**Diese Arbeit wurde vorgelegt am Institut für Arbeitswissenschaft**

Projekt-/Bachelor-/Masterarbeit

**Interactive Assembly Systems in Virtual Reality**

vorgelegt von

Zhihui Che

Prüfer: Vollständiger Name des 1. Prüfers

Zweitprüfer: Vollständiger Name des 2. Prüfers

Mitbetreuender

wissenschaftl. Mitarbeiter: Michael Preutenborbeck

Aachen, XXXXX

Sperrvermerk:

Die vorliegende Arbeit beinhaltet interne vertrauliche Informationen der <Unternehmen>. Die Weitergabe des Inhaltes der Arbeit und eventuell beiliegender Zeichnungen und Daten im Gesamten oder in Teilen ist grundsätzlich untersagt. Es dürfen keinerlei Kopien oder Abschriften – auch nicht in digitaler Form – gefertigt werden. Ausnahmen bedürfen der schriftlichen Genehmigung der <Unternehmen>.

Eidesstattliche Erklärung (RWTH-Vorlage) -HIER EINBINDEN-

Die jeweils aktuell gültige Eidesstattliche Versicherung ist von der Homepage des ZPA zu beziehen und hier unterschrieben einzubinden. Um dies nicht zu vergessen, ist es sinnvoll diese Seite stehen zu lassen und erst beim Druck der Arbeit diese Seite gegen das vom ZPA bezogene Dokument physisch in Papierform auszutauschen.

Antrag auf Verwendung des RWTH Logos (RWTH-Vorlage) – nicht einbinden -

Neben der Eidesstattlichen Versicherung ist es weiterhin nötig, einen Antrag auf Verwendung des RWTH Logos beim ZPA zustellen. Der Antrag muss nicht in die Arbeit eingebunden werden. Der Antrag kann entweder persönlich im ZPA abgeholt werden oder aber er kann auf der Seite des ZPAs herunterladen und dann ausgedruckt werden. Dieser Antrag ist spätestens zusammen mit den Pflichtexemplaren bei der Abgabe der Arbeit dem ZPA unter-schrieben vorzulegen. Es sollte in diesem Zusammenhang auch immer die aktuell geltenden Gestaltungsrichtlinien für das Titelblatt von Abschlussarbeiten geprüft werden.

**Zusammenfassung**

**- Zusammenfassung in deutscher Sprache hier einbinden –**

Formulieren Sie bitte eine Zusammenfassung Ihrer Arbeit in 100 Wörtern und fügen Sie diese an dieser Stelle in Ihre Arbeit ein. Aus unserer Erfahrung heraus hat es sich als Vorteilhaft gezeigt, die Zusammenfassung als letztes zu formulieren. Um Ihre Arbeit auch einem nicht deutschsprachigen (Fach-)Publikum zu öffnen, übersetzten Sie Ihre Zusammenfassung im Anschluss ins Englische und binden diese ebenfalls auf dieser Seite ein.

**Abstract**

**- Zusammenfassung in englischer Sprache hier einbinden -**

Für die Übersetzung Ihrer Zusammenfassung können Sie Hilfsmittel, wie etwa den kostenfreien Übersetzungsservice www.Deepl.com benutzen. Stellen Sie bitte sicher, dass Sie Ihre Zusammenfassung in „British Englisch“ formulieren.

1. Inhaltsverzeichnis

[I Inhaltsverzeichnis i](#_Toc133485525)

[II Formelzeichen und Abkürzungen (Beispiel) iii](#_Toc133485526)

[III Abbildungsverzeichnis iv](#_Toc133485527)

[IV Tabellenverzeichnis v](#_Toc133485528)

[1 Introduction 6](#_Toc133485529)

[1.1 Motivation 6](#_Toc133485530)

[1.2 Thesis Structure 6](#_Toc133485531)

[1.3 Objectives 6](#_Toc133485532)

[2 Theoretical Background 8](#_Toc133485533)

[2.1 Virtual Reality 8](#_Toc133485534)

[2.1.1 The Mixed Reality Continuum 8](#_Toc133485535)

[2.1.2 Components of Virtual Reality 9](#_Toc133485536)

[2.1.3 VR Devices 10](#_Toc133485537)

[2.1.4 Haptic Devices 12](#_Toc133485538)

[2.1.5 Simulation Environment 13](#_Toc133485539)

[2.1.6 Applications in Engineering Industry 13](#_Toc133485540)

[2.2 Virtual Assembly 14](#_Toc133485541)

[2.3 Unity 15](#_Toc133485542)

[2.3.1 Game Object 15](#_Toc133485543)

[2.3.2 Collision Detection 16](#_Toc133485544)

[2.3.3 Rigid Body 18](#_Toc133485545)

[2.3.4 Raycasting 18](#_Toc133485546)

[2.4 Data Exchange 19](#_Toc133485547)

[2.5 Program Design 20](#_Toc133485548)

[2.5.1 Object-Oriented Programming (OOP) 20](#_Toc133485549)

[2.5.2 Six Design Principles 21](#_Toc133485550)

[2.5.3 Data Structure 22](#_Toc133485551)

[3 Method and Implementation 24](#_Toc133485552)

[3.1 The Concept of Solution 24](#_Toc133485553)

[4 Results and Discussion 26](#_Toc133485554)

[5 Conclusion 27](#_Toc133485555)

[ Literaturverzeichnis 28](#_Toc133485556)

1. Formelzeichen und Abkürzungen (Beispiel)

| Formelzeichen | Einheit | Beschreibung |
| --- | --- | --- |
| ae | mm | Eingriffsweite |
| ap | mm | Schnitttiefe |
| fz | mm | Vorschub je Zahn |
| z |  | Zähnezahl [Anmerkung: Sortiert wird alphabetisch, ungeachtet der Groß- und Kleinschreibung] |
| α |  | Einfallswinkel |
| Φ |  | ... [Anmerkung: Griechische Buchstaben werden nach den lateinischen angeführt] |
|  |  |  |
| Abkürzung |  | Beschreibung |
| DP |  | polykristalliner Diamant |

1. List of Figures

[Figure 1 The diagram of the mixed reality Continuum 8](#_Toc133485653)

[Figure 2 Ocules Rift and HTC Vive 11](#_Toc133485654)

[Figure 3 RWTH aixCAVE 12](#_Toc133485655)

[Figure 4 Rendered Hand Model in Unity 17](#_Toc133485656)

[Figure 5 Hand Shape is approximated by multiple colliders 17](#_Toc133485657)

[Figure 6 A Game Object is selected by raycasting 19](#_Toc133485658)

[Figure 7 Graphic Demonstration of Linked Lists 23](#_Toc133485659)

[Figure 8 Graph Representation of Assembly Parts Connection 24](#_Toc133485660)

[Figure 9 Partitioned Assembly Parts 24](#_Toc133485661)

1. List of tables

[Table 1 The table of comparison of the pros and cons of Real Environment, AR, AV and VE 9](#_Toc133485787)

[Table 2 The diagram of XR markt size worldwide from 2021 to 2026 14](#_Toc133485788)

# Introduction

## Motivation

Nowadays, the manufacturing companies need to change under the pressure of global mega trends. On the one hand, an increase of worldwide-connected business activities will raise the complexity within manufacturing networks. On the other hand, a volatile demand and customized products of the companies will influence the production and planning processes. (Bartodziej 2017, S. 1) In order to face these challenges, as well as to ensure competitiveness in the global market, the term of ‘Industry 4.0’ is in Germany proposed and has become an eponym for a future project in the context of the high-tech strategy 2020. (Lasi et al. 2014)

Industry 4.0 is primarily understood as a new concept for industrial production in which digitalization technologies are increasingly used. (Steven 2018) Cyber-Physical Systems (CPS) are one of essential components of Industry 4.0, each physical object is accessible and controllable in information technology (IT) systems, which can also mean that each physical object has a virtual model in IT systems. Thus, an interface to interact with virtual objects without touching or changing on physical objects is necessary. Due to its immersive visualization, realistic rendering, and intuitive interactions, virtual reality (VR) offers such an interface with great potential.

In response to increasingly competitive market and technological developments, VR is applied in the whole product life cycle. On the one hand, VR can be used in product development. The product can be modified and explored virtually, thus development times and costs are directly reduced. On the other hand, in production planning VR offers a virtual environment that product can be interacted intuitively to increase the working efficiency, and in this environment different departments can easily work collaboratively, which reduces the invisible costs. More concrete examples are given in the section 2.2.

With the advancement in immersive technologies, haptic equipment has also been developed, which adds haptic feedback in addition to the visual and auditory experience of VR. The simulation of the virtual environment is more realistic, due to increase of the immersion. The additional feedback also promises increased quality of the information provided in the virtual environment (Preutenborbeck et al. 2022) to enhance the aimed functionality of application.

## Thesis Structure

## Objectives

Research in the field of virtual reality shows that virtual reality technologies can help engineers and other stakeholders to accomplish tasks in product development or product ramp-up better. Under the tide of digital transformation, more potential of virtual reality is being tapped. One potential application for virtual reality is the assembly training. The first step to make use of this technology is to develop a virtual environment, in which assembly parts are visualized, their physical behaviors are simulated, so that users can interact with them as naturally as possible.

The objective of this thesis is to design and develop an interactive virtual assembly system. In order to achieve that, two parts must be taken into account. The first part focuses on transforming CAX data of products or workstations into a pre-defined data structure for virtual environments. This also includes the product hierarchy, which is needed to determine relationships between parts, assemblies and the final product for the correct assembly sequence.

The second part focusses on building an interactive assembly environment in Unity3D. The assembly system needs to be aware if workers interact with assembly parts correctly in correct sequence. In order to support the immersion of the virtual assembly system, haptic gloves are used for the haptic interaction with the assembly process. At the end, the virtual assembly system will be explored and evaluated in a user study.

# Theoretical Background

## Virtual Reality

Virtual Reality (VR) is a technology that creates a simulated environment in which users can interact with digital objects and spaces in real-time. VR has its roots in the early 1960s, when Ivan Sutherland developed the first head-mounted display (HMD) at the Massachusetts Institute of Technology (MIT) (Sutherland 1968). Since then, VR has evolved and expanded in different areas of application, from entertainment and gaming to education and training, and has become increasingly accessible to a wider audience due to the development of more affordable and portable devices.

### The Mixed Reality Continuum

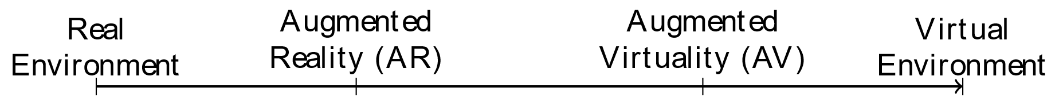


Figure 1 The diagram of the mixed reality Continuum

The Mixed Reality Continuum (MRC) is a framework that describes the range of virtuality and reality that can be experienced by users in digital environments (Bamodu und Ye 2013). The MRC includes four categories:

* Reality-Reality (RR), which refers to the physical environment without any digital augmentation.
* Augmented Reality (AR), which overlays digital content onto the physical environment.
* Augmented Virtuality (AV), which places users in a virtual environment with some elements of the physical environment.
* Virtual Reality (VR), which immerses users in a fully digital environment.
* The MRC highlights the different levels of immersion, interactivity, and presence that can be achieved with different types of digital environments and devices.

This table( (Billinghurst et al. 2015), (Bower et al. 2014), (LaViola et al. 2017)) provides a concise comparison of the pros and cons of Real Environment, Augmented Reality (AR), Augmented Virtuality (AV), and Virtual Environment (VE). It highlights the unique advantages and challenges associated with each type of environment, offering a clear understanding of their distinct characteristics and applications.

Table 1 The table of comparison of the pros and cons of Real Environment, AR, AV and VE

|  |  |  |
| --- | --- | --- |
| Environment | Pros | Cons |
| Real Environment | * Direct physical interaction * No technological barriers | * Limited by physical constraints * Lacks real-time data manipulation |
| Augmented Reality (AR) | * Enhances real environment * Facilitates learning and decision-making * Promotes collaboration | * Dependence on hardware * Limited field of view * Potential user distraction |
| Augmented Virtuality (AV) | * Integrates real-world * Enhanced immersion and interactivity | * Requires advanced technologies * Latency and synchronization issues |
| Virtual Environment (VE) | * Fully immersive experiences * Simulates complex scenarios * Remote collaboration | * Motion sickness * Limited by hardware and software |

### Components of Virtual Reality

VR experiences can be analyzed in terms of three dimensions of immersion, interactivity, and impression (Bamodu und Ye 2013). The concept of 3I is a powerful framework for understanding the potential of VR technology and its applications.

1. Immersion:

The concept of immersion in VR refers to the extent to which the user feels present and engaged in the virtual environment. According to Slater (Slater 2009), immersion can be achieved through a combination of sensory inputs, including visual, auditory, and haptic feedback, as well as through the creation of a plausible and coherent virtual world. Bailenson and Blascovich (Bailenson et al. 2008) further argue that the use of avatars, which allow users to represent themselves in the virtual world, can enhance immersion by enabling social interaction and a sense of embodiment.

1. Interaction:

The concept of interaction in VR refers to the ways in which the user can engage with and manipulate objects and environments in the virtual world. According to Laviola et al. (LaViola et al. 2017), interaction can be facilitated through a range of input devices, including controllers, haptic interfaces, and speech recognition. Slater and Wilbur (Slater und Wilbur 1997) suggest that interaction is essential for maintaining immersion and a sense of presence in the virtual environment, as it allows the user to actively explore and manipulate the virtual world.

1. Imagination:

The concept of imagination in VR refers to the ways in which the user can use their creativity and cognitive processes to interpret and engage with the virtual world. According to Riva et al. (Riva et al. 2014), imagination is closely related to the concept of presence, as it involves a sense of being mentally and emotionally present in the virtual environment. Sanchez-Vives and Slater (Sanchez-Vives und Slater 2005) further argue that the use of imagination in VR can lead to a sense of consciousness and self-awareness, as users become more attuned to their thoughts, feelings, and bodily sensations.

### VR Devices

Virtual reality devices are an essential component of the VR experience, as they enhance immersion by providing users with a sense of presence and realism within the virtual environment. This heightened sense of immersion is achieved through several key factors:

* Stereoscopic displays: VR headsets utilize stereoscopic displays, which create a sense of depth by presenting slightly different images to each eye. This binocular disparity simulates the manner in which our eyes perceive depth in the real world, making the virtual environment appear more realistic (Tam et al. 2011).
* Field of view: VR devices offer an expansive field of view (FOV), typically ranging from 100-110 degrees, encompassing the user's peripheral vision. This wide FOV contributes to the sensation of being surrounded by the virtual environment, thereby enhancing the sense of presence (Lin et al. 2002).
* Head tracking: VR devices employ various sensors to accurately track the user's head movements. This head tracking allows the virtual environment to respond in real-time to the user's head movements, creating a natural and intuitive interaction (Stauffert et al. 2020).
* Motion tracking: Some VR devices also incorporate motion tracking for the user's hands and body, enabling more natural interaction within the virtual environment. This interactivity improves immersion by allowing users to manipulate virtual objects similarly to how they would interact with objects in the real world (.N.iehorster et al. 2017).
* 3D audio: Spatial audio systems in VR devices contribute to a more immersive sound experience by simulating how sounds travel and interact within the environment. This audio processing offers a more accurate and engaging sound experience, adding to the overall sense of immersion (Coughlan et al. 2020)
* High refresh rate and low latency: VR devices require high refresh rates (typically 90 Hz or higher) and low latency to ensure smooth and responsive visuals. These factors are essential for maintaining immersion and preventing motion sickness in VR experiences (LaValle et al. 2014).

By combining these elements, VR devices deliver an immersive experience that allows users to feel genuinely present within the virtual environment. This immersion opens up new possibilities for various applications, including gaming, education, training, and more.

Two of the most commonly used types of VR devices are head-mounted displays (HMDs) and room-scale motion tracking devices (RMDs).

HMDs:

|  |  |
| --- | --- |
|  | Ein Bild, das Elektronik, Kamera, Projektor enthält.  Automatisch generierte Beschreibung |

Figure 2 Ocules Rift and HTC Vive

HMDs are devices that are worn on the user's head, covering the eyes and sometimes the ears, and display a stereoscopic image of the virtual environment. These devices typically include a tracking system that detects the user's head movements and adjusts the displayed image accordingly, providing a seamless and immersive experience. HMDs range in quality and price, from basic models like Google Cardboard to high-end devices like the Oculus Rift and HTC Vive.

HMDs are particularly well-suited for applications that require a high degree of immersion and individual exploration, such as gaming and entertainment. HMDs are also commonly used for virtual training simulations, where users can practice real-world scenarios in a controlled and safe environment.

RMDs:

RMDs, also known as positional tracking devices, are devices that track the user's movements in a physical space and translate them into the virtual environment. RMDs can include handheld controllers or full-body motion tracking systems, such as the Microsoft Kinect. RMDs are particularly useful for applications that require physical movement and interaction, such as virtual sports and physical rehabilitation.

RMDs can also be combined with HMDs to provide a more immersive and interactive experience. For example, the Oculus Rift S includes both HMD and RMD capabilities, enabling users to move and interact with objects in the virtual environment using handheld controllers.

Overall, HMDs are ideal for individual exploration and immersion, while RMDs are well-suited for physical interaction and collaboration.

Ein Bild, das Im Haus, Decke, Zimmer enthält.

Automatisch generierte Beschreibung

Figure 3 RWTH aixCAVE

### Haptic Devices

In addition to the key factors mentioned in the last section, haptic feedback is important to improve the VR experience, as it provides a sense of touch and physical presence in the virtual environment. Haptic devices are designed to simulate tactile and kinesthetic stimuli, such as pressure and vibration. Tactile feedback refers to the sensation of touch on the skin, such as texture and temperature while kinesthetic feedback is related to the perception of body movement and position in space (Lackner und DiZio 2005) (Lederman und Klatzky 2009). By simulating these sensations, haptic devices allow users to interact with virtual objects and environments in a more realistic and immersive way. This can improve the user's engagement and enjoyment of VR applications and provide more effective training in certain industries. Additionally, haptic feedback can provide valuable information to users in situations where visual or auditory feedback may not be sufficient, such as in high-pressure or high-risk environments (Jones und Sarter 2008).

Haptic gloves are a common type of haptic device that use embedded sensors and actuators to simulate the sensation of touch on the user's hands. Haptic gloves are particularly well-suited for applications that involve fine motor skills, such as surgery training and mechanical assembly. According to a literature review by Coles et al. (Coles et al. 2011), haptic feedback has been shown to be effective in medical training simulators, including those that use haptic gloves for surgical training. It is also found that haptic feedback enhances force skill learning, which may have implications for the use of haptic gloves in other domains (Morris et al. 2007).

Other types of haptic devices include haptic vest, haptic chairs, haptic floors, and haptic feedback controllers, each designed to provide a different type of haptic feedback and enhance the VR experience.

### Simulation Environment

Simulation environments are software platforms that provide developers with the necessary tools and resources to create and develop virtual reality applications and experiences. These platforms serve as a foundation for creating virtual environments, modeling objects and characters, and integrating different features and functionalities.

Two of the most widely used simulation environments are Unity and Unreal. Unity is a cross-platform engine that supports the development of VR applications for various platforms such as Windows, macOS, Android, iOS, and Web. It is known for its intuitive user interface and extensive library of assets, which includes pre-made 3D models, animations, and sound effects. Unity also offers advanced features such as real-time rendering, physics simulation, and artificial intelligence.

Unreal, on the other hand, is a game engine developed by Epic Games that has gained popularity for its photorealistic graphics and advanced physics simulation. It offers a range of tools and features for creating immersive VR experiences, including support for VR headsets, advanced lighting and rendering options, and an extensive material editor for creating realistic textures and surfaces. Unreal also supports a wide range of programming languages, including C++, Python, and Blueprint.

Both Unity and Unreal provide developers with access to a range of development resources, including online communities, documentation, and tutorials. This makes it easier for developers to learn the ropes of VR development and troubleshoot any issues that may arise during the development process.

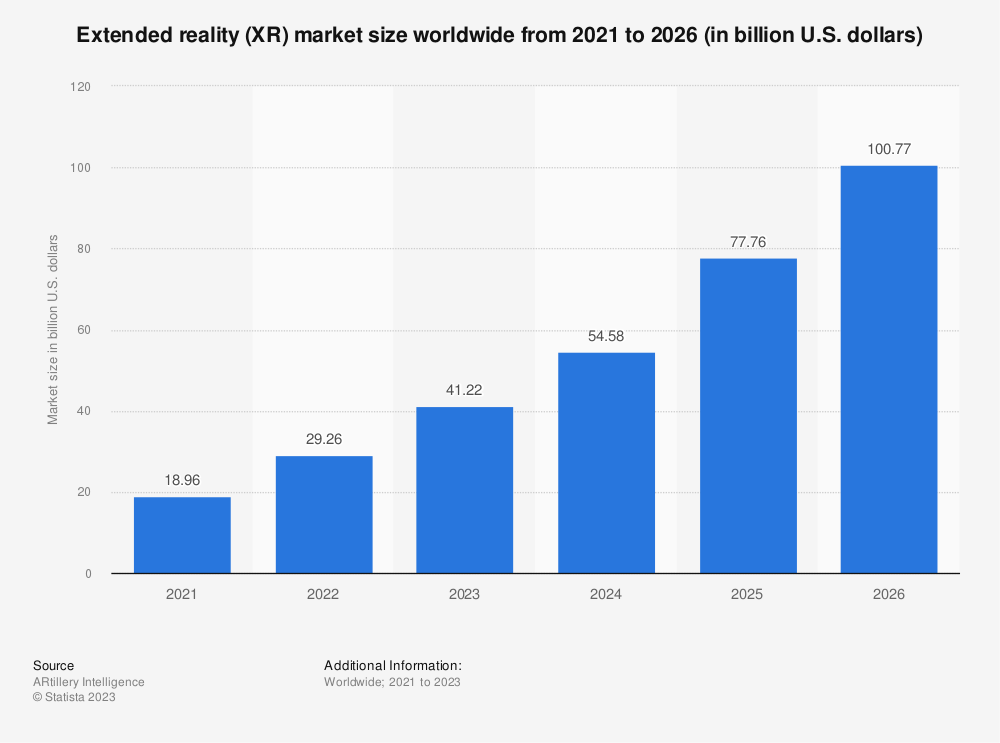
In conclusion, simulation environments such as Unity and Unreal are essential tools for developers looking to create VR applications and experiences. They provide a range of features and resources that simplify the development process and enable developers to create immersive and engaging virtual worlds.

### Applications in Engineering Industry

Virtual reality has emerged as a promising technology with vast applications in different industries, including training and education, design and prototyping, marketing and advertising, remote collaboration, and maintenance and repair. Its adoption is rapidly increasing, driven by the potential to improve efficiency, effectiveness and safety.

The extended reality (XR) market, encompassing augmented reality (AR), virtual reality (VR), and mixed reality (MR), has witnessed a remarkable growth rate, reaching 29.26 billion U.S. dollars in 2022, with a projected increase to over 100 billion U.S. dollars by 2026, according to industry statistics (ARtillery Intelligence. 2023).This growth offers unprecedented opportunities for businesses and organizations to utilize these immersive technologies to revolutionize a variety of fields, including entertainment, healthcare, education, and more. One of the key applications of VR in industry is training and education. VR can provide a safe and controlled environment for individuals to learn and practice complex tasks, such as surgical procedures and flight simulations (Jimoyiannis). VR training has been shown to be more effective than traditional methods in improving learning outcomes and retention (Desiderio Cano Porras et al. 2018).

Table 2 The diagram of XR markt size worldwide from 2021 to 2026



Another application of VR in industry is design and prototyping. VR can be used to create digital prototypes and models, allowing designers and engineers to test and refine products before they are physically built. This can save time and money and lead to better designs . VR can also be used for marketing and advertising, allowing customers to experience products and services in a more engaging way. For example, real estate companies have used VR to provide virtual tours of properties, leading to increased sales (Ganapathy 2016).

VR can also facilitate remote collaboration and communication, allowing team members from different locations to work together in a virtual environment. This can be especially useful in industries such as architecture and engineering (Rainer Müller, Leenhard Hörauf, Max Eichenwald). Finally, VR can provide maintenance and repair technicians with virtual training and support, allowing them to diagnose and fix problems more efficiently (Pidel und Ackermann 2020).

VR has numerous applications in industry, and its adoption is rapidly increasing. The benefits it offers, such as improved efficiency, safety, and effectiveness, make it an attractive technology for many industries. As technology continues to advance, it is expected that the potential uses of VR will continue to expand, leading to further adoption in industry.

## Virtual Assembly

Virtual assembly (VA) is a computer-aided process that enables engineers and designers to simulate the assembly of complex products in a virtual environment. By using advanced modeling and simulation techniques, virtual assembly allows for the identification and resolution of potential design and assembly issues before actual production. This approach reduces the need for physical prototypes, streamlines the product development process, and facilitates efficient collaboration among multidisciplinary teams.

Virtual assembly plays a significant role in the product life cycle, from the initial design phase to the final stages of production and maintenance. Some examples of how virtual assembly assists in the product life cycle include:

* Design Optimization: Virtual assembly allows designers to identify and address potential assembly issues early in the design phase, reducing the need for costly and time-consuming design revisions later on. For example, Zhu et al. (Zhu und Du 2021)] demonstrated how virtual assembly could improve the design and manufacturability of automotive components.
* Manufacturing Planning: By simulating assembly processes in a virtual environment, engineers can optimize manufacturing workflows, reduce assembly time, and minimize the risk of production delays. A study by Bennett et al. (G.R. Bennett) showed that virtual assembly could significantly reduce assembly time and improve manufacturing efficiency in the aerospace industry.
* Training and Skill Development: Virtual assembly can be used to train assembly line workers, helping them become familiar with new assembly processes and procedures before they are implemented on the production floor. This approach has been successfully applied in industries such as aerospace (Piechowski et al. 2020) and automotive manufacturing (Chryssolouris et al. 2000).

In conclusion, virtual assembly is an essential tool in modern product development, offering numerous benefits throughout the product life cycle. The integration of haptic technology further could enhance the virtual assembly experience, providing users with a more realistic and immersive interaction with virtual objects, leading to improved efficiency and reduced errors in the assembly process.

## Unity

In this section, we will introduce Unity, a cross-platform game engine and development environment used to create 2D and 3D games, simulations, and other interactive experiences. It provides a range of tools and features to help developers create rich and engaging content, including a visual editor, scripting tools, physics engine, and support for multiple platforms and devices. Unity is widely used in the game industry, but it also has applications in fields such as architecture, engineering, and education.

### Game Object

A Game Object is a fundamental concept in game development, as it is a generic container for components that define its behavior, appearance, and functionality within a game or virtual environment. It allows developers to organize, manipulate, and interact with various elements in a scene, such as characters, items, terrain, or user interface elements.

Moreover, Game Objects can be thought of as a collection of components that work together to create a specific functionality or behavior. Components can include graphical elements, such as 3D models or 2D sprites, physics properties, audio sources, and custom scripts that define interactivity and game logic. By combining various components, developers can create complex and interactive objects that exhibit unique behaviors and properties.

In addition, Game Objects can be organized in a hierarchical structure, with parent and child relationships that define their relative positioning, scaling, and rotation. This hierarchy allows for easy management and organization of objects within a scene, as well as the creation of complex, interconnected systems and interactions.

Specifically, in Unity, Game Objects are represented as instances within the game world and can be created, manipulated, and destroyed during runtime. The Unity engine provides a built-in component system that enables developers to easily add, remove, and modify components to create a wide range of behaviors and interactions. Furthermore, the Unity scripting system allows developers to extend the functionality of Game Objects and components by writing custom scripts using C# programming language. This flexibility and extensibility make Unity a powerful tool for creating immersive and interactive virtual environments.

### Collision Detection

Collision detection is a crucial aspect of creating realistic and interactive virtual environments in Unity. It involves determining when and where objects within a scene come into contact with each other. In this section, we will discuss how collision detection works in Unity and some strategies for efficiently managing multiple colliders.

Unity's physics engine, which is built on top of the PhysX library, handles collision detection for 3D objects. The engine uses various types of colliders, such as primitive colliders (spheres, boxes, or capsules) and mesh colliders, to approximate the shapes of objects. When objects with colliders and attached Rigidbody components come into close proximity, the physics engine calculates potential collision pairs and generates collision events if necessary.

Efficiently managing multiple colliders is important in Unity and other game engines due to several interconnected factors. One key aspect is performance optimization, as it helps reduce unnecessary collision calculations, resulting in better overall performance. This is especially crucial for VR applications, where maintaining a high frame rate is essential to ensure a smooth and comfortable user experience.

Moreover, using multiple colliders allows developers to create more realistic and interactive virtual environments. By approximating complex object shapes with a combination of simpler colliders, developers can achieve accurate collision detection and response, leading to more convincing interactions between objects.

Additionally, properly managing multiple colliders can help conserve system resources, such as memory and CPU usage. Techniques like layering, collision filtering, and bounding volume hierarchy can be employed to minimize the computational overhead associated with collision detection and keep resource usage in check.

Scalability is another important factor, as virtual environments become larger and more complex, the number of colliders and potential collision pairs can grow exponentially. Efficiently managing multiple colliders ensures that applications can scale up to accommodate more complex scenes without compromising performance or interactivity.

Furthermore, employing multiple colliders allows developers to create custom collision interactions tailored to specific gameplay requirements or design constraints. For example, developers can use trigger colliders to create invisible boundaries or zones that generate events without simulating a physical collision, allowing for unique game mechanics or special effects.

In conclusion, effectively managing multiple colliders in Unity enables developers to create rich, immersive, and interactive virtual environments that maintain optimal performance and deliver a satisfying user experience.

|  |  |
| --- | --- |
| Figure 4 Rendered Hand Model in Unity | Figure 5 Hand Shape is approximated by multiple colliders |

To manage multiple colliders efficiently, developers can use a combination of layering, collision filtering, bounding volume hierarchy (BVH), compound colliders, and trigger colliders:

* Layering and Collision Filtering: Assign objects to specific layers and use the Physics Manager's "Layer Collision Matrix" to specify which layers can interact with each other. This approach helps reduce unnecessary collision calculations and improves performance.
* Bounding Volume Hierarchy (BVH): Organize colliders into a hierarchical tree structure, with larger bounding volumes enclosing smaller ones, to quickly discard colliders that are not in close proximity. Unity's physics engine automatically builds and updates BVH structures for static colliders, while dynamic colliders can be managed using custom scripts or third-party plugins.
* Compound Colliders: Use multiple primitive colliders to approximate the shape of complex objects, rather than using a single mesh collider. This technique can significantly improve collision detection performance, especially for objects that frequently interact with other objects in the scene.
* Trigger Colliders: Create invisible boundaries or zones that generate events when objects pass through them, without simulating a physical collision. By setting the "Is Trigger" property on a Collider component, it becomes a trigger collider that will trigger OnTriggerEnter, OnTriggerStay, and OnTriggerExit callback methods.

### Rigid Body

In Unity, a Rigidbody is a component that enables physics-based interactions for a 3D object. By attaching a Rigidbody component to an object, you allow it to be influenced by forces, gravity, and other physical effects present in the scene. This component essentially brings the object under the control of Unity's built-in physics engine, which is based on the popular PhysX engine developed by NVIDIA.

The Rigidbody component works by applying Newton's laws of motion to the object. When a force is applied to an object with a Rigidbody, it causes the object to accelerate, depending on its mass. Similarly, torque can be applied to the object, resulting in angular acceleration and rotation. The physics engine also takes into account other factors such as friction, drag, and collision response when calculating the motion of a Rigidbody.

When an object with a Rigidbody component collides with another object that has a Collider component, the physics engine calculates the collision response and updates the position and rotation of the Rigidbody accordingly. This allows for realistic interactions between objects in the scene, enabling a wide range of applications, such as simulations, virtual reality experiences, and interactive 3D visualizations.

It is essential to note that Rigidbody objects should not be moved using the Transform component directly, as this can cause unrealistic behavior and inconsistencies in the physics simulation. Instead, forces, torques, or the Rigidbody's velocity and angular velocity properties should be used to manipulate the object's movement.

### Raycasting

Raycasting is a technique used in computer graphics and game development to simulate the behavior of a ray of light or a line segment, which is cast or projected from an origin point into a scene. The primary purpose of raycasting is to determine if the ray intersects any objects within the scene and to gather information about those intersections.

In Unity, raycasting is implemented using the Physics.Raycast() function, which takes several parameters, including the origin point of the ray, the direction in which the ray is cast, and an optional distance parameter that limits the length of the ray. When the raycast is performed, it calculates the intersections between the ray and the colliders of objects in the scene. If the ray intersects an object, the function returns true, and additional information about the intersection point, such as the distance from the origin and the normal vector of the surface at the intersection, can be accessed through a RaycastHit object.

The theory behind raycasting is based on the concept of linear algebra and vector mathematics. By representing the ray as a parametric equation, we can determine the points in 3D space where the ray intersects the surfaces of objects. These intersections are then tested against the geometric representations of the objects (e.g., bounding boxes, spheres, or more complex mesh colliders) to determine if the ray truly intersects the object.

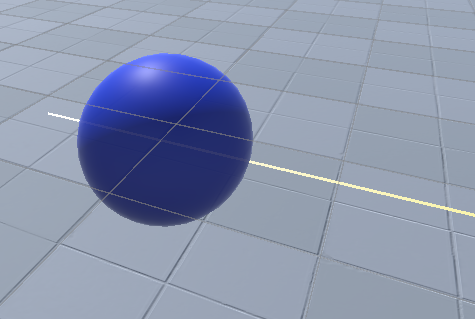


Figure 6 A Game Object is selected by ray casting

## Data Exchange

CAD data exchange is a critical aspect of modern product development that enables the sharing of 3D models, simulation data, and other product-related information between different software systems. Understanding the compatibility and conversion options for CAD data exchange to VR environments is crucial for creating immersive and interactive experiences for product design and development.

Compatible data types for CAD software include IGES, STEP, and STL, which are widely supported by most 3D modeling and CAM software. For VR environments, formats such as OBJ, FBX, and Collada are commonly used, with Unity supporting a wide range of formats. However, compatibility issues such as scale and axis differences can arise when importing CAD data into Unity, requiring additional manual intervention to achieve the desired results.

One approach to changing CAD data to be compatible with VR without using third-party software is to export CAD data to a common format such as OBJ, FBX, or Collada and import the data into Unity. However, this may require additional manual intervention to achieve the desired results, such as adjusting the scale and orientation of the model to match the Unity environment. Another approach is to use Unity's built-in 3D modeling tools to modify the CAD data directly within the Unity environment. This requires a solid understanding of Unity's 3D modeling tools and may not be suitable for more complex CAD models.

Third-party software such as Pixyz and Datasmith can be used to streamline and automate the CAD data exchange to VR environments. Pixyz offers features such as geometry simplification, mesh optimization, UV mapping, and material conversion, among others, enabling the conversion of CAD data into various game engines and VR environments. Datasmith provides a data import and optimization plugin for Unreal Engine, allowing users to import CAD data directly into Unreal Engine with features such as automated data processing, UV mapping, and material conversion.

Geometry simplification reduces the complexity of 3D models by removing redundant or unnecessary geometry, improving overall performance, and reducing file size. Mesh optimization improves the quality of 3D meshes by reducing polygon count and improving topology, improving visual quality, and reducing file size. UV mapping maps 2D textures onto 3D models, essential for creating realistic textures and materials. Material conversion converts materials from a CAD format to a format compatible with VR environments, ensuring compatibility with VR engines.

Overall, a comprehensive understanding of different data types, formats, and conversion options is necessary for effective CAD data exchange to VR environments. Leveraging compatible data types, converter software, and scaling techniques, and features such as geometry simplification, mesh optimization, UV mapping, and material conversion can help designers and engineers create immersive and interactive VR experiences for product design and development, reducing design time and costs, and improving collaboration among team members.

## Program Design

### Object-Oriented Programming (OOP)

Object-Oriented Programming (OOP) is a programming paradigm that is based on the concept of objects, which encapsulate data and behavior. OOP is widely used in software development because it offers several benefits over other programming paradigms. OOP provides a modular, reusable, and extensible approach to programming, making it easier to develop and maintain large-scale software projects.

OOP should be used when a program requires a modular and extensible design, where different parts of the program can be easily modified or extended without affecting other parts. OOP also provides a natural way to model real-world objects, making it easier to understand and implement complex systems. The benefits of OOP include code reusability, modularity, extensibility, maintainability, and scalability.

OOP thinking involves the creation of classes, which define the attributes and behaviors of objects. A class is a blueprint for an object, which defines its attributes (data) and behaviors (methods). An object is an instance of a class, which can be created and manipulated at runtime. Objects can communicate with each other by sending messages, which are calls to their methods.

### Six Design Principles

There are six fundamental design principles in object-oriented programming, which are commonly known as SOLID principles. These principles provide guidelines for creating maintainable and extensible software systems. The six principles are:

Single Responsibility Principle (SRP):

The Single Responsibility Principle states that a class should have only one responsibility. This means that the class should only have one reason to change. By adhering to this principle, each class becomes focused on a specific task, which leads to more maintainable and testable code. Breaking up large classes into smaller, focused classes can make code more modular, easier to understand, and less prone to bugs.

Open/Closed Principle (OCP):

The Open/Closed Principle states that a class should be open for extension but closed for modification. This means that the behavior of a class can be extended without changing its source code. By following this principle, new functionality can be added without modifying existing code, which reduces the risk of introducing bugs or breaking existing code. This principle is often implemented through inheritance, composition, or the use of interfaces.

Liskov Substitution Principle (LSP):

The Liskov Substitution Principle states that subtypes should be able to be used in place of their supertypes without affecting the correctness of the program. This means that derived classes should be able to be used in place of their base classes without any unexpected side effects. This principle ensures that the code is flexible and that new subtypes can be easily integrated into the codebase without affecting the correctness of the program.

Interface Segregation Principle (ISP):

The Interface Segregation Principle states that clients should not be forced to depend on interfaces they don't use. This means that interfaces should be fine-grained and focused on specific behaviors. By following this principle, the code becomes more modular, and dependencies between classes are reduced. This principle also promotes code reuse and makes it easier to extend or modify the code.

Dependency Inversion Principle (DIP):

The Dependency Inversion Principle states that high-level modules should not depend on low-level modules. Instead, both should depend on abstractions. Abstractions should not depend on details. Details should depend on abstractions. This means that modules should not depend on each other directly, but instead, they should depend on abstractions, such as interfaces or abstract classes. By adhering to this principle, the code becomes more modular, extensible, and easier to test.

Composition Over Inheritance:

The Composition Over Inheritance principle states that inheritance should be used sparingly, and composition should be favored. This means that behavior should be added to a class by composing it with other objects rather than inheriting behavior from parent classes. By following this principle, the code becomes more flexible and easier to extend. Inheritance can often lead to tight coupling between classes, making the code less modular and harder to maintain over time.

By following these principles, software developers can create systems that are modular, extensible, and maintainable over time. Meeting these principles ensures that the code is easy to understand, modify, and extend, reduces the likelihood of bugs and unintended consequences, and improves the overall quality of the software.

### Data Structure

A data structure is a way of organizing and storing data in a computer so that it can be accessed and used efficiently. It provides a framework for how data can be organized and manipulated and is an important concept in computer science and software engineering. Data structures can be classified as primitive data structures or abstract data structures, with abstract data structures being more complex and typically implemented using primitive data structures.

Data structures are important because they allow programmers to manage large amounts of data efficiently, which is critical in many applications. Good data structures can reduce the time and space complexity of algorithms, making them faster and more efficient. They can also help to organize data in a way that makes it easier to search, sort, and manipulate, and can provide a foundation for more complex algorithms and data analysis.

Lists are a basic data structure that can represent a collection of items, such as a list of names or a list of numbers. There are different types of lists, including arrays, linked lists, and dynamic arrays, and they can be implemented in many different programming languages. Linked lists are a type of data structure that consists of a sequence of elements, where each element contains a reference to the next element in the sequence. Unlike arrays, linked lists can be stored in non-contiguous memory locations, allowing for dynamic memory allocation.

A linked list is constructed by defining a node class that contains a data attribute and a reference to the next node in the sequence. The first node in the sequence is called the head node. To add a new node to the linked list, a new instance of the node class is created with the desired data, and its next reference is set to None. If the list is empty, the new node becomes the head node. Otherwise, the next reference of the last node in the sequence is set to the new node.

Linked lists can be used to implement algorithms for searching, sorting, and manipulating data, and can be used to implement dynamic data structures, such as hash tables or trees. In a singly-linked list, each element contains a reference to the next element in the sequence. In a doubly-linked list, each element contains references to both the next and previous elements in the sequence.

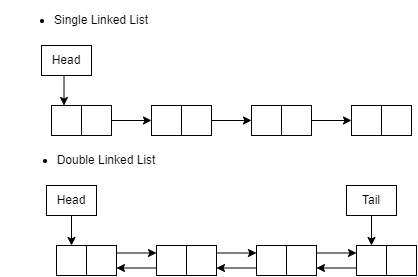


Figure 7 Graphic Demonstration of Linked Lists

# Method and Implementation

## Concepts

### The Concept of System

The primary objective of this thesis is to develop a comprehensive virtual assembly system designed to enhance industrial assembly training. The system aims to accurately track and evaluate each assembly process, enabling it to determine the correct sequence of assembly and operation. By focusing on the application scenario of industrial assembly training, we can better define the components and functionalities of the system.

There are two main user groups for this system: trainers and trainees. The trainers are responsible for initially defining the correct assembly sequence through interaction with the system. They can then obtain various data from the system regarding the training process, such as any errors made by trainees during the assembly. This information allows trainers to evaluate the effectiveness of the training, providing opportunities to optimize the teaching process and adapt the training program based on the trainees' performance.

The second user group, the trainees, can practice assembly tasks within a virtual environment that offers a certain level of guidance and enables natural interaction with the various assemblies. This immersive virtual training environment aims to improve the trainees' understanding and proficiency in assembly tasks, ultimately enhancing their ability to perform these tasks in real-life situations.

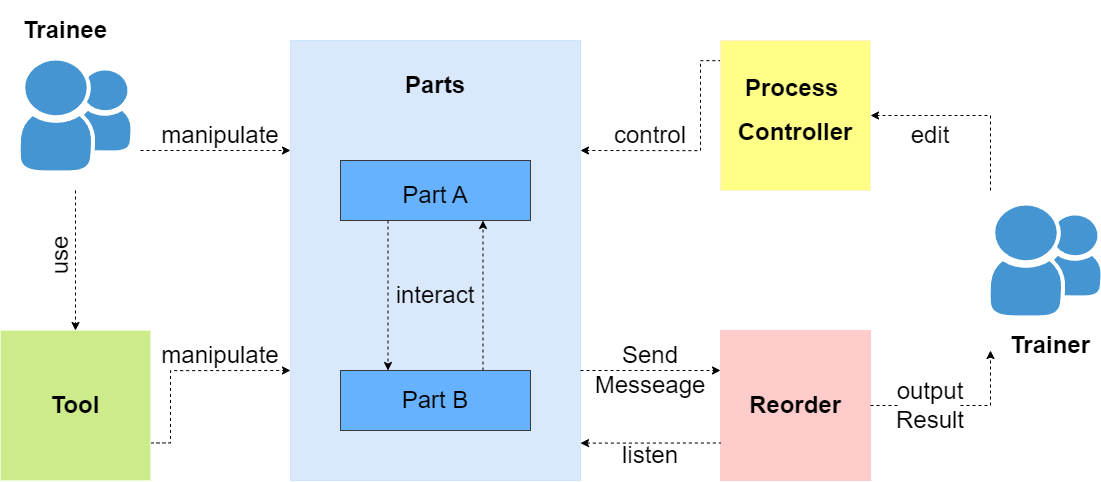


Figure 8 the graphic demonstration of assembly system

The graphic demonstration of system is presented in Figure 10. The main modules of the system include the Tool, Parts, Process Controller, and Recorder. These modules facilitate a range of interactions between the Parts, which are managed by users, both trainers and trainees. Each interaction, whether correct or incorrect, is recorded by the Recorder module and subsequently sent to the trainer for review and analysis.

Moreover, the trainer has the ability to define the correct order of assembly in the Process Controller module. This ensures that the Process Controller can effectively oversee the Parts throughout the assembly process, offering guidance and direction to trainees as needed. By allowing the trainer to input the appropriate assembly sequence, the system can tailor the training experience to suit the specific requirements of each assembly task, enhancing trainees' learning and skill development.

This thesis will investigate the design and implementation of these key modules, focusing on their functionalities, interactions, and underlying technologies.

### The Concept of Solution

The primary objective of this thesis is to develop an Interactive Assembly System in Virtual Reality (VR) for efficient virtual assembly training. To achieve this, we must first address two key challenges. The initial challenge involves creating assemblable assembly parts within the VR environment. Subsequently, the system must accurately identify the specific assembly task being executed, as well as determine its correctness. As such, the second challenge is defining the assembly task. In the following paragraphs, we will address each of these two challenges separately.

Addressing the first challenge, the proposed solution involves creating a socket-based connection between the two assembly parts. By representing the connection/socket between the parts as a line segment, we can effectively model the assembly process using a graph representation, as illustrated in the Figure 1. This connection graph can be partitioned by assigning the sockets to individual parts. As depicted in the Figure 2, the partitioned components can be leveraged to reconstruct the original relationships between the parts.

Ein Bild, das Diagramm enthält.

Automatisch generierte Beschreibung

Figure 9 Graph Representation of Assembly Parts Connection

Ein Bild, das Diagramm enthält.

Automatisch generierte Beschreibung

Figure 10 Partitioned Assembly Parts

Addressing the second challenge, the solution draws inspiration from Terje K. Lien's work ref, which breaks down an assembly task into three to four fundamental subtask types: identification and gripping of a part, transportation of the part to the placement or insertion position, execution of fitting or insertion, and, if necessary, securing the part. Following a similar rationale, we can assess the correctness of an assembly task by dividing the evaluation process into these four distinct steps.

The training system is designed to enhance users' learning experience by requiring them to acquire key information throughout the training task. To achieve this, a systematic approach was employed to identify and break down the essential information for each of the four steps involved in the task. The resulting diagram, presented below, effectively illustrates the key information that users should acquire at each step:

Table 3 the table of assembly subtask

|  |  |  |
| --- | --- | --- |
| **Subtask** |  | **Key information to be acquired** |
| Identification and gripping of a part |  | The correct assembly parts required to complete the assembly |
| Transportation of the part to the placement or insertion position |  | The correct preparation orientation and position for the following insertion |
| Execution of fitting or insertion |  | The correct orientation and position of the insertion |
| Securing the part |  | The correct of connecting staff and tool is applied for securing |

The implementation of each step within the Unity engine is a crucial aspect of our training system. The subsequent section delves into the specific methods and tools utilized to bring these steps to life, ensuring an immersive and interactive learning experience for users.

## Implementation

### Data Exchange

In situations where the complexity of the mesh is not particularly high and performance requirements are moderate, manual conversion of CAD data can be a cost-effective approach. In this thesis, all CAD data, including STEP files, were initially opened in AutoCAD. Subsequently, the data was manually exported to the VR environment and Unity environment as FBX files. This method of manual conversion helps strike a balance between cost and the desired level of detail and performance within the virtual assembly system. It is not necessary to utilize third-party or commercial software when there is no specific requirement for their use.

### Modules

In this section, we elucidate the implementation details of each module, offering a comprehensive explanation of their individual functionalities. We will discuss the design choices and methodologies utilized in the development of these modules, in order to provide a clear understanding of the system's architecture.

Finally, we will explore the interactions between modules and how they are integrated to form a cohesive system. By providing a thorough account of the implementation details, we aim to foster a deeper appreciation for the intricacies of the project.

#### Process Controller

The trainer can input the correct order of assembly into the Process Controller module. Each assembly step is defined as a list containing the IDs of two parts that should be assembled together in that specific step. As mentioned in the previous section, an assembly process always occurs between two parts. Connecting materials, such as screws and safety rings, do not play a role in defining the assembly process and are not considered in the list. These lists will be stored in the Process Controller as data in the form of a linked list.

The linked list data structure is utilized to represent the correct assembly sequence within the Process Controller module. The head of the linked list corresponds to the first assembly process, and each subsequent process in the correct sequence is connected to the previous one. This arrangement ensures that the assembly steps are linked together in the proper order, from beginning to end.

Initially, a pointer is set to point to the first ID in the first list, representing the starting point of the assembly process. As the system identifies that an assembly process has been successfully completed, the pointer advances and points to the next list, indicating the next assembly process to be performed. This dynamic management of the assembly sequence allows the Process Controller to effectively guide trainees through the assembly process while ensuring that each process is completed in the correct order.

It is crucial to emphasize that the order of elements within the list is significant. As previously discussed and in line with the Socket-based concept introduced earlier, an assembly process involves two parts: one being the main body with the socket, to which the other part is attached. To accurately represent this relationship in the list, a clear distinction between the two parts is necessary.

In the list, the first element represents the main body with the socket, while the second element corresponds to the part that will be fitted to the body. By maintaining this consistent order within the list, the Process Controller can effectively manage the assembly process, guiding trainees through each step while ensuring that the correct parts are being assembled in the appropriate manner.

Additionally, the process controller is responsible for managing the switching of sockets. Given the presence of multiple assembly parts within the scene simultaneously, opening all sockets for all assembly parts would result in considerable complications and superfluous calculations. To address this issue, we have refined the assemble process definition. Specifically, the socket is open only after the first step, during which the two parts are identified, in order to prepare the position for the subsequent step. This procedure is also overseen by the process controller.

The process controller has a component of collider, which restricts access to only the two parts in the list indicated by the pointer. In this setup, only the first part in the list is granted permission to open the socket, streamlining the process and minimizing any potential difficulties or computational overhead.

The process controller attains a significant level of control over the parts, primarily through the manipulation of the sockets. Various strategies exist for operating the switches, and the selection of an appropriate approach depends on the system's objectives. In the context of a training system, the conditions for opening the sockets need to be more stringent in order to ensure accurate learning outcomes.

However, when the system serves a different purpose, it may be feasible to have multiple switches open simultaneously. This approach offers greater flexibility and is more pertinent to scenarios where display and demonstration are the primary focus, as opposed to strict training requirements. By tailoring the socket operation strategy to the system's goals, the process controller can effectively adapt to diverse use cases and accommodate the specific needs of each situation.

#### Part

The part is the main module of the system. In the previous section on defining an assembly process, we split the assembly process into four small subtasks, the first of which is determined in conjunction with the process controller. The determination of the remaining three subtasks is entirely dependent on the interaction between the parts.

First, let's look at the second subtask: Transportation of the part to the placement or insertion position. The key information to be acquired in this subtask are the correct preparation orientation and position for the following insertion.

#### Recorder

# Results and Discussion

## User Study

- Purpose of user study( training? Immersion?usability?)

- what kind of questionnaire is used for which purpose

### Experiment Task

* Describe the assembly process

### Procedure

In this study, participants are randomly divided into two. Initially, all participants undergo a training session. Group 1 receives training using a VR system with guidance, while Group 2 learns the assembly process through reading instructions. After the training session, both groups attempt to assemble parts together in the VR system without guidance. The time taken and the number of errors are recorded for each participant. Finally, participants are asked to complete a questionnaire to gather their feedback and opinions on the VR system.

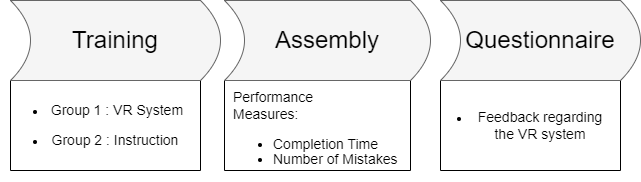


Figure 11 schematic process of user study

### Results and Discussion

# Conclusion

* Literaturverzeichnis

.N.iehorster, Diederick C.; Li, Li; Lappe, Markus (2017): The Accuracy and Precision of Position and Orientation Tracking in the HTC Vive Virtual Reality System for Scientific Research. In: *i-Perception* 8 (3), 2041669517708205. DOI: 10.1177/2041669517708205.

ARtillery Intelligence. (2023): Extended reality (XR) market size worldwide from 2021 to 2026 (in billion U.S. dollars) In Statista.

Bailenson, Jeremy; Blascovich, Jim; Guadagno, Rosanna (2008): Self‐Representations in Immersive Virtual Environments. In: *Journal of Applied Social Psychology* 38, S. 2673–2690. DOI: 10.1111/j.1559-1816.2008.00409.x.

Bamodu, Oluleke; Ye, Xu Ming (2013): Virtual Reality and Virtual Reality System Components. In: *AMR* 765-767, S. 1169–1172. DOI: 10.4028/www.scientific.net/AMR.765-767.1169.

Bartodziej, Christoph Jan (2017): The Concept Industry 4.0. Wiesbaden: Springer Fachmedien Wiesbaden.

Billinghurst, Mark; Clark, Adrian; Lee, Gun (2015): A Survey of Augmented Reality. In: *FNT in Human–Computer Interaction* 8 (2-3), S. 73–272. DOI: 10.1561/1100000049.

Bower, Matt; Howe, Catherine; McCredie, Nerida; Robinson, Austin; Grover, David (2014): Augmented reality in Education — Cases, places, and potentials. In: *Educational Media International* 51. DOI: 10.1080/09523987.2014.889400.

Chryssolouris, George; Mavrikios, Dimitris; Fragos, Dimitris; Karabatsou, Vassiliki (2000): Virtual reality-based experimentation environment for the verification of human-related factors in assembly processes. In: *Robotics and Computer-Integrated Manufacturing* 16, S. 267–276. DOI: 10.1016/S0736-5845(00)00013-2.

Coles, Timothy; Meglan, Dwight; John, Nigel (2011): The Role of Haptics in Medical Training Simulators: A Survey of the State of the Art. In: *IEEE T. Haptics* 4, S. 51–66. DOI: 10.1109/TOH.2010.19.

Coughlan, James M.; Biggs, Brandon; Rivière, Marc-Aurèle; Shen, Huiying (2020): An Audio-Based 3D Spatial Guidance AR System for Blind Users. In: *Computers helping people with special needs : … International Conference, ICCHP … : proceedings. International Conference on Computers Helping People with Special Needs* 12376, S. 475–484. DOI: 10.1007/978-3-030-58796-3\_55.

Desiderio Cano Porras; Petra Siemonsma; Rivka Inzelberg; Gabriel Zeilig; Meir Plotnik (2018): Advantages of virtual reality in the rehabilitation of balance and gait. In: *Neurology* 90 (22), S. 1017. DOI: 10.1212/WNL.0000000000005603.

G.R. Bennett: The application of virtual prototyping in the development of complex aerospace products.

Ganapathy, Apoorva (2016): Virtual Reality and Augmented Reality Driven Real Estate World to Buy Properties. In: *Asian j. humanity, art lit.* 3 (2), S. 137–146. DOI: 10.18034/ajhal.v3i2.567.

Jimoyiannis, A.: A systematic review of Virtual Reality in education.

Jones, Lynette; Sarter, Nadine (2008): Tactile Displays: Guidance for Their Design and Application. In: *Human factors* 50, S. 90–111. DOI: 10.1518/001872008X250638.

Lackner, James R.; DiZio, Paul (2005): Vestibular, proprioceptive, and haptic contributions to spatial orientation. In: *Annual review of psychology* 56, S. 115–147. DOI: 10.1146/annurev.psych.55.090902.142023.

Lasi, Heiner; Fettke, Peter; Kemper, Hans-Georg; Feld, Thomas; Hoffmann, Michael (2014): Industry 4.0. In: *Bus Inf Syst Eng* 6 (4), S. 239–242. DOI: 10.1007/s12599-014-0334-4.

LaValle, Steven M.; Yershova, Anna; Katsev, Max; Antonov, Michael (2014): Head tracking for the Oculus Rift. In: 2014 IEEE International Conference on Robotics and Automation (ICRA). 2014 IEEE International Conference on Robotics and Automation (ICRA). Hong Kong, China, 31.05.2014 - 07.06.2014: IEEE, S. 187–194.

LaViola, Joseph J.; Kruijff, Ernst; McMahan, Ryan P.; Bowman, Doug A.; Poupyrev, Ivan (2017): 3D user interfaces. Theory and practice. Second edition. Boston, Amsterdam, London: Addison-Wesley.

Lederman, S.; Klatzky, Roberta (2009): Haptic Perception: A Tutorial. In: *Attention, perception & psychophysics* 71, S. 1439–1459. DOI: 10.3758/APP.71.7.1439.

Lin, James; Duh, Henry; Parker, Don; Abi-RAched, Habib; Furness, Thomas (2002): Effects of field of view on presence, enjoyment, memory, andsimulator sickness in a virtual environment. In: *Proceedings - Virtual Reality Annual International Symposium*.

Morris, Dan; Tan, Hong; Barbagli, Federico; Chang, Timothy; Jr, John (2007): Haptic Feedback Enhances Force Skill Learning.

Pidel, Catlin; Ackermann, Philipp (2020): Collaboration in Virtual and Augmented Reality: A Systematic Overview. In: Lucio Tommaso de Paolis und Patrick Bourdot (Hg.): Augmented Reality, Virtual Reality, and Computer Graphics, Bd. 12242. Cham: Springer International Publishing (Lecture Notes in Computer Science), S. 141–156.

Piechowski, Sarah; Pustowalow, Willi; Arz, Michael; Rittweger, Jörn; Mulder, Edwin; Wolf, Oliver Tobias et al. (2020): Virtual reality as training aid for manual spacecraft docking. In: *Acta Astronautica* 177, S. 731–736. DOI: 10.1016/j.actaastro.2020.08.017.

Preutenborbeck, Michael; Usai, Marcel; Herzberger, Nicolas; Flemisch, Frank (2022): Multimodal Human Systems Exploration with Tangible XR for the Internet of Production: An expert survey. In: Production Management and Process Control. 13th International Conference on Applied Human Factors and Ergonomics (AHFE 2022), July 24-28, 2022: AHFE International (AHFE International).

Rainer Müller, Leenhard Hörauf, Max Eichenwald: Collaborative Virtual Reality System for Industrial based Assembly Training.

Riva, Giuseppe; Waterworth, John; Murray, Dianne (2014): Interacting with Presence: HCI and the Sense of Presence in Computer-mediated Environments.

Sanchez-Vives, Maria; Slater, Mel (2005): From presence to consciousness through virtual reality. In: *Nature reviews. Neuroscience* 6, S. 332–339. DOI: 10.1038/nrn1651.

Slater, Mel (2009): Place illusion and plausibility can lead to realistic behaviour in immersive virtual environments. In: *Philosophical transactions of the Royal Society of London. Series B, Biological sciences* 364 (1535), S. 3549–3557. DOI: 10.1098/rstb.2009.0138.

Slater, Mel; Wilbur, Sylvia (1997): A Framework for Immersive Virtual Environments (FIVE): Speculations on the Role of Presence in Virtual Environments. In: *Presence: Teleoperators & Virtual Environments* 6 (6), S. 603–616. DOI: 10.1162/pres.1997.6.6.603.

Stauffert, Jan-Philipp; Niebling, Florian; Latoschik, Marc Erich (2020): Latency and Cybersickness: Impact, Causes, and Measures. A Review. In: *Front. Virtual Real.* 1, Artikel 582204. DOI: 10.3389/frvir.2020.582204.

Steven, Marion (2018): Industrie 4.0 Grundlagen - Teilbereiche - Perspektiven. Stuttgart: W. Kohlhammer Verlag.

Sutherland (Hg.) (1968): Proceedings of the April 30--May 2, 1968, spring joint computer conference on - AFIPS '68 (Spring). the April 30--May 2, 1968, spring joint computer conference. Atlantic City, New Jersey, 30.04.1968 - 02.05.1968. New York, New York, USA: ACM Press.

Tam, Wa; Speranza, Filippo; Yano, Sumio; Shimono, Koichi; Ono, Hiroshi (2011): Stereoscopic 3D-TV: Visual comfort. In: *IEEE Transactions on Broadcasting* 57.

Zhu, Zhongcui; Du, Yan (2021): Research on Interior Design Optimization based on Virtual Reality Technology. In: *J. Phys.: Conf. Ser.* 1746 (1), S. 12063. DOI: 10.1088/1742-6596/1746/1/012063.