NTNU
Norwegian University of
Science and Technology

Master's degree thesis

IP501909 MSc thesis, discipline oriented master

Point cloud from 3D scan to robot programmable surfaces – an application on robot cleaning of fish processing equipment

Tor-Arne Magerholm

Number of pages including this page: 76

Aalesund, 11.06.2019

NTNU
Norwegian University of
Science and Technology

Mandatory statement

Each student is responsible for complying with rules and regulations that relate to examinations and to academic work in general. The purpose of the mandatory statement is to make students aware of their responsibility and the consequences of cheating. **Failure to complete the statement does not excuse students from their responsibility.**

*Please complete the mandatory statement by placing a mark **in each box** for statements 1-6 below.*		
1.	**I/we herby declare that my/our paper/assignment is my/our own work, and that I/we have not used other sources or received other help than is mentioned in the paper/assignment.**	☒
2.	**I/we herby declare that this paper** 1. Has not been used in any other exam at another department/university/university college 2. Is not referring to the work of others without acknowledgement 3. Is not referring to my/our previous work without acknowledgement 4. Has acknowledged all sources of literature in the text and in the list of references 5. Is not a copy, duplicate or transcript of other work	Mark each box: 1. ☒ 2. ☒ 3. ☒ 4. ☒ 5. ☒
3.	**I am/we are aware that any breach of the above will be considered as cheating, and may result in annulment of the examination and exclusion from all universities and university colleges in Norway for up to one year, according to the Act relating to Norwegian Universities and University Colleges, section 4-7 and 4-8 and Examination regulations paragraph 31.**	☒
4.	**I am/we are aware that all papers/assignments may be checked for plagiarism by a software assisted plagiarism check**	☒
5.	**I am/we are aware that NTNU Ålesund will handle all cases of suspected cheating according to prevailing guidelines.**	☒
6.	**I/we are aware of the University's rules and regulation for using sources paragraph 30.**	☒

Publication agreement

ECTS credits: 30

Supervisor: Ola Jon Mork
Co-Supervisor: Vegar Kobbevik

Agreement on electronic publication of master thesis

Author(s) have copyright to the thesis, including the exclusive right to publish the document (The Copyright Act §2).
All theses fulfilling the requirements will be registered and published in Brage HiÅ, with the approval of the author(s).
Theses with a confidentiality agreement will not be published.

I/we hereby give NTNU Ålesund the right to, free of charge, make the thesis available for electronic publication: ☒**yes** □**no**

Is there an agreement of confidentiality? □**yes** ☒**no**
(A supplementary confidentiality agreement must be filled in)
- If yes: **Can the thesis be online published when the period of confidentiality is expired?** □**yes** □**no**

Date: 11.06.2019

Abstract

Fish is an important industry for Norway and especially on the west coast. Many companies reside there where they control the whole process from egg hatching, to breeding and to slaughtering. In this final step where the fish are brought in to a fish processing facility, a huge amount of fish goes from living fish to salmon servings. Within a process like this, a potential threat resides which is the dangerous Listeria bacteria. To prevent this type of bacteria each fish processing facility has its own cleaning procedures to keep the facility clean for bacteria growth. This is usually solved with manual labor which forces workers to work in heavy duty conditions with a lot of chemicals and none-ergonomic work procedures.

The objective of this thesis is to look at the possibilities of developing procedures to create automatic cleaning solutions for fish processing equipment and facilities. The goal is to reduce the risk of Listeria growth with proper and traceable cleaning to exactly know which part of the facility has been cleaned. With the use of 3D spatial data and offline robot programming, such solutions can be solved off-site to prevent disturbing the ongoing fish processing process.

A thoroughly state-of-the-art literature review has been conducted to understand possible possibilities, but also get a grasp of which knowledge and research are missing. By looking at both how 3D scanning can be used for certain projects and how simulation can be used to develop automatic cleaning solutions.

The research approach is executed in 3 main steps. A case study for a test cell has been acquired and will be used to develop procedures and workflow based on the case study method and simulation method. Qualitative research is what mostly comes out of this thesis, but the possibilities for doing quantitative research for future work is definitely there.

The results of this thesis show that the possibility of creating automatic cleaning solutions are there. Where the cut of on-site measuring and on-site programming can be reduced heavily to do this off-site instead. The developed procedure can be executed with the workflow developed to create robot programmable surfaces in a simulation environment on a common software platform.

This thesis concludes with the possibilities of using software to create robot programmable surfaces to create automatic cleaning solutions. There are however certain elements such as data capturing and efficient robot programming which needs to be looked further upon. This thesis can function as a building block for upcoming projects trying to solve the manual labored cleaning procedures in fish processing facilities.

Sammendrag

Fisk er en viktig industri for Norge og spesielt på vestkysten. Mange selskap befinner seg der hvor de kan kontrollere hele prosessen fra egg klekking, til avl og slakting. I det sistnevnte steget hvor fisken blir fraktet til et prosesseringsanlegg for fisk, går en stor menge til lakseporsjoner. Inne i en prosess som dette, finnes en mulig risiko som er den farlige Listeria bakterien. For å unngå denne typen bakterie har hvert prosesseringsanlegg sine vaskeprosedyrer for å holde hele området rent. Dette er vanligvis løst med manuell vasking som fører til at arbeidere må jobbe i tøffe arbeidsforhold med kjemikalier og ikke ergonomiske prosesser.

Oppgaven til denne avhandlingen er å se på muligheten til å utvikle prosedyrer som kan lage automatiske vaskeløsninger for prosesseringsutstyr og -anlegg. Målet er å redusere risikoen for Listeria vekst med skikkelig og påviselig vasking for å vite nøyaktig hvilken del av fabrikken som er vasket. Med bruken av 3D data og offline robotprogrammering en slik løsning kan bli løst utenfor fabrikken for å unngå og forstyrre den pågående prosesseringen av fisk.

En grundig litteratur søk har blitt gjort for å forstå de mulige mulighetene, men også forstå hvilken kunnskap og forskning som mangler. Ved å se på hvordan 3D skanning kan be brukt for enkelte prosjektet og hvordan simulering kan bli brukt for å utvikle automatiske vaskeprosedyrer.

Forskningsmetoden er utført i 3 steg. En sak undersøkelse for et testrom har blitt brukt for å utvikle prosedyrer og arbeidsflyt basert på sak undersøkelse metoden og simulerings metoden. Kvalitativ forskning er hva som kommer ut av avhandlingen, men mulighetene til å gjøre kvantitative forskning for videre arbeid er definitivt der.

Resultatene av avhandlingen viser at det er mulig å lage automatiske vaskeløsninger. Hvor arbeidet er kuttet fra innsiden til å jobbe med det på utsiden av fabrikken. Den utviklede prosedyren kan i lag med arbeidsflyten utvikle robot programmerbare overflater i et simulerings miljø for en samlet programvare plattform.

Denne avhandlingen konkludere med mulighetene av bruken av programvare til å lage robot programmerbare overflater for å lage automatisk vaske løsningen. Det er enkelte elementer som datahåndtering og effektiv robotprogrammering som må fortsatt videreutvikles. Denne avhandlingen kan være en byggestein for kommende prosjekter som prøver å løse det manuelle vaskearbeidet i prosesseringsanlegg for fisk.

Preface and Acknowledgement

This thesis is a part of the Master of Science degree in Product- and System Design at the Norwegian University of Science and Technology in Aalesund. This thesis was written in its entirety during the spring of 2019, and its workload equates to 30 ECTS.

The main idea behind this thesis is to look at an alternative solution for the cleaning of older fish processing facilities, with the help of different methodologies. NTNU has been for years looking at the fish industry with aspects of a cleaner design, slimmer robots and now 3D scanning. Exploring the possibilities of combining 3D scanning with robot programming to remove time necessary on site for programming and onto the virtual model instead. This could potentially lead to a new way of thinking and perhaps an automatic cleaning solution for fish processing equipment in the future.

I would first want to thank my supervisor Prof. Ola Jon Mork for the patients and guidance during my thesis writing. Without your calm and collected mind, I would definitely not be where I am currently now.

I would also like to thank my co-supervisor Vegar Kobbevik for the time we spent figuring out the software and how to handle it. With your knowledge and expertise, this thesis has been brought up to another level.

I would also like to thank Paul Steffen Kleppe for providing me contacts and equipment necessary for this thesis. Also, Andreas Longva for giving me access to their computer so I could continue working on this thesis.

Finally, I would like to thank my family for consistent support throughout this long journey. The encouragement and support helped me finish this thesis and I greatly appreciate that. And an extra thank you to my little sister for always keeping me positive and giving me lots of laughs.

Ålesund, 11th of June 2019

Tor-Arne Magerholm

Tor-Arne Magerholm

Contents

List of figures vii
List of tables viii
Abbreviations viii
1 Introduction 1
1.1 Background 1
1.2 Problem definition 2
1.3 Scope 3
1.4 Motivation 3
1.5 Objective and Research Questions 4
1.6 Structure of the Thesis 4
2 Literature Review 5
2.1 State of the art in Spatial data acquisition 5
2.1.1 3D scanning technology 5
2.1.2 3D scanning techniques 7
2.1.3 Alternative to 3D scanning 9
2.2 Guidelines for Simulation 9
2.2.1 Simulation theory method 9
2.2.2 Simulation procedures 11
2.3 Existing approaches for 3D scan to 3D solid 14
2.3.1 A theoretical framework for generating solids from spatial data 14
2.3.2 Industrial use of 3D scan procedures and software 15
2.3.3 3D solid workflow for civil engineering 17
2.3.4 A study comparing 3D scanning versus 3D modeling 20
2.4 Requirements for solid models 22
3 Methodology 24
3.1 Assumptions and Limitations 24
3.2 Research method 25
3.2.1 Action based method 25
3.2.2 Case study method 26
3.2.3 Discrete event simulation method 27
3.3 Flow Chart 27
4 Case study 31
4.1 Case introduction and background 31
4.1.1 Simple case 31
4.1.2 Main case 32

4.2 The current situation in fish processing production 33
4.2.1 Production and operating hours at Marine Harvest.................................33
4.2.2 Current cleaning solution...33
5 Result and analysis 35
5.1 Procedure to generate 3D solids and programmable surfaces.........................35
5.1.1 Previously developed procedures..35
5.1.2 Developed procedure on a common platform 35
5.2 Workflow to create robot programmable surfaces on a common platform 37
5.2.1 Processing point cloud data 37
5.2.2 Generating mesh from point cloud data...40
5.2.3 Handling certain errors from mesh generation 43
5.2.4 Generating robot programmable surfaces from mesh..............................46
5.2.5 The solid requirement for robot programming 50
5.3 Simulation results 51
5.3.1 Simulation set-up...51
5.3.2 Simulation procedure...52
6 Discussion...58
6.1 Developed procedure 58
6.2 Workflow and simulation 59
7 Conclusion 61
8 Future Work 62
9 References...63

List of figures

Figure 1 - Manual labor cleaning.. 1
Figure 2 - Scope of the thesis.. 3
Figure 3 - Data Acquisition Methods [6].. 5
Figure 4 - Instruments used to capture spatial data [11] 8
Figure 5 - Alternative tools to 3D scanning [11] 9
Figure 6 - Model of simulation [15] 11
Figure 7 - Model conceptualization [16]..12
Figure 8 - Workflow for indoor assets [7]..15
Figure 9 - Software procedure [7]...16
Figure 10 - Industrial workflow [17]..17
Figure 11 - Level of effort in a 3D Scanning process [11]......................................18
Figure 12 – BIM 3D Solids procedure [11] 19
Figure 13 - Workflow achieving 3D solids [10]..21
Figure 14 - Framework to define solid model [13]...22
Figure 15 - The three main levels for the framework [13] 22
Figure 16 - Action based method for test cell 25
Figure 17 - Flow Chart...27
Figure 18 - Top of a band saw 32
Figure 19 – 3D model of the testing cell at Optimar 32
Figure 20 - Cleaning procedure at Marine Harvest...33
Figure 21 - Procedure to develop robot programmable surface 35
Figure 22 - Digitized Shape Preparation..37
Figure 23 - Import dial box 38
Figure 24 - Test cell as Spatial data 38
Figure 25 - Align using the Robot...39
Figure 26 - Robot Manipulation..39
Figure 27 - Removing unwanted noise..40
Figure 28 - Mesh creation 41
Figure 29 - Mesh preparation 42
Figure 30 - Mesh preparation complete 42
Figure 31 - None-smooth mesh and smooth mesh 43
Figure 32 - Fresh mesh 43
Figure 33 - Holes and boundaries menu 44
Figure 34 - Holes marked in the mesh...44
Figure 35 - Command tool menu 45
Figure 36 - Smooth boundaries + Fill the hole..45
Figure 37 - Digitized Shape to Surface 46
Figure 38 - Complete mesh of test cell 46
Figure 39 - 3D Primitive Recognition 47
Figure 40 - Angular Tolerance 20 %...47
Figure 41 - Angular Tolerance 10%..48
Figure 42 - Surface editing 48
Figure 43 - Intersecting surfaces and connected corners 49
Figure 44 - Near complete surface creation...49
Figure 45 - Fish Stunner..50
Figure 46 - Input, Variables, State, Entities and Time...51
Figure 47 - Activities, Output, Resource, Events, Attributes 52

Figure 48 - Surface Simulation 52
Figure 49 - Test cell with a linear guiding system and a FANUC Robot 53
Figure 50 - Paint Profile 53
Figure 51 - Surface Trajectory 54
Figure 52 - Approach Direction and Sweep Direction 55
Figure 53 - Creating Robot Tasks 55
Figure 54 - Unreachable targets 56
Figure 55 - Rail Axes Programming 57
Figure 56 - Running the simulation 57

List of tables

Table 1 - 3D Technologies [9] 6
Table 2 - Simulation components [15] 10
Table 3 - Time spent 3D modeling and scanning [10] 20

Abbreviations

NTNU	The Norwegian technical and natural science university
CAD	Computer-Aided Design
BIM	Building Information Modeling
MBOM	Manufacturing Bill of Material
LoD	Level of Development
LoA	Level of Accuracy
LoR	Level of Recognizability
AR	Action Research
DES	Discrete Even Simulation

1 Introduction

1.1 Background

Fish is an important industry for Norway and especially on the west coast. Many companies reside there where they control the whole process from egg hatching, to breeding and to slaughtering. In this final step where the fish are brought in to a fish processing facility, a huge amount of fish goes from living fish to salmon servings. Within a process like this, a potential threat resides which is the dangerous listeria bacteria. To prevent this type of bacteria each fish processing facility has its own cleaning procedures to keep the facility clean for bacteria growth. This is usually solved with manual labor which forces workers to work in heavy duty conditions with a lot of chemicals and none-ergonomic work procedures.

NTNU has for years been in development with different companies to solve the manual labored fish cleaning procedure in fish factories. It is believed that an automated solution could potentially cover and clean the whole facility, by offline-programming robots to clean different surfaces. One of the major issues is that the old factory do not have any 3D models or layout for their current factory neither are any factory equal. To be able to program robots to clean surfaces, the accurate digital representation is required, which means 3D scanning the environment and creating 3D shapes to then achieve programmable surfaces. This is where this thesis comes in, to develop a procedure to create and test how this could be solved on a common platform. Then test it for offline programming to program cleaning procedures.

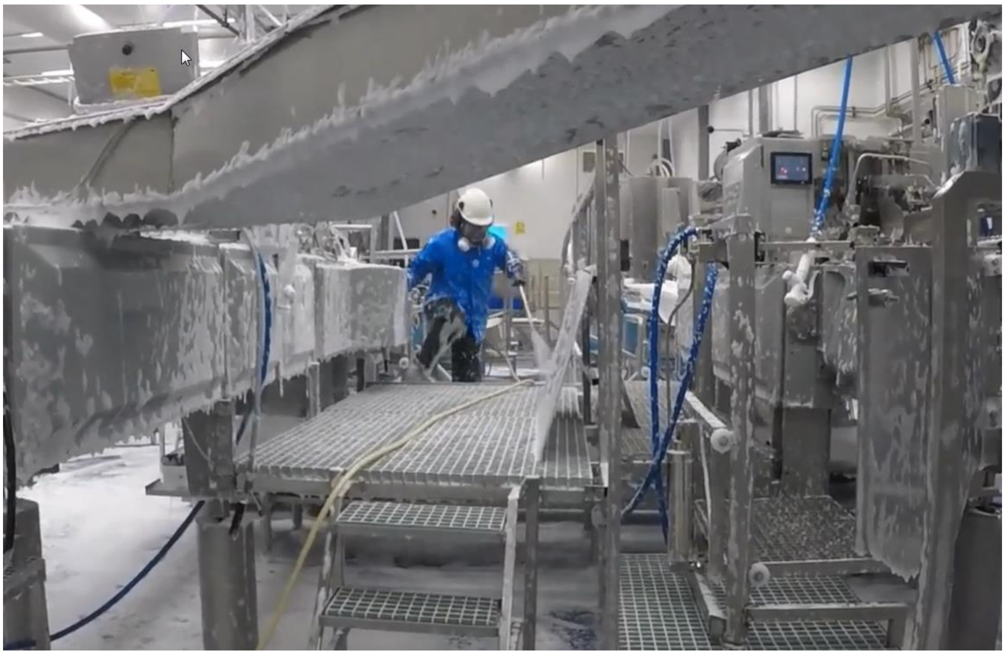


Figure 1 - Manual labor cleaning

3D scanning has been around since the 1960s but not commonly used as a commercial product before the early 21st century. The early prototypes for 3D scanners used lights, camera, and projectors which overall was not accurate nor time efficient enough to be used commercially [1]. Today 3D scanning is used in a whole lot of different fields including Archeology, construction, mapping, topography, city modeling, and general engineering. [2] Most commonly used 3D scanners are laser scanners which emit a laser beam at surfaces and captures the returning reflection to measure the traveled distance. [3] 3D scanning has a huge potential in capturing data to create 3D models of objects that do not have any 3D models or blueprints. Using this technique efficiently with a procedure and workflow to achieve a robot programmable surface could be revolutionary.

1.2 Problem definition

Today fish processing facilities are cleaned by hand, leaving a group of workers to work in heavy duty conditions. These facilities can be enormous and the time and effort to clean them properly is both times consuming and exhausting. Working with fish can potentially be rather dangerous if not cleaned properly. Listeria is a real threat in these facilities and without proper cleaning, an outbreak may occur. Knowing which equipment or conveyor are cleaned or not cleaned is nearly impossible to remember. There are so many surfaces and nooks and crannies, leaving it very difficult for the worker to remember if that side has been cleaned or if it was the other.

Most of the old fish processing facilities do not have any complete blueprint of their current layout nor any 3D models of their equipment. Creating an automatic solution for cleaning requires data from both the layout and 3D models. The solution that would normally be used, would be to measure the whole facility and re-draw every equipment and facility layout from scratch. This method could take months if not years to accomplish with constantly measuring the facility and building up new 3D models.

Not only would it take an extremely long time to keep measuring and drawing components from scratch, but the time constraints from the facility is also a problem. Due to the amount of money a fish processing facility generates during a day, shutting down the production to measure the facility is not acceptable. This means the only viable option for measuring would be to come back when the production is closed for the week, meaning weekends. Some fish processing facilities run from Monday to Friday and some even run during the weekends, making it even harder to measure.

Time constraint is one of the major issues creating an automatic cleaning solution. Even if the measurement and 3D models have been created, the installation of these solutions has to be quick as well. As with the measuring of the production, it will not shut down for installation either. Some of the larger fish processing facilities makes millions of NOK every day, and investment towards automatic cleaning which will shut down the factory for weeks/months is not profitable. One example is Marine Harvest which produced 5,5 million salmon servings every day combined (for five different factories) [4] and shutting down only one factory for a week could potentially lose 7 million salmon servings. [5] The amount of fish produced compared to what they have to gain with an automatic cleaning solution is uncalculable.

1.3 Scope

For this thesis, a set scope will be held to limit the viewpoints and narrow the objectives. Due to the time constraints of 6 months, not every part of a topic can be covered. The three main topics for this thesis will be to look at 3D scanning, solid generation, and robot simulation. For the 3D scanning topic, a closer look at how one can handle the data received from point clouds and how these are refined is essential. It is important to get a grasp of which scans are useful and how scanning in general works. Normally point clouds are used to generate a mesh which gives a visual representation of the object. Further study for how mesh generation can be achieved and refined to get any usable surfaces will take place. To create visual results, robot simulation will be used to offline-program robots. It is necessary to evaluate the simulation results and further testing. How can simulation be used efficiently to generate any results found from the analysis of the 3D scans and meshes?

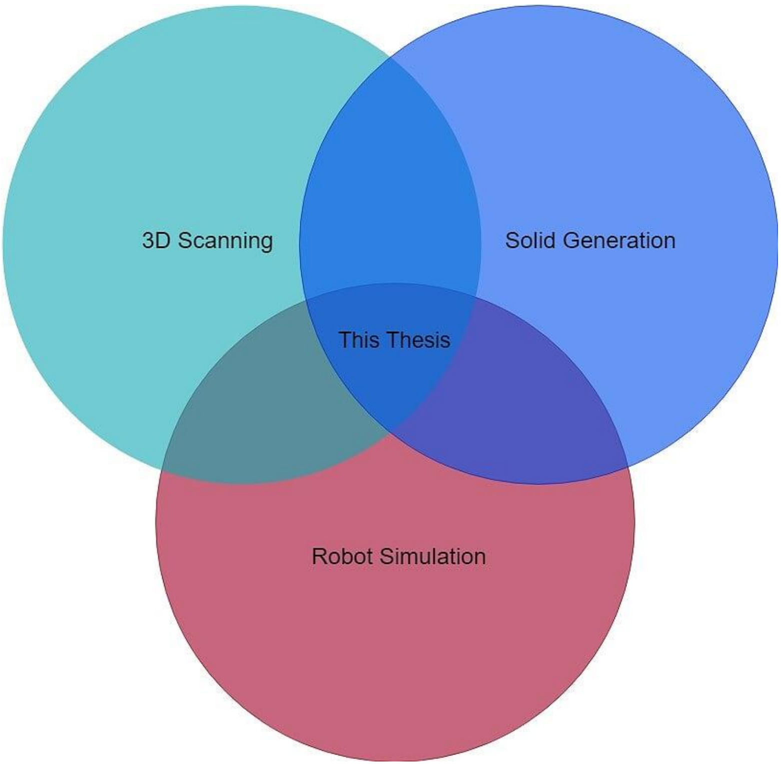


Figure 2 - Scope of the thesis

1.4 Motivation

Creating automated solutions for cleaning within fish factories requires the whole factory to be virtually represented. Most factories do not have 3D files of old equipment or layout, meaning it has to be 3D modeled from the ground up which could take months. By doing some 3D scans and converting them into programmable surfaces could possibly save months of work. Creating an efficient procedure can save so much time and effort, and potentially give the means to create automated solutions for already existing factories. Just by knowing that this is possible is a motivation in itself. The possibilities this could open for safer and more efficient cleaning can be game-changing.

1.5 Objective and Research Questions

The objective for this master thesis is to develop a procedure for how to create robot programmable surfaces from a point cloud, which can be used to offline program a robot cleaning solution for fish factories. The procedure will be further developed and tested on a common software platform for more efficient work. Further, a simulation cell will be set up, with the scans and the programmable surfaces from the procedure, to test if it is possible to achieve an efficient workflow.

- What are the procedures required to obtain virtual robot programmable surfaces from point cloud?
- How can the procedure be executed on a common software platform?
- How can the procedure help lower the time of on-site robot programming for robot cleaning of fish factories?

1.6 Structure of the Thesis

This thesis is divided into 8 main chapters, built up by a standard technical thesis outline.

Chapter 2 will present state of the art based on the scope of this thesis. The relevant theory captures the essence in 3D scanning, surface construction and the use of simulation.

Chapter 3 describes the methodology used for this thesis, explaining the action based, case study and discrete event simulation methods. A detailed flowchart for the workflow of this thesis, and assumptions and limitations in this thesis.

Chapter 4 introduces the case background and further detailing the given case study. Knowledge and background from where this thesis came from.

Chapter 5 explains the results found by using the presented methodology. This chapter will present a procedure developed to generate surfaces from 3D spatial data and workflow for how this can be executed on a common platform.

Chapter 6 will discuss the found results and combine previous found state of the art with the credibility with the methodology.

Chapter 7 will conclude the thesis and answer the given research questions.

Chapter 8 presents some ideas for future work and possibilities based on the findings in this thesis.

2 Literature Review

2.1 State of the art in Spatial data acquisition

2.1.1 3D scanning technology

In 3D scanning and spatial data acquisition, one commonly divides it into two major categories. The first category is the hardware. This is normally referred to as the tools for capturing point clouds. The important factors for hardware are how accurate, fast, reliable and cost efficient the tool can be. The other category is software, and this part contains the aftermath of scanning. The software should be able to process the raw data acquired from the hardware and further process it. By processing it that means remove noise and distorted data, integration of multiple scans and angles, some form of smoothing and simplification [6].

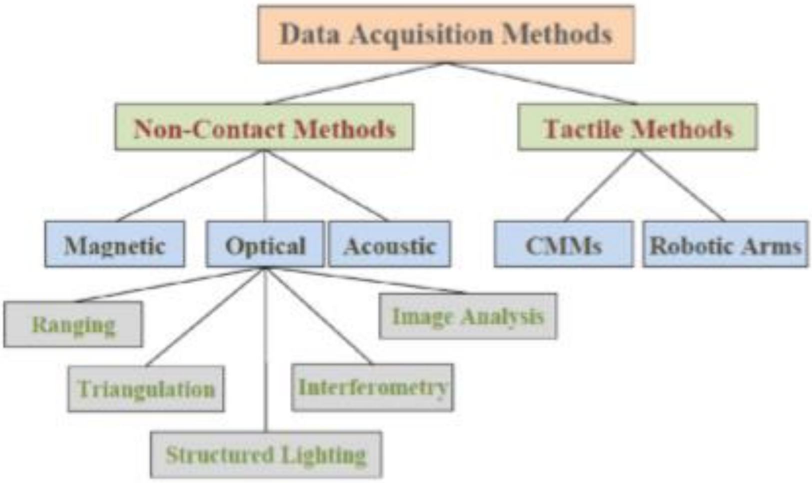


Figure 3 - Data Acquisition Methods [6]

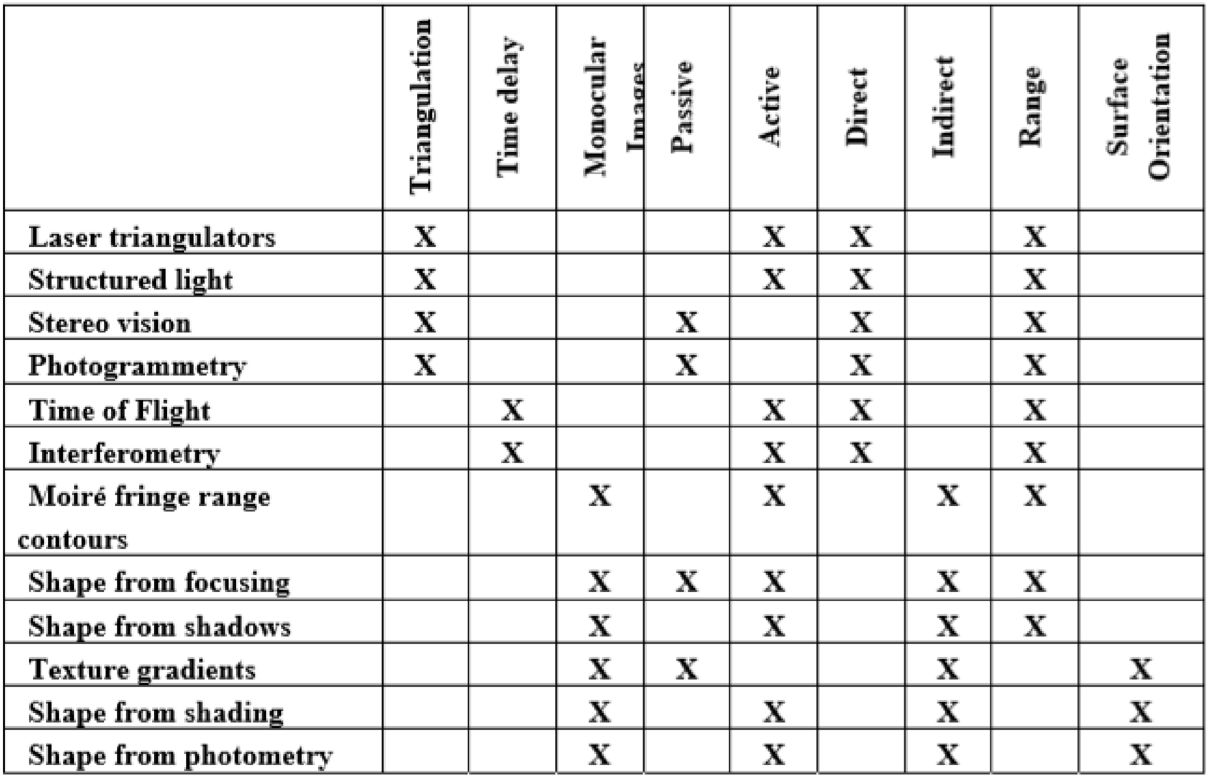
When discussing methods of gathering data two methods are commonly used, non-Contact method and contact method [6]. Contact method is rarely used these days due to the time consumption setting up a scan. Contact scanning or tactile method uses a contact probe to measure and gather data. This method is limited to the range of the robot controlling the probe or the general geometric of the object. The most commonly used method is the non-contact method. Within the non-contact method, as can be seen in figure 3 above, three sub-categories for non-contact is viable. One could either send out a magnetic field, light or even sound to capture data [6].

Further, within the non-contact method are two systems. These systems are known as passive and active systems. The passive system focuses on the object itself in different conditions. Examples of passive systems are shape-from-shading and shape-from-motion. Shape-from-shading is a single camera put up to measure the brightness of a surface where depth information based on reflection and brightness are calculated [6]. The advantages of using a passive system are that it is less sensitive to the environment, but it is very sensitive to other light sources and is limited by complex geometry. An active system, however, is actively measuring the object by either triangulation, time-of-flight or laser pulse [6].

Triangulation is based on triangulating a given spot capturing the coordinate x, y, and z with angles ω, φ, κ. This method is more commonly known as photogrammetry. The range for this method can be from a small machine to larger factories [7]. When operating with distinctive surfaces, photogrammetry is commonly used. But due to the overall time consumption and relatively low accuracy, it is not that efficient. Most point clouds gather by photogrammetry are very dense and heavy [2].

Time-of-flight or laser pulse is the most commonly used 3D scanning method which works from medium to long ranges and has an accuracy of a couple of centimeters or millimeters depending on scanner [6, 7]. A time-of-flight scanner sends out a laser beam which is reflected from the object back at a rotating mirror which registers the spatial data [8, 9]. These types of scanners have normally a shooting range of 360 degrees horizontal and 300+ degrees vertical. Some scanners are so advanced it can capture the reflection rate of the object and give it the color, based on how much of the laser beam is reflected [3]. The reason why this is the most commonly used method is for it's fast capturing. Depending on how much one would spend on a scanner, the scan rate can come up to above 25 000 points per second [10].

There is however a method which combines both the laser pulse and triangulation. By inserting another camera on a laser pulse scanner, the camera can gather color information about the object scanned with a laser. This in total can give an accurate scan faster with color information [3, 7].



	Triangulation	Time delay	Monocular Images	Passive	Active	Direct	Indirect	Range	Surface Orientation
Laser triangulators	X				X	X		X	
Structured light	X				X	X		X	
Stereo vision	X			X		X		X	
Photogrammetry	X			X		X		X	
Time of Flight		X			X	X		X	
Interferometry		X			X	X		X	
Moiré fringe range contours			X		X		X	X	
Shape from focusing			X	X	X		X	X	
Shape from shadows			X		X		X	X	
Texture gradients			X	X			X		X
Shape from shading			X		X		X		X
Shape from photometry			X		X		X		X

Table 1 - 3D Technologies [9]

As stated in table 1 every known and used 3D scanning technology is discussed and considered. This covers most scanning methods, and some of them are not relevant enough to be covered in this thesis. Therefore, a limited view is presented for what could potentially be useful.

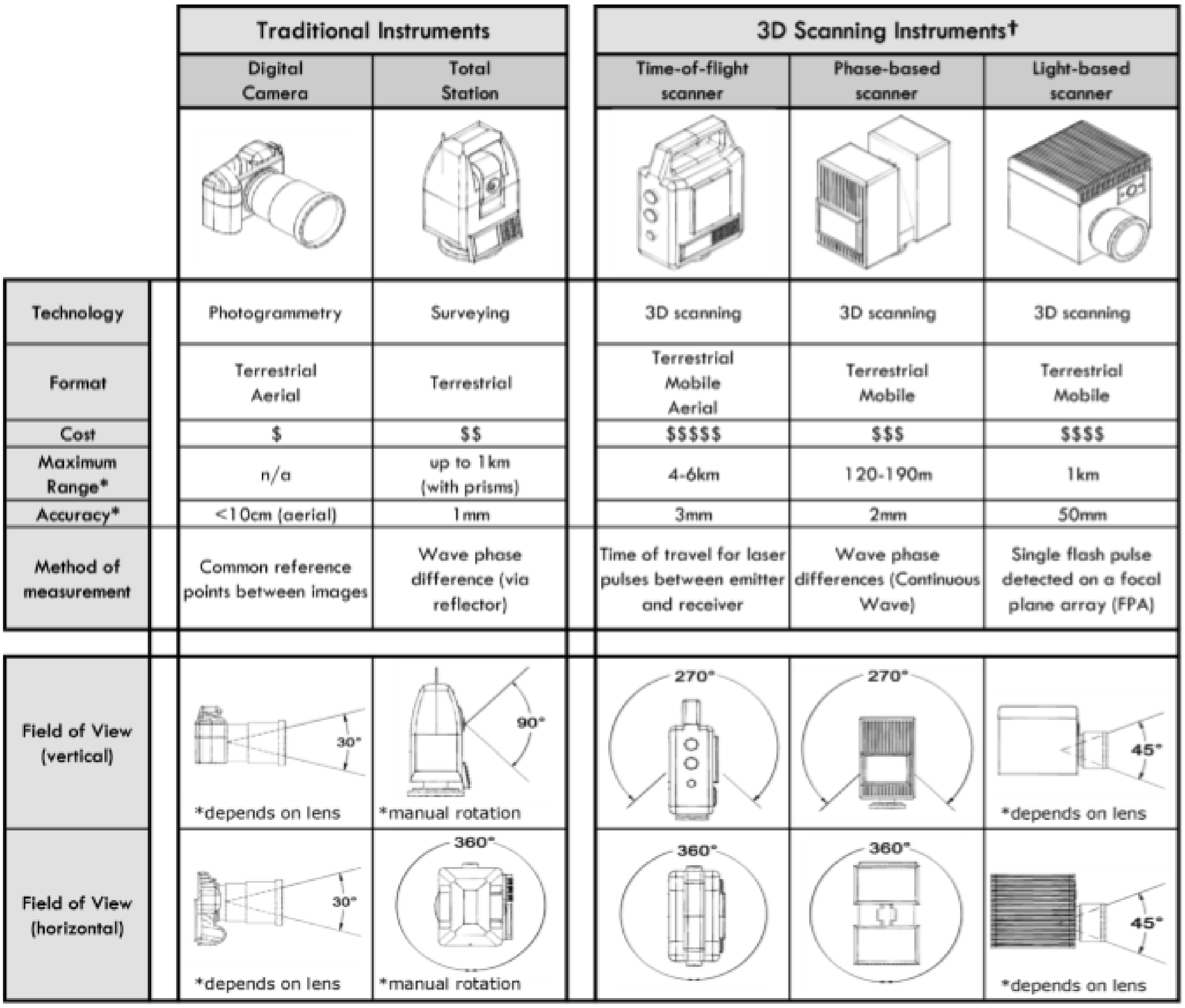
2.1.2 3D scanning techniques

To start out scanning certain requirements and preparations has to be in order. The first element to consider before capturing any spatial data is to define the main goals of the data capture. It is both important to grasp how much time and effort one could put into the scanning before a given deadline. Certainly, without a time limit, every component in every detail can be scanned very detailed, this is however never the case. A given time-frame is therefore important [11].

After having a general grasp of what to scan, a method of scanning has to be decided. By conducting field research tests, a given method can be obtained by inspecting the area, objects and environment [11]. Based on how large the scan and accurate the scan needs to be, a given tool can be found. It is unwise to invenst in any equipment before a field research test has been conducted. Not only is the area important, but also surfaces and environment. Is it outside, is it inside, is there enough light, is there a lot of dust, is it raining and so on. All of these questions have to be answered before any equipment can be bought and set up for scanning.

When a clear goal is in mind and the environmental concerns have been covered, the set-up for the scan can begin. Commonly when working with either large areas or areas which needs high detailing, reference points are set up [12]. The reasoning behind this type of reference points is to gather scans from different angles of an object or an area. It is, however, important to mark areas which can be seen from different angles, or else there is no point adding references. These points will make it easier later in processing to adjust and merge spatial data acquired from different angles.
Certain complex objects could have multiple reference points to cover all sides or even the whole object. The reference points need to be distinguishable from the rest of the scan, for missing two reference point in merging could potentially distort the scan data by multiple millions of points [3].

The capturing of the spatial data has to be under certain circumstances. One of the more important decision when taking scans is to let the environment be motionless. It takes time for the scans to emit laser to all the objects, so preventing movement within the scanned area is a necessity. Give the scan time to process and be patient is also key. The final step while performing 3D scanning is to process the data. This is where the fine tuning of the scans comes into play, where reference objects are used to merge the whole image together and remove noise from the image [12].



	Traditional Instruments		3D Scanning Instruments†		
	Digital Camera	Total Station	Time-of-flight scanner	Phase-based scanner	Light-based scanner
Technology	Photogrammetry	Surveying	3D scanning	3D scanning	3D scanning
Format	Terrestrial Aerial	Terrestrial	Terrestrial Mobile Aerial	Terrestrial Mobile	Terrestrial Mobile
Cost	$	$$	$$$$$	$$$	$$$$
Maximum Range*	n/a	up to 1km (with prisms)	4-6km	120-190m	1km
Accuracy*	<10cm (aerial)	1mm	3mm	2mm	50mm
Method of measurement	Common reference points between images	Wave phase difference (via reflector)	Time of travel for laser pulses between emitter and receiver	Wave phase differences (Continuous Wave)	Single flash pulse detected on a focal plane array (FPA)
Field of View (vertical)	30° *depends on lens	90° *manual rotation	270°	270°	45° *depends on lens
Field of View (horizontal)	30° *depends on lens	360° *manual rotation	360°	360°	45° *depends on lens

Figure 4 - Instruments used to capture spatial data [11]

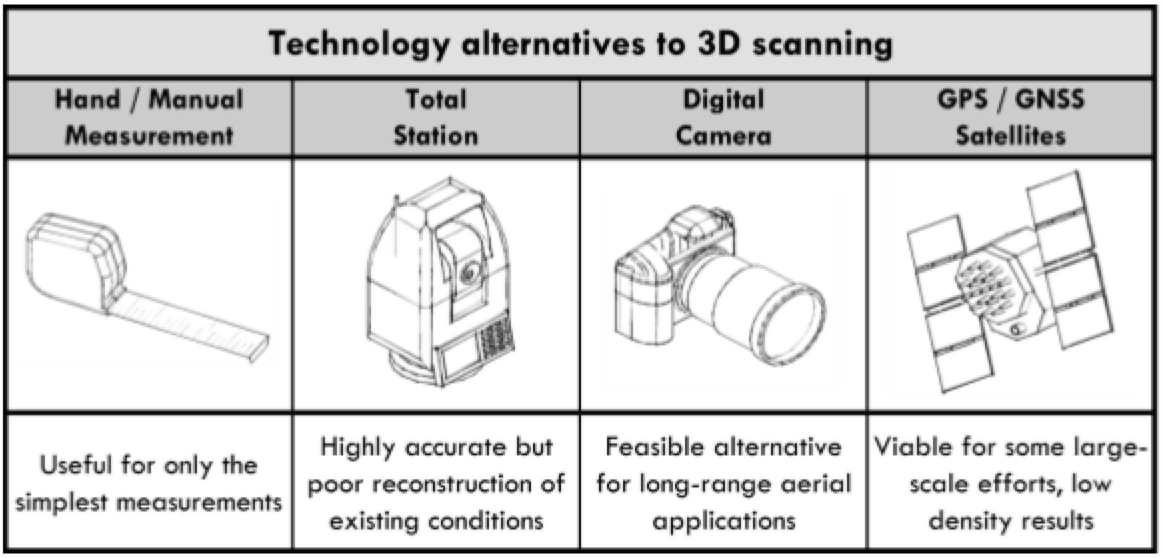
Figure 4 represents a variety of ways to capture spatial data, where the three most right ones are considered 3D scanning instruments. Most of these have already been covered earlier in chapter 2.1.1, but what has not been discussed is what type of format these scanners can be used for. The most common formats used are Aerial, Mobile and Terrestrial [11].

Aerial are most commonly used with triangulations or photogrammetry. Even though it could possibly be used with time-of-flight scanners, but less common. Aerial is usually used for scanning large open areas, such as pictures for Google Earth. In [2] a couple of good examples can be drawn from what is viable for aerial.

The mobile fits all 3D scanning instruments. These often are a moving camera on a vehicle gather spatial data all around itself. This can be used in road mapping and general outdoor environment.

Terrestrial is the most standard and well-rounded method of capturing 3D spatial data. There is really no limit to what and how it can be used, but it has to be stationary at the ground for a certain timeframe to capture the scan. These could typically be used indoor to visualize a new facility or a new production line [13] or even be used in architectural 3D modeling [11].

2.1.3 Alternative to 3D scanning



Technology alternatives to 3D scanning			
Hand / Manual Measurement	Total Station	Digital Camera	GPS / GNSS Satellites
Useful for only the simplest measurements	Highly accurate but poor reconstruction of existing conditions	Feasible alternative for long-range aerial applications	Viable for some large-scale efforts, low density results

Figure 5 - Alternative tools to 3D scanning [11]

Starting from the left, a handheld measurement tool could potentially do some form of measurement but how accurate would these be? Any major component or equipment would have some sort of variable surface which would become nearly impossible to measure at a high accuracy level. For some reference and quick measurement this could be used, but for anything that requires it to build up a digital version would be hopeless.

Total stations, however, are a more reliable source of measurement, but it lacks the speed and is cumbersome. Finding a spot of reference or a single measurement can be achieved with high accuracy but trying to represent a component in 3D is too slow. Could potentially be a supplement to 3D scanning but not a real alternative to it [11].

Digital cameras are one of the closest alternatives to 3D scanning. The amount of development with lenses and cameras has actually put this into a viable 3D scanning technique. A lot of people have changed their view to accept this as viable [2]. While others still regard it as not a 3D scanning technique. The main advantage with digital cameras is aerial shots, just have a look at Google Earth where this technique is used [14]. It could also be used for smaller objects and close-range photogrammetry. Mostly why this technique is not that commonly used on an industrial level, is because most 3D scanners these days already have a digital camera on top of the 3D scanner. This creating a multipurpose tool [11].

GPS could potentially be used for large area scanning but would most likely have a limited amount of points. Measuring terrain could work but achieving the points might be harder than expected. Anything indoor as well is hopeless to capture [11].

2.2 Guidelines for Simulation

2.2.1 Simulation theory method

A simulation model is a set version of the real world, but a more simplified abstract version with set scope and limitations. These types of models might even be easier to analyze than the real system due to it being slow, safe and inexpensive [15]. Some systems could potentially use days or weeks to perform a given task and having to sit through that process every time to capture data is a huge waste of time. Not only would the time be wasted but some systems are so complex that actually visualizing and analyzing the change or performance could be impossible. The simulation uses these models to understand and analyze results in a much smaller timeframe and doable environment.

There are two main categories of simulation. The first category is man-in-loop, this cover mostly simulation where a given scenario for training or entertaining purposes is performed. By having a safe environment for pilots, engineers or even chemist to test out solutions or scenarios that might occur, one can give training and get experience. The second category is the analytic one, this is normally what the technical domain calls a simulation. Here mathematic models solved by algebra and calculus is used to determine the results of the given scenario. These give various forms of results, depending on the input and how the model is set up [15].

Input	Actions on the system
State	Input causing changes
Output	Measured quantities
Entities	Affected by inputs, flows through the system/simulation
Attributes	Characteristics for entities
Activities	Processes in a simulation
Events	When an entity performs an activity
Resource	Has a constrained capacity
Variables	Anything with variables
Time	The time aspect of the simulation

Table 2 - Simulation components [15]

When looking at simulation models certain components, as stated in Table 2 above, is necessary to understand what the model does, and how to use it properly. In general, the output is what measures the results from the model. Within the model *entities, attributes, activities, and events* form most of the model for how it will act and react. The input could be numbers, orders, actions needed to start out the simulation. *Resource, variables and time* are all aspects that can be changed to look at certain scenarios. Changing how much is in stock, how fast it will run, how long will it run and so on.

All of these components are interconnected with each other in some form, creating a chain or a balance between each other. This will eventually be put together based on a real-world system, often a simplification, to measure something desirable. Setting up the correct connections and behavior is important for the model to run as intended.

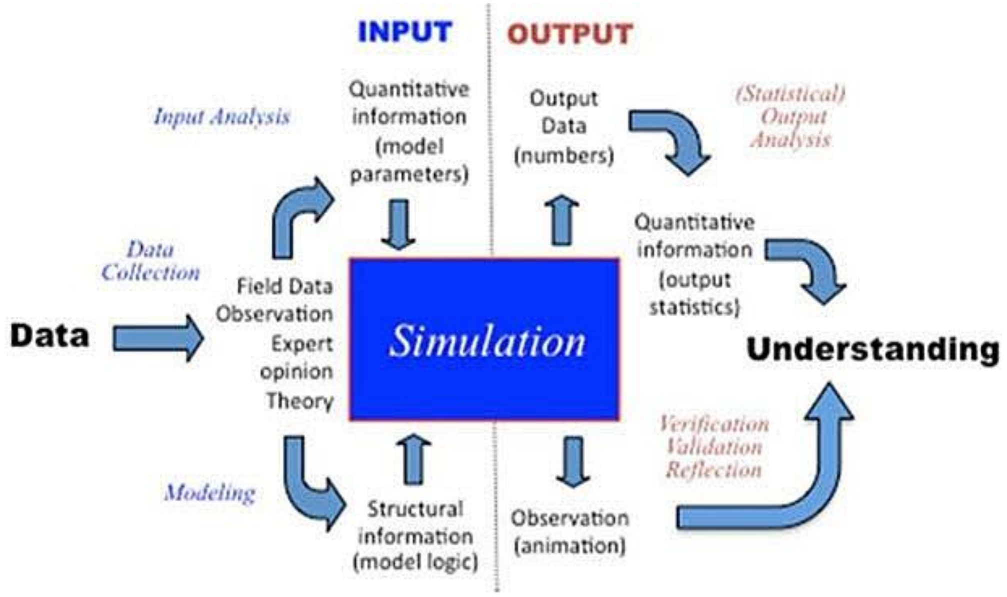

INPUT
OUTPUT
Input Analysis
Quantitative information (model parameters)
Output Data (numbers)
(Statistical) Output Analysis
Data Collection
Quantitative information (output statistics)
Field Data Observation Expert opinion Theory
Data
Simulation
Understanding
Verification Validation Reflection
Modeling
Structural information (model logic)
Observation (animation)

Figure 6 - Model of simulation [15]

Figure 6 above shows how simulation is put together. In one end Data from the system flows in, either by manual collection or some sort of system. In the other end, understanding is the desired outcome of the simulation. The simulation is built on models, data inputs and logic which is represented to the left of the dotted line. This is the building and input stage of simulation. What comes out of the right side of the dotted line is normally data and quantitative information. This information and data are gathered and analyzed to get an understanding of the current simulation and model.

2.2.2 Simulation procedures

Kenneth J. Musselman has come up with a guideline for success in a simulation project. In his paper "Guidelines for Simulation Project Success" [16] is fairly known in the simulation community and is highly appreciated. The paper contains 8 steps for what is important to build up a simulation and use it properly. It does not contain any technical challenges to a simulation, but rather a general perception of how one should use the simulation as a tool in the ongoing project.

Project steps and guidelines

There are steps for how simulation should be set-up which is: problem formulation, model conceptualization, data collection, model building, verification, validation, analysis, documentation, and implementation [16].

1 Problem formulation

"Define the problem to be studied, including a written statement of the problem-solving objective" [16].

Have a written document of the problem with objectives and work on the right problem. Write proper documentation with understood problems and objectives to avoid working on things not relevant. Listen to the customers need, let them talk before you answer. Understand the problem and their views before judging. Important to ask questions to not get any vague or unclear answers to the problem or objective. Communication is important as well, keeping the customer close for guidance and validation.

2 Model conceptualization

"Abstract the system into a model described by the elements of the system, their characteristics, and their interactions, all according to the problem formulation" [16].

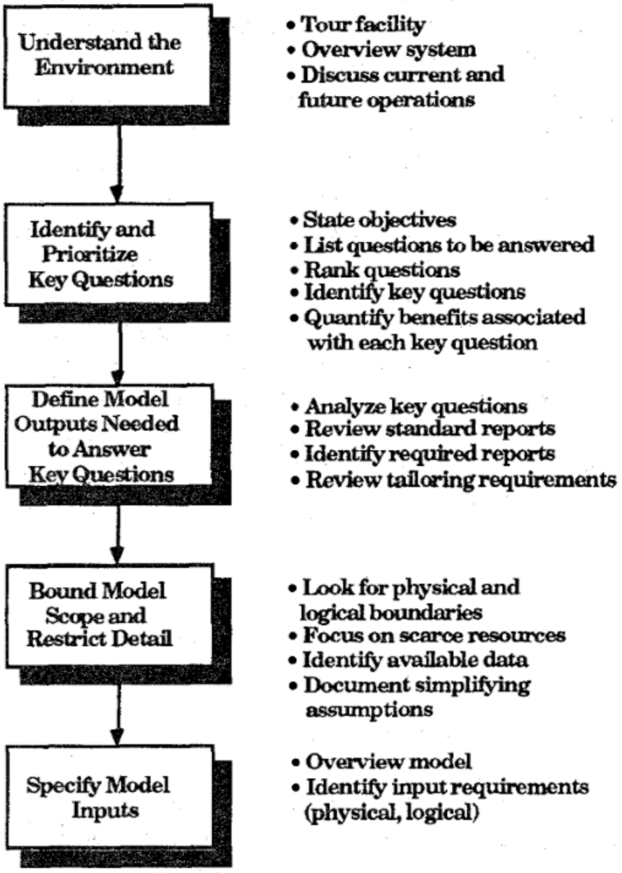

Understand the Environment
• Tour facility
• Overview system
• Discuss current and future operations
Identify and Prioritize Key Questions
• State objectives
• List questions to be answered
• Rank questions
• Identify key questions
• Quantify benefits associated with each key question
Define Model Outputs Needed to Answer Key Questions
• Analyze key questions
• Review standard reports
• Identify required reports
• Review tailoring requirements
Bound Model Scope and Restrict Detail
• Look for physical and logical boundaries
• Focus on scarce resources
• Identify available data
• Document simplifying assumptions
Specify Model Inputs
• Overview model
• Identify input requirements (physical, logical)

Figure 7 - Model conceptualization [16]

A clear set of objectives will eventually give criteria for the simulation model. These objectives should contain clear questions needing to be answered. From Figure 7 above Musselman had developed a procedure to build up a simulation model. He states that working on the procedure backward will give a leaner model and quicker initial results. Instead of starting out with inputs and working your way through the system to get output, start backward as figure 7 shows [16].

3 Data collection

"Identify, specify, and gather data to support the model [16]."

When gathering data always question the data. Before entering it blindly into the simulation, consider where the data came from and what it tells you. Are these data within the scope and necessary for the simulation, or is it too detailed? All of these questions have to be considered when working with data, one miss click, or small error could potentially give a very inaccurate output. Also, if the data collected is not enough and there is no legitimate way of obtaining them within a given time-frame, assumptions can and will be used. Based on your knowledge of the system an educated guess can be made to move the project and simulation forward. These assumptions have to be noted and tested later with perhaps other data or assumptions when the simulation model is complete. This way one can almost verify an assumption [16].

4 Model building

"Fit the conceptualized model to constructs available in a simulation language or system" [16].

Model building is to fit the conceptualized model already established with a simulation in either a language or system. Musselman states that in a project, building up a model takes most of the priority, meaning it will shift the goals for the project [16]. This could mean problem solving and solutions become sub priorities compared to building simulation. He suggests more time thinking about the problems and solutions that could come with it, and less on the building.
Start out by a simple model, is the correct inputs and build up as they should? Try to capture the essence of the model in a smaller case. Building a complex one at once will undermine your ability to understand the outputs [16]. Also building a simpler one will not end in huge restrictions for improvements. A complex one will not have much room for adjustments and improvements.

"Don't let the model become so sophisticated that it compensates for a bad design, or so complex that it goes beyond your ability to implement." [16]

Complexity will eventually come into a model but making the model too good will hide problems not visible due to a well put together model. Some models might be intelligent enough that the improvement is visible in the model, but hard to implement for a project. Be realistic [16].

5 Verification and Validation

"Establish the model executes as intended and the desired accuracy or correspondence exists between the model and the real system" [16].

Be aware that requests for change will eventually happen. By changing just, a fraction of the original plan, many more changes might come from the result of it. Look within the set scope and objectives and only change if it really is necessary. If you, however, do want to please the customer, get your model to a state it can produce baseline results before implementing any changes. In this way, both sides are cover, the original objectives and scope and also the request [16].

6 Analysis

"Define the problem to be studied, including a written statement of the problem-solving objective" [16].

Check and question the output from the simulation. Do they make sense, are they reasonable? You need to believe in your outputs as much as in yourself, if anything is off, check the assumptions, reverify the model and check inputs. If it doesn't make sense check it out! It is also important to grasp what is the limitation of your model and simulation. It is not replacing reality and at best could be less than reality [16]. A solution for the model is not necessarily the answer to the real system.
After a while, improving the simulation and the model is more or less pointless. Sure, the outputs could be refined even more, or the answers could give analytic stats, but in the end, a certain endpoint is necessary.
In the end present a couple of suggestions for the customer to select, for possible ways to solve the problem [16].

7 Documentation

"Supply supportive or evidential information for a specific purpose" [16].

It is important to keep documentation and logs for further projects. Not only can certain assets be re-used but the experience and model build-up could potentially be pretty close to another case. By using a certain program for simulation, program documentation is also desirable. This will keep the knowledge for the software and program to be run some other time with guidelines and helpful information [16].

8 Implementation

"Fulfill the decisions resulting from the simulation" [16].

Musselman states that after a simulation project, the team can look back at great work with building a good model and simulating it to give the desired answer. But it also shows how well the team functions together. Key personnel with communication, discussion and decision making are important for a project like this. Without those many complications could go wrong [16].

2.3 Existing approaches for 3D scan to 3D solid

2.3.1 A theoretical framework for generating solids from spatial data

Generating surfaces or solids can be achieved in multiple ways. In [6] a bunch of different hardware and software has been found and which of them are connected. Normally a supplier of hardware has some sort of corresponding software to either handle scan data or solids. The author in the paper then further explains they did not take any given software but developed a thought process for what has to be solved to achieve a solid model from scan data.

Data filtering is perhaps one of the more important functions or processes needed. By having a tool to remove noise, distorted data and invalid data it can greatly reduce the overall workload further down the process [6]. Before any work can be done on a scan, data filtering is a must or else the generated mesh or surface will eventually be distorted and perhaps even wrong in terms of scale or boundaries.

Data integration is important to gather and merge different views of a scan [6]. Normally a few different scans are taken when wanting to generate 3D objects. These can vary anywhere from two scans to a couple of tenfolds. When scans have been filtered, it is a necessity to have them merged correctly with corresponding reference points. Without a complete merge, distances and edges might not be where they intentionally should be. This could create a massive amount of error when generating solids.

Data segmentation is a way of creating smaller segments to work with [6]. This is mainly done to lessen the computational power needed for that segment, or to distinctly split an object from the main scan. This is somewhat close to what Data filtering does, but it splits it into chunks that are workable.

Surface reconstruction is the next step using the raw spatial data now filtered and adjusted to generate surfaces [6]. This is where the start of a 3D object begins. By constructing surfaces and edges from spatial data, 3D solids can be created and used in varies forms.

Data simplification and smoothing takes the surfaces and either simplifies it to lessen the computational load or smooths the surfaces and edges [6]. This will create a solid model which is comprehensible and eventually will be used for comparison, visualization or simulation.

2.3.2 Industrial use of 3D scan procedures and software

There are multiple cases where 3D scanning into 3D solids have been used. There are three cases which have been selected due to containing workflow for generating solids from spatial data and the amount of software needed to obtain solids. Both relevant for this thesis to obtain knowledge for how this has been solved in both other software and which workflow is used.

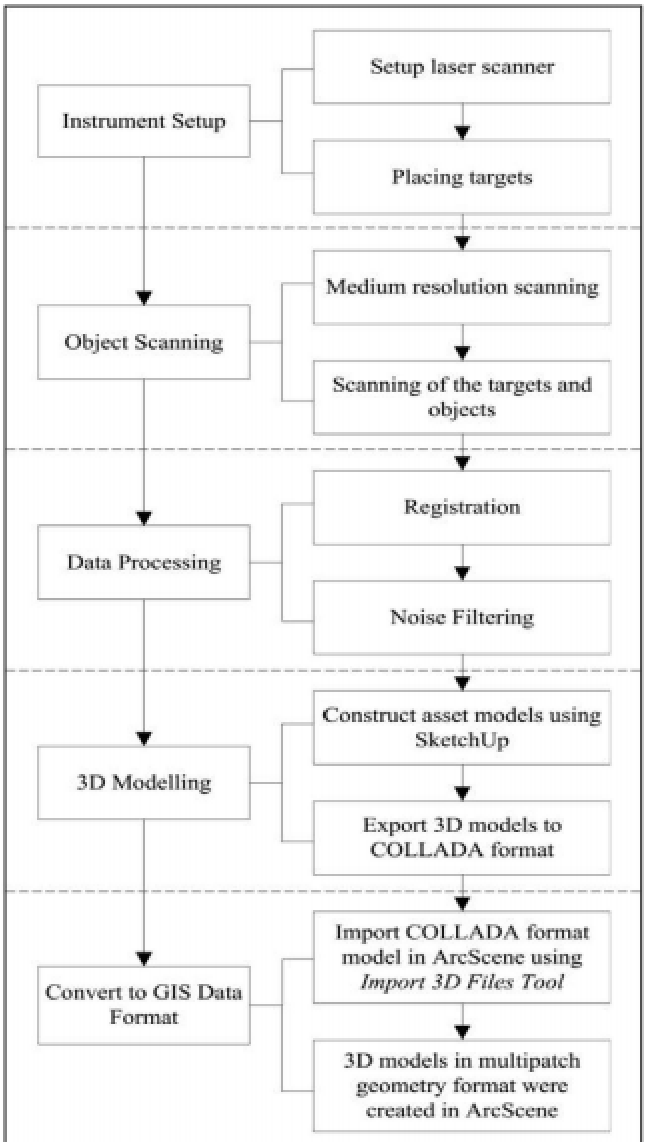

Instrument Setup
Setup laser scanner
Placing targets
Object Scanning
Medium resolution scanning
Scanning of the targets and objects
Data Processing
Registration
Noise Filtering
3D Modelling
Construct asset models using SketchUp
Export 3D models to COLLADA format
Convert to GIS Data Format
Import COLLADA format model in ArcScene using *Import 3D Files Tool*
3D models in multipatch geometry format were created in ArcScene

Figure 8 - Workflow for indoor assets [7]

The first case is the use of a terrestrial laser scanner indoors in a data laboratory. The main idea is to capture spatial data from within the lab from four scans and then turning it into 3D solids [7]. Figure 8 above shows the general workflow for what was done to achieve a 3D model of the lab. From figure 8 five steps are needed to achieve a 3D model.

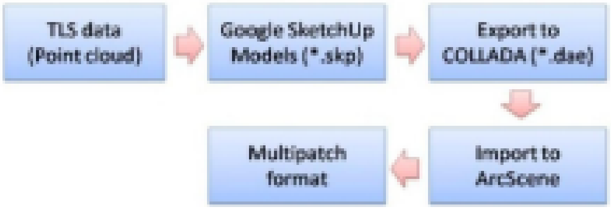


Figure 9 - Software procedure [7]

The given workflow is commonly used in different aspects and fields which will be covered later, but the amount of software needed to achieve a 3D model is shown in figure 9. TLS (Terrestrial Laser Scanner) data is captured and imported into the scanner's software Cyclone [6], in this case, a Leica Scan Station C10 was used. After filtering the noise, it was then exported into Google SketchUp where surfaces were generated. Later the SketchUp file was exported to ArcScene in a COLLADA format to create a whole 3D solid model of the computer lab [7].

In this paper, the authors present a workflow for creating 3D solids from spatial data, with detailed step for what is required, but also how many software are required to achieve the given results. In total, three different software were used going from raw cloud data to a solid 3D model.

The second case coming more from an industrial perspective challenges the use of digital factories. In building newer models of products and developing faster methods of creating goods, a digitized form of their layout is almost required [17]. The author expressed this by giving a few examples of how much this could impact the overall production and also highlights current problems and challenges with this type of technology. It comes down to either having some sort of blueprint and 3D models for the current layout or use the newer technology such as 3D laser scanning. The main issue with using 3D models could potentially be in precision. While the models might be very detailed or contain very few details, placing the object in a virtual environment could prove difficult. With some basic and simple measurement, it could be rather close, but there might still be some centimeters off, while the 3D scanning method often comes down to millimeter tolerance [17]. This means that doing it with 3D scanning could give more precise results for how the digital factory should be set-up.

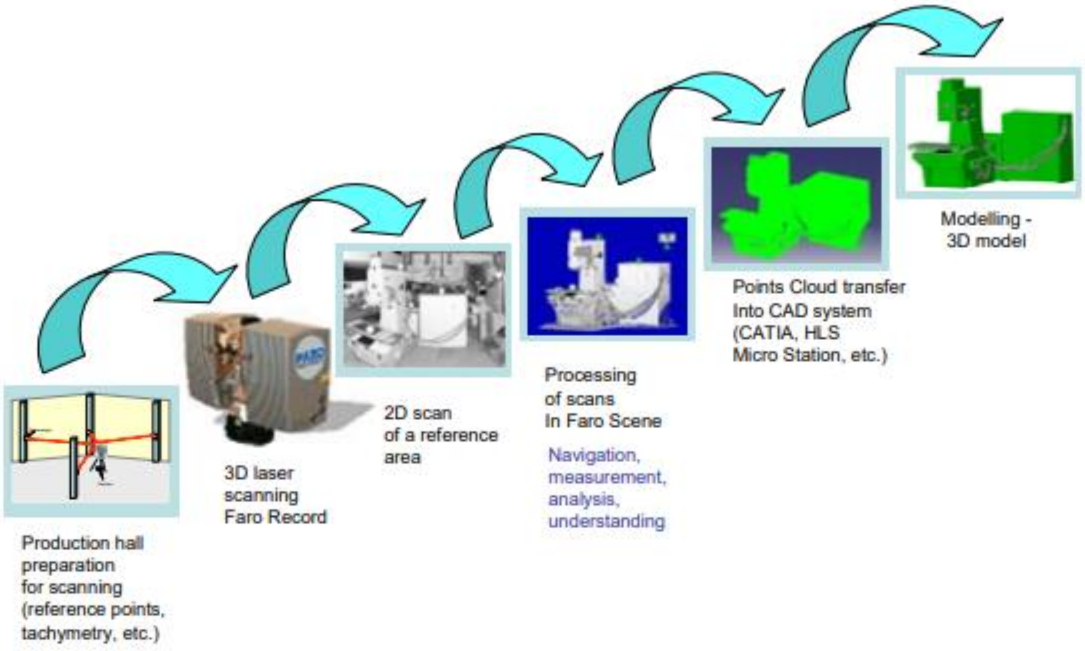


Figure 10 - Industrial workflow [17]

Figure 10 above shows the general developed procedure for an industrial digital factory. This is somewhat similar to other developed procedures but covers other types of software. The author states more software used than figure 10 shows [17]. Using a FARO scanner will eventually lead to the use of their software, which in this case was FARO Clouds to collect the point cloud data, and FARO Scene collection and export. Further Polyworks was used to polygonization of 3D digital models to mesh the current clouds. Then it was imported into a CAD system, mentioning some alternatives such as CATIA, HLS, Micro Station, for surface creation eventually leading to a 3D model.

In this paper, the author uses a minimum of four software going from raw point cloud data to a finished 3D model of a given product in their production hall. Some of these could be skipped due to other software being more efficient than others.

The last case is a less industrial case but proves that generating solids can be solved in a simpler way with fewer software tools for reverse engineering. The paper presented contains the scan of a radio volume button which will be scanned, generated solids and then 3D printed [18]. For the sake of this thesis only the first two parts are important, which has been solved using the software SolidWorks. SolidWorks is a software delivered by Dassault Systèmes which focuses on the CAD aspect of the software. The authors state that the only software used to generate the solids required for the 3D model in SolidWorks, which is then exported into a 3D printer to be printed.

The work procedure is not really fleshed out in this paper, but it contains the same way as the industrial ones. By scanning, meshing then generating surfaces a 3D model of a radio volume button can be created in just one software.

2.3.3 3D solid workflow for civil engineering

There are cases in civil engineering where 3D scanning has been used to both verify architectures and control on on-going buildings. In a civil engineering project, there is a certain difference from normal production. Ways of dividing the building structure and set-up called BIM is often used but could be referred as to what production knows as

MBOM. This means that the current way developed in generating spatial data for civil engineering methods can be applied elsewhere.

The BIM Task group has created a guideline for how this can be used in practical ways. 3D data collection could be used for an unlimited time to get every little detailed scanned, but in all cases, time is of a constraint. This means that certain scans or details have to be left out, and that could be a challenge in itself choosing what stays and what goes [11]. That is why the BIM task group has developed five main questions that need to be answered before continuing collecting spatial data. The questions are as follows:

- *What is the detailed goal that you are trying to achieve with this effort?*
- *What are the budgetary constraints for solving this particular problem?*
- *What type of deliverable is needed to achieve the goal of the effort?*
- *What local qualified and experienced service providers can perform this service?*
- *How will the point cloud data set or other deliverables be used in the future?*

After these questions are answered, first then can a collection of spatial data start. Depending on project certain objectives might be larger than some, but in the end, a general level of effort can be obtained and usually follows as figure 11 shows. The most time spent on a 3D scanning project is the process of the aftermath of the spatial data. This could mean fixing point clouds or refining mesh. Depending on software and skill these level of effort could be tweaked a little, but would mostly remain the same throughout the project.

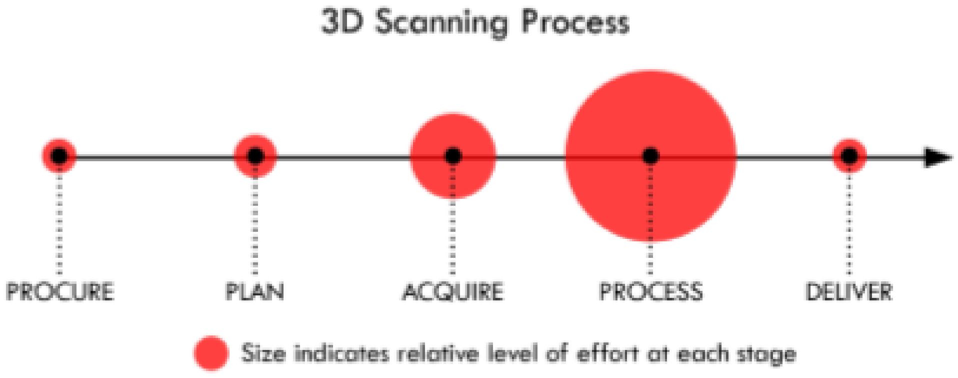


Figure 11 - Level of effort in a 3D Scanning process [11]

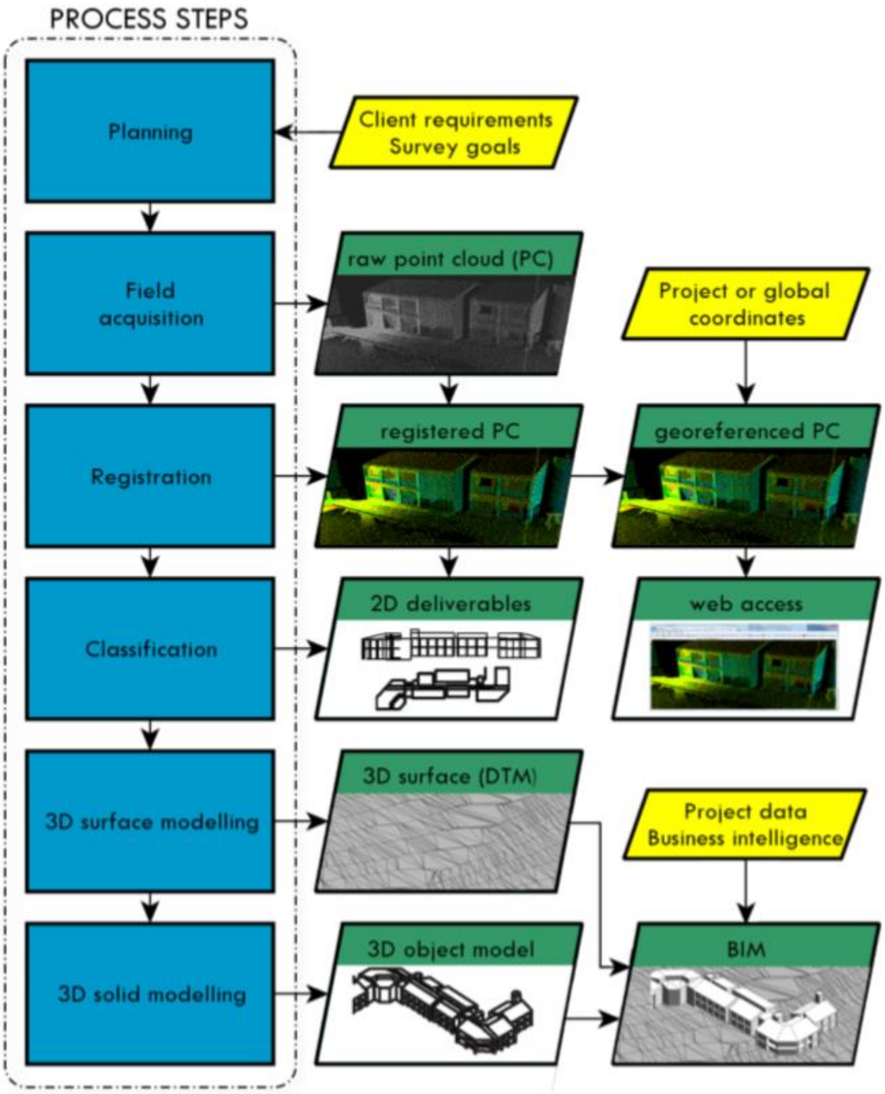


Figure 12 – BIM 3D Solids procedure [11]

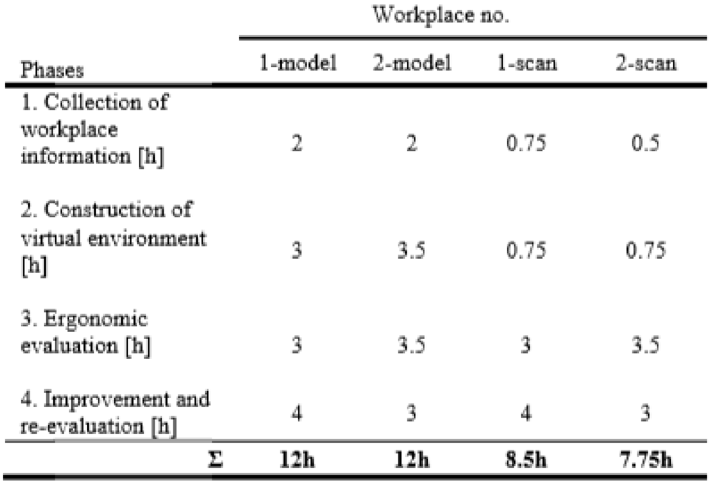
As stated by the BIM task group, the level of effort for how much time is needed and a general effort put into that specific task is found, but they also developed a general procedure to generate solids or 3D models from spatial data. Figure 12 above shows the general steps needed in a civil engineering project to properly use spatial data.

Most of this procedure in any field starts out by planning and field acquisition. It is important to grasp which component, wall, structure, and object are vital for spatial data. Depending on project or interests these can vary, but the procedure for capturing and setting up the scanner goes the same way. After capturing the required data, the step known as registration for civil engineering begins. This is a way of rearranging all data to a common global (or local depending on the situation) coordinate system [11]. Based on other drawings or project coordinates, all spatial data and scans are registered in the same orientation. Once all of these are registered in the right orientation, classification starts. The given data gathered from the 3D scanner only contains positions of the points (and perhaps color), which makes it hard to differentiate walls from ceiling. Therefore a classification is initiated to keep track of which points contains what features, making it easier to segment [11]. When the classification is successful the next big step can continue, which is adding surfaces to the spatial data. To achieve surfaces a mesh has to be created in order to generate a flat surface from the spatial data [11]. After every spatial data is converted into meshes, surfaces can be generated.

At this point, there are segments from different rooms segmenting floor from walls with meshes which has been turned into surfaces. The last step in a BIM is to add these together to create a 3D solid model [11]. Since all of the segmented parts now have surfaced, adding them together will create a 3D model of the scanned data.

2.3.4 A study comparing 3D scanning versus 3D modeling

A study has been conducted comparing 3D scanning and 3D modeling in terms of speed in an ergonomic analysis. The authors state early in the paper that the expected results would be that 3D scanning is faster than 3D modeling. To test this, they took a workstation and told an engineer to create a 3D model, while the authors worked on scanning the workstation.



	Workplace no.			
Phases	1-model	2-model	1-scan	2-scan
1. Collection of workplace information [h]	2	2	0.75	0.5
2. Construction of virtual environment [h]	3	3.5	0.75	0.75
3. Ergonomic evaluation [h]	3	3.5	3	3.5
4. Improvement and re-evaluation [h]	4	3	4	3
Σ	**12h**	**12h**	**8.5h**	**7.75h**

Table 3 - Time spent 3D modeling and scanning [10]

As table 3 shows, the amount of time the engineer spent to measure the workstation was around two hours, while the time spent on scanning was about 45 minutes. Processing the scan took about the same time, but the modeling of the workstation took about three hours. In total that means creating a 3D model from no blueprint would take roughly around five hours and scanning plus processing it takes about one hour and 30 minutes [10]. The rest of the numbers from table 3 are neglected due to not being relevant.

The authors reflected on their findings and found out a couple of important things. Regarding 3D scanning as a tool for generating 3D models, it actually gave a more consistent model. It was nearly 1 to 1 in size and distance, while the 3D model built by the engineer was close to its original but not perfect [10]. This was due to weird angles and brackets placed in very odd distances from another. Another thing that got their attention was comparing the 3D model from the engineer and the processed 3D scanned model. They found that certain surfaces and edges are not 100% accurate for the scanned model, due to distorted points. Some bins are not totally square, and the table had a weird edge at one corner [10]. All in all, the 3D model had better quality, but the 3D scanned model was more precise and accurate.

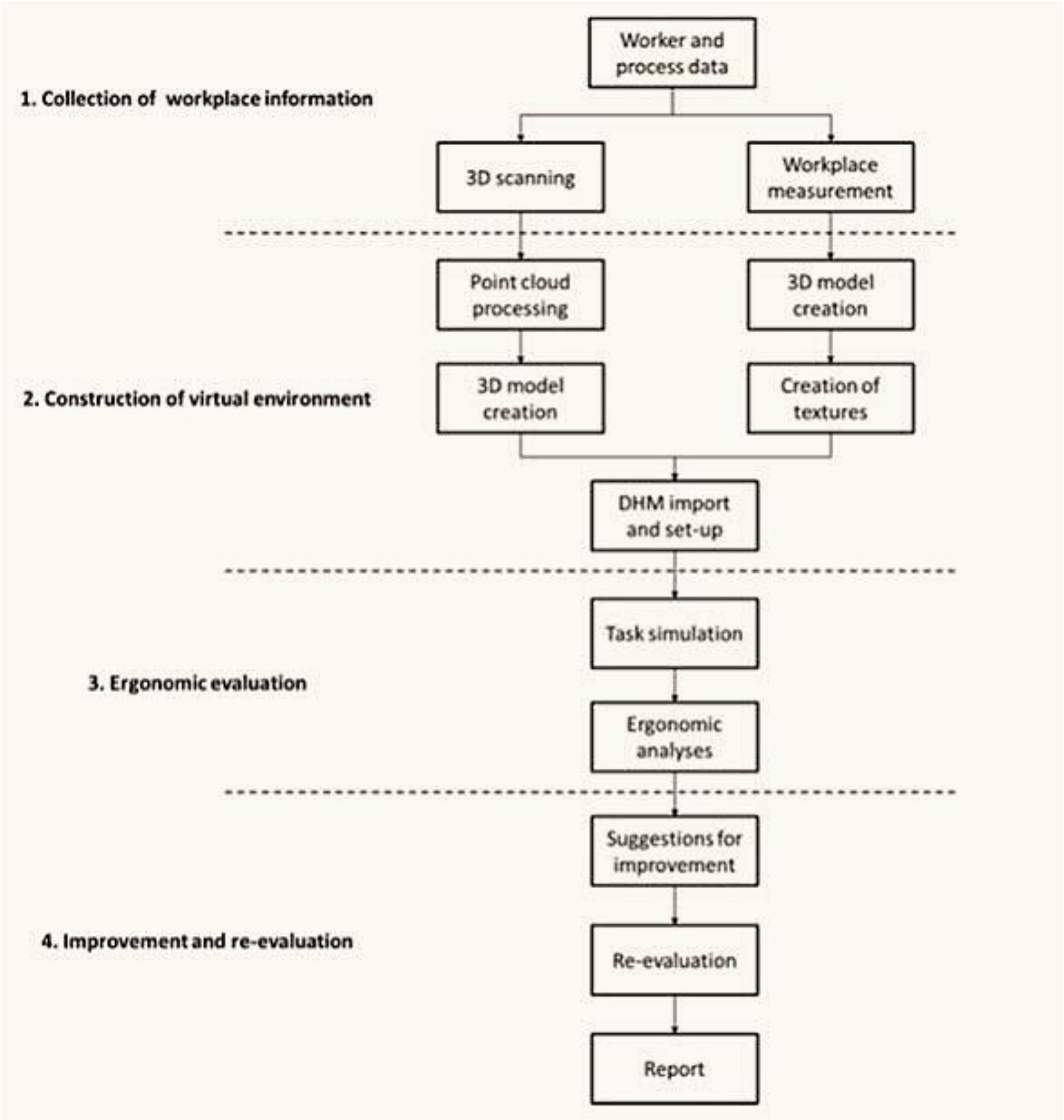


Figure 13 - Workflow achieving 3D solids [10]

Further, the authors presented a workflow for both how the 3D scanning should take place and how to create a 3D model from scratch. Figure 13 above shows four steps, where the two first are the interesting part. They both start out with some form of measuring. For the normal method, a ruler or laser guider could be used to measure every angle and distance required to build a 3D model. The 3D scanning is mostly a set-up of a scanner and scanning, potentially using reference objects for larger scans. This is the first step in the workflow named Collection of workplace information.

The second step which differentiates the two methods by either first processing the raw data gathered from scans, or actually starting to create the 3D model itself. Spatial data always needs some form of processing before it can eventually be used as a 3D model, and this step is normally what takes time in 3D scanning. For the 3D modeling a lot of reference points, planes and axis will be needed to fulfill the 3D model in an accurate way. This is also heavy work which needs a lot of time spent before the actual model can be drawn.

Both methods use more or the less the same method of achieving a virtual 3D model but doing it the old faction way one starts out with an empty sheet containing no information what so ever, except the one gathered by hand. The 3D scanning method has spatial data which can be refined to then be usable to work with. For this research, a small workstation was tested and eventually led the 3D scanned model to be finished over twice as fast as the regular 3D modeling method. How would this have looked for a whole factory or an assembly line?

2.4 Requirements for solid models

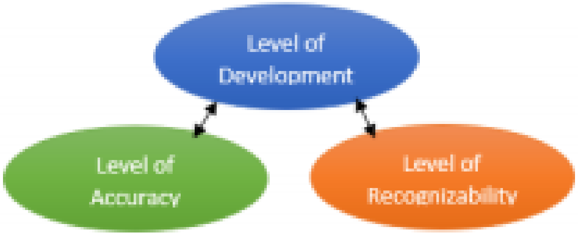


Figure 14 - Framework to define solid model [13]

The authors [13] start out by saying it is necessary to have a Virtual Factory layout to work in parallel with production. This means work "offline" and to not disturb the production. To create virtual models, 3D scanning is a viable technology which can scan the production floor and eventually create a virtual model [13]. The authors discuss how detailed these 3D models have to be for its purpose. In figure 14 the developed framework is presented, showing three major levels which correspond to each other. It is important when discussing a virtual model for how complex and accurate that given object or model should be. Therefore, these three levels are developed to ensure that the correct amount of effort and detail is given for the virtual model.

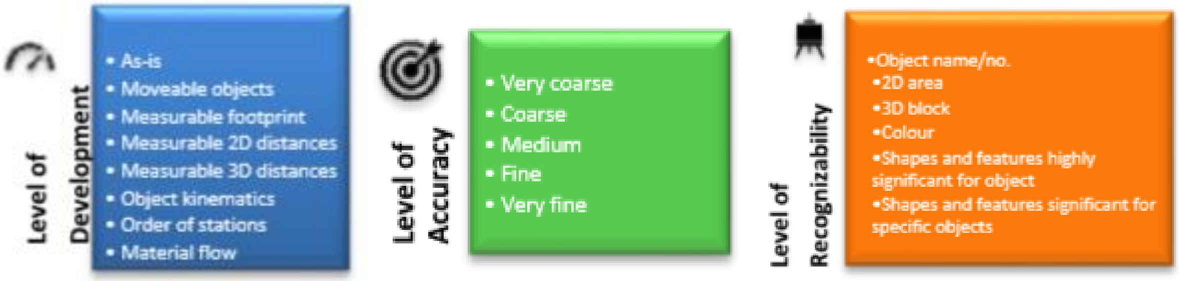


Figure 15 - The three main levels for the framework [13]

The first box is called *Level of Development,* LoD for short. This branch covers the actual function of the component or object that is scanned [13]. It is important to know if this component should, let's say have kinematics to be able to move, or perhaps it should only contain a 2D footprint. Depending on what the actual use for that given component should also come how much should it be developed in the virtual world.

The second box is called *Level of Accuracy*, LoA for short. This branch covers how refined the given component should be [13]. LoD gives the shape and function while LoA makes sure how accurate these models should be. Having a 2D footprint, the accuracy should be at a medium range, but with a robot the accuracy level should be at a very fine range. Depending on what the component should be able to do, and what it should express, it should be reflected by the accuracy level.

The last box is called *Level of Recognizability*, LoR for short. This branch covers how detailed the component should be modeled [13]. What information should it give the viewer? Shapes and features recognized anywhere, or perhaps naming or distinct colors and so on. This branch is mostly for either engineers or higher-ups to understand what is actually on screen. While LoD will give a correct sized block moving, and LoA a very fine tolerance, LoR gives it a shape.

In total, these branches can function as a guiding framework to developed detailed enough models for its purpose. Wasting more time than necessary is not beneficial for any parties.

3 Methodology

3.1 Assumptions and Limitations

For this thesis, a given scan was provided by a contact in Optimar for their testing cell. This scan has been scanned by people at Optimar, meaning a non-professional scan. It is clear that the given scan is on the low end of quality and can be assumed that scans performed by professionals would lead to a better quality in the scan. Unfortunately, due to no contact available in this thesis, this scan is the only industrial scan the author got his hands on. It would be desirable to have scans from a fish processing facility to grasp the scope of this thesis, but a larger room with fish processing equipment should suffice. It is assumed that the workflow and procedures developed in this thesis would not have changed regardless of the size of the scan, due to the similarity to a fish processing facility.

It would be desirable to have more than one industrial scan to look at different aspects of parts of a fish processing facility. In such a facility, each room has distinct equipment in the fish slaughtering procedure, which has complex geometry. With more scans available it could be set up with a quantitative comparison between different parts of the facility. It could highlight what is the most time-consuming equipment or part within these factories and perhaps sheds some light on the level of efforts needed to create robot programmable surfaces.

The testing has been performed on a common software platform delivered by Dassault Systèmes 3D Experience. Comparing other software solutions to this platform would be desirable to see how time efficient or time-consuming some options are. These forms of testing would create a brilliant quantitative and qualitative result showing positive and negative sides to different software packages. What this would require is for the author to perhaps learn 2 new software platforms and generate the same workflow as presented in this thesis. By looking at the time perspective of about 5 months of work, creating workflows for multiple software is not realistically achievable. That is why based on time limits one software has been selected due to it being sufficient for solving the stated research questions.

3.2 Research method

3.2.1 Action based method

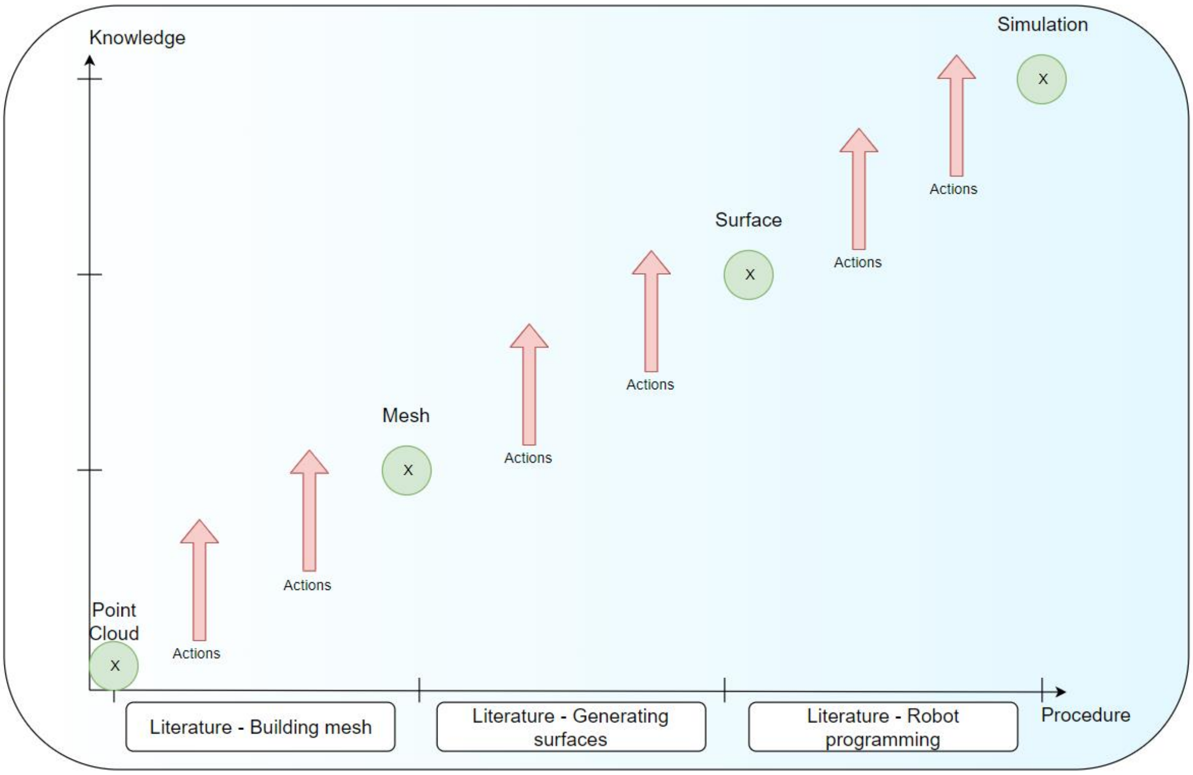


Figure 16 - Action based method for test cell

The main idea behind the Action-based research method is to build or test theories to solve a problem. Action research is three different types of approaches that can be used. The first is *technical* which aims to improve effectiveness, the second is *practical* to improve or transform knowledge and the last *emancipatory* which acts as process moderator for organizational systems [19]. By using the *Technical* one combining already established state-of-the-art procedures and research in similar fields, an action map can be drawn. Figure 16 above shows how this can be mapped. The baseline of knowledge is the literature found for each individual state (mesh, surface, simulation) and by doing certain actions on the system, knowledge for that specific procedure can be established.

In an AR method, a five-step phase cycle is generally used to solve a problem. The first phase is to identify the research problems which normally leads to some assumptions [19]. In this case, the starting point, midpoints and end points are known, but not how to get there. In figure 16 above, the green circles mark what is currently known and what is desired in these states, but it is unclear how to reach that specific point from the previous one. The problem is how do we find the connections between one circle to the other and in which way.

Phase 2 is known as *Action Planning* and evaluates the problems stated in phase 1 [19]. From what we currently knew, procedures and actions going from one state to another is unknown and is what that needs to be researched. By looking at the research already established in chapter 2 and combining the known states of our procedure a plan can be developed to achieve the desired results.

Phase 3 is *Action taking*, and here the actions established can be performed [19]. This means by combining the state-of-the-art with this procedure in mind a common procedure can be established. Phase 4 is *Evaluating*, which more or less evaluates the process performed on the system to see what is achieved and what can be improved [19]. The last phase is known as *Specifying learning*, which expresses the sole reason behind the process and finding of the results. It is obviously important to learn while doing a project to properly understand what is really going on [19].

3.2.2 Case study method

The case study method is a method generally used when certain events happen on a real-life system, which is then created into a case. Normally these cases can be divided into explanatory case studies or descriptive case studies [20]. To figure out which one of these is appropriate for your given case, three conditions can be set. The first condition is what type of research question is posed, the second; the extent of control an investigator has over actual behavioral events and the third; the degree of focus on contemporary as opposed to historical events [20].

Starting out with the research questions proposed, different types of questions such as who, what, where, how and why are stated. For this specific case for conducting a case study on a test cell at Optimar, research questions are proposed as "how", which lead to an explanatory approach. As stated by R. K. Yin an explanatory approach will most likely lead to the use of case studies [20]. Looking at what option the author has over the behavioral events, the end results are defined based on the real-life system. There is however a possibility to change the outcome slightly with some manipulation within the steps needed to achieve robot programmable surfaces. For last the degree on focus for contemporary is highly used due to it looking at the current state and not for the historical events.

"It should also be noted that case study is not a qualitative method of research. It is an empirical method and could be limited to entirely quantitative or qualitative data, depending upon the case being studied" [21]

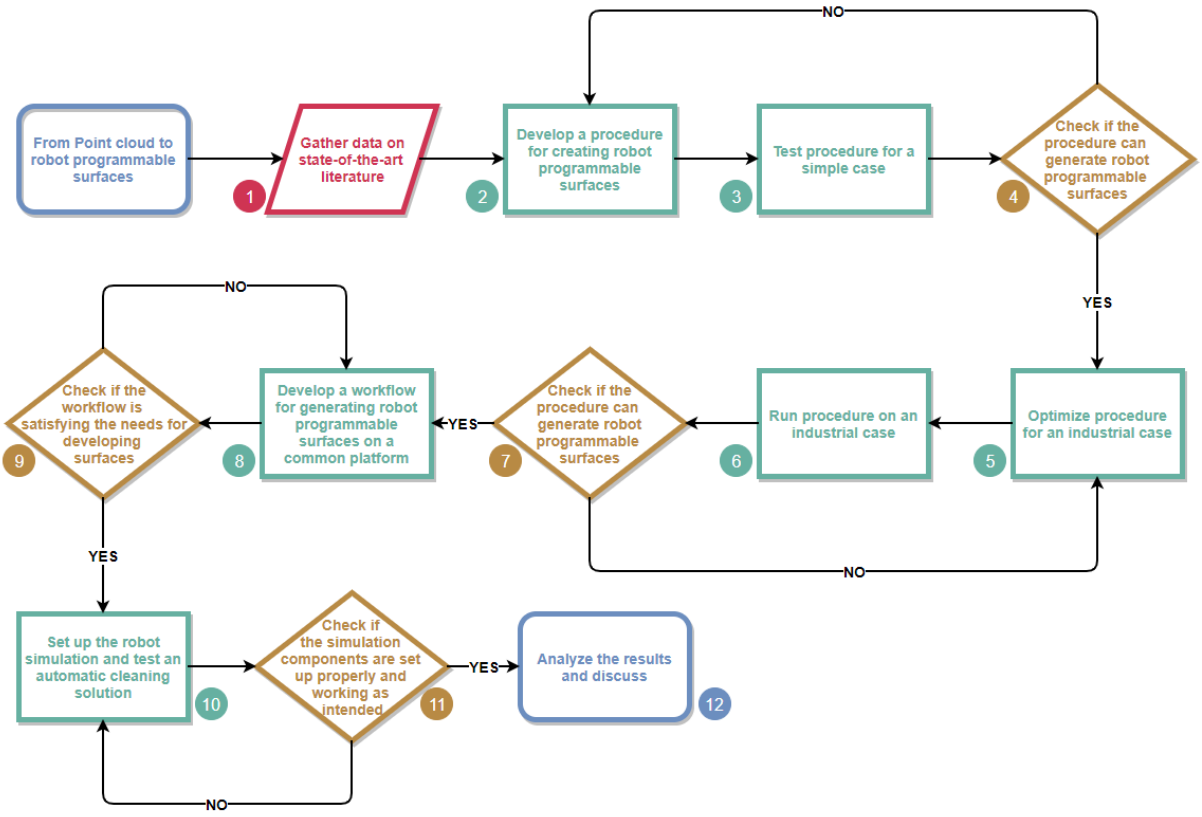
As stated by Teegavarapu S. Summers J. and Mocko G. it depends on what case is currently being investigated and for this thesis, qualitative fit more. Using the case study method certain steps has to be executed to perform a proper case study. In [21] on page 4, a step by step method is proposed for how to use the case study method. The first step is to identify/define the problem, which in this case has already been established from early on in chapter 1.2. Further, a theory is built up and in this case that would be the procedure that will be tested on different cases. When the problem and theory can be applied, the application can start where the outcome is documented. In their proposed step by step, a conclusion to each case and comparison is adviced, but not deemed necessary in this thesis. They propose a more extended study with different cases, but here used to that extent to have one case to check and validate the current conducted theory before using it on the main case.

3.2.3 Discrete event simulation method

An extensive description of how a discrete event simulation is both set-ups and used can be found in chapter 2.2.1. For this specific case, discrete event simulation or DES was to perform a cleaning operation in the test cell provided by Optimar. By setting the correct values and correct input/output for that specific case a result showing how the given simulation can generate programs for automatic cleaning solutions. The set-up can be found further down in the thesis at chapter 5.3.1

In some cases, a simulation will put out quantitative data but that does not mean it only gives out quantitative results. For this specific case with the test cell at Optimar with a common platform for generating robot programmable surfaces, a mix of both data is what occurs. With variables and heavy influenced by theoretical approaches, some results are quantitative and can be gathered from the DES [15]. In this simulation case, more than one result occurs but is chosen like this to get appropriate and desired result, meaning a closer relation to qualitative. DES method is a numerical approach which in this case gives a set of programs for the robot to clean a fish processing equipment.

3.3 Flow Chart



From Point cloud to robot programmable surfaces
1
Gather data on state-of-the-art literature
2
Develop a procedure for creating robot programmable surfaces
3
Test procedure for a simple case
4
Check if the procedure can generate robot programmable surfaces
NO
YES
5
Optimize procedure for an industrial case
6
Run procedure on an industrial case
7
Check if the procedure can generate robot programmable surfaces
NO
YES
8
Develop a workflow for generating robot programmable surfaces on a common platform
NO
9
Check if the workflow is satisfying the needs for developing surfaces
YES
10
Set up the robot simulation and test an automatic cleaning solution
11
Check if the simulation components are set up properly and working as intended
NO
YES
12
Analyze the results and discuss

Figure 17 - Flow Chart

Step 1: Gather data on state-of-the-art literature

The first step in the overall workflow of this thesis will be to gather state-of-the-art papers and documents on literature. These literature needs to cover 3D scanning, procedures for surface creation from spatial data and robot simulation. 3D scanning is vital to know how the spatial data are captured and in what ways it can be captured. By knowing the limitations and techniques it is possible to build up the knowledge to understand why a method of capturing is chosen. There are perhaps as well procedures already established for generating surfaces or solids from spatial data, but in more ways than the thought of. The procedures can be a foundation for understanding what the procedure must do, and from there build upon the knowledge found to develop an own procedure on a common platform. Setting up a simulation correctly is also vital for the purpose of this thesis. The robot programmable surfaces generated will be tested in a simulation environment and therefore knowledge and state-of-the-art simulation methods are important. Set-up and what a simulation brings for a result is key to this thesis.

Step 2: Develop a procedure

Once the author has a general understanding of how to both understand and use the state-of-the-art literature, a procedure for developing robot programmable surfaces can be produced. Based on literature and findings, procedures already established will be understood and developed further to fit the needs of this thesis. It will be based on a common platform with a new view on surface creation with the use of robot simulation and programming. Most of the found procedures are either used for visual representation or 3D printing and not robot programming, which is the main new research in this thesis.

Step 3: Test procedure for a simple case

With a procedure is developed a simple case must be tested. For a simple case, the top of a band sawing machine has been selected due to the low complexity of both the surfaces and the geometry. The procedure will be tested step by step to verify if it is possible to achieve robot programmable surfaces. It is important to understand that this procedure is used on a smaller case to verify but will later be put to the test on an industrial case.

Step 4: Check the developed procedure

When working with the procedure it is necessary to check if the developed procedure is satisfying the needs for robot programmable surfaces. Did the procedure generate the expected results or were changes within the procedure needed? Sometimes it can happen that the procedure is changed midway through because it was clear that the steps developed did not satisfy.

Step 5: Optimize procedure for industrial case

After the procedure is verified and usable it can continue further for the real case. But before it can be tested it has to be modified for an industrial case. For the test case, a low complex component was selected with relative few surfaces to be able to check and verify the procedure. Now that it is satisfying expected results it can be used further, but now with much higher complexity. The complexity will stretch the procedure to see if it can be used on an industrial level and not on a small test subject. Therefore a step before testing has been added to check for problems with the procedure due to complexity and size.

Step 6: Run procedure for industrial case

Now that the procedure is adjusted to be more optimized for more industrial use, the procedure can now be run on the test cell. The test cell spatial data acquired from Optimar can be put to the test with the developed procedure to see how the procedure handles a more complex geometry to create robot programmable surfaces. The amount of data compared to the simple case is about 100 times larger, meaning both the computational load and effort level is skyrocketing.

Step 7: Check optimized procedure

When the optimized procedure has been run through the industrial case, a test cell with robot programmable surfaces should be the outcome. As with the other check for the simplified test case, the procedure needs a check if it fully could give robot programmable surfaces without any adjustment. If there is any adjustment to the procedure the developed procedure needs to be evaluated before proceeding further.

Step 8: Develop a workflow

With a working procedure able to generate robot programmable surface, workflow on the common software platform must be developed. In this case, the author has selected Dassault Systèmes platform 3D Experience as the common platform. It is a necessity to be able to use this procedure on a common platform, that workflow will be developed. The workflow will present views and ways of achieving the end results desired. Going all the way from spatial data acquired either by, someone else or scans performed by yourself, to robot programmable surfaces. The workflow will also contain the steps required to be able to put a scenario of simulation into the 3D scanned model.

Step 9: Check the workflow

The workflow created and developed is in the views and interpretation of the author. To check if the workflow is actually understandable by other users than the author himself, the supervisors will be set up to see if the developed workflow is acceptable and understandable. It is necessary to have personnel with less experience to understand the workflow for what it presents, and that the description and views are clear. If the stated views and flow is not received as intended, the workflow has to be modified. Having a workflow not understood by its reader is not desirable at all.

Step 10: Set up simulation

Based on the theory found for how the different components behave in a simulation, a simulation set up can be built. The 3D scanned test cell which now contains robot programmable surfaces can be put through a simulation to get the desired result behind this whole thesis, a robot program that can automatically clean fish processing equipment. Therefore, a setup is built around the test cell with a general solution for cleaning, a robot moving on a linear guided railing system. With the correct setup, the output from this simulation should contain a program for cleaning procedure for that given cleaning solution.

Step 11: Check simulation

With different inputs and information within the simulation, it is important to check if the results are as they should be, or desirable at all. The simulation set up needs to be complete and the components building up the simulation needs to be set up correctly. Without the information and components going as they should, an incomplete result may arrive. Therefore, a check with different inputs and outcomes can be tested to verify if the simulation is up to standards and generating results accordingly to specification.

Step 12: Analyze and discuss

Now with a complete procedure, workflow and simulation to generate robot programmable surfaces it can finally be addressed, analyzed and discussed. A combination of theory, methods, and results will be the prime discussion point on how these results were achieved and what they bring for the newly developed research. All the stated research questions will also be answered accordingly to the findings from the results.

4 Case study

4.1 Case introduction and background

For this thesis, a set case has been selected to develop a procedure to go from raw spatial data to robot programmable surfaces. A visit to Marine Harvest last year gave some insight into how they operate for both production and cleaning, which is the baseline for this thesis. As mentioned earlier there are quite a few different ways of going about production and cleaning, so in this case, they are based fully on what insight and information were gathered from Marine Harvest.

This thesis is a continuation of a previous master thesis on NTNU in Aalesund delivered by Tommy Benjaminsen named "*3D-scanning in Aquaculture Engineering: Improving Spatial Data Capture Methods Using Terrestrial Laser Scanning*". In his last chapter for future research opportunities, he mentioned that it could potentially be possible to use the spatial data and the surfaces to offline program robots. Based on his finding this thesis will continue down that line looking at how 3D spatial data acquisition and robot offline programming can be put together [22].

His effort focused mainly on how to acquire spatial data and how important it could be for Optimar in its development of their equipment. Not only discussing how it could improve the engineering of certain equipment, but also installation. This thesis will mainly focus on the aftermath of acquiring raw spatial data and how it can be used properly to potentially be used for offline robot programming.

4.1.1 Simple case

To start off developing a procedure to generate solids from spatial data, a simpler case was selected to lower the computational load to test out different scenarios. The selected case was the top of a band sawing machine acquired from the co-supervisor. This set of point cloud data was acquired with the use of photogrammetry which leads to a rather large set of points. The main idea behind a smaller component is to develop a proper procedure with a less complex structure. Starting out with something complex could distort the procedure and even take a lot of time into efforts not needed. Hence a small component to develop a procedure has been selected and test. The procedure developed from this testing will be stated further down in the result chapter.

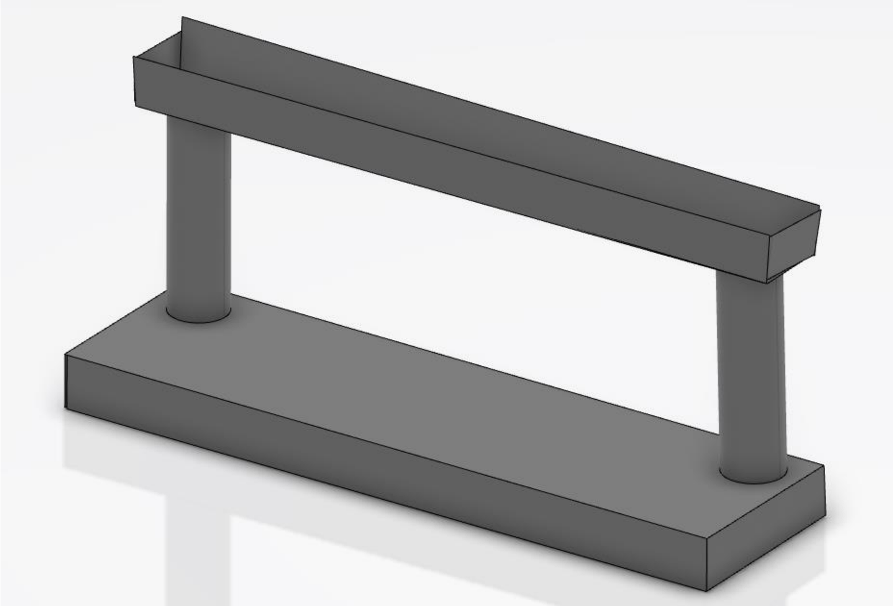


Figure 18 - Top of a band saw

4.1.2 Main case

The main case for this thesis is a test cell built at Optimar which was given permission to get access to raw point cloud data from that specific test cell. As mentioned in chapter 3.1 for limitation, this was the largest scan available at this given time. There was no equipment available, so the given scan is in an "as-is" state. FAR0 X130 HD was the equipment used by Optimar for capturing the spatial data, and by the information given this scan was taken by non-professional.

To realize the full potential of this technique of 3D scanning an environment to build it up virtually to program robots, a more complex factory should have been acquired, but that is unfortunately not the case. The same method of the workflow will be usable for this test cell as with any factory. The test cell is a 7x5x3-meter room with beams and pillars with one fish processing equipment, the stunner.

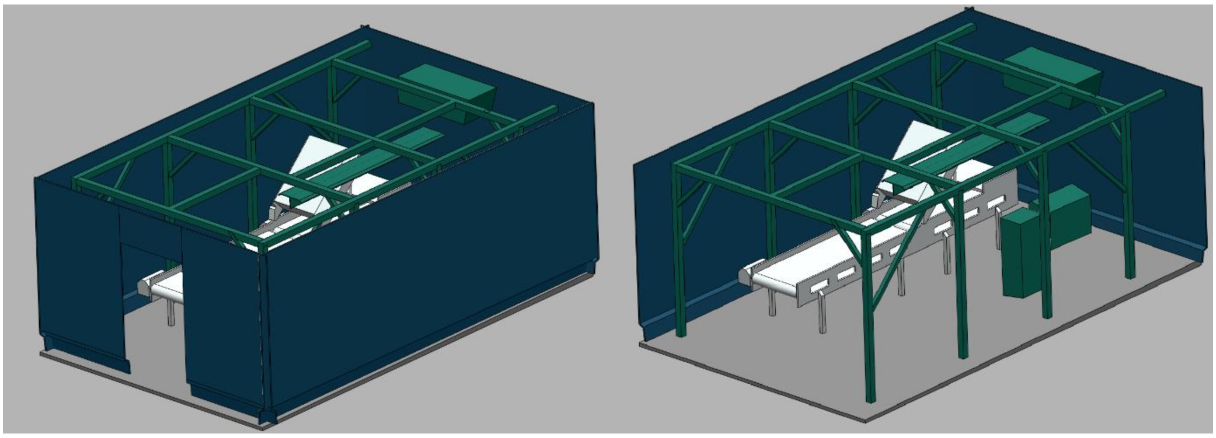


Figure 19 – 3D model of the testing cell at Optimar

As can be seen from figure 19 above, there are eight pillars, four overhead beams, two equipment control cabinets, one ventilation unit and one stunner. All in all, this is a close resemblance to a real factory setting, but on a smaller scale.

4.2 The current situation in fish processing production

4.2.1 Production and operating hours at Marine Harvest

Currently, at Marine Harvest stationed at Fosnavågen in Møre og Romsdal, they have operating hours for at least 5 days a week 24 hours every day. Depending on the amount of delivery, this might be shifted towards 6 or 7 days a week 24 operating hours. Production normally starts out at 8 AM and last until 4 PM, unless there are double shifts which start at 4 PM lasting until midnight. At best case, the cleaning group coming at night have from 4 PM to 8 AM the next day to clean the whole facility and at worst only eight hours. Within this time limit, they have to clean the whole factory from floor to ceiling with both soap and chemicals.

4.2.2 Current cleaning solution

There is about 20-30 personnel coming in every night to clean the whole facility with a given procedure which is using a lot of chemicals and manual labored work. Every worker has to be equipped with protectives gloves, suits, helmets, goggles, and masks before entering the facility. This is the safety regulation at Marine Harvest.

The general equipment used by the workers is low-pressure stations, high-pressure stations, chemical stations, and brushes. These are all used accordingly to a developed procedure shown in figure 20 below.

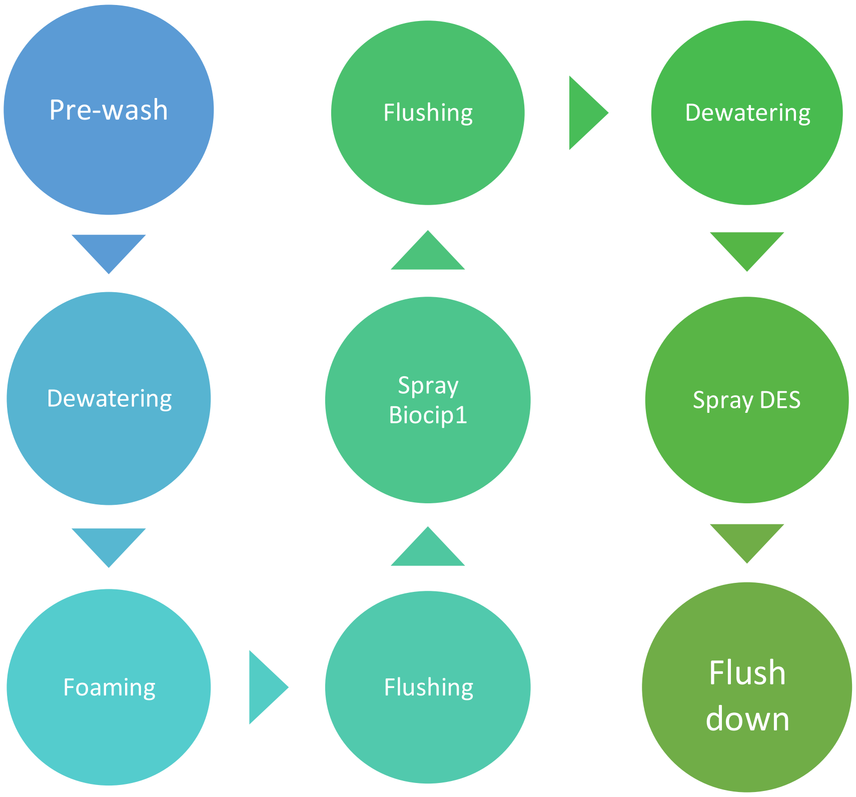


Figure 20 - Cleaning procedure at Marine Harvest

The first major step in the cleaning procedure is Pre-wash. This phase is meant to remove visible organic materials with a water temperature not exciding 35 degrees Celsius. Exceeding the temperature will eventually lead to “burning” of proteins and calcium salt to surfaces, making it even harder to remove. All arm movements are to go from top to bottom to drain off any material lying around. After the pre-wash is finished a visual check is initiated to ensure that all organs and leftovers are removed.

When the facility is clean for any leftovers, a dewatering phase begins. Due to the effect foam has, all surfaces must be somewhat clear for excessive water. Even a little break is recommended to let the facility dry a little. After dewatering the facility and perhaps a little break the foaming of the equipment starts. In this phase, it is very important that every surface and equipment is covered in a smooth and thin layer of foam for the maximum effect. Every nook and corner have to be foamed and within the next 10 to 15 minutes the foam will penetrate the fat and membrane layers that are left on any surface. Conveyor transports are also loosened to make foam stick within the transmission and underneath every rotating part.

Once every surface is foamed the first flushing phase is initiated, which lasts for about 2 hours. The flushing phase contains high-pressure water with a temperature of 55-65 degrees Celsius with the extra use of brush, cloth, and sponge. With the given foaming, the layers of fat and proteins should come off, and it is important to work systematically. When all the surfaces are cleaned from foam a disinfection chemical known as BIOCIP 1 is used. A thin layer will be showered onto every surface which is again left for a little while to take effect, then flushed off with the same procedure as for flushing foam.

In between the last step and the previous flushing, a check and dewatering start. This is the last check to see if any leftover chemicals are present. Any water left should also be removed before the disinfection can begin. The disinfection phase is a rather important process which requires good timing and efficient cleaning. The chemical is showered onto every surface and equipment and should only be left there for about 10-15 minutes. Leaving it on for too long will create discolored effects on certain materials. For this phase, the water temperature that shall rinse of the chemical should be at least 80 degrees Celsius.

The last phase in the procedure is to flush off any chemical left and is important to get everything off, due to it being in contact with food. Flushing should be done with low pressure with no more than 35 degrees Celsius temperature. When the chemicals are cleaned off a drying phase is initiated to prevent bacteria growth in moist areas. Either sterile swabs or air pressure is used to remove water and letting the facility dry up before the start of the next production shift.

One of the major issues with this procedure is to know which surface has been cleaned and washed properly. Humans will always be humans which makes mistakes. This procedure is a long and tiresome process which takes about 7-8 hours with a lot of arm movements and a steady hand. The work conditions with chemicals and work positions are not healthy which in long terms will eventually lead to sick leave. This was also confirmed at Marine Harvest that some personnel had back and arm problems due to the heavy-duty working condition.

5 Result and analysis

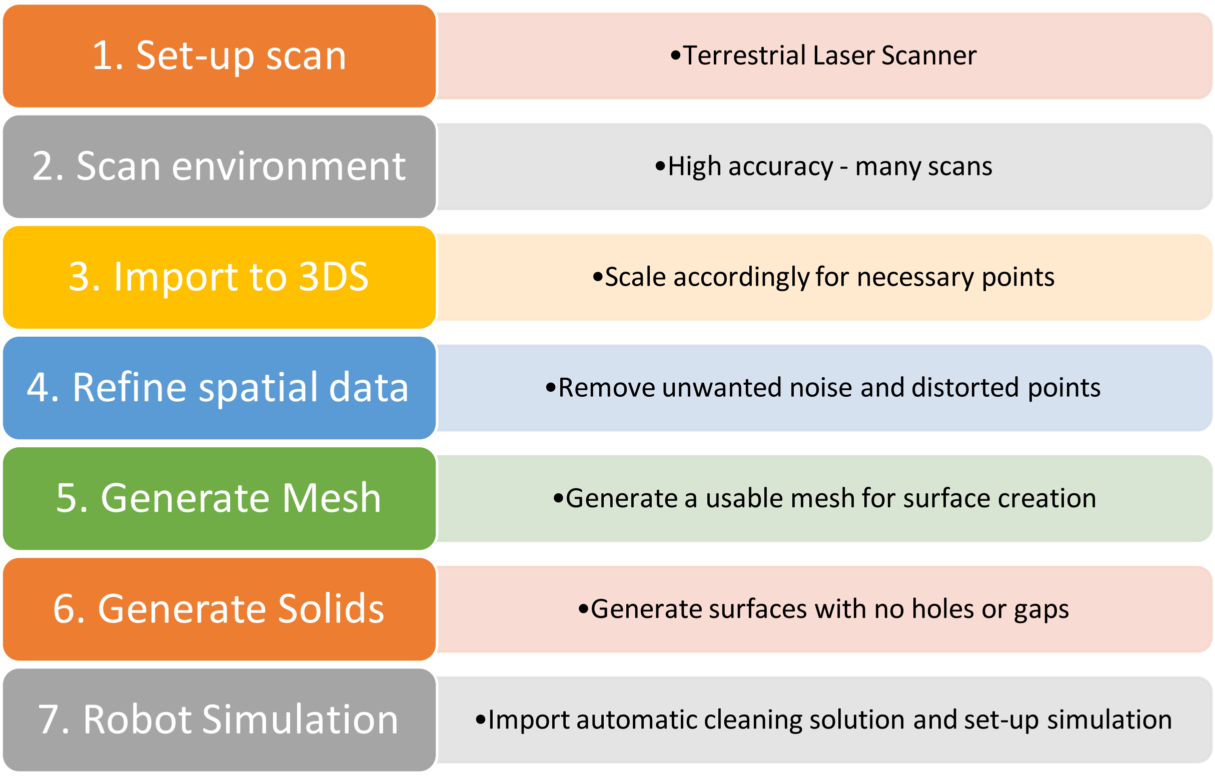
5.1 Procedure to generate 3D solids and programmable surfaces

5.1.1 Previously developed procedures

As previously mentioned in chapter 2.3 there are quite a few different procedures already developed to generate surfaces from 3D spatial data. This thesis will base the procedure on already existing knowledge and develop a procedure on a common software platform, which will be covered in the next chapter. All of the procedures have a common goal in mind, which is going from 3D spatial data to any solid model. This has been solved in a lot of different software and in different fields. Most of the procedures have a common flow from where to go and how to end up with the desired result.

There are a few uses of the 3D solid models established from 3D spatial data such as 3D printing, 3D layout visualization and BIM in civil engineering. None of the found research has used robot programming from 3D spatial data yet.

5.1.2 Developed procedure on a common platform



1. Set-up scan
- Terrestrial Laser Scanner

2. Scan environment
- High accuracy - many scans

3. Import to 3DS
- Scale accordingly for necessary points

4. Refine spatial data
- Remove unwanted noise and distorted points

5. Generate Mesh
- Generate a usable mesh for surface creation

6. Generate Solids
- Generate surfaces with no holes or gaps

7. Robot Simulation
- Import automatic cleaning solution and set-up simulation

Figure 21 - Procedure to develop robot programmable surface

Figure 21 above shows the overall full procedure needed to be able to capture any spatial data and use it to create and test automatic solutions for cleaning procedures. This developed procedure can technically be used in any software capable of handling offline programming of robots but is based on the use of Dassault Systèmes 3D Experience.

To start out any project with spatial data a 3D scanning tool is needed and based on knowledge found from the literature review, a Terrestrial Laser Scanner seem to fit the best purpose of scanning indoor facilities. A combination of photogrammetry and TLS could potentially be used to get colors involved to create a more realistic scanned area, but not necessary.

When scanning the given area, reference points have to be put out to be able to merge corresponding scans together. It is advised to have more angles and camera positions to get every edge and corner on most components. Depending on what type of equipment, piping, and machinery is inside, the pattern of scanning can change. Any square box machine is rather simple to scan, but any distinct machinery with a lot of bends and complex angles needs a more intense scan. The more camera angles and surfaces scanned, the better which makes surface reconstruction later smoother.

Ones the facility has been scanned, all of the spatial data can be imported into 3D Experience for further processing. The first option when importing spatial data into 3DS is how dense the point cloud should be generated. Depending on the resolution from the scanner, any point cloud containing more than 10 million points are too large. The computational power needed to run 100 of millions of points is extreme and can eventually lead to failure in processes further in. Depending on an area as well, 10 million could be very dense or perhaps a little too dense. It is recommended to scale it down to about 1 million points, to be able to handle the point clouds rather efficiently, otherwise, a lot of time is spent waiting for the software to calculate million on millions of points.

Step 4 is to refine the given spatial data. First of triangulation and merging of point cloud has to be in effect, to get a complete picture of the scanned facility. By using the reference points all camera angles and scanned area can be merged successfully as one large point cloud. It is now important to get rid of unwanted points which are either distorted or reflections of stainless material. The time spent on cleaning the point cloud is time well spent. Going about this step too quick will definitely lead to problems later on, so it is advised to the fullest to spend a lot of time on this procedure, it will most likely save you time, in the end, either way. To quickly get rid of unwanted points, sectioning of the scanned area should be considered. How one would like to section is rather hard to say, it will depend on how equipment is placed, how large areas are, etc. One way of sectioning could be to have a ceiling, walls, and floor come off, and then each and every equipment can be sectioned by itself. This could potentially be very time consuming, so depending on the quality one wish for it is advised to figure out this beforehand.

When the given point cloud is sectioned as intended and free from unwanted points, meshing can proceed. Meshing will leave small triangles connected from points in the point cloud. At first glans, this will look like a surface but is based on mesh connecting the dots. The mesh can either be computed from the complete point cloud or from smaller sections. It is advised to mesh every section individually, which will prevent certain types of error in the mathematics behind it. Not only will the mesh contain a lot of errors, but the generation of mesh will perhaps take days and the potential of failing is high. Meshing the whole facility at once will lead to tens of thousands of errors such as gaps and holes, non-manifold edges, non-connected zones, etc. Every mesh will contain errors unless the point cloud is 100% perfect, which is almost impossible to achieve, at least for facility sized point clouds. Creating the mesh from smaller sections will leave a smaller amount of errors for each individual mesh. Trying to fix tens of thousands of errors takes hours, if not days to fix depending on the number of points in the point cloud. Most meshes with that amount of errors are impossible to fix 100% as well. Therefore stick to smaller sections and the repair of mesh errors will not take days to fix, but rather a couple of hours.

After the mesh is free from most errors and looks usable it is time to start creating surfaces. Any mesh can be generated surface onto but is advised to start out with larger area surfaces, such as; ceiling, walls, and floor. When working with surface creation, be sure to hide the unnecessary segments of mesh not used at that point. Navigating the mesh and selecting the correct mesh will be a lot easier while the other not used parts are hidden. Also start out by hiding the floor, ceiling, and walls immediately after surface creation to have a broader view of the remaining mesh. Focus on the most recognizable surfaces first, to get larger and smoother surfaces out of the way faster. Starting out with smaller details, adding just a few square meters of the surface is disheartening for the overall process. Smaller details could take some trial and error to get the correct surface angle one wishes for, which is best left for the later stages of the process. This will eventually lead to a 3D solid model for the scanned facility with robot programmable surfaces.

The last step in the developed procedure is robot simulation. As mentioned in the introduction to this thesis, the main goal for this thesis is to find a usable procedure to generate offline programmable robot surfaces from spatial data. Now that the surfaces have been created it can be imported into Delmia, the simulation part of 3D Experience, and be set up for simulation. Any solution that might be considered with robots or robot guide linear rails can be imported into the 3D scanned model and put through a simulation. In here every offline programmed solution by robot movements can be saved and exported to a real robot, where the robot will run the offline programmed program. Instead of having a real robot in the facility taking up space and perhaps shutting down the production, everything can be programmed beforehand.

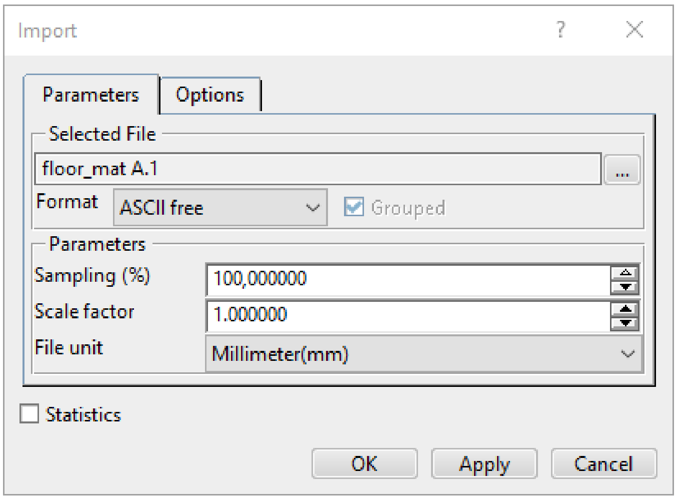
5.2 Workflow to create robot programmable surfaces on a common platform

In this chapter a detailed workflow for how someone can go from raw spatial data to a finished 3D model in a simulation setting with the use of Dassault Systèmes 3D Experience. Also, an important aspect of how to handle certain types of errors and how to set up the simulation environment properly.

5.2.1 Processing point cloud data

The first thing that has to be done is to open a brand-new 3D part file and saving it with a specific name or id to remember it later. This file will be the building block of the point cloud. Make sure the spatial data acquired is of a standard 3D Experience can handle, otherwise, there might be some complication. After the file is created, one must find the app named *Digitized shape preparation,* which is the app used to prepare point clouds for meshing and prepare mesh for surface creation. By dragging the desired spatial data into the open 3D part file, an import dial box appears shown in figure 23 below.

Figure 22 - Digitized Shape Preparation



Selected File: floor_mat A.1

Format: ASCII free ☑ Grouped

Parameters

Sampling (%): 100,000000

Scale factor: 1.000000

File unit: Millimeter(mm)

□ Statistics

Figure 23 - Import dial box

From the *Format* drop-down menu, a selection of different formats can be selected to fit the current format. The rest of the dial box is always set to 100% sampling, no scaling and in millimeters. If the user knows the point cloud is above 10 million points it is advised to sample it down to about 10%. This, however, can be left untouched and later adjusted or if the area of the point cloud is massive, 10 million points might not be that much after all. It depends on the size and area. Normally it is preferred to keep the point cloud underneath 10 million points.

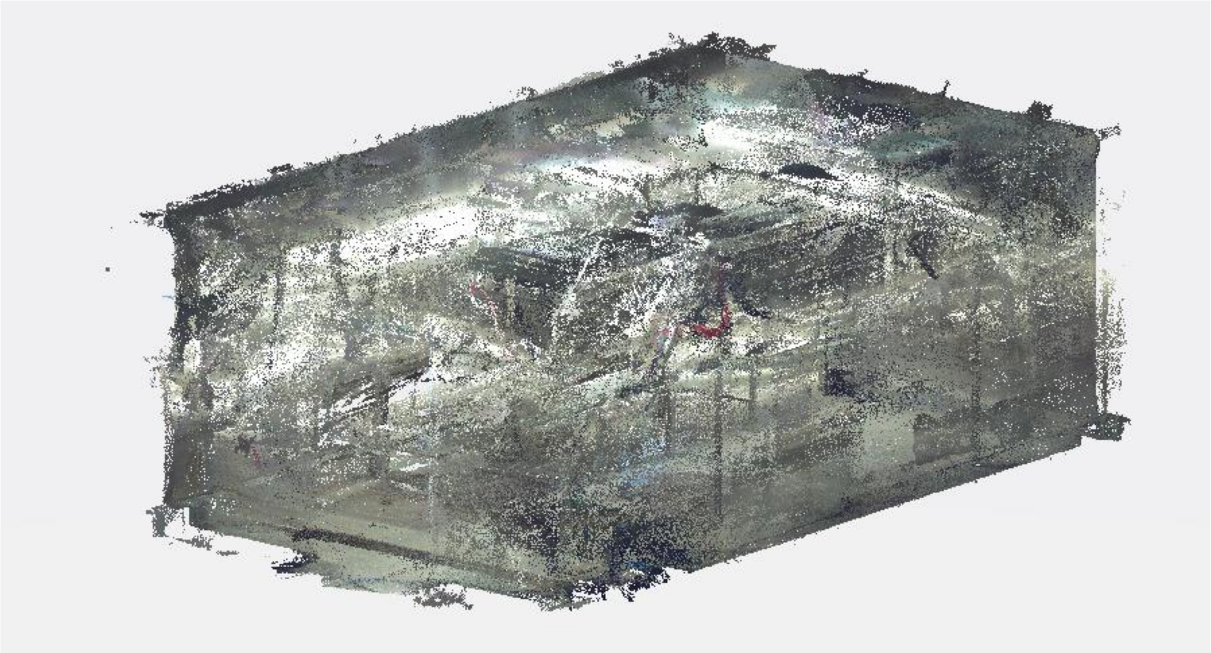
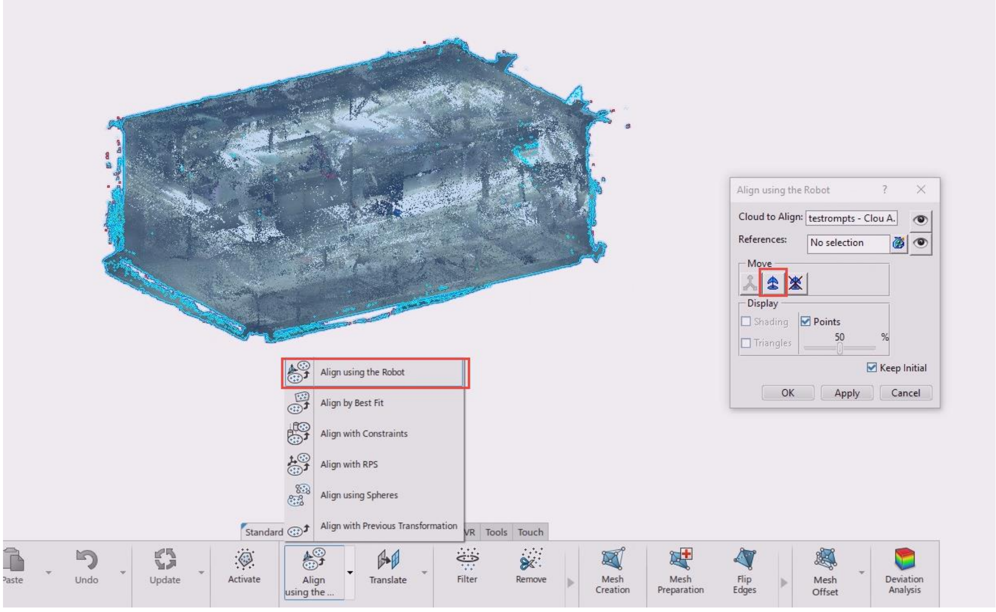


Figure 24 - Test cell as Spatial data

When the desired sampling and format has been changed and accepted, the point cloud is represented. In this case, this specific point cloud is no more than 2.8 million points sampled down from 28 million. Even with a computer running one of the most advanced 3D software graphics cards, 28 million is simply too much.

Now that the point cloud is imported and in correct sampling size, point cloud refining can begin. In some cases, point clouds imported are not aligned either horizontally or vertically correct, which means it has to be adjusted angle wise. In 3D Experience, there are quite a few ways of turning and aligning the point cloud to the global coordinate system within the software. In the action bar at the bottom of figure 25 shows the possibilities of using commands to align the point cloud. The most efficient is by the command named *Align using the robot*.



Align using the Robot

Cloud to Align: testrompts - Clou A.

References: No selection

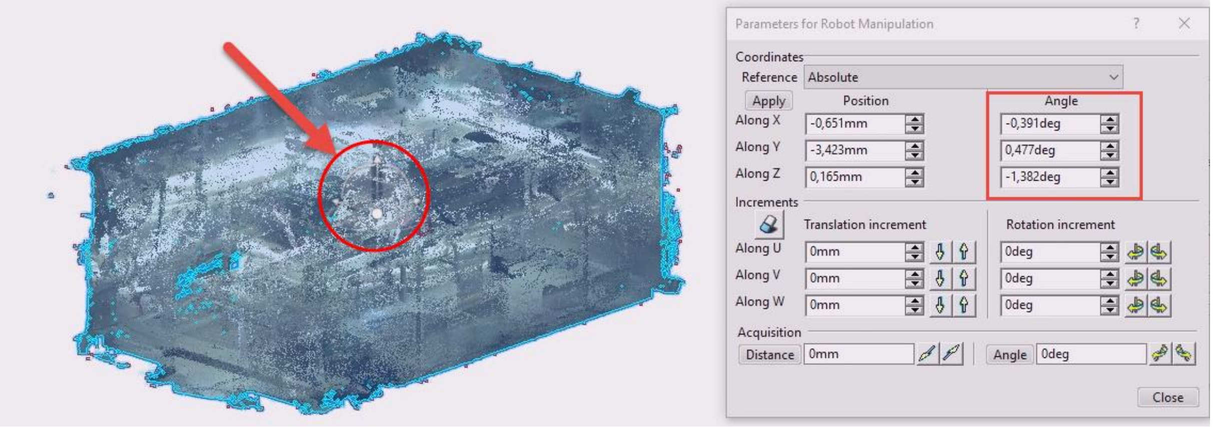
□ Shading ☑ Points

□ Triangles 50 %

☑ Keep Initial

Figure 25 - Align using the Robot

In the dial box appearing after selecting the desired point cloud and then command *Align using the Robot,* certain options are available. The first option to be selected is the one marked with a red square in the dial box, as shown in figure 25 above. This will move the robot (the name of the global coordinate system) onto the gravitational point of the cloud.



Parameters for Robot Manipulation

Reference: Absolute

	Position	Angle
Along X	-0,651mm	-0,391deg
Along Y	-3,423mm	0,477deg
Along Z	0,165mm	-1,382deg

	Translation increment	Rotation increment
Along U	0mm	0deg
Along V	0mm	0deg
Along W	0mm	0deg

Distance 0mm Angle 0deg

Figure 26 - Robot Manipulation

The robot is now visible (barely) on figure 26 above. By double-clicking the center point of the coordinate system, the white dot, another dial box appears. In this box, movement along all axis and angles can be made to correct the errors when importing. As seen in figure 26 the angles needed to balance the point cloud were rather odd, but by looking at the view from different angles it can be changed accordingly to the global reference point. It is, however, necessary to do this at the start rather than later. The reasoning behind it is that surfaces created from the mesh will not be rotated when the point cloud is rotated. Meaning surfaces will always be at an angle not straight.

Once the point cloud is firmly placed according to the global coordinate system the next step can proceed. As stated earlier in chapter 5.1.2 splitting up the point cloud in manageable sizes is advised. For this case, it was split in 4, close to equal sizes to manage the points easier. In figure 24 further up, the point cloud imported was rather messy and contains a lot of distorted and unnecessary points. The function used to filter out these distorted points was *Remove,* which highlights the given scan and then removing unwanted noise.

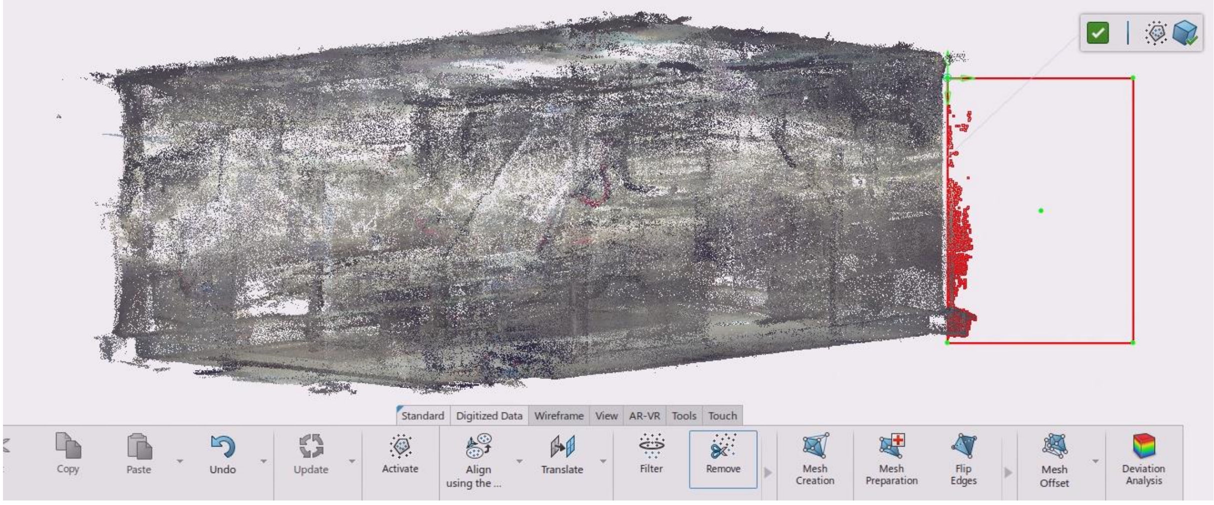
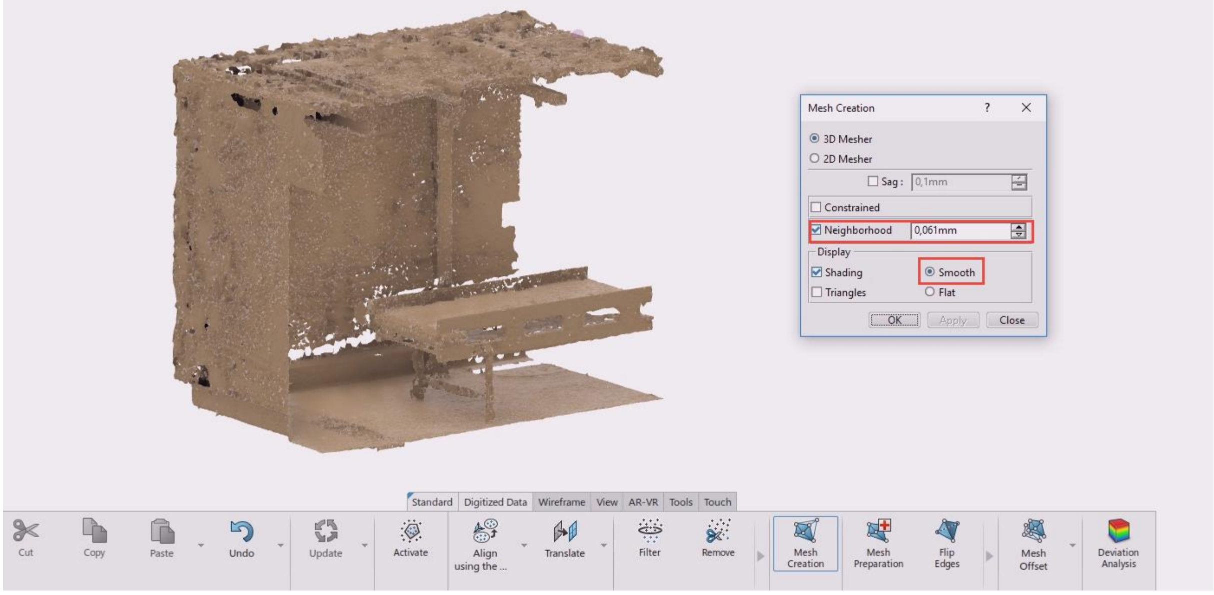


Figure 27 - Removing unwanted noise

Figure 27 above shows the use of *Remove*. When clicking the command *Remove* you are prompted to select which cloud to refine. Then after selecting the desired point cloud, move your cursor over an area you wish to refine and click. A red box, as shown to the right on figure 27, will appear which can be moved in either direction to remove the unwanted noise. Then simply click the green verification symbol and all the marked red points are removed. It is advised that the user spent a lot of time removing unwanted noise, which will leave processes later to be much smoother.

5.2.2 Generating mesh from point cloud data

When the desired state of the point cloud is acquired, the next step will be to start meshing. The mesh will try to create small triangles in between points to create a mesh over the whole point cloud. Since the point cloud now is divided into four chunks, each and every one of these will mesh separately. It has been mentioned previously in chapter 5.1.2, but the main reason behind separating the mesh is to have a manageable amount of mesh errors. Too many errors on one mesh make it almost impossible to solve and repair. It could potentially be split up even further for even more manageable error handling.



Mesh Creation

● 3D Mesher
○ 2D Mesher
□ Sag : 0,1mm
□ Constrained
☑ Neighborhood 0,061mm

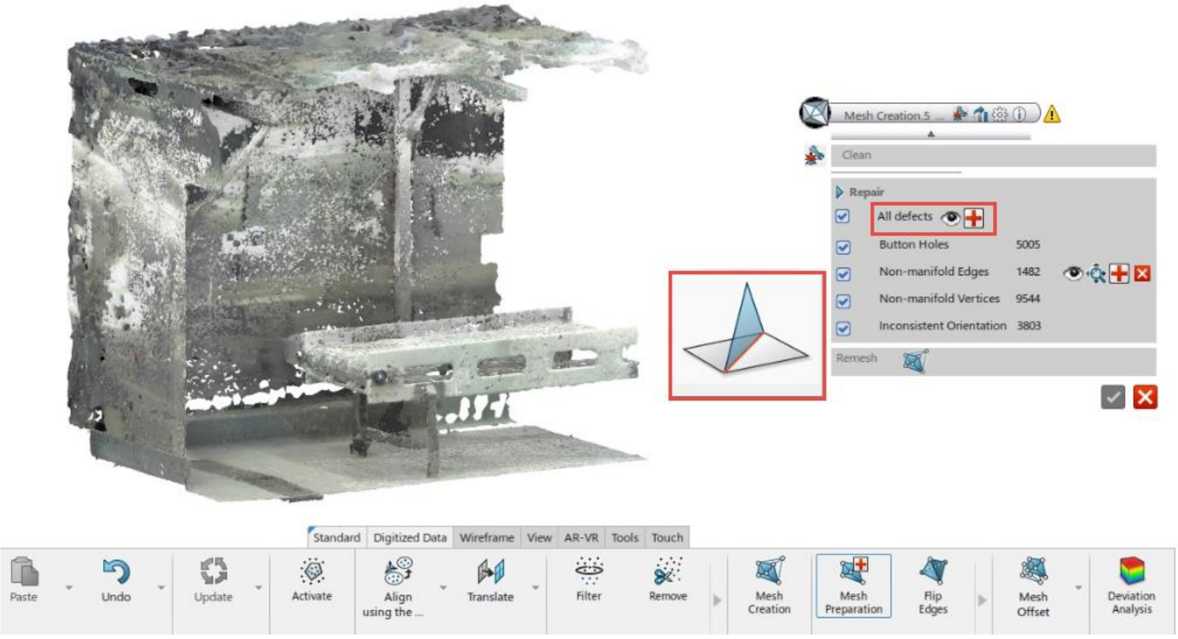
Display
☑ Shading ● Smooth
□ Triangles ○ Flat

Figure 28 - Mesh creation

By selecting the desired point cloud and clicking the command *Mesh Creation* a dial box appears. In this dial box, a certain option is important for the integrity of the mesh. As a standard; 3D Mesher, shading and neighborhood is selected. To create a better-looking mesh, *Smooth* is recommended to be checked off. For certain cases, Neighborhood can be unchecked, but what it does is to not rely on any distance to the next point. What this means is that the software will try to connect as many points as possible without restrictions. This can potentially be very useless if there is any point deviating from the geometry captured, such as distorted points. In most cases, Neighborhood should be active, but for the given parameter it can be changed. The parameter sets a maximum limit for how far a mesh can travel between points, meaning any points not closer than 0,061mm (in this case) will not mesh together. There is a way of testing this, by clicking "Apply" and see how the model looks. By setting a specific value, clicking Apply and then later change it, is possible. There is no way of telling what is good and what is bad, so have a visual check when tampering with the numbers before accepting. Repeat this procedure for all segments.

Now that the point cloud finally meshes, two options are available. Either you continue on with the mesh and go straight for surface generation, or you can refine the mesh even further making it easier to handle. It is advised to refine the mesh before continuing due to the time consumption and effort to create surfaces from an unrefined mesh. As seen in figure 28 above the mesh is not looking too great which means that it needs repair and healing.

In the action bar at the bottom of the software is a command known as *Mesh Preparation*. This command will let you heal and repair faults and errors in the selected mesh. By clicking the desired segment of the mesh and clicking the command will open up a dial box with some alternatives, which is shown in figure 29 below. The errors founds are stated in how many errors and what types of error occur in the given mesh. For this segment, it contains about 20 000+ errors in different aspects. By hovering over one of the error types, a display window marked in red shows up to let the user know what this type of error actually means. To start repairing the mesh you can click on the healing cross on the top menu marked in a red box in figure 29 below. The software will try to solve all the error stated below as much as possible. Since this case has a lot of them, it will only fix a portion of them. What the user can do, is keep running it over and over until it eventually goes to a very few numbers of errors.



Mesh Creation.5

Clean

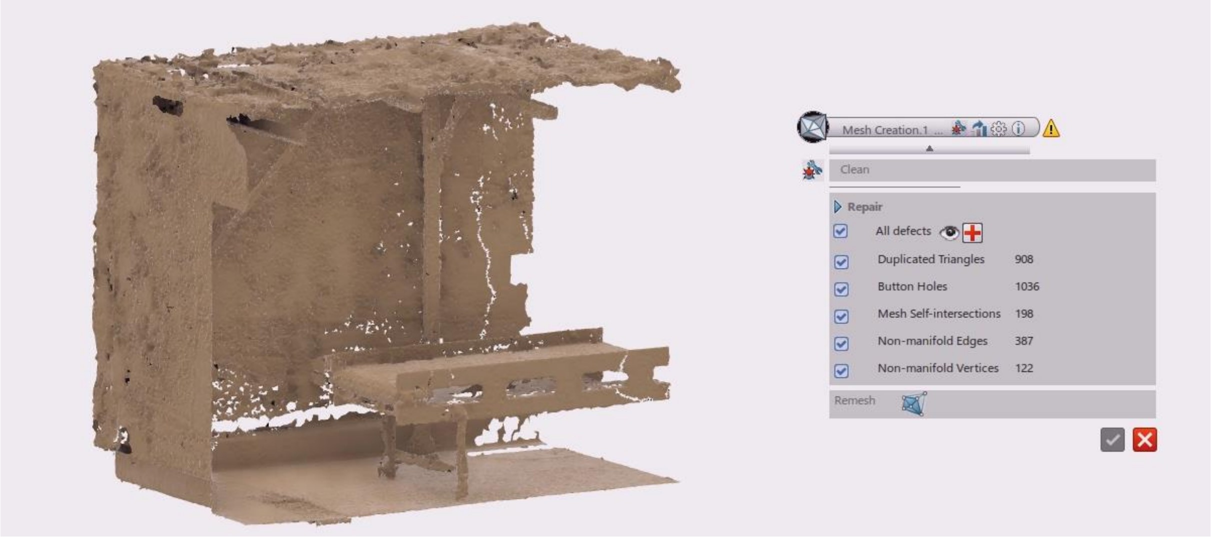
Repair

☑	All defects	
☑	Button Holes	5005
☑	Non-manifold Edges	1482
☑	Non-manifold Vertices	9544
☑	Inconsistent Orientation	3803

Remesh

Figure 29 - Mesh preparation

It would be desirable to remove all the errors, but some of them are almost impossible to fix. One case can be to remove a hole, but in the process, a non-manifold edge is created to fix the hole, or perhaps the other way around. This point cloud has a lot of faults from capturing the spatial data, making the process now to fix the errors troublesome.



Mesh Creation.1

Clean

Repair

☑	All defects	
☑	Duplicated Triangles	908
☑	Button Holes	1036
☑	Mesh Self-intersections	198
☑	Non-manifold Edges	387
☑	Non-manifold Vertices	122

Remesh

Figure 30 - Mesh preparation complete

As figure 30 above shows, a certain limit is reached. Every iteration of solving now will only result in less than 1% error fixing. At this point, there is no need to continue, even though the errors are among the thousands. This is common for incomplete spatial data or captures by non-professionals. At this point, you can either continue, but it will require more time creating surfaces, or perhaps get new complete spatial data. For this case, this was the appointed point cloud meaning there is no other choice than to continue forwards.

There is however a backlash for not having a 100% error free mesh. In 3D Experience, there is a command called *Mesh Smoothing* which flips and change the orientation of triangles in the mesh to give it a smoother surface.

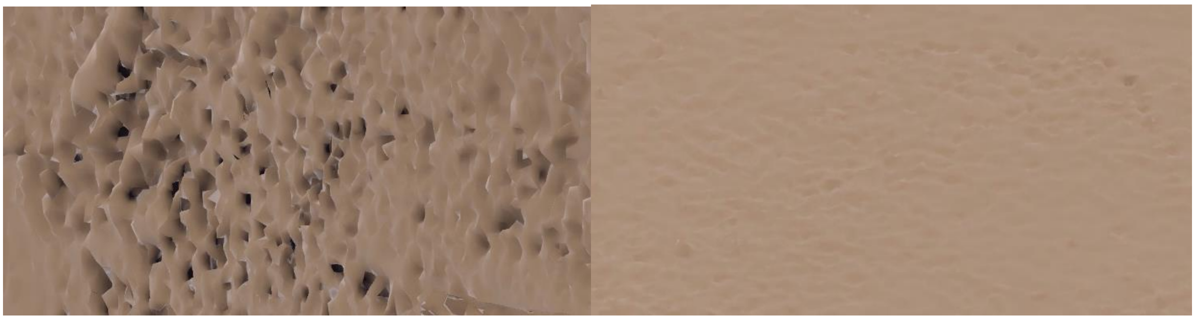


Figure 31 - None-smooth mesh and smooth mesh

The problem using this function is that the mesh needs to be 100% error free. Trying to use the command without fixing the errors will prompt in a failure of the process. Figure 31 above shows how drastically the changes can be when *Mesh Smoothing* has been used. This does not mean that the mesh has to look like the right-hand side, but it does help the procedure. For this case, there is no way of fixing all the errors meaning *Mesh Smoothing* was never used.

When the first segment has been run through the *Mesh Preparation* repair process a couple of times, one can continue with the remaining segments. There is however a more manual way of fixing larger errors such as holes which will be covered in the next chapter. Depending on the size of the hole, it can be recommended to try fixing some of them, but there are ways of working around holes.

5.2.3 Handling certain errors from mesh generation

One of the most common errors gotten from mesh creation is holes. To highlight how this can be solved, the simplified model was chosen to show a clear perspective of how to handle holes in a mesh. Holes make it way harder for the software to detect which mesh is connected to that specific surface. This means that a way of plugging holes is a necessary approach to get a fully fleshed out mesh which makes the surface detection much smoother. In some cases, the software will not even recognize the selected mesh as a surface due to it being filled with either too many holes or too large holes.

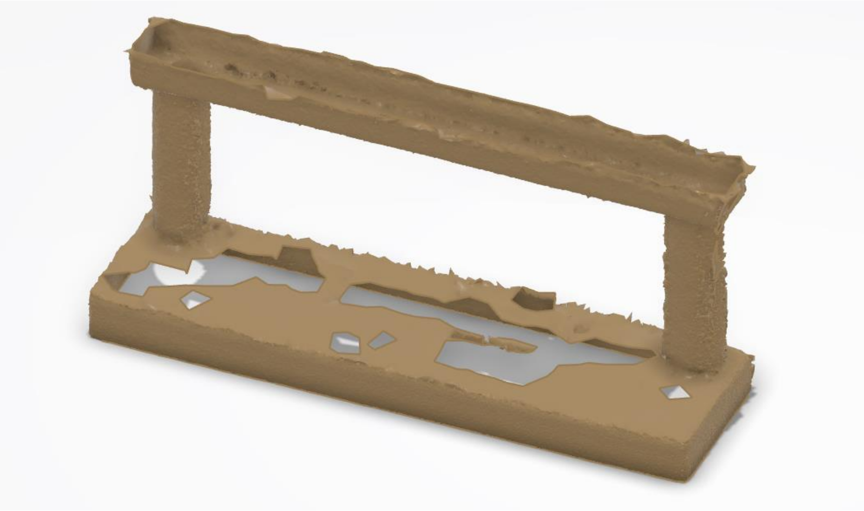
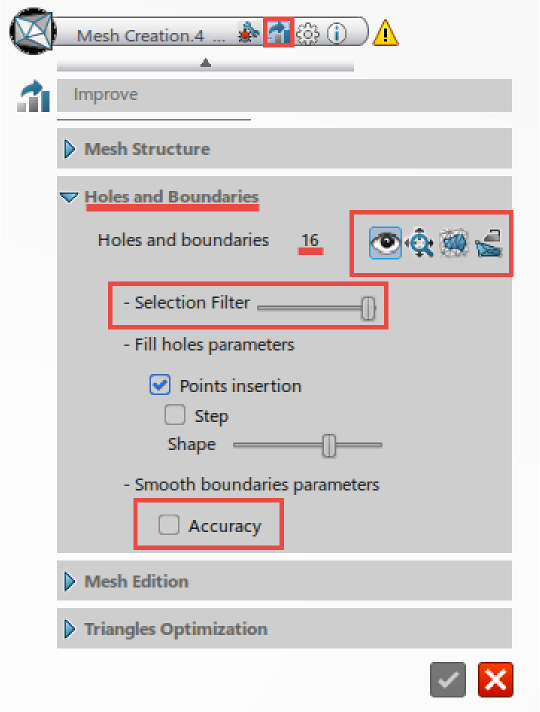


Figure 32 - Fresh mesh

By looking at figure 32, one can see, there are quite a few holes which need repairing for the surface to be complete. As mentioned earlier, the quickest ways of fixing errors in a mesh are by using the command *Mesh Preparation.* This will heal some of the errors but will not solve the hole-issue.



Mesh Creation.4 ...

Improve

Mesh Structure

Holes and Boundaries

Holes and boundaries 16

- Selection Filter

- Fill holes parameters

☑ Points insertion

□ Step

Shape

- Smooth boundaries parameters

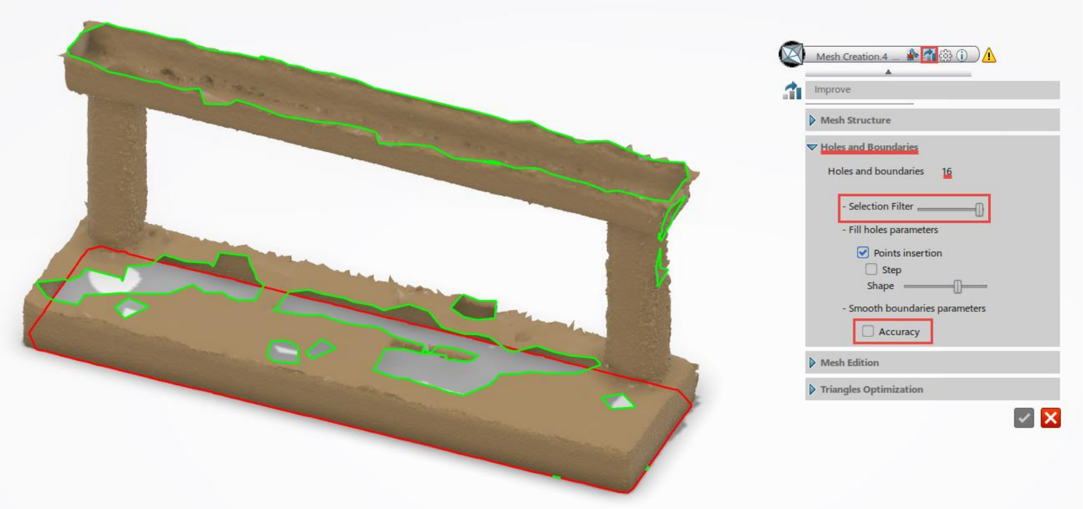
□ Accuracy

Mesh Edition

Triangles Optimization

Figure 33 - Holes and boundaries menu

Within the *Mesh Preparation* menu are quite a few tabs with extra sets of tools to refine the mesh. The second tab marked with a small square is *Mesh Improve* tab which contains specific tools to fix errors mentioned earlier such as holes, non-manifold edges and etc. This tab is a tool to fix them manually and not letting the program try its best, and sometimes this needs to happen. Further down in the menu are *Holes and Boundaries* which is what we are currently looking at. The software has detected 16 holes which needs repair. To fix the holes there are 3 approaches that will fix the holes.



Mesh Creation.4 ...

Improve

Mesh Structure

Holes and Boundaries

Holes and boundaries 16

- Selection Filter

- Fill holes parameters

☑ Points insertion

□ Step

Shape

- Smooth boundaries parameters

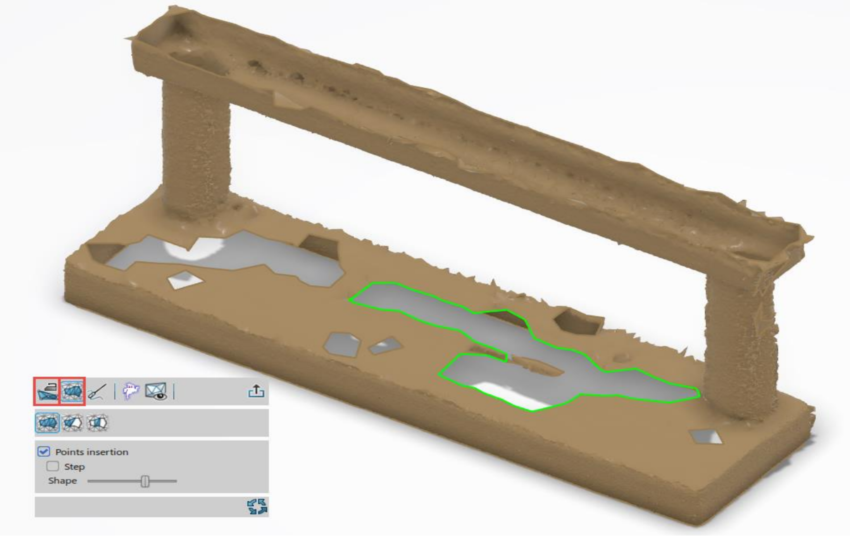
□ Accuracy

Mesh Edition

Triangles Optimization

Figure 34 - Holes marked in the mesh

The first setting is the *Selection filter* which is a sliding bar. This bar will allow one to section out holes that are really not holes, just as figure 34 shows. Red lines indicate they are not present in what it considers as a hole, but the green ones are. To prevent confusion, it can be put further to the left making it only select smaller holes. The second approach is to check to mark accuracy. This will lead to more accurate and smooth boundaries, making it easier for the software to calculate how the mesh should be connected to another surface. The third and last step is to use the command tools given, they are located just to the right of how many holes and boundaries the software detects, which can be seen in the red square over at figure 33.



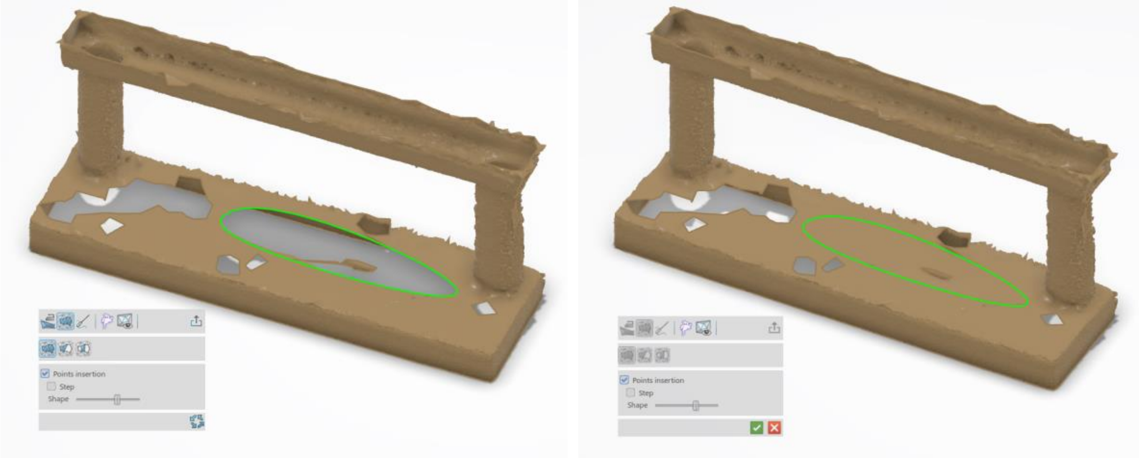
☑ Points insertion

□ Step

Shape

Figure 35 - Command tool menu

By clicking the eye-icon in the holes and boundaries menu, a closer look at each individual hole will show up, with another menu. In this command tool menu, there are two functions which can fix the given hole. As seen in figure 35, the green is marked as the given hole currently looking at. To start fixing, one can either go straight for the *Fill Hole* function and see if that works or perhaps the *Smooth boundaries* has to be used first.



☑ Points insertion

□ Step

Shape

☑ Points insertion

□ Step

Shape

Figure 36 - Smooth boundaries + Fill the hole

In figure 36 above both functions were used to first smoothen the boundaries, and then filling the hole. This is the general procedure for fixing holes in a mesh. The steps mentioned here can be used for all holes, and for some other errors such as nonemanifold edges. The auto-repair command normally takes care of most errors naturally but struggles more with holes, which is why these steps are necessary.

5.2.4 Generating robot programmable surfaces from mesh

Up until this point the workflow going from importing point clouds, how to refine the point clouds, how to mesh point clouds, how to refine meshes and how to fix major errors in meshes has been presented. The next stop is generating robot programmable surfaces from the mesh. What this means is to generate surfaces and 3D models allegeable to be simulated with robots in 3D Experience.

The first half of the workflow is solved in the app *Digitized Shape Preparation*, but for the next part, the app *Digitized Shape to Surface* is used. This app features a variety of different functions and commands. Before the user starts on anything new, make sure the whole structure is meshed and in the desired state. It should look like figure 38 below.

Figure 37 - Digitized Shape to Surface

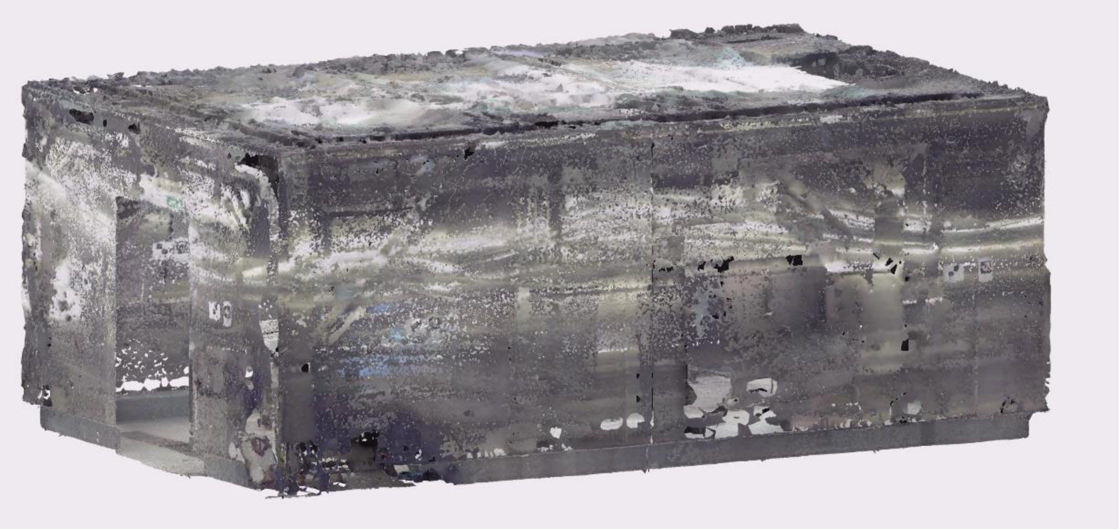


Figure 38 - Complete mesh of test cell

Now once the app is open and the mesh is in front of the user, the next step would be to add surfaces. The main command used is called *3D Primitive Recognition*. By selecting the command, a small toolbar appears on the screen just as in figure 39 below. In this toolbar certain options for how to detect surfaces are available. It will recognize cylinder shapes, planes, spheres, and cones. Select the appropriate shape before clicking on any surface of the mesh.

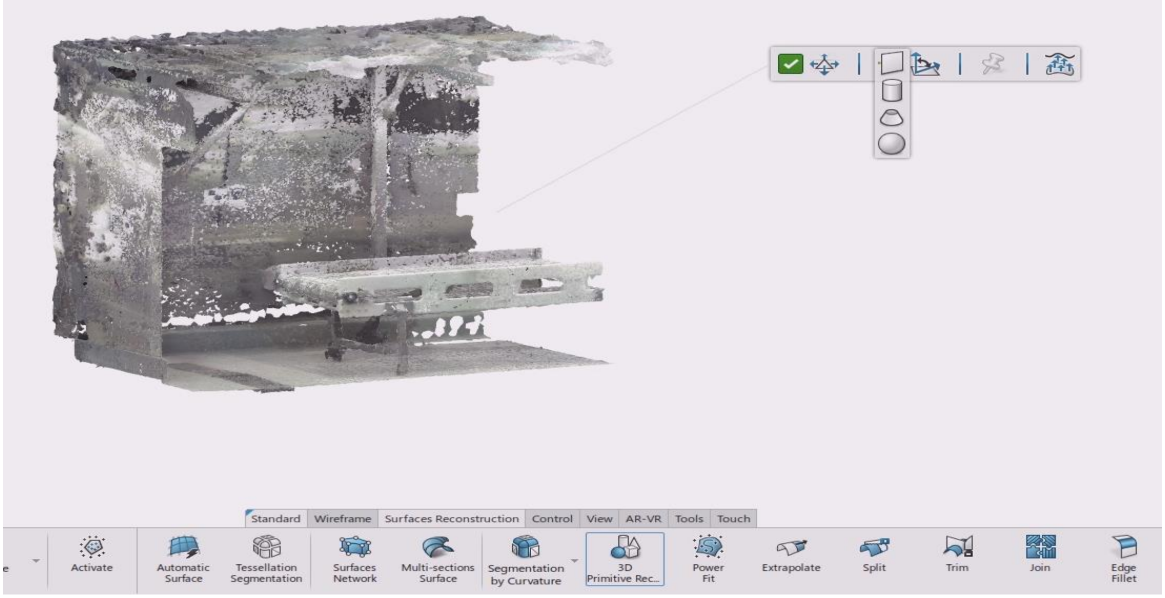
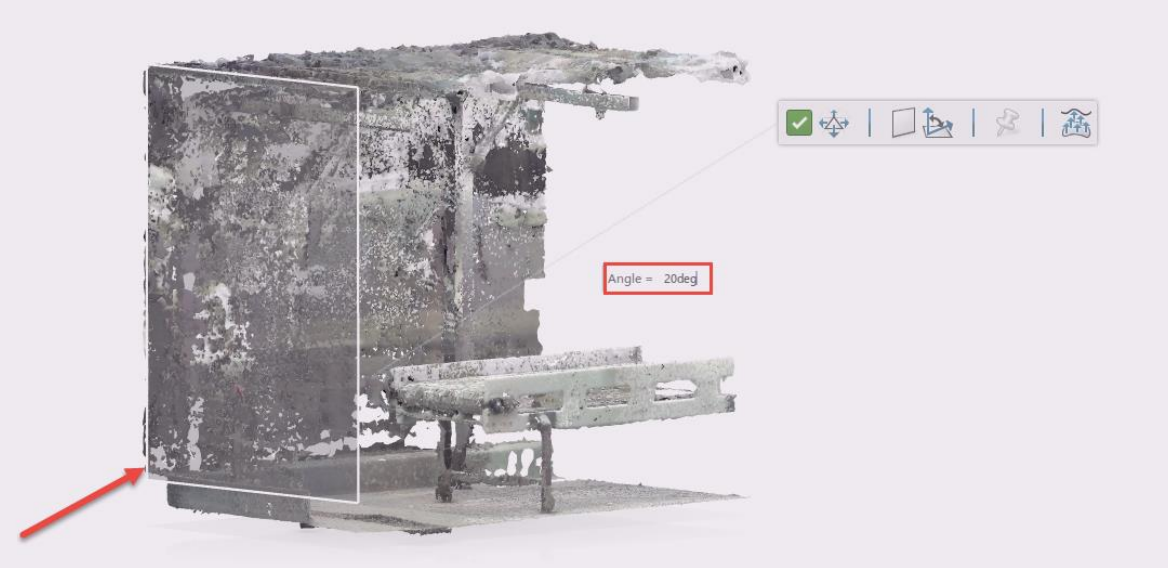


Figure 39 - 3D Primitive Recognition

But before clicking on the desired surface of a mesh, be sure to check the *Angular tolerance.* In figure 40 below, it shows that the angular tolerance is 20 degrees. What this tells the user is how much of an angle deviation the software will count as the same surface as the selected mesh. Any triangles in the mesh containing a deviation degree of over 20 degrees angle, will not be recognized as part of that selected surface. The lower the angle, the smaller the surface recognition, and the higher angle the bigger surface recognition. That still does not mean that the bigger the angle, the better the quality.



Angle = 20deg

Figure 40 - Angular Tolerance 20 %

It is important to understand that the tolerance angle needs to change from time to time depending on how the mesh looks. As just shown in figure 40 above, 20% angular tolerance for that side of the wall covers the whole surface. Reducing the angular tolerance to 10% as figure 41 below shows will only cover a small portion of the wall. This command requires some testing and experience before it can be used properly, and it is advised to keep angles between 8-40 degrees. Going over 40 degrees will distort the mesh and not give the desired result, and anything below 8 degrees will only net a very small surface. Even though it is recommended to keep it in between 8 to 40 degrees, keeping it between 15-25 degrees will most likely net a satisfying result.

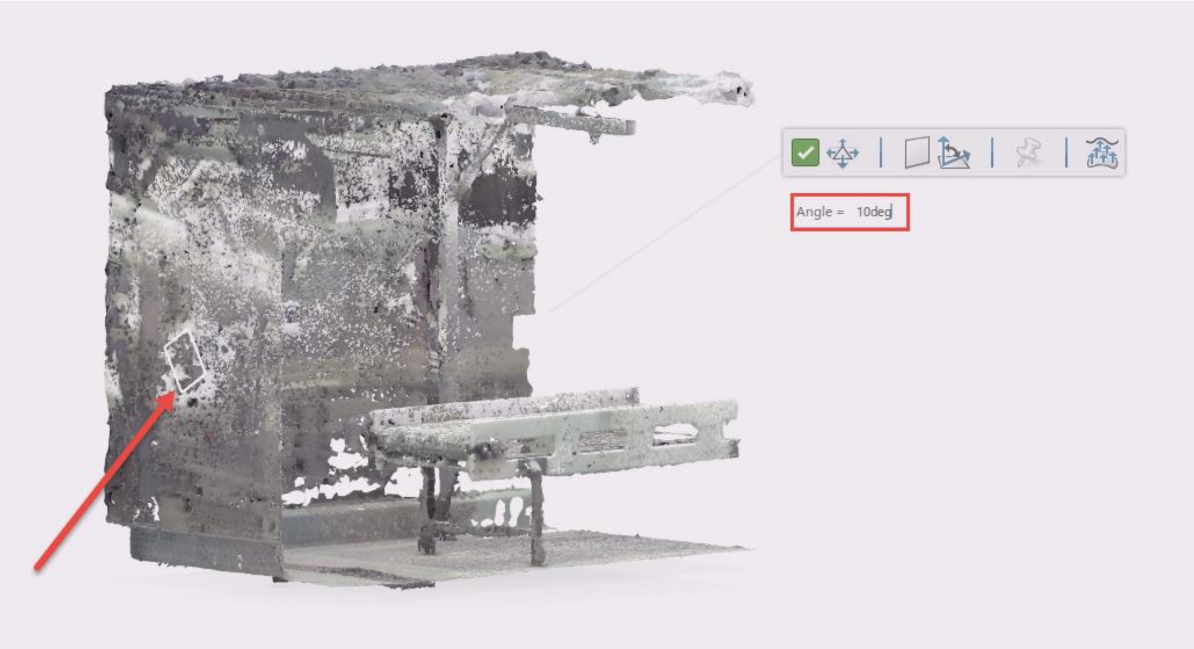


Figure 41 - Angular Tolerance 10%

Now with a desirable angular tolerance for selecting the surface, the actual surface creation can begin. It is advised to start out by larger surfaces and larges areas which are highly recognizable. Ones the larger areas are out of the ways, smaller and more distinct surfaces can be selected. In most cases the surface selected is too small for the actual surface it should represent.

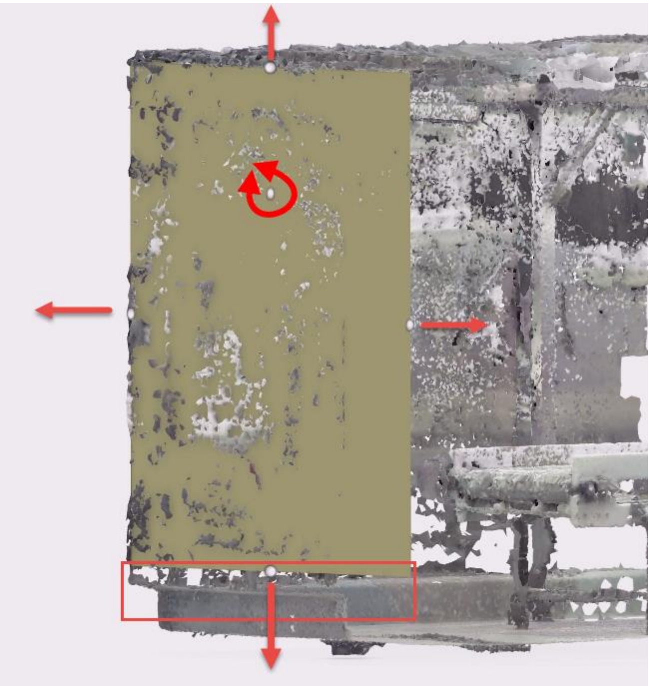


Figure 42 - Surface editing

Looking at figure 41 one page back, it shows a barely visible surface showing how the software intends to put the surface. These might not be as desirable as the user intended, but by double-clicking the surface it can be edited just as figure 42 above shows. It is possible to drag the surface in the plane direction while rotating it as well in-plane direction. As the red square box highlights on figure 42 above, there is a small gap not connecting the lower edge to the wall. By simply dragging the white dot downwards it will adjust the size of the surface to fit the geometry. When the desired geometry of the surface is created, it can simply be saved with double click once again on the surface, then immediately click the green verification symbol to save the current surface. Be aware that accepting a surface will consume that given mesh selected.

When the user has started to generate a certain number of surfaces, some surfaces are intersecting each other. This will come in most cases of edges and corners connecting different surfaces. In figure 43 below, one example of a corner can be seen intersecting each other. By using the command *Split* and selecting both surfaces they can together generate a connected corner. This is the main function for removing exceeding material. In some cases, *Extrapolate* must be used to extend the surface to fit the corner or edge, and then later be trimmed down with the use of *Split.*

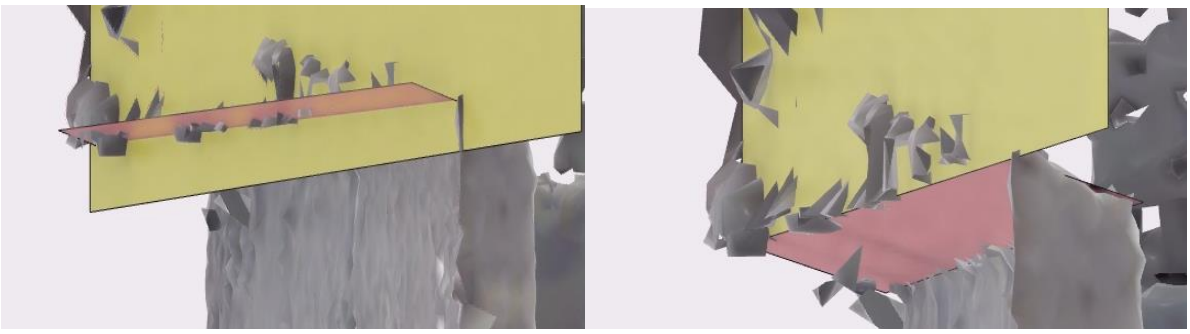


Figure 43 - Intersecting surfaces and connected corners

Combining the 3D primitive recognition and the split function will lead to a 3D surface model of the test cell. The steps presented here can be reused and repeated for all the different segments to get a total surface creation. Some surfaces require more patience due to difficult mesh, while others, such as the floor, is pretty much straight forward.

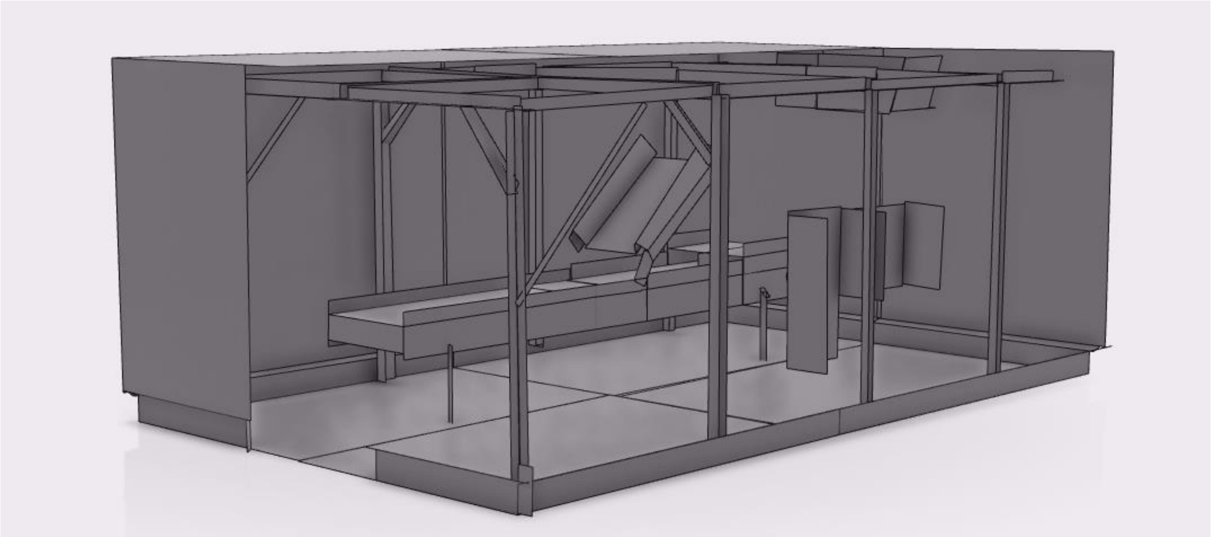


Figure 44 - Near complete surface creation

Figure 44 above shows a nearly complete model of robot programmable surface creation in 3D Experience. The walls are hidden to get a view inside. The author will further explain some aspect of the journey towards this result in the discussion section later. As it can be seen in figure 44 some components seem missing, which is an indirect effect of a messy point cloud.

5.2.5 The solid requirement for robot programming

In 3D Experience surfaces as created in *Digitized Shape to Surface* are ready and eligible for robot programming. The app called *Robot Surface Simulation* is what simulation app will be used to generate robot programs for the robot. Any surface generated on figure 44 above can be used in robot programming of the automatic cleaning solution, but they vary in accuracy and completeness. Some of the beams holding the roof are missing some spatial data, meaning some of the beams are extrapolated to create an illusion of reality. No information about the test cell was given outside of the point cloud and attached photos further back in the thesis. In some cases, the extrapolated surface might pose a problem, but it can work as well. While the surface might not be 100% accurate for the original test cell, a deviation of a couple of centimeters will not hurt. The cone for the nozzle for cleaning will be somewhere between 200 to 250 mm, meaning if it is a couple of centimeters off, it would not really hurt. As long as the nozzle covers the general area of the beams it should not pose a problem. It can either be extrapolated which has been done in this case, or it can be manually modeled and added into the test cell later, but without any drawings, the latter is not executed.

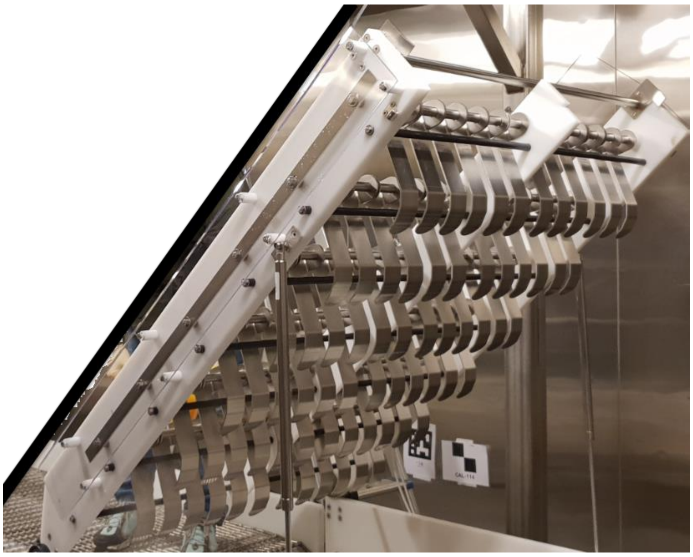


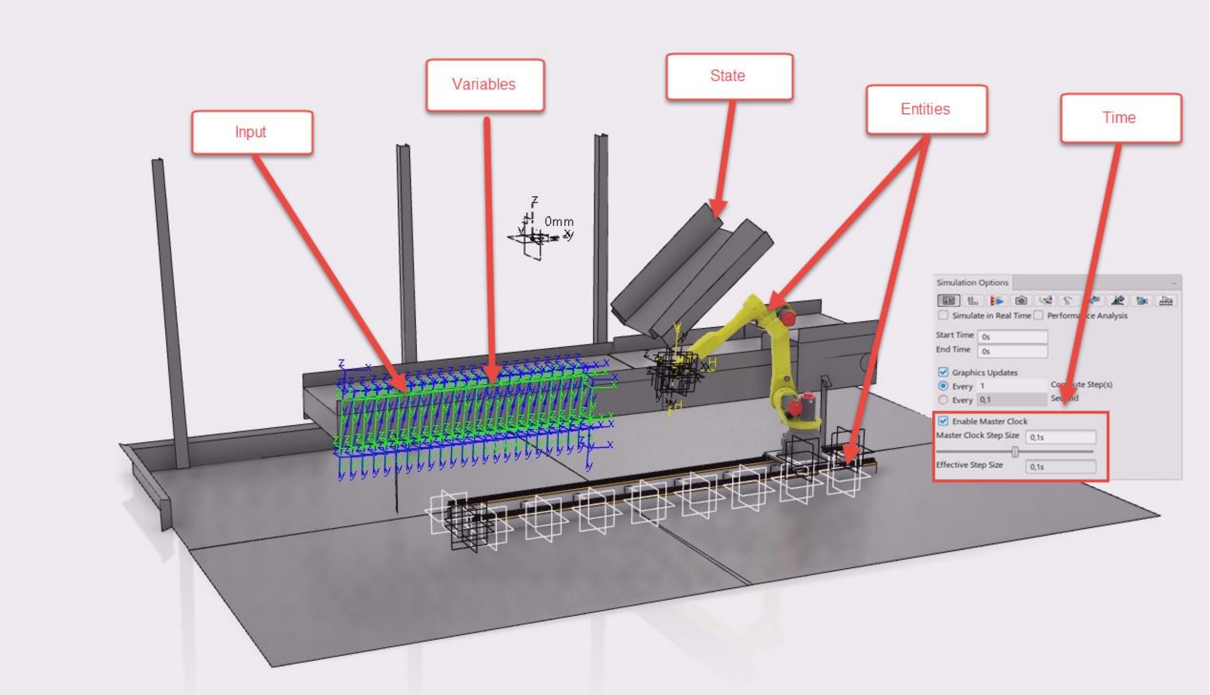
Figure 45 - Fish Stunner

Same can be said about the equipment placed in the test cell. The stunner has tenfolds of stunner blades and modeling them individually from spatial data is a real hassle. Instead of wasting a lot of effort into trying to create advanced surfaces onto the stunner a single surface covering the whole stunner can be added. The spray nozzle will move over the whole surface covering overall the whole stunner and not to cover each individual blade. From this, any potential solution can be added into the test cell and be run in a simulation environment.

5.3 Simulation results

5.3.1 Simulation set-up

When doing a case in a simulation it is important for the user to remember what that given simulation presents. By varying the inputs and some attributes, the whole output might change. Not only to change these input/outputs but understand what they are and what they do. These can change from the simulation environment to another simulation environment, they are not always equal. In this case there a quite a few of these components which will be covered to ensure the model and simulation set-up is understood by the reader.



Simulation Options

□ Simulate in Real Time □ Performance Analysis

Start Time 0s
End Time 0s

☑ Graphics Updates
● Every 1 Compute Step(s)
○ Every 0,1 Second

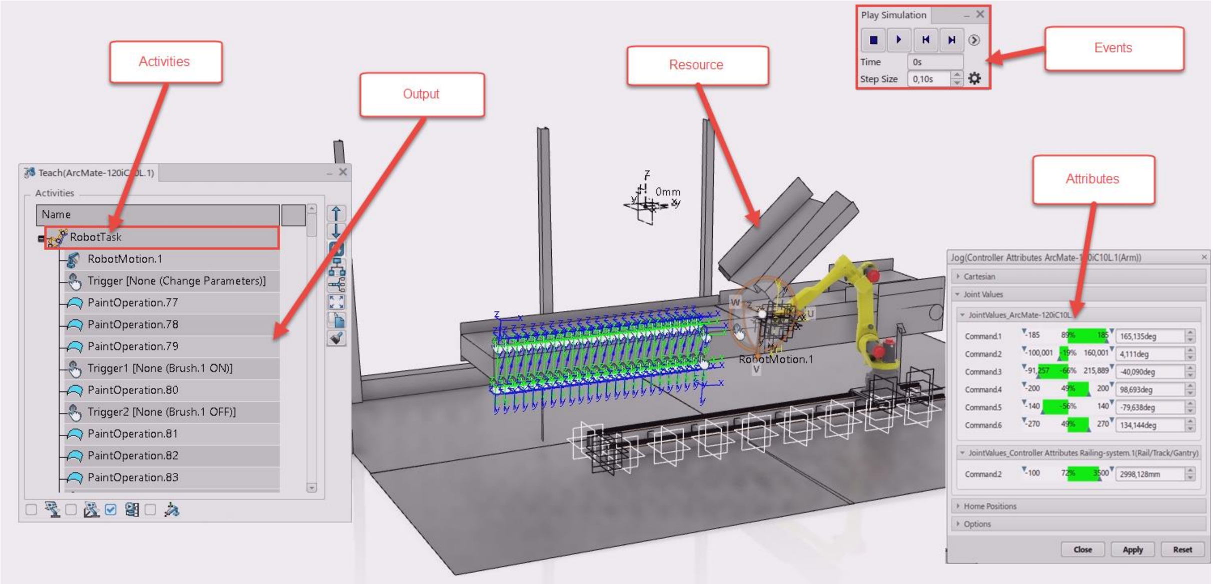
☑ Enable Master Clock
Master Clock Step Size 0,1s
Effective Step Size 0,1s

Figure 46 - Input, Variables, State, Entities and Time

The *Input* of this simulation case is the paint trajectories. An *Input* is the actions of the environment on a system, wherein this case where the robot should move and clean. These trajectories are inputs from the user but can be changed which leads them to also be a *Variable.* The same goes for the cleaning cone, which can be changed in size, flow and length.

When the input has been used on a surface it changes the *State* of the surface. It goes from being a surface for a visual representation of the stunner to a robot programmable surface. *Entities* are affected by inputs, and in this case, both the robot and the railing system are *Entities.* The input for trajectory will lead the robot which is now affected by what the user puts in a trajectory.

In this simulation environment, there are certain options regarding *Time*. A simulation model should contain the possibility to change the speed, both faster and slower if necessary. For this simulation model in 3D Experience are options for setting to run in real time, speed it up multiple times and also slowing it down multiple times.



Teach(ArcMate-120iC10L.1)

Activities

Name

- RobotTask
  - RobotMotion.1
  - Trigger [None (Change Parameters)]
  - PaintOperation.77
  - PaintOperation.78
  - PaintOperation.79
  - Trigger1 [None (Brush.1 ON)]
  - PaintOperation.80
  - Trigger2 [None (Brush.1 OFF)]
  - PaintOperation.81
  - PaintOperation.82
  - PaintOperation.83

Time: 0s
Step Size: 0,10s

Jog(Controller Attributes ArcMate-120iC10L.1(Arm))

Cartesian

Joint Values

JointValues_ArcMate-120iC10L.1

Command.1	-185	89%	185	165,135deg
Command.2	-100,001	-19%	160,001	4,111deg
Command.3	-91,257	-66%	215,889	-40,090deg
Command.4	-200	49%	200	98,693deg
Command.5	-140	-56%	140	-79,638deg
Command.6	-270	49%	270	134,144deg

JointValues_Controller Attributes Railing-system.1(Rail/Track/Gantry)

Command.2	-100	72%	3500	2998,128mm

Home Positions

Options

Close Apply Reset

Figure 47 - Activities, Output, Resource, Events, Attributes

Activities are processes in a simulation which in this case is the Robot Task performed by the *Entities.* These are different tasks or activities which defines a given process within the simulation. *The output* is often a measured quantity and in this case, there are two forms of output. The first is the robot program for how the robot should move. These can be extracted and used on a real robot after the *Output* is collected. The other *Output* is the visual aspect of seeing colors being sprayed onto the stunner. It can measure the overall coverage of the stunner.

Resource for this case is simply the surfaces which can be cleaned. There are a limited number of surfaces and eventually, every surface will be covered. It has constrained capacity. *Events* are when the *Entities* perform an *Activity.* By clicking the run simulation, the activity driven by the entity will perform an event. The last component used in a simulation environment is *Attributes. Attributes* are characteristic for the entities, and in this case, that is how the robot and railing moves. There is certain reach limit both for how far the railing system can move, but also how the joints of the robot can turn. Even the speed and inertia can be manipulated.

5.3.2 Simulation procedure

When the user has set up the simulation environment and simulation model to the specific case, a simulation procedure can be used. As shown earlier, the case is now in a state where all the point cloud has been turned into robot programmable surfaces. It is now time to use these surfaces to simulate options for automatic solutions. The app used for this purpose is the *Surface Simulation*, which primarily is used for painting and coating of components. Since the spray cone of a painting gun is quite similar to a hose nozzle spray, this simulation app can be used for simulating cleaning procedures.

Figure 48 - Surface Simulation

In this state, any desired solution or test rig can be implemented. There is no real limitation, except for it fitting within the test cell. To show and present the procedure a standard linear guiding system and a FANUC robot has been imported to showcase the procedure for developing programs for programmable surfaces.

The test cell will showcase an open room rather than the whole box. This is to highlight and showcase easier without having beams, roofs or walls cover important scenarios.

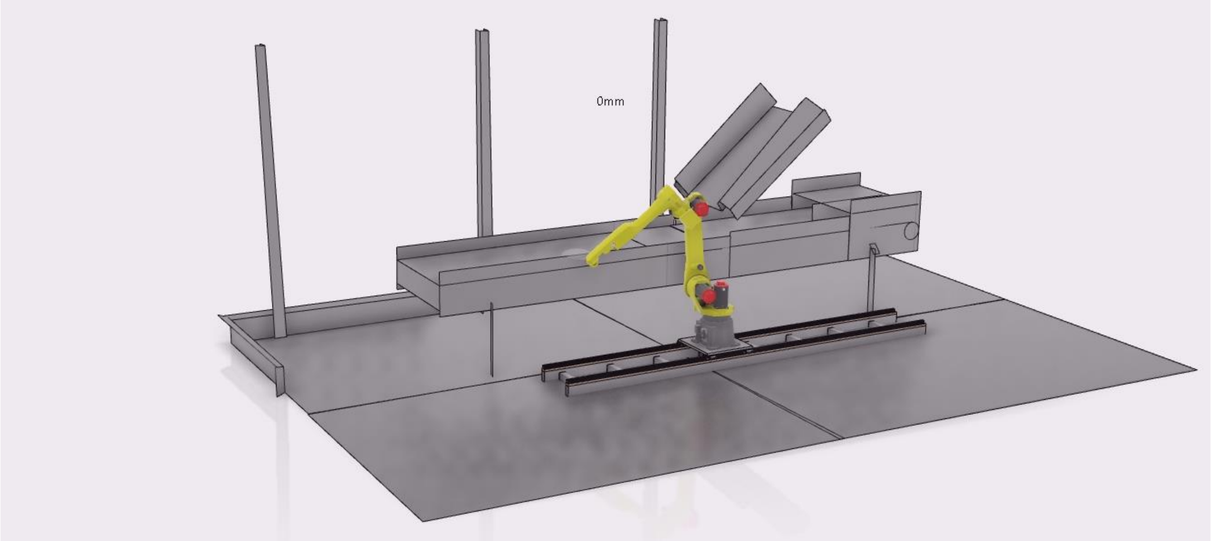
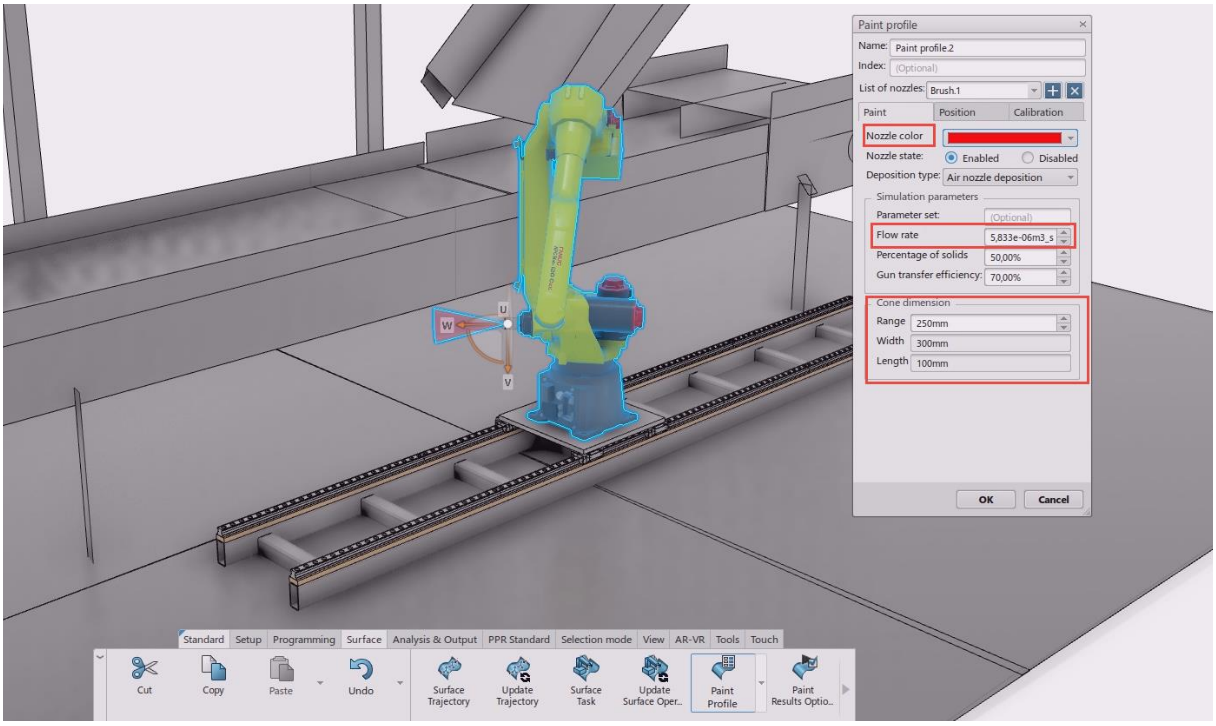


Figure 49 - Test cell with a linear guiding system and a FANUC Robot

Now that a given solution has been imported into the test cell, setting up the simulation can begin. The first approach that has to happen is to adjust the nozzle based on the real nozzle. A nozzle which is modeled is based on a real version, and the real version has some data specifying cone width, length, etc.



Paint profile

Name: Paint profile.2

Index: (Optional)

List of nozzles: Brush.1

Paint | Position | Calibration

Nozzle color: 🟥

Nozzle state: ● Enabled ○ Disabled

Deposition type: Air nozzle deposition

Simulation parameters

Parameter set: (Optional)

Flow rate: 5,833e-06m3_s

Percentage of solids: 50,00%

Gun transfer efficiency: 70,00%

Cone dimension

Range: 250mm

Width: 300mm

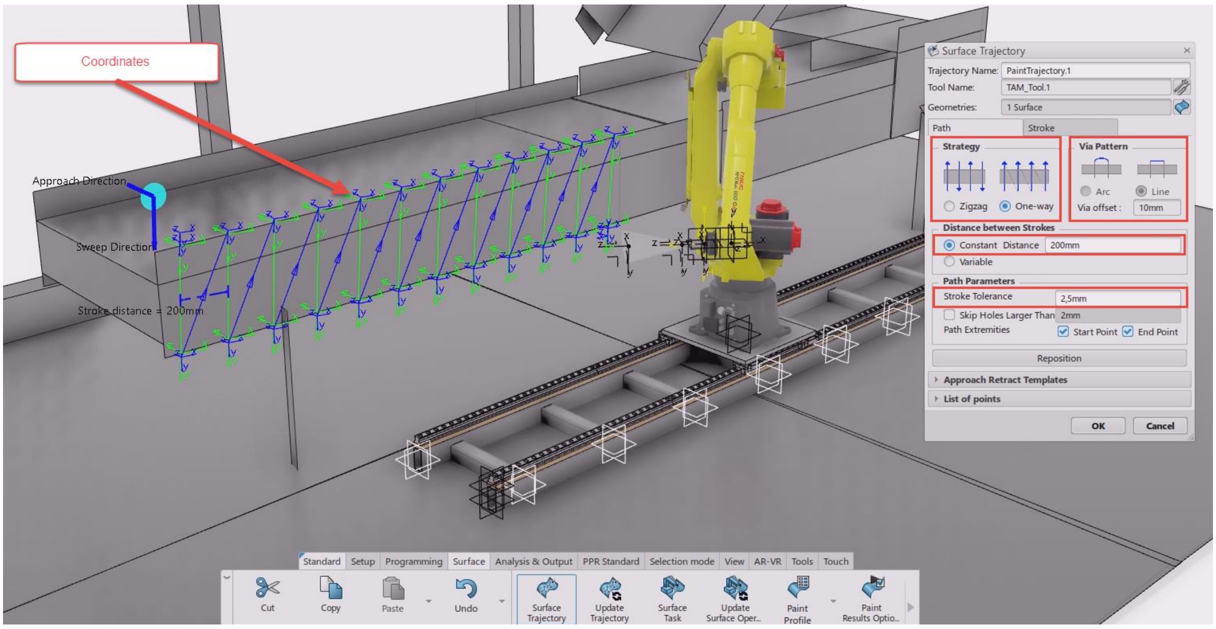
Length: 100mm

OK | Cancel

Figure 50 - Paint Profile

By clicking the command *Paint Profile* and then the robot, a dial box will appear. In this dial box, multiple options are available to change to fit the needs of the nozzle. For a visual representation, the color can be changed to anything that is more satisfying. Even the flow rate of the nozzle can be changed, together with how large that given cone is.

When the desired cone of size and color has been selected, the next step can proceed. The robot needs to know what is selectable and usable. A command known as *Surface Trajectory* will set up a moving path for the robot by giving it coordinates based on the robot programmable surfaces.



Surface Trajectory

Trajectory Name: PaintTrajectory.1
Tool Name: TAM_Tool.1
Geometries: 1 Surface

Path | Stroke

Strategy
○ Zigzag ● One-way

Via Pattern
○ Arc ● Line
Via offset : 10mm

Distance between Strokes
● Constant Distance 200mm
○ Variable

Path Parameters
Stroke Tolerance 2,5mm
□ Skip Holes Larger Than 2mm
Path Extremities ☑ Start Point ☑ End Point

Reposition

Approach Retract Templates

List of points

OK Cancel

Figure 51 - Surface Trajectory

As figure 51 above shows, when pressing the *Surface Trajectory* and the robot a dial box with even more options appear. Based on how the robot should move, if it should move zigzag or in an arc or perhaps in straight line movements. The important option is the distance between each coordinate point. Having a cone of 100 millimeters width and a gap between each point for 200 millimeters will leave a gap with no color. Therefore, it is important to remember what the cone is set to and base the distance coordinate thereafter. Even a tolerance can be set for how much deviation the robot could potentially move.

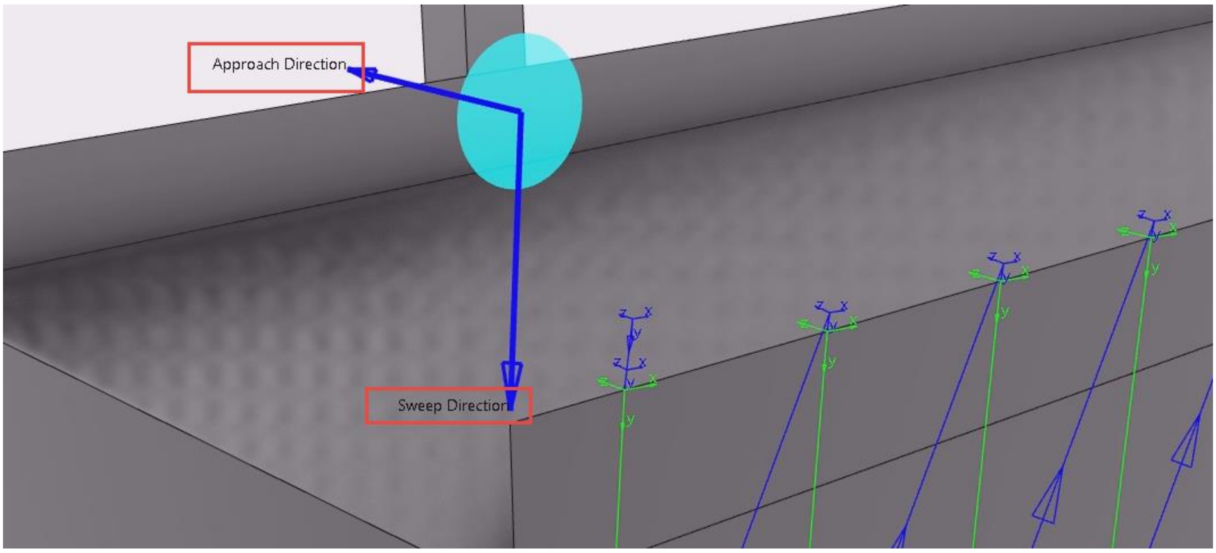
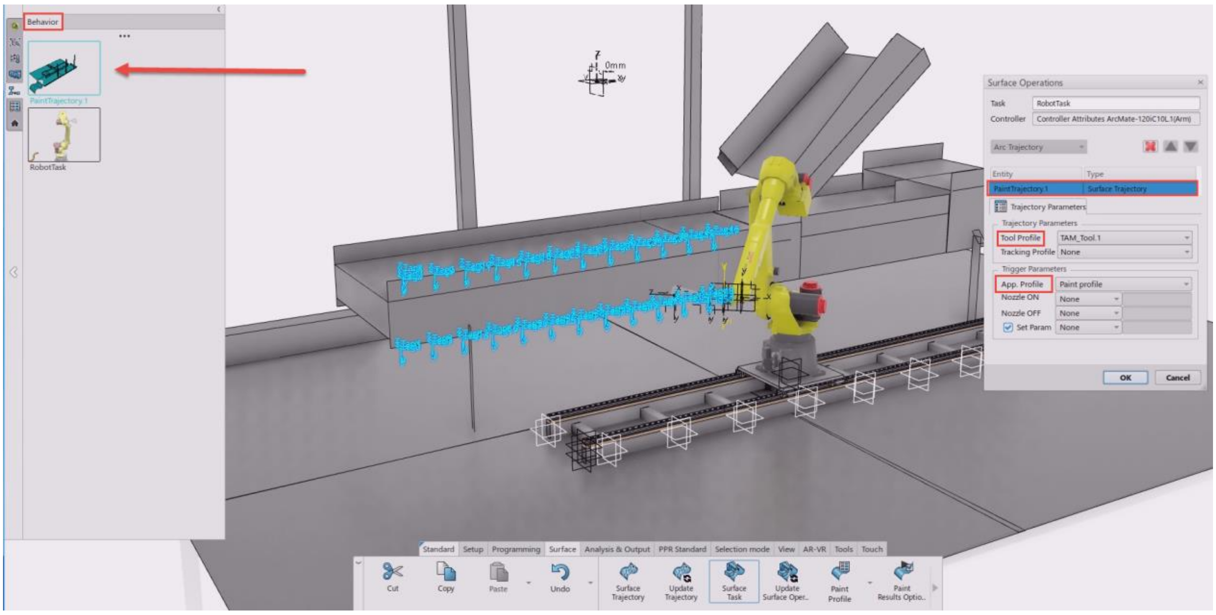


Figure 52 - Approach Direction and Sweep Direction

There is however one specific illustration on the surface trajectory which is important. In figure 52 above it shows two arrows facing a direction named *Sweep Direction* and *Approach Direction*. What these tell the user is how the movement of the trajectory should behave. The approach direction is facing towards the stunner, which means the robot will face towards the stunner as well. Sometimes these arrows will point outwards, meaning the robot will try to clean it from inside the stunner, which is impossible. This is extremely important to pay attention to otherwise the whole robot program will crash in an instance. The other arrow showing *Sweep Direction* is showing which way the movement "Sweeps" over the surface. Since the arrow is facing down, that means the movement will always start on top and move downwards. This arrow can be turned up, left, right or down depending on how the movement should be. Leaving it facing left, will let the robot always try to move the tool from right to left.



Surface Operations

Task: RobotTask
Controller: Controller Attributes ArcMate-120iC10L.1(Arm)

Arc Trajectory

Entity	Type
PaintTrajectory.1	Surface Trajectory

Trajectory Parameters

Trajectory Parameters
Tool Profile: TAM_Tool.1
Tracking Profile: None

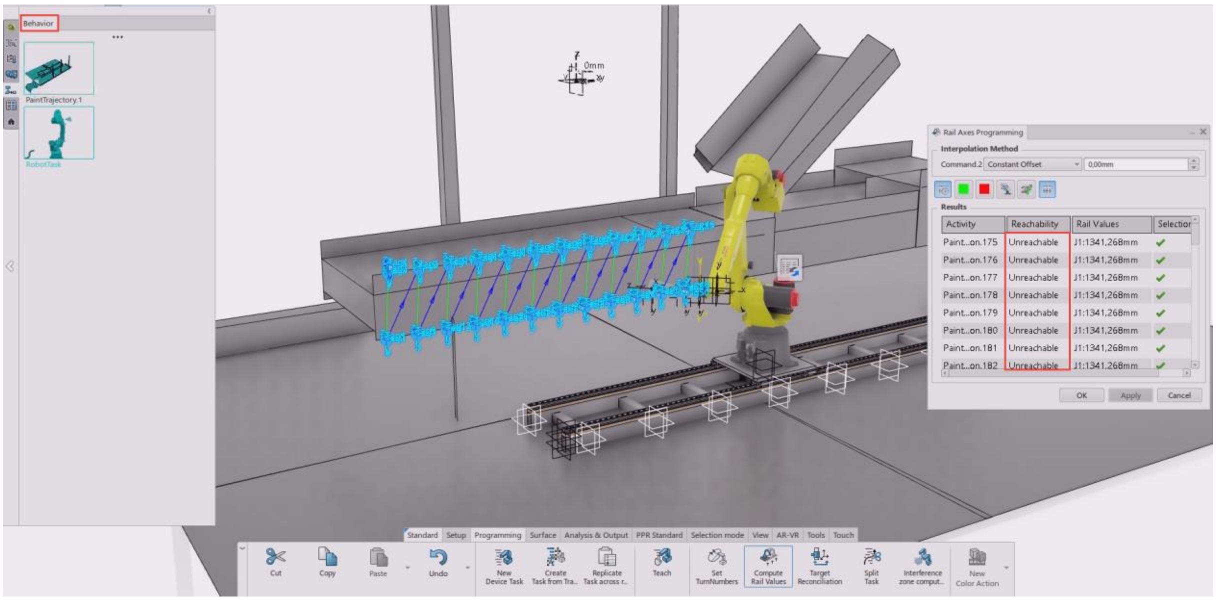
Trigger Parameters
App. Profile: Paint profile
Nozzle ON: None
Nozzle OFF: None
☑ Set Param: None

OK
Cancel

Figure 53 - Creating Robot Tasks

Now that the tooling of the robot, the nozzle, has a cone attached and the robot has some coordinates to follow, a robot task can be created. By finding the command *Surface Task* in the action bar and selecting the robot a dial box will show. The dial box asks the user for what trajectory the robot should follow to complete its task, and with which tool profile. In the hot bars to the left under *Behavior* is where all tasks and trajectory are stored. By selecting the desired trajectory, the coordinates become turquoise and the trajectory is added to the task. Be sure to check if the tool profile and paint profile is correct as well. In some cases, there can be multiple cone sizes depending on which area it should clean, or perhaps the flow and pressure of the nozzle are adjusted.

Clicking okay will save the desired task like a robot program for that specific program. Normally that program can now be exported and put into the real version of that robot where the programmed program will run. But before that can happen one last thing has to be implemented into the program. Since the robot is mounted on a railing system both of them have to be connected to each other. Playing the given simulation with the state it is currently in, will leave the robot moving as far as it can, then stop because it is not able to move out of its joint limits. Therefore, by connecting the rail to the robot motion a combined effort to move and clean all the appointed coordinates will happen.



Interpolation Method

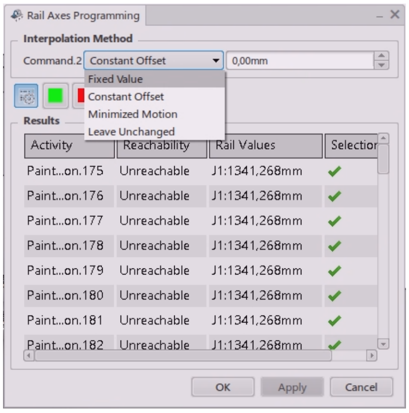
Command.2 Constant Offset 0,00mm

Results

Activity	Reachability	Rail Values	Selection
Paint...on.175	Unreachable	J1:1341,268mm	✓
Paint...on.176	Unreachable	J1:1341,268mm	✓
Paint...on.177	Unreachable	J1:1341,268mm	✓
Paint...on.178	Unreachable	J1:1341,268mm	✓
Paint...on.179	Unreachable	J1:1341,268mm	✓
Paint...on.180	Unreachable	J1:1341,268mm	✓
Paint...on.181	Unreachable	J1:1341,268mm	✓
Paint...on.182	Unreachable	J1:1341,268mm	✓

Figure 54 - Unreachable targets

In the action toolbar, a specific function to fix the connecting issue with the robot and rail can be found. By clicking the command *Compute Rail Values* and then selecting the desired robot task, a dial box will appear. Looking at figure 54 above it shows clearly that most of the points are not reachable in the position of the robot. There is however a tool inside the dial box which will allow the rail system to calculate the necessary position of itself to be able to get the robot to reach the coordinates.



Command.2 Constant Offset 0,00mm

Activity	Reachability	Rail Values	Selection
Paint...on.175	Unreachable	J1:1341,268mm	✓
Paint...on.176	Unreachable	J1:1341,268mm	✓
Paint...on.177	Unreachable	J1:1341,268mm	✓
Paint...on.178	Unreachable	J1:1341,268mm	✓
Paint...on.179	Unreachable	J1:1341,268mm	✓
Paint...on.180	Unreachable	J1:1341,268mm	✓
Paint...on.181	Unreachable	J1:1341,268mm	✓
Paint...on.182	Unreachable	J1:1341,268mm	✓

Figure 55 - Rail Axes Programming

Within the drop-down menu inside *Rail Axes Programming* are three options. Having it set to a fixed value, will look at the parameter next to it showing a given length. Setting this to 1000 mm will leave the rail at that specific point. This does normally not work, because we want the rail to be able to move together with the robot, not just stay in one place. The second option is a constant offset. Setting the parameters to 200 mm will leave the robot to move 200 mm at the time from point to point. This option needs more testing by the user to get it right, but normally when the coordinate gap distance is set to 200 mm, so can the constant offset in rail axes programming. The third option is minimized motion, which will try to follow all the coordinate points and move as little as possible. This is normally a good option because the rail will follow the robot for each point, meaning the robot will never be out of range.

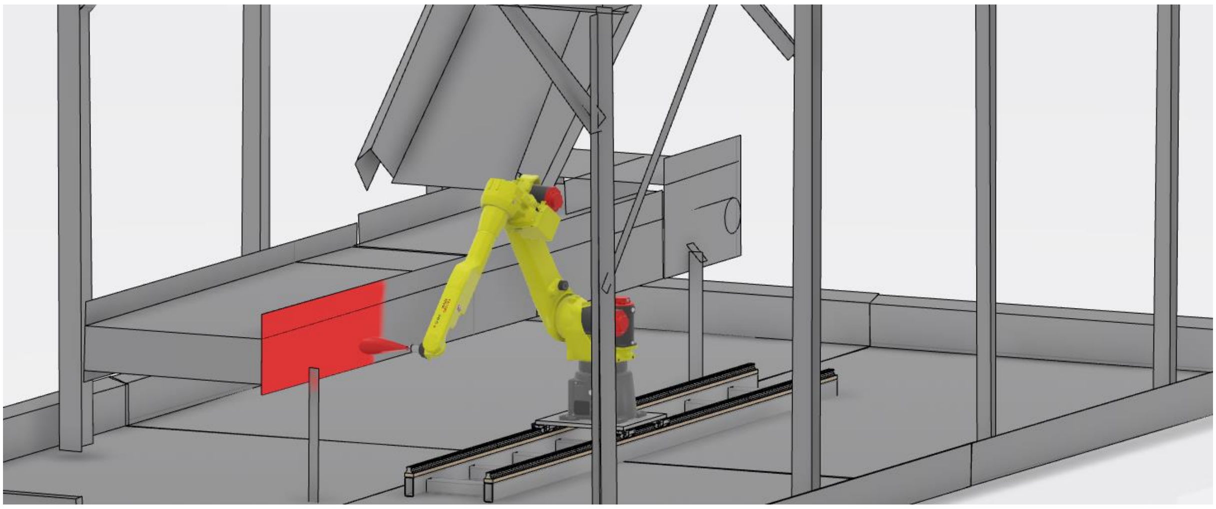


Figure 56 - Running the simulation

With the railing working together with the robot, the simulation can be played to see the coverage of the procedure. As it can be seen from figure 56 above, it cleans the desired surface selected. All that is left now is to program every single surface and then install the robot onto the test cell and plug in the memory card and let the program run.

6 Discussion

It is important when analyzing and discussing this thesis that the given results are based on the methods used to solve the research questions. By solving it by other methods and other boundaries, other results may occur.

6.1 Developed procedure

The procedure stated in the previous result chapter shows a way of achieving robot programmable surface from scratch with the use of a common platform, in this case, 3D Experience. The procedure is optimized iteratively to give a more general approach for how this can be achieved but works best on the intended software. The procedure is developed by theory from chapter 2.3 and through the author's understanding and testing as well. Chapter 2.3 shows different ways of going from spatial data to a 3D model in different software and fields. These have some common aspects which can be generalized for all fields and 3D spatial data handling. By looking at the given research and trying to implement this into a given software, a procedure is developed.

This procedure is similar to the ones gathered in the state-of-the-art chapter, but it contains one new major field, robot simulation. As mentioned earlier there are certain fields where the spatial data has been used for visualization and 3D printing. These are the major uses of 3D spatial data and as far as the author can tell, this method of using 3D spatial data together with robot programming is brand new. With the use of AR, a small test could be run on the software to see how these stated state-of-the-art procedures held up while adding a new step. By taking actions trying to execute what was gathered in the literature review the procedure could be developed further, which is the end results presented above.

RQ1: "What are the procedures required to obtain virtual robot programmable surfaces from point cloud?"

To be able to handle and use point clouds a certain amount of work has to be put into the given spatial data. As the author already has established, a certain amount of procedures already exists up until 3D models. Depending on field and use, these can vary and have alternative methods of achieving 3D models from spatial data. Since it is already possible to achieve 3D models from spatial data, adding one more step for robot programmable surfaces should be possible, which it is. It is important that the procedure can handle the spatial data correctly to be able to generate mesh, which is further converted into 3D solids or surfaces. These can then be adjusted in 3D Experience to be usable as robot programmable surfaces. The procedure is a general step towards robot programmable surfaces which can be used by other platforms or software.

The general procedure developed is a 7-step process which goes all the way from start to finish, which can be found in chapter 5.1.2. This procedure is not a detailed step by step guide to achieving robot programmable surfaces, but the process required to develop robot programmable surfaces.

6.2 Workflow and simulation

The simple case was the initial starting point for finding a suitable workflow for how to use the procedure properly for the common platform. By setting up a case study a thorough testing process could be initiated to gather information and suitable commands for the given software. Once the simple case had been run through various scenarios and testing, a general workflow was developed. This workflow was then attempted to be used on the industrial case, which leads to the improvement and the given workflow stated above in the result chapter.

The workflow was developed with an industrial mindset of efficiency and ease. It is important that this workflow can be used for something other than testing and case studies. By narrowing down the number of functions used and optimizing the use of these function can lead to a possible opportunity for use in the industry. When Manufacturing Lab is up and running at NTNU, this could potentially be run through some testing and verification before putting it into a fish processing facility. It would be recommended to test it out, but in theory, this should be usable.

RQ2: "How can the procedure be executed on a common software platform?"

Now with an addressed procedure for how to develop robot programmable surfaces, a way of doing the procedure for a common platform has been presented. The whole idea of doing it on a common software is to remove the unwanted switching of software which leads to a lot of licenses and time to learn all the different software. When a common platform can deliver all the stated 7-step procedure going from spatial data to robot programmable surfaces, and also generate robot codes for the robot, that should be considered a top priority. 3D Experience was the selected software due to the fact that the software could handle the whole procedure on that specific platform. To be able to trace what has been tested and executed on this common platform a workflow for how to achieve robot programmable surfaces was conducted. This workflow can be reused for any similar case handling spatial data within 3D Experience, meaning it can be influenced further and be used industrial.

RQ3: "How can the procedure help lower the time of on-site robot programming for robot cleaning of fish factories?"

The whole idea of doing everything on software is to remove on-site measuring and programming. By conducting a 3D scan of the facility to then move everything off-site to program offline to generate robot procedure is desirable. There is no way a fish processing facility will shut down for days or weeks at the time to add robots. The amount of money they would lose in such an investment is probably not going to be worth it. But with this method, spending time when the facility is empty to scan the room to then fix everything except the installation phase, to work off-site is possible. Not only would it not disturb the ongoing production, but with simulation, the board for that fish processing facility can have some previews before deciding the final design. Unfortunately, the installation will take the same amount of time, but compared to what it would be for programming on-site is very low. By doing it off-site and shutting down for installation could be beneficial, but this has to be further checked and verified. There is no way of telling at this point in time how long it would take to install and no way of telling what the cleaning design might be.

The most suitable approach would be to 3D scan a facility to then work off-site to create a virtual model of the given facility. By contacting the manufacturer of the fish processing equipment, 3D models should be obtainable and can then be put together with the virtual model. This way a detailed model of the equipment can be put into the virtual environment with spatial data showing exactly where it should be. By combining the models with the virtual model, a near complete virtual representation of that facility can be made and then further set up for robot programming in a simulation environment. By using the simulation tool within 3D Experience, robot codes can be generated based on the simulation. This simulation could be improved multiple times before actually used on a real-life robot. Doing this could prevent bad robot movement and also bad cleaning.

7 Conclusion

The findings from this thesis show that it is possible to use spatial data in aquaculture to create automatic cleaning solutions. By removing a huge amount of on-site work to put it all virtually to be able to work off-site is very beneficial. There is no fish processing company which would allow to shut down their production for weeks or months to install and program robots. Therefore, by scanning the environment, creating a virtual model and then implementing robots, much of the work can be achieved off-site. Simulating and designing the given cleaning solution can be run through multiple iterations to get the best results. By looking at the movement of the robot together with the desired railing system, simulation can give you how much that specific design is covering the facility. Improving on the simulation and the design could potentially lead to a fully automatic cleaning solution of fish processing facilities, but it needs more testing and verification.

With an automatic solution manual labor for cleaning of fish processing facilities can be reduced and leaving the heavy-duty work for the robots instead of workers. The harsh environment the works are exposed off is not healthy and the movement required could lead to back and shoulder issues. Not only will it lighten the workload of workers, but it will have a more systematic approach for cleaning. The robot will clean exactly what it was told to do in the same order and precision every time, a robot cannot have a “bad” day at work. With traceable cleaning and simulation, every surface or corner can be cleaned properly to reduce listeria and prevent bacteria growth.

In theory, this procedure and workflow developed should be able to be used at an industrial level, but it would be advised to test it out further by a more experimental method of actual robots and equipment. This thesis can be the build blocks for future projects for automatic cleaning solutions for fish processing facilities.

8 Future Work

With this thesis, a developed procedure has been developed but used for a testing cell at Optimar. As stated in chapter 3.1 assumptions and limitations, a more industrial case would be more desirable but was unfortunately not acquired. It would be valuable to continue working with this procedure for a more industrial case, by looking at bigger aspects of the facilities. This means a thorough scan of a given facility with high precision and of a large scale. Eventually design and develop a thought automatic cleaning solution with sprinklers, robots, linear guiding system and what not, to fully test and implement a given solution to the 3D scanned version. This can also be taken a step further by actually investing in that given system and test it in the future Manufacturing Lab at NTNU in Ålesund.

Another aspect which can be looked at further is how one could refine the mesh easier and faster. Till this date, everything has been executed manually in the given software and could potentially be generated by a code or program to run or to refine the mesh to the desired state. Not only refine the mesh but perhaps also detection of surfaces from mesh having that part created automatically. 3D Experience as a tool has the potential to develop own modules or programs to run parallel on the platform, giving it the possibilities to create almost whatever is necessary, with enough experience that is. Having such a program run and detect what would be a surface and thereby create robot trajectories automatically would be really interesting. What could happen if these procedures presented in this thesis was automatically adapted to do it for you?

A possible quantitative study can also commence from this study, with the look at other software solutions. This could highlight possible positives or negatives with each respective program to fully understand the capability of what has been solved. The aspect of time, effort levels, ease, and usability can be looked at. There might be other and better alternatives to offline program robots from spatial data.

9 References

1. Smith K. History of 3D scanners https://www.modena.co.za/history-of-3d-scanners/: MODENA; [

2. Daneshmand M, Helmi A, Avots E, Noroozi F, Alisinanoglu F, Arslan H, et al. 3D Scanning: A Comprehensive Survey2018.

3. Lindskog E, Vallhagen J, Berglund J, Johansson B. Improving Lean Design of Production Systems by Visualization Support2016. 602-7 p.

4. Våre norske regioner http://marineharvest.no/about/vare-regioner/: Marine Harvest; 2019 [

5. Norges største oppdrettsselskap marineharvest.no/about/norges-storste/: Marine Harvest; 2019 [

6. Bi ZM, Wang L. Advances in 3D data acquisition and processing for industrial applications. Robotics and Computer-Integrated Manufacturing. 2010;26(5):403-13.

7. Y. Lee S, Majid Z, Setan H. 3D data acquisition for indoor assets using terrestrial laser scanning2013. 221-6 p.

8. Gong L, Berglund J, Wang Z, Larborn J, Skoogh A, Johansson B. Improving Manufacturing Process Change by 3D Visualization Support: A Pilot Study on Truck Production. Procedia CIRP. 2016;57:298-302.

9. Sansoni G, Trebeschi M, Docchio F. State-of-The-Art and Applications of 3D Imaging Sensors in Industry, Cultural Heritage, Medicine, and Criminal Investigation. Sensors (Basel, Switzerland). 2009;9(1):568-601.

10. M. Bures JP. Comparison of 3D Scanning and 3D Modelling of a Workplace from Various Aspects IEEE. 2016.

11. Randall T. Client Guide to 3D Scanning and Data Capture. In: James Kavanaugh CK, Sam Pfeifle, Elmer Bol editor.: BIM Task Group; 2013.

12. Lindskog E, Berglund J, Vallhagen J, Johansson B. Visualization Support for Virtual Redesign of Manufacturing Systems2013. 419-24 p.

13. Eriksson A, Sedelius E, Berglund J, Johansson B. Virtual factory layouts from 3D laser scanning – A novel framework to define solid model requirements. Procedia CIRP. 2018;76:36-41.

14. Chen J, Clarke K. Rapid 3D modeling using photogrammetry applied to Google Earth2016.

15. White P, Ingalls R. Introduction to Simulation2009. 12-23 p.

16. Musselman KJ. Guidelines for Simulation Project Success1993. 58-64 p.

17. Milan Gregor ŠM, Józef Matuszek, Andrej Štefánik. Digital Factory. Journal of automation, mobile robotics & intelligent systems. 2009;3.

18. Paulic M, Irgolic T, Balic J, Cus F, Cupar A, Brajlih T, et al. Reverse Engineering of Parts with Optical Scanning and Additive Manufacturing. Procedia Engineering. 2014;69:795-803.

19. Azhar S, Ahmad I, Sein M. Action Research as a Proactive Research Method for Construction Engineering and Management2010.

20. Yin RK. Case Study Research: Design & Methods2009.

21. Teegavarapu S, Summers J, Mocko G. Case Study Method for Design Research: A Justification2008.

22. Benjaminsen T. 3D-scanning in Aquaculture Engineering: Improving Spatial Data Capture Methods Using Terrestrial Laser Scanning: NTNU Ålesund; 2018.