. Akinci, B., Fischer, M., Levitt, R.: Formalization and automation of time-space conflict analysis (2000)
6. Hartmann, T., Fischer, M.: Supporting the constructability review with 3D/4D models. Build. Res. Inf. **35**, 70–80 (2007)
7. Clayton, M.: Virtual construction of architecture using 3D CAD and simulation. Autom. Constr. **11**, 227–235 (2002)
8. Kang, H., Lho, B., Kim, J.: Development of web-based interactive 4D block tower model for construction planning and scheduling education. In: Proceedings of the 2004 ASEE Annual Conference and Exposition, Salt Lake City, Utah, pp. 1–10 (2004)

9. Wang, L., Messner, J.I., Leicht, R.: Assessement of 4D modeling for schedule visualization in construction engineering education. In: 24th CIB W78 Conference, Bridging ITC Knowledge to Work, Maribor, Slovenia, pp. 737–742 (2007)
10. Kubicki, S., Boton, C.: 4D-based teaching of high-rise structural principles. In: Proceedings of the CIB W78-W102 2011, Sophia Antipolis (2011)
11. Yabuki, N., Shitani, T.: A management system for cut and fill earthworks based on 4D CAD and EVMS. In: Proceedings of Computing in Civil Engineering Conference, Cancun, Mexico, pp. 1–8 (2005)
12. Shah, R.K., Dawood, N., Castro, S.: Automatic generation of progress profiles for earthwork operations using 4D visualisation model. Electron. J. Inf. Technol. Constr. **13**, 491–506 (2008)
13. Boton, C., Kubicki, S., Halin, G.: The challenge of level of development in 4D/BIM simulation across AEC project lifecyle. A case study. Procedia Eng. **123**, 59–67 (2015)
14. Boton, C.: Supporting constructability analysis meetings with immersive virtual reality-based collaborative BIM 4D simulation. Autom. Constr. **96**, 1–15 (2018)
15. Harvey, D.: Invisible site: a virtual sho. George Coates Performance Works, San Francisco, CA, vol. 346, p. 87 (1992)
16. Sherman, W.R., Craig, A.B.: Understanding Virtual Reality: Interface, Application, and Design. Morgan Kaufmann Publishers, San Francisco (2018). ISBN 978-0128009659
17. Bouchlaghem, N.M., Liyanage, I.G.: Virtual reality applications in the UK's construction industry. http://itc.scix.net/data/works/att/w78-1996-89.content.pdf
18. Berg, L.P., Vance, J.M.: Industry use of virtual reality in product design and manufacturing: a survey. Virtual Reality **21**, 1–17 (2017)
19. Issa, R.R.A.: Virtual reality: a solution to seamless technology integration in the AEC industry. In: Construction Congress VI, pp. 1007–1013. ASCE (2005)
20. Whyte, J.: Innovation and users: virtual reality in the construction sector. Constr. Manag. Econ. **21**, 565–572 (2003)
21. Woksepp, S., Olofsson, T.: Credibility and applicability of virtual reality models in design and construction. Adv. Eng. Inform. **22**, 520–528 (2008)
22. Paes, D., Arantes, E., Irizarry, J.: Immersive environment for improving the understanding of architectural 3D models: comparing user spatial perception between immersive and traditional virtual reality systems. Autom. Constr. **84**, 292–303 (2017)
23. Goulding, J., Nadim, W., Petridis, P., Alshawi, M.: Construction industry offsite production: a virtual reality interactive training environment prototype. Adv. Eng. Inform. **26**, 103–116 (2012)
24. Dickson, P.E., Block, J.E., Echevarria, G.N., Keenan, K.C.: An experience-based comparison of unity and unreal for a stand-alone 3D game development course. In: Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education - ITiCSE 2017, pp. 70–75 (2017)
25. Zarrad, A.: Game engine solutions. In: Simulation and Gaming, pp. 75–87 (2018)
26. Liu, Y., Tanudjaja, G., Jiang, Z., Beck, N.: Workflow of Exporting Revit Models to Unity. State College, Pennsylvania (2016)
27. Broadbent, N.: BIM and gaming engines: how we present 4D models using our design suite and gaming engine (2011)