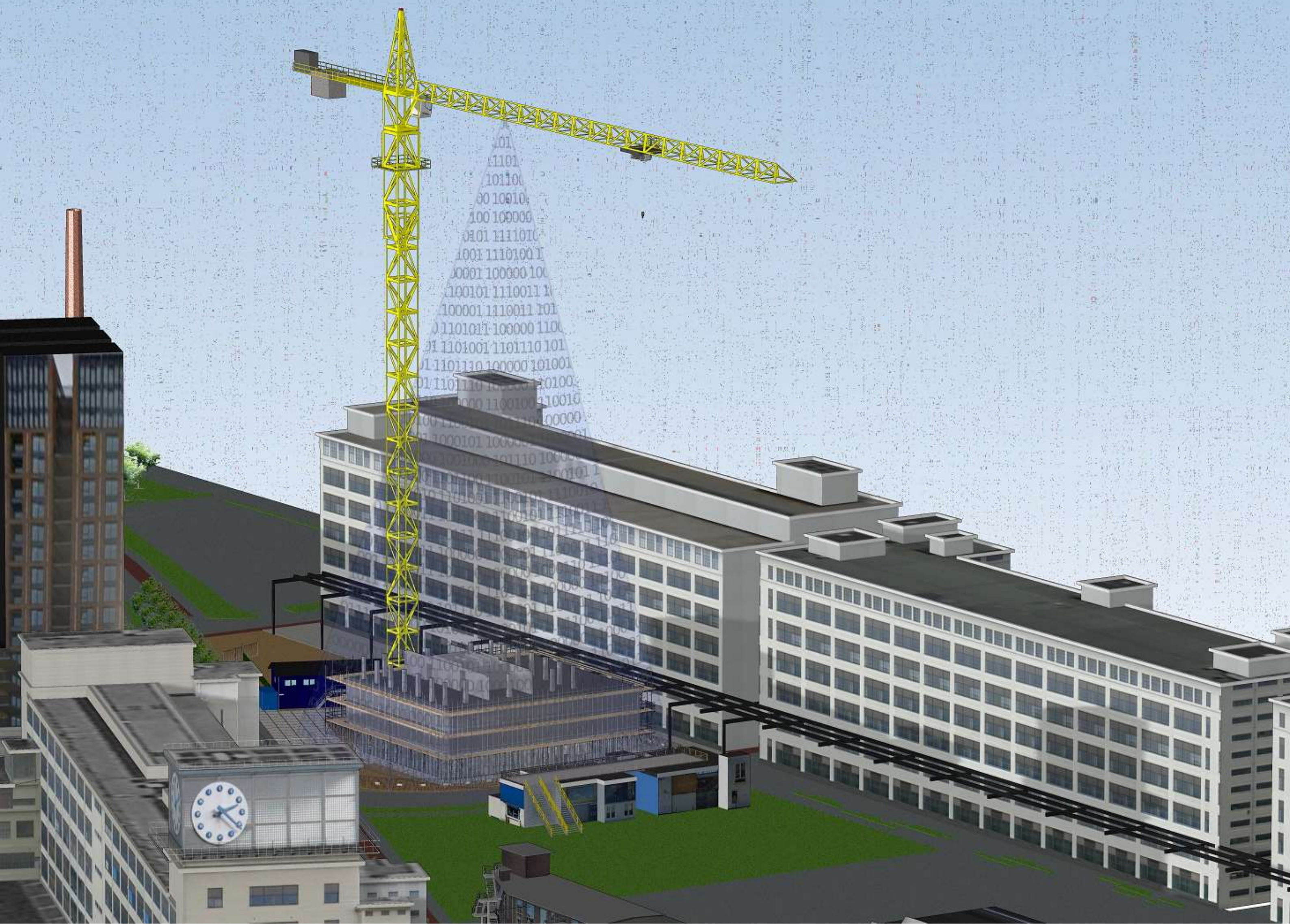
USING POINT CLOUD TO AUTOMATICALLY UPDATE THE BIM 4D-MODEL

2020

MASTER THESIS
Construction Management and Engineering
Eindhoven University of Technology

Selahattin Dülger



Colophon

General

Report	: Using point cloud to automatically update the BIM 4D-model
Date	: 18-03-2020
Date Presentation	: 23-03-2020
Place	: Eindhoven, The Netherlands

Student

Author	: Selahattin Dülger
Student Number	: 1277456
E-mail	: s.dulger@student.tue.nl
	: selo251@gmail.com
University	: Eindhoven University of Technology
Master Track	: Construction Management and Engineering (CME)
Chair	: Information Systems in the Built Environment (ISBE)

Chairman
Prof. Dr. Ir. B. (Bauke) de Vries
University supervisor
b.d.vries@tue.nl

First Supervisor
Ir. J.J. (Jakko) Heinen
University supervisor and BIM-Connected
j.j.heinen@tue.nl

Second Supervisor
ir. L.I.H. (Luuk) Wijnholts
University supervisor and BIM-Connected
l.i.h.wijnholts@tue.nl

External supervision
ir. S. (Stijn) van Schaijk
Stam + De Koning
svschaijk@volkerwessels.com

Construction Management and Engineering
Eindhoven University of Technology
De Wielen
Postbus 513, 5600 MA Eindhoven
Tel. +31 (0)40 247 5051

Preface

My interest in this subject started when I was completing my Bachelors. Since then I wondered how new implementations can change the construction sector, especially within the field of BIM. Combining this with programming languages, like Python, gave me a different perspective about how the construction sector should perform. With the fresh knowledge I gained during my Construction Management & Engineering programme, I was able to bridge the gap between the construction sector and ICT.

When I look back at the time I spent as a student at TU/e, I can say with certainty that I will miss it. This includes, in special, my fellow students and the board members of the study association, 'of CoUrSe!'

In addition, I also want to take a moment to reflect on the process involved in this graduation project. I would like to thank my mentors Jakko and Luuk for their time and hospitality at their office. Their constructive feedback always gave me the motivation to work harder.

Lastly, I want to thank my family and my girlfriend for their emotional support.

I want to wish you all a pleasant time reading this thesis, and I hope you will get inspired of it.

Selahattin Dülger

Eindhoven, March 2020

Table of contents

Abstract

Keywords: Building Information Modelling (BIM), construction planning, BIM 4D, point cloud, automated tool, updating planning, Python.

Abstract: Time is money for all construction projects. Within the domain of Building Information Modelling, new techniques are being developed to control the time aspect, such as BIM 4D. Here the 3D model of BIM is combined with the construction planning, resulting in a visualisation of the planning. Construction projects are characterised as a dynamic process, in which the planning continuously deviates from the actual situation. Keeping track of the planning in BIM 4D is characterized as time-consuming and labour intensive. This research focuses on automating the process, where a Python tool is being developed. As a result of this it is necessary to retrieve construction site information. This is done using a monitoring technique that generates point clouds. The tool has been tested on a case study, and it turns out that it has an accuracy of detecting construction elements of 100%. The data of detected elements are easily being exported to the BIM 4D-model, resulting in updated planning. It is concluded that besides the visualisation possibilities of BIM 4D, also construction progress can be tracked and compared with the planning. In addition, it is concluded that the developed tool is not fully automatic. Despite this, the developed tool can serve superintendents, site managers, project planners, and project managers to compare project planning based on point clouds and identify whether the project is on track or behind planning. The whole process may help against large time-consumption and labour intensiveness, and therefore encourage practitioners to use BIM 4D more and more in the future. How the process can be made fully automatic is an interesting topic for further research.

Summary (English)

In the construction sector, many stakeholders with different expertise are required to deliver a construction project successfully. Close collaboration between stakeholders is very important in this regard. To achieve this, Building Information Modelling (BIM) is used. With BIM, an accurate virtual 3D model of a building is constructed. The 3D model facilitates information sharing that enables a collaborative decision-making platform for stakeholders. Besides the necessity of close collaboration, some responsibilities should be complied. One of them is that the construction project must be completed within a specified date. When this is not respected, fines can be expected.

To manage time, construction planning is managed by the construction planner. Within the planning process, many variables are taken into account to ensure the planning is feasible. Sometimes, however, the interrelationships between the activities become so complex that they are no longer amenable to stakeholders. This is because the traditional construction planning lacks the desired output when it comes to schedule visualisation and thus the comprehensibility of it. That is why BIM 4D has been developed. This is a combination of the 3D model and construction planning. After all activities from the construction planning are linked to the 3D objects of the 3D model, a visualisation of the planning can be created. The application of BIM 4D has, due to its significant advantages, been widely used in recent years in practice. However, BIM 4D-models needs to be updated regularly, which is time-consuming and labour intensive. This discourages their use among industry practitioners.

The essence of BIM 4D is not only visualisation but also: communication; optimization; site logistics; trade coordination; safety planning; and compare schedules and track construction progress. This research focuses on the latter. The literature indicates that there is a need for solutions that automatically incorporate the detected progress data into 4D BIMs and update the schedule and tasks associated with 3D model objects (Hamledari et al., 2017). Through a process called Scan-vs-BIM, it is possible to capture the actual site conditions and state of work for building elements. For this, imaging technologies are used, such as 3D laser scanning; modern digital photogrammetry; and drones. These technologies can generate point clouds. All in all, this has led to the next research goal, and that is: To develop a tool that measures the deviations between the BIM 4D-planning and the actual construction site conditions (point cloud model) and automatically update the BIM 4D-planning and notify it to the designated person(s).

In order to achieve the goal, the research is divided into three phases: Plan, Capture and Analyse phase. Each phase has desired inputs and outputs. For the plan phase, the desired input is the 3D model and construction planning. The desired output is a BIM 4D-model. For the capture phase, the desired input is the scan data (like pictures), and the desired output is a point cloud data. The last phase is the analysis phase. The desired input is the desired outputs of the plan and capture phases. The desired output is data through which the BIM 4D-model can be updated. To obtain the desired output, a Python tool is used. The execution of the analyse phase is divided into three steps, which are (1) combining and cleaning data, (2) intersecting data, and (3) exporting and importing data.

The developed tool is tested in a case study, which is the Trudo Tower in Eindhoven. In the plan phase, a BIM 4D-model is created of the 4th storey of the building. In the capture phase, a point cloud data is generated using the photogrammetry technique via crane cameras. The analyse phase combined the 3D data with point cloud data. In here, the points of the point cloud data are first cleaned. This reduced the number of points by 90%. Thereafter, the points are intersected with 3D objects. When 3D objects contain points that are above the indicated threshold, the tool associates the 3D objects as being built. The Python tool had an accuracy of 100%. This was shown by intersecting the 3D data with different point cloud data. Lastly, the intersected 3D objects can be exported to a data file that can be imported

in the BIM 4D-model. Here, the status of the corresponding activity to the 3D object is automatically updated. The result of this is that an overview can be generated with the as-planned and an as-built situation.

The entire process of updating the BIM 4D-model was found to be not fully automatic. This is because point cloud data had to be manually scaled to the right dimension, the base point should be determined, and the point cloud data had to be converted to a readable data source. These were the only handlings that needed to be done before using the developed Python tool. However, this research made it possible to automate the process of combining 3D and point cloud data to export intersected 3D objects to the BIM 4D-model.

Finally, this research resulted in that one of the essences of BIM 4D can be reached, which is: to compare schedules and track construction progress. Although it is concluded that the developed tool is not fully automatic, it can serve superintendents, site managers, project planners, and project managers to compare project planning based on point clouds and identify whether the project is on track or behind planning. The whole process may help against large time-consumption and labour intensiveness, and therefore encourage practitioners to use BIM 4D more and more in the future. How the process can be made fully automatic is an interesting topic for further research.

Summary (Dutch)

In de bouwsector zijn veel stakeholders met verschillende expertise nodig om een bouwproject tot een succesvol einde te brengen. Nauwe samenwerking tussen de belanghebbenden is daarbij van groot belang. Om dit te bereiken wordt gebruik gemaakt van Building Information Modelling (BIM). Met BIM wordt een nauwkeurig virtueel 3D-model van een gebouw opgesteld. Het 3D model faciliteert het delen van informatie die een gezamenlijk beslissingsplatform voor belanghebbenden mogelijk maakt. Naast de noodzaak van nauwe samenwerking zijn er ook verantwoordelijkheden die moeten worden nagekomen. Eén daarvan is dat het bouwproject binnen een bepaalde datum moet zijn afgerond. Wanneer dit niet wordt nagekomen, kunnen er boetes worden verwacht.

Om tijd te managen wordt er een bouwplanning gemaakt door de bouwplanner. Binnen de planning wordt rekening gehouden met vele variabelen om de planning haalbaar te maken. Soms worden de onderlinge samenhang tussen de activiteiten dermate complex dat ze niet meer vatbaar zijn voor de stakeholders. Dit komt omdat de traditionele bouwplanning de gewenste output mist als het gaat om de visualisatie van de planning en dus de begrijpelijkheid ervan. Om die reden is BIM 4D ontwikkeld. Dit is een combinatie van het 3D model en de bouwplanning. Nadat alle activiteiten uit de bouwplanning zijn gekoppeld aan de 3D-objecten van het 3D-model, kan een visualisatie van de planning worden gemaakt. De toepassing van BIM 4D is, vanwege de grote voordelen, de afgelopen jaren in de praktijk veelvuldig toegepast. Echter, BIM 4D-modellen moeten regelmatig worden bijgewerkt, wat tijdrovend en arbeidsintensief is. Dit ontmoedigt het gebruik ervan door de industrie.

De essentie van BIM 4D is niet alleen visualisatie, maar ook: communicatie; optimalisatie; logistiek op locatie; coördinatie van de werkzaamheden; veiligheidsplanning; en het vergelijken van planningen en het volgen van de voortgang van de bouw. Dit onderzoek richt zich op dit laatste. De literatuur geeft aan dat er behoefte is aan oplossingen die de waargenomen voortgangsgegevens automatisch verwerken in BIM 4D-modellen en de planning en taken gerelateerd aan 3D-modelobjecten actualiseren (Hamledari et al., 2017). Door middel van een proces dat Scan-vs-BIM heet, is het mogelijk om de actuele situatie op de bouwplaats en de staat van de werkzaamheden voor bouwelementen vast te leggen. Hiervoor wordt gebruik gemaakt van beeldverwerkingstechnologieën, zoals: 3D-laserscanning; moderne digitale fotogrammetrie; en drones. Deze technologieën zijn in staat om puntenwolken te genereren. Al met al heeft dit geleid tot het volgende onderzoeksdoel en dat is: Het ontwikkelen van een tool die de afwijkingen meet tussen de BIM 4D-planning en de actuele bouwplaats omstandigheden (puntenwolk model) en de BIM 4D-planning automatisch bijwerkt en notificeert aan de aangewezen persoon of personen.

Om het doel te bereiken wordt het onderzoek in drie fasen verdeeld: Plan, Capture en Analyse fase. Elke fase heeft een gewenste in- en output. Voor de Plan fase is de gewenste input het 3D-model en de bouwplanning. De gewenste output is een BIM 4D-model. Voor de Capture fase is de gewenste invoer de scangegevens (zoals foto's) en de gewenste output een puntenwolkgegevens. De laatste fase is de Analyse fase. De gewenste input is de gewenste output van de Plan- en Capture fase. De gewenste output is data waarmee het BIM 4D-model kan worden geüpdatet. Om de gewenste output te verkrijgen wordt een Python-tool gebruikt. De uitvoering van de Analyse fase is verdeeld in drie stappen, namelijk (1) het combineren en opschonen van data, (2) het intersecten van data, en (3) het exporteren en importeren van data.

De ontwikkelde tool wordt getest in een case study, namelijk de Trudo-toren in Eindhoven. In de Plan fase wordt een BIM 4D-model gemaakt van de 4e verdieping van het gebouw. In de Capture fase wordt met behulp van de fotogrammetrietechniek via kraancamera's een puntenwolk gegenereerd. In de Analyse fase worden de 3D-data gecombineerd met de gegevens van de puntenwolk. Hierbij worden

de punten van de puntenwolkgegevens eerst opgeschoond. Dit heeft het aantal punten met 90% gereduceerd. Daarna worden de punten intersect met 3D-objecten. Wanneer 3D-objecten punten bevatten die boven de aangegeven drempelwaarde liggen, associeert de tool de 3D-objecten als zijnde gebouwd. De Python-tool had hierbij een nauwkeurigheid van 100%. Dit werd aangetoond door de 3D-data te intersecten met verschillende puntenwolkgegevens. Tenslotte kunnen de waargenomen 3D objecten geëxporteerd worden naar een databestand dat geïmporteerd kan worden in het BIM 4D-model. Hier wordt de status van de corresponderende activiteit van het 3D-object automatisch geüpdatet. Hierdoor kan een overzicht worden gegenereerd met de geplande en een as-built situatie.

Het gehele proces van het updaten van het BIM 4D-model bleek niet volledig automatisch te verlopen. Dit komt doordat: puntenwolkgegevens moesten handmatig worden geschaald naar de juiste dimensie, de nulpunt moest worden bepaald en de puntenwolkgegevens moesten worden geconverteerd naar een leesbare databron. Al deze zaken waren de enige handelingen die gedaan moesten worden voordat de ontwikkelde Python-tool gebruikt kon worden. Dit onderzoek maakte het nog steeds mogelijk om de automatisering van het proces van het combineren van 3D- en puntenwolkdata en de status van de activiteiten naar het BIM 4D-model te exporteren.

Uiteindelijk heeft dit onderzoek ertoe geleid dat een van de essenties van BIM 4D kan worden bereikt, namelijk: het vergelijken van planningen en het bijhouden van de voortgang van de bouw. Hoewel geconcludeerd wordt dat de ontwikkelde tool niet volledig automatisch is, kan deze opzichters, bouwplaatsmanagers, werkvoorbereiders en projectmanagers ondersteunen bij het vergelijken van de projectplanning op basis van puntenwolken en bij het vaststellen of het project op schema ligt of achterloopt volgens de planning. Het hele proces zal wellicht helpen tegen grote tijdrovendheid en arbeidsintensiviteit en zal daarom de gebruikers van BIM 4D in de toekomst meer en meer aanzetten tot het gebruik ervan. Hoe het proces volledig automatisch kan worden vormgegeven is een interessant onderwerp voor verder onderzoek.

1. Introduction

In the construction sector, different parties are needed to realise a construction project, each with their own specialisation in a particular discipline. The disciplines can be roughly divided into Architecture, Engineering and Construction (AEC). To ensure that the project runs as smoothly as possible, a good working relationship is needed between those disciplines. This is even more important when construction projects become larger and more complicated (Oh et al., 2015).

In recent years, there have been some developments that have influenced the common way of working and collaboration, including the organisation of construction projects and the roles of different parties, one of which is Building Information Modelling (BIM) (Liu et al., 2017). An accurate virtual model of a building, known as a Building Information Model, is digitally created using BIM technology. Once finished, the building information model contains accurate geometry and relevant data necessary to support the design, procurement, fabrication, and construction activities needed to complete the building (Eastman et al., 2008). As the model is being created, team members are continually refining and adjusting their portions according to project requirements and design changes to ensure that the model is as accurate as possible before the project physically breaks ground (Carmona & Irwin, 2007).

Besides the necessity of close collaboration between disciplines, each of them also has specific responsibilities in terms of complying to project requirements. One of the requirements is that the building (or a part of it) must be completed within a specified time frame. In order to ensure this, a construction schedule is made. In this construction schedule, the activities of the construction work (tasks) are indicated over time. The construction planner, who makes the construction schedule, uses the following sources of information to make the construction schedule:

- The building drawings derived from BIM-model;
- Technical description;
- Construction method (how the elements can be built);
- The teams or equipment to be used;
- Contract terms and conditions (start and end date);
- Environmental factors (location).

After the aspects mentioned above have been processed by the planner, it will be shown to the construction organisation in the form of a Gantt chart. Although Gantt charts have been used for decades as a tool for project planning and scheduling, they lack the desired output when it comes to schedule visualisation (Kumar, 2014) and thus the comprehensibility of it. In this way, problems may arise because Gantt charts cannot show the idea behind the method of construction that the planner has devised. In some cases, the interrelationship of construction activities from the Gantt chart is not always clear to the executing party, resulting in delays or construction errors. This is due to the number of variables in the construction schedule, which are sometimes so large that they can no longer be understood.

1.1. BIM 4D

To make the construction schedule easier to interpret, the use of BIM is further developed to meet different necessities. One of them is called BIM 4D. This is a method in which the BIM-model is linked to a construction schedule. More specifically, the construction activities are linked to different 3D-objects from BIM. In this way, it is possible to create a simulation where construction processes are shown at any point in time. This will lead to that more insight is created in how the building is constructed each day and it reveals potential problems and opportunities for possible improvements

(Eastman, 2011). The use of the BIM 4D-planning can, therefore, offer several advantages for construction projects in the following areas:

- **Visualisation**: The construction schedule is made transparent through a simulation or step-by-step plan. The executing party can follow the planner's thinking.
- **Communication**: The communicative barrier is reduced because the executing party sees what it is about. This makes it easier to understand what is meant by certain building methods.
- **Optimisation**: The visualisation of the BIM 4D planning provides more insight, allowing errors in the construction schedule to be noticed and corrected earlier.
- **Site logistics**: Planners can manage laydown areas, access to and within sites, locations of large equipment, trailers, and so forth (Eastman, 2011).
- **Trade coordination**: Planners can coordinate the expected time and space flow of materials on the site as well as the coordination of work in small spaces (Eastman, 2011).
- **Safety planning**: The utilisation of 4D-BIM technology can result in improved occupational safety by connecting the safety issues more closely to the construction planning, providing more illustrative site layout and safety plans, providing methods for managing and visualising up-to-date plans and site status information, as well as by supporting safety communication in various situations, such as informing site staff about coming safety arrangements or warning about risks (Sulankivi, 2010).
- **Compare schedules and track construction progress**: Project managers can compare different schedules effortlessly, and they can quickly identify whether the project is on track or behind schedule (Eastman, 2011).

1.2. The research problem and research gap

The use of BIM 4D has been widely used in practice in recent years, due to its significant advantages. Unfortunately, BIM 4D also has negative consequences. Hamledari et al. (2017) say that one of the biggest issues of a BIM 4D-model is that it needs to be updated regularly to reflect the actual (as-built) conditions during construction. This is time-consuming and labour intensive, which discourages their use.

Hamledari et al. (2017) cite that there is a need for solutions that automatically incorporate the detected progress data into 4D BIMs and update the schedule and tasks associated with 3D model objects. Tracking and visualisation techniques are one of the solutions and are continuing to develop. Through a process called Scan-vs-BIM (Bueno et al., 2018) it is possible to compare an image or scan of the as-built (or as-is) environment against the 3D BIM model of a building. It has been shown that it has great potential for supporting activities such as construction progress control, quality control and eventually, life-cycle monitoring (Bueno et al., 2018). To capture the actual site conditions and state of work for building elements, researchers have studied the use of various field data capture solutions, such as imaging and geospatial technologies (Turkan et al., 2012; Shahi et al., 2013; Kopsida et al., 2015; Omar & Nehdi, 2016). These studies compare the actual site conditions with the as-planned data, to verify schedule deviations, detect the state of progress, and visualise it (Hamledari et al., 2017).

While progress tracking and visualisation techniques continue to develop, modelling and automatically updating the BIM 4D from the acquired information by progress tracking techniques has been identified as one of the grand challenges in the construction industry and is left open for further research (Heesom & Mahdjoubi, 2004; Issa et al., 2005; Hartmann et al., 2008; Lopez et al., 2015; Van Schaijk, 2016; Hamledari et al., 2017). This is because prior research is one way or another cumbersome and a lot of manual input is needed to detect deviations. This will lead to that proposed methods are used less, and if they are being used, it will be only during critical moments. Automating the process of detecting deviations may, therefore, be able to encourage more use of BIM 4D.

Additionally, it is observed that this issue is in line with the practice, where no construction company is currently working on tracking the actual site conditions and automatically updating this in the BIM 4D-model. To reach BIM 4D's full potential (Eastman, 2011), it is necessary to research how the progress of the construction can be tracked and registered in order to automatically update the BIM 4D-planning.

As already indicated, there are many imaging and geospatial technologies available that can track the actual construction progress. It has been found that using these technologies on the construction site is faster, cheaper, accurate and provides more reliable data, compared to the traditional labour ground-based surveying method (Turkan et al., 2012). By using point clouds, derived from imaging and geospatial technologies, it is possible to survey and map large areas of the site. A simple workflow and easily repeatable data recording allow maps and models to be updated as often as needed. For example, earthworks or building parts can be quantified, and the progress of a construction activity (task) can be monitored using geographical reference points and 3D data. In this way, a window for opportunity arises by applying these technologies and using its as-built (point cloud) information to match the as-planned model from the BIM 4D (Bosche et al., 2009) and may, therefore, offer advantages in the workflow of automatically updating the BIM 4D-model. The purpose of this research is not primarily focused on why a particular technology should be chosen, but rather how the generated point cloud can be used as an input for the research. After all, the bottom line is that all the mentioned technologies can generate point clouds.

Due to the research problem, the following theoretical research goal is formulated:

To automatically detect the construction progress data from point clouds and incorporate it into BIM 4D in order to optimise the construction planning so that one of the potentials of BIM 4D can be reached.

There is also a practical research goal, which is:

To develop a tool that measures the deviations between the BIM 4D-planning and the construction progress from point clouds and automatically notify them to the designated person(s).

Based on the theoretical and practical research goals, the following hypothesis is formulated:

If a tool is developed that measures the deviations between the BIM 4D-planning and the construction progress from point clouds and the deviations are automatically notified to the designated person(s), the construction planning can be optimized, and one of the potentials of BIM 4D can in that way be reached.

In order to reach the main goal, a primary research question and multiple sub-questions are developed.

The main research question is:

How can a tool be developed in order to measure the deviations between the BIM 4D-planning and the construction progress from point clouds and automatically notify them to the designated person(s)?

The following sub-questions are divided into two categories, which are literature study and practice study:

Literature study

1. *What makes it so complex that construction progress does not go as initially foreseen?*
2. *What is needed in order to complete a construction project successfully?*
3. *How are construction activities traditionally managed in terms of project management?*
4. *What is the main purpose of BIM and in which way can this improve construction projects?*
5. *What is the State of the Art of BIM 4D, and how can this improve construction projects even more?*
6. *What are monitoring systems, and how can this support BIM 4D?*
7. *How can point clouds be an added value for BIM 4D?*

Practice study

1. *Which technological elements are needed that can measure the deviations between the BIM 4D-planning and point cloud?*
2. *How can point cloud data be linked to BIM-4D in order to measure deviations?*
    a. *How is it possible to transform a BIM 4D-model and point cloud model to a readable and structured data source in order to overlay both data and measure deviations?*
3. *How can the deviations between the BIM 4D-planning and the construction progress be notified to designated person(s)?*

1.3. Research approach

As seen in the research questions, they are split into two categories, which are literature study and practice study. This setup will also be used in this research. The following page, Figure 1, shows the research approach.

1.4. Research objectives and limitations

The objective of this research is to capture actual site conditions through imaging and geospatial technology. This research will also identify objects by matching them with the as-planned model, automatically update the BIM 4D-planning using a tool that is going to be developed, and finally notify deviations to the designated person(s). To prove if the proposed tool works, it is tested in a real case.

A lot is happening on the construction site, and therefore the objective of this research is somewhat limited. First, due to many different construction phases, only the execution phase is considered and only focuses on the main construction elements of the building (e.g. columns, floors and façade). Second, the locations and the number of on-site objects constantly change during construction (occlusions) and should be filtered out.

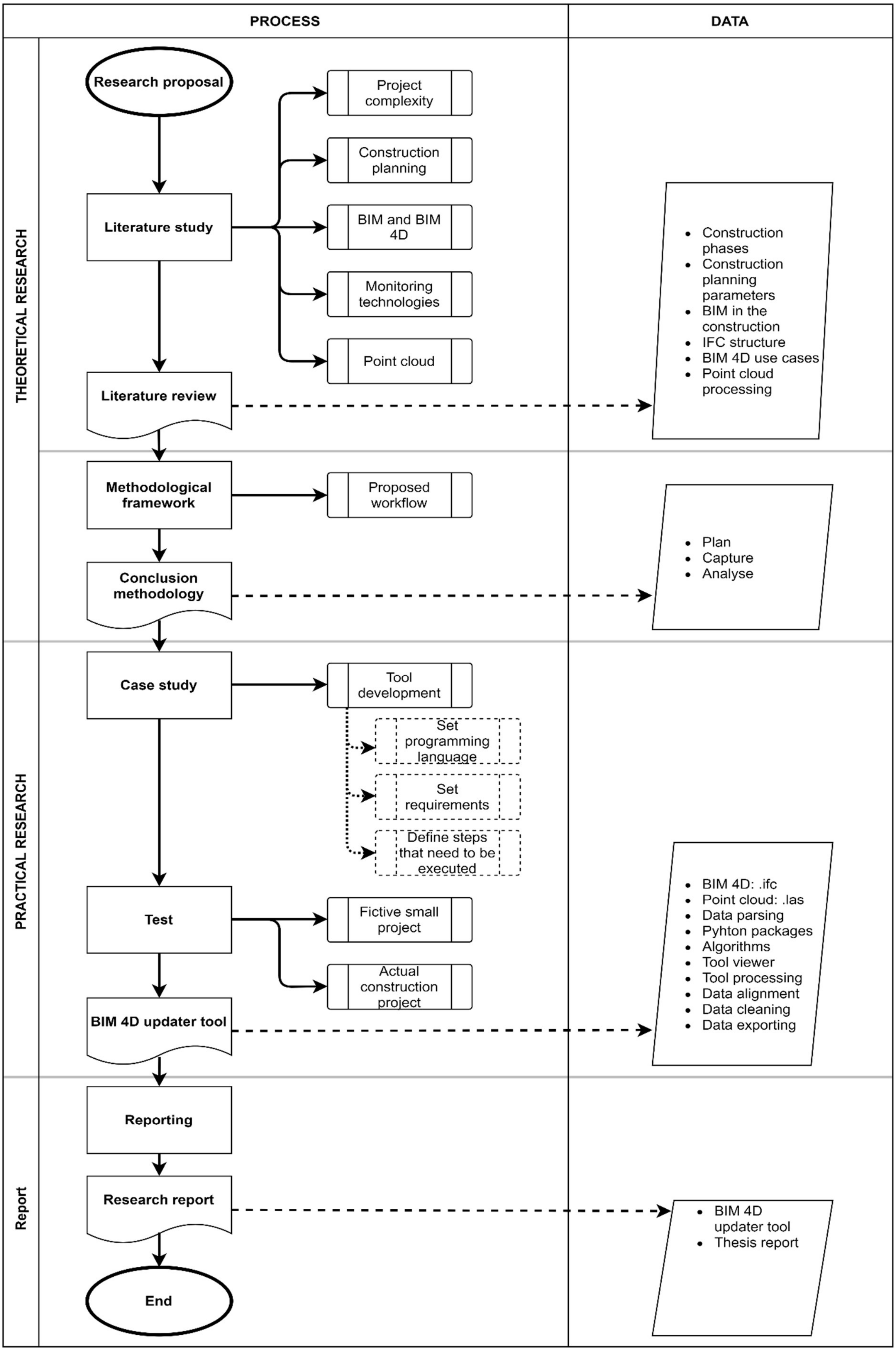


Figure 1: The research approach

1.5. Expected results

The expected results of this research are divided into two parts, as the sub-questions are structured in that way. These are the theoretical part and the practical part. For the theoretical part, a state-of-the-art of progress tracking techniques is explored. In addition, a qualitative analysis about optimization points of construction planning and BIM 4D is expected. The theoretical part will help to support the practical part, leading to an enhanced research methodology based on Bosche et al. (2009). Furthermore, for the theoretical part, it is expected to develop a tool, enabling automatization of tracking the construction site and update the BIM 4D-model. This will possibly lead to process improvement, streamlining and promoting of the usage of BIM 4D in the construction industry.

1.6. Reading guide

The following chapters outline the execution of the research approach. Chapter 2 covers the literature review. This chapter contains five parts, which are: (1) the construction industry, characteristics of the construction industry, and project complexity; (2) construction planning; (3) BIM and BIM 4D; (4) monitoring techniques; and (5) point cloud. Chapter 3 contains the methodological framework, which is divided into three parts: plan, capture, and analyse phase. In Chapter 4 and 5, the tool is being tested on a case study. The conclusions are drawn in Chapter 6. Chapter 7 includes a critical discussion on the research questions and recommendations for further research are drawn.

1.7. Related work

Many research, ranging from graduation theses to articles, are conducted in terms of BIM 4D and the use of imaging and geospatial technologies. The graduation report of Van Schaijk (2016) looked into realizing a shorter construction time for the execution phase of a construction project by determining the bottlenecks and planning deviations with process mining techniques and Building Information Models. Van Schaijk (2016) captured the site with a drone and overlaid the scan with the BIM 4D model. He mentions that it took quite some time to generate point clouds of the images from the drone and, as a result, more than five moments of analysis were not feasible within his study. He recommends that the process of generating point clouds and comparing them with IFC models and generating lists of missing elements should be fully automated. Another graduation thesis is from Pörteners (2018). He researched how a point cloud scan can be segmented and classified semi-automatically through machine learning, within the context of the AEC industry, while retaining user interaction. Although Pörteners (2018) focuses on machine learning, which is not the case in this thesis, it provides a broad overview of the importance of using point cloud data in construction management. He recommends to expand research related to automated processes for point cloud datasets in the AEC industry, which could greatly benefit the sector, such as Scan-to-BIM, and comparing the as-built and as-designed situation of a building for progress monitoring and checking for structural deficiencies.

As for the related works from articles, many researchers (Abeid & Arditi, 2003; Golparvar-Fard et al., 2009; Wu et al., 2010; Golparvar-Fard et al., 2012) looked into an intuitive way to assess the progress by geometrically compare the as-built condition via a variety of imaging and geospatial technologies with the planned condition. However, their method requires manual input, like selecting common points between the 3D scans in order to calculate the volume of investigated objects. This means that calculating work progress for an entire construction project with the prior methods would require a significant amount of manual data processing.

In all, the mentioned research will function as an inspiration resource and basis for this thesis.

2. Literature review

In order the answer the sub-questions related to the literature part, a literature study is conducted. This literature study consists of five parts. It first starts with a general description of the construction industry and why it is essential for the country's gross domestic product. Also, the characteristics of the construction industry are researched, including project complexity. The second part is about construction planning and why this is an essential factor to deliver the project successfully. The third part is the use of BIM and why developments like BIM 4D are crucial for improving the construction planning. The fourth part is how construction planning can be improved even more by utilizing monitoring techniques. The final part is about how monitoring techniques generate point clouds and how these can support the research goal. Figure 2 shows how this chapter is structured.

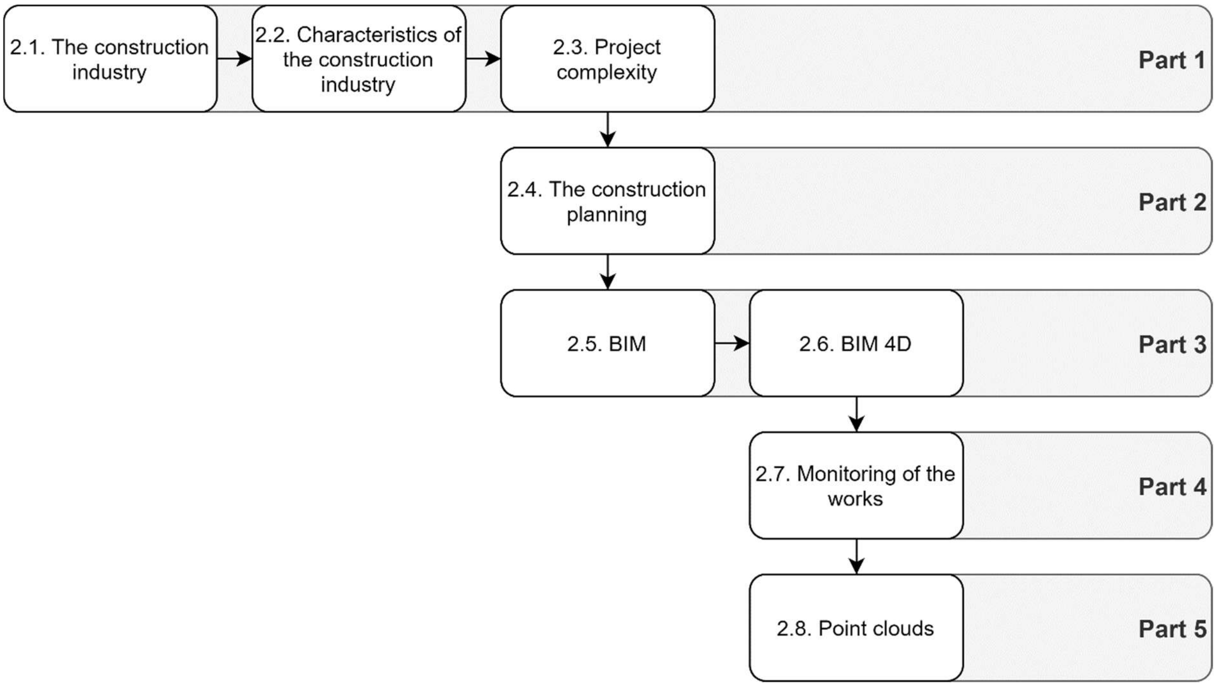


Figure 2: The five parts of the literature review

2.1. The construction industry

The construction industry is one of the main contributors to socio-economic development, owing to its significant contribution to gross domestic product (GDP) (Lopes et al., 2002; Aibinu & Jagboro, 2002; Wigren & Wilhelmsson, 2007), integral role as an economic regulator (Akanbi et al., 2019), employment-generating potential, and strong inter-sectoral linkages (Hua, 2009). Several authors have studied the role of construction in economic development and are based upon the vital linkages between the construction industry and the national economy. Ofori (1990) emphasized that beside the critical role that the construction industry plays in the economic development, the success of development also leads to an increase in disposable income, generating a never-ending cycle of more construction activities. He refers to the construction industry as the "engine of economic growth" (Ofori, 1990 in Hua, 2009).

Some authors focused on the role of construction among different stages of economic development of countries. Strassman (1970) noted that in countries with higher average income level, which signifies higher economic development, there is a more significant contribution of construction to gross national product (GNP) and total employment. Turin (1973) observed the differences between developing countries and industrialised countries concerning contribution to GDP in terms of value-added in construction, contribution to GDP in terms of capital formation in construction, the amount of labour force employed in construction, the amount of input purchases from other sectors of the economy for construction, and the relative share of different types of construction such as dwellings and civil engineering work (Hua, 2009).

Besides the never-ending cycle of construction activities (Ofori, 1990) because of the high economic development (Strassman 1970; Turin 1973), Stone (1983) cited that population size, size and condition of the built stock, technological change, changing lifestyles, and long-and short-term economic conditions are essential factors that can influence demand for development and construction. As the world population grows significantly day by day (WEF, 2016) is the demand for more construction activities, especially for residential buildings, amenable.

2.2. Characteristics of the construction industry

Projects in the construction sector are generally characterized by time delays and cost overruns (Hossen et al., 2015; Mukuka et al., 2015; Abbas et al., 2016; Sohi et al., 2016; Akanbi et al., 2019; Heigermoser et al., 2019). When projects overrun, there is a delay. Aibinu & Jagboro (2002) define delay as a situation when the contractor and the project owner jointly contribute to the non-completion of the project within the original, or the stipulated or agreed contract period. Hossen et al. (2015) define delay as the time overrun either beyond the completion date specified in a contract or beyond the date that the parties agree upon for delivery of a project.

In some cases, a delay can result in dispute, arbitration, loss of revenue, total abandonment, increased costs and protracted litigation by the parties (Aibinu & Jagboro, 2002; Hossen et al., 2015). The conventional approach to managing the extra cost because of time overruns is to include a percentage of the project cost as a contingency in the pre-contract budget (Aibinu & Jagboro, 2002). According to Akinsola (1996) and Sambasivan & Soon (2007), the amount of percentage is hard to predict and are based on judgments. This is because construction projects are unique and have a distinctive set of objectives (Hossen et al., 2015). Sambasivan & Soon (2007) identified the causes of delays through questionnaires, which were sent to clients, consultants, and contractors. After the data collection, their analysis showed that the most important cause of delays was the contractor's improper planning. Another reason for poor performance and delays was assigned to project complexity (Sohi et al., 2016).

The following two paragraphs specify more into project complexity and how this relates to project planning.

2.3. Project complexity

Studies point out that the processes during construction projects are complex. With that, the question arises in terms of what the term complexity exactly is. Mitchell (2009) states that this question is one of the most challenging questions of all. There are many different uses of this term, which are:

- Complexity helps to deal with interdependence, multidimensionality and paradox (Morin, 1986).
- Complexity is a problem construct rather than a solution provider (Morin, 1990).
- Complexity is to account for the articulations between the disciplinary fields (Morin, 1990).
- Complexity is what escapes us, what we have difficulty with to understand and to control (Genelot, 1998).
- Complexity is defined as uncertainty (Lissack, 1999).
- Complexity is defined by its sources, its principles and its objective (Browaeys & Baets, 2003).

Baccarini (1996) states that the construction process is one of the most complex processes in any industry, hence the difficulty of managing major construction processes. Saqib et al. (2008) and Zhang et al. (2013) say that projects are complicated due to the dynamic nature of the construction industry, where uncertainties in technology and budgets are increasing.

Understanding how complex construction projects can be managed is of significant importance. Baccarini (1996) mentions that project-based management is often accompanied by the management of complexity. The importance of complexity to the project management process is widely acknowledged, for example (Baccarini, 1996) states:

- Project complexity helps determine planning, coordination and control requirements.
- The complexity of major projects hinders a clear identification of its objectives and goals.
- Complexity is an essential criterion in the selection of a suitable form of project organisation.
- The complexity of the project affects the selection of project inputs, e.g. the expertise and experience requirements of the managerial staff.
- Complexity is often used as a criterion in the selection of a suitable project procurement plan.
- Complexity influences the project objectives of time, cost and quality. Broadly speaking, the higher the complexity of the project, the greater the time and cost.

Zhang & Hu (2011) identified four characteristics that make the construction process very complex: (1) construction products are fixed while the construction process is in motion, which is the central conflict of spatial planning with time sequences; (2) tasks are often co-executed by workers from different professions, using different types of construction machinery; (3) construction is a long-running, extensive procedure with a significant number of activities; and (4) the structural characteristics during the construction period are very different from those during the maintenance period. Therefore, the most dangerous situation may occur during the construction period.

Complex projects require an outstanding level of management. Thereby, the utilisation of traditional systems developed for traditional projects has been found to be inappropriate for complex projects. Therefore, there is a need for technology. Broadly speaking, technology can be defined as the transformation processes that convert input into output. With this, the transformation process involves the use of material resources, techniques, knowledge and skills (Baccarini, 1996). Akinsola (1996) says that new technology or technical approaches should be used to achieve the desired result. A combination of lean construction and agile project management is assumed to be a possible solution to deal with the complexity of the project (Sohi et al., 2016).

2.3.1. The foundation of lean management and agile project management

Many researchers (Koskela et al., 2002 in Heigermoser et al., 2019; Akanbi et al., 2019) conducted a study to improve construction processes, which in the most part focused on construction productivity and delays. They showed that most of the problems could be solved through an all-embracing project and production management system, which is inspired by the car industry. They later converted to a suitable form for use in the construction industry (Akanbi et al., 2019). By applying the Toyota Production System (TPS), lean management in the construction industry was developed. This principle is used to optimize processes in all kind of fields to achieve more with less (Sohi et al., 2016), ranging from concept development to concept design, production, and even operation and management (Heigermoser et al., 2019). Also, this principle aims to eliminate waste during process activities to reduce process cycles, improve quality, and increase efficiency (Ahuja et al., 2017; Akanbi et al., 2019). In the context of lean management, waste includes all forms of overproduction, over-processing, delay, excess inventory and motions, failure and defects (Small et al., 2017; Khodeir & Othman, 2018; Akanbi et al., 2019). It was a long time before the construction industry adopted Lean Production methods due to its unique characteristics. This is because unlike other industries like the car industry, the construction industry is producing stationary objects, which, to a certain extent, are too large to be moved (Heigermoser et al., 2019).

A different development in project management was the implementation of Agile project management, which aims to increase the relevance, quality, flexibility and business value of software solutions (Owen & Koskela, 2006). This is designed specifically to address the problem that has historically plagued software development and service operations in the IT industry. This includes budget overruns, missed deadlines, low quality output and dissatisfied users (Sohi et al., 2016). Sohi et al. (2016) mention in Cooke (2012) and Johansson (2012) that all agile methodologies have the same primary objectives. These objectives include replacing upfront planning with incremental planning that adapts to the most up-to-date information available, building quality upfront and addressing technical risks as early as possible. Furthermore, it also incorporates minimizing the impact of changing requirements, delivering consistent and continuous business value to the organization, entrusting and engaging staff, encouraging continuous communication between the business areas and project team members, and increasing customer engagement (Sohi et al., 2016).

Sohi et al. (2016) claim that planning features of lean and agile have a significant correlation with three groups of complexity elements, namely technical complexity, uncertainty and organizational complexity. They conclude that proper planning can reduce uncertainty as well as technical and organizational complexity. Thus, proper planning not only reduces the complexity, but also manages the complexity.

2.4. The construction planning

Construction planning is the crucial driver to ensure the success of a project (Ciribini et al., 2016). Meredith & Mantel (2008) state that the primary purpose of construction planning is to define a set of detailed directions. These directions explain to the project team what must be done, when it must be done, and what resources will be needed to produce the deliverables of the project successfully. They add that construction planning must be developed in such a way that the project outcome meets the objectives of the client, where time and cost are the most important factors to meet their satisfaction (Meredith & Mantel, 2008).

The construction planning is characterized as an iterative process in consequential phases and with increasing levels of detail (Ciribini et al., 2016), like the master schedule, the development and testing schedule and the assembly schedule (Meredith & Mantel, 2008). The construction schedule is the reference point for calculating the deviations and determining corrective measures as construction progresses. To date, the burden of planning construction activities has been carried almost completely by the project planner (Ciribini et al., 2016). Project planners have a challenging role, as they are not only concerned with the generation of a feasible plan, but also with the generation of a good one.

Many constraints complicate the planning process such as those related to the availability of resources, the completion times for the work tasks or the limitations on project budget (Zozaya-Gorostiza et al., 1990). Also, the quality and validity of construction planning depend on the experience of the project planner. As there is no database to refer to gather the necessary information, construction planning is based on judgements and experience-based learning (Zozaya-Gorostiza et al., 1990; Winch & Kelsey, 2005; Ciribini et al., 2016). This implies a high probability of making mistakes even for the most experienced planner, especially when there is not a strong collaboration with other actors of the process (Ciribini et al., 2016).

Because the construction planning is only an estimation of what and when things must be done to achieve the scope or objectives of the project, it is always carried out in an environment of uncertainty

(Maylor, 2001; Meredith & Mantel, 2008). Poor estimates or schedules can easily result in significant construction cost increases or delays (Zozaya-Gorostiza et al., 1990). While it may be possible in the short-term that the estimates for the construction planning are correct, in the longer-term stronger factors are coming into play and thus increases the chances for uncertainty (Maylor, 2001). Similar effects may be obtained because of inappropriate or inconsistent decisions concerning the technologies to be used when performing the work tasks (Zozaya-Gorostiza et al., 1990). Therefore, construction planning must include allowances for risk and features that allow it to be adaptive (Meredith & Mantel, 2008).

2.4.1. Scheduling techniques

Some planning techniques prove to be useful in project management. Evaluation and Review techniques (PERT), the Critical Path Method (CPM) and Gantt charts are the common techniques. The basic idea behind these techniques is to create a network of activity and event relationships that visualizes the sequential relationships between the tasks in a project. Tasks that precede or follow other tasks are then clearly identifiable over time. Such a network is a very useful tool for planning and controlling a project (Meredith & Mantel, 2008).

As the use of PERT and CPM have sharply decreased in the last decennium, Gantt charts are always to be found on construction sites. According to Meredith & Mantel (2008), one of the oldest but most useful methods of presenting project schedule information is the Gantt chart, developed around 1917 by Henry L. Gantt, a pioneer in the field of scientific management. In a Gantt chart, the project's activities are shown on a horizontal bar chart with the horizontal bar lengths proportional to the activity durations. The activity bars are connected to predecessor and successor activities with arrows (Meredith & Mantel, 2008). Gantt charts have several advantages. Meredith & Mantel (2008) explain that Gantt charts are easily understood and easy to maintain. In addition, they explain that Gantt charts are an easy way to show the current state of a project. However, Gantt Charts also have disadvantages. Meredith & Mantel (2008) explain that when a project is complex and consists of a large set of activities, it may be very difficult to follow multiple activity paths through the project. The Gantt Chart is, therefore, sometimes superficial and does not clearly illustrate the relationships between activities.

As Gantt charts are representing the fundamentals of project management, it is indicated that it has limitations. The Gantt chart alone is characterized as a blunt instrument (Maylor, 2001).

2.4.2. Scheduling tools

There are different commercial scheduling tools available, where Primavera Project Planner, Asta Powerproject and Microsoft Project are the most common software applications for scheduling and controlling construction projects. The tools provide relatively accurate scheduling and tracking of project progress, where the planning information can be updated, such as actual start time and percentage completion to-date (Omar & Nehdi, 2016). Moreover, several commercial tools enable practitioners to use small tablets on construction sites to efficiently collect and sent back information to head office, like daily site reports, notes and photos.

Besides using lean management in construction planning, there is also agile management. This will be handled in the following paragraph.

2.5. BIM

What is typical for the construction industry is the fact that there are many parties involved in construction projects, working towards a mutual goal (Tauriainen et al., 2016). However, many challenges are emerging when there are multiple parties involved with different perspectives and expertise, mostly leading to mistakes and misunderstandings. Improvement of this interaction may, therefore, contribute to an improvement of the construction process as a whole (Hassanien Serror et al., 2008).

As the construction industry has missed the opportunity to adapt to the digital revolution that significantly improved productivity, cost-efficiency and sustainability in other industries (Heigermoser et al., 2019), it was mostly driven by Computer-Aided Design (CAD). Currently, the construction industry is undergoing significant changes, where the implementation of the new technology Building Information Modelling (BIM) (Bonduel et al., 2017) and Lean management (Sohi et al., 2016) are the game-changers.

BIM is one of the most significant changes in the design industry, after the use of CAD, and, according to (Tauriainen et al., 2016), it has the potential to enhance the whole construction industry. With a variety of software systems, BIM is transforming the way construction projects are designed, engineered, built and managed (Ahuja et al., 2017). Many definitions are given for Building Information Modelling (BIM). Bonduel et al. (2017) state that BIM is a digital 3D object-oriented database of a building. Heigermoser et al. (2019) cite in Sacks et al. (2013) that BIM is a modelling technology to produce, communicate, and analyse building models. They add that particular emphasis should be given to the process part of BIM, and not the model itself. In particular, BIM provides a collaborative decision-making platform to facilitate information sharing for model simulation and project management disciplines (Heigermoser et al., 2019). Eadie et al. (2013) define BIM as the process of generating, storing, managing, exchanging, and sharing building information in an interoperable and reusable way. They add that it requires the development and use of a computer-generated model to simulate the planning, design, construction and operational phases of a project. The U.S. national BIM standard (NBIMS-IS) in Kamel & Memari (2019) defines BIM as the act of creating an electronic model of a facility for visualisation, engineering analysis, conflict analysis, code criteria checking, cost engineering, as-built product and budgeting. Tauriainen et al. (2016) cite that BIM significantly reduces design conflicts by relying on one information source and enabling clash checking. It has enabled a better visualisation of form and evaluation of function. Tauriainen et al. (2016) add that other benefits of using BIM include easier generation of design alternatives, better maintenance of information, and design model integrity including reliance on a single source of information and active clash detection. In addition, design requirements are easier to define, and information flows are improved (Tauriainen et al., 2016). Eadie et al. (2013) state that by using BIM across the lifecycle of a project, it produces the highest positive financial impact. The use of BIM is expanding in different areas (Kamel & Memari, 2019). Fadeyi (2017) states that BIM knowledge repository is in n-Dimensions: 3D (model), 4D (model with time), 5D (model with cost) and 6D (maintenance and operations). He adds that BIM 3D constitutes virtual modelling, virtualization of building and its related site location conditions and features, model walkthrough, prefabrication of building system and clash detection information.

The concept of BIM has also been adopted by most of the commercial CAD software, including Autodesk Revit, Bentley Architecture, GraphiSoft ArchiCAD and Tekla (Zhang & Hu, 2011).

2.5.1. The concept of the interaction of Lean management and BIM

Lean Construction is a construction management philosophy focused on creating value for the customer and eliminating non-value adding activities. Whereas, BIM is mainly focused on applying information technology over the entire project lifecycle (Heigermoser et al., 2019). Although Lean management and BIM are approaches with quite different initiatives and are not dependent on each other, both have a profound impact on the industry (Ahuja et al., 2017; Heigermoser et al., 2019). It is indicated that their interaction enables various opportunities to create a more efficient workforce and more effective processes of construction projects as well as challenges in its implementation (Heigermoser et al., 2019). Sacks et al. (2013) in Heigermoser et al. (2019) assessed the contribution of the dependency of these two approaches and made a matrix of it, which were also supported with empirical evidence. The BIM functionalities with the highest concentration of unique interactions were defined Sacks et al. (2013) in Heigermoser et al. (2019):

a) Aesthetic and functional evaluation;
b) Multi-user viewing of merged or separate multi-discipline models;
c) 4D visualisation of construction schedules;
d) Online communication of product and process information.

In all, having a synergy between these two approaches and applying them simultaneously can efficiently enhance the productivity of construction projects (Tauriainen et al., 2016; Heigermoser et al., 2019). Additionally, BIM provides an effective platform for implementing lean principles (Ahuja et al., 2017).

2.5.2. Industry Foundation Classes

Laakso & Kiviniemi (2012) in Hamledari et al. (2017) say that the industry foundation classes (IFC) are the neutral and open data exchange format introduced by BuildingSMART to enable interoperable and collaborative use of BIMs over the whole life cycle of a building. The use of IFC in the AEC industry and different stages of a project's life cycle provides a joint solution for the exchange of large amounts of data between project members (Dankers et al. 2014; Oh et al. 2015; Shen et al. 2010 in Hamledari et al., 2017).

There are different IFC specifications available, where the IFC2x Edition 3 (IFC2x3) is the common one. The reason is because of its common use and reliable support of BIM tools. The IFC2x3 data contains 653 entities with a prefix of IFC, like IfcWall. This entity directs the user to the walls that are modelled. An IFC file that is exported from a BIM model contains various sets of entities. If a BIM user exports a model to IFC, then this is usually done in a so-called 'STEP' file. STEP is a data format, also referred to as ‘standardized graphic exchange format’, developed by ISO and is published as ISO 10303. STEP covers a scope broader than many other BIM file formats. The STEP format is always expanding to cater to new requirements defined by the industry (Chua et al., 2017).

When opening such a format in a text editor like 'notepad', the user will see a text file with ID numbers and references. The object types such as 'IfcBuilding' are also visible. By referring to ID numbers, the computer can build an entire IFC model.

Hamledari et al. (2017) say that entities that can be used independently are subtypes of IfcRoot and are given a globally unique ID (GUID) with a string of 22 characters and a property history (OwnerHistory attribute). Entities in the IFC source schema, such as IfcLocalTime, IfcCalendarDate, and IfcDateAndTime, which act as shared time sources, do not own these attributes (Hamledari et al., 2017).

Hamledari et al. (2017) add that each entity can be cited as an attribute of another entity, see Figure 3. The OwnerHistory attributes, as in instances #18695, #18696 and #18706, have a value equal to #5, which are already defined at the beginning of the data model. The order in which the instances appear in an IFC file does not affect the semantics of the model, and their IDs can be changed after each export and import of the model.

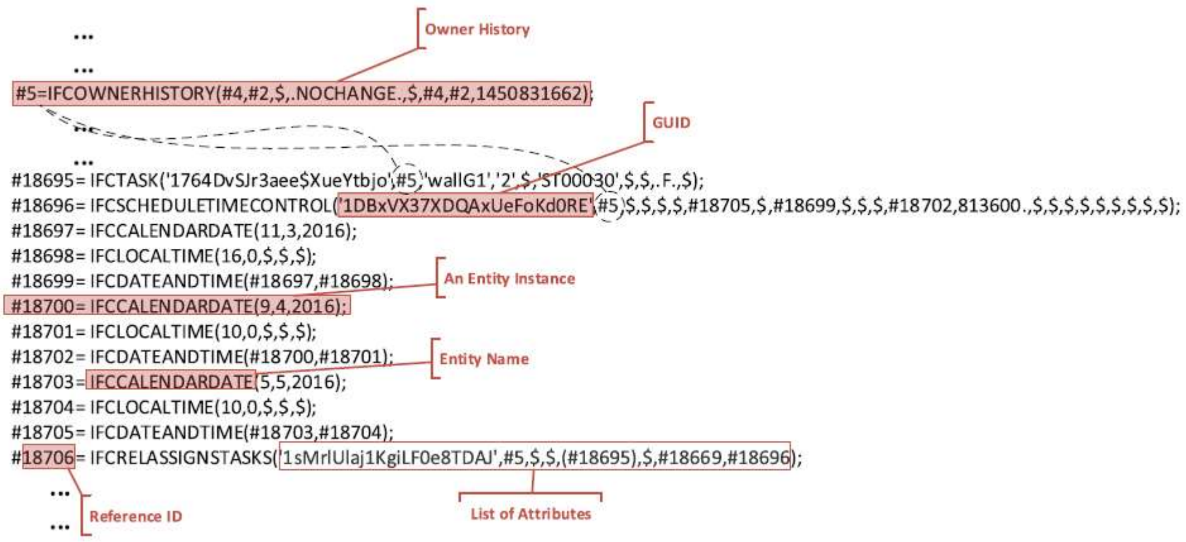


Figure 3: Parts of an IFC data model (Hamledari et al., 2017)

2.6. BIM 4D

The emergence of BIM technologies in the architecture, engineering and construction industry have allowed researchers to combine it with construction planning, which has led to the development of 4D modelling (Omar & Nehdi, 2016). Because of the high popularity of use and developments in the last decade, a variety of terms are mentioned in the literature, including 4D Modelling (Golparvar-Fard et al., 2011; Zhou et al., 2015; Omar & Nehdi, 2016), 4D planning and Scheduling (Fadeyi, 2017), 4D Plan (Han & Golparvar-Fard, 2015) and 4D CAD (Reizgevičius et al., 2013; olde Scholtenhuis et al., 2016; Zhang & Hu, 2011). Although these are mentioned as different terms, it has one thing in common, and that is that the BIM 4D-concept is defined as the "linking a schedule to a 3D-model to improve construction planning techniques" (Jupp, 2017). Koo & Fisher (2000) in Jupp (2017) state that to generate a BIM 4D-model, there are three requirements needed. These requirements are (1) 3D geometric model with building components; (2) construction planning (with activity data, running times, logical relationships); and (3) 4D simulation tool that makes it possible to link 3D model elements to activity data. For the sake of consistency, the term BIM 4D is used.

2.6.1. The fundamental reasons for using BIM 4D

Many researchers conducted studies in order to prove how effective BIM 4D is during construction processes. Tauriainen et al. (2016) say that construction planning can be improved by utilizing BIM 4D. Jupp (2017) indicates that BIM 4D makes it possible to carry out constructability analyses during the pre-construction phase and that the monitoring of activities on the construction site during construction has increased because the use of it. Furthermore, a high level of onsite construction performance is identified by using BIM 4D as a method to track the actual progress and analyse the effects of delays on the overall project planning. Also, BIM 4D can improve the management of construction safety, workspaces and waste (Jupp, 2017). Reizgevičius et al. (2013) compared the use of BIM 4D with traditional (2D) planning through an experiment by two groups of participants. They concluded that the use of BIM 4D allowed under the limited resources of time, to achieve almost twice bigger efficiency. Also, they add that BIM 4D can help to increase the effectiveness of communication and reduce the opportunity for all interested in the interpretation of the construction project participants. Zhou et al. (2015) say that by integrating the temporal and spatial information in BIM 4D, it can assist and facilitate the efficiency of communications among decision-makers. In addition, as BIM 4D can visually simulate a proposed construction sequence, planners could quickly identify potential constructability problems before construction begins (Zhou et al., 2015). Akanbi et al. (2019) say that BIM 4D is a lean management method to increase visualisation and help the communication effectivity between practitioners. They noted that because of the increased visualisation by BIM 4D, it made operations and quality requirements clearer using charts, displayed schedules, highlighted inventory designation and tool locations. Also, Khodeir & Othman (2018) concluded that by utilising BIM 4D as a lean tool to their test pilot project in phase two, it leads to 20% reduction of labour hours and material wastes by 6% compared to phase one, where only the traditional planning was used. In addition, the reduction of material waste reduced 7.5 tons of CO_2 emission during the production, transportation and installation (Khodeir & Othman, 2018). Jupp (2017) states that because the fact that construction and management processes rely on the collaborative effort of different parties within the construction industry, the success of the project rely on the quality of the information flows between de parties. The use of BIM 4D shows the improvement of integration and continuity in information flow (Jupp, 2017). Also, gaps in communication and poor information flow between project participants, from the upper to lower levels of the project management hierarchy is improved by BIM 4D (Jupp, 2017).

In all, BIM 4D can improve many aspects of construction planning, such as scheduling, communication, information flow, production control and the onsite management of safety, workspaces and waste (Jupp, 2017; Khodeir & Othman, 2018; Akanbi et al., 2019). Utilising the use of BIM 4D lead to an improvement of communication and information flow throughout the initial to execution phase (Jupp, 2017).

2.6.2. BIM 4D tools

Improvements to the flow of information and communication, as well as the outcomes of environmental planning and management activities, can be made possible by leveraging BIM 4D tools, methods, and visual simulations (Jupp, 2017).

In the context of methods and visual simulations, BIM 4D tools provide in-depth construction conflict analysis and collision detection for movable site facilities (Zhang & Hu, 2011). In the context of management activities, BIM 4D tools can help to: develop an intuitive understanding of a project, better communicate, review designs and to detect conflicts and review project progress (Zhou et al., 2015; olde Scholtenhuis et al., 2016). The progress of a construction project can be simulated, and it can be used to: evaluate the quality and quantity of constructability suggestions, identify the conflicting space usage of activities, movement of personnel and positioning of temporary works and monitoring on-site progress (Zhou et al., 2015). Since the first proposals of the BIM 4D approach, special BIM 4D tools now allow for the integration of multiple aspect models and planning data to link 3D objects to individual activities (Jupp, 2017). In addition, BIM 4D tools also potentially support cost estimating (Zhang & Hu, 2011; olde Scholtenhuis et al., 2016). Different BIM 4D tools are emerging in the commercial market, where Common Point Project 4D, 4D Suite, Visio, Vico Office Suit, ConstrucSim, Synchro Pro, Autodesk Navisworks and Primavera are one of them (de Soto et al., 2017).

2.6.3. BIM 4D uses

Different BIM 4D uses are allocated in literature. In Jupp (2017), the capabilities of BIM 4D is divided into two categories, namely construction planning and site planning. Jupp (2017) states that for construction planning it includes: winning work at the tendering stage, planning the construction method, communication over time, review of the design, management of resources, workspace planning, hazard identification and safety planning. For construction site planning, Jupp (2017) states that it includes the management of site logistics, pedestrian and traffic flows, the supply and storage of materials, major site activities, temporary works, welfare facilities and safety on the construction site. In Zhou et al. (2015), the application of BIM 4D is categorized into two groups, which are at the activity level and the operations level. At the activity level, BIM 4D is achieved by linking the 3D model and construction planning. However, at this level, there is a lack of attention to temporary components and resources such as scaffolding and crane movements which are not integrated into the 3D model. Also, the space needs and congestions that possibly can arise during temporary works cannot be made visible. In this way, it is hard to improve the safety, quality and productivity during temporal works. For that reason, Zhou et al. (2015) addressed the second group of application of BIM 4D, which is the operations level. A significant difference based on the activity level is the fact that the BIM 4D is enriched with 3D objects that represent temporal materials. With this approach, several benefits are identified by Zhou et al. (2015), which are: being able to view the interaction of the various resources, increasing the accuracy of a project's schedule and improving coordination of activities. Zhou et al. (2015) conclude that by using BIM 4D at the operations level, it supports planners in the process of examining the schedule and identify potential problems before actual construction. In addition, it allows for a more intuitive comprehension of the construction process than traditional 2D drawings and scheduling applications such as Gantt chart and Pert chart information. Another study (Zhang & Hu, 2011) categorised BIM 4D use into four levels. The first level is a simple combination of 3D model and schedules, and then construction activities and resources (labour, material, machinery included) can be imported to level up the model. The third level is an extension of site entities, which are also connected with a schedule. In the fourth level, structural information for mechanical analysis is further augmented (Zhang & Hu, 2011).

2.6.4. The development of BIM 4D with field data technologies

Besides BIM 4D is used by only combining 3D models and construction planning, some studies focus on other themes that extend the capabilities of BIM 4D. One of them is the focus on developing methods and algorithms that integrate BIM 4D with field data technologies. Field data capturing technologies include imaging technologies, geospatial technologies and radiofrequency technologies. The BIM-based progress monitoring method uses the obtained on-site (point cloud) data and compare it with BIM 4D, and identify the differences between both models in a well-interpreted way (Pučko et al., 2018). This approach, also known as Scan-vs-BIM (Bonduel et al., 2017; Bueno et al., 2018), provides efficient and timely data analysis and visualisation of deviations in an early stage to prevent potential upcoming delays and enables timely decisions for corrective actions (Pučko et al., 2018). Also, experiments showed that such automatic updating of BIM 4D-models meets or exceeds manual progress tracking and monitoring (olde Scholtenhuis et al., 2016). Development of new tools and methodologies allowing automated progress monitoring, in terms of data acquisition, information retrieval, progress estimation and visualisation of the results, is therefore of vital importance (Pučko et al., 2018). It is stated that the development of automated solutions, whereby imaging technology is used has proven to be the most effective data acquisition method (Han & Golparvar-Fard, 2015; Kopsida et al., 2015; Pučko et al., 2018).

<u>State-of-the-art of field data technologies used in combination with BIM 4D</u>

To the authors best knowledge, progress tracking with the use of 3D sensing technologies and the linking with construction planning was first initiated by Bosche et al. (2009). They introduced a semi-automated approach for project progress tracking by fusing 3D BIM-model and time-stamped 3D laser scanned data. The approach is based on a surface-based recognition method. The recognized area is calculated for each object, and when that area exceeds a minimum threshold, the object is considered as recognized. Turkan et al. (2012) took the approach of Bosche et al. (2009) to the next level. The progress tracking system presented in Turkan et al. (2012) uses a BIM 4D-model to improve recognition of BIM model objects from their 3D laser scans. The approach also implements an automated progress feedback loop and uses a logical inferencing algorithms. All that is required is to acquire manually 3D laser scan data in the same coordinates as the BIM model by selecting at least three pairs of corresponding points both in the scan and in the model. Turkan et al. (2012) tested their approach with the data collected from a concrete structure on the construction site. Brilakis et al. (2010) proposed an approach to automate the generation of as-built BIM of constructed facilities by using composite video and 3D laser scan data as input. They stated that as-built data could be collected automatically using laser scanners, but interpretation and merging of point clouds, stitching and object fitting are all performed manually. Cheok et al. (2000) performed real-time assessment and documentation of construction activities such as site preparations based on 3D as-built models using a 3D laser scanner. Abeid & Arditi (2003) introduced a planning and progress control system called Photo-net. The system links time-lapse images of construction works to the Gannt chart for monitoring of the progress. Golparvar-Fard et al. (2009) proposed an alternative image-based method for monitoring construction progress using daily images taken from a construction site. They have calibrated series of images of the construction site and reconstructed a low-cost 3D as-built point cloud. This made it possible to visually compare the as-built data with the planned 3D data and monitor the construction progress. Golparvar-Fard et al. (2012) use voxels and a probabilistic model to detect construction progress. Wu et al. (2010) developed an object recognition system to recognize construction objects from the construction site and estimate project status information by using digital images as an image-based method. Their approach exploits advanced imaging algorithms and a 3D BIM perspective view to increase the accuracy of the object recognition, and thus enables acquisition of project status information. However, the method of Wu et al. (2010) requires a manual selection of common points between the scan and the images to calculate the volume of the objects. Calculating the progress for an entire construction project using this method would therefore require a considerable amount of manual data processing. Omar & Nehdi (2016) used range images to manually compare it with BIM 4D to measure deviations from the actual planning. Pučko et al. (2018) presented a novel approach to automate construction project progress monitoring, based on the comparison of the BIM 4D and point clouds (Scan-vs-BIM). The acquisition of point clouds on the construction site went through 3D scanning devices, which were equipped on the protection helmets of workers. Zhang & Arditi (2013) developed a method that counts the number of points in the related portions of the point clouds. Han & Golparvar-Fard (2015) proposed a new appearance-based material classification method for operation-level monitoring of construction progress using BIM and daily construction photologs. Point cloud models were generated from the construction site images using 3D reconstruction procedures. An average accuracy of 95.6% was achieved on appearance-based recognition of progress, and the effects of deviations between BIM 4D and the point cloud on this accuracy were thoroughly investigated. Rebolj et al. (2008) compared site images with 3D model using an algorithm that recognizes differences between element characteristics. The method cannot however be applied for interior works such as painting or tiling.

2.6.5. Challenges of detecting progress using field data technologies

Kopsida et al. (2015) compared the advantages and disadvantages of real-time data capturing techniques on the construction site based on photogrammetry, videogrammetry, and 3D laser scanning. In addition, Omar & Nehdi (2016) illustrated the categorization of the monitoring methods and the data acquisition technologies, as well as methods for data processing. Although scanning technologies have already been used in the construction industry for several applications, their full potential has not been achieved yet. This is because the currently available commercial packages do not allow automated segmentation of the data at the object level. Also, some manual and sometimes semi-automated approaches exist, but they are very time consuming, must be operated by experts, and are thus very expensive (Turkan et al., 2012).

Regardless of the sensing method used, the common challenges among these methods for detecting operation-level progress are the following (Han & Golparvar-Fard, 2015):

1. Lack of detail in the plan model: The level of detail (LOD) in BIM used in most projects for pre-construction coordination is not sufficient to monitor construction progress per element. This is because BIM models are modelled LOD300 to 400. In this way, many elements in BIM 4D-models have a one-to-many relationship with planning activities. For example, the same 3D element representing a concrete foundation wall corresponds to both "placement" and "waterproofing" activities, without necessarily having unique 3D physical representations for each layer. Also, these models typically do not contain any representation for the temporary structures as formwork or shoring (Figure 4).
2. High-level of work breakdown structure (WBS) in construction planning: In many projects, the construction planning remains at the execution-level. For example, there might be one activity "Form/Rebar/Pour/Strip Concrete Foundation Walls" representing the placement of all concrete foundation walls in a building project. Although operational details may include forming, installing reinforcement bars, placing concrete, and finishing (Figure 4), yet such details are typically not available in the underlying WBS of the planning tied into a BIM 4D.
3. Static and dynamic occlusions: Occlusions result in incomplete data capturing due to limited visibility by a scanner. There are two kinds of occlusions, which are static and dynamic occlusions. Static occlusions are caused by fixed objects, like a wall blocking the line of sight of a scanner to a column. Dynamic occlusions are caused by temporary and moving objects, such as scaffolding, shoring, equipment and workers.

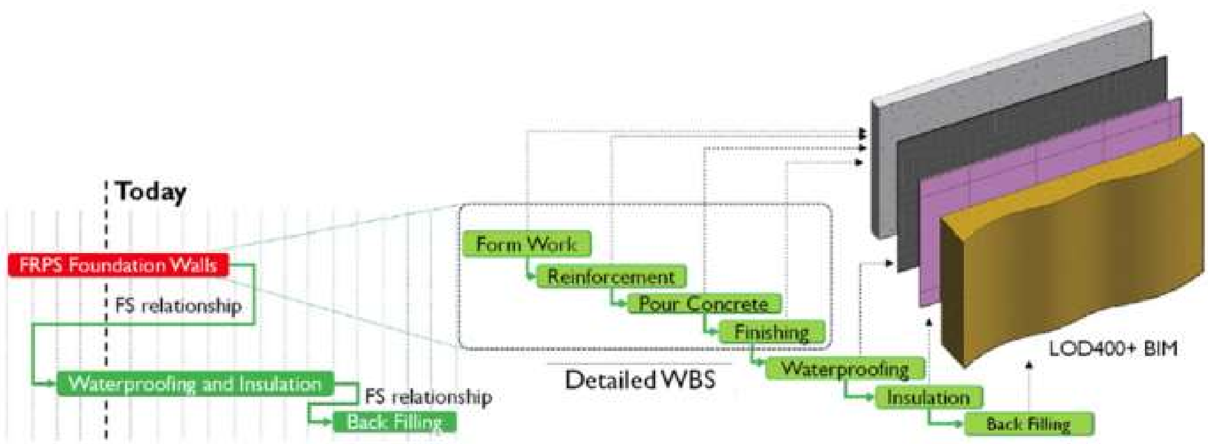


Figure 4: The contrast in LOD in BIM and construction planning for a concrete wall: (left) a construction planning on an execution-level which is assigned to a BIM object, (right) the necessary LOD and WBS for execution-level progress monitoring (Han & Golparvar-Fard, 2015)

As the field data technologies are already introduced, the following paragraph goes into detail about how construction progress can be monitored using these technologies.

2.7. Monitoring of the works

Falling behind schedule and having deviations between the as-built and designed baseline plans are unfavourable events that often occur in construction projects (Omar & Nehdi, 2016). In order to know what the deviations are, monitoring is needed. Golparvar-Fard et al. (2009) define the term monitoring as collecting, analysing, recording and reporting information concerning key aspects of project performance at the appropriate level of detail required by project managers and decision-makers.

Constant monitoring helps to detect early or potential schedule delays at construction sites and is crucial for reducing cost and schedule overruns and enhances quality control, documentation and communication (Golparvar-Fard et al., 2009; Omar & Nehdi, 2016; Hamledari et al., 2017b). In this way, it provides the opportunity to initiate remedial actions and increases the chance of controlling such overruns or minimizing their impacts (Golparvar-Fard et al., 2009). Currently, inspections and monitoring of the works on constructions sites are done by superintendents and site managers. They execute this manually and gather information about the progress of activities (Golparvar-Fard et al., 2011; Kopsida et al., 2015; Pučko et al., 2018). Hence, they can measure the actual progress of the construction and detect deviations from the baseline construction planning (Behnam et al., 2016).

However, the process of inspections and monitoring is identified as labour intensive, expensive, time-consuming, infrequent, prone to human errors, uncertain, inconsistent, and inefficient (Golparvar-Fard et al., 2009; Han & Golparvar-Fard, 2015; Kopsida et al., 2015; Omar & Nehdi, 2016; Hamledari et al., 2017b; Pučko et al., 2018).

Such an approach has also been recognized as one of the major problems that cause project delays and cost overruns (Kopsida et al., 2015; Omar & Nehdi, 2016).

Progress monitoring is considered as a critical success factor for projects to be delivered on time and within budget (Saqib et al., 2008) and as one of the most challenging tasks due to the complexity and interdependency of activities (Kopsida et al., 2015). As a result, project managers face considerable challenges to encounter these. Despite the fact the progress monitoring is crucial, the construction industry does not have efficient monitoring systems compared to other industries (Navon & Sacks, 2007). This is because the progress data that are acquired during manual monitoring are documented in monthly/weekly progress reports in the form of Gantt charts or other formats as well as construction site photos. In some large projects, these monthly reports may include a few hundred pages of long Gantts charts and descriptions which are challenging to store, read and analyse (Behnam et al., 2016). It has been identified that preparing and collecting the data of the progress during on-site monitoring is difficult, time-consuming and costly (Kopsida et al., 2015; Behnam et al., 2016). In addition, the manual collection of actual progress information from numerous work locations and the preparation of progress reports are often delayed and subject to inaccuracies and misunderstandings (Kopsida et al., 2015; Behnam et al., 2016). Also, it is noted that the estimation of the percentage of completion is highly subjective (Kopsida et al., 2015).

On 14 May 2019, the Dutch parliament adopted the Quality Assurance for Construction Act (Wet kwaliteitsborging voor het bouwen). This Act aims to improve the quality of construction and to increase the liability of contractors. The contractor must be able to prove to the client that the requirements set for construction have also been met after execution. The Quality Assurance Act has three principles: improved (safeguarding) building quality, an improved position of the consumer and stimulating quality improvement, and reducing failure costs. From 1 January 2021, the new system will enter into force gradually. One of this Act's requirements is to verify if the construction project that is delivered conforms to the planning. Currently, in order to prove the progress of construction, contractors are providing clients with progress data. Occasionally, however, these contain old and wrong data. Behnam et al. (2016) state that the potential delays in the preparation of progress data and its delivery to clients cause inefficiency in the communication process on construction progress.

In order to improve the current way of monitoring, many research focussed on the continuous data capture and (automated) tracking methods of works on the construction site which may benefit many disciplines (Hamledari et al., 2017b) and achieve project objectives (Omar & Nehdi, 2016). Efficient progress monitoring systems can help automate progress inspections, reduce the risks of error, facilitate proper and timely corrective actions, and prevent deviations in terms of cost and schedule (Kopsida et al., 2015).

In recent years, there has been increasing momentum toward utilization of digital images for automated data collection at construction sites using imaging and geospatial technologies (Hamledari et al., 2017b). These visual resources are easy to capture, and they provide a less computationally intensive and more cost-effective alternative to other reality capture technologies (Hamledari et al., 2017b). The use of digital images offers a robust means of detecting small changes in the appearance of construction elements.

Hence, there is a need for solutions that can automate both the visual detection of components of construction elements and the inference of the construction elements' current state (Hamledari et al., 2017b). The visual detection of these elements can help document and identify the current state of work, provide many disciplines with situation awareness, and provide future vision-based progress tracking systems with information about the actual state of work (Hamledari et al., 2017b). It can also aid in the timely detection of potential time delays and construction deviations and directly supports project control decision-making (Omar & Nehdi, 2016).

2.7.1. Construction site information

Before going onwards to how construction sites can be captured in order to track construction activities, it is essential to know what information can be extracted. Omar & Nehdi (2016) say that site information can provide feedback for various purposes, including progress measurement, equipment and material tracking, safety planning and productivity tracking. Besides that, construction site information has generally been organized into three main categories, namely finance, quality, and progress (Omar & Nehdi, 2016). One of the necessary actions in construction management systems is to early assess the as-built status during construction for effective and efficient corrective construction planning (Omar & Nehdi, 2016). Omar & Nehdi (2016) cite in NYSOT (2016) that the characteristics of efficient methods of progress and performance measurement can be summarized as follows: measurable, reliable, consistent, understandable, verifiable, timely, and unaffected by external influences, cost-effective, useful and suitable for effective decisions. At the moment, there is a lack of such a systematic evaluation and monitoring of the construction site.

2.7.2. Capturing methods

In the past, many enquiries were devoted to project control, where progress tracking is a critical part of it. Construction progress tracking is, however, not a simple task and is associated with many challenges. This is because, as indicated before, construction projects are complex and involve large amounts of information related to a variety of functions, such as scheduling, construction methods, cost management, resources, quality control and change order management (Omar & Nehdi, 2016). Also, it generates a wide variety of forms of information provided by several different sources. In this way, the information flow is not consistent, leading to errors.

A number of advanced automated data collection technologies are used today for real-time on-site progress tracking. In addition, automated data collection at construction sites has been studied using different technologies such as imaging technologies, geospatial technologies and radiofrequency technologies. The technologies that can generate point clouds are discussed below, which are in this case, imaging technologies.

2.7.2.1. Imaging technologies

Research on progress tracking using imaging technologies has been rapidly growing in recent years, such as photogrammetry, 3D laser scanning, videogrammetry and range images (Behnam et al., 2016; Omar & Nehdi, 2016; Hamledari et al., 2017b). All these imaging technologies have already been studied to generate 3D information about various objects on site for use in progress analysis.

<u>Photogrammetry</u>

In recent years, there has been a dramatic increase in the number of photos being captured on construction sites, due to the availability of inexpensive point-and-shoot and time-lapse cameras as well as smartphones (Golparvar-Fard et al., 2009; Golparvar-Fard et al., 2011; Han & Golparvar-Fard, 2015; Kopsida et al., 2015; Omar & Nehdi, 2016; Hamledari et al., 2017b). Besides the fact that the cameras are inexpensive nowadays, it does also not require much training (Golparvar-Fard et al., 2011). Construction photos are mostly used for the following: visualisation of construction operations and their sequences, progress monitoring and tracking of construction crew and machinery, productivity measurements, accident investigation, dispute resolution and quality assurance or quality control (Golparvar-Fard et al., 2011). The use of capturing photos provided opportunities for researches to develop and introduce computer vision techniques for automating the extraction of project-related information. As a result, many research were devoted to automated vision-based methods in the context of the construction industry, leading to image-based 3D reconstruction solutions to generate point clouds using thousands of overlapping images (Hamledari et al., 2017b). This 3D reconstructed model is also called an as-built 3D model, which can be compared to 3D BIM models to automatically calculate the percentage completion of each component and measure the progress of construction projects (Omar & Nehdi, 2016). Image processing studies proved to be cost-effective and efficient for automated recognition and tracking of resources such as workers and equipment, classification of materials, productivity analysis, recognition of structural elements, and condition assessment (Golparvar-Fard et al., 2009; Hamledari et al., 2017b). Conversely, the application of photogrammetry was limited in construction projects due to: (1) the considerable time of computation process; (2) the sensitivity of the region of interest and detectors to different lighting conditions, particularly in the presence of severe shadow lines, affecting image processing; (3) progress can be monitored only on the closest structural frame of the component to the camera and; (4) it requires an extensive amount of human intervention, making the application me-consuming and less attractive for repetitive progress-monitoring tasks (Kopsida et al., 2015; Omar & Nehdi, 2016).

<u>3D laser scanners</u>

Three-dimensional (3D) laser scanners, also known as LADAR (Laser Detection and Ranging), was first used as a method to capture construction sites accurately and analyse if this deviates from the BIM model (Krijnen & Beetz, 2017; Bonduel et al., 2017). Gradually, this technique can also be found in the literature to capture the current status of construction projects. 3D laser scanners acquire 3D point clouds by emitting a pulse of laser light to an object and calculating the distance to the object by timing the round-trip time of the pulse of light. One scan with this technology can result in millions of 3D points in a few minutes, where every point is described by x, y and z coordinates (Turkan et al., 2012; Kopsida et al., 2015; Omar & Nehdi, 2016; Sánchez Rodríguez et al., 2019). The final scan inputs the results into an as-built model and can be aligned and compared with the as-planned BIM model to detect deviations (Han & Golparvar-Fard, 2015; Hamledari et al., 2017b). 3D laser scanners have proven to offer more accurate point clouds compared to the ones generated using image-based solutions like photogrammetry. Still, there are some limitations to the use of it. Hamledari et al. (2017b) cite that the limitation of laser scanning-based methods includes its lower accuracy and data loss at spatial discontinuities caused by the mixed pixel phenomenon. In addition, laser scanners are challenged to generate accurate point clouds for reflective materials, and they are currently unable to provide semantic information for 3D models (Hamledari et al., 2017b; Krijnen & Beetz, 2017). Working without any semantic information of the scene, geometric reasoning is challenging and induces estimation errors (Golparvar-Fard et al., 2009). Golparvar-Fard et al. (2009), Turkan et al., 2012, Kopsida et al. (2015) and Omar & Nehdi (2016) state that 3D laser scanning is still not widely employed due to its high-cost need for a clear line of sight and the difficulty of using it in congested interior work. Also, like other sensing devices that depend on the line of sight, as the distance between the laser scanner and the building components increases, the level of detail that can be captured is reduced. Finally, since scanners are not easily portable, they cannot efficiently be used for scanning indoor environments (Golparvar-Fard et al., 2009; Kopsida et al., 2015; Omar & Nehdi, 2016).

Videogrammetry

The videogrammetry technique is based on video recordings, which are sequential still frames that are composed in order. Since pixels are existing in each frame, a 3D model can be reconstructed based on the previous frame. This characteristic of videogrammetry enhances the reconstruction of the building site more quickly (Omar & Nehdi, 2016). Also, continuous advancements on cameras and performance processing units enhance the accuracy of the obtained data, reduce the time of processing and increase the potential of using visual data for as-built data acquisition purposes (Kopsida et al., 2015). Videogrammetry is most of the time combined with an Unmanned Arial Vehicle (UAV) in order to reach different angles of the building site easily. The level of accuracy for progressive reconstruction using videogrammetry has become reasonable using high-resolution cameras (Omar & Nehdi, 2016), but are, however subject to motion blur (Hamledari et al., 2017b). Hamledari et al. (2017b) state that the performance on images with higher resolution does not always result in higher performance and add that this problem can be tackled by increasing the number of initial labelling and iterations for high-resolution images.

Range images

Range images, also called depth images, are advanced technology based on digital images. They are equipped with sensors that measure the range to a target and offer an inexpensive and accurate means for digitizing the shape of 3D objects (Omar & Nehdi, 2016). This technology is used for many purposes in the area of tracking moving objects like construction equipment and materials. Also, range images are less costly than laser scanners but more expensive than photogrammetry. Furthermore, the application can only scan in limited range distance.

Omar & Nehdi (2016) compared the different imaging technologies with each other, focused on affordability, portability, processing time, point cloud accuracy and range distance. Table 1 shows the results.

Table 1: Comparison of 3D imaging methods of data acquisition (Omar & Nehdi, 2016)

Method	Affordability	Portability	Processing time	Point cloud accuracy	Range distance
Photogrammetry	✓✓✓	✓✓✓	✓	✓✓	✓✓
Laser scanning	✓	✓✓	✓✓✓	✓✓✓	✓✓✓
Videogrammetry	✓✓	✓✓	✓	✓✓	✓
Range images	✓✓	✓✓	✓✓	✓	✓

✓: Low; ✓✓: Medium; ✓✓✓: High.

Also, the combination of IT technique and imaging technology, such as Augmented Reality (AR) has made its appearance in the construction industry. This term is defined by Omar & Nehdi (2016), who define AR as "a live, direct or indirect view of a physical, real-world environment whose elements are augmented by virtual, computer-generated imagery." Examples of application areas are visualisation or simulation, communication or collaboration, information modelling, information access or evaluation, and safety or inspection (Omar & Nehdi, 2016). For this research, however, AR falls out of scope. This is because only image data acquisition of as-built data will be covered.

As imagining technologies can generate point clouds, it is the essence to discuss what it is, and how it can serve this research. The following paragraph specifies more into it.

2.8. Point clouds

A coordination system is a system where positions of points or other geometric elements can be determined. In this system, three axes intersect each other at point (0,0,0). This is also a basepoint in three-dimensional space. The first axis is labelled as x (x,0,0), the second axis is labelled as y (0,y,0) and the third axis is labelled as z (0,0,z). As in the case of planes from a cube, the coordinates on the first two axes can be defined as the sum of the points, like (x, y, 0) = (x, 0, 0) + (0, y, 0). The same applies to axis like (x, 0, z) and (0, y, z). Furthermore, three planes that are parallel to coordinate (0,0,0) is labelled as (x,y,z). This means that the first, second and third axis ((x, 0, 0), (0, y, 0), (0, 0, z)) meet each other. With this coordinate system, a cube with dimensions 1x1x1 can be defined as: (0, 0, 0), (1, 0, 0), (1, 1, 0), (0, 1, 0), (0, 0, 1), (1, 0, 1), (0, 1, 1), and (1, 1, 1).

Point clouds have the same character as a coordination system and include data points in the Euclidian space $\mathbb{R}^3$, represented by their corresponding X, Y and Z-coordinates (Nourian et al., 2016; Chua et al., 2017). This can be written as the following mathematical expression:

$p_i = \{x_i, y_i, z_i\};$

Resulting in:

$P = \{p_1, p_2, p_3 \dots, p_i \dots, p_n\}$

Besides the X, Y and Z-coordinates, each point P can hold additional data like RGB colours, point density, distances, vectors and curvature. Altogether, point clouds result in an n-dimensional data structure. A rich point cloud data can provide geometric information with more important details (Gao et al., 2015). However, point clouds are used to represent the surface of an object and thus do not contain data of any internal features, colour or materials (Chua et al., 2017).

There are different types of formats in which point clouds can be generated. The most common formats are PCD, PTS, PCG, RCS, PTX, RCP, E57, PLY, IMP, POD, ZFS, XYZ, TXT and PTG.

2.8.1. Types of point clouds

Two kinds of point clouds can be distinguished, namely organized point clouds and unorganized point clouds. An organized point cloud is a set of arrays where a data point has a fixed position in that set and array, similar to pixels having a fixed position in a flat image (Pörteners, 2018). Such organized point clouds are generated by RGB-D cameras, which creates a range of images. Pörteners (2018) explains that range images give a 2.5D representation of a scene. A 3D representation is created by reconstruction every pixel in a 3D coordinates system. In this sense, the XY location of the image and the Z location of the depth are merged. Pörteners (2018) adds that the resulting point cloud has to be scaled with a particular factor (usually included in the metadata of the image) to transform it to its true XYZ representation in meters.

In contrast with organized point clouds, Pörteners (2018) explains that unorganized point clouds contain a set of data points that do not hold a fixed ordering in relation to each other, i.e. points do have a fixed position in the Euclidian space. However, the arrangement of the data points in the set does not affect the point cloud representation. Pörteners (2018) states that to exemplify, every point $ppii$ in a given unstructured point cloud dataset PP described by an array with the point coordinates $\langle xx, yy, zz\rangle$, possible colour information $\langle RR, GG, BB\rangle$, point normal vectors $\langle uu, vv, ww\rangle$, curvature $\langle\sigma\sigma\rangle$ and extendable to other features such as intensity and category. It is irrelevant if point $ppii$ would be positioned before or after $ppjj$ in dataset PP, since the position is given by the point coordinates $\langle xx, yy, zz\rangle$.

The advantages of such organized point clouds are the efficient calculation of point normal vectors, faster computation times in nearest neighbourhood searches and their functionality within the machine learning domain for object recognition techniques (Pörteners, 2018).

2.8.2. Point cloud data acquisition

Point cloud data can be generated by the following field data technologies: 3D laser scanners, photogrammetry, videogrammetry and range images. 3D laser scanners are also called remote sensing technologies.

Remote sensing is a broad concept, which includes measuring the properties of objects without direct contact. So, all the data that are acquired from a distance. There are different ways of acquiring data by means of passive sensors and active sensors. Passive sensors capture reflected rays, which means that the sensor only observes the targeted object. Active sensors send information to the targeted object and receive feedback from it. Lidar, radar and sonar are one of the active sensors.

It is essential to understand the difference between photogrammetry and remote sensing. The application of remote sensing is to analyse and interpret images for deriving information about the targeted object. The application of photogrammetry, however, is to produce maps and precise 3D positions of points in order to make measurements.

Looking in-depth about the definition of photogrammetry, it is a 3D measuring technique that uses images as a fundamental resource for measurement. In other terms, it is a science of taking precise measurements from images. Photogrammetry can be classified into two types, namely: aerial and terrestrial.

Aerial photogrammetry is executed with the use of UAVs. These are equipped with cameras and fly over a predesignated flight route with different altitudes and camera angles. During the flight, the camera makes many images. More frequent overlaps of the images taken from the UAV provides a better 3D reconstruction of the area (Kattenborn et al., 2014). Aerial photogrammetry is commonly used for vast mapping areas of land and is not and will not be addressed further in this research. For more information about aerial photogrammetry, it is recommended to go through the book of Wolf & Dewitt (2000) and Dewitt et al. (2014).

Terrestrial photogrammetry is based on photos that are taken on the surface of the earth. The cameras that are used for terrestrial photogrammetry may be handheld, mounted on tripods or suspended from towers. Two kinds of terrestrial photogrammetry can be distinguished, namely static and dynamic capturing. Static capturing is used for stationary objects, and dynamic capturing is used for moving objects.

Figure 5 shows the difference between remote sensing and photogrammetry and also shows their application.

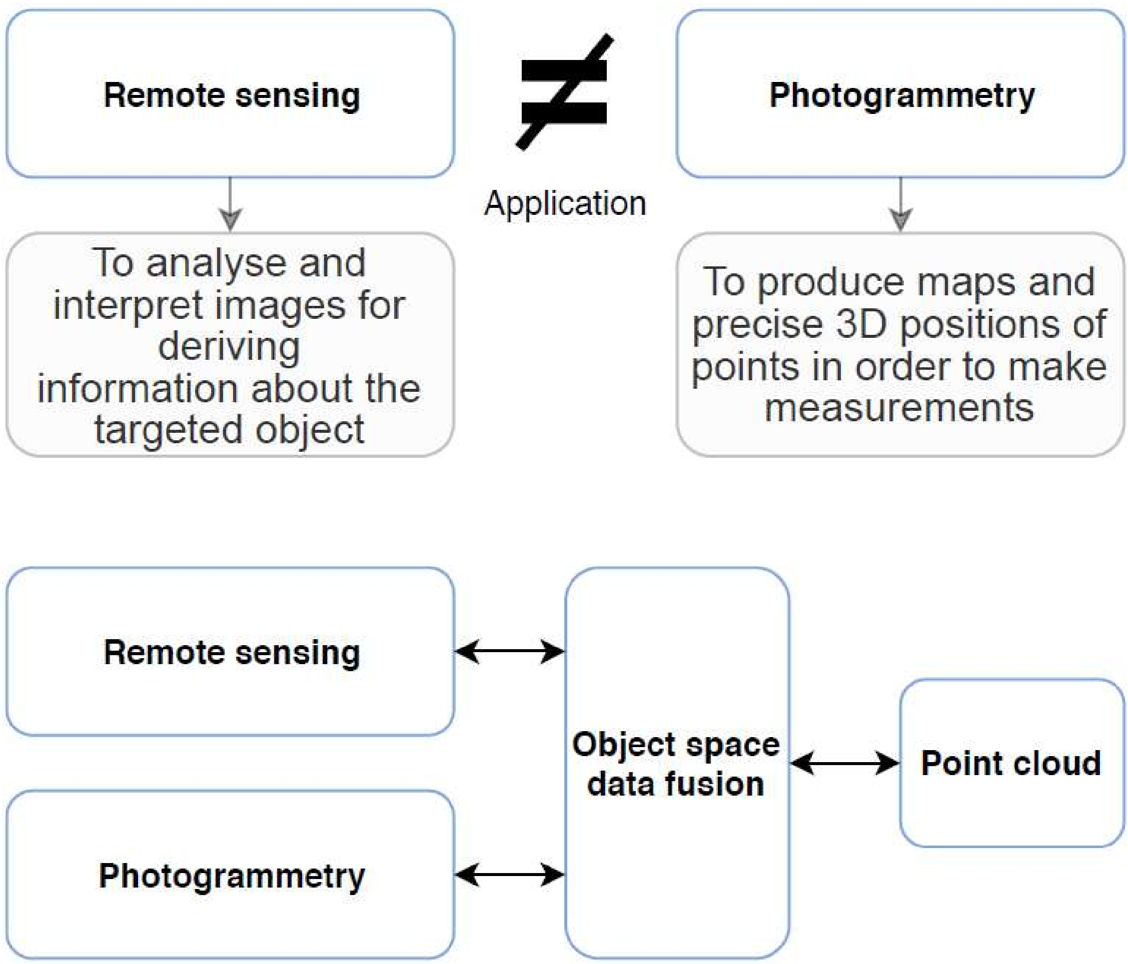


Figure 5: The differences between remote sensing and photogrammetry (Schenk, 2005) modified

2.8.3. Processing of point clouds

Point clouds contain immense data structures, and therefore, it requires tremendous computational power to handle such data. Different spatial index structures do exist in order to visualize efficiently and model point cloud data, such as kd-tree, quadtree, or octree derivations (Richter et al., 2014). These structures divide the point cloud into hierarchical cells that contain subsets of the total point cloud. Also, these cells are only visible when visualisation requires it, so that irrelevant points are not displayed (Krijnen & Beetz, 2017). Richter et al. (2014) argue that the construction of quadtrees and octrees can be done faster in contrast to Kd-trees, because there is no necessity to sort the points. However, the use of quadtrees and octrees for uneven and sparsely distributed data results in tree nodes with a varying number of points. This means that much memory and caching is needed in order to handle the data.

The use of 'voxels' is another way of processing point cloud data (Golparvar-Fard et al., 2012). Pörteners (2018) calls a voxel a portmanteau of the word 'volume' and 'pixel', and enables fast 3D visualisation or rendering in games because they can be efficiently stored in memory as they can store little in-memory data. This approach can be reproduced into point cloud data, where clusters of points can be grouped in a voxel. Pörteners (2018) explains that one way to store this voxelized point cloud data is by using octrees. Octrees divide the 3D space into eight octants, which are called 'nodes'. These nodes contain a specified number of data points for point clouds, and dense point clouds contain many small nodes. Such nodes can be visualized in 3D by giving them volumetric properties, each single node (the 'pixel') being given a volume, depending on the amount of parent nodes it holds.

Mesh-based modelling, also known as polygonal modelling, is another way of processing point cloud data and uses polygons, usually a triangle, to create a surface or a 3D model. Chua et al. (2017) state that in this way, the point cloud can be easily converted to a mesh-based model as the points can be easily connected to form a mesh. However, they add that with a more significant number of points, the mesh becomes very fine and more computational power is required to handle the file.

As the above-mentioned processing techniques do apply for unorganized point clouds, there are also techniques for organized point clouds. The structure from motion (SfM) technique is one of the remarkable ones. SfM dates back to early techniques in photogrammetry and has now emerged as an inexpensive method for extracting the 3D structure of a scene from multiple overlapping photographs using bundle adjustment procedures (Golparvar-Fard et al., 2011; Mlambo et al., 2017). The ability of SfM to generate high-quality 3D point clouds similar to the ones generated from 3D laser scanners is now widely understood. It has been demonstrated in several studies (Mlambo et al., 2017).

2.8.4. Quality level of point clouds

The quality level of point clouds depends on the Level of Accuracy (LOA) and Level of Detail (LOD). LOA represents the tolerance of positioning and dimensional errors (Zhang et al., 2016). It is therefore remarkable what level of accuracy is wanted and is acceptable. Taking more scans (or pictures for photogrammetry) can increase accuracy significantly, but it will also increase the data size, which is not practical. LOD measures the data density within the neighbourhood of each point goal in a point cloud (Zhang et al., 2016). The LOD has different levels, ranging from LOD 100 to 500. The LOD levels are applicable for different purposes. A simple geometry like a wall may not need dense data and thus have a low LOD, while complex shapes such as openings, edges and curves may need higher LOD level. When the level of LOD is not chosen wisely, it may lead to missing details (when chosen for a low level of LOD) or extra time and effort in data collection (when chosen for a high level of LOD). Zhang et al. (2016) explain that engineers should specify the LOD level for each point cloud data acquisition. This is so that they can collect 3D imageries containing all required geometric information while avoiding unnecessary dense data, which contributes to wasted time in data collection and processing.

2.9. Conclusion literature study

The construction industry is one of the major industries in the world and contributes to a big part of a country's gross domestic product. Because of the never-ending population size, technological change, changing lifestyles, and long-and short-term economic conditions, the demand for new buildings is growing. In addition, municipalities and clients are setting the standards high, leading to that projects are becoming much more complex and challenging. New (technological) inventions should be utilized to cope with these high standards, like lean management and agile project management.

Projects in the construction sector are generally characterized by time delays and cost overruns because of improper planning. That is why lean management is used in the form of construction planning. The construction planning is formulated by a construction planner, who makes a set of directions (also called activities) to the project team about what and when it must be done. Falling behind the devised planning is an unfavourable event that often occurs in construction projects, and therefore, the works are monitored to ensure that the overruns are minimal. This is done by site managers. However, monitoring of the works is identified as labour intensive, expensive, time-consuming, infrequent, prone to human errors, uncertain, inconsistent and inefficient.

The construction industry is very slow in adopting new technologies than other industries. However, new implementation in the form of agile management like Building Information Modelling (BIM) and monitoring systems arise. BIM is one of the game-changers in the construction industry, which has emerged decades ago, and it is now being used by most of the parties in the construction industry. With BIM, the way of designing, engineering, building and managing projects has led to the improvement of productivity, cost-efficiency and sustainability significantly, compared to the traditional way. BIM knowledge repository is in n-Dimensions, where BIM 4D is one of them. This is a method where the BIM-model is combined with construction planning. The emergence of BIM 4D has led to an efficient, safer and faster construction planning to compare with the traditional method. The visualisation and communication aspects provided by BIM 4D are hereby the most critical drivers. However, because construction projects are complex and dynamic during construction, updating the BIM 4D is discouraged, as it is labour intensive. By so far, BIM 4D-models are only made far after the construction planning and usually serves only for visualisation purposes.

Another type of technology that can be used in the construction industry is monitoring systems. Monitoring systems are technologies that are able to track construction activities such as imaging technologies, geospatial technologies and radiofrequency technologies. Monitoring systems generate point clouds. This is a set of points in the Euclidian space, registered in the X, Y and Z-coordinates, which provides a 3D model of the as-constructed building project.

The combination of point clouds and BIM 4D does exist in literature and provides a positive outlook for how BIM 4D-models should be updated. However, as of today, the proposed methods are in some way or other cumbersome, and most of the time, manual input is needed. This gap in literature may also be the reason that parties in the construction industry are not using this strategy to update the BIM 4D-model. A new method should be proposed in order to automate the whole process of updating the BIM 4D-model.

3. Methodology

In order to reach the goal of this research, a methodological framework is proposed, which are based on Kopsida et al. (2015) and van Schaijk (2016). Kopsida et al. (2015) divided the process of automated progress monitoring into four steps, which are: (1) data acquisition, (2) information retrieval, (3) progress estimation and (4) visualisation. The first step involves a technique where data is obtained, like sensing technology or photogrammetry. The second step is the processing and extracting of the obtained data into a point cloud model. The next step includes a comparison between the point cloud and the as-planned model. The last step is the visualisation of the results. On the other hand, van Schaijk (2016) proposed a framework inspired by the Plan-Do-Check-Act (PDCA) cycle. Van Schaijk (2016) explains that the PDCA cycle consists of four phases, starting with: (1) the plan phase, (2) the do and capture phase, (3) the check and analyse phase and (4) the act and reuse phase. In addition, every phase is described by the desired input and the desired output, where different tools can be used per phase to generate the desired output. It should be noted that this research does not concentrate on the act and reuse phase (phase 4), but rather on plan, capture and analyse (phase 1 up to 3). The PDCA cycle can be seen in Figure 6.

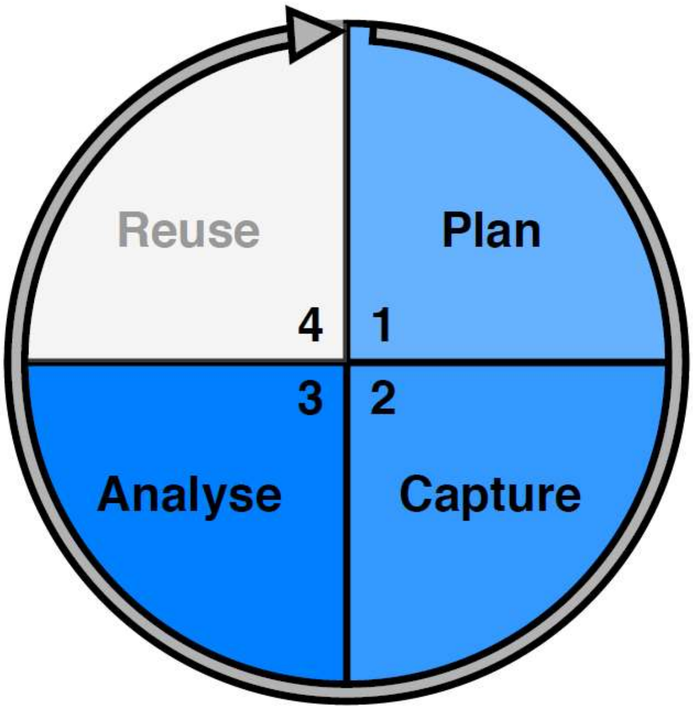


Figure 6: The PDCA cycle (van Schaijk, 2016) modified

Although the mentioned frameworks form a basis of this research, some other steps should be added and/or modified. The following paragraphs are structured conform to the PDCA cycle.

3.1. Plan

This phase consists of making a construction planning, followed by a BIM 4D-model. In order to create such a BIM 4D-model, it is necessary also to have an IFC model. The IFC model can be generated by different BIM tools and can contain plenty of data in the objects, such as GUID, geometry, cartesian points and colour. The construction planning and the IFC model should be, in this case, the desired output. The desired output for this phase is the as-planned IFC model. This is the result after every construction activity in the construction planning is assigned to a specific IFC object in the BIM 4D-tool. Besides this model contains the standard data from the standard IFC model, it also contains the time aspect. Hence, every IFC object contains extra parameters like activity name, activity start time, activity duration and activity end time. Different tools can be applied in order to generate the desired output and are already mentioned in Chapter 2.6.2. Figure 7 shows the desired in- and output of the plan phase.

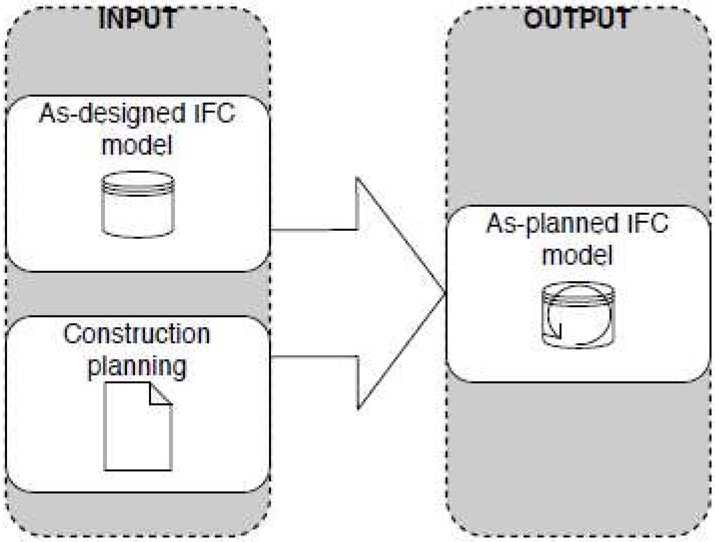


Figure 7: The desired in- and output of the plan phase

3.2. Capture

Monitoring the construction progress can be done via different technologies, including imaging, geospatial and radiofrequency technologies (Omar & Nehdi, 2016). However, capturing the construction site conditions can only be done via imaging technologies like photogrammetry, 3D laser scanning, videogrammetry or range images. These technologies have advantages and disadvantages, as described by Omar & Nehdi (2016). Although all mentioned technologies are able to generate point cloud models, as a desired input for the capture phase, only one technology is chosen in this research. Consideration should be made between the advantages and disadvantages of imaging technologies. In addition, the criteria affordability, portability, processing time, point cloud accuracy and range distance that are set by Omar & Nehdi (2016) should be weighed up against each other. For this study, it is considered that affordability and point cloud accuracy are the two most important factors by choosing imaging technology. This is because (1) construction firms are looking for solutions that are cheap to purchase and operate and (2) technologies that generate qualitative point clouds are denser, which are more reliable and accurate than less dense point clouds. Therefore, the criteria affordability and point cloud accuracy for every imaging technology are weight up against each other.

Range images and videogrammetry perform by means of affordability and point cloud accuracy medium to low. In addition, it is not allowed to obtain videogrammetry with a UAV in the Netherlands. Therefore, range images and videogrammetry are disregarded. Although photogrammetry and 3D laser scanning are different methods to capture construction site data, they both result in a point cloud data. A small remark can be made about the quality of the point clouds from both capturing methods. As already turned out from Omar & Nehdi (2016), photogrammetry is less accurate than 3D laser scanning. On the other hand, photogrammetry is more affordable than 3D laser scanners, which is a big factor that can encourage its use in construction projects. For this reason, photogrammetry is used as a capturing method.

A desired output of photogrammetry is a point cloud model, using the SfM method. In Chapter 2.7.2.1, an introduction about photogrammetry is already given. In the next section, an elaboration about this technique is given.

3.2.1. Photogrammetry in-depth

Photogrammetry, also called triangulation, is the technique to derive 3D data from (2D) photos. By photographing the object from at least two positions, a 3D-coordinate of the object can be determined. This is done by placing a spatial transformation between the pixels of the camera sensor and the object point. For this principle, a mathematical equation is formulated:

$[x_i, y_i, z_i] = R_i \cdot [X_i, Y_i, Z_i] + C_i$, where:

x_i, y_i, z_i describes the pixel coordination on the sensor, whereby z_i is the focal length. R_i means the rotation matrix of the camera, which describes the angle of the camera oriented to the object. X_i, Y_i, Z_i is the spatial coordination of the point. Finally, C_i is the position of the camera in a room.

As already indicated, photogrammetry uses the SfM technique to obtain point cloud data. This is done by overlapping images that are taken from multiple angles and viewpoints. A scale-invariant feature transform (SIFT) then identifies common feature points across the image set and makes relationships between them, resulting in a 3D coordination system (Micheletti et al., 2015). Also, a sparse bundle adjustment is needed in order to transform the measured image coordinates into 3D points. The result of the sparse bundle adjustment is a dataset of 3D locations of the feature point of the object, which is called a sparse point cloud. Micheletti et al. (2015) say that accurate feature point correspondence requires the availability of visually distinct texture appearing in the imagery, which can present a problem with some objects and/or lighting conditions. The sparse point cloud can then be intensified using the Multi View Stereo (MVS) technique. This is a technique that generates a high resolution of datasets, resulting in a dense point cloud. In addition to that, gross errors are isolated or even being removed. After this step, it is possible to mesh the point cloud. This is a simple triangular surface patch that meets at their shared point cloud edges (Yang et al., 2010; Park & Lee, 2019), resulting in a 3D mesh model.

Figure 8 shows an example of photogrammetry data acquisition, where multiple images with large overlap are collected from different angles, directions and positions.

In essence, when working with photogrammetry, the quality of the data is really important. This is dependent on the number of images taken: the more images were taken, the more overlap of the images to cover the full geometry of the object. An indication is that around 100 images should be a good starting point. Moreover, large datasets can help to remove outliers when the number of images is already sufficient enough for a good point cloud model (Micheletti et al., 2014).

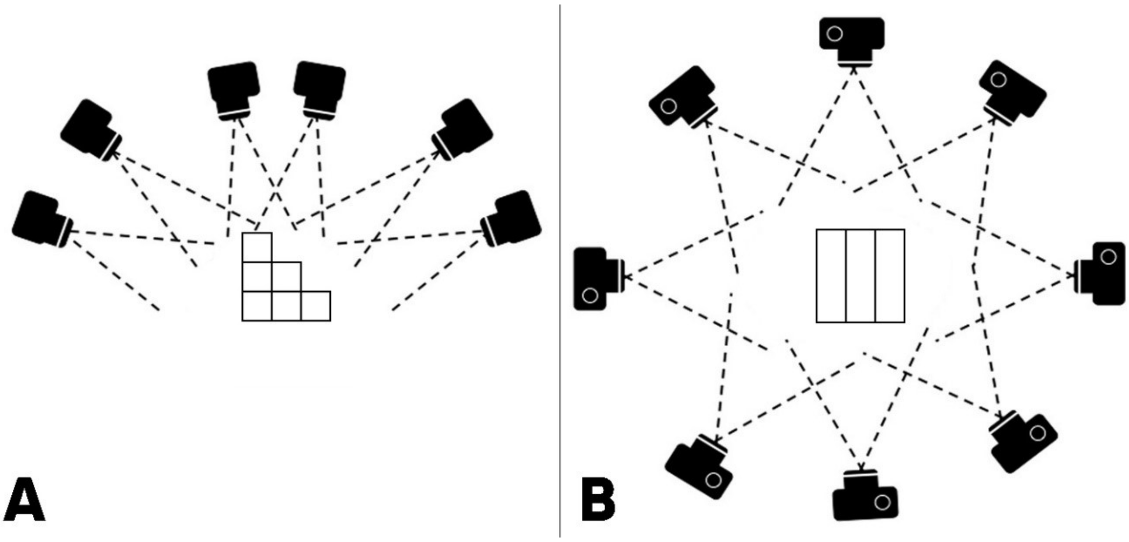


Figure 8: Data acquisition of photogrammetry, where A is a side view, and B is a top view of an object

Points of attention by using photogrammetry

Micheletti et al. (2015) state that images do not need to be acquired from the same distance or have the same scale. On the contrary, it is advised to acquire multiple pictures from a distance to capture the object in a wide view, before capturing it closer. The wide view picture will help the SfM technique to set the boundaries of the space, and the close images will help to obtain the desired detail at the required precision. Capturing the object in detail is also important where areas should not be physically obscured by other features (occlusions), and the scene should be captured static. Other points of attention are that: photographing objects with flash can create inconsistent image textures and confuses the SfM process; using consistent lighting in order to avoid overexposing and underexposing images; keep the camera stable with a high shutter speed to avoid blurred images and; avoid transparent or reflective surfaces.

Coordination system, scaling and positioning

Photogrammetric mesh models are inherently dimensionless. This means that models are neither scaled, nor aligned to local gravity, and is therefore unrelated to an established external coordinate system (Micheletti et al., 2015). When using photogrammetry for only visualisation purposes, this is not a big issue. However, if the photogrammetric model needs to be computed in real-world coordinates, a transformation of the model is necessary. This includes changing rotation, translation and scaling parameters of the generated model. In order to know what these parameters are, a reference point should be added in the scene during data acquisition. In many practises a target or a scale bar are used as a reference point. In this way, there is a known distance, and the parameters of the model can be changed automatically in some software. However, manual handling always remains universal and can be used in multiple situations. In addition, targets should be clearly visible in the images, and a minimal of five should be used in the scene in order to improve the quality of the model transformation.

Software

Different software packages are available, enabling users to compute photogrammetry with their computer or via a server of a company. Computing photogrammetry via a personal computer is most of the time cheaper but requires a lot of computing power (and thus time) to compare with a server. A server is provided by a company who offer their services to compute the SfM technique. The images are uploaded to the server by the user, and the point cloud model can be downloaded after the server processed the images. This method is faster than computing photogrammetry via a single computer, but the providers of the server demand subscription costs for using their services.

Photogrammetry software also varies in their characteristics and options. One is only applicable for aerial data acquisition and the other for both aerial and terrestrial data acquisition. The results of the data can be delivered to the user in a variety of file formats. The most common ones are hereby .las, .ply, .obj and .xyz. The more variety of file formats the software can export, the more possibilities for the user to conduct further analysis of the point cloud and use in other software.

All well-known photogrammetry software can be operated on Windows computers, and some of them are open-source and freely available. Figure 9 shows the desired in- and output of the capture phase.

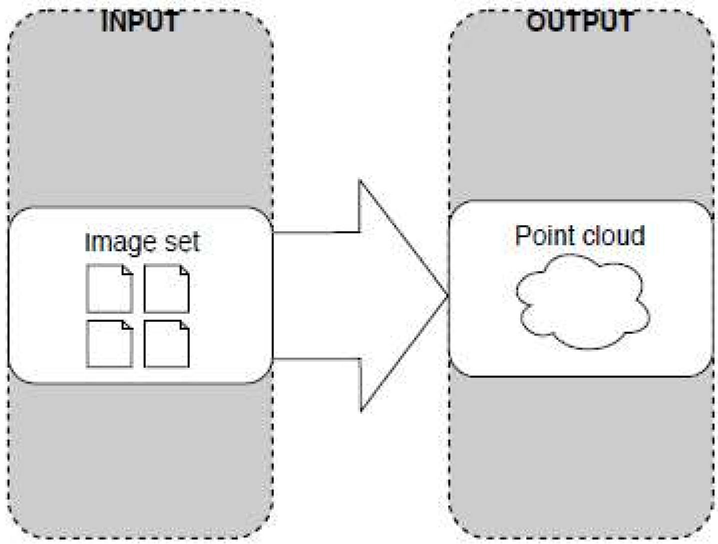


Figure 9: The desired in- and output of the capture phase

3.3. Analyse

The last phase is the analyse phase. Within this phase, the desired outputs of the plan and capture phase are analysed. These outputs form at the same time, the desired input for this phase. The desired output is a format where information of the construction deviation is stored which can be imported in the BIM 4D-tool.

A way of combining the as-designed IFC model with the point cloud model is by means of tooling. This is a technique that can bridge the gap between IT and construction management. In order to develop a tool, natural language processing (NLP) is used. This is an interdisciplinary topic which involves in computational linguistics, AI, and computer science that deals with the interactions between computer and human languages (Zou et al., 2017; Zhang et al., 2019). Multiple programming languages exist for this. Prolog, C++, Java and Python are the most common ones. These languages have constraint libraries and are defined over finite sets. A constraint solver can hereby check whether a constraint or a set of constraints, can be satisfied (Camacho et al., 2018).

3.3.1. Python

Python is going to be used for developing the tool. This is because it has the ability to quickly modify numerical data and simulate machine operations through widely available simulation and graphical packages (Malik et al., 2019), like NumPy and Matplotlib. In addition, Python has an open environment structure, which allows to program fast and powerful, because of its ease of reading, processing, and manipulating information.

Datatypes

A python script works with data and can be stored in variables with certain datatypes. The following data types (or functions) are commonly used in python:

- Integer (int): integer value without floating-point;
- Float (float): floating-point number;
- String (str): a sequence of characters;
- List (list): list of datatypes;
- Boolean (bool): binary value (True or False);

Variables

Values in a certain datatype can be stored in variables. These variables need to have a well-defined name so it can be understood by users. An example can be given with a string in a variable:

```
In [1]: msg = "Hello World!"

In [2]: print (msg)

Out []: Hello World!
```

There are also points of attention when given names to variables:

- It can only be one word.
- It can use only letters, numbers, and the underscore (_) character.
- It cannot begin with a number.
- Variable names starting with the underscore (_) character are considered as not useful.

Operators

Operators allow users to perform operations on variables of a particular data type. Different operators are available. A summation of two variables can be given as an example:

```
In [1]: x = 2

In [2]: y = 3

In [3]: print (x + y)

Out []: 5
```

Another one is the selection of characters in a string (note that python counts from 0):

```
In [1]: msg = "Hello World!"

In [2]: print (msg[0:5])

Out []: Hello
```

Loops

Loops are used for iterating over a sequence. There are three different types of loops, namely:

- If-loop: “If this is true, do that”;
- While-loop: “While this is true, do that”;
- For-loop: “For this entire sequence, do that”.

An example is given with an if-then-else loop, which states that if this is true, do this; else, check if this is true, and do that; else do this:

```
In [1]: x = 6

In [2]: y = 2

In [3]: if x > y:

            print ("x is greater than y")

        elif x == y:

            print ("x is equal to y")

        else:

            print ("y is greater than x")

Out []: x is greater than y
```

Code editors

Programming with python starts with an editor. An editor is an environment where python scripts are developed and executed. Examples of code editors are Microsoft Visual Code (VSCode), Atom and Sublime. There are also editors that are more advanced and need more programming skills than regular editors. These are called Integrated Development Environment (IDE). These include auto-completion, debugging and testing. Examples of IDEs are PyCharm and Spyder.

3.3.2. Alignment of point cloud and IFC data

In order to calculate what the deviations are, the difference between the point cloud and IFC data should be measured. There are different points of attention to be considered, like scaling, overlay and calculating deviations.

Scaling

As already mentioned, point cloud models that are generated by photogrammetry are dimensionless. IFC models, however, have fixed measurements, and this should be used as a standard to compare with the point cloud model. This means that the point cloud model should be scaled as the exact measurements of the IFC model. In order to scale the model, a known distance should be used like target points or scale bar. Many tools are available to scale point cloud models. The basic workflow is as follows: the user imports the point cloud model and draws a line from the beginning point to the endpoint of a target. After that, the user enters the known (real) distance, and the tool scales the model to the right measurement.

Overlay

Point cloud data consist of multiple dots, and the IFC data consist of surfaces. Comparing these with each other is not easy, and therefore a solid model of the point cloud data should be generated. There are three dominant methods to generate solid models directly from point cloud data: (1) Constructive Solid Geometry (CSG), where objects are represented using Boolean combinations of simpler objects; (2) Boundary Representations (B-Reps), where object surfaces are represented either explicitly or implicitly, and (3) spatial sub-division representations, where an object domain is decomposed into cells with simple topological and geometric structure, such as regular grids and octrees (Hinks et al., 2013). The last one is extensively used in many research, because it can be efficiently stored as they have very little in-memory data, allowing for a fast 3D visualisation. Grids and octrees can, therefore, be used for speeding up the point cloud segmentation process (Pörteners, 2018). This method is ideal for this research.

Grids or octrees that are employed to decompose an entire object into non-overlapping 3D regions are referred to as voxels. Voxels are equivalent to pixels, but then in 3D space, divided into uniform 3D cells like cubes. It can be seen as a subdivision of smaller cubes in a grid where the planes are parallel to the Cartesian coordinate system axes. The location of a voxel is not explicitly stored as a set of XYZ coordinates, but rather is determined by its relative position to other voxels and the coordinate origin of the data set (Guarato et al., 2017; Nourian et al., 2016b; Gebhardt et al., n.d.).

The concept of voxels is displayed in Figure 10. It can be seen that voxels may have different depths. In each depth, an initial voxel is divided into eight sub-voxels, which is why the name octrees (octa means eight in Latin) is given. Sub-voxels are also called nodes. For point clouds, these nodes contain a specified number of data points, and as such, dense point cloud chunks contain many small nodes (Pörteners, 2018). The size of each node is dependent on the input of the user, which is defined before voxelization, where three pairs of coordinate values ($[x_{min}, x_{max}]$, $[y_{min}, y_{max}]$, $[z_{min}, z_{max}]$) are created along the three axes (X, Y and Z). Voxels may be labelled white, black, or grey based on their positions. White voxels are outside the solid and black voxels are inside the solid. Gray voxels have both black and white children and are thus subdivided into an octree (Hinks et al., 2013).

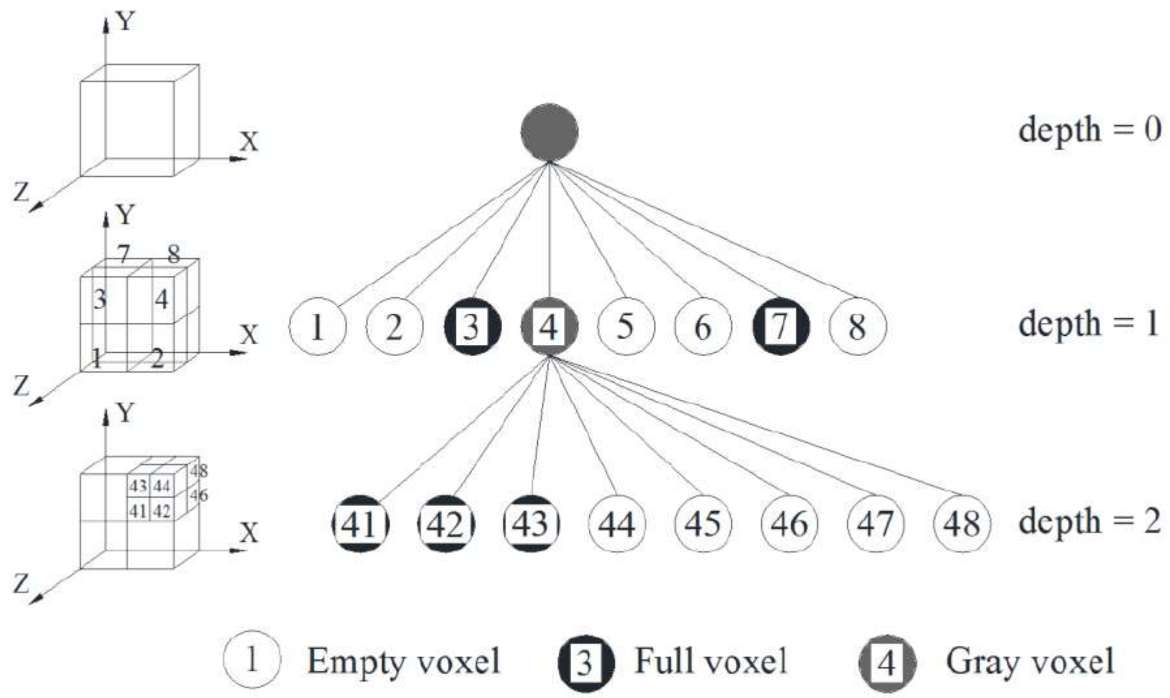


Figure 10: The concept of voxels (Hinks et al., 2013)

A voxel in point cloud data are divided into subset voxels by grids along x-, y-, and z- coordinates in a Cartesian coordinate system. Each voxel in the subset is represented by an index $v(i, j, k)$, where $i \in [0; N_x - 1]$, $j \in [0; N_y - 1]$ and $k \in [0; N_z - 1]$. This equation is also visualized in Figure 11. With the size of individual voxels ($\Delta x, \Delta y, \Delta z$), a number of voxels ($N_x, N_y, N_z$) in each direction are given in Equations 1-3 (Hinks et al., 2013):

$$N_x = \frac{(x_{max} - x_{min})}{\Delta x} + 1 \qquad (1)$$

$$N_y = \frac{(y_{max} - y_{min})}{\Delta y} + 1 \qquad (2)$$

$$N_z = \frac{(z_{max} - z_{min})}{\Delta z} + 1 \qquad (3)$$

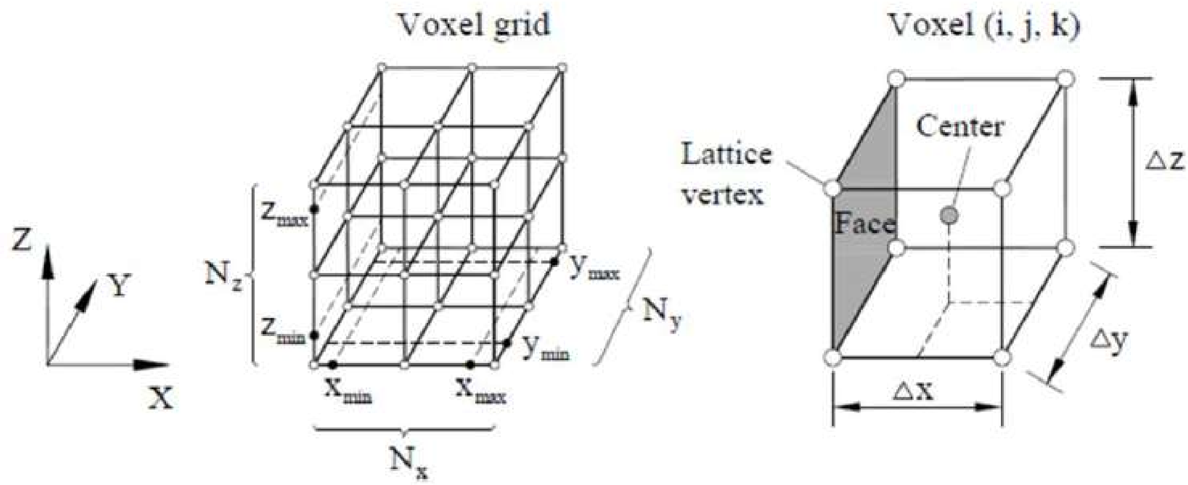


Figure 11: A voxel grid of a point cloud bounded by (xmin, xmax), (ymin, ymax), and (zmin, zmax) (Hinks et al., 2013)

Besides generating voxels of point cloud data, it is also possible to do this with 3D objects from IFC data, for example. This principle of doing this is different than mentioned before. As already mentioned, a voxel can be defined by a 3D pixel where it has equals lengths like a cubic shape. In the context of generating voxels of 3D objects, it can be constructed based on the external geometry of it. Each voxel can hereby be associated with a group of digitized points which represents the face of the part. Besides, black voxels can be indexed as $\beta = 1$, which are voxels inside the solid and white voxels can be indexed as $\beta = 0$, which are voxels outside the solid. In Figure 12 is shown how 3D objects are voxelized.

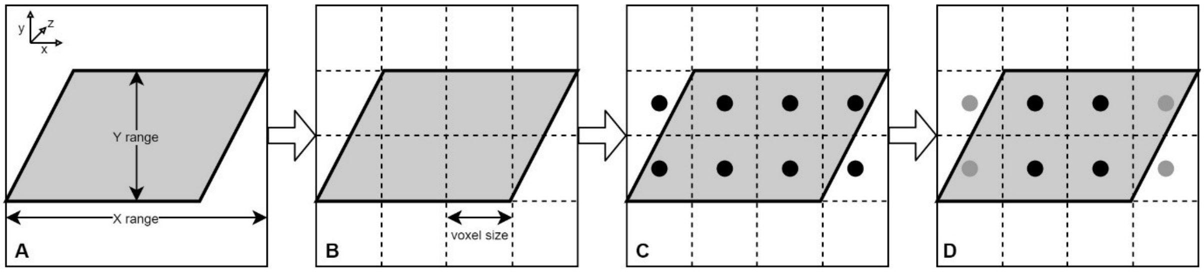


Figure 12: Voxelization of a part section (A-B) and determination of contained voxels (C-D) (Guarato et al., 2017) modified

A 3D structure of voxels is also called a voxelmap, which detects the voxels that are inside a 3D solid object. A voxelmap also consists of a coordination system, where the edges of the voxels are parallel to the axes of the coordination system (Guarato et al., 2017). In order to compute a voxelmap, the object should be split into planar objects. After that, it can be rasterized with cell size r for further processing. Tuttas et al. (2014) defined two different states for each raster cell. State A is based on visibility constraints calculated using an octree with voxel size o, state B is based on the points within a bounding box. State A is used for deriving construction parts for inside, and outside its own boundary, when points are detected inside the visibility constraints, the octree cell is then occupied. Since the octree cell size is too large, an occupied octree cell does not prove that the building part at this position is really existing. Because of this, state B can be used. This determines if a point is inside the actual boundary. Figure 13 shows a schematic representation of the difference between state A and state B.

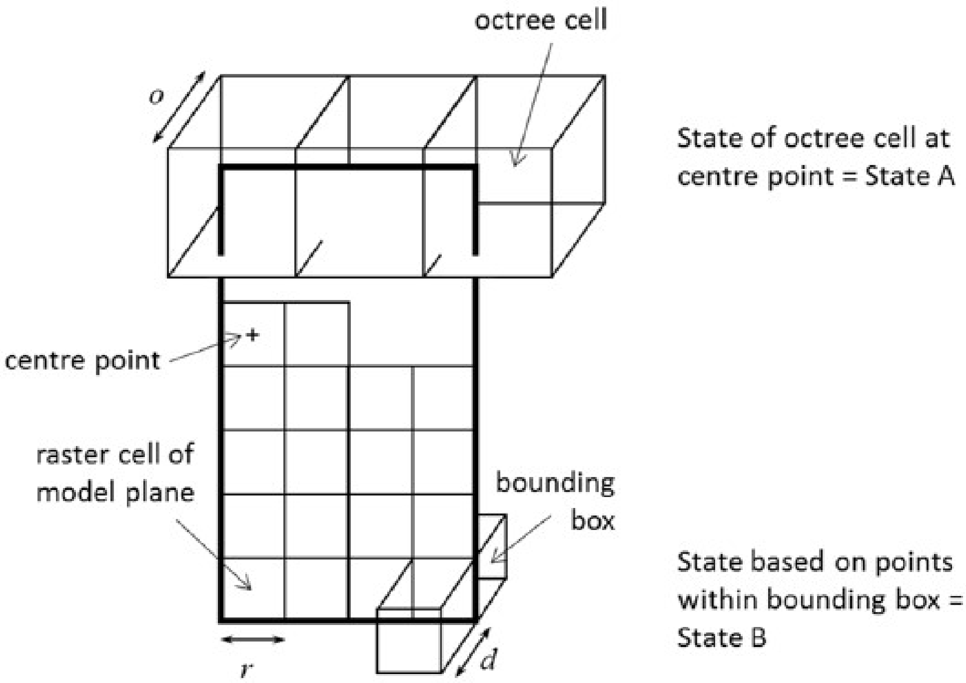


Figure 13: Schematic depiction of a model plan with octree cells (used for determination of state A) and raster cells (used for determination of state B) (Tuttas et al., 2014)

3.3.3. Calculating deviations

After the point cloud data is scaled and overlaid with IFC data, it is time for calculating deviations. Turkan et al. (2012) already proposed an algorithm to calculate the difference between the actual construction progress and scheduled progress. Although their algorithm is based on 3D object recognition, it forms a good starting point for this research.

As already indicated, the point cloud and IFC data can be voxelized to compare with each other. The IFC data can hereby have a grid structure in order to create a bounding box as state B. After overlaying point cloud voxels, sometimes these voxels can be contained inside the bounding box. In this way, there is a great chance that the actual wall, for example, is constructed within the specified time conform the construction planning. It can also happen that the point cloud data contains errors and leads to an incorrect calculation of the output. In order to avoid this, each bounding box should be constrained. This means that for every bounding box, a minimum amount of point cloud voxels should be contained. If this is the case, the algorithm should give a value of 1 (true) for every contained point cloud voxel inside a bounding box. If not, it should give a value of 0 (false). To finally calculate if the exampled wall is constructed, the number of true bounding boxes should be divided by the total amount of bounding boxes of the wall times 100. The output will give a percentage of the wall constructed. The algorithm can be formulated as follows:

$$P_i = \frac{\sum Voxels_{c,i}}{Voxels_{a,i}} \cdot 100\%$$

$$Voxels_{c,i} \geq T = 1$$

$$Voxels_{c,i} < T = 0$$

Where P_i is the progress estimated of a certain construction element, $Voxels_{c,i}$ is the contained bounding box, $Voxels_{a,i}$ is the total number of bounding boxes of the construction element and T is the threshold.

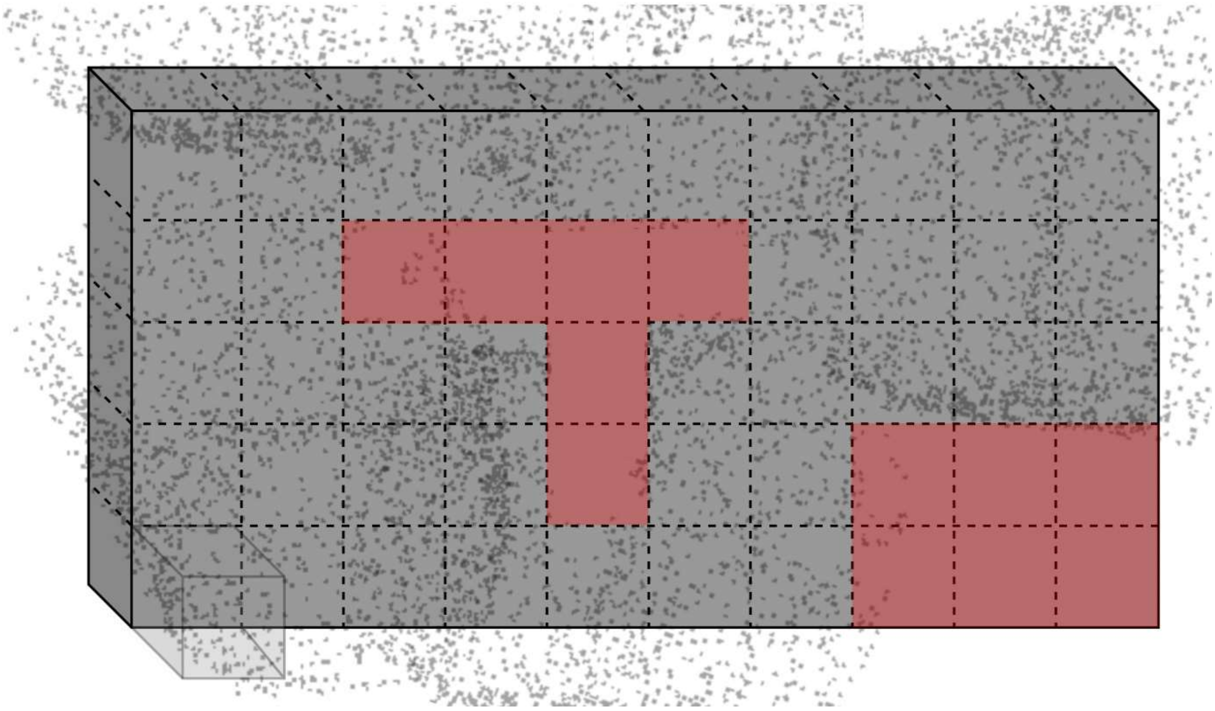


Figure 14: An example with a simple wall

An example can be illustrated in Figure 14. This is a representation of state B, where the face of a simple wall is rasterized into 10x5 grid lines. On the left bottom corner, a bounding box is created, which has a distance equal to the thickness of the wall. This bounding box applies to every raster cell, but for visualisation purposes, only one bounding box is created. In addition, a point cloud is created in and around the wall. It can be seen that for some bounding boxes, the points are dense. Some have little to none points within the bounding box. If the threshold T is set to 30 points, the contained bounding box $Voxels_{c,i}$ for one face can be calculated.

The total number of raster cells is 50. Each raster cell where the bounding box is contained with points that are equal or greater than 30, it gets a value of one. If it is under this threshold, it gets a value of zero. It is determined that 38 cells are contained, and 12 cells are not (see red area). The percentage can, in this way, be calculated:

$$P_i = \frac{38}{50} \cdot 100\% = 76\%$$

The result gives an indication that the face has a probability of 76% that it is existing.

3.3.4. Import data to BIM 4D-model

Feedback should be given to the BIM 4D-tool, where the data contains information about which 3D object has deviated from the actual construction planning. This can be done via CSV, which stands for comma-separated values. CSV is a file that can be generated by the developed tool and can be directly imported to a BIM 4D-tool.

To finally conclude this paragraph, Figure 15 shows the desired in- and output of the capture phase.

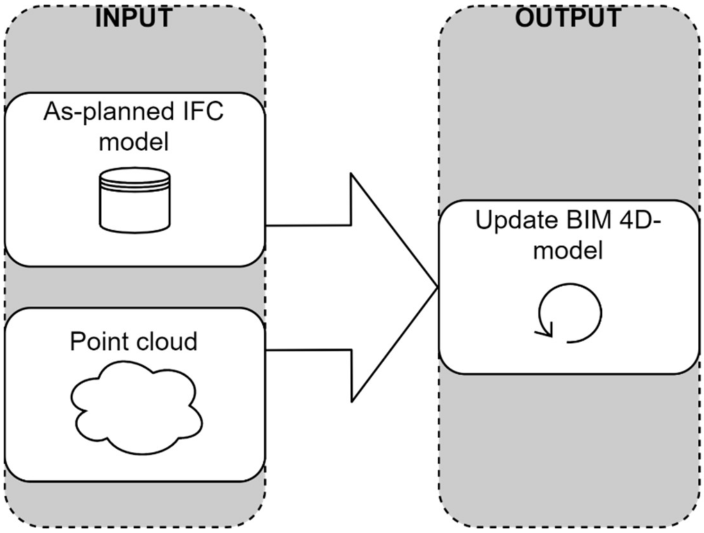


Figure 15: The desired in- and output of the analyse phase

3.4. Methodological framework

The full methodological framework of this research can be seen in Figure 16. The framework shows how the plan, capture and analyse phase come together. Each phase generates an output which is an input for the other phase. The in- and outputs for each phase are already described earlier. The whole framework can be seen as an iterative process, where each component can be systematically repeated. This means that it is possible, in any moment of time, to measure the construction deviation(s) automatically.

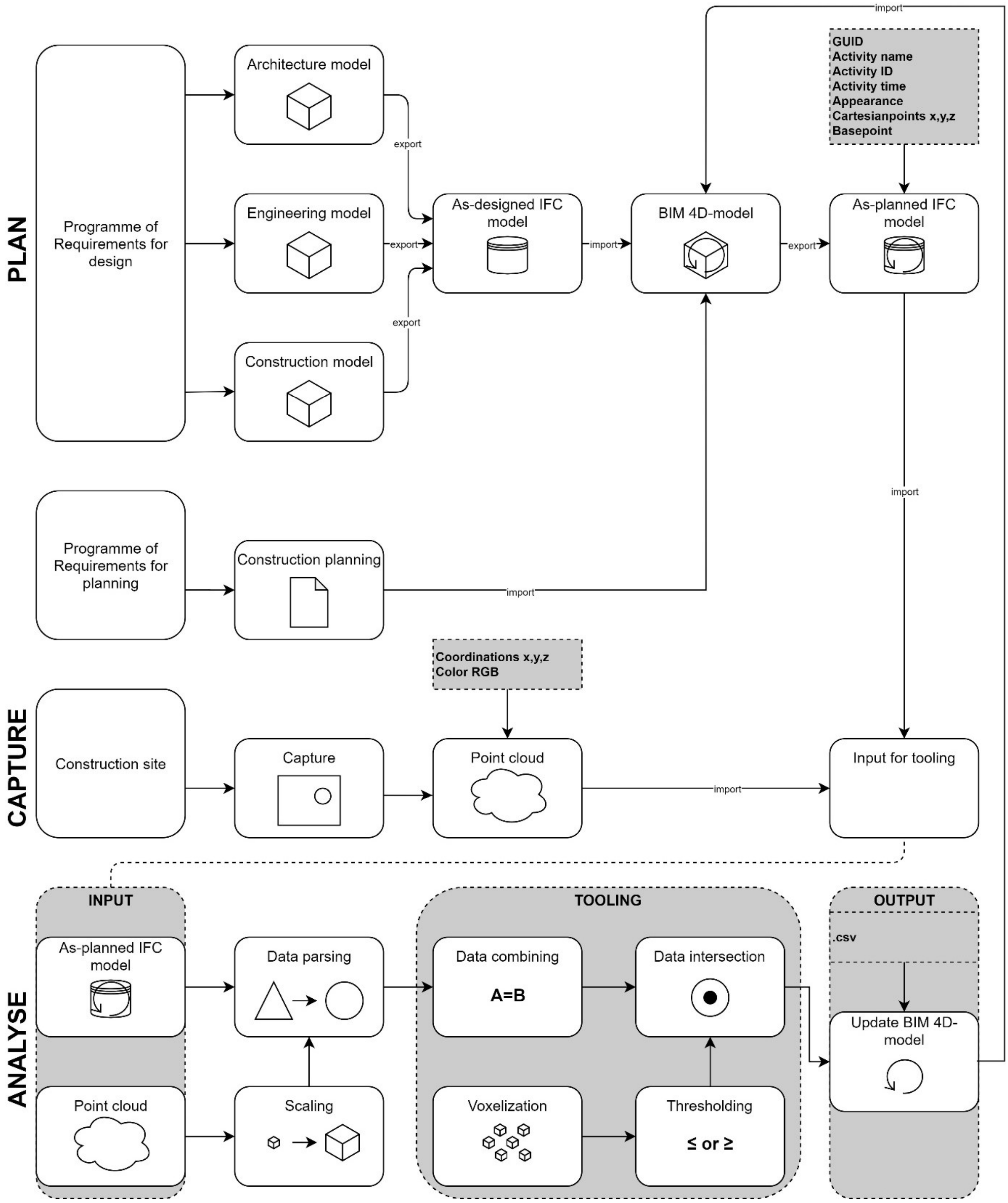


Figure 16: The full methodological framework

3.5. Conclusion methodology

This chapter described in detail how the research will be conducted. The basic framework of this research is derived from Kopsida et al. (2015) and van Schaijk (2016). Both authors proposed a framework, which uses sequential phases in order to meet their goal. The phases that are used in this research are: plan, capture and analyse phase. Each phase has a required in- and output. For the plan phase, it is required to have a 3D model, which must be exported to an IFC format. At the same time, construction planning is required. The IFC and planning data are needed to make a BIM 4D-model. This is also the required output of the plan phase. For the capture phase, point cloud data is needed. It is concluded that the photogrammetry technique will be used to acquire the point cloud data. The reason to use this technique is based on the advantages and disadvantages of imaging technologies, described by Omar & Nehdi (2016). The required input for this phase is a set of pictures of the project. The photogrammetry technique will use this set of pictures to generate a point cloud data, which is also the required output of the capture phase. The analyse phase is the last phase, where the plan and capture phase come together. The required input for this phase is the required output of the plan and capture phase. Within this phase, a lot of information should be processed. This is done in three steps. The first step is about that the point cloud data should be scaled in order to align with the IFC model. Step two is about calculating the intersection points of the point cloud with the IFC model. Step three exports the information derived from step two as a CSV file, which can be imported in the BIM 4D-tool. This is at the same time the required output of the analyse phase. All steps are hereby executed in Python language.

The PCDA cycle also showed a fourth phase. In this research, this phase will not be used.

4. Preparation for case study

Before testing the proposed methodology in a real case, a simple fictive project is first initiated. The reason for this is to get familiar with the different tools and handlings needed in order to meet the research goal. This simple fictive project is about a rectangular house. At such a time during the construction, the construction planner wants to know if there are any deviations based on construction planning. Therefore, the progress is being monitored by an imaging technology and analysed with the proposed tool. In addition, chapter 2.6.3 already described the different levels of BIM 4D-uses. This simple fictive project will be considered on a macro level. The following paragraphs show step by step how the whole process is executed.

4.1. Plan

<u>BIM-model preparation</u>

The rectangular building is designed in the BIM tool Revit. The boundaries of the project are determined to be 30x22,5 meters. The grid of this project consists of five vertical and four horizontal gridlines. This means that the distance between the vertical and horizontal gridline is 7,5 meters. On the outer gridlines, the shell of the building is modelled by means of structural walls. On top of the walls, a roof is placed. The roof is segmented into multiple components. In addition, all structural components have the dimension 7500x1500x2500 millimetres. Figure 17 shows the design of the fictive project. The walls have a grey colour, and the roof has a green colour.

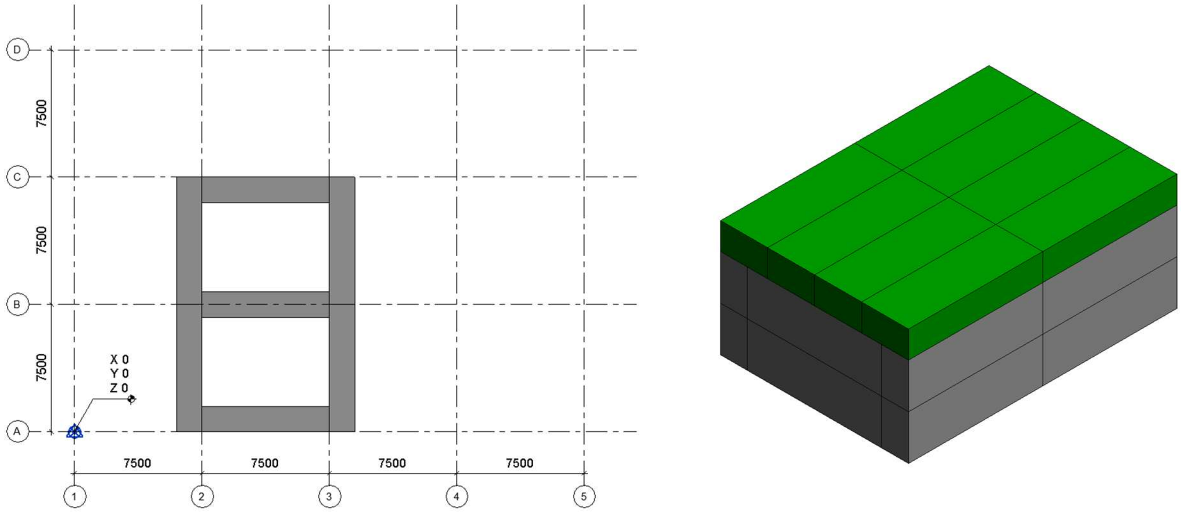


Figure 17: The floor plan of the fictive project (left) and a 3D model of the fictive project (right)

It is important to define the base point of the 3D model. This is the project origin, which is determined in coordinates. The base point (0, 0, 0) of the fictive project is determined at the grid intersection of A and 1.

After creating the 3D model, it is exported to an IFC format. This format can be imported into a BIM 4D tool. The GUIDs (see chapter 2.5.2) shown in the IFC file is important for the next paragraphs.

With the Python package IfcOpenShell, it is possible to scan the IFC data and see which GUID is attached to which 3D object. The following code imports the IfcOpenShell package, opens the IFC data from the computer drive, and gives a list of the GUIDs of the wall and slab objects:

```
In [1]: import ifcopenshell

In [2]: ifc_file = ifcopenshell.open("data/3Dmodel/Building.ifc")

In [3]: for wall in ifc_file.by_type("IfcWall"):

            print ("wall with global id: "+str(wall.GlobalId))

        for slab in ifc_file.by_type("IfcSlab"):

            print ("slab with global id: "+str(slab.GlobalId))

Out []:

wall with global id: 2UZmjjyaz2TgXXHAB9mMrK
wall with global id: 2X$MCHRGHEdwZBdEYerMDS
wall with global id: 2X$MCHRGHEdwZBdEYerMDz
wall with global id: 2X$MCHRGHEdwZBdEYerM3K
wall with global id: 2X$MCHRGHEdwZBdEYerM01
wall with global id: 2X$MCHRGHEdwZBdEYerM0W
wall with global id: 2X$MCHRGHEdwZBdEYerM0i
wall with global id: 2X$MCHRGHEdwZBdEYerM1b
wall with global id: 2X$MCHRGHEdwZBdEYerM6H
wall with global id: 2X$MCHRGHEdwZBdEYerM6T
wall with global id: 2X$MCHRGHEdwZBdEYerM69
wall with global id: 2X$MCHRGHEdwZBdEYerM6r
wall with global id: 2X$MCHRGHEdwZBdEYerM6X
wall with global id: 2X$MCHRGHEdwZBdEYerM6j
slab with global id: 39hf$m_iX1HvnfWRQmpotl
slab with global id: 39hf$m_iX1HvnfWRQmpouV
slab with global id: 3XYVoqAYvAp9L0gWFfgEUl
slab with global id: 0qYTB_GR59PgkUUxFR_isB
slab with global id: 0qYTB_GR59PgkUUxFR_irl
slab with global id: 0qYTB_GR59PgkUUxFR_irR
slab with global id: 0qYTB_GR59PgkUUxFR_irN
slab with global id: 0qYTB_GR59PgkUUxFR_ir3
```

It can be seen in the output that there are 14 wall objects and 8 floor objects. Also, it can be noted that all 3D objects have a unique ID.

BIM 4D preparation

For the BIM 4D-planning, a construction planning is first made. The planning consists of three main construction activities, which are: “Wall – Base”, “Wall – Top”, and “Roof”. Within these main construction activities, sub-activities are added, which is also called “children”. The first two main construction activities have seven children, and the third main activity has eight children. All children activities have a duration of one day, which is, in this case, a working day of eight hours. Also, all activities are placed in subsequent order and have a finish-to-start relationship. This means that all activities are in the critical path. When at least one activity is behind schedule, it will lead to late project delivery.

After the construction planning is ready, a BIM 4D-model can be made. This will happen with the BIM 4D-tool Synchro. With this tool, it is possible to import the IFC model and import the construction plan. It is also possible to make the construction planning directly in the BIM 4D-tool, which is here the case. After all 3D objects are assigned to designated construction activity, an animation can be displayed. This animation shows on which date a specific construction activity should be conducted. In this way, it is possible to stop the animation in any moment of time and see which construction activity should be executed at that specific date, see Figure 18, Figure 19, and Figure 20.

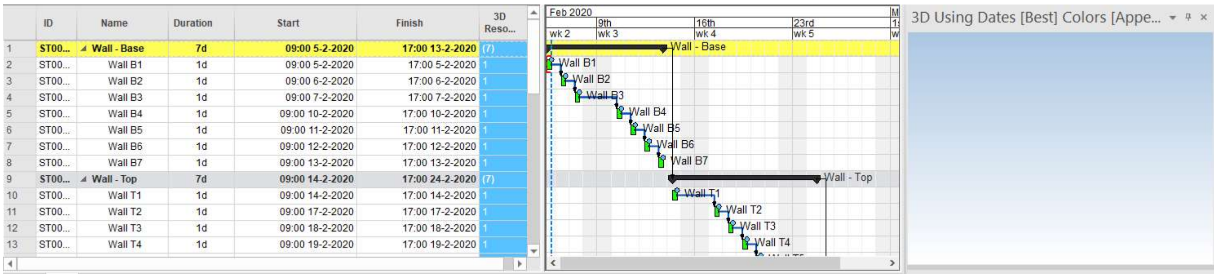


Figure 18: Before the start of the project planning

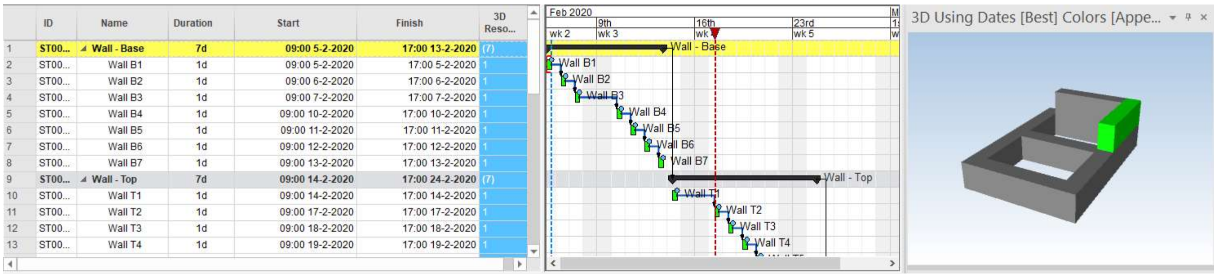


Figure 19: Halfway through the project planning

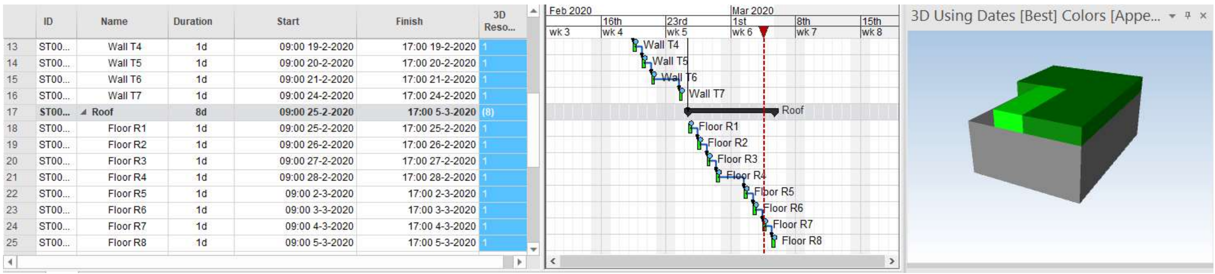


Figure 20: The planning reaching its end

It is possible to export the BIM 4D-model to IFC. Synchro will add extra parameters to the file, like IFCtask, IFCscheduletimecontrol, IFCdateandtime, IFCcalendardate, and IFCrelassignstoprocess. IFCtask is a parameter that shows the name of the activity/task with its GUID. IFCscheduletimecontrol includes the begin, intermediate, and end date of the task. IFCdateandtime includes the IFCcalendardate. IFCcalendardate shows the begin, intermediate, or end dates. In order to visualize how Synchro assigns 3D objects with tasks, a list can be created. This list is generated with the Python package IfcOpenShell.

```
In [1]: import ifcopenshell

In [2]: model = ifcopenshell.open(r"data/4Dmodel/Building4D.ifc")

In [3]: field_width = 25

In [4]: print ("{}\t{}\t{}".format("GUID".ljust(field_width),
"Object_ID".ljust(field_width), "Task_Name"))

In [5]: for object in model.by_type("IFCRELASSIGNSTOPROCESS"):

    try:

        Object_ID = wall[4][0][2]

        GUID = wall[4][0][0]

        Task_Name = wall[6][2]
```

```
        print ("{}\t{}\t{}".format
(GUID.ljust(field_width),Object_ID.ljust(field_width), Task_Name))

    except:

        pass
```

```
Out []:

GUID                        Object_ID                         Task_Name
11WEL5bJ5BrRaBTyUDt59r      Walls 2:Walls 1:347628_(#310)     Wall B1
27LLqJ_T5CoxqrLeNZkksE      Walls 6:Walls 1:347825_(#493)     Wall B2
0yFbWYX9P2khHEPvdqIn$Y      Walls 7:Walls 1:347837_(#554)     Wall B3
2uBfCwIOz3UhsN6_AMMVyZ      Walls 3:Walls 1:346524_(#161)     Wall B4
3BLxyytjn5vw6DT5M08mgk      Walls 5:Walls 1:347792_(#432)     Wall B5
0asFhNjRTBNPa4KHmeHCtZ      Walls 4:Walls 1:347717_(#371)     Wall B6
3A8jbHwwDDwwR5GjIy6FeW      Walls 1:Walls 1:347597_(#249)     Wall B7
174hSFmtb7aufL678BYOzF      Walls 10:Walls 1:347916_(#737)    Wall T1
01f6B0pe10HujK8erRe8hB      Walls 13:Walls 1:347952_(#920)    Wall T2
2hurcXpOr9KwUr00YmJ$qy      Walls 14:Walls 1:347964_(#981)    Wall T3
2ysZcvGnj0SP2fw6$fYZ0U      Walls 8:Walls 1:347892_(#615)     Wall T4
1qL5FuTI17hwpoN3NY3MfL      Walls 12:Walls 1:347940_(#859)    Wall T5
0Jfb2SQ9HCAfxisKfwYAxl      Walls 11:Walls 1:347928_(#798)    Wall T6
2g5zs89f5FKgDmj4a2$B$L      Walls 9:Walls 1:347904_(#676)     Wall T7
0SbB11xRv1igTCVZuLSEwP      Roofs 3:Roofs 1:348430:1_(#1060)  Floor R1
3d2EX_ubT06P1kOHoEMgA5      Roofs 1:Roofs 1:348926:1_(#1131)  Floor R2
2o5twiFFv7ggasQ$4xRUVt      Roofs 2:Roofs 1:348966:1_(#1199)  Floor R3
3m8397_JH6g8xmABENAFGG      Roofs 4:Roofs 1:349042:1_(#1267)  Floor R4
3138C65eb9BPn1ayYeATkp      Roofs 5:Roofs 1:349078:1_(#1335)  Floor R5
3SkySXbrzDfA3Fzom3iW8Z      Roofs 6:Roofs 1:349090:1_(#1403)  Floor R6
0HD2NOx8L7vA80TsM76SLQ      Roofs 7:Roofs 1:349102:1_(#1471)  Floor R7
0YY4LqofPDs8ZCPnv_DMNu      Roofs 8:Roofs 1:349114:1_(#1539)  Floor R8
```

The code first imports the python package IfcOpenShell; opens the IFC file; makes a spacing of the list; makes the headers ‘GUID’, ‘Object_ID’, and ‘Task_Name’; searches the lines IFCrelassignstoprocess of the IFC; searches what the GUIDs are for each 3D object, with their Object ID’s, and task names; and prints the results.

The result of the code shows that Synchro makes new GUIDs of each 3D objects. This means that object association (will be handled in detail in the Analyse phase) need to be done via task names or object ID’s instead of GUIDs.

4.2. Capture

In order to capture the construction progress, a 1:100 scale model is created. The scale model is a model of the simple fictive project. To reconstruct a situation where the planner wants to measure the deviations between the construction site and the construction plan, some building blocks will be removed. This creates a perception where the construction of the project is still in progress.

In order to monitor the construction progress, pictures are taken from different angles. The pictures are generated by the smartphone Samsung Galaxy S8. The specifications of the camera of this smartphone are shown in Table 2:

Table 2: Specifications of the camera

Number of Megapixels	12
Pixel size (in μm)	1.4
Sensor size (in inch)	1/2.55
Aperture	F1.7
Field of view (in degrees)	77

As previously mentioned, photogrammetric models are dimensionless. This means that X, Y, Z distances have no meanings. Nevertheless this issue, the ratio between X, Y, Z distances are the same. In this way, it is possible to scale the model to the right dimension. To make the scaling right, the photogrammetric model should contain a known distance. For this reason, some target points are placed in the area where the photos will be taken. The blue and pink target points have a relative distance of 22.5 cm, and the green and yellow target points have a relative distance of 30 cm. The height of all target points is 7.5 cm. Also, a ruler is placed under the first horizontal gridline. The length of this ruler is 30 cm. Figure 21 shows the setup for generating a photogrammetric model of the scale model of the simple fictive project.

After placing the target points, the scale model is created. Figure 22 shows the scale model, where the floors are not finished.

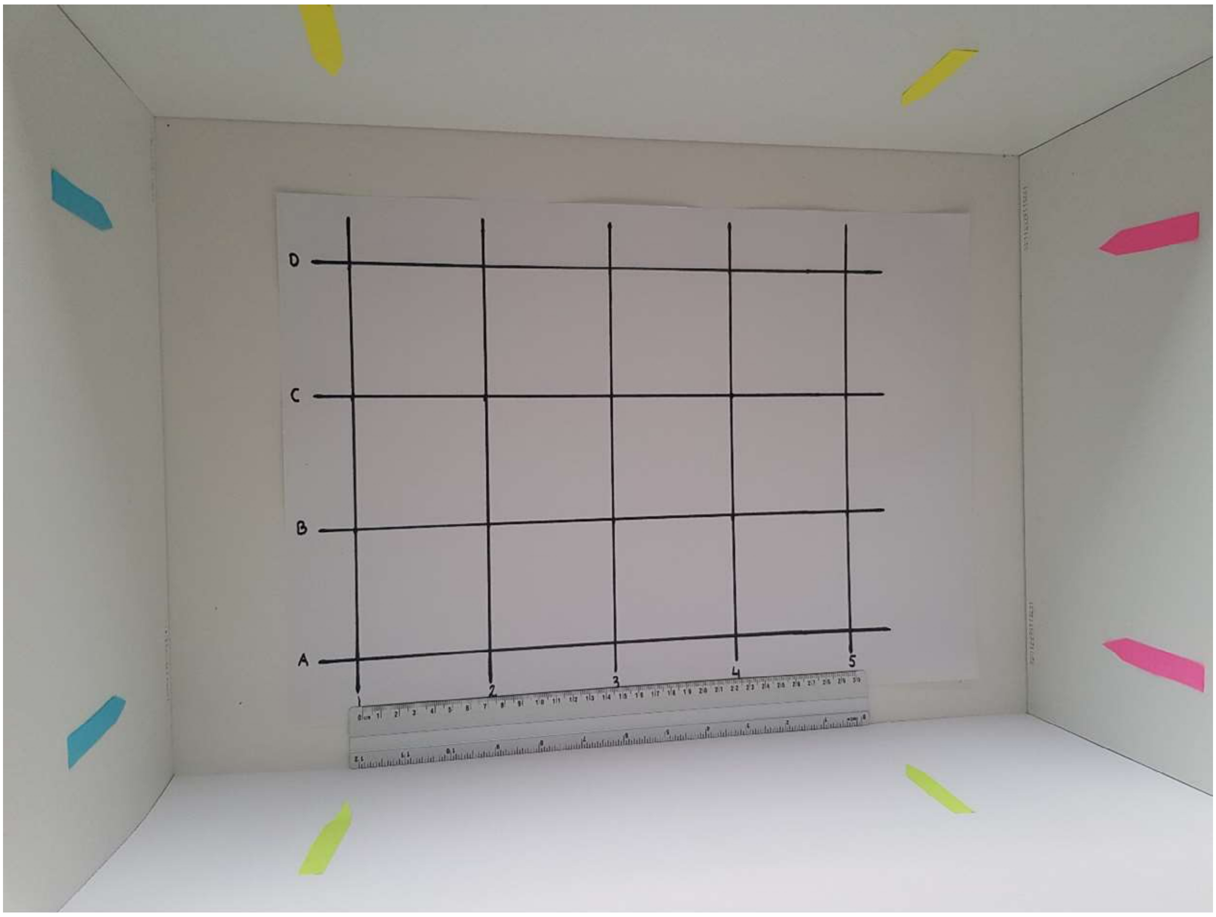


Figure 21: The setup for generating a photogrammetric model

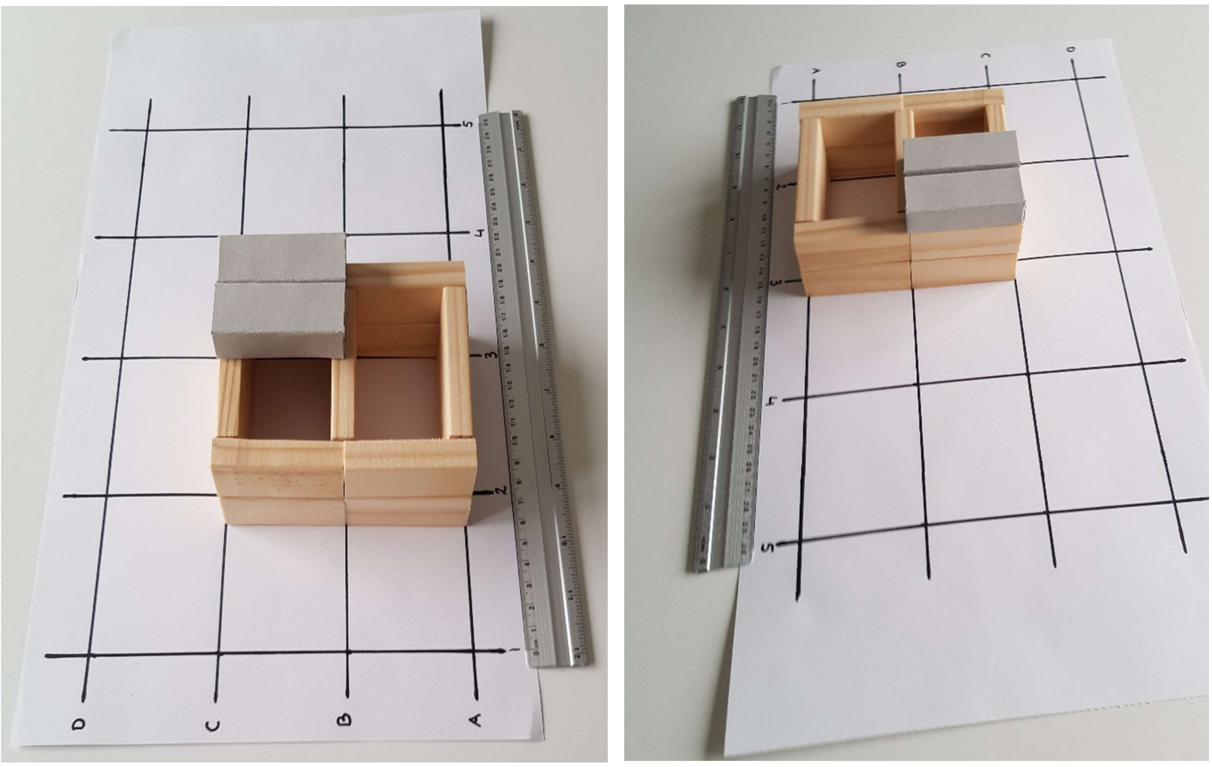


Figure 22: The scale model of the simple fictive project

Point cloud processing

To get a usable point cloud model, many photos must be taken. For this small case study, 217 photos were taken of the scaled model. The photos are taken first from a distance and then from close. In addition, the whole scaled model is covered from many angles with the camera. After making the dataset of the photos, these are ready to be processed for the point cloud model. The tool that is used is called Pix4D. Pix4D is a tool that focuses on the field of photogrammetry and drone mapping. Whether pictures are taken with a smartphone, camera or drone, with Pix4D, it is possible to convert these to higher referenced 2D and 3D models easily. The strength of Pix4D is that it is very user friendly and that some processes in the tool are performed automatically. For example, with a few mouse clicks, aerial photos can be processed into a geo-referenced orthophotograph, 3D model or a point cloud model. Processing of the point cloud model is done via an online server of Pix4D. The benefits of this have already been mentioned in Chapter 4.2.

After uploading the photos in Pix4D, it started to generate a point cloud model. Generating a point cloud model took around 15 minutes. After the point cloud model is processed, it can be visualized. Figure 23 shows the point cloud of the scaled model. As can be seen, the target points are not processed. There can be multiple reasons for this, for example, the target points were too small to detect, or there was not enough lighting coming on the target points. The ruler is, however, processed in the point cloud model. In addition, it is remarkable that the gridlines are also processed. There are also errors to be seen in the point cloud model, where most of them are around the area of the scaled model. Some gaps in the point cloud model are also to be noted. The possible reason for this is that Pix4D could not find enough photos that overlap each other. In other words: the dataset provided may be small. This is common in photogrammetry, but also in other techniques. For this research, it is determined that the point cloud data is sufficient to analyse because all blocks can visually be seen. There is also the option to make the surfaces smoother by generating a textured mesh, but this will be disregarded for this research.

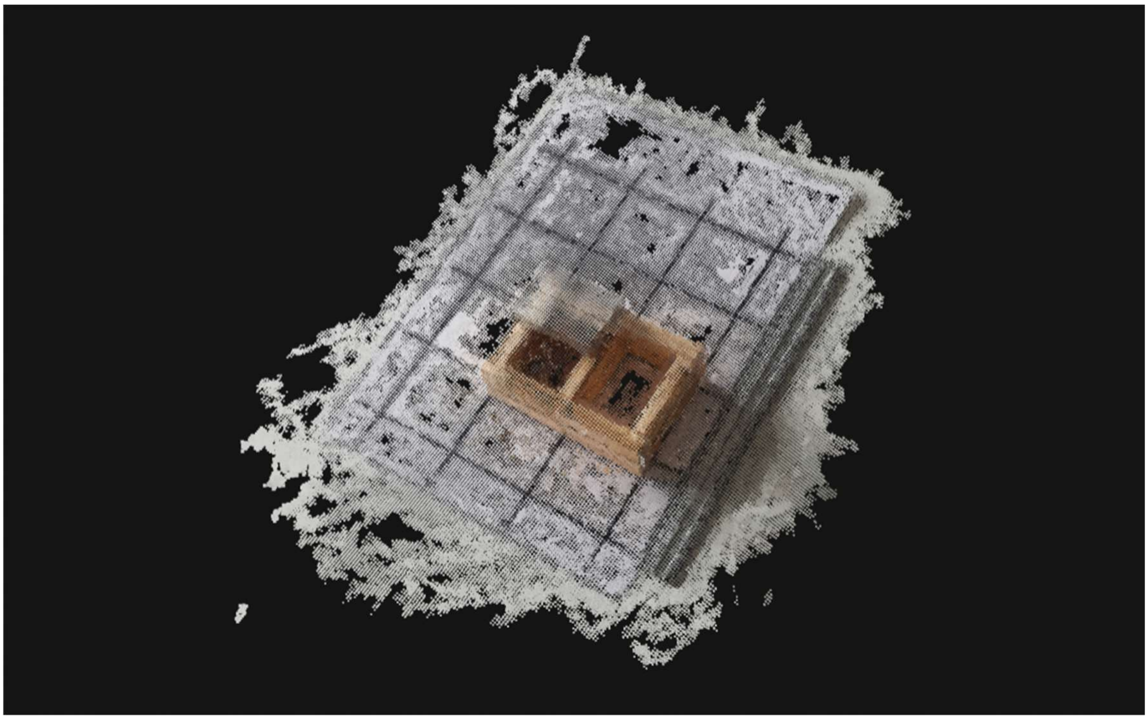


Figure 23: The point cloud model processed with Pix4D

The original data file from Pix4D that can be saved to the local computer is a .LAS file. LAS stands for the first three letters of the word ‘laser’ and is a file format designed for the exchange and archiving of lidar point cloud data. The file has an open and binary format. With the Python package Pyntcloud, the data can be analysed. With the following command, it is possible to read and determine how many points are generated by Pix4D:

```
In [1]: from pyntcloud import PyntCloud

In [2]: cloud = PyntCloud.from_file("data/pointcloud/Project.las")

In [3]: print (cloud.points)

Out []:

            x           y           z           red   green blue

0           934.354     185.329     94.763      36352 37120 35072

1           934.354     185.329     94.763      37376 38144 36096

2           934.354     185.329     94.763      36864 37632 35584

3           934.354     185.330     94.764      35840 36352 34048

4           934.354     185.331     94.764      38912 39424 37120

...         ...         ...         ...         ...   ...   ...

3145479     934.553     185.430     94.882      51968 52224 51456

3145480     934.553     185.431     94.883      52480 52736 51968

3145481     934.553     185.429     94.883      52736 52992 52224

3145482     934.553     185.430     94.883      51968 52224 51456

3145483     934.553     185.430     94.883      53504 53760 52992


[3145484 rows x 6 columns]
```

It can be seen that the point cloud data contains 3.145.484 points and has six columns, which includes the coordinates (X, Y, Z) and the colour code per point (R, G, B). The original .LAS file is 118.531 kilobytes.

After the point cloud data is saved locally, and it is analysed, it is time to scale it. Many tools are available to do this. For this research, it is chosen for the tool CloudCompare. The reason for this is that it is in the first place an open-source tool, which means that it can be used for free. Another reason is that the tool is very simple, and no expertise is needed in order to perform scaling.

During the scaling, it is important to note for a few areas of attention. First, the base point should be determined in order to have a good overlay with the IFC model. Second, a well-known distance should be picked. Since the gridlines and the ruler are visible in the point cloud model, it is chosen to do the scaling via gridlines. To finally scale the model, the gridline intersection A1 is chosen as a base point, and A4 is chosen as a secondary point. Because the distance between A1 and A4 is a well-known distance, the value of 30.000 millimetres can be entered in CloudCompare. The reason to fill in the value of 30.000 millimetres is the fact that the scaled model is at scale 1:100. As the well-known distance is 300 millimetres, it should be multiplied by 100.

4.3. Analyse

In this paragraph, both BIM 4D and point cloud data come together and will be computed with the pythonOCC package. PythonOCC is a python package that provides 3D modelling features and is intended for developers who aim at developing amongst others CAD and BIM applications. The execution of the analyse phase is divided into three steps, which are (1) combining and cleaning data, (2) intersecting data, and (3) export and import data. Note that a planning deviation is made on purpose by adding more 3D objects in the BIM 4D-model (see Chapter 4.1), so some 3D objects cannot be associated with a point from the point cloud. In this way, a scenario can be simulated that the construction is behind schedule.

Step 1: combining and cleaning data

In the plan and capture phase, two files were generated. One was an IFC file from Synchro, and the other was a .LAS file from Pix4D. These two files will be used in this step.

In the world of tooling, it is not possible to communicate with two different data files, unless the process is executed in a systematically way. In this sense, the data should be parsed. Parsing means that something should be made understandable by analysing its parts. In a programming language, it is to convert data represented in one form to another so that it is easier to work with.

Unfortunately, by using only the pythonOCC package, it is not possible to read and view IFC data, because this is not directly supported. So, there should be thought about another way to make it work in pythonOCC. Looking at the bright sight, pythonOCC can import STEP (.stp) files. The meaning and use of STEP is already covered in Chapter 2.5.2. The conversion of IFC to STEP is done via an online converter called 'CAD Exchanger'. This is a really simple tool where IFC models can be uploaded, viewed, and exported to a .stp file.

As the 3D model being sorted out, now the focus is on the point cloud data. The generated point cloud data from Pix4D is a .LAS file. This, however, cannot be read and viewed with the pytonOCC package. In order to do so, the data should be converted to an understandable file, which is, in this case, a .pcd file. During the research, there was not a single converter available that converts .LAS files directly to .pcd files. For this reason, the conversion to a .pcd is done stepwise. First, the .LAS file is converted to a .ply format in CloudCompare. Next, a script is written that converts a .ply file to a .pcd file. Although all coordination files such as .LAS, .pcd and .ply contain the same X, Y, Z coordinates, they have different header text and additional information after the coordinates such as colour and normals. These are one of the important factors that various file formats cannot be read in some tools. In this case, the mentioned script deletes the header text of the .ply file, replaces it with the header text of a .pcd file, and splits each line up to the X, Y, Z coordinates and deletes all excessive information such as colour and normals. The full script for converting .ply to .pcd file can be found in the Appendix.

After the data sources have been parsed, a script can be written. This script combines the .stp and .pcd files together and visually presents the outcome. The tool is structured really simply. On the top, menu bars are created, which are 'PointCloud' and '3Dmodel' (see Figure 24). Inside the 'PointCloud' menu, there is the option to open point cloud data. Inside the '3Dmodel' menu, there is an option created to open the 3D model. Under the menu, there is a viewer created. This viewer shows the output of both files (see Figure 25 and Figure 26).

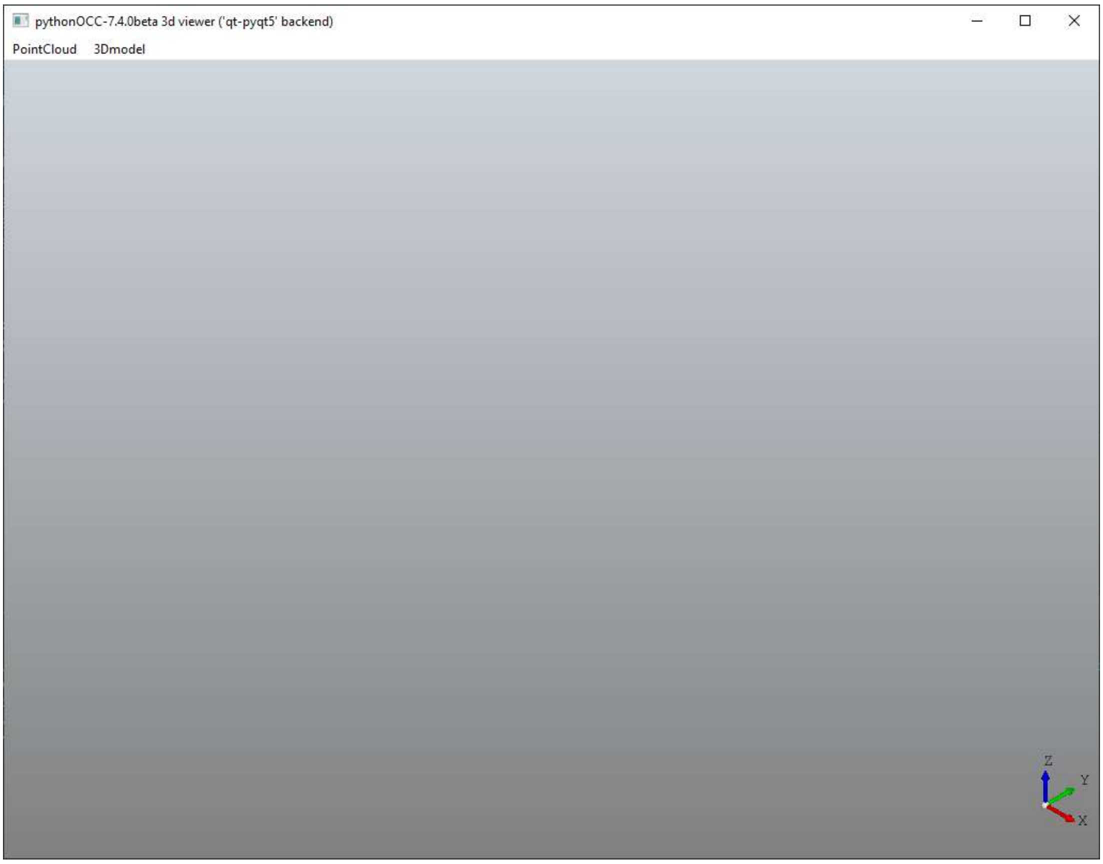


Figure 24: The overall overview of the tool

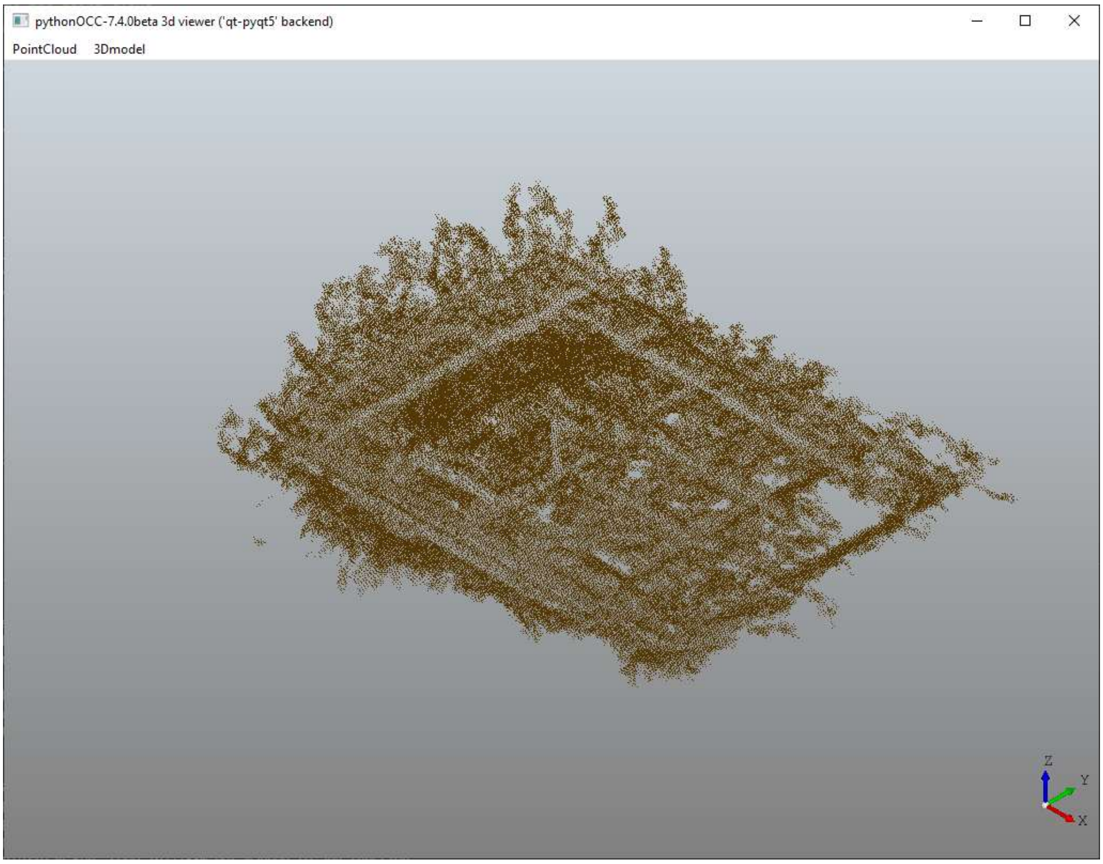


Figure 25: The point cloud data after loaded into the tool

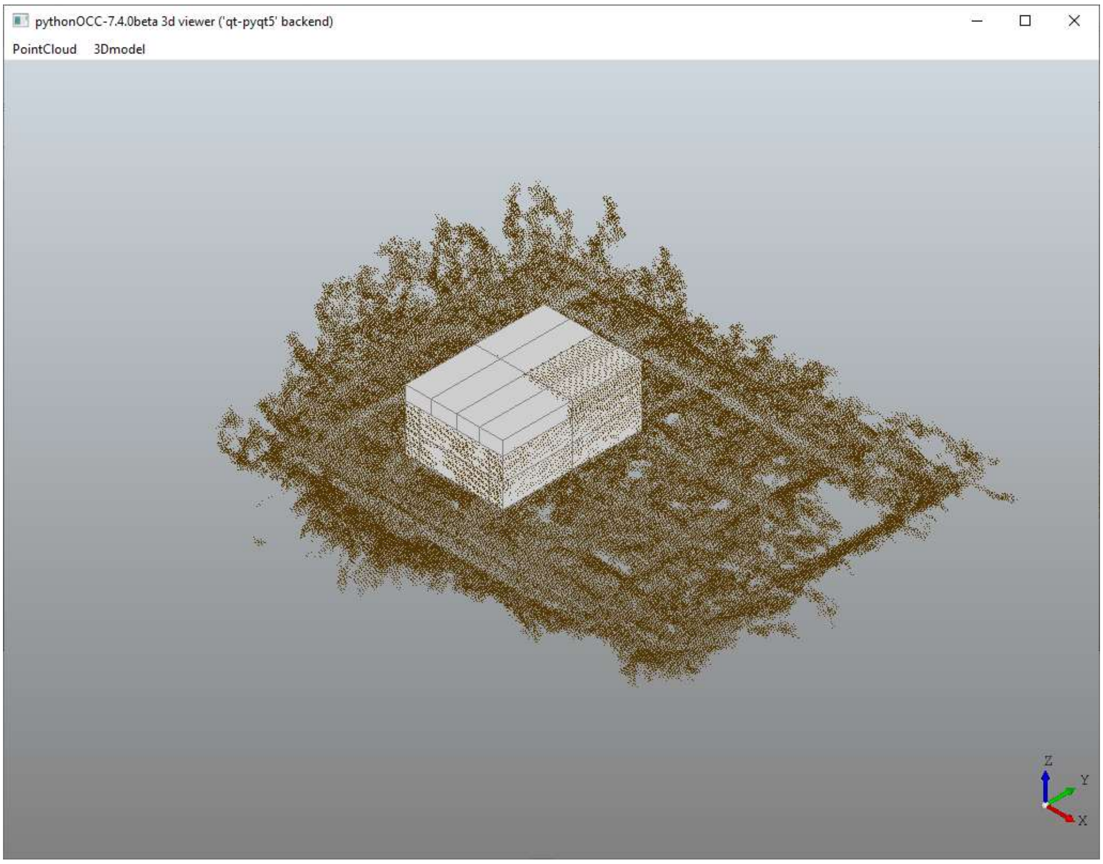


Figure 26: The point cloud data with the 3D model together

It can visually be seen that there is a planning deviation because there are no points in the neighbourhood of some of the blocks that are located on the top of the building. It can also be noted that points may be in the neighbourhood of the surface of the 3D elements, but the distance between the points and the surfaces are not consistent. This phenomenon has to do with scanning error and the natural behaviour of the construction site. The last one mentioned is about that 3D elements are diametrically opposed, and the real constructed elements are not.

As already indicated, the point cloud data consists of 3.145.484 points, which means that the computer should calculate 3.145.484 times, for each point, whether it is near an object surface to finally make an association. To make it easier for the user and also have a less computational time, the point cloud data should be cleaned. This will be done by calculating the normals of all faces of the 3D element object and find the near points for it. A normal (see Figure 29) is a line or a vector that is perpendicular to a given object surface and are heavily used in many areas including 3D shape reconstruction, plane extraction, and point set registration (Poux & Billen, 2019). After calculating all normals of all faces, the distance must be set whether a point is deviated from the actual object, as taken into account that points may have some errors. The maximum distance that is set can be seen as a boundary space, which means that when a point is outside the boundary, the point from the point cloud must be filtered.

For the purpose of cleaning point cloud data, a script is written. This script calculates normal vectors of each plane, creates a boundary space, and deletes point data are outside the boundary space. The distance between the face and the bounding space is set at 25 cm. In addition, a new menu is created, which is called ‘CleanData’. After executing the function inside this menu, the tool deletes irrelevant points. The output of this script is shown in Figure 27.

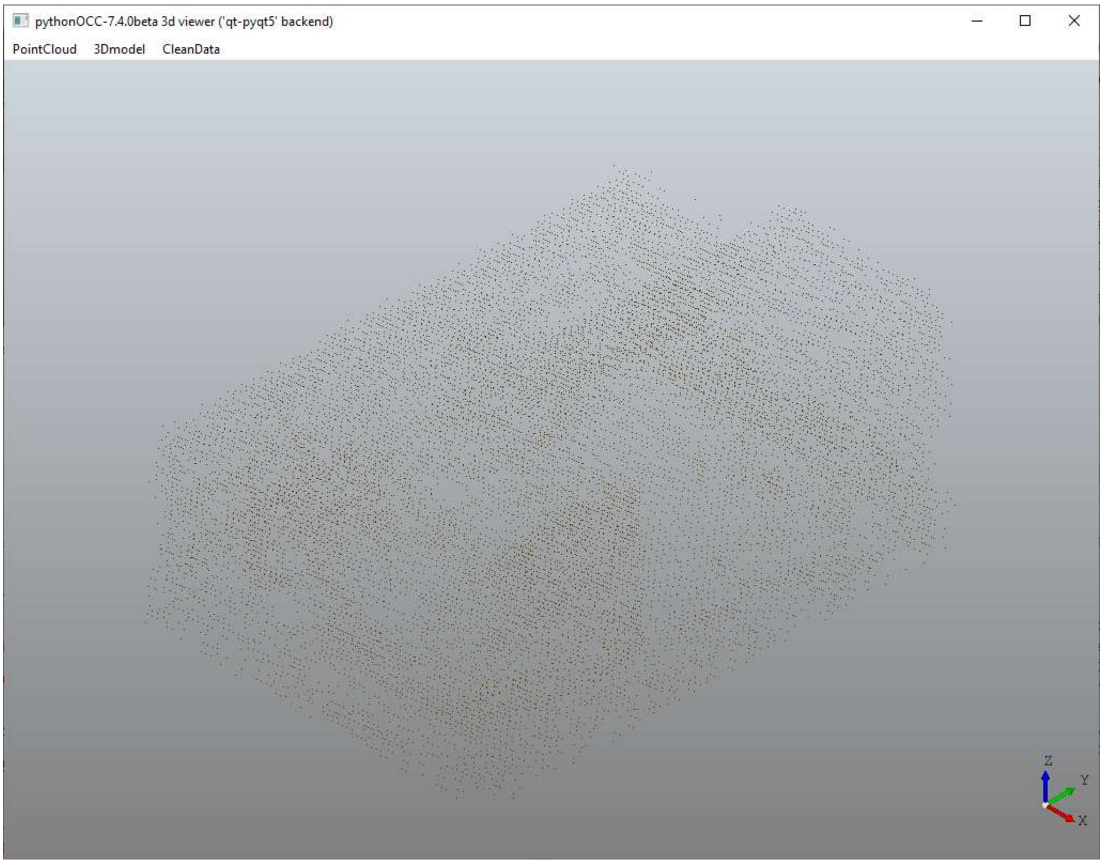


Figure 27: The cleaned point cloud data

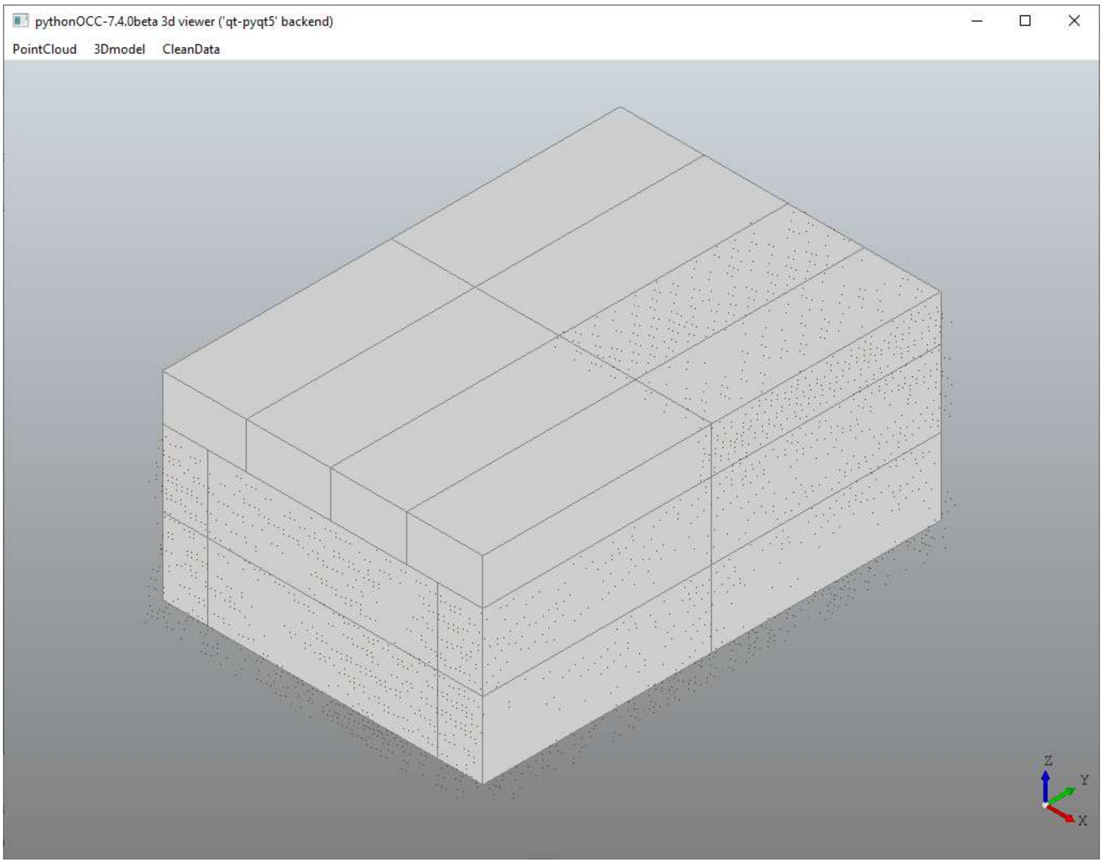


Figure 28: The combined cleaned point cloud data and 3D model

It took about 10 seconds to clean the point cloud with an Intel Core i7-3840QM processor. After cleaning the point cloud data, it is possible to save it. A function in the Python tool is hereby created, which is named: ‘ExportPC’. With the Python package Pyntcloud, the cleaned data can be analysed. It turns out that the cleaned point cloud data consists of 462.661 points, instead of the original 3.145.484 points. This means that the script reduced about 85% of the irrelevant points. In addition, the file size is reduced to 16.977 kB.

After loading the 3D model, it can be seen that, again, there is a deviation between the planning and the construction (see Figure 28). However, this is concluded by the human eye. In the case of the tool, it does not know if there is a deviation. In order to let the tooling compute whether there is a deviation or not, another script should be written. This is done in Step 2.

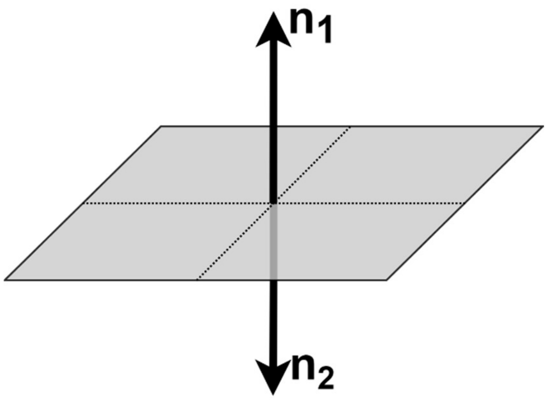


Figure 29: A surface with its two normal vectors n_1 and n_2 on the centre point

Step 2: intersect data

In Step 1, the data was combined and cleaned. Now, it is time to intersect the data and let the tool compute whether the construction has deviated from the 3D model. Intersecting the data, in this case, means that the tool must find all relevant points which are in the neighbourhood of a surface of a 3D object. Also, the tool should tell which element ID has deviated from the actual building. The result for each 3D element will be a 'yes' or a 'no'. In order to do the intersecting, a script needs to be written. It should also be noted that it is possible that a 3D element is intersecting a few points, but in reality, there is no construction element. This can be the result of a point cloud processing error (see Chapter 3.2.1). For that reason, a threshold is created to make sure that the tools are as accurate as possible.

In order to calculate the intersection of points, the method of Tuttas et al. (2014) is going to be used. They defined two different states (A and B) to calculate if a voxel (or a point) is within or without a defined object. State A is based on calculating parts inside and outside the boundary of an object plane by using octrees. This state is found to be not precise, because the octree cell sizes are too large, leading to that the probability that the building part is really existing is low. Therefore, State B is used. This state determines if a point is inside the actual boundary. This is done by making grids for each plane (as described in Chapter 3.3.3).

The 3D objects of the IFC data can be seen as a cube, each with six surface planes. Following State B, each surface can be divided into grids. For this case, a 3x3 grid structure is made. This means that for each plane, there are nine raster cells. Each cell here has its own XY coordinates. In order to know if an object is really existing or not, an equation can be formulated. This is already mentioned in Chapter 3.3.3. The outcome of the equation is in percentages. The higher the percentage, the more likely that the object is existing. For this case, a threshold of 65% is defined. When the result for one plane is above the defined threshold, the plane is existing. To be sure if an object is existing, at least two planes of the 3D object should be above the threshold.

For the purpose of intersecting data, a script is written, which makes 3x3 grids per face; calculates per raster cell if the points are above the threshold; calculates if 65% of raster cells are intersected for one plane; indicates that the 3D object is recognised when two or more planes are recognised with a 'True' value when this is not the case, it gives a 'False' value. For the viewer, a new menu is added, which is called 'IntersectData'. The visualisation of the script can be seen in Figure 30. In here, it can be seen that some 3D objects are deleted. These are the objects that were not intersected with the point cloud because it was not satisfied with the indicated threshold. By comparing this 3D representation with the actual building, it can be seen that the tool indicated fully correctly. It is also possible to colour the 3D objects: green for 3D objects that were intersected and red for 3D objects that are not intersected. In that way, it can visually be seen which object is deviated and which is not (see Figure 31).

The script also gives a textual output, which is the following:

```
Out []:

[True, True, True, True, True, True, True, True, True, True, True, True,
True, True, True, True, False, False, False, False, False, False]
```

Also here, it can be counted that six objects are indicated as 'False'. In order to know specifically which task is corresponded with 'False' and 'True', a small script is written that reads the parameter 'NEXT_ASSEMBLY_USAGE_OCCURRENCE'. This parameter contains object IDs that can be read by the Python tool. The following output is provided:

```
Out []:
Status:                  Object Names:
 False                    Roofs 7:Roofs 1:349102:1_(#1471)
 False                    Roofs 6:Roofs 1:349090:1_(#1403)
 False                    Roofs 8:Roofs 1:349114:1_(#1539)
 True                     Roofs 1:Roofs 1:348926:1_(#1131)
 False                    Roofs 4:Roofs 1:349042:1_(#1267)
 False                    Roofs 2:Roofs 1:348966:1_(#1199)
 False                    Roofs 5:Roofs 1:349078:1_(#1335)
 True                     Roofs 3:Roofs 1:348430:1_(#1060)
 True                     Walls 10:Walls 1:347916_(#737)
 True                     Walls 14:Walls 1:347964_(#981)
 True                     Walls 9:Walls 1:347904_(#676)
 True                     Walls 11:Walls 1:347928_(#798)
 True                     Walls 2:Walls 1:347628_(#310)
 True                     Walls 6:Walls 1:347825_(#493)
 True                     Walls 7:Walls 1:347837_(#554)
 True                     Walls 8:Walls 1:347892_(#615)
 True                     Walls 3:Walls 1:346524_(#161)
 True                     Walls 4:Walls 1:347717_(#371)
 True                     Walls 1:Walls 1:347597_(#249)
 True                     Walls 5:Walls 1:347792_(#432)
 True                     Walls 12:Walls 1:347940_(#859)
 True                     Walls 13:Walls 1:347952_(#920)
```

It now can be seen that True and False values are assigned to 3D objects with their object ID. Unfortunately, the .stp file does not contain data about task names, because it is lost after converting the file. The essence of getting the task names is because it should be known which tasks are deviated from the planning. Step 3 will handle this subject.

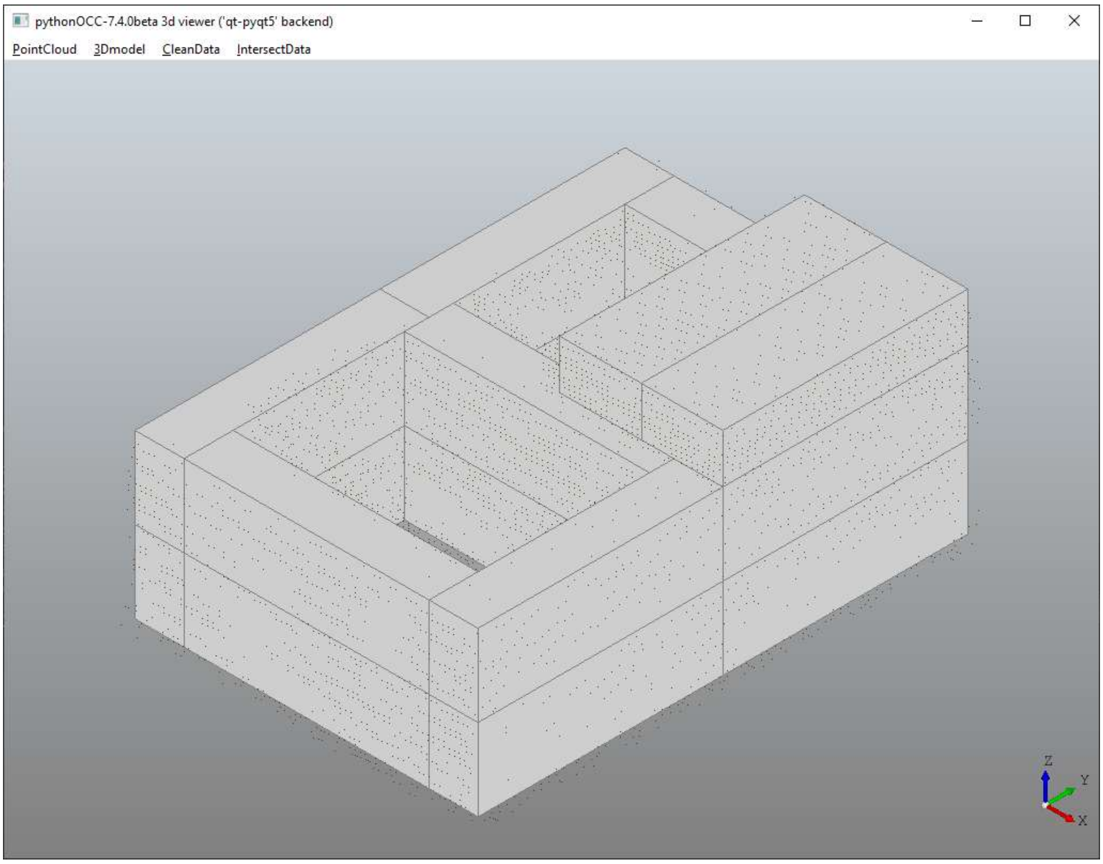


Figure 30: The intersection of data, where some objects that are not intersected are deleted

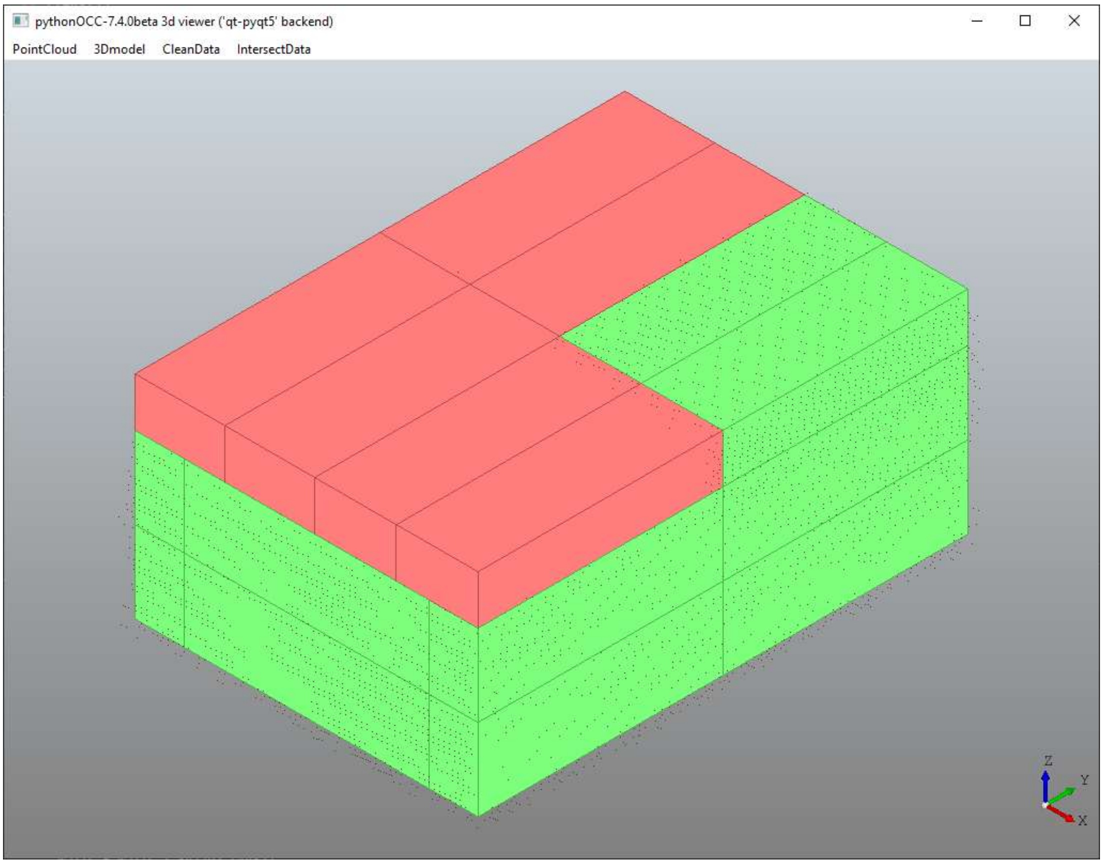


Figure 31: Coloured objects; green indicates that objects are intersected, and red indicates that objects are not intersected

Step 3: Exporting and importing data

After conducting Step 2, the tool now shows which construction elements are placed and which are not, based on the point cloud data. However, it is still only a visual representation where only the 3D objects are shown and not the planned activity. In order to know which 3D objects have deviated from the planning, the output data from the Python tool should be assigned to task activities from the BIM 4D-model. In this way, the planning will finally be updated, and it is clear for the user which activity has deviated.

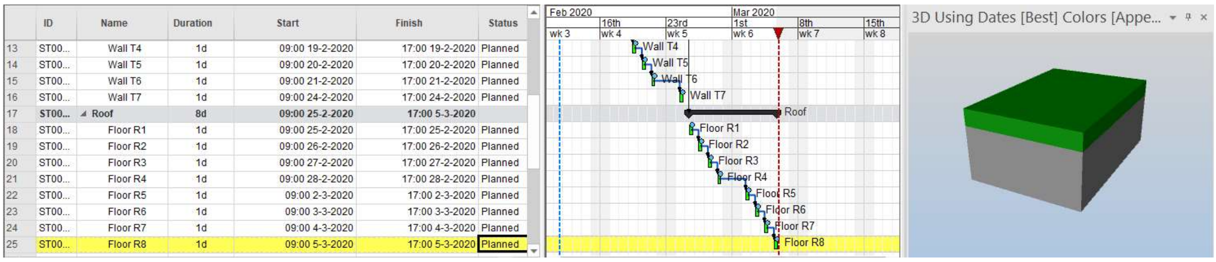
In the BIM 4D-model, there is the possibility to add columns next to the activities. In this case, the column 'Status' is added. With this, it is possible to manually define a task to be ‘Finished’ or ‘Planned’. Finished means that a certain activity has a status of 100% and planned means that the activity did not start yet and has a status of 0%. This information can be used for the Python tool to associate ‘False’ objects with ‘Planned’ status and ‘True’ objects with ‘Finished’ status automatically. As Synchro generates no consistent GUIDs for 3D objects, the association should be done via object IDs. After obtaining all object IDs, the task names can be invoked because they are assigned to each other.

The fictive project started on 05-02-2020 and is planned to finish on 05-03-2020. Figure 32 shows how the building looks like when the project finishes, including the ‘Status’ column. The planned data can be exported as .csv. This file shows multiple tabs. The following tabs are used to obtain task names: ‘Resources’, ‘Assignments’, and ‘Tasks’. Within the Resources tab, the columns ResourceName and ResourceID are shown. ResourceName includes object IDs like ‘Walls 5:Walls 1:347792_(#432)’ and Resource ID includes unique IDs which are assigned to object IDs. For ‘Walls 5:Walls 1:347792_(#432)’ the ID of ‘SR00000450’ is given. The ‘Assignments’ tab includes the columns ‘ResourceID’ and ‘TaskName’. Within this tab, an association is made with ResourceID and TaskName. For Resource ID ‘SR00000450’, the task name is ‘Wall B5’. The final tab is called ‘Tasks’, which is shown in Figure 33. This tab is an exact copy of what it is represented in the BIM 4D-tool Synchro, where the Status is also shown (depicted as number 1). In this column, it is possible to add or change information.

For the purpose of changing the status of each task, a script is written. This script first imports the .csv from the BIM 4D-model; maps object IDs with resource ID in the ‘Resources’ tab; maps subsequently resource ID with task ID in ‘Assignments’ tab; then maps task ID with task names in ‘Task’ tab; finally, changes the status of the activity with the intersected object ID (depicted as number 2 and 3 in Figure 33). This means that when a 3D object is intersected, the task name will finally be linked, and the status will be changed from ‘Planned’ to ‘Finished’. After that, the file can be saved and imported in the BIM 4D-tool. In this way, the status of the activities can be updated automatically, as shown in Figure 34. There is also a condition defined. When a task is already finished, it is not possible to degrade it to planned. This is because when there are scans made for new upper storeys, the construction elements under the new floor that were intersected before cannot be seen by the camera. This means there are no points being generated. No points mean no intersection, and no intersection would lead to ‘False’ objects.

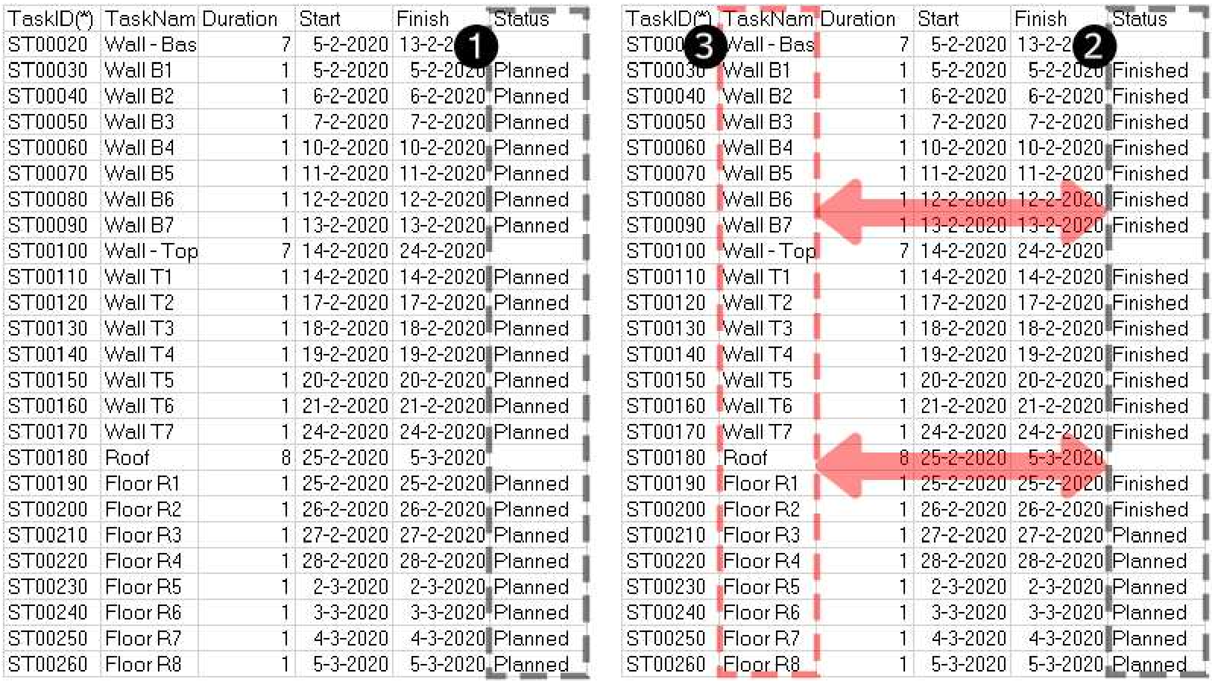
After importing the updated .csv file, the BIM 4D-tool synchronizes the Status column and the timeline shifts directly to the last finished task. It can be seen that the finished activities are up to 26-02-2020. Also, the taskbars become blue, indicating that the task is finished. The bars of the planned tasks are remaining green. The way of updating the BIM 4D-tool with the actual state of works, it is possible to compare with the planned date. In that way, it can be calculated what the deviation is. In Synchro, it is possible to compare this automatically. There are two time windows needed for this: one with the planned task and the other with the actual tasks. Synchro also shows scenarios after tasks have deviated. This is really important, as there are activities that are within the critical path. When these deviate, it has consequences with the date of project delivery. Figure 35 shows the difference between the planned state and the actual state. It can also be seen that Synchro indicates via red lines which tasks have deviated. In this case, floors R3 up to R8 are deviated.

To export the intersected 'True' or 'False' objects with task names, a function is added in the viewer. This function is called 'ExportData'.



	ID	Name	Duration	Start	Finish	Status
13	ST00...	Wall T4	1d	09:00 19-2-2020	17:00 19-2-2020	Planned
14	ST00...	Wall T5	1d	09:00 20-2-2020	17:00 20-2-2020	Planned
15	ST00...	Wall T6	1d	09:00 21-2-2020	17:00 21-2-2020	Planned
16	ST00...	Wall T7	1d	09:00 24-2-2020	17:00 24-2-2020	Planned
17	ST00...	Roof	8d	09:00 25-2-2020	17:00 5-3-2020	
18	ST00...	Floor R1	1d	09:00 25-2-2020	17:00 25-2-2020	Planned
19	ST00...	Floor R2	1d	09:00 26-2-2020	17:00 26-2-2020	Planned
20	ST00...	Floor R3	1d	09:00 27-2-2020	17:00 27-2-2020	Planned
21	ST00...	Floor R4	1d	09:00 28-2-2020	17:00 28-2-2020	Planned
22	ST00...	Floor R5	1d	09:00 2-3-2020	17:00 2-3-2020	Planned
23	ST00...	Floor R6	1d	09:00 3-3-2020	17:00 3-3-2020	Planned
24	ST00...	Floor R7	1d	09:00 4-3-2020	17:00 4-3-2020	Planned
25	ST00...	Floor R8	1d	09:00 5-3-2020	17:00 5-3-2020	Planned

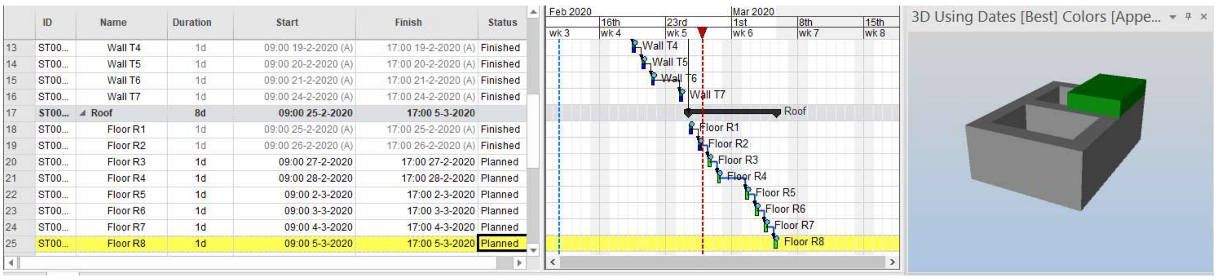
Figure 32: The state of the fictive building after reaching the end of the project



TaskID(*)	TaskNam	Duration	Start	Finish	Status
ST00020	Wall - Bas	7	5-2-2020	13-2-2	
ST00030	Wall B1	1	5-2-2020	5-2-2020	Planned
ST00040	Wall B2	1	6-2-2020	6-2-2020	Planned
ST00050	Wall B3	1	7-2-2020	7-2-2020	Planned
ST00060	Wall B4	1	10-2-2020	10-2-2020	Planned
ST00070	Wall B5	1	11-2-2020	11-2-2020	Planned
ST00080	Wall B6	1	12-2-2020	12-2-2020	Planned
ST00090	Wall B7	1	13-2-2020	13-2-2020	Planned
ST00100	Wall - Top	7	14-2-2020	24-2-2020	
ST00110	Wall T1	1	14-2-2020	14-2-2020	Planned
ST00120	Wall T2	1	17-2-2020	17-2-2020	Planned
ST00130	Wall T3	1	18-2-2020	18-2-2020	Planned
ST00140	Wall T4	1	19-2-2020	19-2-2020	Planned
ST00150	Wall T5	1	20-2-2020	20-2-2020	Planned
ST00160	Wall T6	1	21-2-2020	21-2-2020	Planned
ST00170	Wall T7	1	24-2-2020	24-2-2020	Planned
ST00180	Roof	8	25-2-2020	5-3-2020	
ST00190	Floor R1	1	25-2-2020	25-2-2020	Planned
ST00200	Floor R2	1	26-2-2020	26-2-2020	Planned
ST00210	Floor R3	1	27-2-2020	27-2-2020	Planned
ST00220	Floor R4	1	28-2-2020	28-2-2020	Planned
ST00230	Floor R5	1	2-3-2020	2-3-2020	Planned
ST00240	Floor R6	1	3-3-2020	3-3-2020	Planned
ST00250	Floor R7	1	4-3-2020	4-3-2020	Planned
ST00260	Floor R8	1	5-3-2020	5-3-2020	Planned

TaskID(*)	TaskNam	Duration	Start	Finish	Status
ST000	Wall - Bas	7	5-2-2020	13-2-2	
ST00030	Wall B1	1	5-2-2020	5-2-2020	Finished
ST00040	Wall B2	1	6-2-2020	6-2-2020	Finished
ST00050	Wall B3	1	7-2-2020	7-2-2020	Finished
ST00060	Wall B4	1	10-2-2020	10-2-2020	Finished
ST00070	Wall B5	1	11-2-2020	11-2-2020	Finished
ST00080	Wall B6	1	12-2-2020	12-2-2020	Finished
ST00090	Wall B7	1	13-2-2020	13-2-2020	Finished
ST00100	Wall - Top	7	14-2-2020	24-2-2020	
ST00110	Wall T1	1	14-2-2020	14-2-2020	Finished
ST00120	Wall T2	1	17-2-2020	17-2-2020	Finished
ST00130	Wall T3	1	18-2-2020	18-2-2020	Finished
ST00140	Wall T4	1	19-2-2020	19-2-2020	Finished
ST00150	Wall T5	1	20-2-2020	20-2-2020	Finished
ST00160	Wall T6	1	21-2-2020	21-2-2020	Finished
ST00170	Wall T7	1	24-2-2020	24-2-2020	Finished
ST00180	Roof	8	25-2-2020	5-3-2020	
ST00190	Floor R1	1	25-2-2020	25-2-2020	Finished
ST00200	Floor R2	1	26-2-2020	26-2-2020	Finished
ST00210	Floor R3	1	27-2-2020	27-2-2020	Planned
ST00220	Floor R4	1	28-2-2020	28-2-2020	Planned
ST00230	Floor R5	1	2-3-2020	2-3-2020	Planned
ST00240	Floor R6	1	3-3-2020	3-3-2020	Planned
ST00250	Floor R7	1	4-3-2020	4-3-2020	Planned
ST00260	Floor R8	1	5-3-2020	5-3-2020	Planned

Figure 33: The exported file from the BIM 4D-tool (left) and the linked tasks with status after running the script (right)



	ID	Name	Duration	Start	Finish	Status
13	ST00...	Wall T4	1d	09:00 19-2-2020 (A)	17:00 19-2-2020 (A)	Finished
14	ST00...	Wall T5	1d	09:00 20-2-2020 (A)	17:00 20-2-2020 (A)	Finished
15	ST00...	Wall T6	1d	09:00 21-2-2020 (A)	17:00 21-2-2020 (A)	Finished
16	ST00...	Wall T7	1d	09:00 24-2-2020 (A)	17:00 24-2-2020 (A)	Finished
17	ST00...	Roof	8d	09:00 25-2-2020	17:00 5-3-2020	
18	ST00...	Floor R1	1d	09:00 25-2-2020 (A)	17:00 25-2-2020 (A)	Finished
19	ST00...	Floor R2	1d	09:00 26-2-2020 (A)	17:00 26-2-2020 (A)	Finished
20	ST00...	Floor R3	1d	09:00 27-2-2020	17:00 27-2-2020	Planned
21	ST00...	Floor R4	1d	09:00 28-2-2020	17:00 28-2-2020	Planned
22	ST00...	Floor R5	1d	09:00 2-3-2020	17:00 2-3-2020	Planned
23	ST00...	Floor R6	1d	09:00 3-3-2020	17:00 3-3-2020	Planned
24	ST00...	Floor R7	1d	09:00 4-3-2020	17:00 4-3-2020	Planned
25	ST00...	Floor R8	1d	09:00 5-3-2020	17:00 5-3-2020	Planned

Figure 34: The current state of the fictive project after importing the updated .csv file

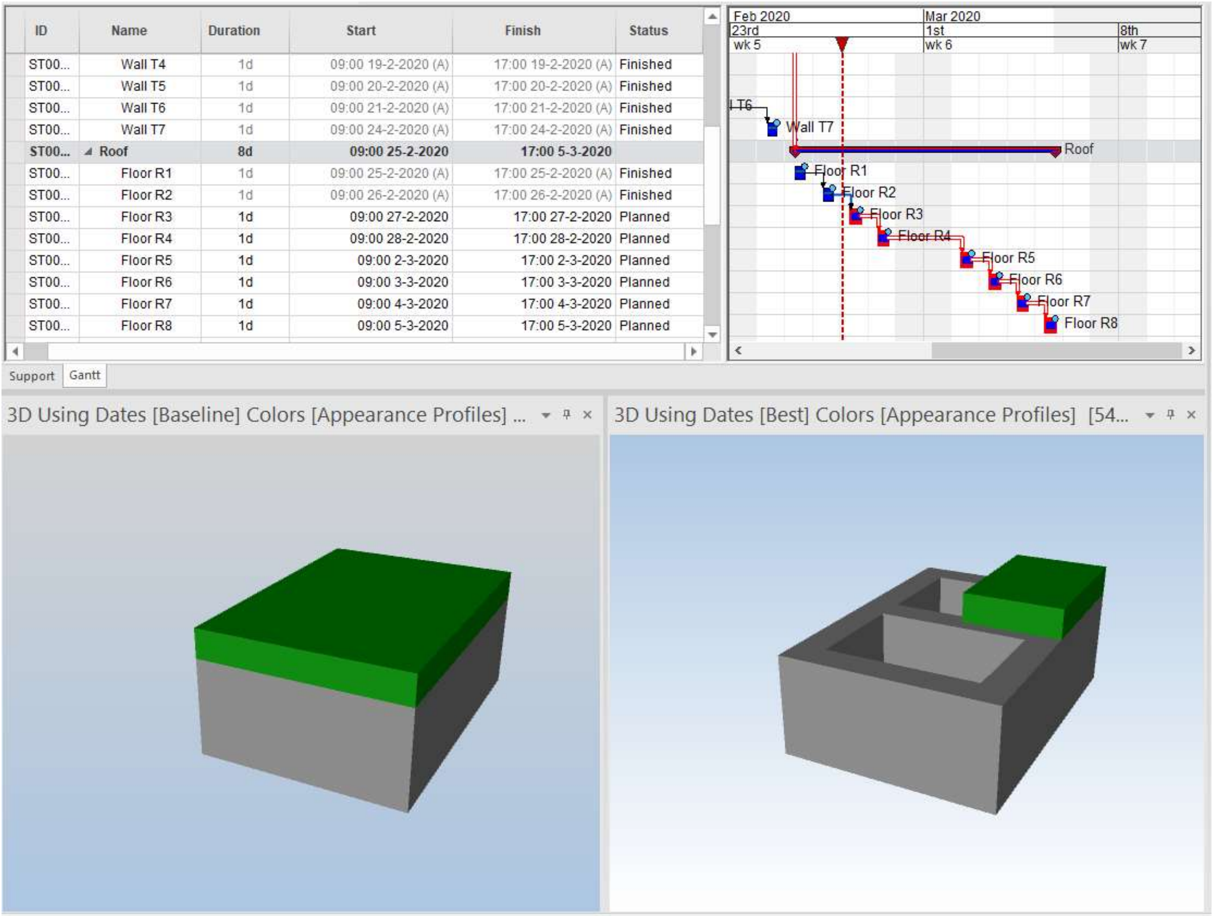


Figure 35: The planned state (left) and actual state (right)

4.4. Conclusion preparation for case study

This small case study gave insight into the capabilities of the Python tool that was developed. The chapter was structured in a stepwise process, where three phases were defined. The phases are: the plan, capture and analyse phase.

In the plan phase, a BIM 3D and a BIM 4D-model of the fictive project are made. In the capture phase, a photogrammetric point cloud model is generated from the scaled model. The analyse phase was divided into three steps, namely: (1) combining and cleaning data; (2) intersecting data; and (3) export and import data. In the first step, the 3D model and the point cloud data are parsed, and the point cloud data is scaled. Further, a script is written to combine both 3D and point cloud data. Next, the point cloud data is cleaned by an additional script. This gave a point reduction of about 85%, which will be beneficial for computing the next steps. The second step intersects point cloud data with 3D objects and associates both data with each other. This gives a list and visualisation of 3D objects which are intersected and which are not. The last step imports BIM 4D-data to the Python tool so that 3D objects can be associated with task names. The status of tasks will change from ‘Planned’ to ‘Finished’ when they are intersected. The updated file can be exported and read by the BIM 4D-tool Synchro. This tool updates the initial planning and notifies which activities are late and which are on time.

Based on the visual comparison of the scaled model and the point cloud data, the tool prediction was 100%. This means that the tool recognised all 3D objects that were situated on the scaled model. This resulted that the information added in the BIM 4D-model was entirely accurate.

This chapter served only to test the capabilities of the tool on a fictive project. As the results provide a positive outlook, it should be tested on a real case study, since this tool should be applied on a real construction site.

5. The case study

After the tool is being used on a small-scale project, it is now time to see whether the tool can be applied on the construction site. This chapter has the same setup as the previous chapter, where the plan, capture and analyse phases are described further in detail.

The construction project where the tool is being tested on is called the Trudo Tower, which is located in Eindhoven. This building is 70 meters high, consists of about 14.000 square meters of floor space area, and has 18 floors. The building has both residential and commercial use. In order to conduct this case study, data is needed. This concerns both BIM 4D-model and point cloud data. These data points will be provided by the construction company Stam + De Koning Bouw bv.

5.1. Plan

The purpose of this paragraph is to collect the BIM-model and construction planning in order to use it in the analyse phase.

<u>BIM-model preparation</u>

The Trudo Tower is designed in the BIM tool Revit. There are five aspect models made, which are: architectural, structural, electrical installation, water installation, and planting model. The boundaries of the project are determined at 32,72x23 meters. The grid of this project consists of nine vertical and seven horizontal gridlines. The mutual distance of the grid lines varies. The most common distance is 3,9 meters. The design of the building is based on the architecture of the building Bosco Verticale. The construction of the Trudo Tower will be very efficient. It will be a prefabricated construction in which the floor plan of each floor is exactly the same. Figure 36 shows the design of the project.

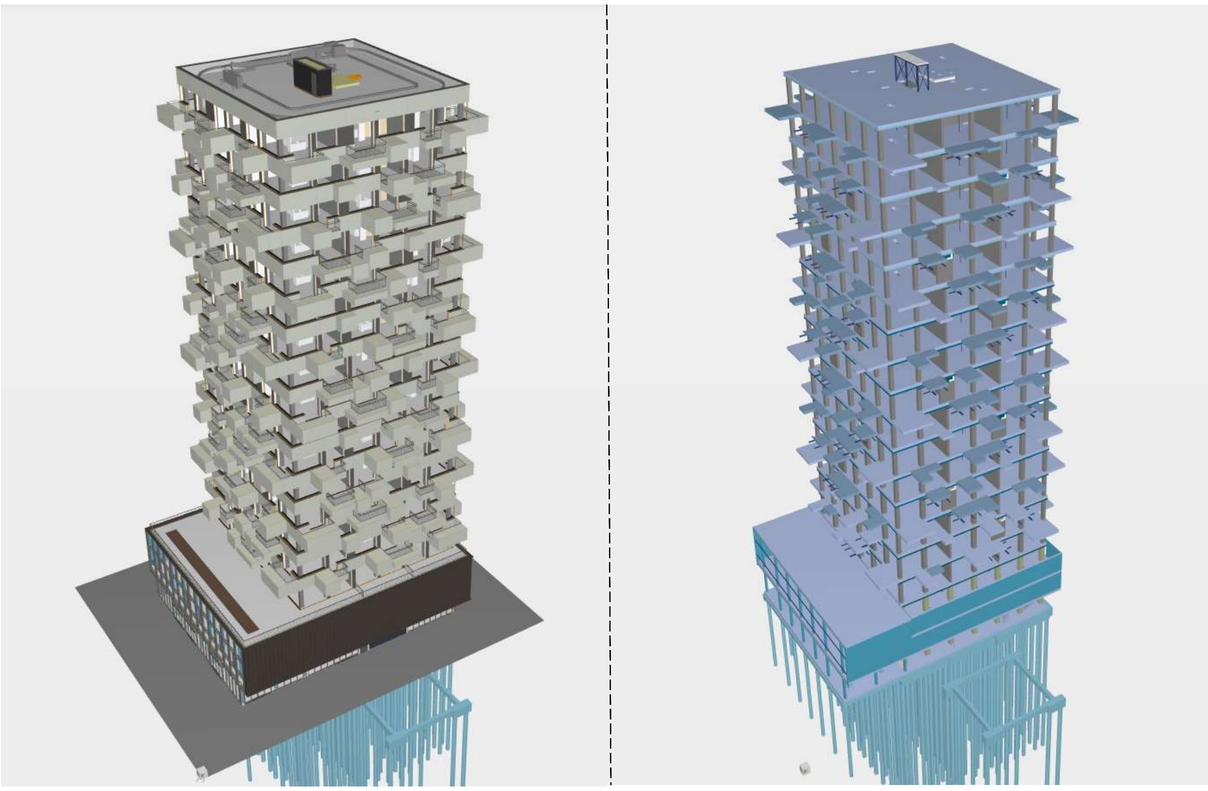


Figure 36: The Trudo Tower, where the architectural model (left) and the structural model (right) can be seen

It is important to define the base point of the 3D model. This is the project origin, which is determined in coordinates. The base point (0, 0, 0) of the project is determined at 15 meters from grid line 2 and 10 meters from grid line A. This can also be seen in Figure 37.

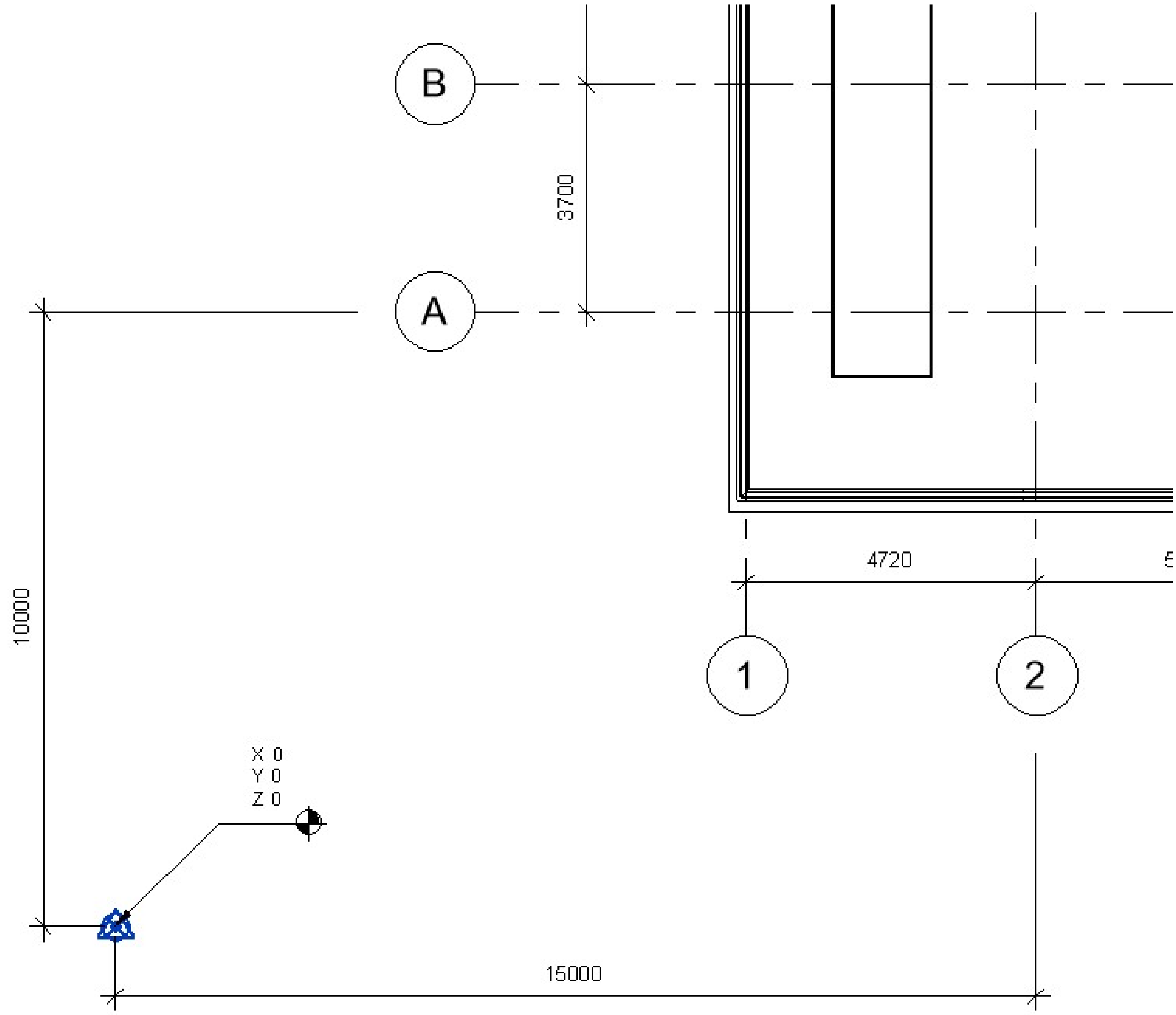


Figure 37: The base point of the BIM model of the Trudo Tower

The 3D model is also available in IFC format. This format can be imported in the BIM 4D tool. The GUIDs (see chapter 2.5.2) shown in the IFC file may be used in the next paragraphs.

With the Python package IfcOpenShell, it is possible to scan the IFC data of the structural model and see which GUID is attached to which 3D object. The following code imports the IfcOpenShell package, opens the structural IFC data from the computer drive, gives a list of the GUIDs of the wall and slab objects, and counts the 3D objects that are listed:

```
In [1]: import ifcopenshell

In [2]: ifc_file = ifcopenshell.open("data/3Dmodel/ O-TTK-CON-
constructief.ifc”)

In [3]: for wall in ifc_file.by_type("IfcWall"):
            print ("wall with global id: "+str(wall.GlobalId))
        for slab in ifc_file.by_type("IfcSlab"):
```

```
            print ("slab with global id: "+str(slab.GlobalId))
        for column in ifc_file.by_type("IfcColumn"):
            print ("column with global id: "+str(column.GlobalId))
In [3]:
num_lines_wall = sum(1 for line in ifc_file.by_type("IfcWall"))
num_lines_slab = sum(1 for line in ifc_file.by_type("IfcSlab"))
num_lines_column = sum(1 for line in ifc_file.by_type("IfcColumn"))
print ("There are [" +str(num_lines_wall) +str("] wall elements"))
print ("There are [" +str(num_lines_slab) +str("] slab elements"))
print ("There are [" +str(num_lines_column) +str("] column elements"))
Out []:
wall with global id: 0BUz$tl2f7huf0YCYUMk0h
wall with global id: 0BUz$tl2f7huf0YCYUMkPv
wall with global id: 0BUz$tl2f7huf0YCYUMkTp
wall with global id: 0BUz$tl2f7huf0YCYUMkHb
wall with global id: 0BUz$tl2f7huf0YCYUMjhi
...
slab with global id: 0BUz$tl2f7huf0YCYUMko$
slab with global id: 3Om0faaGn8ZuKfCOYDbGWT
slab with global id: 0F1jdFA1960RBAmKooYRib
slab with global id: 3AmmXSrzvAje1ZYpLv7Dd4
slab with global id: 2UBPL3QJ1BjxlhpWJXoOXa
...
column with global id: 2q3699IhvB1gmSQ2An4ANZ
column with global id: 3uitQTzrL28RqxedTu8bEW
column with global id: 3uitQTzrL28RqxedTu8bEm
column with global id: 3uitQTzrL28RqxedTu8b90
column with global id: 3uitQTzrL28RqxedTu8b80
...
There are [363] wall elements
There are [293] slab elements
There are [646] column elements
```

The script shows that the structural IFC model contains 363 wall elements, 293 slab elements, and 646 column elements.

BIM 4D preparation

For the BIM 4D-planning, construction planning should be needed. This is most of the time made before the construction phase starts. At later stages, the construction planning was used by Stam + De Koning Bouw bv to create a BIM 4D-model. The purpose was to make a visualisation of the actual construction planning and show this to different stakeholders.

The planning consists of basic construction activities, which are placed in subsequent order. Also, the start and end times are defined for each activity. After the construction planning is ready, a BIM 4D-model can be made. This will happen with the BIM 4D-tool Synchro. With this tool, it is possible to import the IFC model and import the construction planning. Only the fourth floor is created in BIM 4D and is executed at the macro level (see Chapter 2.6.3).

The construction planning of the fourth floor is constructed in the following way. The planning has one main construction activity, which is called '4th floor'. Within this main activity, there are several child activities: thirty-seven to be precise. These activities mostly consist of placing a prefabricated floor, wall or a column. The duration of each activity is also different. This depends on the elements' size: the bigger, the more time it needs to be placed. Also, all activities are placed in subsequent order and have a finish-to-start relationship. This means that all activities are in the critical path. When at least one activity is behind schedule, it will lead to late project delivery.

At first sight, the construction activities look to be placed in random order. It can be seen, for instance, that first two columns are placed, and then three wall elements, following again with columns. There is a reason for this. The order of placing each construction element is in a specific way. This is defined by the construction planner, having the delivery of elements and crane capacity in mind.

After the 3D objects are assigned to a designated construction activity, an animation can be displayed. This animation shows on which date a specific construction activity should be conducted. In this way, it is possible to stop the animation at any moment of time and see which construction activity should be conducted at that specific date see Figure 38, Figure 39, and Figure 40.

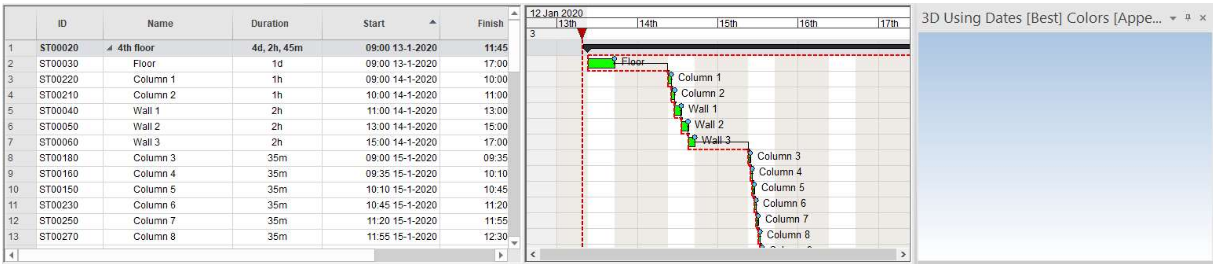


Figure 38: Before the start of the project planning

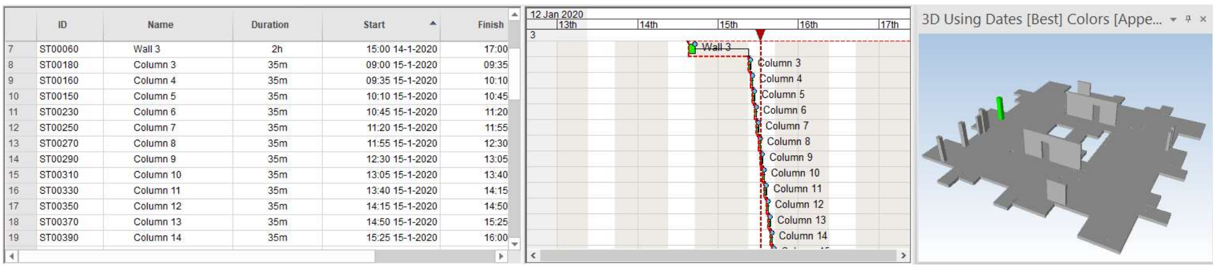


Figure 39: Halfway through the project planning

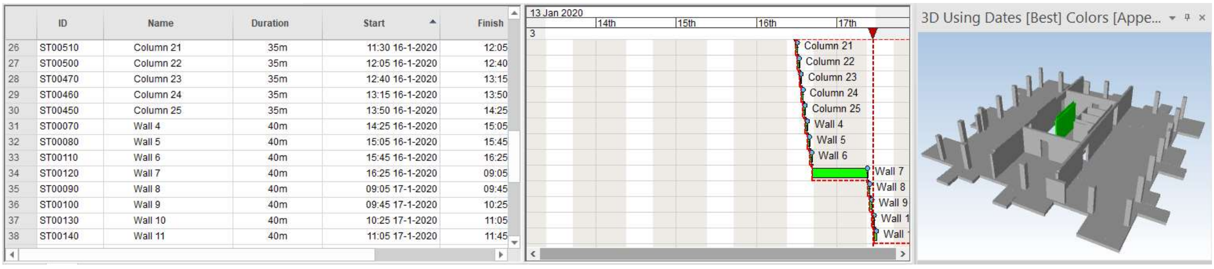


Figure 40: The planning reaching its end

After the BIM 4D-model is finished, it can be exported as IFC. In order to visualize how Synchro assigned 3D objects with tasks, a list can be created. This list is generated with the Python package IfcOpenShell.

```
In [1]: import ifcopenshell

In [2]: model = ifcopenshell.open(r"data/4Dmodel/Trudo4D.ifc")

In [3]: field_width = 25

In [4]: print ("{}\t{}\t{}".format("GUID".ljust(field_width),
"Object_ID".ljust(field_width), "Task_Name"))

In [5]: for object in model.by_type("IFCRELASSIGNSTOPROCESS"):

    try:

        Object_ID = wall[4][0][2]

        GUID = wall[4][0][0]

        Task_Name = wall[6][2]

        print ("{}\t{}\t{}".format
(GUID.ljust(field_width),Object_ID.ljust(field_width), Task_Name))

    except:

        pass

Out []:

GUID                      Object_ID                     Task_Name
1qxVFbEm57zRc3HfzVhRTi    Floor:Floor 1:2849_(#152)     Floor
1owCuZTI96PRSE47$p5$Sa    Columns 3:Columns 2:7147_(#2536) Column 1
13O7btW2z86ersxZZy0Css    Basic Wall:Wall 2:7541_(#3605)  Column 2
2W1NUcRKP0P8Rjn3LpbmLn    Basic Wall:Wall 2:6216_(#1846)  Wall 1
2YJ6H4BcT3rPk$vbWKAzaq    Basic Wall:Wall 2:6214_(#1734)  Wall 2
0ALLw8Aj14JR7uo1j5Jozc    Basic Wall:Wall 2:7713_(#3773)  Wall 3
1kHM0Gzbj1Kh9NxgN0Qxdv    Basic Wall:Wall 2:7651_(#3717)  Column 3
1nTt5CZe14tu9hLHFecsEa    Basic Wall:Wall 2:7620_(#3661)  Column 4
0qwVrCxSfEZwYmGxh1NgKf    Columns 17:Columns 1:7429_(#3407)Column 5
2WazizndH45uYcNV7x9bwP    Basic Wall:Wall 2:7897_(#3885)  Column 6
34NiLHWTL1gBUrK4bWO_VP    Columns 4:Columns 2:7165_(#2603) Column 7
00u40RQfr1x8vsl1ArZoKk    Basic Wall:Wall 2:7850_(#3829)  Column 8
2Uv_jksp514B3vH7fg7DE8    Columns 5:Columns 2:7183_(#2670) Column 9
1IYtdjPVf84BisAL_eUWzu    Columns 6:Columns 2:7201_(#2737) Column 10
0wEjN6P9PEBP7_pT98oCCW    Columns 7:Columns 2:7221_(#2804) Column 11
1GCI_2H998hBYHaqeeV1PF    Columns 18:Columns 1:7447_(#3474)Column 12
0jmMJtwO10Uv_byv_oQzyO    Columns 8:Columns 2:7239_(#2871) Column 13
22VV4S8GvDqu04hKybHada    Columns 9:Columns 2:7257_(#2938) Column 14
25ZqFixs9B0v8lQTyMe4KS    Columns 11:Columns 2:7275_(#3005)Column 15
```

```
0MRNoDZxL1MfvH73XcafBL   Columns 20:Columns 1:7467_(#3541)Column 16
33uXJS0IrDD8yeCm57qCaE   Columns 12:Columns 2:7293_(#3072)Column 17
0w6mSACXb27edZqcEW3sIw   Columns 13:Columns 2:7315_(#3139)Column 18
0jx1ZJEg55jxJ7zl3zv6Tt   Columns 14:Columns 2:7333_(#3206)Column 19
3UON1Ewq55Zh7Oeb1zR2AM   Columns 15:Columns 2:7351_(#3273)Column 20
3D$m_Mt5rCxebgm92125IS   Columns 16:Columns 2:7369_(#3340)Column 21
2adO3lTjn7VxaL9qWkn89V   Columns 19:Columns 1:6450_(#2334)Column 22
1Bym6Zmd969Q2P5yWLwnMH   Columns 10:Columns 2:6342_(#2267)Column 23
1Eo1MYF_vFGBLTQP3dqjDm   Columns 1:Columns 2:7107_(#2402) Column 24
0QHEWheWb5sfPTayzMlLMb   Columns 2:Columns 2:7129_(#2469) Column 25
2$RwRN2rzCYgLUJqNDadyy   Basic Wall:Wall 2:6213_(#1639)   Wall 4
0WWfpnWGjBGhVS_DLr2H9l   Basic Wall:Wall 2:6215_(#1790)   Wall 5
1R33MQ1dr4zvLnR53YHMLy   Basic Wall:Wall 1:6219_(#2039)   Wall 6
3fYPsiSb1C1vLHeJKXD1MG   Basic Wall:Wall 1:6221_(#2151)   Wall 7
0T6kWdEwvD4OaSpe128sL7   Basic Wall:Wall 1:6220_(#2095)   Wall 8
0__$R_5QrDCQGiEktlhavt   Basic Wall:Wall 1:6222_(#2207)   Wall 9
3UAoA3WtTBVget09bsYY1u   Basic Wall:Wall 1:6218_(#1983)   Wall 10
0cSJbHGsf2QxMCw2Ls$de8   Basic Wall:Wall 1:6217_(#1902)   Wall 11
```

Also here, the result of the code shows that Synchro makes new GUIDs for IFC objects. This means that object association (in the Analyse phase) need to be done via Object ID’s or task names instead of GUIDs.

5.2. Capture

The construction of the Trudo Tower was started in 2019 by the contractor Stam + de Koning Bouw bv. During the construction, the contractor records a lot of scan data of the building. The scans are made twice a day with the photogrammetry technique, where the cameras are placed under the jib of the crane. Every five degree of rotation, the crane camera makes pictures. Every picture is automatically aligned with each other and processed by the Pix4D tool, creating a point cloud model. All point cloud data have been stored in a server and can be viewed at any moment of time. In this way, it is possible to go back in time and see how the whole construction process went.

Crane cameras are really ideal because the whole construction area is covered at one single rotation of the crane. To ensure that the areas where construction activities happen are covered, a sufficient picture alignment is needed. Therefore, several cameras are mounted on the jib. The amount of cameras that should be mounted on the jib is dependent on the size and the length-to-width ratio of the building. Also, the cameras are always located on the same plane as the jib. Hence, camera calibration is essential before use.

With the whole process of taking pictures with the crane camera and processing this in Pix4D, not a single human interaction is needed, making the generation of a point cloud model fully automatic.

The specifications of the crane camera are listed in Table 3.

Table 3: Specifications of the crane camera

Number of Megapixels	20
Pixel size (in μm)	1.8
Sensor size (in inch)	1/2.3
Aperture	F2.1
Field of view (in degrees)	69

Point cloud processing

To get a usable point cloud model, many photos must be taken. The number of photos taken is dependent on crane activities. The more crane activity, the more photos. For this case, the scan from 17 January 2020 is used because construction activities of the fourth floor are taking place on that date. From the database, it can be seen that there are 137 photos taken with the crane camera. From these photos, a point cloud model was generated by Pix4D. Generating a point cloud model takes around 1 hour, depending on the server load and the number of photos.

After the point cloud model is processed, it can be visualized. Figure 41 shows the point cloud of the building. As can be seen, the target points are processed, which is a crucial element to make a point cloud model by the photogrammetry technique. In this project, there are four target points placed (see Figure 42). The target points can be used for the scaling. In addition, the location of the target points is based on National Triangle Coordinates (Rijksdriehoekcoördinaten, or in short RD). These are used as a basis for geographical descriptions and files, such as in a geographical information system (GIS) and on land registry maps. The following RD coordinates are determined:

- Target point 1: 159984.695,384198.664;
- Target point 2: 159971.643,384191.238;
- Target point 3: 159964.927,384242.981;
- Target point 4: 159969.334,384221.233.

Figure 41: The point cloud data of Trudo Tower from 17 January 2020

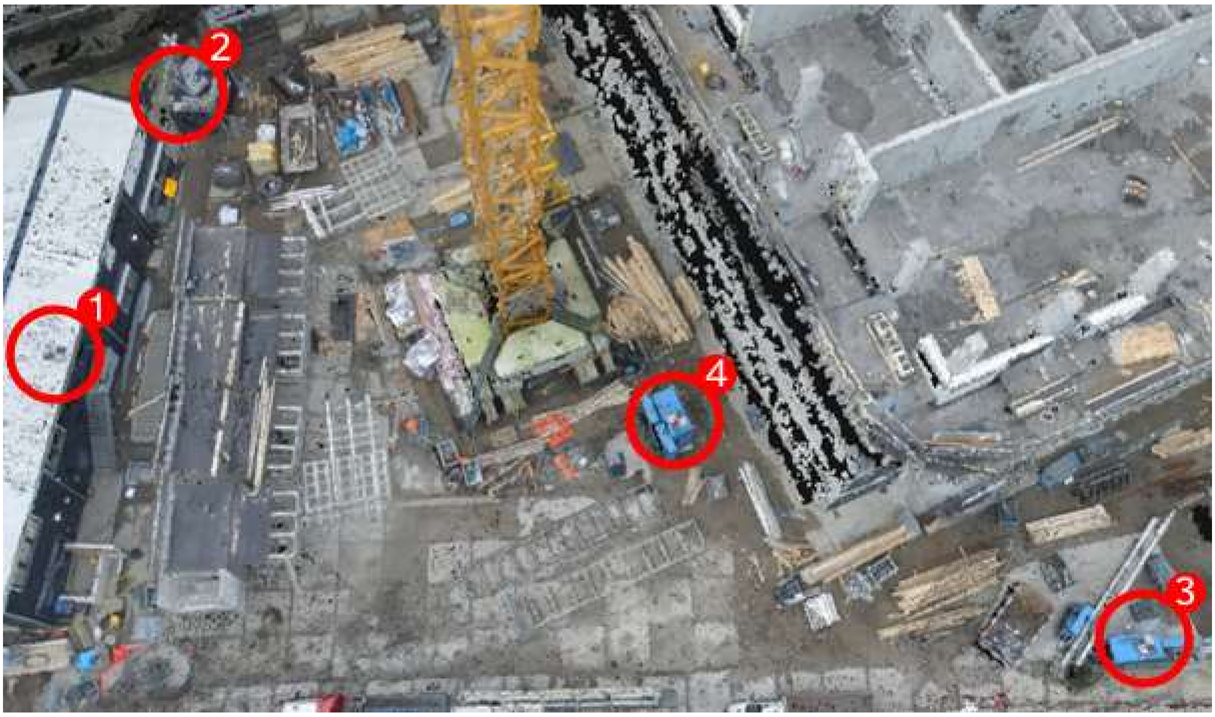


Figure 42: The four target points placed on the construction site

There are also points to be seen that are not related to the building itself, like neighbouring buildings, machinery and materials. In addition, gaps to be noted in the point cloud model. The most interesting part of it is that the level under the current floor cannot be seen. The possible reason for this is that Pix4D could not find enough photos that overlap each other. This is because the angles are too sharp for the crane camera to be processed. For this research, it is determined that the point cloud data is sufficient to analyse because all construction elements on the specified floor can visually be seen.

The original data file from Pix4D that can be saved to the local computer is a .LAS file. With the Python package Pyntcloud, the data can be analysed. With the following command, it is possible to read and determine how many points are generated by Pix4D:

```
In [1]: from pyntcloud import PyntCloud

In [2]: cloud = PyntCloud.from_file("data/pointcloud/Trudo_17012020.las")

In [3]: print (cloud.points)

Out []:

            x           y           z           red   green blue

0           758.849     407.890     13.175      19200 20480 20736

1           758.863     408.307     13.225      22272 24320 24064

2           758.916     407.799     13.188      19200 20736 21248

3           758.941     408.249     13.295      21760 23552 23040

4           758.947     407.953     13.185      23808 25088 25344

...         ...         ...         ...         ...   ...   ...

9751731     926.505     455.879     14.390      6656  7680  7680

9751732     926.538     455.512     14.834      7168  7936  8192

9751733     926.540     455.742     14.532      7680  8192  8448

9751734     926.593     455.691     14.543      7936  8960  8960

9751735     926.622     455.747     14.565      8192  9216  9216


[9751736 rows x 13 columns]
```

It can be seen that the point cloud data contains 9.751.736 points and has six columns, which includes the coordinates (X, Y, Z) and the colour code per point (R, G, B). The original .LAS file is 323.790 kilobytes. The point cloud data that is generated is more than three times larger than the fictive project. This is obvious since the real construction project is bigger in size. However, the expectation was that the more pictures are taken, the more points are generated, but that is not the case.

After the point cloud data is saved locally, and it is analysed, it is time to scale it with the tool CloudCompare. During the scaling, the target points are used to have a good overlay with the IFC model.

5.3. Analyse

In this paragraph, both BIM 4D- and point cloud data come together and will be computed with the pythonOCC package. The execution of the analyse phase is divided into three steps, which are (1) combining and cleaning data, (2) intersect data, and (3) export and import data.

Step 1: combining and cleaning data

In the first step, the point cloud and IFC data are combined together. The point cloud data is converted from .las to .pcd, and the BIM 4D-model of the fourth floor is exported as .IFC and converted to .stp. The Python tool is used to combine both data. Figure 43 shows that both data are overlaid very well. There are no dense points to be seen that deviate from the 3D objects.

By zooming the viewer out, it can be seen that there are a lot of points that are unnecessary to process, like surrounded buildings, materials, and equipment (see Figure 44). In order to reduce the points that are not relevant, it should be cleaned. The point cloud data of the fictive project was already cleaned by the Python tool. The same script will be used again for this building.

When executing the script, the Python tool gave an error:

```
Out []:

Traceback (most recent call last):
File "Step 1 intersect data trudo.py", line 314, in DataClean
cubes.append(Cube(vertices))
File "Step 1 intersect data trudo.py", line 95, in __init__
v = [x for x in list(l1) if x in list(l2)][0]
IndexError: list index out of range
```

This error indicates that the list index of the 3D model is out of range. This has to do with that there were assumptions made that 3D objects have six faces. In this case, there were 3D objects that have seven or more faces, because objects have different geometry other than rectangular. In the 3D model, it can be seen that the floor has holes, the walls have door entry and extra geometry added on top, leading to the indicated error. It is possible to develop a script that calculates only the six major faces and excluding the others. Due to time constraint, this is not possible. Another way was found by making a new file where only six faces exist per object, so without holes, door openings, and extra geometry that may lead to more faces than six. The new 3D model can be seen in Figure 45. There is also a BIM 4D-model made based on the new 3D model.

After the new 3D model is loaded in the Python tool, it is possible to intersect it with point cloud data. In Figure 46, it can be seen that the surrounded points are cleaned, and only the points that intersect with the 3D objects are visible. It took about 1 minute and 8 seconds to clean the point cloud. The cleaned point cloud data consists of 944.759 points, instead of the original 9.751.736 points. This means that the script reduced about 90% of irrelevant points. In addition, the file size is reduced to 36.841 kB. Figure 47 shows that 3D objects fit snugly into the set of points. Step 2 will go further about intersecting the data and indicate which 3D objects are built.

Figure 43: The point cloud data with the 3D model together

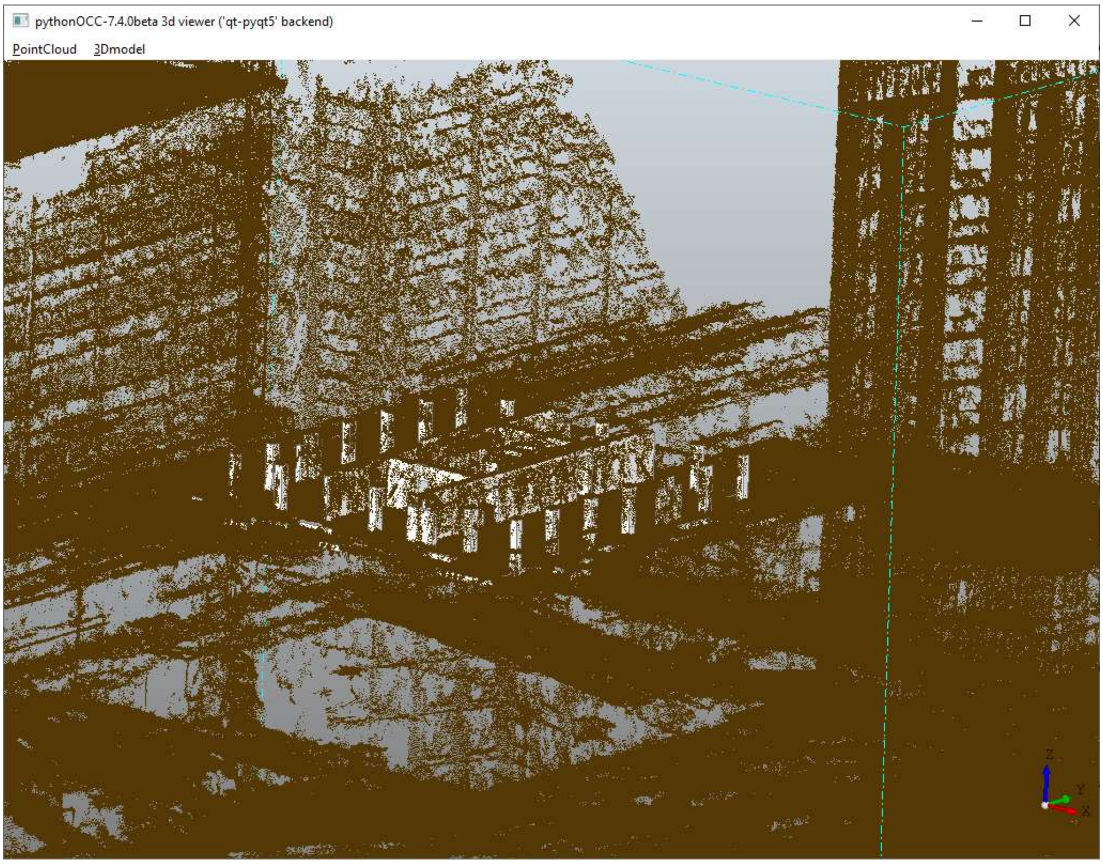


Figure 44: The 3D model with surrounded points

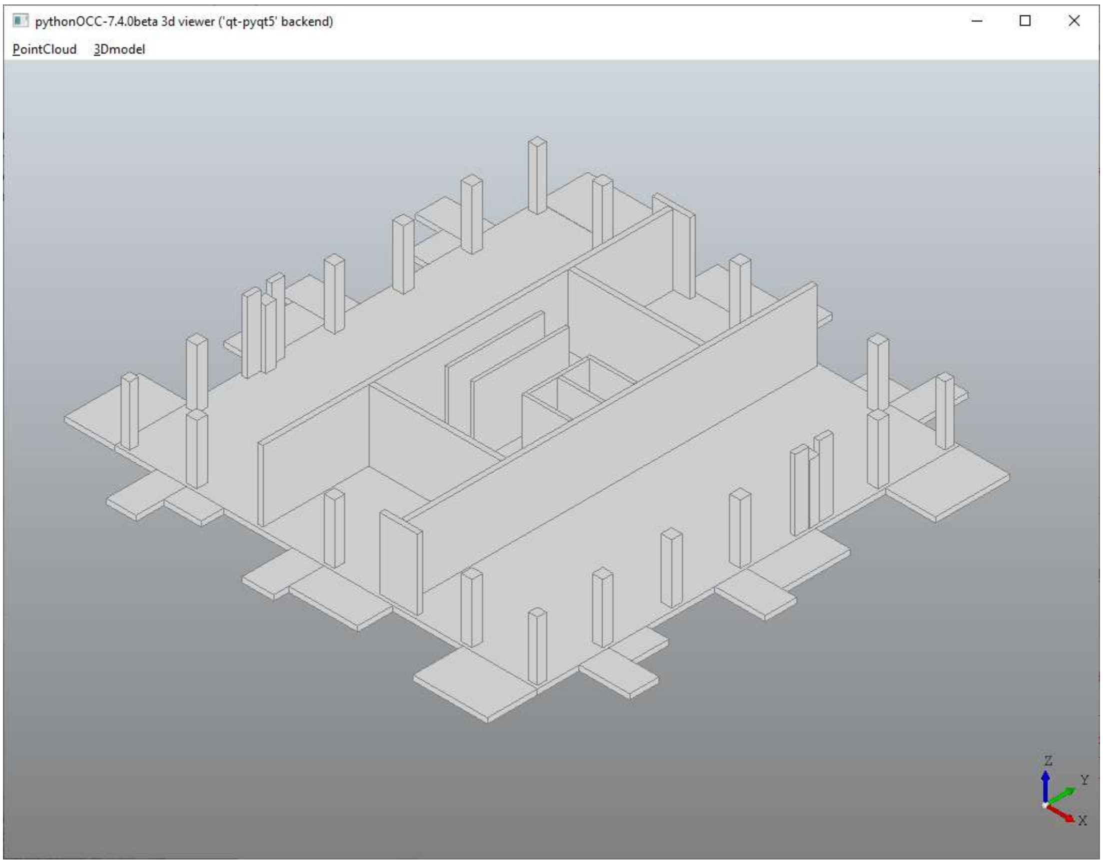


Figure 45: New 3D model with only six faces per 3D object

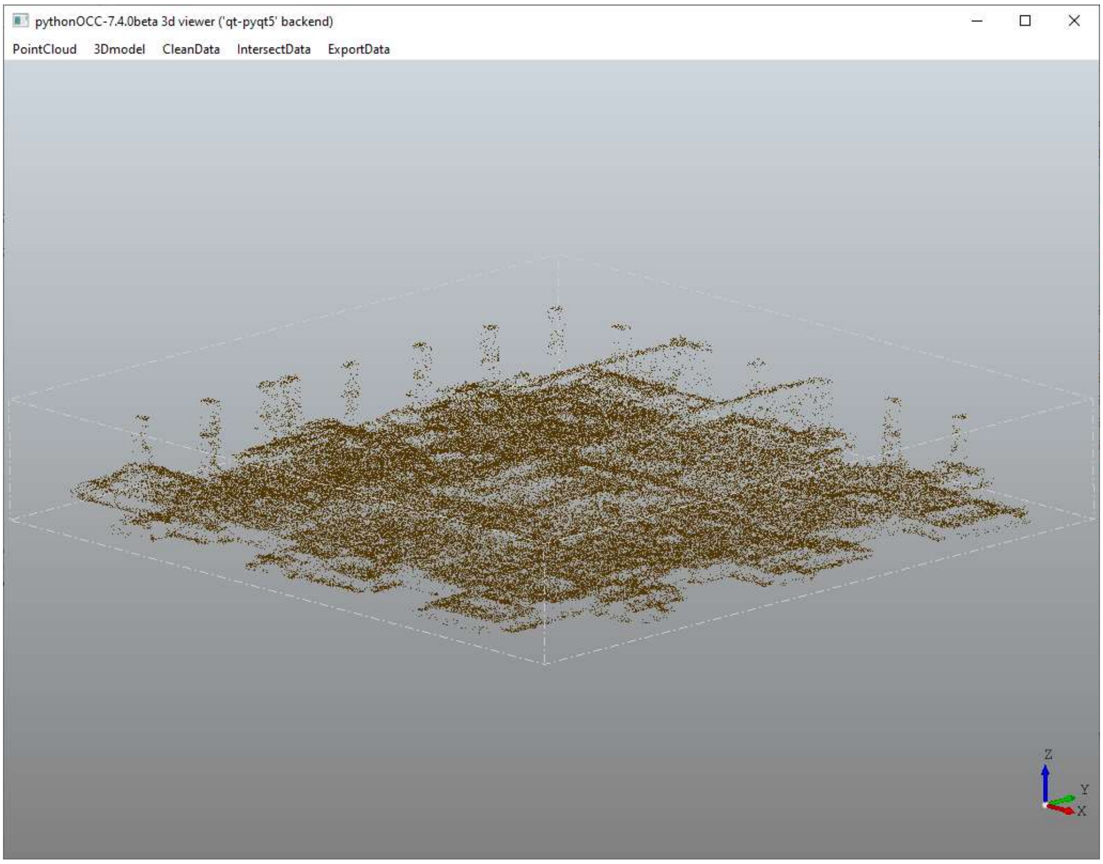


Figure 46: The cleaned point cloud data

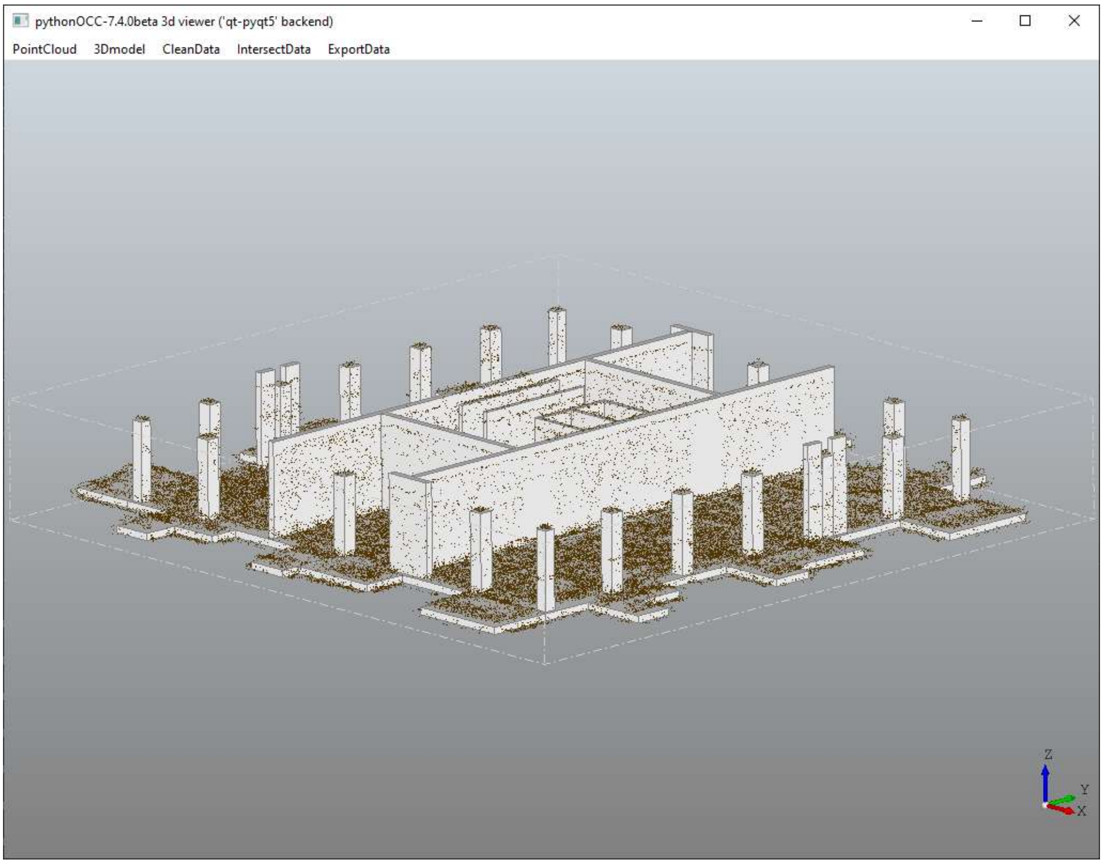


Figure 47: The combined cleaned point cloud data and 3D model

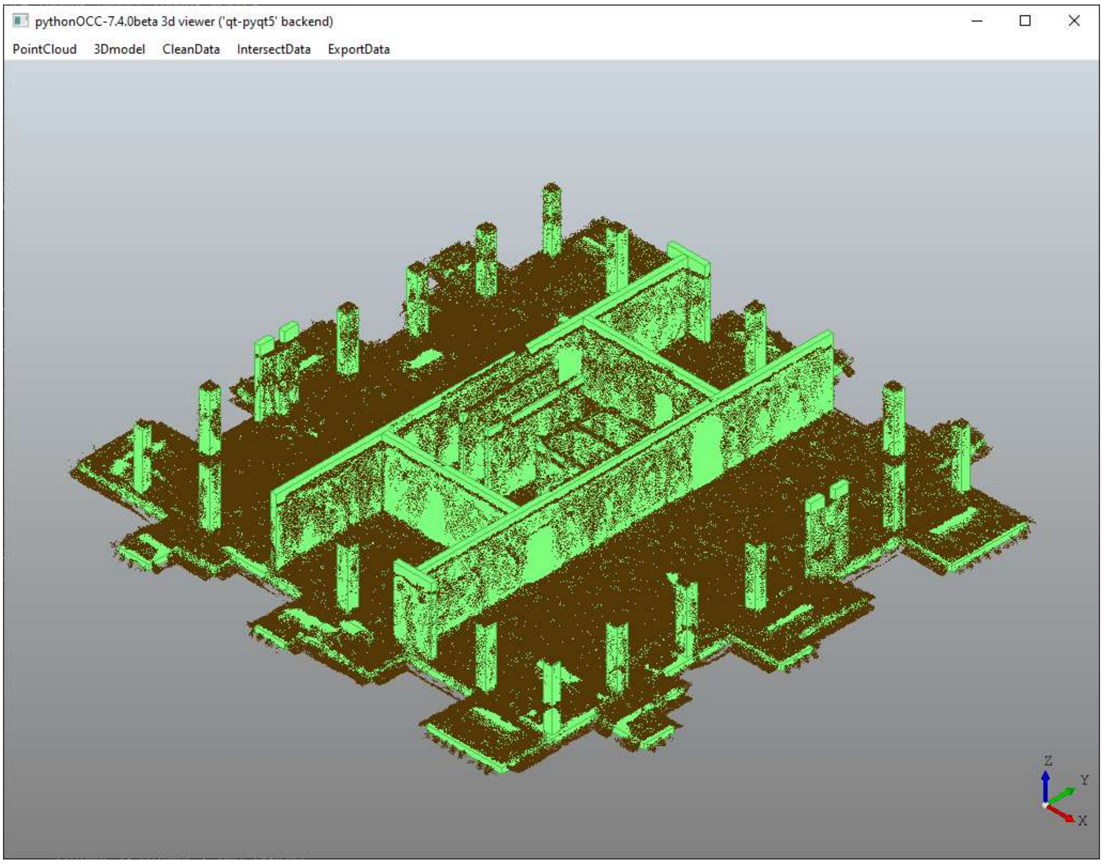


Figure 48: Coloured objects; green indicates that objects are intersected

Step 2: intersect data

After the data are combined and the point cloud is cleaned, the data can be intersected. The same script will be used as handled in the previous chapter. The visualisation of the script is the same as in Figure 47. This means that all objects are intersected with the point cloud because it is satisfied with the indicated threshold. Comparing this 3D representation with the actual construction site, it can be seen that the tool indicated fully correctly. It is also possible to colour code the 3D objects: green for 3D objects that were intersected and red for 3D objects that are not intersected. In that way, it can visually be seen which object is deviated and which is not. Figure 48 shows that all 3D objects turned green.

The script also gives a textual output, which is the following:

```
Out []:

[True, True, True, True, True, True, True, True, True, True, True, True,
True, True, True, True, True, True, True, True, True, True, True, True,
True, True, True, True, True, True, True, True, True, True, True, True,
True, True, True, True, True, True, True, True, True, True, True, True,
True, True, True, True, True, True, True, True, True]
```

Executing the script where also object IDs are shown, is summarized below in the form of a list:

```
Out []:

Status:                 Object Names:

 True                    Basic Wall:Wall 1:6221_(#2151)

 True                    Basic Wall:Wall 2:7850_(#3829)

 True                    Basic Wall:Wall 2:6215_(#1790)

 True                    Basic Wall:Wall 1:6218_(#1983)

...                      ...

 True                    Columns 13:Columns 2:7315_(#3139)

 True                    Columns 18:Columns 1:7447_(#3474)

 True                    Columns 19:Columns 1:6450_(#2334)

 True                    Columns 17:Columns 1:7429_(#3407)
```

Also here, it can be counted that all objects are indicated as ‘True’. In order to know which task corresponds with the ‘True’ value, the planning data from the BIM 4D-model must be used. This will be handled in Step 3.

Step 3: Exporting and importing data

Step 2 showed which construction elements are placed and which are not. This was all based on the point cloud data. As this is only a visual representation, the results should be associated with the data from the BIM 4D-model. The same setup will be used in the small fictive project, where the object association is done via Object IDs and task names.

The construction of the fourth floor of the Trudo Tower started on 13-01-2020 and is planned to finish on 17-01-2020. Figure 50 shows how the building looks like when the fourth floor finishes. The planned data (see Figure 49) is imported in the Python tool as .csv. The same overview can be seen in the previous chapter. The only difference is the content of activities, like activity name, start and finished dates, and status. The status of each activity is depicted as number 1. This is a raw content derived from the BIM 4D-tool. After executing the Python tool to export the intersected data, the Status column of the .csv file changes (depicted as number 2 and 3). It can be seen that all activities changed from ‘Planned’ to ‘Finished’. This new file can be imported in the BIM 4D-tool. It can be seen that the status of the activities is updated automatically, as shown in Figure 51. The Gantt Chart shows that the bars of all activities are coloured in blue. This indicates that all activities for the fourth floor are finished. Putting two 3D frames side by side (see Figure 52), where one is the planned data and the other actual data, it can be seen that there is no planning deviation.

TaskID(*)	TaskNam	Duration	Start	Finish	Status
ST00020	4th floor	4	13-1-2020	17-1-20[illegible]	1
ST00030	Floor	1	13-1-2020	13-1-20[illegible]	Planned
ST00220	Column 1	0	14-1-2020	14-1-2020	Planned
ST00210	Column 2	0	14-1-2020	14-1-2020	Planned
ST00040	Wall 1	0	14-1-2020	14-1-2020	Planned
ST00050	Wall 2	0	14-1-2020	14-1-2020	Planned
ST00060	Wall 3	0	14-1-2020	14-1-2020	Planned
ST00180	Column 3	0	15-1-2020	15-1-2020	Planned
ST00160	Column 4	0	15-1-2020	15-1-2020	Planned
ST00150	Column 5	0	15-1-2020	15-1-2020	Planned
ST00230	Column 6	0	15-1-2020	15-1-2020	Planned
ST00250	Column 7	0	15-1-2020	15-1-2020	Planned
ST00270	Column 8	0	15-1-2020	15-1-2020	Planned
ST00290	Column 9	0	15-1-2020	15-1-2020	Planned
ST00310	Column 10	0	15-1-2020	15-1-2020	Planned
ST00330	Column 11	0	15-1-2020	15-1-2020	Planned
ST00350	Column 12	0	15-1-2020	15-1-2020	Planned
ST00370	Column 13	0	15-1-2020	15-1-2020	Planned
ST00390	Column 14	0	15-1-2020	15-1-2020	Planned
ST00410	Column 15	0	15-1-2020	15-1-2020	Planned
ST00430	Column 16	0	15-1-2020	16-1-2020	Planned
ST00660	Column 17	0	16-1-2020	16-1-2020	Planned
ST00650	Column 18	0	16-1-2020	16-1-2020	Planned
ST00570	Column 19	0	16-1-2020	16-1-2020	Planned
ST00560	Column 20	0	16-1-2020	16-1-2020	Planned
ST00510	Column 21	0	16-1-2020	16-1-2020	Planned
ST00500	Column 22	0	16-1-2020	16-1-2020	Planned
ST00470	Column 23	0	16-1-2020	16-1-2020	Planned
ST00460	Column 24	0	16-1-2020	16-1-2020	Planned
ST00450	Column 25	0	16-1-2020	16-1-2020	Planned
ST00070	Wall 4	0	16-1-2020	16-1-2020	Planned
ST00080	Wall 5	0	16-1-2020	16-1-2020	Planned
ST00110	Wall 6	0	16-1-2020	16-1-2020	Planned
ST00120	Wall 7	0	16-1-2020	17-1-2020	Planned
ST00090	Wall 8	0	17-1-2020	17-1-2020	Planned
ST00100	Wall 9	0	17-1-2020	17-1-2020	Planned
ST00130	Wall 10	0	17-1-2020	17-1-2020	Planned
ST00140	Wall 11	0	17-1-2020	17-1-2020	Planned

TaskID(*)	TaskNam	Duration	Start	Finish	Status
ST000[illegible] 3	4th floor	4	13-1-2020	17-1-2[illegible]	2
ST000[illegible]	Floor	1	13-1-2020	13-1-20[illegible]	Finished
ST00220	Column 1	0	14-1-2020	14-1-2020	Finished
ST00210	Column 2	0	14-1-2020	14-1-2020	Finished
ST00040	Wall 1	0	14-1-2020	14-1-2020	Finished
ST00050	Wall 2	0	14-1-2020	14-1-2020	Finished
ST00060	Wall 3	0	14-1-2020	14-1-2020	Finished
ST00180	Column 3	0	15-1-2020	15-1-2020	Finished
ST00160	Column 4	0	15-1-2020	15-1-2020	Finished
ST00150	Column 5	0	15-1-2020	15-1-2020	Finished
ST00230	Column 6	0	15-1-2020	15-1-2020	Finished
ST00250	Column 7	0	15-1-2020	15-1-2020	Finished
ST00270	Column 8	0	15-1-2020	15-1-2020	Finished
ST00290	Column 9	0	15-1-2020	15-1-2020	Finished
ST00310	Column 10	0	15-1-2020	15-1-2020	Finished
ST00330	Column 11	0	15-1-2020	15-1-2020	Finished
ST00350	Column 12	0	15-1-2020	15-1-2020	Finished
ST00370	Column 13	0	15-1-2020	15-1-2020	Finished
ST00390	Column 14	0	15-1-2020	15-1-2020	Finished
ST00410	Column 15	0	15-1-2020	15-1-2020	Finished
ST00430	Column 16	0	15-1-2020	16-1-2020	Finished
ST00660	Column 17	0	16-1-2020	16-1-2020	Finished
ST00650	Column 18	0	16-1-2020	16-1-2020	Finished
ST00570	Column 19	0	16-1-2020	16-1-2020	Finished
ST00560	Column 20	0	16-1-2020	16-1-2020	Finished
ST00510	Column 21	0	16-1-2020	16-1-2020	Finished
ST00500	Column 22	0	16-1-2020	16-1-2020	Finished
ST00470	Column 23	0	16-1-2020	16-1-2020	Finished
ST00460	Column 24	0	16-1-2020	16-1-2020	Finished
ST00450	Column 25	0	16-1-2020	16-1-2020	Finished
ST00070	Wall 4	0	16-1-2020	16-1-2020	Finished
ST00080	Wall 5	0	16-1-2020	16-1-2020	Finished
ST00110	Wall 6	0	16-1-2020	16-1-2020	Finished
ST00120	Wall 7	0	16-1-2020	17-1-2020	Finished
ST00090	Wall 8	0	17-1-2020	17-1-2020	Finished
ST00100	Wall 9	0	17-1-2020	17-1-2020	Finished
ST00130	Wall 10	0	17-1-2020	17-1-2020	Finished
ST00140	Wall 11	0	17-1-2020	17-1-2020	Finished

Figure 49: The exported file from the BIM 4D-tool (left) and the linked tasks with status (right)

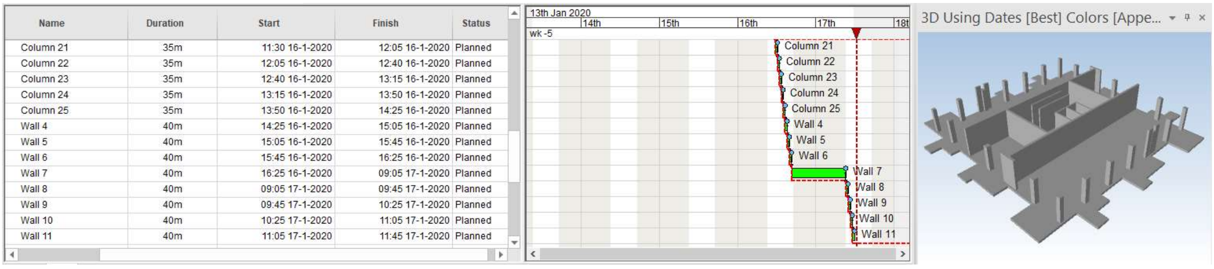


Figure 50: The state of the fictive building after reaching the end of the project

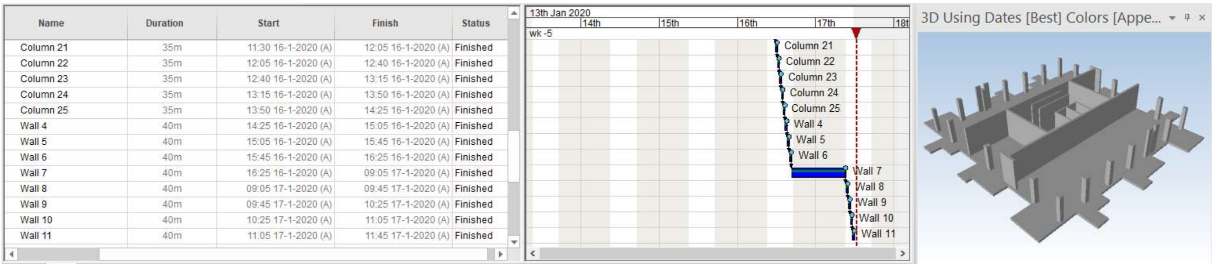


Figure 51: The current state of the fictive project after importing the updated .csv file

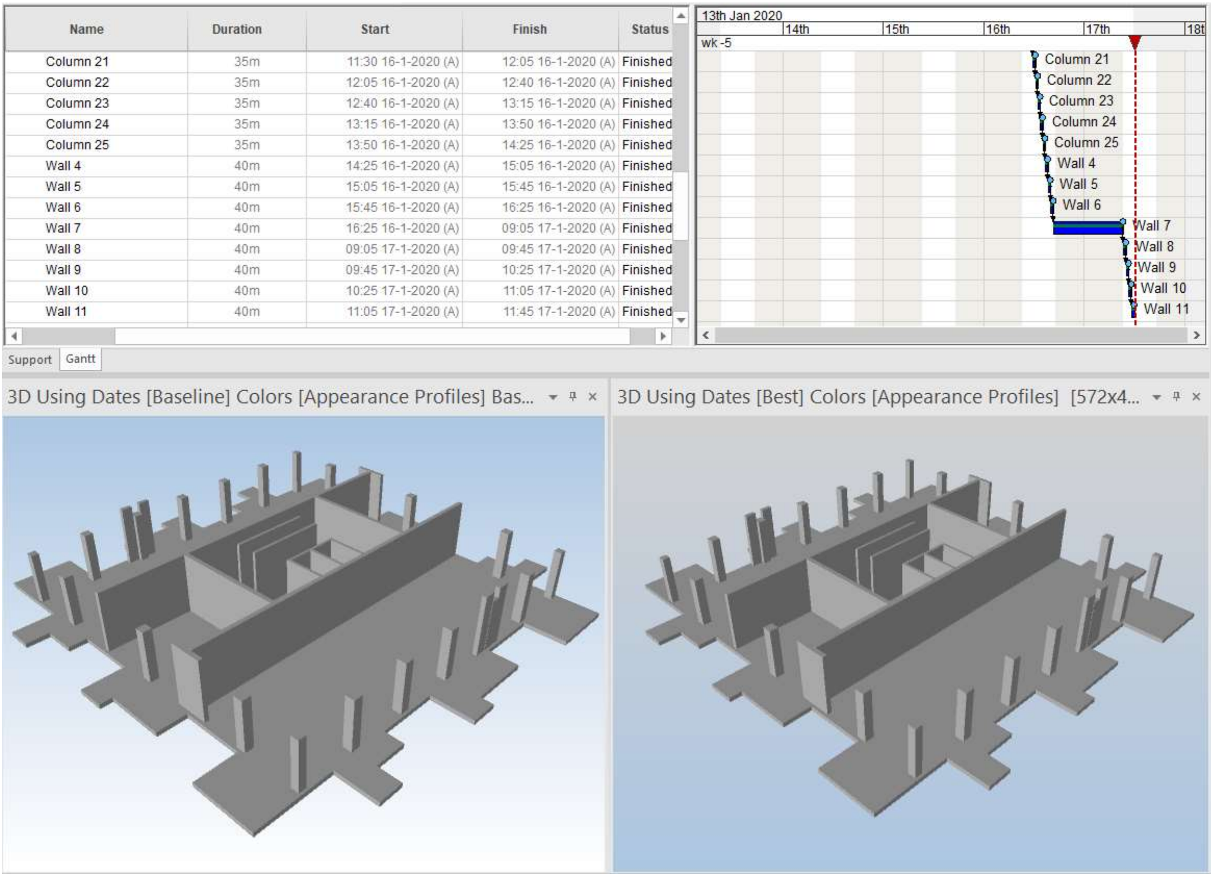


Figure 52: The planned state (left) and actual state (right)

5.4. The validation of the tool

In order to show the full capabilities of the Python tool, another scan date of the Trudo Tower is used, which is 16 January 2020. This is a scan which is a day before the initiated date which was discussed before. This paragraph only shows the results of the Python tool and does not go in detail about how the tool works, because this is already discussed. In addition, only the Analyse phase with the three steps are discussed below.

<u>Step 1: combining and cleaning data</u>

The 3D model and the point cloud data are combined together. After this, the point cloud data is cleaned. It took 1 minute and 10 seconds to clean it. The total number of points went from 7.282.377 to 748.767 points. This is a reduction of about 90%. Figure 53 shows the cleaned data.

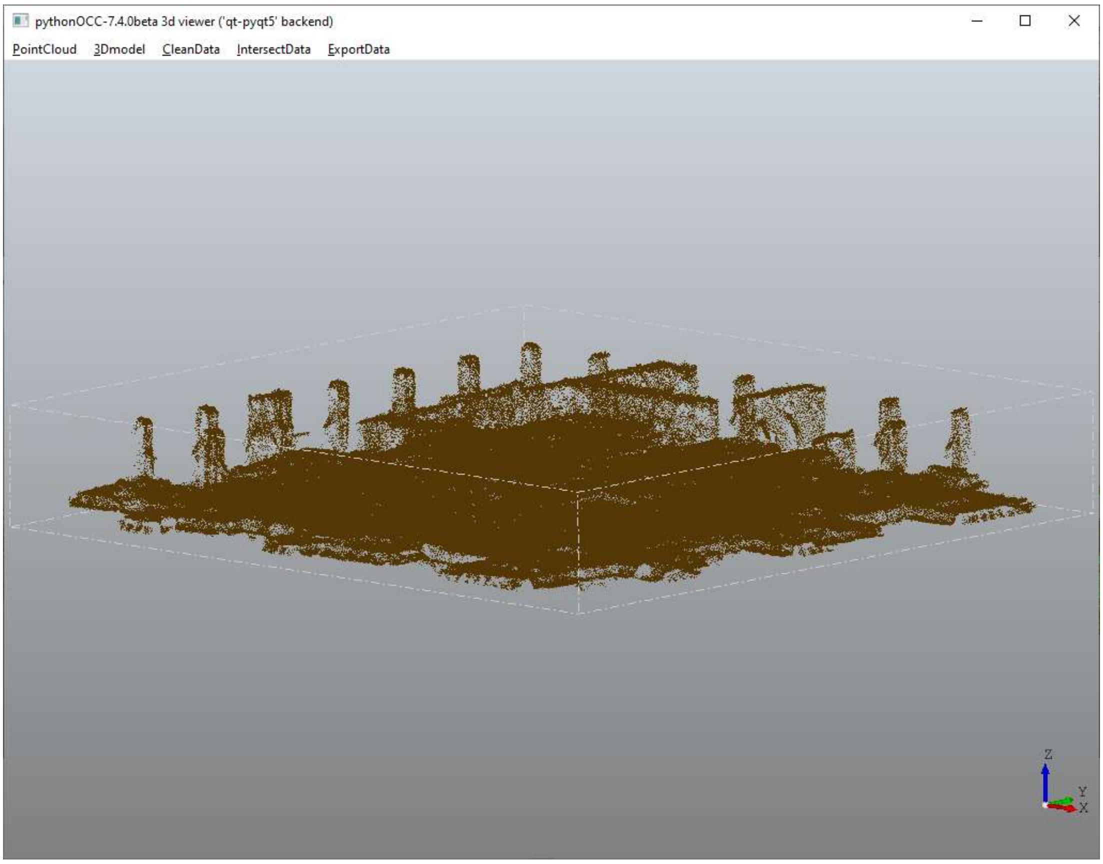


Figure 53: The point cloud data after cleaning

<u>Step 2: intersect data</u>

After the data is cleaned, the points can be intersected to the 3D model. Underneath, a summary is given of the output:

```
Out []:
Status:                  Object Names:
 False                    Basic Wall:Wall 1:6221_(#2151)
 True                     Basic Wall:Wall 2:7850_(#3829)
 False                    Basic Wall:Wall 2:6215_(#1790)
 False                    Basic Wall:Wall 1:6218_(#1983)
```

```
...                        ...
True                       Columns 13:Columns 2:7315_(#3139)
True                       Columns 18:Columns 1:7447_(#3474)
True                       Columns 19:Columns 1:6450_(#2334)
True                       Columns 17:Columns 1:7429_(#3407)
```

The result gave seven false objects, which is also shown in Figure 54.

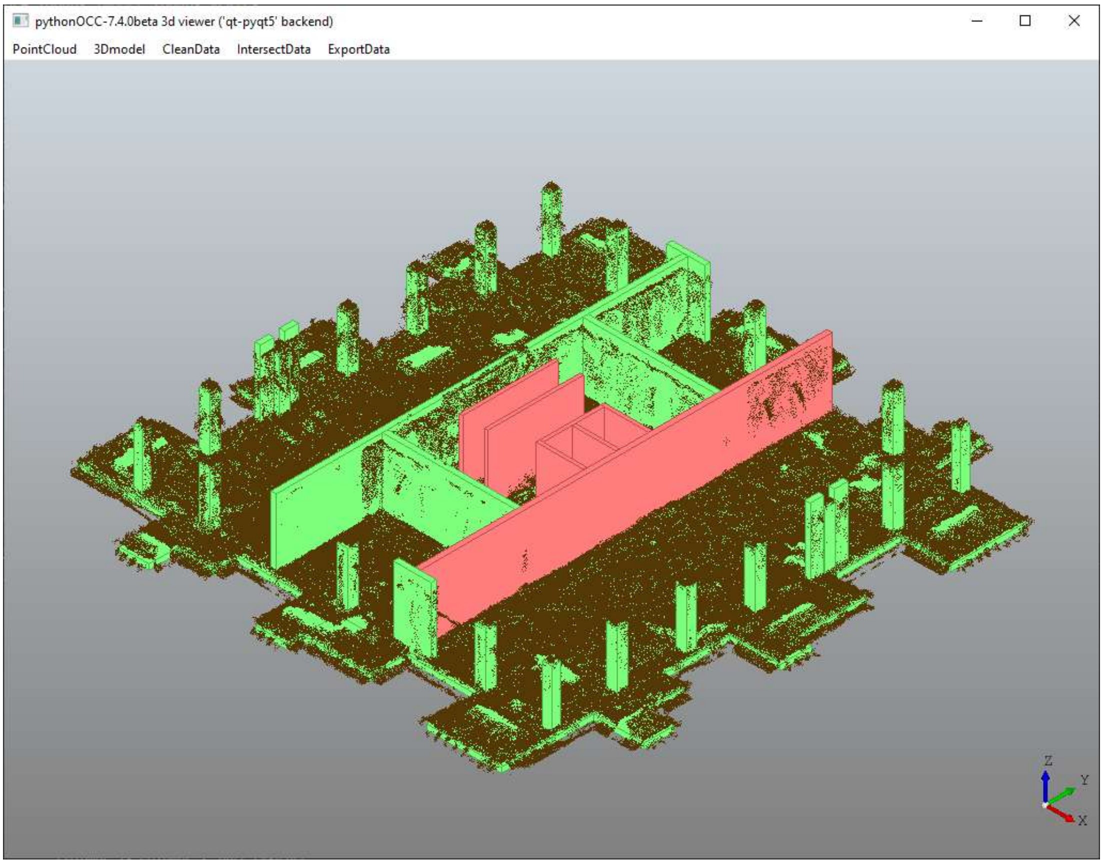


Figure 54: Coloured objects; green indicates that objects are intersected and red indicates that objects are not intersected

Even though the points are marginally less than the previous scan date, the results show that the prediction was 100% accurate. However, there is also a remark. The problem lies in how the 3D objects are created and in which level of detail it is modelled. As seen in Figure 55, two walls are highlighted. These two walls form one of the structural components of the building. These walls are modelled as a whole, instead of multiple parts. Number 1 shows that points around the wall are not dense. Although the wall is existing, the Python tool would probably indicate that the wall is not intersected if the 3D object was modelled in parts. Number 2 shows a part of another big wall. The Python tool did not intersect the wall because the points did not meet with the threshold. Overall, if the big walls were partially modelled, then the wall highlighted in number 1 could be not intersected and the wall highlighted in number 2 could be intersected. The threshold can, at all times, be changed manually to meet the actual situation.

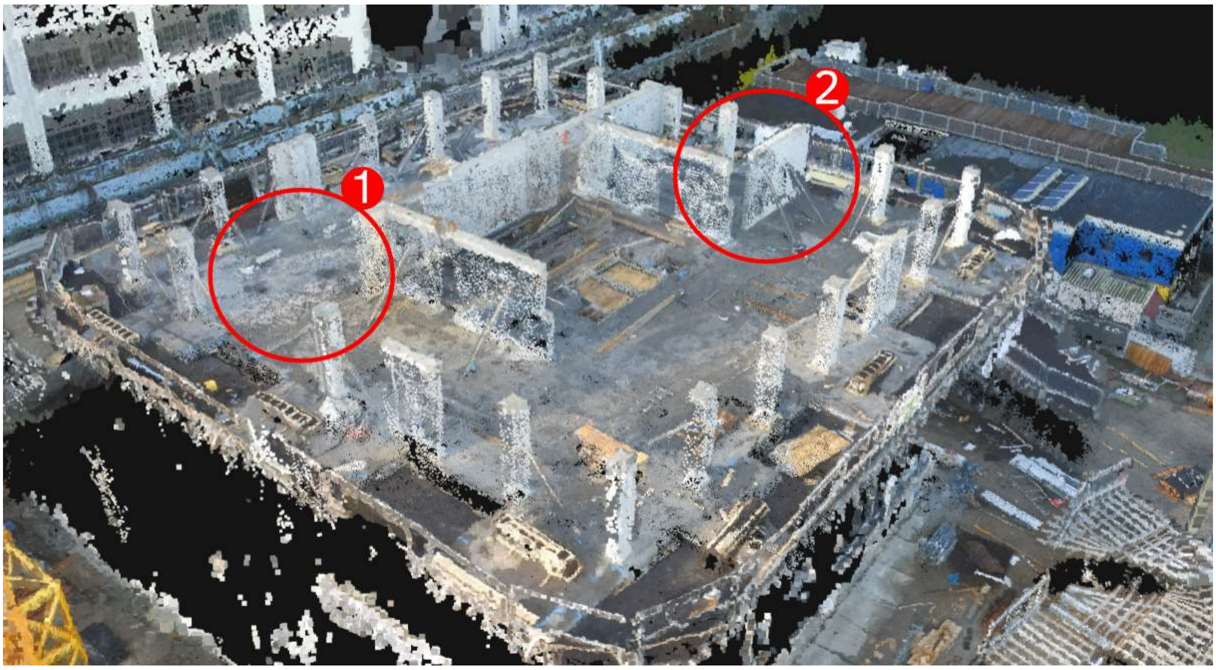
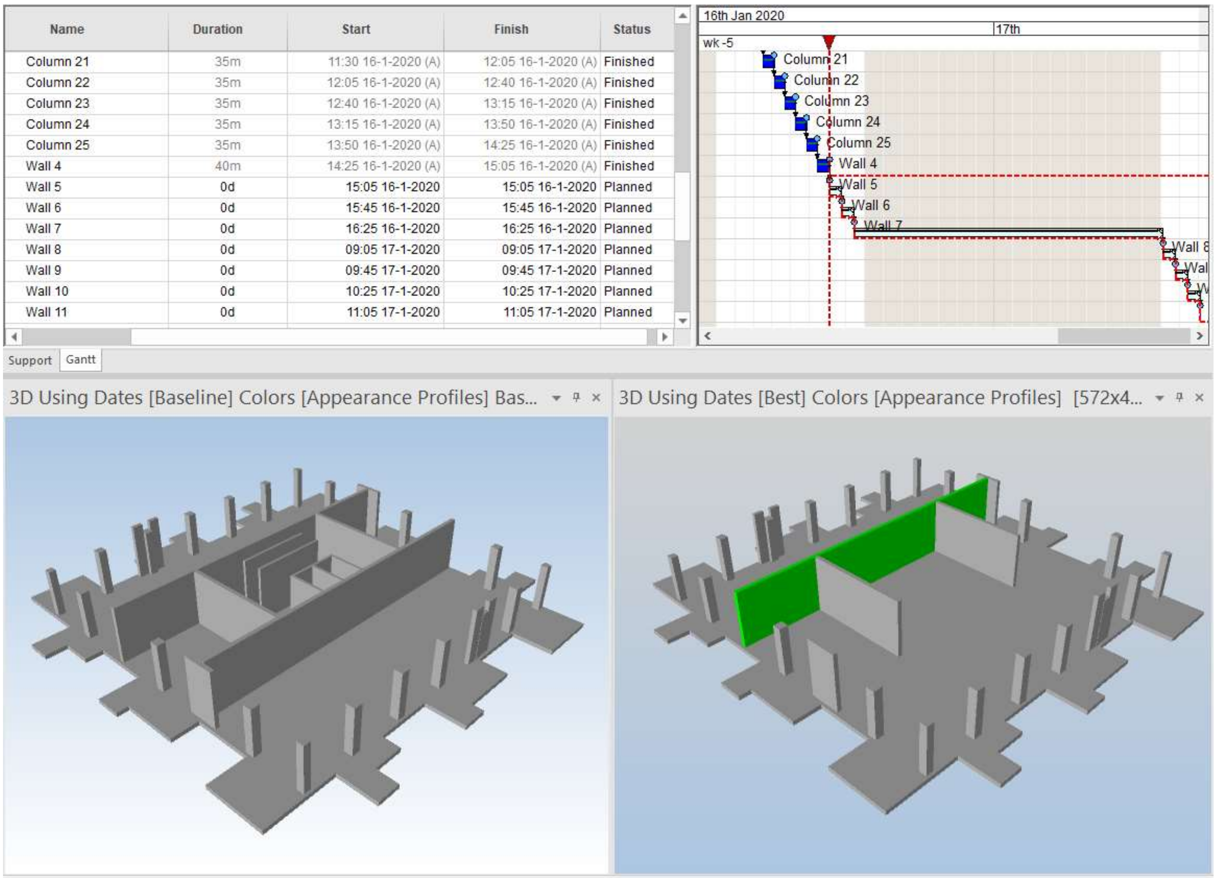


Figure 55: Another scan date of the Trudo Tower, where 1 has no dense points around the wall and 2 has dense points

Step 3: Exporting and importing data

To see which activities are finished and which are not, the intersected data can be imported in the BIM 4D-tool Synchro, see Figure 56. The left viewer shows the planned state and the right shows the actual state. This indicates that 'Wall 4' up to 'Wall 11' have deviated.



Name	Duration	Start	Finish	Status
Column 21	35m	11:30 16-1-2020 (A)	12:05 16-1-2020 (A)	Finished
Column 22	35m	12:05 16-1-2020 (A)	12:40 16-1-2020 (A)	Finished
Column 23	35m	12:40 16-1-2020 (A)	13:15 16-1-2020 (A)	Finished
Column 24	35m	13:15 16-1-2020 (A)	13:50 16-1-2020 (A)	Finished
Column 25	35m	13:50 16-1-2020 (A)	14:25 16-1-2020 (A)	Finished
Wall 4	40m	14:25 16-1-2020 (A)	15:05 16-1-2020 (A)	Finished
Wall 5	0d	15:05 16-1-2020	15:05 16-1-2020	Planned
Wall 6	0d	15:45 16-1-2020	15:45 16-1-2020	Planned
Wall 7	0d	16:25 16-1-2020	16:25 16-1-2020	Planned
Wall 8	0d	09:05 17-1-2020	09:05 17-1-2020	Planned
Wall 9	0d	09:45 17-1-2020	09:45 17-1-2020	Planned
Wall 10	0d	10:25 17-1-2020	10:25 17-1-2020	Planned
Wall 11	0d	11:05 17-1-2020	11:05 17-1-2020	Planned

Figure 56: The planned state (left) and actual state (right)

5.5. Conclusion case study

This chapter described the capabilities of the Python tool on a real construction site. The framework that is used is the same as the chapter before.

Different than the previous chapter, the point cloud data was generated by crane cameras. The advantage of this is that point clouds are generated automatically, because every 5 degree of rotation, the camera takes a picture and uploads it to the server of Pix4D. After the data set of pictures are sufficient, Pix4D generates a point cloud model. The disadvantage of the crane camera is that it cannot make a scan of the whole building. The storey under the initially constructed floor was during the case study not visible. This is because the angles are too sharp to be captured. The Python tool could, therefore, not intersect the construction objects situated under the floor. For that reason, only one floor that is under construction was imported in the tool. The results showed that the tool recognised all 3D objects. In this case, the accuracy rate is 100%. In order to validate the tool, another point cloud data from another date of the Trudo Tower was used. This showed that the accuracy was also 100% because it detected all 3D objects. A remark was that the 3D model should be detailed to some extent, so it reflects the actual construction. With that, some issues occurred about that the big walls were not modelled as constructed. In that sense, the 3D object of the wall was not divided into actual construction parts. It turned out that the Python tool is also usable for point clouds that are not dense. If the data is insufficient, there is always the option to reduce the threshold manually.

After importing the intersected data to the BIM 4D-tool Synchro, it gave an overview of which objects are deviated based on the planning. The scan data of 17 January 2020 gave no planning deviation. The scan data of 16 January 2020 gave a planning deviation. This is of course, not strange since there were construction elements to be installed on that day.

The BIM 4D-model was created at the macro level. This gave an excellent prospect about the Python tool. It is expected that the situation will slightly change when a BIM 4D-model is intersected on the micro-level. This is because first, there are relatively more objects to be processed, having in mind that equipment is also incorporated into the model. More 3D objects mean more faces to be calculated by the Python tool. In addition, the equipment models should only consist of six faces. More than that gives errors.

6. Conclusion

This chapter aims to finally answer the sub-questions, hypothesis and the main research question.

The hypothesis was:

If a tool is developed that measures the deviations between the BIM 4D-planning and the construction progress from point clouds and the deviations are automatically notified to the designated person(s), the construction planning can be optimized, and one of the potentials of BIM 4D can in that way be reached.

The main research question was:

How can a tool be developed in order to measure the deviations between the BIM 4D-planning and the construction progress from point clouds and automatically notify them to the designated person(s)?

In order to answer these questions, multiple sub-questions were formulated. These sub-questions were divided into two categories, namely literature study and practice study. The two categories with their associated sub-questions are addressed below.

Literature study

1. *What makes it so complex that the construction progress does not go as foreseen?*

In literature, construction projects are characterized as complex, which leads most of the time to time delays and cost overruns. This is because (1) construction projects are unique; (2) have a distinctive set of objectives; (3) improper construction planning; (4) construction projects are static and should be 'produced' on the spot; (5) expertise is needed from beginning to end: more people means more management; (6) depending on the size of the project, there is also a considerable amount of construction activities needed, leading to incomprehensive procedures.

2. *What is needed in order to complete a construction project successfully?*

To reduce and manage project complexity, and thus have a high success factor of the project, lean and agile management can be applied. Lean management is a principle to optimize construction processes and achieve more with less. Agile management aims to increase the relevance, quality, flexibility, and business value using software solutions. Lean management can be used for construction planning, and agile management can be used as a tool to generate 3D models and construction planning. Lean and agile management should be applied in harmony to have a high success factor for the project.

3. *How are construction activities traditionally managed in terms of project management?*

Construction activities are documented in construction planning. The planning shows on which date an activity should be executed and by whom. Construction activities must comply with the planning. When this is not happening, it leads to late project delivery, which has negative consequences. In order to manage this, the construction planner should formulate the planning in such a way that it is feasible. Also, during construction, the works should be monitored. Monitoring is the way of collecting, analysing, recording and reporting information about the progress of the construction activities. Monitoring of the work is done by superintendents and site managers. They should do this continuously, which makes the chance of detecting early or potential schedule delay high. Although continuously monitoring is favourable, it is characterized as time-consuming and costly.

4. *What is the main purpose of BIM and in which way can this improve construction projects?*

After the use of CAD, a new game changer has been introduced in the last decade that will improve productivity, cost-efficiency and sustainability of the construction industry, which is BIM. Besides BIM is characterized as a technology to generate 3D models, it is also a way of working, leading to a transformation of how construction projects are being designed, engineered, built and managed. BIM can be applied in n-amount of dimensions, and support many different facets in the construction industry. BIM 4D and BIM 5D are hereby the most well-known applications, where BIM 4D is the assigning of the construction planning, and BIM 5D is the assigning of the cost calculation. BIM is also narrowly aligned to the philosophy of lean management. The interaction of those two creates many opportunities for a more efficient workforce and effective processes of construction projects.

5. *What is the State of the Art of BIM 4D, and how can this improve construction projects even more?*

As already indicated, unique and large-scale projects are more complex. A construction planner should formulate their planning in such a way that it is comprehensive for different stakeholders. Sometimes, the mutual dependencies of the activities are such complex, leading to that the planning is not understandable anymore. Therefore, BIM 4D is applied. For generating a BIM 4D-model, three requirements are needed. First, a 3D geometric model of the project should be generated in the form of an IFC file. Second, a construction planning with activity names, activity duration and relationship between activities should be made. Lastly, a BIM 4D-tool is needed that allows to link 3D objects from the IFC data with the construction activities from the planning.

BIM 4D visualizes the devised planning of the planner and acts as a communication tool for stakeholders. In that way, the planning will be much more understandable, and possible bottlenecks can be spotted earlier. This will improve the planning, and thus also the efficiency and safety of the project.

At the moment, BIM 4D-models are made far after the execution phase has started, and is only used to make a fancy simulation. A lot can be done in order to achieve the full potential of BIM 4D. One of them is to track the actual construction progress and update the BIM 4D constantly. Tracking the activities and updating this in the BIM 4D-model, it provides efficient and timely data analysis and visualisation of deviations in an early stage to prevent potential delays and errors and enables timely decisions to take corrective actions by stakeholders.

6. *What are monitoring systems, and how can this support BIM 4D?*

In order to track the actual construction activities, it should be monitored. This is traditionally done by superintendents and site managers. Another way of monitoring should be explored, as the traditional way of working has been found inefficient. A possibility is the application of monitoring systems. There are different technologies available to (semi)automatically record the actual situation of the construction progress, which are: imagining, geospatial and radiofrequency technologies. Studies have shown that imagining technology suits the best for supporting BIM 4D. Imagining technologies is the collection of visual data, generated by photogrammetry, 3D laser scanners, videogrammetry, and range images.

7. *How can point clouds be an added value for BIM 4D?*

As many new technologies are being applied in different industries, the construction industry, however, is lagging behind. To keep on track of the latest technology, many studies are proposing how technology can be served in the construction industry. One of them is called Scan-vs-BIM. Through this process, it is possible to compare a scan from an imagining technology against the 3D BIM-model and may, therefore, help to update the BIM 4D-model. In addition, the output of a scan is a point cloud, which are points in the Euclidian space $\mathbb{R}3$, where each point has its own XYZ coordinates.

<u>Practice study</u>

1. *Which technological elements are needed that can measure the deviations between the BIM 4D-planning and point cloud?*

In order to measure the planning deviation(s), data is needed. This implies both IFC data and point cloud data. The elements that are needed to generate these data is a BIM 4D-tool and photogrammetry software. In this research, the BIM 4D-tool Synchro and the photogrammetry software Pix4D are used. Another tool is also needed that can read both data and is able to measure the differences between the data. As this is at the same time, a research gap, a tool is developed in Python. Python is able to quickly modify numerical data and simulate these through widely available packages. The package that is used in this research is called PythonOCC. This package is able to read, process, manipulate and view both IFC and point cloud data.

2. *How can point cloud data be linked to BIM-4D in order to measure deviations?*
    a. *How is it possible to transform a BIM 4D-model and point cloud model to a readable and structured data source in order to overlay both data and measure deviations?*

Point cloud data, which are generated by photogrammetric software, are dimensionless. In order to overlay this with the IFC data, the point cloud data should be scaled. To scale, a well-known distance should be visible in the point cloud data. This is done by target points. After the point cloud data is scaled, both point cloud and IFC data should be parsed. By parsing, the data is converted to a specific file format that is amenable for the Python tool. In that sense, the point cloud data is converted from a .ply file to a .pcd file, and the IFC data is converted to a .stp file. After both data are overlaid, the planning deviation(s) should be measured. First, the point cloud data is cleaned by creating normals per object face and making a space boundary around the 3D object. When points are outside this boundary, it will be removed. After reducing all irrelevant points, the 3D objects can be intersected with the reduced point cloud data. With this, for each face of 3D objects, a 3x3 grid structure is created. Each face has thus nine raster cells. Each raster cell is intersected with points and is calculated to determine whether this is above the threshold. If the number of recognised raster cells are above the specified percentage, the plane is recognised. When two or more planes of a specific 3D object are recognised, the whole 3D object is recognised. This means that the construction element is associated with the 3D object.

3. *How can the deviations between the BIM 4D-planning and the construction progress be notified to a designated person(s)?*

The tool indicated whether a 3D object is recognised with a 'True' value. When it is not intersected, it is given a 'False' value. As this is only an indication of how many 3D objects have deviated or not, it does not indicate which activity is included. The reason that the planning data from the BIM 4D-model is included in the Python tool is the fact that .stp files did not contain planning parameters. This is because during the conversion from .IFC to .stp, this data was lost. With planning parameters, it is possible to read which 3D object belongs to which construction activity. With a script that is developed, the 3D objects were associated with task names, and it could be seen which activity was intersected and which not. This information is exported to a .CSV file. In this file, an overview was given about the planned activities. Each activity also has a 'Status' option. The status of the activity can be 'Planned' or 'Finished'. Planned means that the construction activity had not started yet and finished means that the activity is complete. The Python tool changed the 'Status' option from planned to finished when

3D objects are intersected. After the information is filled in automatically by the Python tool, it is possible to import this in the BIM 4D-tool. The BIM 4D-tool Synchro synchronized the new information with the old planning data and showed which tasks were deviated in the 3D viewers and Gantt Chart. The visualisation of this acts as a notification for the user to take actions.

Now that the sub-questions are drawn, the hypothesis and main research question of this thesis can be answered.

BIM 4D is a method where the construction planning is being made visual. To actually make a BIM 4D-model, a 3D model and construction planning are needed. With these two data sets, it is possible to make a simulation of the construction project from beginning to end, based on construction planning. As BIM 4D is applied more and more in the construction sector, this however only acts as a one-time animation for commercial uses. According to literature, there is more to be achieved than just simulating the construction planning, because BIM 4D is also about communication, optimization, site logistics, trade coordination, safety planning and comparing schedules and track construction progress. The last one was covered in this research. This is because the most significant issue of BIM 4D-model is that it needs to be updated regularly to reflect the actual construction site conditions. This is time-consuming and labour intensive, which discourages the further use of BIM 4D.

In order to track construction progress, monitoring techniques can be used. The literature indicated that the full potential of capturing actual site conditions by monitoring techniques had not been achieved yet. This is because the sites are being captured manually. As a result, this capturing is very time consuming and must be operated by experts, and is therefore very expensive. In addition, literature indicated that a point cloud must be manually generated from the data that was captured, mostly via 3D laser scanners. The approach which is covered in this thesis was totally automatic by capturing the site via crane cameras in combination with the photogrammetric software Pix4D. The point cloud data that is generated by Pix4D can be used in the developed Python tool.

The developed tool has delivered an easy to use interface. On the top ribbon of the tool, there are functions created that lead the user to a stepwise process. This implies that the first function opens the 3D and point cloud data, the second function cleans the point cloud data, the third function intersects the points with the 3D data, and the last function exports the intersected data that can be imported in a BIM 4D-tool. As the tool is simple to use, no expertise is needed. This is necessary for practitioners who, most of the time, are not skilled in new and complicated tools. This tool provides them with the essential functions needed to update the BIM 4D-model.

The Python tool had an accuracy of 100%. This was shown by intersecting the 3D data with different point cloud data. However, it needs to be verified by practitioners to make sure everything is working correctly. When the accuracy is less, the threshold can always be adjusted within the tool.

The entire process of updating the BIM 4D-model was not fully automatic, because point cloud data had to be manually scaled to the right dimension, the base point should be determined, and the point cloud data had to be converted to a readable data source. These were the only handlings that needed to be done before using the developed Python tool. This research made it still possible to automate the process from combining 3D and point cloud data to export intersected 3D objects to .CSV, see Figure 57.

Lastly, one of the essences of BIM 4D can be reached, which is: to compare schedules and track construction progress. Although it is concluded that the developed tool is not fully automatic, it can serve superintendents, site managers, project planners, and project managers to compare project planning based on point clouds and identify whether the project is on track or behind planning. The whole process may help against large time-consumption and labour intensiveness, and therefore encourage practitioners to use BIM 4D more and more in the future.

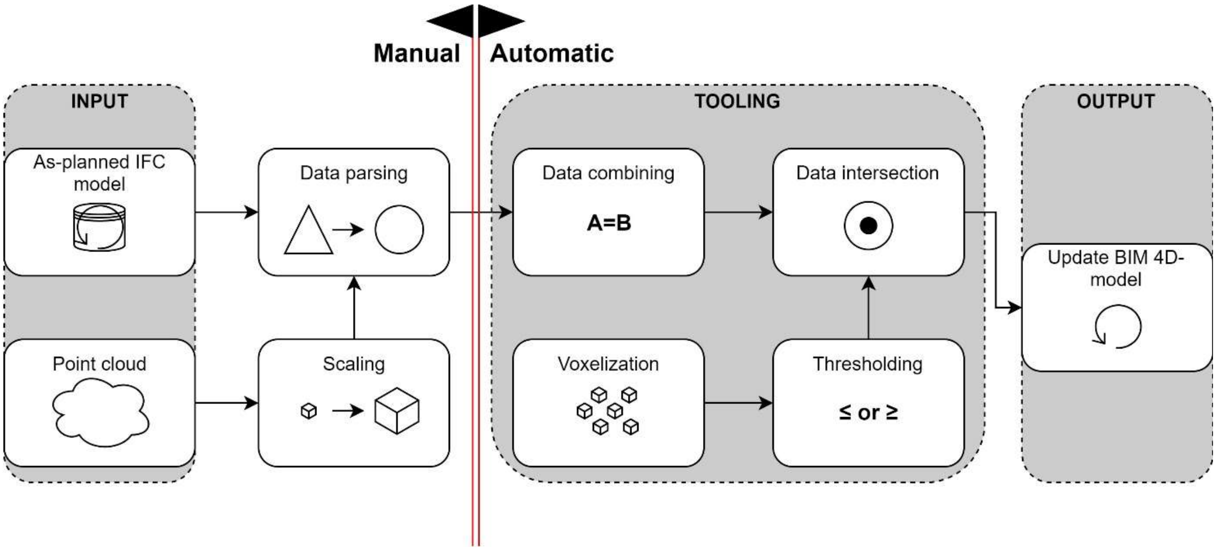


Figure 57:The line between manual and automated handlings

7. Discussion

In this thesis, there has been attempted to develop a tool that automatically updates the BIM 4D-planning with point cloud data. In this chapter, the issues that are surrounded by the research topic are discussed.

The focus of this research was on the situation where the construction activities are behind schedule. The empathy to research only that part is because falling behind schedule is an unfavourable event, leading to higher construction costs and also fines because of late project delivery. In this case, the tool that is now developed compared the BIM 4D-model against the point cloud data. Besides falling behind schedule, there are also cases where the construction project is ahead of schedule. In that case, the tool that is developed may be able to detect 3D objects. This is not tested yet.

This research also focused only on prefabricated construction elements in the structural construction phase. Regarding the construction planning, there were no pre-processing needed to put as a sub-task and associate this to a 3D object. This resulted in that the developed tool easily could tell which construction elements are placed and which are not. It becomes a lot more challenging when traditional construction techniques are used, such as in situ concrete walls. In order to realize this, pre-processings are needed, and therefore sub-tasks are needed in the planning. Unfortunately, it becomes really hard to visually show in the BIM 4D-model what these pre-processings are because 3D objects are needed for this. In many times, however, the level of detail of the BIM models are not sufficient enough to cater to this. At the moment, the 3D objects in the BIM 4D-models have a one-to-many relationship with planning activities. For the tool, it means that, in this situation, there is no 3D object for a specific sub-task, thus no intersection, so no measurement for the planning deviation. Therefore, for the traditional construction techniques, the tool cannot give an accurate measurement in comparison with prefabricated construction technique.

It is also necessary to consider the equipment that is used on the construction site for the construction of building elements. In all times, the placing and removing of the equipment are included in the construction planning. This also should be visualized in the BIM 4D-model, as this is a really important task. However, in most cases, the BIM model does not include such a 3D model of the equipment. It is the essence to know beforehand what the purpose of the application of BIM 4D is. When it is only a visualisation, which shows in a short time the summary of the whole construction process, then the BIM 4D-model should be performed on the macro level. When it is needed to go in detail and seek for potential bottlenecks, then the BIM 4D-model should be performed on the micro level. Within the micro level, all sub-tasks must be modelled in 3D and be put in the BIM 4D-simulation.

This research was also about the outdoor environment of the building site, where the structural construction was visible during the structural construction phase. The question might also be what happens when this phase is finished, and the crane is disassembled. In that case, it means that there would not be a point cloud generated from the crane camera and another imagining methodology or technology must be used. For the tool itself, it does not matter how the point cloud data is obtained, as long as it is a dense point cloud data.

For the tool, tolerance is added for construction elements that are not exactly built conform to the 3D model. This feature is added in order to take into account that in most cases, the geometrical aspects of a construction element are not always diametrically opposed, compared with a 3D model. The objects in the 3D model are always perpendicular. The tolerance that is added is, to a certain extent. When the point cloud data is outside this tolerance, the tool cannot recognise the 3D object. This also applies to wrongly placed construction elements.

A dense point cloud data is characterized as a data where points around all construction elements are visible. The point cloud data of the fictive pilot project resulted that all blocks were visible. However, this was small-scaled, and there was the opportunity to take pictures in every corner of the scaled model. With the real case study of the Trudo Tower, the point cloud data was at some locations not dense, leading to the possibility that the tool could give false indications. There are multiple reasons that the point cloud data is not dense in this situation. First, because of the fact that the point cloud is generated via a crane camera, this is only directed in one direction. Compared with the small case study, generating point cloud data via a crane camera is thus a disadvantage, because it is not possible to freely move around the building. Second, the weather conditions on the construction site are sometimes not optimal. This leads to that shadows are in the way, preventing the algorithm of Pix4D from overlapping pictures with each other. Also, the operation of the crane is dependent on the weather. When the weather condition is bad, there is the possibility that the crane may not operate because of safety regulations. No crane operation means no point cloud data. The third reason is that on the construction site, there are except construction elements, also materials and equipment to make the actual construction elements. Sometimes, these block the visual sight of the crane camera, leading to inaccurate point cloud data. This phenomenon is also called occlusion. In order to avoid false indications, the threshold can be changed manually in the Python tool.

In the case study, it showed that storeys under the initial floor that is constructed could not be seen. This has to do with the fact that the crane camera has a sight to a certain extend. When angles are too sharp, there will not be a point cloud generated at that specific area. As one floor was being made in BIM 4D-model, this was not an issue. Supposing that every building storey is finished per three weeks, the Python tool should be used regularly (not exceeding around three days) to intersect all building elements.

For this research, the online version of Pix4D was used. Their actual software that can be downloaded from their site; however, called Pix4D Desktop, promises that point clouds can also be exported as a .ply file. In that way, another tool (CloudCompare) is not needed to convert .LAS to .ply and only the developed script can be used to export .ply to .pcd. The scaling of the point cloud data should also be taken into consideration because now it must be done manually. The new version of Pix4D will provide automatic scaling by using the fixed ground control points on the construction site as the well-known distance to scale with. Also, the base points of both IFC and point cloud data must be located at the GPS location based on National Triangle Coordinates (Rijksdriehoekcoördinaten). For the Netherlands, it means that the location must be set at Amersfoort EPSG:28992. After the point cloud data is converted to a .pcd file and both IFC and point cloud data are placed at the same GPS coordination, it is possible to update the BIM 4D-model without manual handlings outside the developed tool. In that way, the whole process of updating the BIM 4D-model can be made fully automatic from beginning to end.

Usually it is possible to combine PythonOCC with IfcOpenShell. In this way, IFC files can be read in a pythonic way and be viewed in PythonOCC. For this research, a newer PythonOCC version (7.4) is installed so point clouds can be viewed, which was not possible with the older version. As it is required to have python packages that are suitable for the same python version, it was not possible to combine PythonOCC with IfcOpenShell. That was also the reason that IFC files were converted outside the Python tool to a data file that can be read by PythonOCC, which was, in this case, a step file. In order to streamline this process, IfcOpenShell must be updated to the same version as PythonOCC, so it is not needed to convert IFC files anymore.

The Python tool only recognises 3D objects that have six faces. It turned out that the 3D model in the case study had more faces than six. When executing the tool, it gave errors. Due to time constraints, the script in the Python tool is not further developed, but rather the 3D model is remodelled in rectangular objects. In that way, it was possible for the tool to recognise 3D objects. When using this tool in practice, it is inefficient, because the whole 3D model should be remodelled, and a new BIM 4D-model should be made.

The framework that is used for this research conforms to the PDCA cycle. The basis of this consisted of three phases, namely, plan, capture, and analyse phase. Each phase had a desired output, which was beneficial for the following phase. Structuring the phases in a consecutive way, it turned out that the whole process went systematic. There were no obstacles to be found with the proposed framework.

7.1. Recommendations for further research

This paragraph provides recommendations for further research.

- This research was divided into three phases, based on the PDCA cycle of Van Schaijk (2016). Within this cycle, there is also the fourth phase, which is the reuse phase. Collecting actual planning data can be beneficial in the future.
- The Python tool now only can read 3D objects with six faces. As IFC models are generic, there are objects that contain more than six faces. For that reason, the Python tool should be optimized, so 3D models do not have to be remodelled again. It can be looked to calculating only six major faces of an object and calculate deviations from that basis.
- This research concerned the outdoor environment. This Python tool should also be tested on indoor environments, including another imagining technology.
- There can be looked to object recognition in terms of machine/deep learning and the added value of it.
- Because the fact that one 3D object can represent multiple tasks, it is for the tool hard to know on which state the construction element is. As this study looked into if an element is placed (yes or no), for further research, it is useful to research how this can be detailed (in micro level). This also includes temporary construction elements and equipment.
- The formulated equation (see below) is able to provide the exact percentage of the 3D object. In this research, there was a threshold included to indicate if a bounding box is contained with points (yes or no). When there is a BIM 4D-planning made at the micro level, then it is possible to define the real percentage of a certain task instead of yes or no. This is useful for structures that are large and take longer to complete, such as brickwork.

  $$P_i = \frac{\sum Voxels_{c,i}}{Voxels_{a,i}} \cdot 100\%$$

- Point clouds are acquired by sensing technologies like 3D laser scanners and photogrammetry. However, for both of the applications, the acquisition is made in a different way. It should be looked if other sensing technology then photogrammetry can be an added value.
- It is also worth noting that this research was only focused on the situation where the construction is behind schedule. In this sense, the BIM 4D-model was compared against the point cloud data. The question is also what happens when the construction is ahead of schedule?
- The point clouds that are generated by Pix4D contain only one storey. The problem here is that the underlying storey cannot be seen by the Python tool. For that reason, it is a great value to include all point clouds from different scan dates together. This may be done via Iterative Closest Point (ICP) algorithm. In addition, the Python tool is not bound to the Pix4D software. There should be looked to open-source software.

- There has been found that there are several obstacles that withheld the full automation of updating the BIM 4D-model. How the whole process can be made fully automatic is an interesting topic for further research.

References

Abbas, A., Din, Z. U., & Farooqui, R. (2016). Achieving Greater Project Success & Profitability through Pre-construction Planning: A Case-based Study. *Procedia Engineering*. https://doi.org/10.1016/j.proeng.2016.04.105

Ahuja, R., Sawhney, A., & Arif, M. (2017). Driving lean and green project outcomes using BIM: A qualitative comparative analysis. *International Journal of Sustainable Built Environment*, *6*(1), 69–80. https://doi.org/10.1016/j.ijsbe.2016.10.006

Aibinu, A. A., & Jagboro, G. O. (2002). The effects of construction delays on project delivery in Nigerian construction industry. International journal of project management, 20(8), 593-599.

Akanbi, O. A., Oyedolapo, O., & Steven, G. J. (2019). Lean Principles in Construction. *Sustainable Construction Technologies*, 317–348. https://doi.org/10.1016/B978-0-12-811749-1.00010-9

Akinsola AO. A Neutral network model for predicting Building projects' Contingency. Conference: Proceedings of Association of Researchers in Construction Management, ARCOM 96, Sheffield Hallam University, England, 11–13 September 1996, pp. 507–516.

Baccarini, D. (1996). The concept of project complexity—a review. International journal of project management, 14(4), 201-204.

Behnam, A., Wickramasinghe, D. C., Ghaffar, M. A. A., Vu, T. T., Tang, Y. H., & Isa, H. B. M. (2016). Automated progress monitoring system for linear infrastructure projects using satellite remote sensing. *Automation in Construction*, *68*, 114–127. https://doi.org/10.1016/j.autcon.2016.05.002

Bonduel, M., Bassier, M., Vergauwen, M., Pauwels, P., & Klein, R. (2017). SCAN-TO-BIM OUTPUT VALIDATION: TOWARDS A STANDARDIZED GEOMETRICQUALITY ASSESSMENT OF BUILDING INFORMATION MODELS BASED ON POINTCLOUDS. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *XLII-2/W8*, 45–52. https://doi.org/10.5194/isprs-archives-xlii-2-w8-45-2017

Bosche, F., Haas, C. T., & Akinci, B. (2009). Automated Recognition of 3D CAD Objects in Site Laser Scans for Project 3D Status Visualisation and Performance Control. *Journal of Computing in Civil Engineering*, *23*(6), 311–318. https://doi.org/10.1061/(ASCE)0887-3801(2009)23:6(311)

Browaeys, M., & Baets, W. (2003). Cultural complexity: a new epistemological perspective. The Learning Organization, 10(6), 332–339. https://doi.org/10.1108/09696470310497168

Bueno, M., Bosché, F., González-Jorge, H., Martínez-Sánchez, J., & Arias, P. (2018). *4-Plane congruent sets for automatic registration of as-is 3D point clouds with 3D BIM models. Automation in Construction, 89, 120–134. https://doi.org/10.1016/j.autcon.2018.01.014.*

Camacho, A., Cañizares, P. C., Estévez, S., & Núñez, M. (2018). A tool-supported framework for work planning on construction sites based on constraint programming. *Automation in Construction*, *86*, 190–198. https://doi.org/10.1016/j.autcon.2017.11.008

Carmona, J., and Irwin, K. (2007). "BIM: Who, what, how and why." Building Operating Management. ⟨http://www.facilitiesnet.com/software/article/ BIM-Who-What-How-and-Why–7546⟩ (August 21, 2009).

Chua, C. K., Wong, C. H., & Yeong, W. Y. (2017). Software and Data Format. *Standards, Quality Control, and Measurement Sciences in 3D Printing and Additive Manufacturing*, 75–94. https://doi.org/10.1016/B978-0-12-813489-4.00004-0

Ciribini, A. L. C., Mastrolembo Ventura, S., & Paneroni, M. (2016). Implementation of an interoperable process to optimise design and construction phases of a residential building: A BIM Pilot Project. *Automation in Construction*, *71*, 62–73. https://doi.org/10.1016/j.autcon.2016.03.005

de Soto, B. G., Rosarius, A., Rieger, J., Chen, Q., & Adey, B. T. (2017). Using a Tabu-search Algorithm and 4D Models to Improve Construction Project Schedules. *Procedia Engineering*, *196*, 698–705. https://doi.org/10.1016/j.proeng.2017.07.236

DeWitt, B. A., Wolf, P. R., & Wilkinson, B. E. (2014). *Elements of Photogrammetry with Application in GIS, Fourth Edition*. Retrieved from https://www.accessengineeringlibrary.com/content/book/9780071761123

Eadie, R., Browne, M., Odeyinka, H., McKeown, C., & McNiff, S. (2013). BIM implementation throughout the UK construction project lifecycle: An analysis. *Automation in Construction*, *36*, 145–151. https://doi.org/10.1016/j.autcon.2013.09.001

Eastman, C., Teicholz, P., Sacks, R., and Liston, K. (2008). BIM handbook: A guide to building information modelling for owners, managers, designers, engineers and contractors, Wiley, New York. Eastman, Charles M. *BIM handbook : a guide to building information modeling for owners, managers, designers, engineers and contractors*. Hoboken, NJ: Wiley, 2011. Print.

Fadeyi, M. O. (2017). The role of building information modeling (BIM) in delivering the sustainable building value. *International Journal of Sustainable Built Environment*, *6*(2), 711–722. https://doi.org/10.1016/j.ijsbe.2017.08.003

G.S. Cheok, W.C. Stone, R.R. Lipman, C. Witzgall, LADARs for construction assessment and update, Automation in Construction 9 (5) (2000) 463–477

Gao, T., Akinci, B., Ergan, S., Garrett, J. (2015). An approach to combine progressively captured point clouds for BIM update. Journal of Advanced Engineering Automatics 29, 1001-1012

Gebhardt, S., Fettinger, A., Rotenberg, E., & Seher, C. (n.d.). *Polygons, Point-Clouds, and Voxels, a Comparison of High-Fidelity Terrain Representations*. Retrieved from https://www.sisostds.org/DesktopModules/Bring2mind/DMX/API/Entries/Download?Command=Core_Download&EntryId=28745&PortalId=0&TabId=105

Genelot, D. (1998), Manager dans la Complexité, Réflexions à l'Usage des Dirigeants, 2nd ed., INSEP Éditions, Paris.

Golparvar-Fard, M., Peña-Mora, F., & Savarese, S. (2011). Integrated Sequential As-Built and As-Planned Representation with D4AR Tools in Support of Decision-Making Tasks in the AEC/FM Industry. *Journal of Construction Engineering and Management*, *137*(12), 1099–1116. https://doi.org/10.1061/(ASCE)CO.1943-7862.0000371

Golparvar-Fard, M., Pena-Mora, F., & Savarese, S. (2012). Automated Progress Monitoring Using Unordered Daily Construction Photographs and IFC-Based Building Information Models. Journal of Computing in Civil Engineering, 04014025.

Golparvar-Fard, M., Peña-Mora, F., Savarese, S. (2009). Application of D4AR - A 4-Dimensional augmented reality model for automating construction progress monitoring data collection, processing and communication, ITcon Vol. 14, Special issue *Next Generation Construction IT: Technology Foresight, Future Studies, Roadmapping, and Scenario Planning*, pg. 129-153, http://www.itcon.org/2009/13

Guarato, A. Z., Quinsat, Y., Mehdi-Souzani, C., Lartigue, C., & Sura, E. (2017). Conversion of 3D scanned point cloud into a voxel-based representation for crankshaft mass balancing. *The International Journal of Advanced Manufacturing Technology*, *95*(1–4), 1315–1324. https://doi.org/10.1007/s00170-017-1319-5

Hamledari, H., McCabe, B., & Davari, S. (2017b). Automated computer vision-based detection of components of under-construction indoor partitions. *Automation in Construction*, *74*, 78–94. https://doi.org/10.1016/j.autcon.2016.11.009

Hamledari, H., McCabe, B., Davari, S., Shahi, A., 2017, Automated Schedule and Progress Updating of IFC-Based 4D BIMs. Journal of Computing in Civil Engineering 31, 1-16.

Han, K. K., & Golparvar-Fard, M. (2015). Appearance-based material classification for monitoring of operation-level construction progress using 4D BIM and site photologs. *Automation in Construction*, *53*, 44–57. https://doi.org/10.1016/j.autcon.2015.02.007

Hartmann, T., Gao, J., and Fischer, M. (2008). "Areas of application for 3D and 4D models on construction projects." J. Constr. Eng. Manage., 10 .1061/(ASCE)0733-9364(2008)134:10(776), 776–785

Hassanien Serror, M., Inoue, J., Adachi, Y., & Fujino, Y. (2008). Shared computer-aided structural design model for construction industry (infrastructure). *Computer-Aided Design*, *40*(7), 778–788. https://doi.org/10.1016/j.cad.2007.07.003

Heesom, D., & Mahdjoubi, L. (2004). Trends of 4D CAD applications for construction planning. *Construction Management and Economics*, *22*(2), 171–182. https://doi.org/10.1080/0144619042000201376

Heigermoser, D., García de Soto, B., Abbott, E. L. S., & Chua, D. K. H. (2019). BIM-based Last Planner System tool for improving construction project management. *Automation in Construction*, *104*, 246–254. https://doi.org/10.1016/j.autcon.2019.03.019

Hinks, T., Carr, H., Truong-Hong, L., & Laefer, D. F. (2013). Point Cloud Data Conversion into Solid Models via Point-Based Voxelization. *Journal of Surveying Engineering*, *139*(2), 72–83. https://doi.org/10.1061/(asce)su.1943-5428.0000097

Hossen, M. M., Kang, S., & Kim, J. (2015). Construction schedule delay risk assessment by using combined AHP-RII methodology for an international NPP project. *Nuclear Engineering and Technology*, *47*(3), 362–379. https://doi.org/10.1016/j.net.2014.12.019

Hua, G. B. (2009). Construction and Economic Development of Four Most Competitive Economies in the World: A Comparison. *International Journal of Construction Education and Research*, *5*(4), 261–275. https://doi.org/10.1080/15578770903355590.

I. Brilakis, M. Lourakis, R. Sacks, S. Savarese, S. Christodoulou, J. Teizer, A. Makhmalbaf, Toward automated generation of parametric BIMs based on hybrid video and laser scanning data, Advanced Engineering Informatics 24 (4) (2010) 456–465.

Issa, R., Flood, I., and O'Brien, W. (2005). 4D CAD and visualisation in construction: Developments and applications, Taylor & Francis, Abingdon, VA.

J. Abeid, D. Arditi, Photo-net: an integrated system for controlling construction progress, Engineering, Construction and Architectural Management 10 (3) (2003) 162–171.

Jupp, J. (2017). 4D BIM for Environmental Planning and Management. *Procedia Engineering*, *180*, 190–201. https://doi.org/10.1016/j.proeng.2017.04.178

Kamel, E., & Memari, A. M. (2019). Review of BIM's application in energy simulation: Tools, issues, and solutions. *Automation in Construction*, *97*, 164–180. https://doi.org/10.1016/j.autcon.2018.11.008

Kattenborn, T., Sperlich, M., Bataua, K., & Koch, B. (2014). Automatic Single Tree Detection in Plantations using UAV-based Photogrammetric Point clouds. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *XL–3*, 139–144. https://doi.org/10.5194/isprsarchives-XL-3-139-2014

Khodeir, L. M., & Othman, R. (2018). Examining the interaction between lean and sustainability principles in the management process of AEC industry. *Ain Shams Engineering Journal*, *9*(4), 1627–1634. https://doi.org/10.1016/j.asej.2016.12.005

Kopsida, M., Brilakis, I., and Vela, P. A. (2015). "A review of automated construction progress monitoring and inspection methods." Proc., 32nd CIB W78 Conf., Endhoven, Netherlands, 421–431.

Krijnen, Thomas, & Beetz, J. (2017). An IFC schema extension and binary serialization format to efficiently integrate point cloud data into building models. *Advanced Engineering Informatics*, *33*, 473–490. https://doi.org/10.1016/j.aei.2017.03.008

Kumar, P. P. (2005). Effective Use of Gantt Chart for Managing Large Scale Projects. *Cost Engineering*, *47*(7), 14–21. Retrieved from https://search.proquest.com/docview/220442584/fulltextPDF/C58406A58B7E446CPQ/1?accountid=27128

Lissack, M.R. (1999), "Complexity: the science, its vocabulary and its relations to organizations", Emergence, Vol. 1 No. 1, pp. 110-26.

Liu, Y., Van Nederveen, S., & Hertogh, M. (2017). Understanding effects of BIM on collaborative design and construction: An empirical study in China. International Journal of Project Management, 35(4), 686–698. https://doi.org/10.1016/j.ijproman.2016.06.007.

Lopes, J., Ruddock, L., & Ribeiro, F. L. (2002). Investment in construction and economic growth in developing countries. *Building Research & Information*, *30*(3), 152–159. https://doi.org/10.1080/09613210110114028

Lopez, R., Chong, H.-Y., Wang, X., & Graham, J. (2016b). Technical Review: Analysis and Appraisal of Four-Dimensional Building Information Modeling Usability in Construction and Engineering Projects. *Journal of Construction Engineering and Management*, *142*(5). https://doi.org/10.1061/(ASCE)CO.1943-7862.0001094

Malik, N., Ahmad, R., & Al-Hussein, M. (2019). Generation of safe tool-paths for automatic manufacturing of light gauge steel panels in residential construction. *Automation in Construction*, *98*, 46–60. https://doi.org/10.1016/j.autcon.2018.11.023

Meredith, J. R., & Mantel, S. J. (2008). *Project Management: A Managerial Approach*. United States of America: Wiley.

Micheletti, N., Chandler, J. H., & Lane, S. N. (2014). Investigating the geomorphological potential of freely available and accessible structure-from-motion photogrammetry using a smartphone. *Earth Surface Processes and Landforms*, *40*(4), 473–486. https://doi.org/10.1002/esp.3648

Micheletti, N., Chandler, J.H. and Lane, S.N., 2015. Structure from motion (SFM) photogrammetry. IN: Clarke, L.E. and Nield, J.M. (Eds.) Geomorphological Techniques (Online Edition). London: British Society for Geomorphology. ISSN: 2047-0371, Chap. 2, Sec. 2.2. http://www.geomorphology.org.uk/sites/default/files/geom_tech_chapters/2.2.2_sfm.pdf

Mitchell, M. (2009). Complexity : a guided tour. Oxford, Oxford University Press.

Mlambo, R., Woodhouse, I., Gerard, F., & Anderson, K. (2017). Structure from Motion (SfM) Photogrammetry with Drone Data: A Low Cost Method for Monitoring Greenhouse Gas Emissions from Forests in Developing Countries. *Forests*, *8*(3), 68. https://doi.org/10.3390/f8030068

Morin, E. (1986), La Méthode, Tome 3: La Connaissance de la Connaissance, Editions du Seuil, Paris

Morin, E. (1990), Science avec Conscience, Nouvelle Édition Sciences, Coll. Points, Editions du Seuil, Paris, originally published in 1982

Mukuka, M., Aigbavboa, C., & Thwala, W. (2015). Effects of Construction Projects Schedule Overruns: A Case of the Gauteng Province, South Africa. *Procedia Manufacturing*, *3*, 1690–1695. https://doi.org/10.1016/j.promfg.2015.07.989

Navon, R., & Sacks, R. (2007). Assessing research issues in Automated Project Performance Control (APPC). *Automation in Construction*, *16*(4), 474–484. https://doi.org/10.1016/j.autcon.2006.08.001

Nourian, P., Gonçalves, R., Zlatanova, S., Ohori, K. A., & Vu Vo, A. (2016). Voxelization algorithms for geospatial applications. *MethodsX*, *3*, 69–86. https://doi.org/10.1016/j.mex.2016.01.001

Ofori , G. (1990). *The construction industry: Aspects of its economics and management* . Singapore : Singapore University Press, National University of Singapore.

Oh, M., Lee, J., Hong, S. W., & Jeong, Y. (2015). Integrated system for BIM-based collaborative design. Automation in Construction, 58, 196–206. https://doi.org/10.1016/j.autcon.2015.07.015.

olde Scholtenhuis, L. L., Hartmann, T., & Dorée, A. G. (2016). 4D CAD Based Method for Supporting Coordination of Urban Subsurface Utility Projects. *Automation in Construction*, *62*, 66–77. https://doi.org/10.1016/j.autcon.2015.10.013

Omar, T., & Nehdi, M. L. (2016). Data acquisition technologies for construction progress tracking. *Automation in Construction*, *70*, 143–155. https://doi.org/10.1016/j.autcon.2016.06.016

Owen , R. L., & Koskela, L. (2006). AGILE CONSTRUCTION PROJECT MANAGEMENT. *3rd International Built and Human Environment Research Week*, 22–33. Retrieved from https://www.irbnet.de/daten/iconda/CIB9021.pdf

Park, H., & Lee, D. (2019). Comparison between point cloud and mesh models using images from an unmanned aerial vehicle. *Measurement*, *138*, 461–466. https://doi.org/10.1016/j.measurement.2019.02.023

Pörteners, G. (2018). Towards the automation of semantically enriching point cloud data sets (master thesis). Retrieved from https://www.ofcoursecme.nl/?mdocs-file=3977

Poux, F., & Billen, R. (2019). Voxel-based 3D Point Cloud Semantic Segmentation: Unsupervised Geometric and Relationship Featuring vs Deep Learning Methods. *ISPRS International Journal of Geo-Information*, *8*(5), 213. https://doi.org/10.3390/ijgi8050213

Rebolj, D., Babič, N. Č., Magdič, A., Podbreznik, P., & Pšunder, M. (2008). Automated construction activity monitoring system. Advanced Engineering Informatics, 22(4), 493-503.

Richter, R., Discher, S., & Döllner, J. (2014). Out-of-Core Visualisation of Classified 3D Point Clouds. *Lecture Notes in Geoinformation and Cartography*, 227–242. https://doi.org/10.1007/978-3-319-12181-9_14

Sambasivan, M., & Soon, Y. W. (2007). Causes and effects of delays in Malaysian construction industry. *International Journal of Project Management*, *25*(5), 517–526. https://doi.org/10.1016/j.ijproman.2006.11.007

Sánchez Rodríguez, A., Riveiro Rodríguez, B., Soilán Rodríguez, M., & Arias Sánchez, P. (2019). Laser scanning and its applications to damage detection and monitoring in masonry structures. *Long-term Performance and Durability of Masonry Structures*, 265–285. https://doi.org/10.1016/B978-0-08-102110-1.00009-1

Saqib, M., Farooqui, R. U., & Lodi, S. H. (2008, August). Assessment of critical success factors for construction projects in Pakistan. In First international conference on construction in developing countries, Karachi, Pakistan (pp. 392-404).

Schenk, T. (2005). *Introduction to Photogrammetry*. Retrieved from https://pdfs.semanticscholar.org/7ed2/25c0799608539512fd84597892a5eb03e0b3.pdf

Shahi, A., West, J. S., & Haas, C. T. (2013). Onsite 3D marking for construction activity tracking. Automation in Construction, 30, 136–143. https://doi.org/10.1016/j.autcon.2012.11.027.

Small, E. P., Hamouri, K. A., & Hamouri, H. A. (2017). Examination of Opportunities for Integration of Lean Principles in Construction in Dubai. *Procedia Engineering*, *196*, 616–621. https://doi.org/10.1016/j.proeng.2017.08.049

Sohi, A. J., Hertogh, M., Bosch-Rekveldt, M., & Blom, R. (2016). Does Lean & Agile Project Management Help Coping with Project Complexity? *Procedia - Social and Behavioral Sciences*, *226*, 252–259. https://doi.org/10.1016/j.sbspro.2016.06.186

Stone , P. A. (1983). *Building economy: Design, production and organization—A synoptic view*, 3rd ed. , Oxford: Pergamon Press.

Strassman , W. P. (1970). The construction sector in economic development . *The Scottish Journal of Political Economy* , November, 390–410.

Sulankivi, K., Kähkönen, K., Mäkelä, T., & Kiviniemi, M. (2010). 4D-BIM for Construction Safety Planning. Retrieved from https://www.researchgate.net/publication/228640694_4D-BIM_for_Construction_Safety_Planning.

Turin, D. A. (1973). *The construction industry: Its economic significance and its role in development*, , 2nd ed. , London : University College Environmental Research Group.

Turkan, Y., Bosche, F., Haas, C. T., & Haas, R. (2012). Automated progress tracking using 4D schedule and 3D sensing technologies. *Automation in Construction*, *22*, 414–421. https://doi.org/10.1016/j.autcon.2011.10.003

Tuttas, S., Braun, A., Borrmann, A., & Stilla, U. (2014). Comparision of photogrammetric point clouds with BIM building elements for construction progress monitoring. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *XL–3*, 341–345. https://doi.org/10.5194/isprsarchives-xl-3-341-2014

Van Schaijk, S. (2016). Building Information Model (BIM) based process mining: Enabling knowledge reassurance and fact-based problem discovery within the Architecture, Engineering, Construction and Facility Management Industry (master's thesis). Retrieved from https://www.dropbox.com/s/ww1htr6kahuv7l7/Master%20Thesis%20Stijn%20van%20Schaijk%20Digital%20Version.pdf?dl=0

WEF. (2016, 20 September). *By 2060, this country will have the world's largest population*. Retrieved on 9 November 2019, van https://www.weforum.org/agenda/2016/09/the-countries-with-the-biggest-populations-from-1950-to-2060/

Wigren, R., & Wilhelmsson, M. (2007). Construction investments and economic growth in Western Europe. *Journal of Policy Modeling*, *29*(3), 439–451. https://doi.org/10.1016/j.jpolmod.2006.10.001

Wolf, P. R., & Dewitt, B. A. (2000). *Elements of Photogrammetry: With Applications in GIS.* United States of America: McGraw-Hill.

Y. Wu, H. Kim, C. Kim, S.H. Han, Object recognition in construction-site images using 3D CAD-based filtering, Journal of Computing in Civil Engineering 24 (2010) 56–64.

Yang, Z., Seo, Y.-H., & Kim, T. (2010). Adaptive triangular-mesh reconstruction by mean-curvature-based refinement from point clouds using a moving parabolic approximation. *Computer-Aided Design*, *42*(1), 2–17. https://doi.org/10.1016/j.cad.2009.04.014

Zhang, C., & Arditi, D. (2013). Automated progress control using laser scanning technology. Automation in Construction, 36, 108-116.

Zhang, C., Kalasapudi, V. S., & Tang, P. (2016). Rapid data quality oriented laser scan planning for dynamic construction environments. *Advanced Engineering Informatics*, *30*(2), 218–232. https://doi.org/10.1016/j.aei.2016.03.004

Zhang, F., Fleyeh, H., Wang, X., & Lu, M. (2019). Construction site accident analysis using text mining and natural language processing techniques. *Automation in Construction*, *99*, 238–248. https://doi.org/10.1016/j.autcon.2018.12.016

Zhang, S., Teizer, J., Lee, J.-K., Eastman, C. M., & Venugopal, M. (2013). Building Information Modeling (BIM) and Safety: Automatic Safety Checking of Construction Models and Schedules. *Automation in Construction*, *29*, 183–195. https://doi.org/10.1016/j.autcon.2012.05.006

Zhou, Y., Ding, L., Wang, X., Truijens, M., & Luo, H. (2015). Applicability of 4D modeling for resource allocation in mega liquefied natural gas plant construction. *Automation in Construction*, *50*, 50–63. https://doi.org/10.1016/j.autcon.2014.10.016

Zou, Y., Kiviniemi, A., & Jones, S. W. (2017). Retrieving similar cases for construction project risk management using Natural Language Processing techniques. *Automation in Construction*, *80*, 66–76. https://doi.org/10.1016/j.autcon.2017.04.003

Appendix

This appendix consists of three parts:

- Part 1: .ply to .pcd converter;
- Part 2: geometry.py file (needed for part 3);
- Part 3: tool.py file (viewer).

Part 1

```
import sys
import os

header = "# .PCD v.7 - Point Cloud Data file format\n\
VERSION .7\n\
FIELDS x y z\n\
SIZE 4 4 4\n\
TYPE F F F\n\
COUNT 1 1 1\n\
WIDTH XXXX\n\
HEIGHT 1\n\
POINTS XXXX\n\
DATA ascii"


def convertionOfPlyToPcd(ply_file, pcd_file):
    input_file = open(r"#insert here your location of the .py file")
    out = pcd_file
    output = open(r""#insert here your designated location for .pcd file
", 'w')
    write_points = False
    points_counter = 0
    nr_points = 0
    for s in input_file.readlines():
        if s.find("element vertex") != -1:
            nr_points = int(s.split(" ")[2].rstrip().lstrip())
            new_header = header.replace("XXXX", str(nr_points))
            output.write(new_header)
            output.write("\n")
        if s.find("end_header") != -1:
            write_points = True
            continue
        if write_points and points_counter < nr_points:
            points_counter = points_counter + 1
            output.write(" ".join(s.split(" ", 3)[:3]))
            output.write("\n")
    input_file.close()
    output.close()

if __name__ == "__main__":
    if sys.argv[0] == "":
        sys.exit(1)
    ply_file = sys.argv[0]

    pcd_file = sys.argv[0]
    convertionOfPlyToPcd(ply_file, pcd_file)
```

Part 2

```
import numpy as np

DISTANCE_THRESHOLD = 0
THRESHOLD = 0

class Cube:
    def __init__(self, vertices):
        assert len(vertices) == 48
        self.center = np.average(vertices[:, 0]), np.average(vertices[:, 1]),
np.average(vertices[:, 2])
        self.min_point = np.array([1000000000, 1000000000, 1000000000])
        self.max_point = np.array([-1000000000, -1000000000, -1000000000])

        vertices = list(map(tuple, list(vertices)))
        self.faces = []
        for i in range(0, len(vertices), 8):
            face_vertices = []
            for j in range(i, i+6, 2):
                l1 = vertices[j: j+2]
                l2 = vertices[j+2: j+4]
                v = [x for x in list(l1) if x not in list(l2)][0]
                face_vertices.append(v)
                if j==i+4:
                    v = [x for x in list(l1) if x in list(l2)][0]
                    face_vertices.append(v)
            vmax = np.amax(face_vertices, axis=0)
            vmin = np.amin(face_vertices, axis=0)
            self.max_point[vmax > self.max_point] = vmax[vmax > self.max_point
]
            self.min_point[vmin < self.min_point] = vmin[vmin < self.min_point
]
            self.faces.append(Face(*face_vertices))

        self.max_point += DISTANCE_THRESHOLD * 2
        self.min_point -= DISTANCE_THRESHOLD * 2
        assert len(self.faces) == 6

    def overlaps(self, p):
        for i in range(len(self.faces)):
            if self.faces[i].overlaps(p):
                return True
        return False

    # def orthogonal_distance(self, p):
    #     distances = []
    #     for i in range(len(self.faces)):
    #         if self.faces[i].overlaps(p):
```

```
    #             distances.append(self.faces[i].orthogonal_distance(p))

    #     if len(distances) == 0:
    #         return 100000000
    #     elif len(distances) == 6:
    #         return 0
    #     else:
    #         print(distances)
    #         return np.amin(np.abs(distances))

    def orthogonal_distance(self, p):
        distances = []
        if ((self.min_point < p) & (p < self.max_point)).all():
            for i in range(len(self.faces)):
                if self.faces[i].overlaps(p):
                    distances.append(self.faces[i].orthogonal_distance(p))
                else:
                    distances.append(10000000)

        if len(distances) == 0 or np.amin(np.abs(distances)) > DISTANCE_THRESH
OLD:
            return -1, 100000000000
        # elif len(distances) == 6:
        #     return 0
        else:
            idx = np.argmin(np.abs(distances))
            # self.faces[idx].add_vertex(p)
            return idx, abs(distances[idx])

    def retrieve_points(self):
        points = []
        counter = 0
        for i in range(len(self.faces)):
            ratio, tmp_pnts = self.faces[i].retrieve_points()
            if ratio > 0.65:
                counter += 1
            points += tmp_pnts
        return counter, points


class Face:
    def __init__(self, v1, v2, v3, v4):
        self.vertices = np.array([v1, v2, v3, v4])
        self.v1 = self.vertices[0, :]
        self.v2 = self.vertices[1, :]
        self.v3 = self.vertices[2, :]
        self.v4 = self.vertices[3, :]

        self.center = np.array([
```

```
                    np.average(self.vertices[:, 0]),
                    np.average(self.vertices[:, 1]),
                    np.average(self.vertices[:, 2])
                ])

        self.face_normal = np.cross(self.v3 - self.v1, self.v4 - self.v2)
        self.face_normal /= np.linalg.norm(self.face_normal)

        self.base1 = self.v1 - self.v2
        self.base1_len = np.linalg.norm(self.base1)
        self.base1 /= self.base1_len

        self.base2 = self.v3 - self.v2
        self.base2_len = np.linalg.norm(self.base2)
        self.base2 /= self.base2_len

        self.plane_normals = np.array([
                    np.cross(self.v2 - self.v1, self.face_normal),
                    np.cross(self.v3 - self.v2, self.face_normal),
                    np.cross(self.v4 - self.v3, self.face_normal),
                    np.cross(self.v1 - self.v4, self.face_normal),
                ])
        self.plane_normals /= np.linalg.norm(self.plane_normals, axis=1)[:, np
.newaxis]

        self.gridlen = 3
        self.points_assigned = [[[] for _ in range(self.gridlen)] for _ in ran
ge(self.gridlen)]

    def overlaps(self, p):
        # if isinstance(p, tuple):
        #     assert len(p) == 3
        # if isinstance(p, np.array([]).__class__):
        #     assert p.shape == (3,)

        # for i in range(4):
        #     sign1 = np.sign(np.dot(p - self.vertices[i, :], self.plane_norma
ls[i, :]))
        #     sign2 = np.sign(np.dot(self.center - self.vertices[i, :], self.p
lane_normals[i, :]))
        #     if sign1 != sign2:
        #         return False

        self.projection1 = np.dot(p - self.v2, self.base1)
        self.projection2 = np.dot(p - self.v2, self.base2)

        if  0 < self.projection1 < self.base1_len \
            and \
            0 < self.projection2 < self.base2_len:
```

```
            return True
        else:
            return False
        # return True

    def add_vertex(self, p):
        i = int((self.projection1 * self.gridlen) / self.base1_len)
        j = int((self.projection2 * self.gridlen) / self.base2_len)
        self.points_assigned[i][j].append(p)

    def orthogonal_distance(self, p):
        return np.dot(p - self.v1, self.face_normal)

    def retrieve_points(self):
        points = []
        area = (self.base1_len / self.gridlen) * (self.base2_len / self.gridle
n) / 1000000
        counter = 0
        for i in range(self.gridlen):
            for j in range(self.gridlen):
                print(len(self.points_assigned[i][j]), THRESHOLD * area)
                if len(self.points_assigned[i][j]) > THRESHOLD * area:
                    counter += 1
                points += self.points_assigned[i][j]

        return counter/ (self.gridlen**2), points

def parse_stp_file(filename):
    points = [["", []]]
    with open(filename, "r") as f:
        lines = f.readlines()
        for i in range(len(lines)):
            if lines[i].find("=VERTEX_POINT(") >= 0:
                begin = lines[i].rfind("#") + 1
                end = lines[i].rfind(")")
                v_id = lines[i][begin: end]

                if "#" + v_id != lines[i+1][:len(v_id) + 1]:
                    print("WARNING: Inconsistent cartesian point line number \
'#", v_id, "\'defined in VERTEX_POINT. (vs ", lines[i+1][:len(v_id) + 1], ")")
                    continue

                begin = lines[i+1].rfind("(") + 1
                end = lines[i+1].rfind(")") - 1
                cartesian_point = lines[i+1][begin: end]
                cartesian_point = tuple(map(float, cartesian_point.split(","))
)
                points[-1][1].append(cartesian_point)
```

```
            elif lines[i].find("=NEXT_ASSEMBLY_USAGE_OCCURRENCE(") >= 0:
                begin = lines[i].find("(") + 1
                end = lines[i].rfind(")")
                values = lines[i][begin: end].split(",")
                flags = list(map(lambda x: (x.count("'") + x.count("\"")) % 2,
values))

                # Commas inside quotes are not to be considered seperators. So
following code merges them back together.
                j=0
                while j < len(values):
                    if flags[j]:
                        k = j+1
                        while k < len(values):
                            if flags[k]:
                                values[j] = ",".join(values[j:k+1])
                                del flags[j+1: k+1]
                                del values[j+1: k+1]
                                break
                            k += 1
                    j += 1

                name = values[1][1:-1]
                points[-1][0] = name
                points.append(["", []])
    return points
```

Part 3

```
import os
import random
import struct
from tqdm import tqdm
import pandas as pd
import numpy as np

import geometry
from geometry import Cube, Face, parse_stp_file

import OCC
from OCC.Core.Graphic3d import Graphic3d_ArrayOfPoints
from OCC.Core.AIS import AIS_PointCloud
from OCC.Core.Quantity import Quantity_Color, Quantity_TOC_RGB
from OCC.Core.gp import gp_Pnt

from OCC.Extend.DataExchange import read_step_file_with_names_colors
from OCC.Display.SimpleGui import init_display

from OCC.Core.TopExp import TopExp_Explorer

from OCC.Core.TopAbs import TopAbs_VERTEX

display, start_display, add_menu, add_function_to_menu = init_display()

# Points per pixel square
FINAL_THRESHOLD = 1.05
GRID_POINTS_THRESHOLD = 50
DISTANCE_THRESHOLD = 500

geometry.GRID_POINTS_THRESHOLD = GRID_POINTS_THRESHOLD
geometry.DISTANCE_THRESHOLD = DISTANCE_THRESHOLD

flags = []
cubes = []
filtered_pointcloud = []
shapes_labels_colors = None

excel_filename =
stp_filename =
pcd_filename =
out_pcd_filename =

# excel_filename = r"newfiles/csvtrudo.xls"
# stp_filename = r'newfiles/3d.stp'
# pcd_filename = r"newfiles/pc.pcd"
# out_pcd_filename = r"newfiles/filtered_pc.pcd"
```

```
def PointCloudModel(event=None):
    # compute number of lines
    nbr_of_vertices = pcd_get_number_of_vertices(pcd_filename)
    print("Number of vertices :", nbr_of_vertices)
    # create the point_cloud
    pc = Graphic3d_ArrayOfPoints(nbr_of_vertices)
    # fedd it with vertices
    fp = open(pcd_filename, 'r')
    # read 11 lines to skip header
    for i in range(10):
        fp.readline()
    for i in range(nbr_of_vertices):
        line = fp.readline()
        x, y, z = map(float, line.split())
        pc.AddVertex(x, y, z)
    point_cloud = AIS_PointCloud()
    point_cloud.SetPoints(pc)
    ais_context = display.GetContext()
    ais_context.Display(point_cloud, True)
    display.View_Iso()
    display.FitAll()

def Model3D(event=None):
    shapes_labels_colors = read_step_file_with_names_colors(stp_filename)
    for shpt_lbl_color in shapes_labels_colors:
        label, c = shapes_labels_colors[shpt_lbl_color]
        display.DisplayColoredShape(shpt_lbl_color, color=Quantity_Color(c.Red
(),
                                                                         c.Green()
,
                                                                         c.Blue(),
                                                                         Quantity_
TOC_RGB))

def make_geometry(stp_filename):
    global shapes_labels_colors

    cubes = []

    if shapes_labels_colors is None:
        shapes_labels_colors = read_step_file_with_names_colors(stp_filename)

    max_point = np.array([-1000000000, -1000000000, -1000000000])
    min_point = np.array([1000000000, 1000000000, 1000000000])
    x=0
    y=1500
    for i, shape in enumerate(shapes_labels_colors.keys()):
        if i<x:
```

```
            continue
        if i>=y:
            break
        TopExp_Explorer(shape, TopAbs_VERTEX)

        exp = TopExp_Explorer(shape, TopAbs_VERTEX)

        vertices = []
        while exp.More():
            pnt = OCC.Core.BRep.BRep_Tool.Pnt(exp.Value())
            vertices.append([pnt.X(), pnt.Y(), pnt.Z()])
            exp.Next()

        vertices = np.array(vertices)
        vmax = np.amax(vertices, axis=0)
        vmin = np.amin(vertices, axis=0)
        max_point[vmax > max_point] = vmax[vmax > max_point]
        min_point[vmin < min_point] = vmin[vmin < min_point]

        cubes.append(Cube(vertices))

    name_to_points_mapping = parse_stp_file(stp_filename)
    assert len(cubes) == len(name_to_points_mapping)
    for i, cube in enumerate(cubes):
        points = []
        for face in cube.faces:
            points.append(face.vertices)
        points = np.concatenate(points, axis=0)
        distinct_points = list(np.unique(points, axis=0))
        distinct_points = np.array(sorted(distinct_points, key=lambda x: (x[0]
, x[1], x[2])))

        assert distinct_points.shape[0] == 8

        # for name, points in name_to_points_mapping:
        #     print(distinct_points - points)
        #     if (distinct_points == points).all():
        #         cube.obj_name = name
        cube.obj_name = name_to_points_mapping[i][0]

    max_point += DISTANCE_THRESHOLD * 2
    min_point -= DISTANCE_THRESHOLD * 2

    return min_point, max_point, cubes

def export_pointcloud(event=None):
    global filtered_pointcloud

    with open(out_pcd_filename, 'w') as f_out:
```

```
        with open(pcd_filename, 'r') as f_in:
            for i in range(10):
                line = f_in.readline()
                if i==8:
                    f_out.write(line.split()[0] + " " + str(len(filtered_point
cloud)) + "\n")
                else:
                    f_out.write(line)

        for point in filtered_pointcloud:
            f_out.write("{:.6f} {:.6f} {:.6f}".format(*point) + "\n")

def export_status_to_excel(event=None):
    global cubes

    excel_sheets = pd.read_excel(excel_filename, sheet_name=None)

    for i, cube in enumerate(cubes):
        r_name = cube.obj_name

        r_id = excel_sheets["Resources"].loc[excel_sheets["Resources"]["Resour
ceName"] == r_name].iloc[0]["ResourceID(*)"]

        tmp = excel_sheets["Assignments"].loc[excel_sheets["Assignments"]["Res
ourceID(*)"] == r_id]
        if tmp.shape[0] == 0:
            continue
        task_name = tmp.iloc[0]["TaskName(*)"]

        tmp = excel_sheets["Tasks"].loc[excel_sheets["Tasks"]["TaskName"] == t
ask_name]
        if tmp.shape[0] == 0:
            continue
        status = tmp.iloc[0]["Status"]

        if flags[i]:
            excel_sheets["Tasks"].loc[excel_sheets["Tasks"]["TaskName"] == tas
k_name, "Status"] = "Finished"

    excel_sheets["Settings"].loc[0, "IgnoreDirectCosts"] = "FALSE"

    writer = pd.ExcelWriter(excel_filename.replace(".xls", "_2.xls"))
    for key in excel_sheets:
        excel_sheets[key].to_excel(writer, key, index=None)
    writer.save()


def pcd_get_number_of_vertices(pcd_filename):
    """ open the PCD file, read header and get number of vertices.
```

```
    Header looks like:
    # .PCD v.5 - Point Cloud Data file format
    VERSION .5
    FIELDS x y z
    SIZE 4 4 4
    TYPE F F F
    COUNT 1 1 1
    WIDTH 397
    HEIGHT 1
    POINTS 397
    DATA ascii
    """
    f = open(pcd_filename, 'r')
    for i in range(8):
        f.readline()
    # the 9th line holds the number of points
    number_of_points = int(f.readline().split()[1])
    f.close()
    return number_of_points


def Clean(event=None):
    global flags
    global cubes
    global filtered_pointcloud

    min_point, max_point, cubes = make_geometry(stp_filename)

    nbr_of_vertices = pcd_get_number_of_vertices(pcd_filename)

    print("Number of vertices :", nbr_of_vertices)

    pc = Graphic3d_ArrayOfPoints(nbr_of_vertices)
    fp = open(pcd_filename, 'r')

    for i in range(10):
        fp.readline()

    lines = fp.readlines()
    vertices = np.array(list(map(lambda line: list(map(float, line.split())),
lines)))

    # Algorithm starts here
    pbar = tqdm(total=100)

    indices = \
                (vertices[:, 0] > min_point[0]) & \
                (vertices[:, 1] > min_point[1]) & \
                (vertices[:, 2] > min_point[2]) & \
```

```
                (vertices[:, 0] < max_point[0]) & \
                (vertices[:, 1] < max_point[1]) & \
                (vertices[:, 2] < max_point[2])
    vertices = vertices[indices, :]

    pbar.update(5)

    distances = []
    indices = []
    pbar.update(90 % len(cubes))
    for c in cubes:
        idx, dist = c.orthogonal_distance(vertices)
        distances.append(dist)
        indices.append(idx)
        pbar.update(90 // len(cubes))
    distances = np.stack(distances, 0)
    face_indices = np.stack(indices, 0)

    pbar.update(2)

    c_indices = np.argmin(distances, axis=0)
    for i, c in enumerate(cubes):
        for j, f in enumerate(c.faces):
            mask = np.where((c_indices==i) & (face_indices[i, :] == j))[0]
            f.add_vertices(vertices, mask)
    pbar.update(1)

    global flags
    for c in cubes:
        faces, points = c.retrieve_points()
        flags.append(faces > FINAL_THRESHOLD)
        # if faces > 1:
        filtered_pointcloud += points
    pbar.update(2)

    pbar.close()
    # Algorithm Ends here

    print("")
    print("Total number of vertices: \t", nbr_of_vertices)
    print("Filtered number of vertices: \t", len(filtered_pointcloud))
    print("")

    print("Status:\t\t\tObject Names:\n" )
    for i, c in enumerate(cubes):
        print("", flags[i], "\t\t\t", c.obj_name)
    print("")

    for point in filtered_pointcloud:
```

```
        pc.AddVertex(*point)

    point_cloud = AIS_PointCloud()
    point_cloud.SetPoints(pc)
    ais_context = display.GetContext()
    ais_context.Display(point_cloud, True)
    display.View_Iso()
    display.FitAll()

def Intersect(event=None):
    global flags
    global shapes_labels_colors

    if shapes_labels_colors is None:
        shapes_labels_colors = read_step_file_with_names_colors(stp_filename)

    x=0
    y=1500
    for i, shpt_lbl_color in enumerate(shapes_labels_colors.keys()):
        if i<x:
            continue
        if i>=y:
            break
        color = (0.55, 0.55, 0.55)
        if len(flags)==len(shapes_labels_colors):
            if flags[i]:
                color = (0.3, 0.7, 0.3)
            else:
                color = (0.7, 0.3, 0.3)

        label, c = shapes_labels_colors[shpt_lbl_color]
        display.DisplayColoredShape(shpt_lbl_color, color=Quantity_Color(*colo
r,
                                                                                Quantity_
TOC_RGB))
if __name__ == '__main__':
    add_menu('PointCloud')
    add_function_to_menu('PointCloud', PointCloudModel)
    add_menu('3Dmodel')
    add_function_to_menu('3Dmodel', Model3D)
    add_menu('CleanData')
    add_function_to_menu('CleanData', Clean)
    add_menu('IntersectData')
    add_function_to_menu('IntersectData', Intersect)
    add_menu('ExportData')
    add_function_to_menu('ExportData', export_pointcloud)
    add_function_to_menu('ExportData', export_status_to_excel)
    start_display()
```