LAB University of Applied Sciences

Creation of Common Hand Tools for Simulation purposes in Motion Simulation Project Platform

Harold Ludric Kah Tujilo
LAB University of Applied Sciences
Bachelor of Engineering, Mechanical Engineering and Production Technology
2022

Abstract

Author	Publication type	Published
Harold Ludric Kah Tujilo	Thesis, UAS	2022
	Number of pages 31	

Title of Publication

Creation of Common Hand Tools for Simulation purposes in Motion Simulation Project Platform

Degree and field of study

Bachelor of Engineering, Mechanical Engineering and Production Technology

Name, title, and organisation of the thesis supervisor

Eero Scherman, Lecturer, LAB University of Applied Sciences

Name, title, and organisation of the client

Aki Mikkola, Professor, LUT University

Adam Klodowski, Associate Professor, LUT University

The ability to realistically predict real-world observations has shown to be a key technology to remain competitive anywhere in the technology sector where for example factory manufacturing could be needed, such as car manufacturing. By addressing this, a software tool providing the required functionality can be realized, ultimately enhancing planning on manual assembly processes. MOSIM is an ITEA 3 project focusing on Modular Simulation of Natural Human Motions.

This thesis goes through the process of creation and implementation of a technical toolkit, which consists of commonly found hand tools. These tools shall serve their purpose in the MOSIM created software used for simulations.

CAD (computer-aided design) software SolidWorks was used to design the tools to realise this work. These were then imported into Blender and exported with the right properties. The files were then inserted into the Unity program where final changes were made. Some of the tools had to be equipped with C# code which would, in turn, create drop-down menus showing the different sizes and options available for that tool.

The resulting tools and the code which were created during this thesis were then handed off to be used in the rest of the project. These tools will be used for human motion simulation purposes in the project.

Keywords

MOSIM, Tools, Design, Creation, Simulation

Contents

1 Introduction

At the moment in Europe and all over the world there are a lot of different human simulation tools available, with Siemens being a great example (Siemens). Globally, human motion simulation and digital modelling activities have been major fields of innovation with potential ramifications for a variety of sectors such as automotive, truck, healthcare, and construction, and that could even affect the gaming industry, so hence despite their differences in scope, the ability to predict real-world observations is a key technology for staying competitive. Research projects related to smart factories (Tulip) already address this trend in mechanical and mechatronic components, whereas commercial tools do not yet address sufficiently the problem of generating a rich repertoire of realistic human movements in complex and collision-prone environments. (MOSIM Project.)

More than 20 international partners are involved in MOSIM, an international project that aims to develop a standard for combining digital human simulation approaches. The project, in this case MOSIM, aims to develop an open standard for integrating approaches to digital human simulation. It will design and implement a generic mock-up interface inspired by the Functional Mock-up Interface standard, also known as FMI (FMI Standard). MOSIM introduces Motion Model Units, also known as MMU, into the field of human simulation as a way of integrating simulation models from several different simulation environments. Motion Model Units are designed to unite arbitrary motion generation approaches and human simulations under a generic framework that would incorporate standardized component structures (Gaisbauer et al. 2018, 769). (MOSIM Project.)

Deeper considerations of human factors are considered can potentially yield significant improvements in productivity and safety (MOSIM Project). This type of digital human simulation will enhance the potential for some cases such as autonomous driving, product design, manufacturing, and customer experience. In product design it will be able to help with assessing comfort, visibility, reaching and grasping activities, multi-person interaction and strength assessments. This sort of software could also be used in different sectors such as the entertainment industry for such things as gaming, movies, and advertisement. Manufacturing is probably the most important here with the current challenges of assembly planning, especially in automotive assembly planning where there is a need for growing efficiency and increased flexibility due to the constantly increasing range of competitors. (Gaisbauer 2019.)

This thesis goes through the process of creation and implementation of a technical toolkit that shall serve in the software used for simulations. This toolkit shall contain different essential hand tools regularly used in the technical sector. In this case, the tools would be the most

basic and widely used, some of which you could find in almost every single mechanic shop or laboratory. These tools would also be able to come in different sizes and serve their respective purposes in the simulation software. They will be used for all the resulting work for the project be it car assembly simulation, truck assembly simulation or wood processing simulation. Success during this work would mean that this toolkit would be convenient to be used in the future for testing and realisation of this simulation software project and henceforth be used by actual people that will be using the software in the future. The resulting tool library could have a similar concept to that list-like set of tools as folders with different sizes in each of the folders.

The clear-cut objectives of this thesis are:

- Creating toolbox components first with the help of a CAD (Chai 2020) software.
- Having the different components tested and calibrated in a simulation software.
- Delivery of the toolbox components which will then be used for simulation purposes

The points above simply describe all necessary steps which will have to be completed before the objectives of the thesis work could be considered to have been met. The MOSIM project gets to focus on modular simulation (Simulistics) of natural human motions so that it can be implemented for company use. This type of simulation for company use would mean simulation of how humans would operate relative to the machine, and this kind of relation would mean the use of some of these simple day-to-day tools in the simulation environment. This is where and how the tools created during this work would come into place.

2 Toolbox and Design background

In many companies such as Daimler for example, which is a partner of the MOSIM project, there are a lot of different manual processes at the manufacturing stage of their cars. This means that there are a lot of manual human activities involved and that in turn means it is important nowadays to be able to simulate digital humans with a sophisticated approach. This makes companies more efficient in the sense that they can predict any sort of event happening during manufacturing before it is physically carried out. There are a lot of projects which have focused on distinct aspects related to human simulation, for example, deep learning which is used to simulate human walking and it works well for that (Song et al. 2021). There also are tools for simulating joining parts with simulated human hands which are physics-based and work great for that (Brubaker et al. 2009). Different approaches need to be applied and combined to allow a realistic simulation of complex human motions (Gaisbauer et al. 2021, 717-734). The barriers to this solution at this moment are for example lack of knowledge, where some tool makers do not have expertise in many different areas, and the latest technologies and documentation on those are not available as source code. Addressing these issues would mean creating a software tool that provides the required functionality that can be realized, ultimately enhancing planning on manual assembly processes. (Gaisbauer 2019.)

Nowadays, most softwares containing any aspect of simulation, be it movement or behaviour under certain conditions, also come along with a set of tools that assist a proper run of the simulations. In some cases, these tools could be something as basic as actions that could be undertaken by or towards the part in the simulation. In most other situations, tools could basically be tools, or rather, a model of tools in the software which could serve as props and help better understand a simulation situation. For example, SolidWorks which is a 3D modelling software that can also do stress simulation has a lot of actions that serve as tools, whereas, a software like Factory I/O which helps to simulate factory environments has 3D modelled tools and gadgets which can be used in the simulation environment (SolidWorks; Factor I/O).

A toolbox can be known as a container used to store, transport, and protect tools. They serve a variety of purposes, including professional and hobby work, and their contents depend on the craft of the owner. It is annoying to create a mess while working because of tools, which can range in size from tiny screws to big hammers. Every type of worker, from amateurs to professional workers, can benefit from a toolbox. As a practical matter, tools in a toolbox are quite useful in the course of daily work. Depending on your needs and the tools you possess, they can help you resolve different types of situations.

Figure 1. DeWalt Toolbox (DeWalt Toolboxes)

Figure 2. Keter Rolling Tool Chest (Amazon)

These types of toolboxes or tool chests, like the one shown in Figures 1 and 2, are usually much bigger than the regular ones for hobbyists containing many more tools and could either be stationary or movable.

An alternate definition of a toolbox as simply stated before is a type of window in some software that contains icons and buttons that are used within the program.

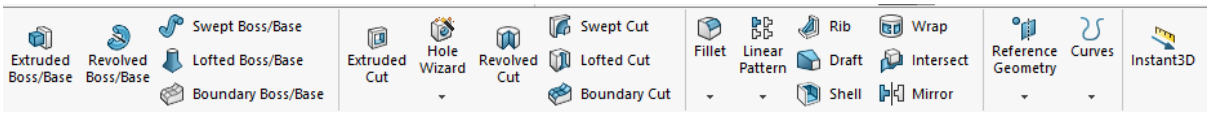


Figure 3. SolidWorks features.

Looking more into the Factory I/O application, it contains more than 20 scenes based on common industrial applications. By creating your training scenarios, you can use a library of industrial parts and extend Factory I/O. This library of industrial parts can be used to create a virtual factory including sensors, conveyors, elevators, stations, and many others. Smart editing tools in Factory I/O make building a 3D scene a natural and comfortable process. If you want to weigh items or control liquid levels, you can use a digital value or an analog value. Among other technologies, Factory I/O supports drivers for PLC (Polycase 2021), SoftPLC (A3 Resources), PLC simulators (MathWorks), Modbus (Schneider Electric 2021), and others. There are drivers of different technologies included in each edition. (Factor I/O.)

The process of tool creation in the context of this thesis would normally start with the tools being modelled. They need to be made to look realistic in the dimensions and looks. These types of models could be done with a computer-aided design program. One of such softwares is SolidWorks which works similarly to most CAD softwares available nowadays.

SolidWorks is a powerful CAD tool with which you can create almost different sorts of models and carry out diverse tasks such as basic stress analysis. There is no doubt that SolidWorks continues to be one of the most used and popular CAD programs across the globe. It has the power and ability to create almost any shape imaginable as it uses geometric parametric modelling methods to allow for creating organic forms. SolidWorks is also a software with so much ready material for learning all over the internet. (SolidWorks.)

The next step would be to get the modelled tools in a file format suitable for the Unity software as it is a partner in the MOSIM project, and is the software used for the project. This is where the final changes and adjustments will be made. The files extracted from SolidWorks cannot be in the correct format needed. At this stage of creation, the files need to be either in “.fbx” (FileInfo a) format or “.gltf” (FileInfo b) for them to be introduced in the Unity program for the next stage of the work. After going through the trial-and-error process, and general research, a proper mainstream modelling software that can export in the correct file type needed for this project and work was not immediately found. In addition, due to the already acquired skills present before the beginning of the thesis work, it was judged to be easier, and less time consuming as no actual brand-new programs had to be learned overall, to use Blender for the next step.

Open-source and free, Blender is used for designing animated films, visual effects, art, 3D printed models, motion graphics, interactive 3D applications, virtual reality, and computer games. Blender also offers many other features. (Blender.)

The CAD models are to be exported from SolidWorks as “.stl” (FileInfo c). These files are imported into Blender and exported as “.fbx” files which can work in the Unity software.

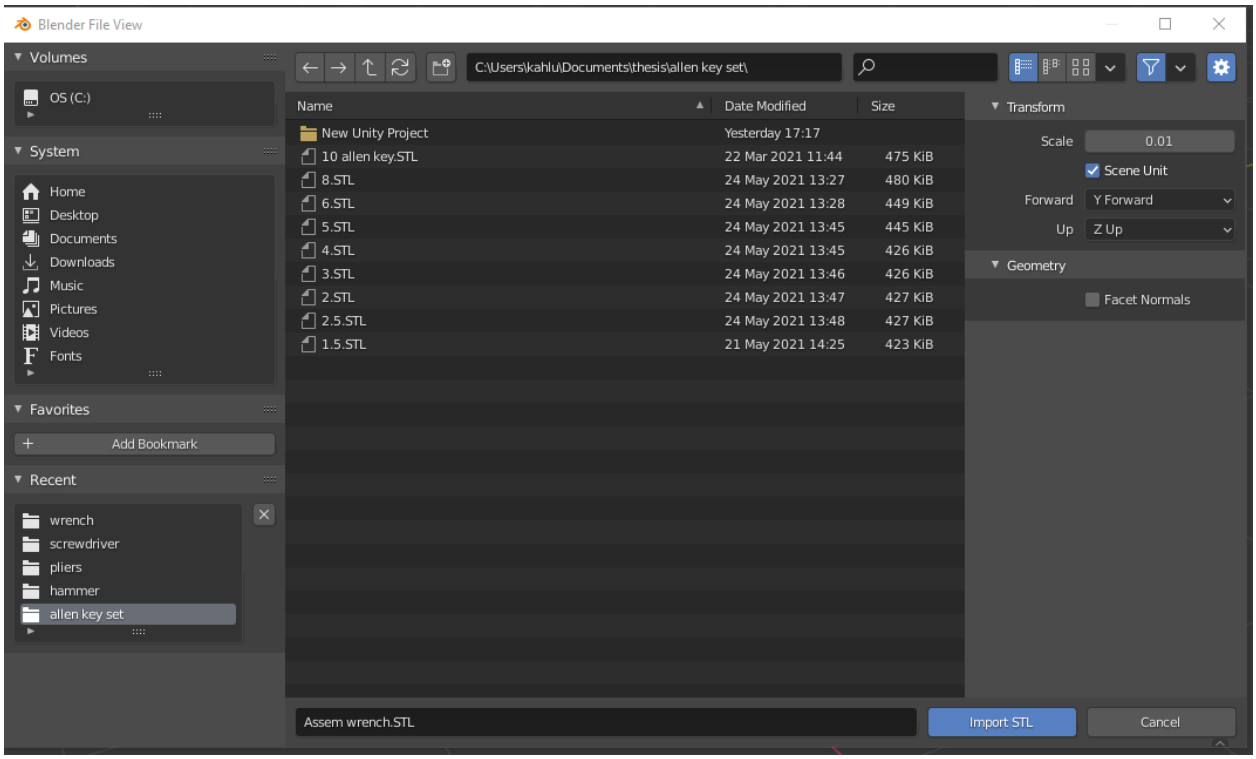


Figure 4. Blender file view

Furthermore, when exporting only the mesh is selected as object type and the correct scaling can be chosen as shown in Figure 5 while unchecking “Apply Unit”.

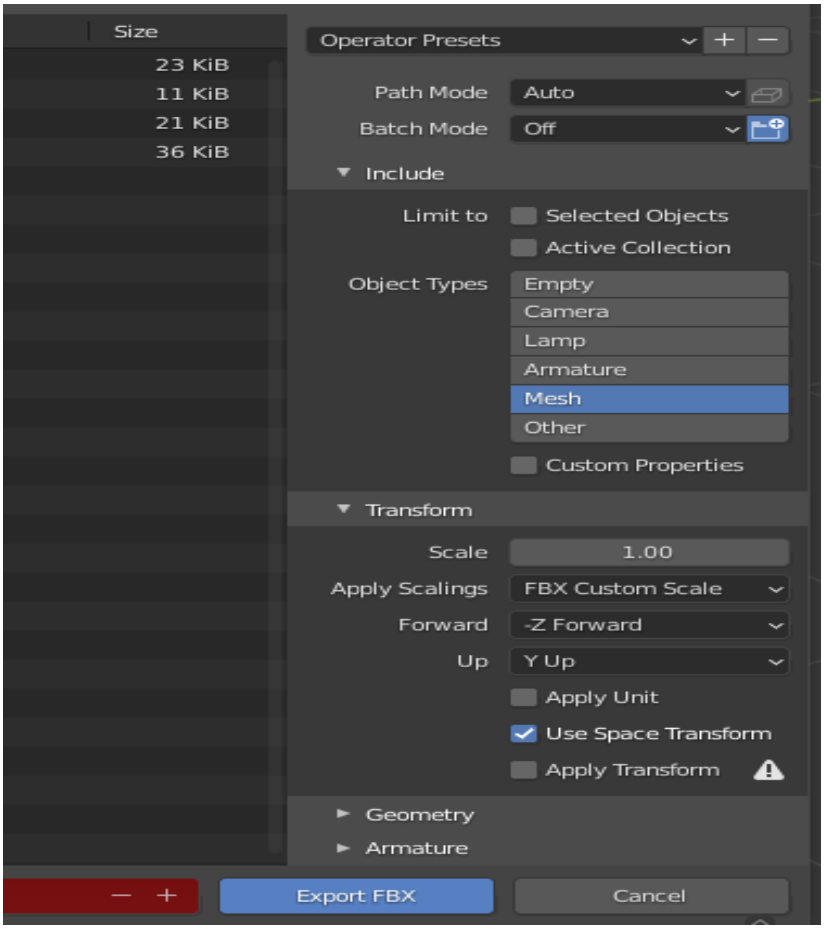


Figure 5. Blender export settings

The tools are at this point finally able to be inserted into Unity where the final section of the work was carried out. Unity is a cross-platform game engine developed by Unity Technologies. The engine has been made to support a variety of desktop, mobile, console and virtual reality platforms and is particularly popular for mobile game development (Unity). It is cited to be easy to use for beginner developers and is popular for independent or personal game development (Unity). The engine can be used to create three-dimensional and two-dimensional games, as well as interactive simulations and other experiences (Unity). The engine has been adopted by industries outside video gaming, such as film, automotive, architecture, and construction (Unity). This sort of simulation could serve strong interest for the project and its outcomes such as early manufacturing process verifications, virtual simulations for assembly and planning, and other things such as pedestrian simulations for autonomous driving.

The parts are arranged into different folders in the assets folder and all these folders for the toolbox that was aimed at being built at the beginning of the work. Some of the parts could have C# code added as components so they can interact as active tools. This is further

discussed in each part's section that needed some code. This workflow is indeed necessary because every single step and software used is important. Any other chosen path will either take an enormous amount of time for the same outcome or just not be able to be done at all.

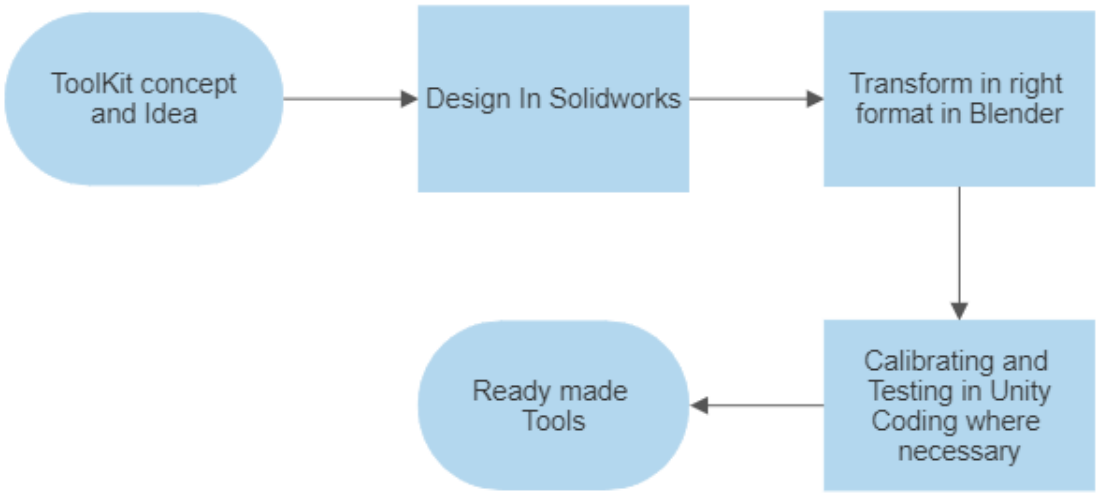


Figure 6. Work Flowchart

This type of work is needed for a project as such because as it can be seen with many other 3D-related softwares such as SolidWorks or Factory I/O, there are always already available tools which can be used for whatever creation one is trying to do with said software. In this case, the MOSIM project is trying to have as outcome some sort of framework which would contain a standard for the simulation of anything human-related.

3 Case study

At the beginning of this thesis, the objectives of this thesis were stated which were to create the tools with a CAD software and have them in a form which is suitable for the Unity software, which is a supporting partner, before they could be put in use. All these steps are important because every single one serves a purpose.

In the previous section of this thesis, the basics of the different methods used were given. In this section, a more detailed account of how these methods are used will be given. A case study will be conducted which will serve as an example to show each process during the creation of the tools. The tool considered here is the screwdriver.

This screwdriver assembly is made up of three different components or parts: the handle, the shaft, and the tip. All these parts are first all designed and made in SolidWorks separately and then assembled. The holder is made by performing an extrusion on a two-dimensional circle drawn in a plane in SolidWorks (Dassault Systemès a). In addition, the hole which holds the shaft is made and tiny dots are added to the surface of the handle to add better grip. This of course does not change much in simulation but is made for the part to look more realistic. The shaft is also created with an extrusion from a circle. A hole which will hold the bits is created. The circumference on the upper end is made so that it can also fit with the hole on the handle. The bits were then made with four different types made available. An extrusion is created, and cuts are made on each part accordingly. In two types out of the four, the bit end cuts were scaled from one original made. In the remaining two, the cuts were all made separately for each bit.

The assembly is created this way so that it is possible to scale them differently and properly in the Unity software. This also helps because it facilitates the use of different tips in the assembly. Several different tip types of different sizes were modelled.

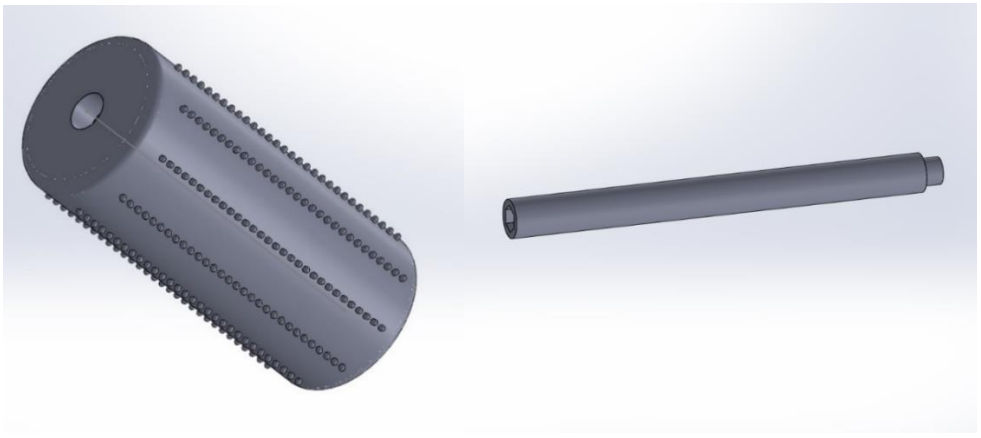


Figure 7. Screwdriver handle and shaft

First, there is the hexagon-shaped bit which carries the letter “H” which shows it is a hexagonal shaped bit. The numbers after H represent the side-to-side length in millimetres. They come in sizes H0.7, H0.9, H1.3, H1.5, H.2.0, H2.5, H3.0, and H4.0.

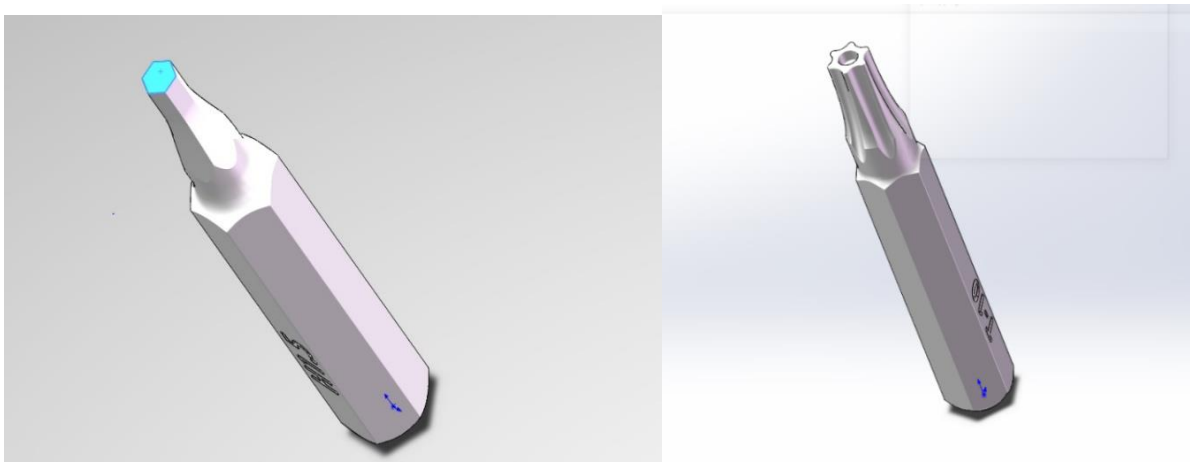


Figure 8. H1.5 Hexagon shaped bit and T10 Torx bit

In addition, there is the Torx type bit which carries the letter “T”. They come in sizes T4, T5, T6, T7, T8, T9, T10, T15, and T20. Furthermore, there is the Philips type bit which all carry the letters “PH". They come in different types which are PH0, PH00, and PH000.

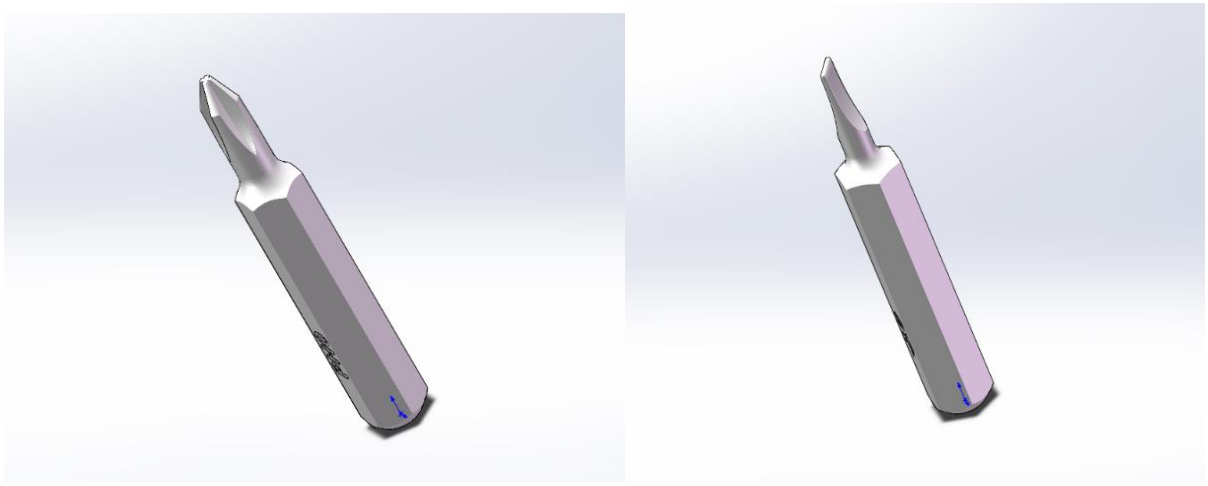


Figure 9. PH0 Philips bit and slotted type bit

Lastly, there is the slotted type bit which all carry the letter “S". They come in sizes S1.0, S1.5, S2.0, S2.5, S3.0, S3.5, and S4.0.

The Torx and hexagon bits are created so that they are all scaled from one original model. The Philips and slotted-type bits are all made individually for more accuracy as scaling does not work well for all the sizes and types. Every detail in the creation must be taken care of so that the parts for the tools turn out to be of great quality. All these bits are made available to the final user and can be interchanged on demand. The files are then also exported as “.gltf”

files because this would work with the project framework and make it convenient for the recipient of the tools in the project to work with it. After having all the parts of the tools put in the right format in Blender, they are then inserted in Unity.

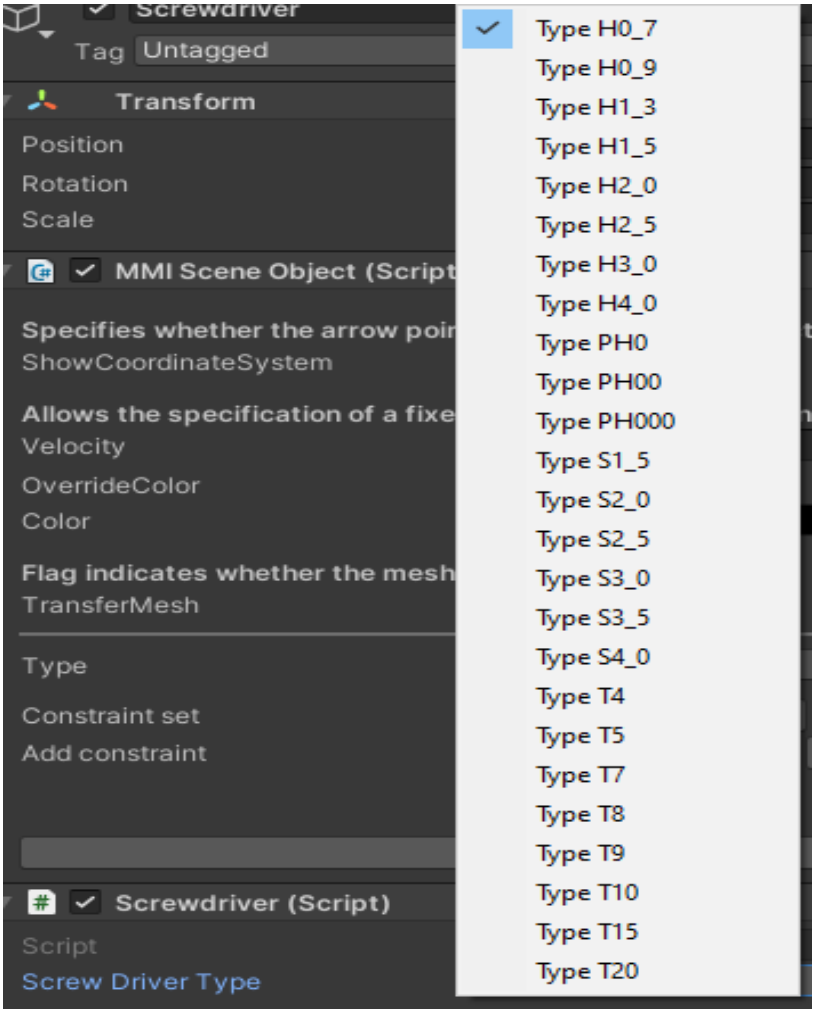
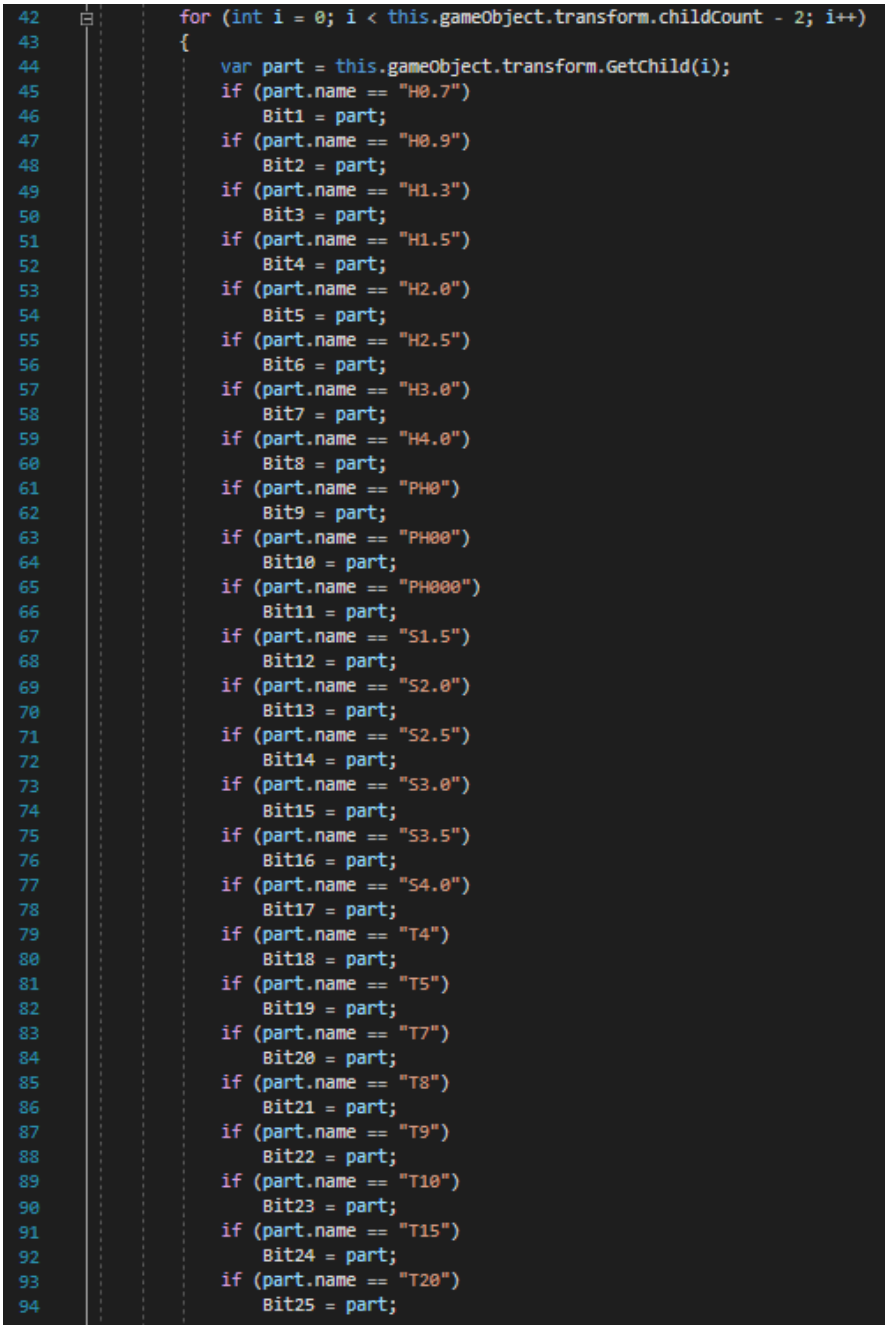


Figure 10. List of Screwdriver tip types

The list in Figure 10 shows the list of different screwdriver tip types with letters H, PH, S, and T meaning Hexagonal, Philips, Slotted and Torx respectively. It is an extensive list as there are many uses for a particular screwdriver handle. In Unity, all these tips cannot be made to appear all at once on the scene view of the program. The user should be able to choose which tip is needed at a particular time and this is done through programming the tool using C# (Microsoft 2022).



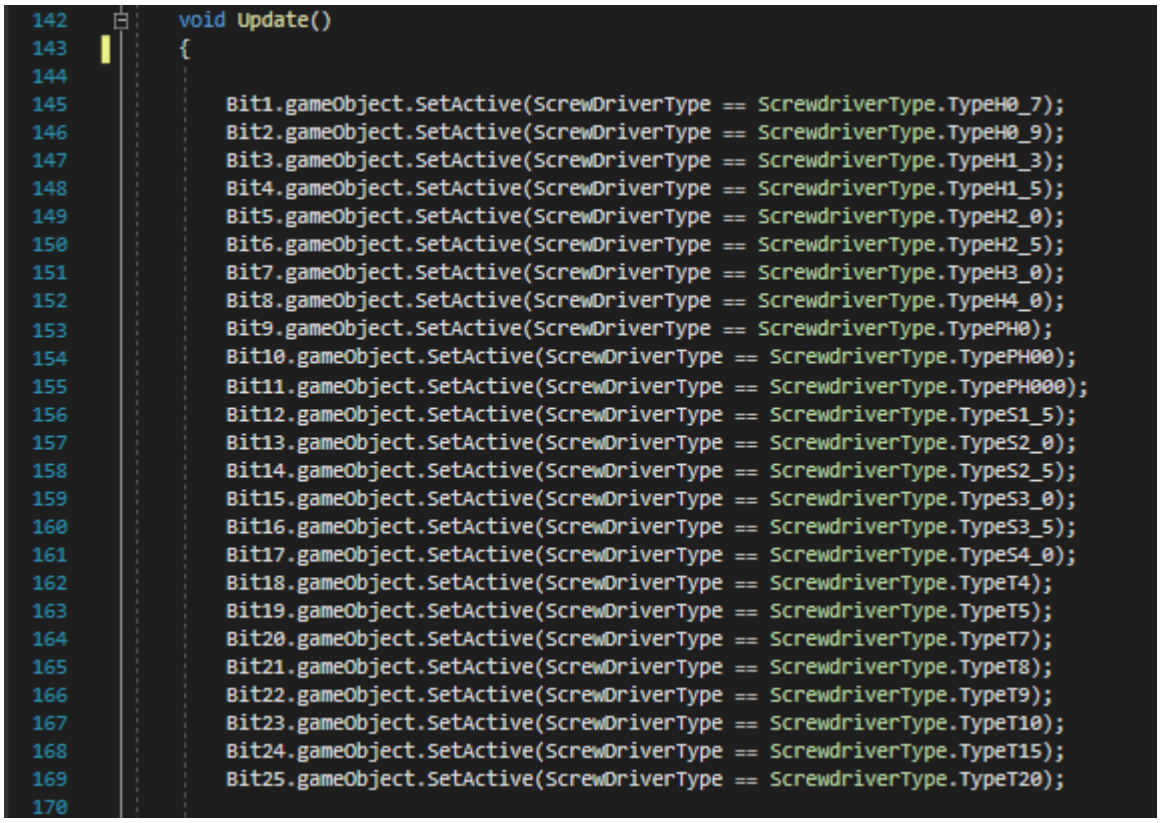
```
for (int i = 0; i < this.gameObject.transform.childCount - 2; i++)
{
    var part = this.gameObject.transform.GetChild(i);
    if (part.name == "H0.7")
        Bit1 = part;
    if (part.name == "H0.9")
        Bit2 = part;
    if (part.name == "H1.3")
        Bit3 = part;
    if (part.name == "H1.5")
        Bit4 = part;
    if (part.name == "H2.0")
        Bit5 = part;
    if (part.name == "H2.5")
        Bit6 = part;
    if (part.name == "H3.0")
        Bit7 = part;
    if (part.name == "H4.0")
        Bit8 = part;
    if (part.name == "PH0")
        Bit9 = part;
    if (part.name == "PH00")
        Bit10 = part;
    if (part.name == "PH000")
        Bit11 = part;
    if (part.name == "S1.5")
        Bit12 = part;
    if (part.name == "S2.0")
        Bit13 = part;
    if (part.name == "S2.5")
        Bit14 = part;
    if (part.name == "S3.0")
        Bit15 = part;
    if (part.name == "S3.5")
        Bit16 = part;
    if (part.name == "S4.0")
        Bit17 = part;
    if (part.name == "T4")
        Bit18 = part;
    if (part.name == "T5")
        Bit19 = part;
    if (part.name == "T7")
        Bit20 = part;
    if (part.name == "T8")
        Bit21 = part;
    if (part.name == "T9")
        Bit22 = part;
    if (part.name == "T10")
        Bit23 = part;
    if (part.name == "T15")
        Bit24 = part;
    if (part.name == "T20")
        Bit25 = part;
```

Figure 11. For-statement in Screwdriver code

The for-statement in Figure 11 is the main asset in the code because it is this that connects the parts in Unity with objects created in the code. These objects will later be renamed in the code when the drop-down list is created.

The code is made using Microsoft Visual Studio which was connected to the Unity program. Variables are created in the code. Those variables are then linked to the actual parts in Unity using a for-statement and the “this.gameObject.transform.GetChild(i)” command which will refer to the actual parts. The variables are then coded with a list so that the drop-down list is

visible as shown in Figure 10. If the user clicks on a particular tip on the list, that tip appears at the end of the screwdriver handle in the scene view of Unity.



```
142     void Update()
143     {
144
145         Bit1.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypeH0_7);
146         Bit2.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypeH0_9);
147         Bit3.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypeH1_3);
148         Bit4.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypeH1_5);
149         Bit5.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypeH2_0);
150         Bit6.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypeH2_5);
151         Bit7.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypeH3_0);
152         Bit8.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypeH4_0);
153         Bit9.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypePH0);
154         Bit10.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypePH00);
155         Bit11.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypePH000);
156         Bit12.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypeS1_5);
157         Bit13.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypeS2_0);
158         Bit14.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypeS2_5);
159         Bit15.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypeS3_0);
160         Bit16.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypeS3_5);
161         Bit17.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypeS4_0);
162         Bit18.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypeT4);
163         Bit19.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypeT5);
164         Bit20.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypeT7);
165         Bit21.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypeT8);
166         Bit22.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypeT9);
167         Bit23.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypeT10);
168         Bit24.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypeT15);
169         Bit25.gameObject.SetActive(ScrewDriverType == ScrewdriverType.TypeT20);
170
```

Figure 12. Main function in Screwdriver code

4 Results and discussion

4.1 Allen key set

Allen keys or hex keys are common tools for driving bolts and screws with hexagonal-shaped sockets. It is made of one piece of hexagonal-shaped steel rod, with ends that fit perfectly into the socket of a necessary screw, bent in a square angle shape with one arm usually longer than the other. The tip of the short arm receives a larger torque when you hold the tool by the long arm.

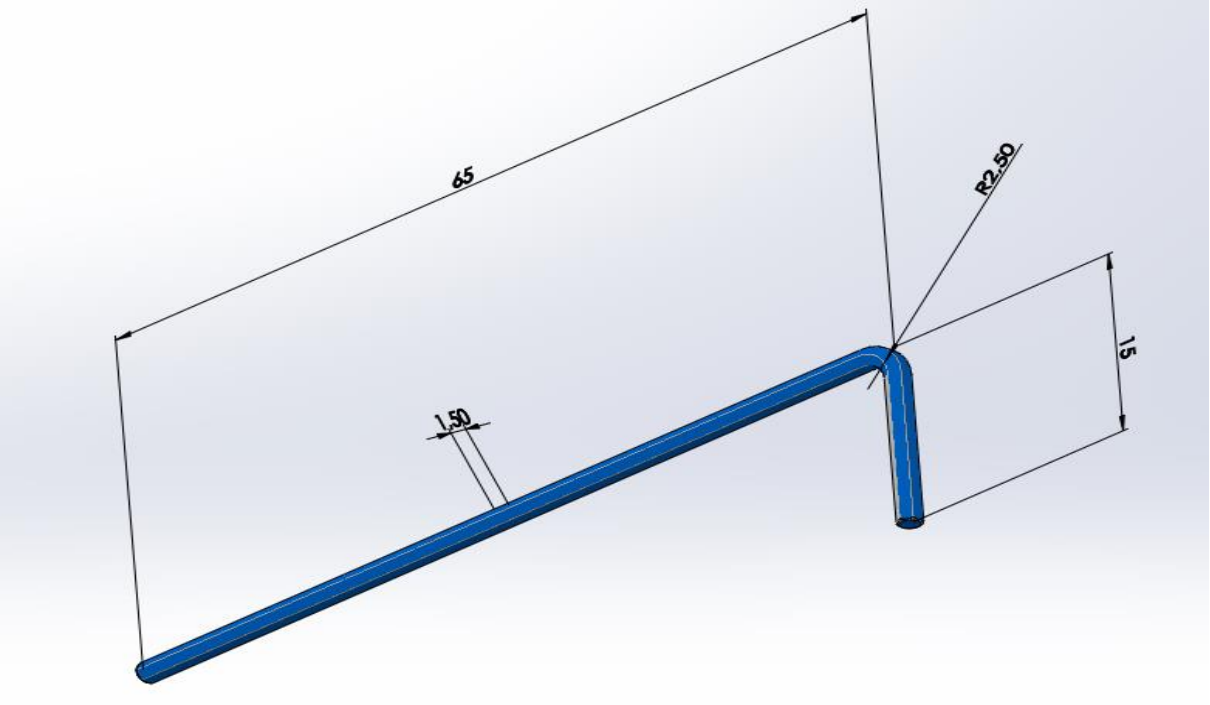


Figure 13. Model of Allen key

The model was created using SolidWorks. The created set of Allen keys has thicknesses ranging from 1.5mm right up to 10mm. They were all made separately using two-dimensional drawings and the sweep command (Dasssaults Systemès b). The reason for creating different Allen keys instead of just scaling the smallest one to give bigger sizes was because in real life the ratio between the length of the handle and the thickness of the Allen key is not the same as you increase in thickness. This means that in the case where an Allen key is created and introduced in Unity and the scale is just changed, the initial part inserted would look normal in a simulation setting but the other scaled parts will not look realistic. The sizes were taken from metric Allen key sizes and dimensions for ISO 2936:2014 which specifies the dimensions, the method of test, the designation, and the marking of hexagon socket screw keys (International Organization for Standardization 2014).

The set of Allen keys was also equipped with a C# code in Unity. This code is present so that all keys are not present in the simulation all at once. A drop-down menu is created because of the code. This menu gives one the chance to choose which size of Allen key is needed at any moment. The part then appears in the scene and can be used henceforth.

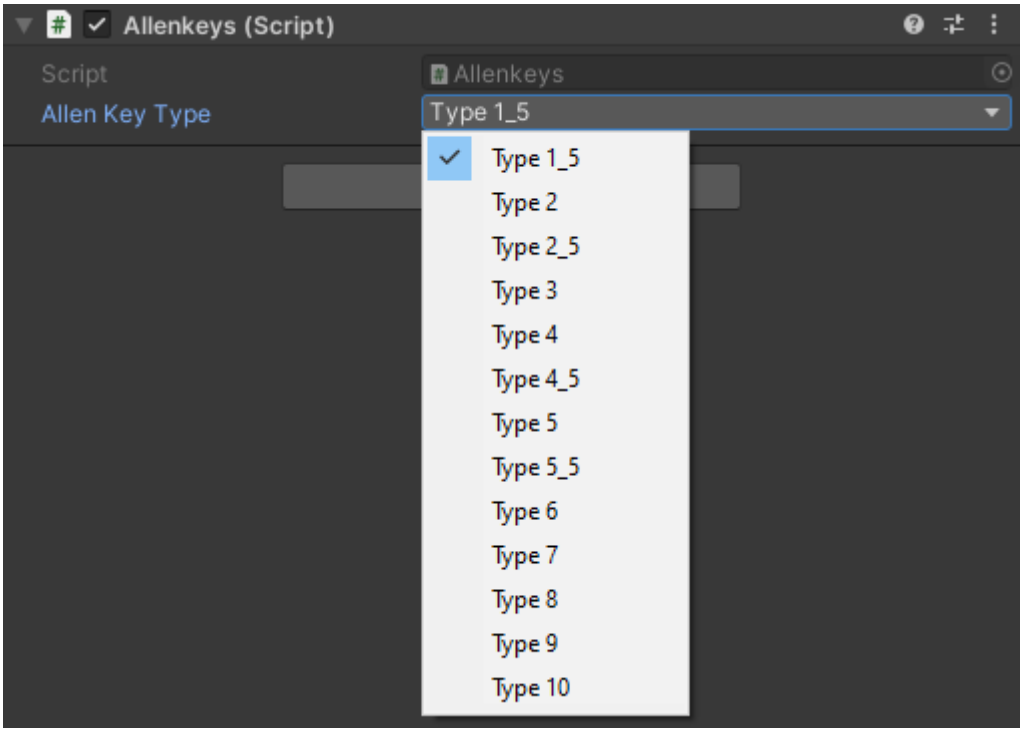


Figure 14. Drop Down menu for Allen keys

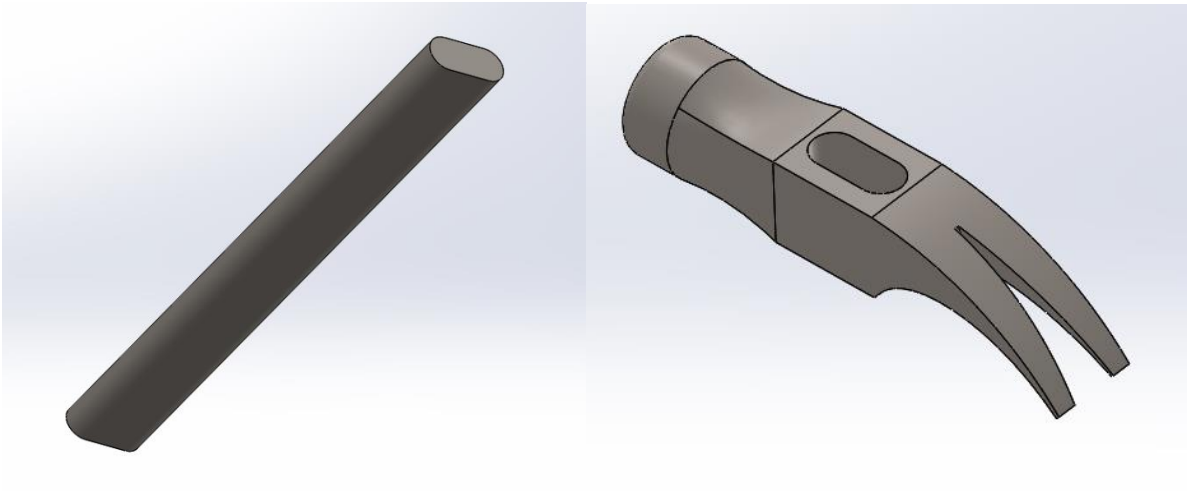
The drop-down menu in Figure 14 shows the list from which the different sizes for the tool can be chosen. The number after the word “Type” represents the size of the Allen key in millimetres. The underscore “_” represents a decimal point. This is also true for the remaining parts.

4.2 Hammer

Most often a hand tool, a hammer is made up of a heavy head with a handle, which can be swung to strike an object. It can be used to punch nails into wood, hitting metal or breaking rock. The wide range of applications of hammers includes driving, shaping, breaking, and delivering all types of impacts.

Figure 15. Hammer Assembly

The hammer handle was made with a simple extrusion. The hammerhead was made with a series of extrusions and cuts. The hammer was made of an assembly of the handle and the hammerhead so that when they are inserted into the Unity program, the properties of the handle and the hammer in Unity, that is, the different parameters such as the length can be separately changed without having any disproportionate looking tool.



Figures 16. Models of hammer handle and head

4.3 Pliers

A plier is a hand tool used for firmly holding objects. A wide variety of materials can also be bent and compressed with them. A pair of pliers usually consists of two metal levers joined closer to an end, which results in a placement made up of short jaws and long handles on opposite sides of the joint. Mechanical advantage is created by this arrangement that amplifies the grip of the hand and magnifies it for precise focus on an object. One can also use the jaws for objects that are too small or unwieldy for your fingers.

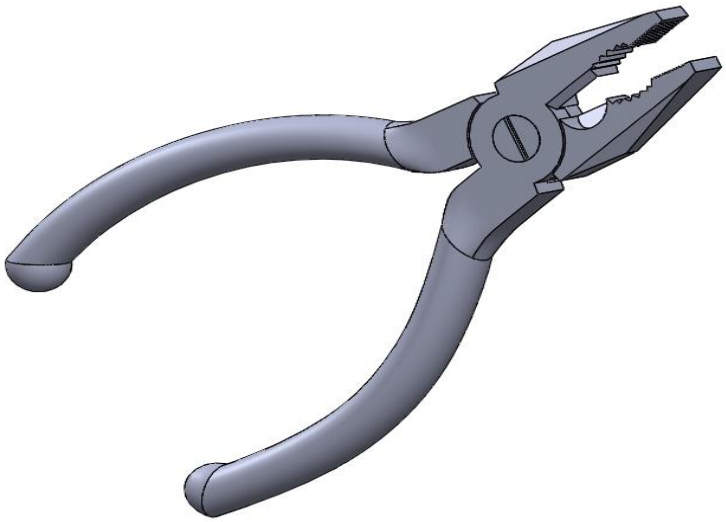


Figure 17. Pliers Assembly

The plier halves are identical. The part was made with a series of extrusions and cuts for the upper part, and a sweep for the handle. The lock was made from an extrusion and a mirrored cut. The plier assembly is made up of 2 plier halves and a lockset in place to lock both parts up. Here below are the various parts of the assembly.

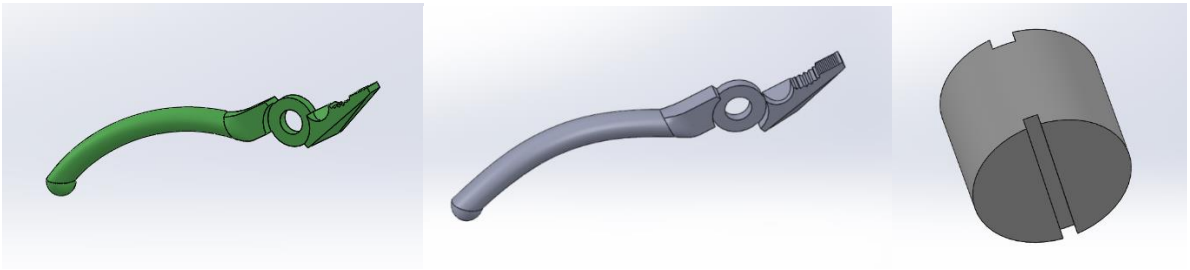


Figure 18. Plier halves and lock

4.4 Screwdriver

A screwdriver is a simple tool that can be used for inserting and removing screws. A normal screwdriver has a handle and a shaft. At the end of the shaft, there is a tip that is inserted in the screw before it can be inserted or removed. Screwdrivers differ from each other depending on which tip it possesses. Correct use would mean that the tip of the screwdriver works with the head of a screw of the same size and type. There are so many kinds of screwdriver tips. The two most common are the blade-type and Phillips generically called crosshead.

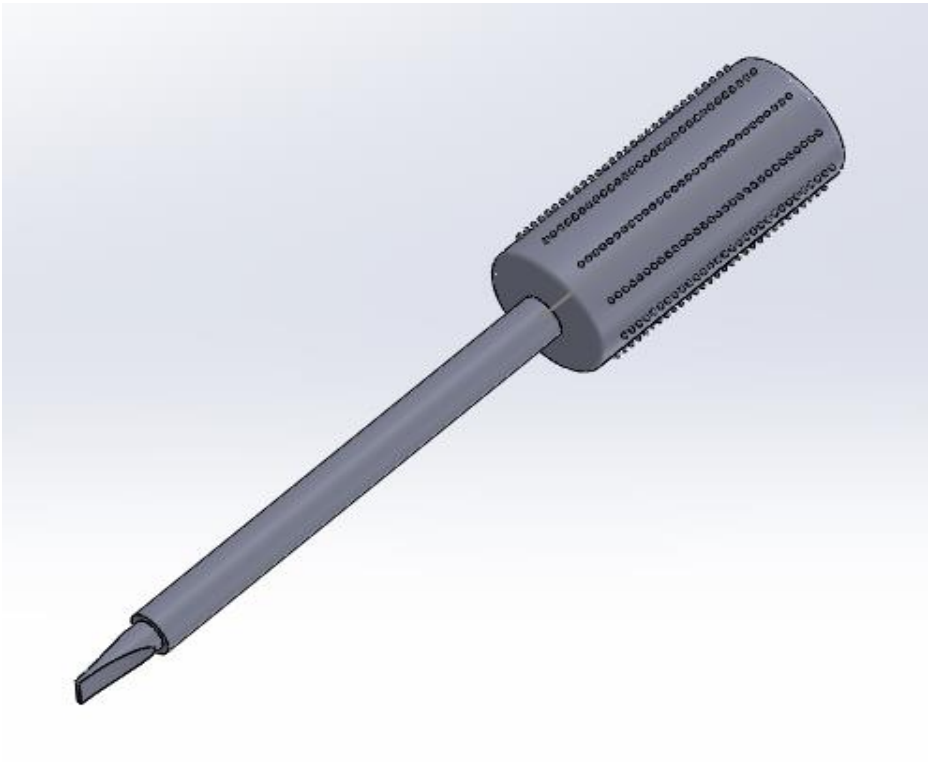


Figure 19. Screwdriver assembly

There are 25 different types of bits available for the screwdriver assembly. Details relating to the screwdriver are studied in the case study.

4.5 Wrench

Spanners and wrenches are used to turn and stop objects from rotating, commonly rotary fasteners, and bolts.

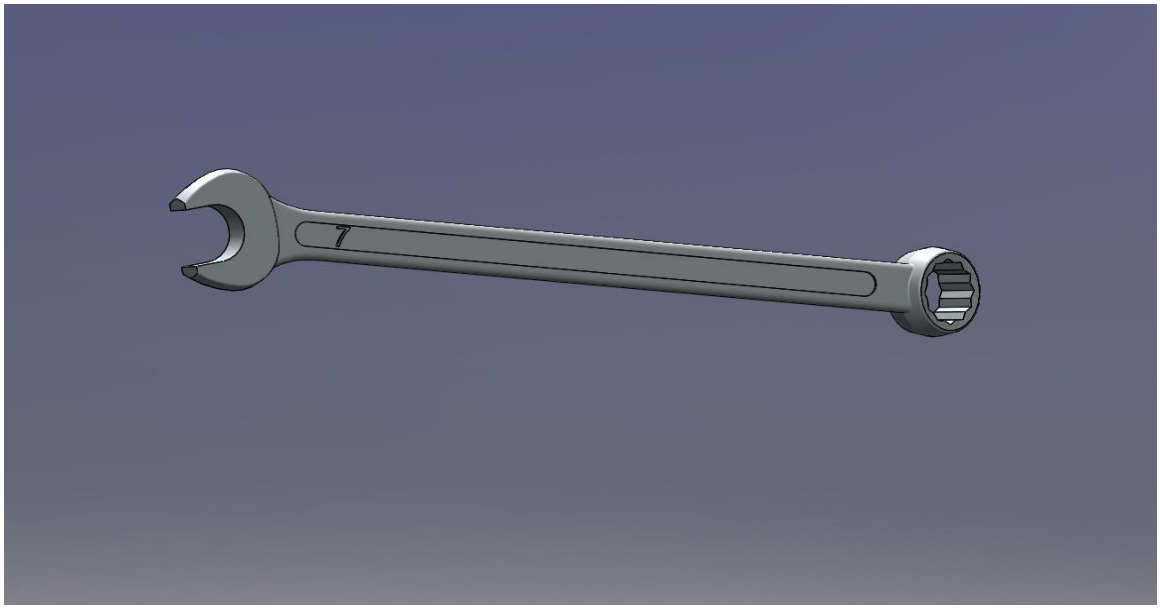


Figure 20. Size 7 Wrench

The wrench in Figure 20 is of size 7mm. The size is indicated on the wrench as is the case for every single wrench. The initial wrench shown in Figure 20 was made from a series of extrusions and cuts. The remaining wrenches were scaled from this one with multiple parameters having the possibility to change. Some of these are the length of the grip area and the size of the cuts made. In addition, each wrench was then further checked, and corrections were made when necessary so that every tool in the set looked proper and normal. A range of sizes was made available ranging from 5.5mm to 19mm, all of which were compliant with ISO 691:2005 which specifies tolerances for wrench and socket openings for screws and nuts with metric width across flat dimensions (International Organization for Standardization 2005).

Figure 21. Wrench List

C# code was also needed for the part. A connection was made with variables in the code to the created part added to Unity such as in the screwdriver shown in the case study. This made

possible the creation of a list which appears in Unity. From this list, one can choose which size of the wrench is suitable at a particular time and that tool is made available in the scene view.

4.6 Level

Levels are tools commonly used with a staff or levelling staff and are used to verify points in a horizontal plane. In construction, it is used to measure height differences and to measure set heights of known places or things.

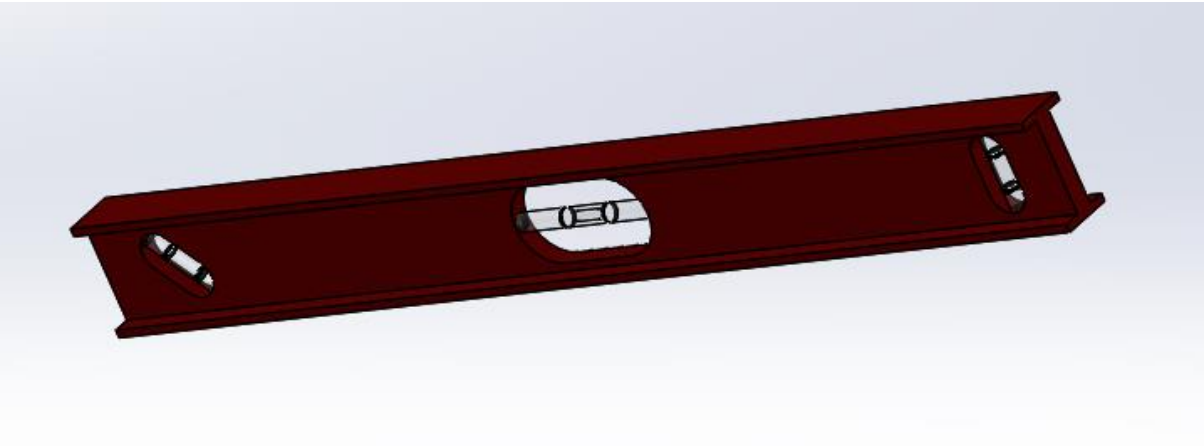


Figure 22. Level Assembly

The level assembly consists of various parts which are: the main body and the liquid-filled glass pockets which would contain a bubble. There would be, as shown in Figure 22, 3 different glass pockets with bubbles, one which would be horizontal, one vertical and one with a 45-degree angle. All the different parts were made was a series of extrusions and cuts. This assembly in theory would permit the tool to be used for different level purposes, but in the project, this tool would mostly be present as a prop during simulation to observe an environment which would include a level.

4.7 Drill

The purpose of a drill is to make a hole or drive fasteners into a surface. In addition to this, the bit is secured to the chuck by a bit, either a drill bit or a driver bit, depending on the application. There are many different sizes, speeds, and powers of drills. These devices can be corded and powered by electricity, or by batteries.

Figure 23. Drill Assembly

It took a few sub-assemblies to fully create the drill model. The different parts in the model were easy to make as the drill interior did not need extensive detail. The drill comes with different drill bits of varying sizes and could also be used for screwing purposes with different types of screwdriver bits. All the bits were scaled from the first one created

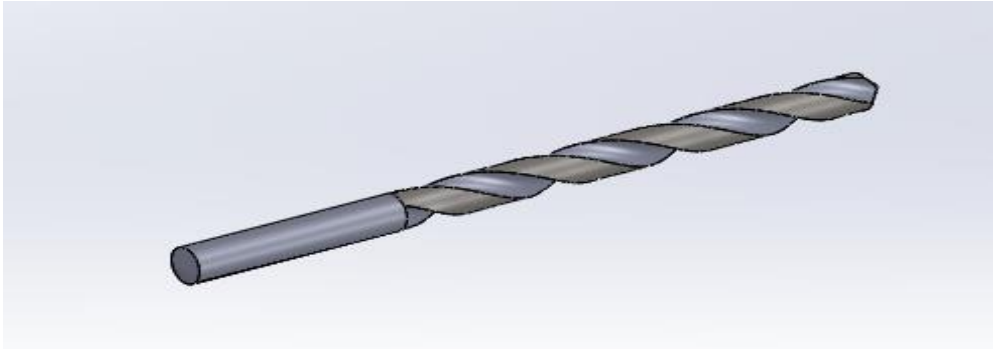


Figure 24. Drill bit

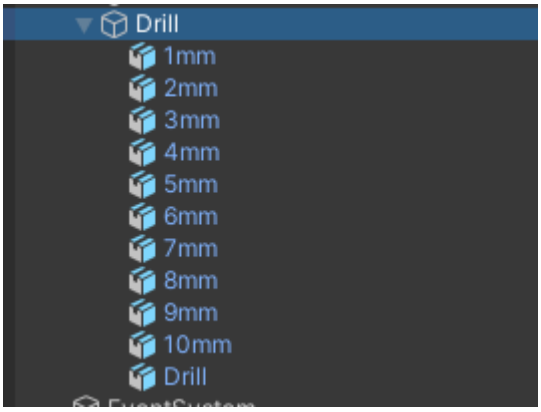


Figure 25. Drill bit sizes

All the bits were scaled from one initial created. Necessary changes such as making additional cuts or changing lengths were made so that the tools were compliant with the necessary standard. The different drill bit sizes are shown in the figure above, all of which were compliant with ISO 9714:2012 which specifies materials and mechanical properties, dimensions and marking requirements for drill bits (International Organization for Standardization 2012). The assembly would serve as a drill representation in simulation, that is, it would not have to necessarily be drilling anything.

4.8 Ladder

Ladders are a straight or inclined array of steps. There are several types: inflexible ladders which can be self-helping and can be leaned towards a vertical surface which could be a wall, and rollable ladders, which include the ones made from rope or aluminium, that can be hung from the top.

Figure 26. Workshop Ladder

The ladder assembly is made of different bars. The bars were simple to make in SolidWorks. This ladder has 7 steps with the top one with more leg space for more stability for the person at the top. This tool can be used in simulation with a simulated human climbing on it and doing some work. There is a possibility to increase the length of the sidebars and the number of steps.

4.9 Scissors

Scissors are shearing tools that are operated with the hand. When the handles are closed, the sharpened edges of the metallic blades pass in opposition to each other to cut whatever is in between them. Scissors can be used to cut many different thin things from paper to metal, cloth, rope, and wire.

Figure 27. Scissors

The scissors assembly is made of two scissor blades and handles with a pin to hold in the middle. The two blades can be rotated about the pin in simulation.

4.10 Knife

There are many types of knives, but the simplest is a metal blade tool attached to a holder or handle. Blade patterns and styles vary by maker and country of origin for modern knives, with fixed blades most common and folding ones quite common.

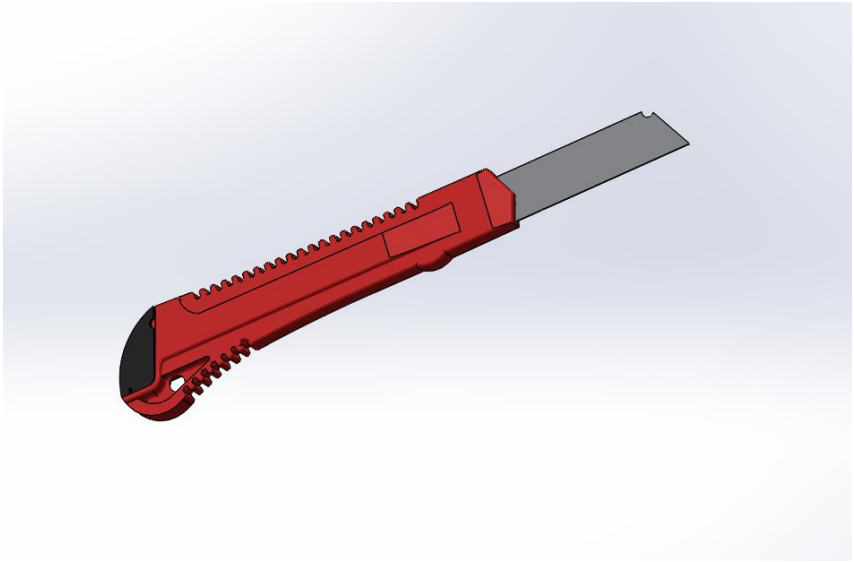


Figure 28. Utility knife

This knife assembly is made up of the protector or pocket, and the knife blade. The knife pocket is made so that it retracts if it is slid in that way it does not harm anybody when it is not in use. It also works this way in simulation.

4.11 Vernier calliper

Vernier callipers are measuring tools used to determine the length of an object. Callipers are available with both a ruled scale and a dial or digital display for reading out measurements. A calliper can be as simple as a compass with no scale, but points facing either inwards or outwards. Using another measuring tool, such as a ruler, you can measure between the tips of the calliper to find out the dimension. Many different fields use it, including mechanical engineering, metalworking, forestry, woodworking, science, and medicine.

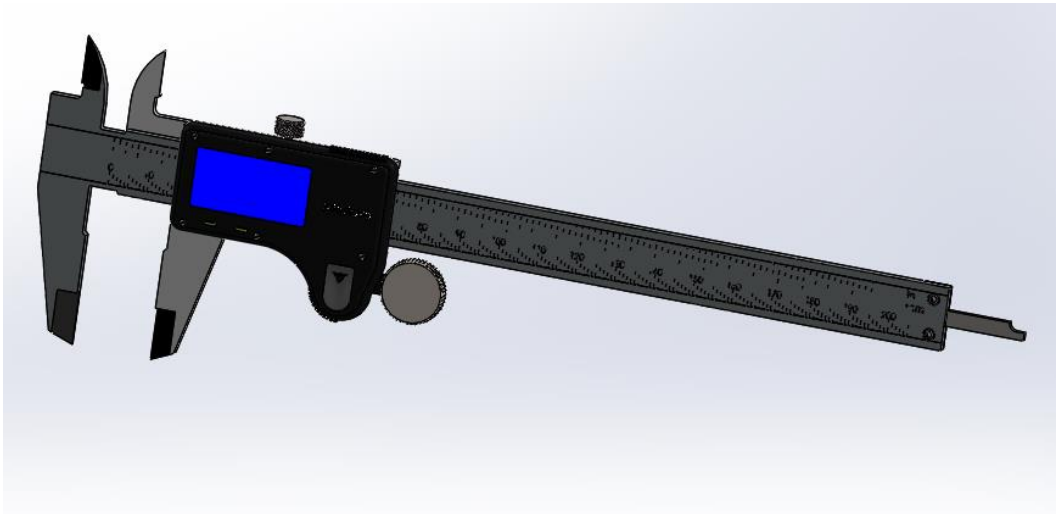


Figure 29. Vernier calliper assembly

The assembly is made up of smaller parts like the square ruler, the display and the measuring jaws. Vernier callipers are particularly useful tools in almost any laboratory to take quick small accurate measurements. The vernier calliper is assembled so that it can slide on the ruler and the measuring length can be changed.

4.12 Measuring tape

A measuring tape is a tool equipped with a calibrated rule tape used to determine the length of objects or measure a required distance. Spring return and long tape measures are the most commonly available. Spring return measures are even more common because they can fit in small spaces and can measure lengths up to 5 metres for some.

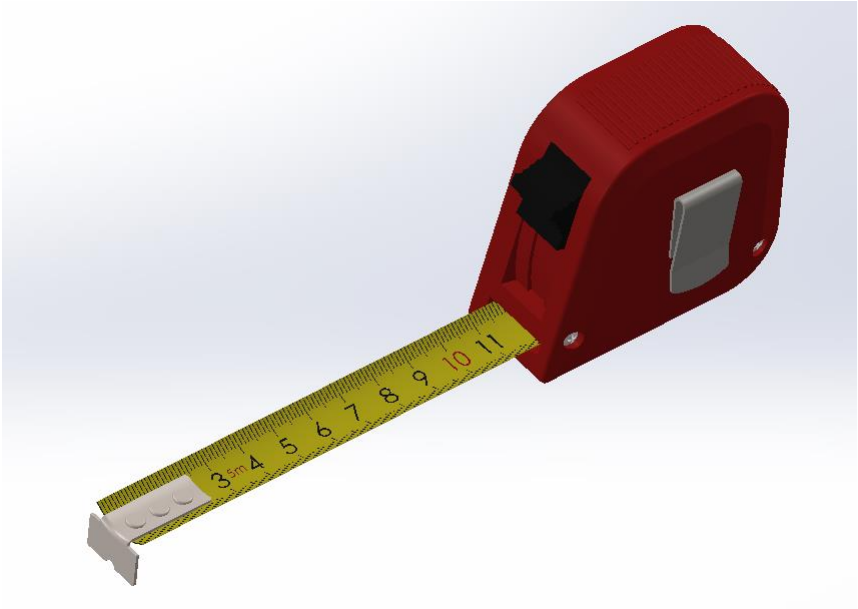


Figure 30. Measuring tape

The measuring tape in Figure 30 is a basic design of a measuring tape that can be used in the framework for display and representation purposes for a better understanding of ongoing activities. This means actual measurement cannot be done with the tool but instead a simulation of a human holding and possibly using it around can be made. The length of the tape could be changed even if the actual reading is not done on the tape.

4.13 Discussion

These results portray the different tools that were considered for creation before the start of the design work and were then realised after many hours of design work. There is a lot of detail which went into all the tools to make them as realistic as possible. This does not come

easy as there is a lot of work done beforehand, videos watched, and skills acquired before these tools could come to completion.

Modelling the tools was the section that was the most time-consuming. Even though a lot of SolidWorks experience had already been gained before the start of the thesis work, some of the tools had shapes which were not easy to make. Watching multiple videos and understanding quicker modelling paths could have made things quicker. Overall, the only thing that could have made it quicker was having more design knowledge at the beginning. This means in the case of another project like this coming up again, the design phase would be done much quicker.

Once in Unity, all the tools that needed to be assembled, like the hammer, knife, pliers, scissors, calliper, knife, and tape measure, were assembled. In the case of the screwdriver for example, the assembly was made such that both the handle and the shaft were constant, and the bits could be changed from the list available. This was the same case for the drill, where the drill body is constant in simulation space, but the drill bits could be changed with the list available. As for the Allen keys and the wrenches, the different sizes are all inserted in the simulation and the needed tool size can be chosen from the available lists.

The lists mentioned above were able to be created using C# coding that was inserted in the different tools as a script. This script once added and the simulation refreshed, makes the lists and other parameters such as max length available to the tools. Having already acquired basic programming knowledge proved to be essential as it made the work go quicker as further extensive research was not required when adding code to the tools which needed lines of code. The coding was also quite an enlightening experience as it revealed how lines of code could be linked with designed tools and work together perfectly.

All these tools were modelled and inserted into the Unity software both as “.fbx” files and “.gltf” files and work properly The tools then handed off were in the “.gltf” format because they were requested as so by the project, as this is the format used by other project sections, with their respective code for all the tools that needed additional coding. The lists created worked well and once a user chooses a particular size or bit, it immediately appears in the scene and the other ones are not visible. This would mean, in the screwdriver for example, the bit at the end of the tool will always be the one chosen. This can be changed by inactivating the code, which would in turn make all the tool sizes or bits appear. All of this means despite it being an incredibly challenging piece of work altogether, the tools created all worked as required.

5 Summary

At the beginning of this thesis, the MOSIM project was introduced as an international project consisting of more than 20 partners with the target of creating a standard for combining digital human simulation approaches. This project aims to generate an open standard for exchanging digital human simulation approaches. MOSIM transfers the idea of co-simulating models from different simulation environments to the field of human simulation by means of introducing the Motion Model Units.

The objectives of this thesis were said to be the creation of a tool kit containing common engineering hand tools for the simulation platform which is to be created by the MOSIM project. These tools were to be created with a CAD software. These were then to be inserted into the Unity software and tested.

The results for this section of this project are a set of tools that were created as CAD models using SolidWorks, modified in both Blender and Unity. These tools will be able to be used in the MOSIM framework. They will be vital in the MOSIM framework for human interactive simulation purposes. This work was completed after countless hours aimed at the design and overall creation of these tools

Reading through this work, one could feel like the goal of the project was achieved with regular day-to-day tools created and assimilated for use in the project. Doing this work, of course, came with its difficulties such as the creation of realistic-looking tools which could be used in the framework, and the use of new and intriguing softwares such as Unity and Blender. These obstacles or rather just minor difficulties did not hold back the work.

These tools are now being utilised by project members and they will continue to be vital in the project until its completion. Working on this section of the project turned out to be a wonderful learning experience. This was crucial to gaining more insight into real-life problems the technology sector faces nowadays, and real-life solutions to these problems.

6 List of references

A3 Resources. Understanding Soft PLC Features and Requirements. Accessed 02 May 2022. Available at

https://www.automate.org/tech-papers/understanding-soft-plc-features-and-requirements#:~:text=Soft%20PLC%20is%20a%20software,data%20handling%2C%20and%20computational%20capabilities.

Amazon. Keter Rolling Tool Chest. Accessed 03 January 2021. Available at

https://www.amazon.com/Keter-240762-Drawer-Modular-Organizer/dp/B07GCK635W

Blender. About Blender. Accessed 14 February 2022. Available at

https://www.blender.org/about/#:~:text=The%20Software,video%20editing%20and%20game%20creation

Brubaker, M & Sigal, L & Fleet, D. 2009. Physics-based human motion modeling for people tracking: A short tutorial. Physics-Based Models of Human Motion.

Chai, W. 2020. CAD (computer-aided design). TechTarget. Accessed 02 May 2022. Available at

https://www.techtarget.com/whatis/definition/CAD-computer-aided-design

Dassault Systemes a. Creating 3D Solids and 3D Surfaces by Extruding 2D entities. Accessed 02 May 2022. Available at

https://help.solidworks.com/2020/english/DraftSight/acis/acis_command_extrude.htm

Dassault Systemes b. Creating 3D Solids by Sweeping 2D entities Along a Path. Accessed 02 May 2022. Available at

https://help.solidworks.com/2020/english/DraftSight/acis/acis_command_sweep.htm

DeWalt Toolboxes. DeWalt Toolbox. Accessed 13 December 2021. Available at

https://www.dewalt.com/products/storage-and-gear/tool-storage/tool-boxes-and-carriers.

Factor I/O. Features. Accessed 12 February 2022. Available at

https://factoryio.com/features/

FileInfo a. Autodesk FBX Interchange File. Accessed 02 May 2022. Available at

https://fileinfo.com/extension/fbx

FileInfo b. GL Transmission Format File. Accessed 02 May 2022. Available at

https://fileinfo.com/extension/gltf

FileInfo c. Stereolithography File. Accessed 02 May 2022. Available at

https://fileinfo.com/extension/gltf

FMI Standard. Functional Mock-up Interface. Accessed 03 May 2022. Available at

https://fmi-standard.org/

Gaisbauer, F. & Agethen, P & Otto, M. & Bär, T. & Sues, J. & Rukzio, E. 2018. Presenting a Modular Framework for a Holistic Simulation of Manual Assembly Tasks. Procedia CIRP. 72. 769.

Gaisbauer, F. & Lampen, E. & Agethen, P. & Rukzio, E. 2021. Combining heterogeneous digital human simulations: presenting a novel co-simulation approach for incorporating different character animation technologies. The Visual Computer 37. 717–734.

Gaisbauer, F. 2019. The MOSIM project: combining digital human simulations - Unite Copenhagen 2019. Accessed 13 December 2021. Available at

https://www.youtube.com/watch?v=kWH6NtR7imA

International Organization for Standardization 2005. ISO 691. Retrieved 14 February 2022. Available at

https://cdn.standards.iteh.ai/samples/37555/9a5edb30557b4a18b913c31686624e1e/ISO-691-2005.pdf

International Organization for Standardization 2012. ISO 9714-1. Retrieved 13 February 2022. Available at

https://cdn.standards.iteh.ai/samples/55438/995d6b72e3a04f2f86a1e92027d02c62/ISO-9714-1-2012.pdf

International Organization for Standardization 2014. ISO 2936. Retrieved 14 February 2022. Available at

https://cdn.standards.iteh.ai/samples/60231/7332b2e7eb6c448b9f8ca6c6ed518206/ISO-2936-2014.pdf

MathWorks. Design and validate PLC algorithms in a simulation environment. Accessed 02 May 2022. Available at

https://se.mathworks.com/discovery/plc-simulation.html

Microsoft 2022. A tour of the C# language. Accessed 02 May 2022. Available at

https://docs.microsoft.com/en-us/dotnet/csharp/tour-of-csharp/

MOSIM Project. Motivation. Accessed 13 December 2021. Available at

https://mosim.eu/index.php

Polycase 2021. What is a Programmable Logic Controller (PLC)? Accessed 03 May 2022. Available at

https://www.polycase.com/techtalk/electronics-tips/what-is-a-programmable-logic-controller.html

Schneider Electric 2021. What is Modbus and How does it work? Accessed 02 May 2022. Available at

https://www.se.com/us/en/faqs/FA168406/

Siemens. Simulation for Automation. Accessed 02 January 2022. Available at

https://new.siemens.com/global/en/products/automation/topic-areas/simulation-for-automation.html

Simulistics. Modular Models. Accessed 02 May 2022. Available at

https://www.simulistics.com/tour/modularmodels.htm

SolidWorks. Success. Accessed 03 January 2021. Available at

https://www.solidworks.com/success

Song, S. & Kidziński, Ł. & Peng, X. & Ong, C. & Hicks, J. & Levine, S. & Atkeson, C. & Delp, S. 2021. Deep reinforcement learning for modeling human locomotion control in neuromechanical simulation. Journal of NeuroEngineering and Rehabilitation 18, 126.

Tulip. What is a smart factory? (And what it means for you). Accessed 02.05.2022. Available at

https://tulip.co/glossary/what-is-a-smart-factory-and-what-it-means-for-you/

Unity. Web page. Accessed 13 December 2021. Available at

https://unity.com/