Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Bachelor Thesis

Nico Ahrens

An Augmented Reality Environment for an Industry 4.0 Plant

Fakultät Technik und Informatik
Department Informations- und Elektrotechnik

Faculty of Engineering and Computer Science
Department of Information and Electrical Engineering

Nico Ahrens

An Augmented Reality Environment for an Industry 4.0 Plant

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang Elektrotechnik und Informationstechnik
am Department Informations- und Elektrotechnik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr.-Ing. Florian Wenck
Zweitgutachter: Prof. Dr.-Ing JianQiang Shen

Abgegeben am July 1, 2019

Nico Ahrens

Thema der Bachelorarbeit

Eine Augmented Reality Umgebung für eine Industrie 4.0 Anlage

Stichworte

Augmented Reality, HoloLens, Industrie 4.0, Smart Factory, SPS

Kurzzusammenfassung

Die vorliegende Bachelorarbeit beschäftigt sich mit der Entwicklung einer Augmented Reality Anwendung auf einer Smart Glasses, durch die eine bereits vorhandene Industrie 4.0 Modelanlage erweitert wird. Dafür wird zunächst der Markt sondiert und anschließend die HoloLens von Microsoft als Anwendungsprodukt ausgewählt. Nachdem, abhängig vom Aufbau der Anlage, der Datenbank und den Möglichkeiten der HoloLens, die zu übertragenden Daten ausgewählt werden, wird eine Kommunikation zwischen Anlage und Datenbrille aufgebaut. Anschließend wird die Anwendung objektorientiert programmiert und angewendet. Die Bachelorarbeit ist für alle Leser geeignet, die Interesse an Augmented Reality im industriellen Kontext oder der Entwicklung einer Applikation für die HoloLens haben.

Nico Ahrens

Title of the paper

An Augmented Reality Environment for an Industry 4.0 Plant

Keywords

Augmented Reality, HoloLens, Industry 4.0, Smart Factory, PLC

Abstract

This bachelor thesis deals with the development of an augmented reality application on smart glasses, which extends an existing industry 4.0 model plant. First, the market for augmented reality glasses is analyzed and the Microsoft HoloLens is chosen. After selecting the data to be transmitted depending on the structure of the plant, the database and the possibilities of the HoloLens, a communication between the plant and the data glasses is established. The application is then programmed and applied in an object-oriented manner. The bachelor thesis is suitable for every reader who is interested in augmented reality in an industrial context or in developing an application for the HoloLens.

Acknowledgement

First of all, I would like to thank the HAW Hamburg and the USST for giving me the opportunity to spend a semester abroad, and further receiving a scholarship for the whole stay in Shanghai. This is a great honor for me and I will never forget it.
I would also like to thank the company Lucas-Nülle, which supported me financially, in order to make it easier for me to travel through China besides my work.
I thank my girlfriend Johanna, because she was always there for me even in difficult moments, notwithstanding the enormous geographical distance.
Last but not least, I would like to thank my parents who made my studies possible at all.

Contents

List of Figures

List of Tables

Listings

Directory of Abbreviations

VR	Virtual reality
AR	Augmented reality
HMD	Head mounted display
WoW	Window on the world
GPS	Global positioning system
DGPS	Differential GPS
SBAS	Satellite based augmentation system
IT	Information technology
CPS	Cyber physical systems
PLC	Programmable logic controllers
SCADA	Supervisory control and data acquisition
MES	Manufacturing execution system
ERP	Enterprise resource planning
RFID	Radio-frequency identification
SQL	Structured query language
PHP	Personal home page
UI	User interface

1 Introduction

1.1 Motivation

With the onset of the industrial revolution and the deployment of the first machines, the industry has changed dramatically. Today, we talk about industry 4.0 when it comes to modernizing production through information and communication technology by connecting people, machines and products [12].
The modern integration of data and information into mechanical processes makes production more efficient and more flexible. At the same time costs are reduced. To achieve this, not only the amount of data and information will increase in the future, but also the way of using this data and information will continue to evolve [9].
One component of industry 4.0 is augmented reality. Augmented reality is a technology that can be used to augment reality with virtual information. This extension of reality opens the possibility of simplifying production processes, results in easier and faster decision making, and ultimately results in increased productivity. Augmented reality can be used at many levels of production, from the customer to the factory worker.
With augmented reality the customer can look at products even before they are produced. Thus, the final products can be modified in advance according to the respective customer's needs. Tailor made products are the result which can in turn lead to an increased demand and production of products.
Assembly work can be greatly simplified in the future by, for example, displaying a manual to the worker with the help of augmented reality glasses. Errors during the assembly can thus be minimized. Less staff can take on more and different tasks. Also, costs for training and training of the staff can be reduced.
Workers could see real-time data of processes while they work on them. For example, the worker may be notified of a machine failure immediately after the error has appeared in real time. This saves time and therefore costs by allowing errors to be processed directly and without further information search.
Additionally, products may contain information about themselves. Processes can thus be accelerated and simplified by, for example, error workpieces showing the worker directly why it is an error workpiece.
Summing up, augmented reality has the ability to simplify processes and ultimately bring

production to the next level, both technically and economically.
The use and development of augmented reality in the industry is still in its infancy but it is a highly topical issue and will play a significant role in the evolution towards industry 4.0.

1.2 Goals

The aim of this thesis is to extend an existing industry 4.0 system with cloud infrastructure and communication through an augmented reality application on smart glasses.
This includes analyzing the market, comparing different hardware and selecting a product for the implementation of an augmented reality application based on these comparisons.
A selection of data will be made which shall be displayed to the user of the application.
A stable connection between the database and application needs to be established which always enables the query of the most current data.
Finally, the application is programmed and made available to the user on the smart glasses to test all chosen functionalities.

1.3 Structure

In *Chapter 2*, the reader of this paper is taught the basics of augmented reality, industry 4.0, the control technology and databases.
After explaining the technical basics, in *Chapter 3* the model plant, on which this work is carried out, is expounded.
In *Chapter 4* the work is conceptualized. The Smart glasses are selected and the requirements for the final application are also defined.
Chapter 5 deals with the product selected in Chapter 4, specifies the hardware and describes the possibilities of the technical usage of the product.
The programming is the base of *Chapter 6*. A database connection, the communication between the application and the database and the application itself is implemented.
Chapter 7 serves to document the results and checks whether the requirements mentioned in Chapter 4 can be implemented.
Finally, the results are summarized in *Chapter 8*. An outlook is also given.

2 Basics

2.1 Augmented Reality

Augmented Reality (AR) is gaining more and more importance both in the industry and as an application for private users. In this section, the basics of AR are presented.

2.1.1 Introduction

As the name suggests AR expands reality and is a variation of virtual reality (VR), in which the user dives in a completely virtual environment [5]. In VR the user looks through glasses and everything he sees, the environment and any objects, is created and does not exist in real life. In comparison to VR, AR lets the user see the real world overlayed with virtual objects. Thus, it mixes reality with virtuality. According to Paul Milgram and Fumio Kishino, mixed reality is a range between real environment and virtual environment, in which AR is close to the real environment (see Fig. 2.1) [24].

Figure 2.1: Mixed reality according to Paul Milgram and Fumio Kishino [24]

To realize AR three key points have to be processed: 1) Virtual objects will be created using a graphic software. In the end, the graphics created are played back on a display, for example, smart glasses, a smartphone or a conventional monitor [32, 10].
2) The point of view and even the position of objects have to be known at any time. AR mostly is not static. The user and objects may change their positions. The process of calculating

the positions of both user and objects is called tracking [32, 10].
3) In addition to the pure video output, the user can also interact with virtual created objects [32, 10].

2.1.2 Augmented Reality Systems

There are different systems to realize AR. These systems can be classified according to Milgram and Kishino [24].

Window on the World

The first system is called “Window on the world“ [11] (WoW). It is a simple Augmented Reality system, in which the user looks on a monitor. This monitor may be a computer monitor or the smartphone display of the user. At first, the environment is recorded. This record gets overlayed with virtual objects. The augmented environment is displayed on the monitor. Thus, the user is always an outsider who does not see his extended environment directly but through a “Window on the world“ [26].

Spatially Immersive Displays

Next to WoW, the spatially immersive system is another AR system. In this system the user is located between monitors or projection screens, which display virtual objects. Thus, the user is in his actual environment. Due to the fixed structure, the user is forced on site and a geographically independent system is therefore not possible [26].

Head Mounted Displays

Third, the head mounted displays (HMD) supplements the two previously mentioned systems. It offers the advantage of freedom of movement and geographical independence. The user uses the HMD to simultaneously record and present the user’s environment. “So in a sense, HMD is merely a layer between user’s senses and the environment in which the user is” [26]. There are two subsystems of HMDs: video and optical see-through. Video see-through is the same as WoW. After recording, the environment gets augmented. Both, virtually and real things get displayed via a display to the user. In optical see-through systems, the user does not see his environment recorded and reproduced by a monitor but sees

it as it is in reality. On the one hand, the environment does not need to be recorded, processed and reproduced. On the other hand, virtual edited objects are lagging behind the real view, contrast changes with lighting and virtual images are not completely stable [26].

2.1.3 Tracking

No matter which system is used, the tracking, that is the location of the viewer, of objects or the place where virtual objects should appear, is essential. There are different tracking systems based on different circumstances. The most important ones are listed and explained in this chapter.

Marker Tracking

The Marker Tracking is an optical tracking method in which markers (as illustrated in Fig. 2.2) are attached to objects.

Figure 2.2: Example of a marker used in ARToolKit, a position calculation library [13]

These markers are recognized by image processing programs. Thus, the position and orientation can be calculated [32].
For marker-based tracking, size and pattern must be known. There are procedures where the inner part of the pattern can be chosen arbitrarily and where the inner part is given. Basically, markers must be completely visible in the camera image in order to achieve successful tracking. If the markers are specified by the system and not freely selectable, the marker may still be recognized by redundancy if it is not completely in the camera picture. This allows a higher performance in tracking [10].
In addition to the correct size of the marker, the resolution of the camera, the distance and the

angle between the camera and the marker and the lighting situation are crucial for successful tracking [10].

Mobile Position Tracking

Besides, the mobile position tracking is another tracking option. For mobile outdoor AR applications, global positioning system (GPS) is a means of position determination. Deviations between ten or more meters are common. In forests, houses or even in cities between tall buildings, GPS-reception can be further reduced. Therefore, the use of conventional GPS is not suitable for AR [10].
Differential methods can significantly improve the accuracy of position determination. There are two different systems: Differential GPS (DGPS) and satellite based augmentation system (SBAS). DGPS calculates a correction signal based on the known position of a reference receiver. Thus, the accuracy can be improved up to a few centimeters. With the SBAS, the reference system is formed by several geostationary satellites, which can be problematic in cities due to the limited view of the reference system [10].

Sensor-based Mobile Orientation Tracking

Furthermore, sensor-based mobile orientation tracking is commonly used. The position estimation via sensors is used as a standard in the field of mobile devices such as smartphones or tablets. Usually a combination of magnetometers and inertial sensors, linear and rotary, is used [10].

Markerless Tracking

With markerless tracking, no additional markers are used. Only the object or its environment must be sufficient for tracking. There are many approaches to realizing this. For example, colors, shapes or differences in contrast e.g. to the environment can be used [32].
Another possibility is to use 3D models of the objects. These will first be calibrated. Afterwards the system can use them internally [32].
Another method to realize markerless tracking is to detect and map the environment by sensors and the camera system. A virtual created map of the real environment allows to place virtual objects in the real environment [27].

2.2 Industry 4.0

2.2.1 Introduction

Production companies are facing new challenges in terms of time constraints, quality and costs. These challenges are caused by the globalized market as well as the accompanied raised competition. The increasing number of competitors allows the customers to choose from different products and suppliers. Due to this development the demand for individual products increases. The cost and quality level should nevertheless remain constant. In order to remain competitive as a manufacturing company in high-wage countries and enable to get down to lot size one and produce individual customized products without extra cost, processes and methods have to be executed in a productive, efficient and flexible manner [33, 29].

2.2.2 Definition

To cope these challanges, the plan of Industry 4.0 has been presented at the Hannover Messe 2011. Industry 4.0 can be defined as "real time, intelligent and digital networking of people, equipment and objects for the management of business processes and value-creating networks" [34]. The new digital industrial technology known as industrie 4.0 is powered by technology improvements, which can be seen below in Fig. 2.3. One of these technologies is AR.

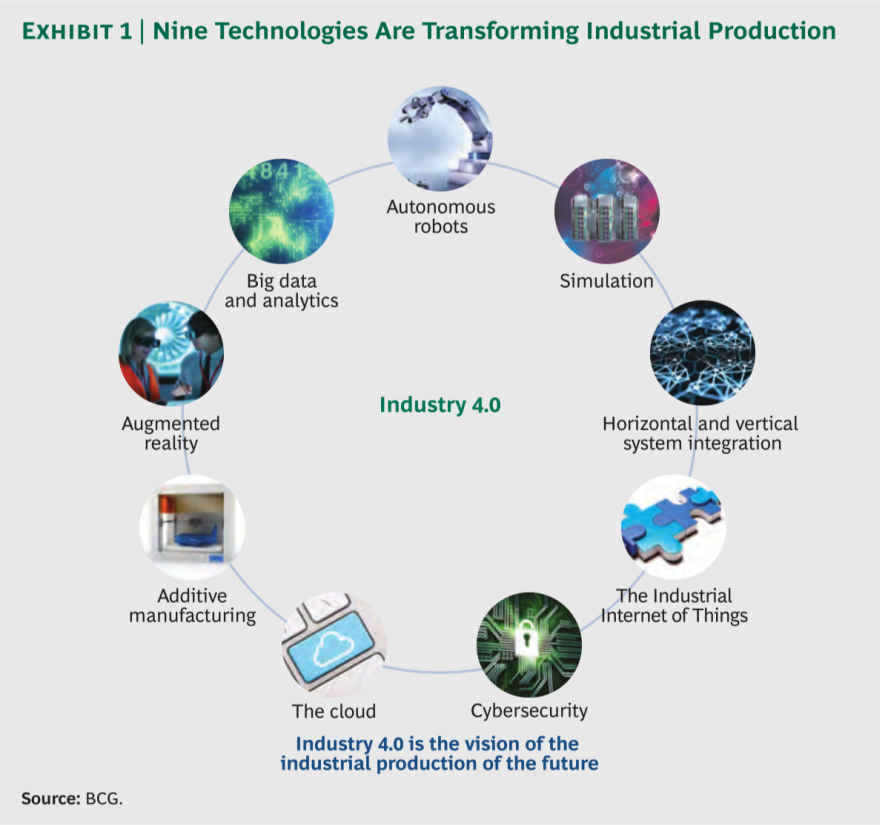


Figure 2.3: Industry 4.0 technology improvements [19]

In industry 4.0 anything like sensors, machines, workpieces and information technology (IT) systems can be connected. These things are not only connected internally to companies, but also via standard internet-based protocols across multiple companies. In industry 4.0, data is collected and analyzed across machines. This allows faster, more flexible and efficient production of goods. This increases quality while reducing costs. Industrial productivity will increase, the economy will shift and industrial growth will be promoted. The competitiveness of companies will change [19].
The nowadays isolated cells will get together as an optimized production flow. This will lead to greater efficiencies and change the relationships among suppliers, customers and producers, as well as the one between machine and human (see Fig. 2.4).

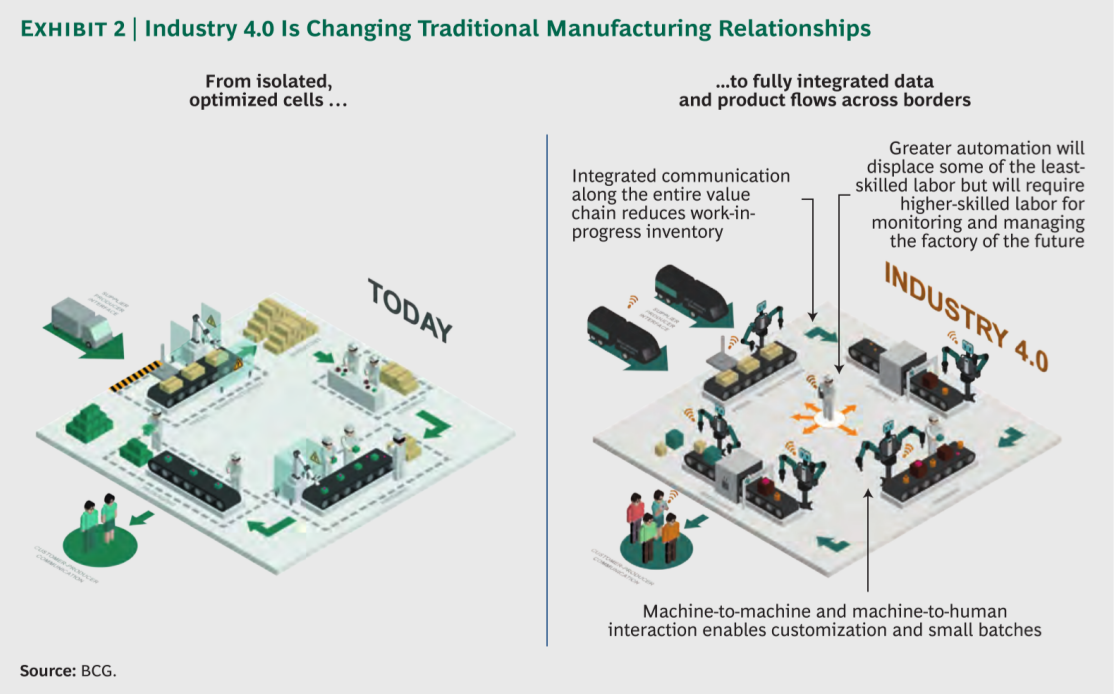


Figure 2.4: Connected future in industry 4.0 [19]

The different levels of automation in a factory can be portrayed as a pyramid, called the automation pyramid (see Fig. 2.5).

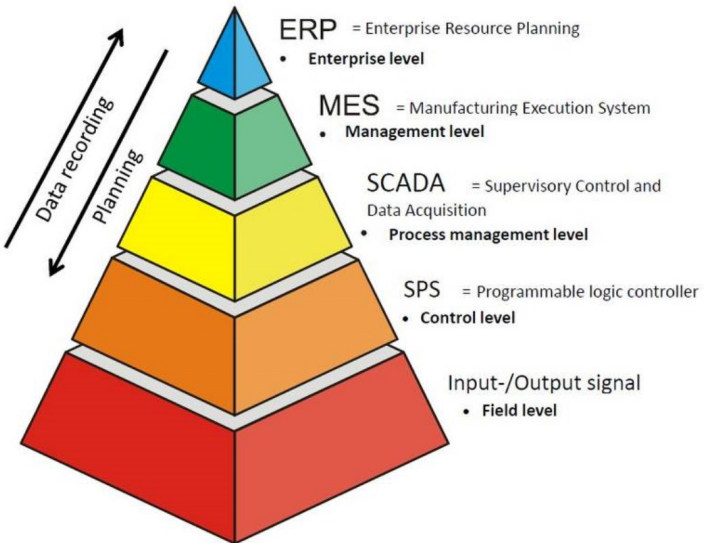


Figure 2.5: Automation pyramid [3]

At the bottom of the pyramid is the field level, which includes physical working devices like actors and sensors.
The next level is the control level. This is where programmable logic controllers (see Chapter 2.3.2) gather importance. They take the information of the sensors and control switches and actors. Depending on the programming task it switches outputs on or off [8, 25].
The process management level is the third level. This level utilizes the supervisory control and data acquisition (SCADA). SCADA is a combination of the lower level devices to get access to data and control systems from a single location [8, 25].
The fourth level of the automation pyramid is the planning level. This level uses a computer management system known as manufacturing execution system (MES). MES controls the entire manufacturing process in the factory from the raw material to the complete product. This allows management to make decisions according to the information they get from the MES about the manufacturing process [8, 25].
The top of the pyramid is called the management level. This level uses the enterprise resource planning system (ERP). This allows the management of a company to see and control their operations. ERP is usually a series of several computer applications which show everything progressing inside a company. This allows the company to watch all levels of the business from manufacturing to sales to buying to finance and payment [8, 25].
In industry 4.0, data, services and functions are held, retrieved and executed where they offer the greatest advantage in terms of flexible and efficient production. This will no longer necessarily be on the classic automation levels. The familiar automation pyramid is thus dissolved by the introduction of networked, decentralized systems. Services, data and hardware components can be distributed to arbitrary nodes of the resulting network and consequently form abstract functional modules from which the automation system is built. This dissolves the automation pyramid into an automation "cloud" [25] (see Fig. 2.6).

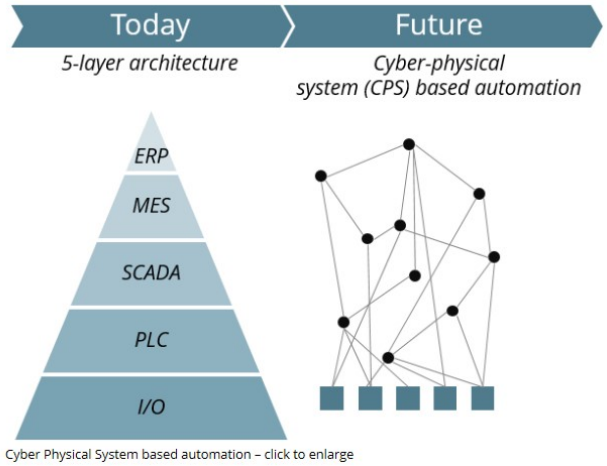


Figure 2.6: Automation cloud [17]

2.2.3 Smart Factory

Smart factory belongs to industry 4.0. It describes the process of reaching industry 4.0. In smart factory production facilities and logistics systems are self-organized. So there is no need for human intervention. The technical basis are cyber physical systems (CPS). CPS is a network of IT and software components that communicate, via the internet of things, a targeted networking of physical objects of the real world with the internet. In a smart factory, workpieces communicate with production facilities. The workpiece stores its manufacturing information. This information can be read out by machines [15].

2.2.4 AR within Industry 4.0

AR-based systems deliver many services, such as sending instructions over mobile devices. Even if this technique just starts to grow, in the future it will bring a benefit for many companys. Within AR, it is possible to work on a real actual system and get simultaneously virtual information about the system. The information can be displayed directly in the workers field of sight looking through devices such as smart glasses [19].

2.3 Control Technology

2.3.1 Introduction

Control and regulation technology are components of automation technology. The aim of automation is to influence processes in technical facilities in a way that they run independently. Thus, complex processes can be mastered, better economy, more safety and reliability can be achieved [35].
The goal of a control system is to generate a scheduled process flow by means of targeted temporal changes of process variables that are executed one after the other or in parallel.
Regulation systems keep process variables constant at specified nominal values despite disturbance variables or adapt them to variable reference variables [35].

2.3.2 Programmable Logic Controller

Programmable logic controllers (PLC) are automation devices, preferably for use in process and time-controlled sequence control systems. PLCs are programmed using Boolean switch-

ing formulas or a ladder diagram. The PLC core is connected to the process via the measuring and control devices and to the operating and monitoring devices via the input and output circuits in the input/output units. Depending on the complexity of the application, simple analog, binary and digital input/output signals and more complex interfaces with processing functions, e.g. counting and time functions, control functions are available [30].
Fieldbus systems are used to network several PLCs or possibly one with other automation and control devices [30]. A typical fieldbus is “Profibus” from Siemens.

2.4 Database

This chapter deals with the basics of databases. First, the term database is defined. After the definition, SQL and PHP are explained.

2.4.1 Definition

A database is a collection of data. These data strands are in logical relationships with each other. The necessity for always up-to-date data is enormous, since often several users read and write this data collection at the same time. The data collection is managed by a data base management system [28]. This is a software that builds and manages the database. The current database management systems are often also referred to simply as database systems [7]. A database system consists of a storage component that stores data in an organized form and a management component that provides a query and manipulation language. It also manages access and administration rights. Frequently, SQL databases (see Chapter 2.4.2) are used in practice [18].
There are four different forms of databases. The first database model is the object-oriented database in which data are organized as objects. Next to object-oriented databases, hierarchical databases exist and were commonly used in the past. They are constructed and organized as a tree structure. As changes to the database structure are not easy, the hierarchical database has been replaced by the model of network-like database. This third form of database is network-organized and thus more flexible, but the structure is also more complex. Consequently both, network-like and hierarchical databases, can no longer meet today’s requirements for a database as their structure is too complicated. The most often used form of databases today is the relational database, which consists of tables. Changes to the database are easy to perform due to the tabular form. Likewise, accesses to them are easy to program [28].

2.4.2 SQL

Structured query language (SQL) is an access language that is used for database programming. SQL is the most widely used programming language to query or interact with data from relational databases. As indicated before, the relational model expresses data in tabular form. Mathematically, a table is a set or subset. Therefore, SQL is a quantity-oriented access language, which always returns a subset of all data in the form of a table as a result of a query [18].
SQL is a descriptive query language. The desired results are defined by the terms “select”, “from” and “where”. An example of a SQL query would be:

```
1 SELECT 'name' FROM 'table' WHERE 'age' = 19
```

Listing 2.1: SQL example

The user does not have to make the calculations that lead to the result. This task is performed by the database system, depending on the input of the user [18].

2.4.3 PHP

Personal Home Page (PHP) is a server-side scripting language for accessing web servers [6], and database servers. Basically, the following steps are performed when accessing a database.
In order to be able to access the data of a database via PHP, the PHP script first establishes a connection to the database server. Afterwards, a connection to the database is made. The SQL code within the PHP script queries data from the database. These are then displayed in the web browser, via which the PHP script is called. Finally, the connection to the database server is terminated [6].

3 Plant and Process Description

This chapter describes the plant on which this work is performed. The structure and functions are explained. All information presented in the following comes from the project documentation [16].

3.1 Introduction

The model plant which is used for this thesis is an industry 4.0 plant. It is a smart factory in which people and machines communicate with each other. The plant also supports intelligent workpieces, manufacturing equipment and flexible value-adding procedures. The products contain information about themselves, such as the construction status, which can be read by machines. This data can be used to monitor the manufacturing processes of the products.

3.2 Components of the Plant

In the following the individual components of the system, that are networked to enable the features of an industry 4.0 system, are portrayed.

3.2.1 Workpieces

Depending on the order, the system produces a product consisting of up to three individual parts. For the end product, the system assembles a lower and upper part and, if applicable, a bolt (see Fig.3.1). The upper and lower parts of the final product can be selected by the user in two different colors. The bolt differs in material. After ordering, during the manufacturing process, the workpiece is transported on a workpiece carrier (see Fig.3.1) through the production process.

(a) Bottom section in black

(b) Top section in black

(c) Bolt in red

(d) Bottom section in white

(e) Top section in white

(f) Bolt in silver

Figure 3.1: Individual parts of the workpiece

Figure 3.2: Workpiece carrier

3.2.2 Dual Belt Conveyor Segments

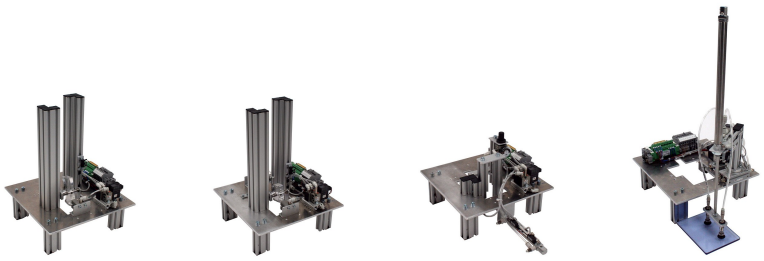
To transport the carriers with the workpieces through the production line, dual belt conveyors are used. Each conveyor is driven by a variable speed 24 V-geared motor. Finally, there are end position sensors at each end of the belts.

Figure 3.3: Dual belt conveyor

The motors of the belts are controlled by Siemens PLCs (see Chapter 3.2.5).
For the construction of the plant eight conveyor belts are used. They are linked through and combined with two curve segments with turntables. This combination results in a production circle (see Fig. 3.13). The curved segments are driven by the conveyor belts. Consequently, they are not directly controlled by the PLC.

Figure 3.4: 180° conveyor belt

3.2.3 Stations



(a) Sorting station (b) Assembly station (c) Processing station (d) Handling station

Figure 3.5: Fully automatic stations for assembly process

The stations, which are illustrated in Fig. 3.5, are connected to the conveyor belt and each station carries out a part of the assembly process. Station a) feeds and sorts the process with bottom sections of the workpiece. So does station b), but with the top section of the workpiece. Station c) is responsible for punching a bolt in the workpiece. Station d) transports the product from the carrier and places it on a storage place. In this place, the finished products are stored.

3.2.4 RFID

Radio-Frequency-Identification (RFID)-chips are data carriers that can store information [4]. To read and write information within the process a RFID evaluation unit is used, to which three read and write heads are connected. By using the RFID heads, position, color and the individual history in the previous production process can be written and read on the workpiece carrier. In Fig. 3.6 a RFID setup is shown.

(a) RFID interface unit

(b) RFID read and write head

Figure 3.6: RFID setup for read and write data

3.2.5 PLC

A “SCALANCE X-200” (see Fig. 3.7) offers the possibility of permanent monitoring of network components via signal contacts, web browser or “Profinet” diagnostics. This enables both network administrators and automatic systems to quickly identify and solve problems.

Figure 3.7: Scalance X-200

The plant is controlled by a PLC from the “SIMATIC S7-1500” series. The openly accessible profile rail is equipped with the input and output modules of the “SIMATIC S7-1500” series.

The PLC is linked to eight “SIMATICS S7-1200”s, each controlling a conveyor belt and a station (see Fig. 3.13). The programming of the PLC (see Fig. 3.8) is done via ethernet.

Figure 3.8: PLC

Moreover, a touch panel (see Fig. 3.9) is used to control and monitor machines and systems.

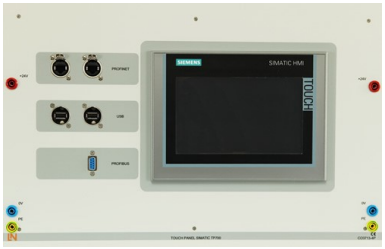


Figure 3.9: Touch panel

3.2.6 ERP-System

The ERP-Lab, the ERP model of the plant, serves, in addition to its actual purpose (see Chapter 2.2.2), also as a web shop, via which the user can place orders. The ERP communicates with the ERP database. Included in the ERP system is an MES, through which the manufacturing process can be monitored. This runs on a server and uploads all data relevant to the production process to the cloud. The ERP Lab handles all processes from ordering in the web shop, through the entire manufacturing process, to storage and product monitoring. The PLC that controls the system communicates with the ERP Lab. Ordering via the ERP system can be carried out from any mobile device in the same network that is capable of opening a web browser.
The ERP lab is designed as an edge cloud. Thus, the cloud functions in its own intranet without a connection to the Internet. The data storage for the ERP lab cloud is provided by a PC. The infrastructure is mapped by a Windows host PC with a virtualization software. The

cloud runtime, a virtual Linux PC, runs on the host.
The PLC communicates with the cloud via an interface (IoT adapter) and thus, has access to the services, logic and data structure of the ERP lab. The IoT adapter is connected to the online shop solution that manages the user interface (frontend) and order processing (back-end). The user has access to the frontend via a web server. All internet-enabled devices act as client devices (see Fig.3.10).

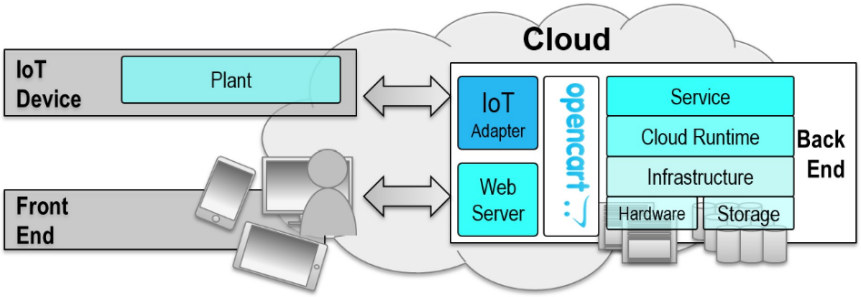


Figure 3.10: Communication of the plant

3.2.7 Testing Station

In addition to the actual production of the workpieces, the produced workpieces can also be checked for correct assembly. The components required for this check are discussed in this chapter.
During the workpiece inspection a Kuka robot (Fig. 3.11) grabs a finished workpiece from the belt and transports it to a workpiece carrier on a conveyor belt, outside of the production circle. The conveyor belt transports the workpiece to the testing station.

Figure 3.11: Kuka robot

The testing station (see Fig. 3.12) then checks the color and material of the workpiece to be tested.

Figure 3.12: Testing station

After checking the workpiece, the robot ejects the workpiece if it is incorrectly assembled. If the test is positive, meaning that the product is built correctly, the robot transports the workpiece back into the production process.

3.3 Manufacturing Process

All components described in this chapter are combined to form the production plant. How the production process works is explained in the following.
The user chooses on the touch panel whether he wants to add the testing procedure to the production process. In the web shop the user can place an order for workpieces. This order is processed in the ERP system. During the process, a carrier travels along the production line through the system. Depending on the colour of the upper and lower part of the currently produced workpiece, and whether a bolt is included, and if so, which material it is made of, the carrier stops at the appropriate stations. The information between the plant and the workpiece carrier is transmitted via the RFID read and write heads. Thus, station IMS3A or IMS3B, IMS4A or IMS4B and in case a bolt belongs to the ordered workpiece, station IMS5A or IMS5B are also approached by the carrier. If a test of the workpiece is carried out, the carrier stops between IMS5B and IMS7. The robot moves to the belt and grabs the workpiece and transports it on a carrier, which transports the workpiece to IMS 6 station (see Fig. 3.12). A test is performed at the IMS 6 station to determine whether the workpiece corresponds to the order or not. If the workpiece corresponds to the order, the conveyor belt and the robot transport the workpiece back to the carrier on the production line. The carrier then moves the workpiece to the IMS7. There, the workpiece is lifted out of the carrier by

the station and placed on a tray next to the system. The production of the workpiece is thus completed. If the inspection of the workpiece reveals that there is a production error, the robot sorts out the workpiece by placing it next to the IMS6 station. This means that the workpiece no longer enters the production process. Instead, the production process starts again and the process is repeated until the IMS6 inspection is positive. Only then IMS7 does complete the production process of the workpiece. Depending on whether the order contains further workpieces or not, the system produces further objects or switches to wait mode. The production process can be graphically traced (see Fig. 3.13).

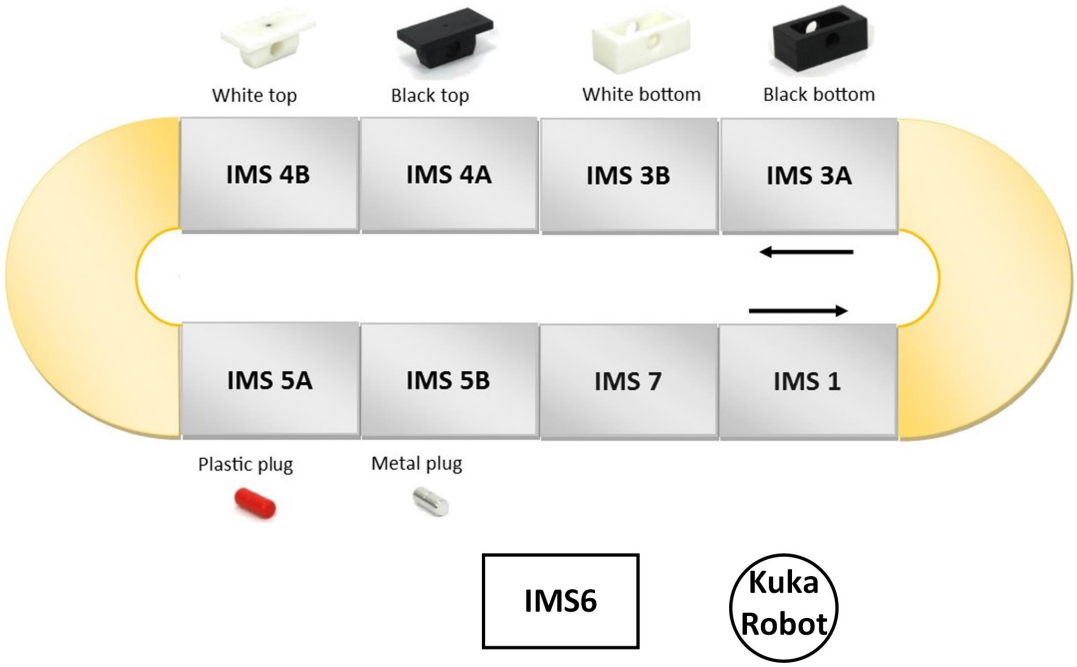


Figure 3.13: Production process [15]

4 Conception

In this chapter the system requirements are analysed and documented. Based on the analysis, a product is selected that best meets the requirements.

4.1 Requirements Analysis

In order to find the most suitable AR glasses for the industry 4.0 project, a requirements analysis is carried out. The requirements for the product are summarized in tabular form (see Table 4.1). The content of the table below is ordered by importance.

Table 4.1: Requirement table for smart glasses

Number	Requirement
1	Documentation and development environment in English
2	Glasses shall provide an opportunity for markerless Tracking
3	The hardware should not only meet the this requirements, but should also offer the possibility of further AR tasks.
4	Resolution
5	Transparent display

The most important requirement is that English documentation is available to collect information about programming.
The data glasses should allow markerless tracking, as this allows more flexibility and the application does not rely on external markers. The hardware of the data glasses should provide the opportunity to build on this work, so that further AR applications can be created within projects and theses. In addition, the resolution of the data glasses should be as high as possible and the display should be transparent.

4.2 Product Analysis

Four competing products are compared with each other and the most suitable product is selected on the basis of the requirements created (see Table 4.1).

Table 4.2: Comparison of the smart glasses with regard to requirements [1, 31, 14]

requirement	Moverio BT300	HiAR G100	Meta 2	Microsoft HoloLens
1	✓	-	✓	✓
2	-	✓	✓	✓
3	-	✓	✓	✓
4	1280 x 720 pixel	1280 x 720 pixel	2560 x 1440 pixel	1600 x 1200 pixel
5	-	✓	✓	✓

Starting with the Moverio BT 300, this is the simplest product. Detailed documentation of the product's functionalities is available. However, further requirements are not fulfilled. Markerless tracking is not possible. Interaction with objects is rarely possible, which means that further AR applications beyond this project are hard to find. In addition, the BT300 offers a semi-transparent display and just low resolution.
The Meta 2 passes all requirements. The glasses provide a markerless tracking and interacting with objects is possible. Thus, further tasks on AR can be executed. Compared to the other products, the glasses have the highest resolution. Additionally, English documentation and software is available. The Meta 2 was developed by a start-up, which is now insolvent. Thus, no support or updates can be offered.
HiAR G100 is a Chinese product. That is why there is no English documentation available. Since this requirement has the highest priority for the final product selection, the fact that all other requirements are fulfilled, can be neglected. Consequently, the product cannot be used for this work.
Looking at the table (see Table 4.1), it becomes obvious that the Microsoft's HoloLens is the only product, next to the Meta 2, that meets all the required characteristics of the table. Similar to the Meta 2, the Hololens provides markerless tracking. Likewise, with the glasses, the user can interact with objects. The resolution of the display, which is transparent, is satisfactorily high (1600x1200 pixel). The advantage of the HoloLens compared to the Meta 2 is that the manufacturer Microsoft offers a much more extensive English documentation. Also, Microsoft provides the product with updates. Following the analysis above, the Microsoft Hololens is chosen for reaching the aim of the work on hand.

4.3 Extracting AR Applications for the Plant

After choosing HoloLens as the hardware to work with, this chapter deals with the requirements for the AR application.

4.3.1 Basic Requirements for the Application

The AR application must read data from the database and display information on the HoloLens. This requires a stable and fast communication between the HoloLens and the database to ensure that the data are always up to date.
Further, the user should be able to use the application regardless of his position in the laboratory. The application offers the user a variety of data. Which of these data the user wants to see where in the environment and when, should be decided by himself. Thus, the tracking method should work without markers, as moving the displayed data would only be possible by moving the marker. The user should also be able to interact with the app. Virtual buttons should change color when being pressed. Further, there should also be virtual surfaces that give the user information as soon as he gazes it.

4.3.2 Choice of Data to be Displayed

Since the basic requirements for the Augmented Reality application are defined, in the following the data the user can view while using the application is selected and described. These are then listed in tabular form below (see Table 4.3).
Starting with the ordering process, the user should have the opportunity to receive information about the order. The ID number of the order and color of the workpiece should be displayed by the HoloLens. In addition, the entire order should be displayed and not only the currently produced workpiece. By this, the user knows how many workpieces follow the current one.
In addition, the application should deal with workpieces that have been sorted out by the robot because they have failed the workpiece inspection. Next to the ID, the user should be shown which color the workpiece should have. Consequently, he can compare the wrong assembled workpiece with the color it should have.
On the other hand, machine data should be shown. If the system stops production because of an error or empty stock, the user should get this information so that he has no longer to search for the error in the ERP system. The application should provide the user with the information which station is processing. The data to be displayed are listed and summarized in tabular form below (see Table 4.3).

Table 4.3: Data to be displayed in the Application

Information about	Data to show
Current order	ID and color value
Sorted out workpieces	ID and color value
Plant	Status of stations: error, maintainence or producing

5 HoloLens

In the last Chapter (4.2), Microsoft's HoloLens is chosen as the optimal product for this project. In the following the HoloLens is described in more detail. After the hardware is specified, the possibilities of using the HoloLens are outlined to emphasize the choice of the product (4.2).

5.1 Hardware

The HoloLens is a self-contained system which can be operated wirelessly. To achieve this, Microsoft has built an entire Windows 10 operating system into the headset, which uses an Intel 32 bit processor. The HoloLens works with many different sensors. An inertial measurement unit, an ambient light sensor, and four environment understanding cameras are used. In addition, a depth sensing camera is used to map spaces. The HoloLens uses a 2-megapixel HD camera to take videos and photos. Four microphones can be used to pick up the user's voice. HoloLens hardware specifications include 2GB of RAM, 64GB of flash storage, and Wi-Fi connectivity and Bluetooth, which allows the user to use a clicker instead of gestures (see Chapter 5.2.2) to control the HoloLens in applications more precisely. HoloLens has a battery life time of 2 to 3 hours, a standby time of two weeks. It is completely functional during charging and is passively cooled. Thus, no fan is in use needed [21, 20]. The Microsoft HoloLens is shown in Fig. 5.1 below.

Figure 5.1: Microsoft's HoloLens

5.2 Capabilities of the HoloLens

The HoloLens offers many possibilities to create an AR environment. This chapter is based on the Microsoft documentation [23].

5.2.1 Gaze

Gaze is a form of input. It tells where the user is actually looking in the world and lets the user determine their intention. In the real world one would usually look at the object, with which the person wants to interact. Gaze within HoloLens makes this connection happen in AR. The HoloLens uses the position and orientation of the user's head to figure out where he or she is looking at.

5.2.2 Gestures

Further, hand gestures enable the user to interact with elements, after targeting an object via gaze (see Chapter 5.2.1). The HoloLens provides the simplicity of interacting with elements without any other accessories, like e.g., markers.
The HoloLens is able to recognize different kinds of gestures, for example the "Air Tap" and "Bloom".
An "Air Tap" is a gesture similiar to a mouse click, and it is in fact included in most Hololens applications. This command is executed via a hand held upright. Besides to speaking "select" using voice command or using the clicker, the "Air Tap" is the way how to select anything. Fig. 5.2 shows the "Air Tap" during an application.

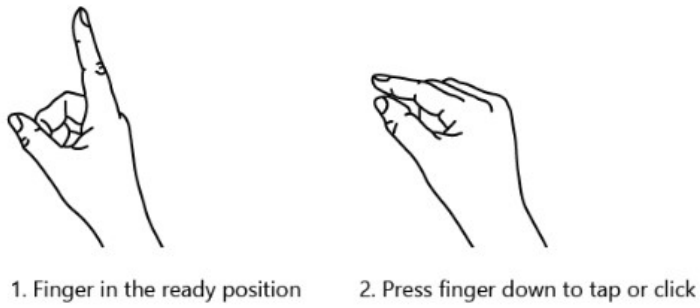


Figure 5.2: The "Air Tap" during an HoloLens application

The “Bloom” gesture, on the other hand, is the “home” gesture. It is used to go back to the start menu. It is equivalent to the Windows key on a keyboard.
To perform the bloom gesture, the hand must be hold out and palmed up with the fingertips together. After that the hand is opened (see Fig. 5.3).

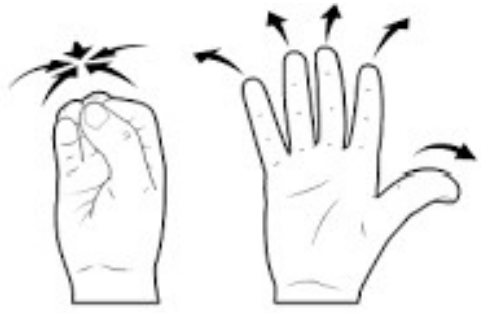


Figure 5.3: The “Bloom” gesture [2]

The “Tap and hold” gesture is like the “Air Tap”, but performed by maintaining the downward finger position. With this gesture things like drag and drop can be done.
Manipulation gestures can be used to move, resize or rotate a hologram. After targeting the element via gaze, “Tap and Hold” can start. Any manipulation of the element is then done by hand movements. Thus, this allows looking around during the gesture.
Navigation gestures can be used to navigate user interface (UI) widges, such as radial menus. The gesture starts with “Tap and Hold”, followed by moving the hand within a normalized 3D cube. Navigation can be used for building velocity-based continuous scrolling or zooming gestures.

5.2.3 Voice Input

Another way to communicate with the HoloLens is the voice command. The user simply gazes the object, he wants to interact with, and instead of making a gesture, he controls it via voice command. The voice command “select” is the same as an “Air Tap”. By saying “Hey Cortana” HoloLens allows the user to communicate with assistant Cortana, the voice of Windows 10, like on other devices like e.g., Windows 10 computers.
HoloLens provides the option to say the labels of buttons, instead of clicking this button. This works only for buttons in the menu bar.

5.2.4 Spatial Mapping

Spatial mapping authorizes the user to interact with virtual objects in the real world. Virtual objects can be placed on existing surfaces.
Furthermore, applications use the shape of real surfaces to place holograms in correct ways. For example, a virtual object cannot penetrate through walls. Spatial mapping is made possible by sensors that scan the room (see Chapter 2.1.3).

5.2.5 Spatial Sound

In order to notice what is happening outside the gaze, hearing is a sense that can be used. This is also possible in AR applications with the HoloLens. The HoloLens simulates 3D sounds using direction, distance and environmental simulations, which the developer can place in a three-dimensional space around the user. Thus, holograms can interact with the user even though they are not in his field of view.

5.2.6 Spatial Anchor

A spatial anchor is an important point in the map (see Chapter 5.2.4), that the system should keep track of over the whole time. Compared to other anchors or frames of reference it is needed to be adjusted to make sure that the anchored hologram stays at its place relative to the real world.

6 Programming

In order to create AR applications and play them with the HoloLens, preparatory steps must be taken first. The computer on which the application is developed has to be configured. The software must be installed and configured as well. This is outlined in this chapter.

6.1 Installation of the tools

At first, it is import to install or update the computer to the latest Windows 10 version, because it is necessary to get the most recent mixed reality features available [22].

6.1.1 Visual Studio 2017

Visual Studio 2017 needs to be installed. It is a fully integrated development environment for Windows. It is used to write codes, debug, test and deploy. By installing Visual studio, a software development kit is concurrently installed, which provides the headers, libraries, etc., which are necessary for deploying the HoloLens. Although a Visual Studio 2019 version exist, up to now, it is highly recommended to use Visual Studio 2017, because there are some issues for mixed reality developing in the newer version of Visual Studio [22].

6.1.2 Unity 2018.3

Unity is a game engine in which the developer can create holograms for the HoloLens. According to Microsoft, it is the easiest way to deploy apps for HoloLens or mixed reality in general. It provides built-in support for “Windows Mixed Reality” features. Thus, it is recommended to use Unity 2018.3.x [22].
During the installation process it is important to install some important components for the development for the HoloLens. If these components are not installed, applications can not be programmed for the HoloLens (see Fig. 6.1).

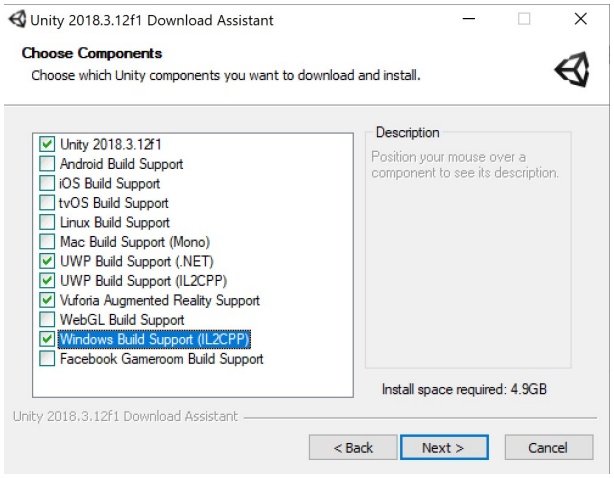


Figure 6.1: Important components to install for Unity

6.1.3 Mixed Reality Toolkit

The installation of the “Mixed Reality Toolkit” is completely optional.
Some components for the start of developing are provided and can get inserted to the project. This simplifies the development process for mixed reality and saves time. But all components can also be programmed by the developer himself.

6.2 Software Configuration

To program AR applications, Unity must first be configured. After starting Unity a new 3D project is created. As soon as the new project has opened, first the “Mixed Reality Toolkit” gets imported. This is added via the “Assets” menu (see Fig.6.2).

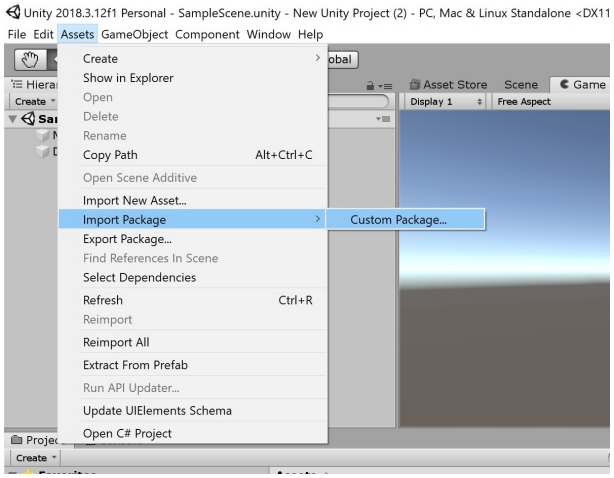


Figure 6.2: Importing the “HoloToolkit”

After selecting the HoloLens toolkit package, all components are imported (see Fig. 6.3).

Figure 6.3: Selection of components to import in “HoloToolkit”

After the packages are inserted, the “Mixed Reality Toolkit” in the upper menu bar is visible (see Fig. 6.4).

Figure 6.4: Menue bar after importing “HoloToolkit”

Since the “Mixed Reality Toolkit” is inserted, all settings that have to be made in order to be able to develop an application for the HoloLens are simplified (see Fig. 6.5).

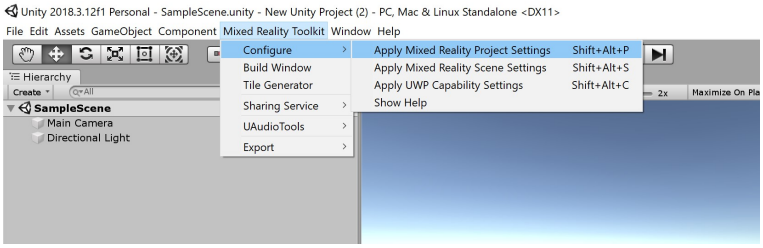


Figure 6.5: Setup of the project with “HoloToolkit”

The following settings are applied. The selection shown allows to program applications for the HoloLens (see Fig. 6.6).

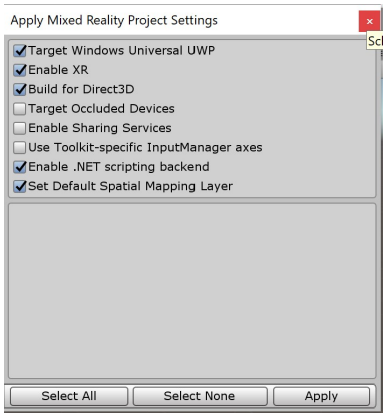


Figure 6.6: Apply the settings in “HoloToolkit”

All settings can also be made manually. Microsoft offers online instructions for this: `https://docs.microsoft.com/en-us/windows/mixed-reality/configure-unity-project`
After working through this chapter, the development of an AR application for the HoloLens can begin.

6.3 Basic Idea Programming

After installing and configuring the required software in the previous chapters, this chapter deals with the basic idea of how the application is implemented. The results are shown in a flow chart (Fig. 6.7) on which the programming is based in the following sections.
The application is object-oriented programmed in C#. The objects give the user information about the plant or a order. These information comes from the database. Due to the fact that each object provides different information to the user, each object communicates with the database individually.
In addition to communicating with the database, the user can also interact with the objects. According to the requirement of objects (see Chapter 4.3.1), the objects have no fixed positions. Therefore, the user can move and place any object everywhere he wants. This moving and placing should be performed by “Air Tap”. Likewise, the objects give the user sometimes information only after an interaction. This method of interaction should be the gaze of the user.
The basic idea of programming the app is presented in a flow chart on the basis of an object. In this case, an object is selected, which reproduces the data only after interacting with the user.

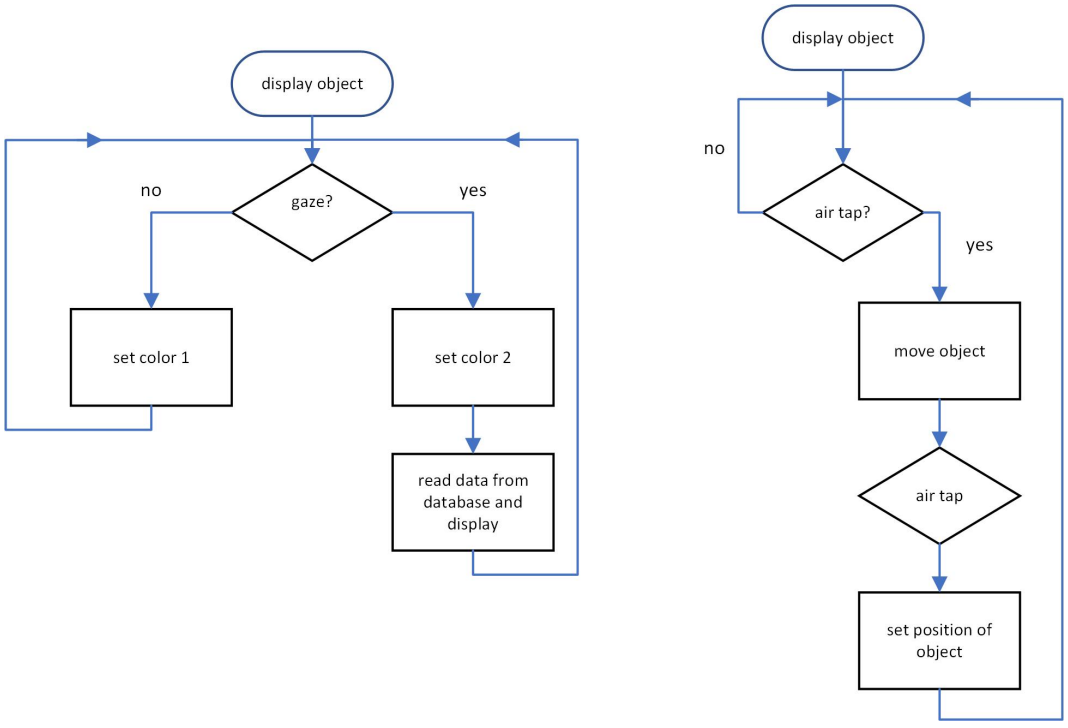


Figure 6.7: flow chart of basic procedure

6.4 Communication with the Database

As already mentioned, the application is built object-oriented. Each object which should show information to the user in the application must therefore communicate with the database. The communication procedure to the database is explained in this chapter.
Referring to the plant description (Chapter 3), the ERP system reads and writes data from a database. This database is installed on the virtual machine of the lab computer. To manage access or organization of the database via the web, the software “phpMyAdmin” is installed on the virtual machine.
As discussed in the basic chapter, PHP offers the possibility to get data from a database via a web browser. To do this, the device and the database must be clients in the same network. Similarly, the name and access data of the database server and database must be known. How to do this in a PHP script is demonstrated in the following program extract.

```
1  $servername = "localhost";
2  $username = "mes";
3  $password = "gx=!1p5ui7)ap4&)";
4  $dbname = "mes";
5
6  //create connection
7  $conn = new mysqli($servername, $username, $password, $dbname);
8
9  //check connection
10 if ($conn->connect_error){
11 die("Connection Failed: " .$conn->connect_error);
}12
```

Listing 6.1: Connection to the database in a PHP skript part 1

If the connection to the database is successful, the results of a SQL query can be stored in a variable. In the following example, data from the MES database is read into a variable. The data from the database is displayed in the web browser. The elements are separated by a "|". The insertion of the character between the elements allows further processing of the data during the program, which will be explained later. If the SQL query does not return results from the database, "0 results" will be displayed in the web browser.

```
1  // get information from database
2  $sql = "SELECT is_production, is_maintainence, is_error FROM
       factory_segment WHERE id = 8";
3  $result = $conn->query($sql);
4  if ($result->num_rows > 0){
5      //output data of each row
```

```
6       while($row = $result->fetch_assoc()){
7       //print data read from database
8       echo $row["is_production"]."|".$row["is_maintainence"]."|".$row["
            is_error"]."|";
9       }
10  }
11  else {
12      echo "0 results"; // if nothing read from database
13  }
14  // close connection
15  $conn->close();
```

Listing 6.2: Connection to the database in a PHP skript part 2

The PHP scripts are stored on the virtual machine in the following file path:

/usr/share/phpmyadmin/BachelorarbeitNico/

By entering the URL

"http://192.168.2.177/phpmyadmin/BachelorarbeitNico/
NameOfTheSript.php/"

in the web browser, the script can be called up from any client that is in the laboratory network.

6.5 Basics of an App for the HoloLens

As already known from the HoloLens chapter (see Chapter 5), the HoloLens offers the possibility of spatial mapping. The user can target real objects or virtual objects by his gaze. A cursor follows his gaze and interacts with the surfaces. For example, when the user looks at a chair in the room, the cursor is on the surface of the chair. The Hololens also offers the possibility to interact with virtual objects by clicking on them, during a gaze.
These skills must be programmed in Unity before the user can use them in an application. The “Mixed Reality Toolkit” is helpful here. It provides the user with prefabs that can be added to the project. These prefabs contain scripts, which allow the user to use the HoloLens skills in an app. Which prefabs are added to the project at hand is explained below.
First, all objects created by Unity are deleted from the “Hierarchy” window. Thus, the project is empty. The following prefabs from the “HoloToolkit” folder under “Assets” are added in the “Project” window (see Fig. 6.8):

Figure 6.8: Hierachry panel after including the mixed reality toolkit prefabs

In the “Inspector” panel of the “Input Manager”, the prefab “Cursor” from the “Hierarchy” panel is added to cursor in the “Single Pointer Selector” script. This is done via drag and drop.
The room manager will now be assigned a room. Using the HoloLens, it continuously stores data from the environment in which it is used. This room data are imported. The room data are located under “3DView“ in the device portal, which can be reached by calling up the IP address of the HoloLens in the web browser.
The room object is stored under Unity in the “Assets” folder. It is assigned to the “Object Surface Observer” script under room model using drag and drop in the “Inspector” panel of the “Spatial Mapping” prefab.

6.6 Air Tap

To allow the user to move an object, the object is assigned the script “Tap To Place” by drag and drop. During the user gazes an object he can “Air Tap”. Then the object is detached from its position. The object now follows the user’s gaze. With another “Air Tap”, the object is assigned the current position and anchored there. The anchor is persistent. Thus, even if the application is closed the object will remember its position after the next start of the app. The script accesses the “Spatial Mapping” prefab. It thus detects real surfaces and allows the object to be placed on real surfaces, like desks or chairs. This script is provided from the “HoloToolkit”.

6.7 Creating Material for Objects

3D game objects used for AR applications with the HoloLens are optically defined by their scale, position and rotation. In addition, every 3D object, which is not a text, for example, but a geometric object, such as a cube or a sphere, consists of a material.

A material of a surface determines how it is rendered. Properties, such as the color, or rounding of the edges of the object, are determined here. The properties of a material which users can change are determined by the shader. A shader contains the calculations of the pixels rendered, based on the material and the lighting input.
The materials created will be shown in the course of this work.

6.8 Programming the AR objects

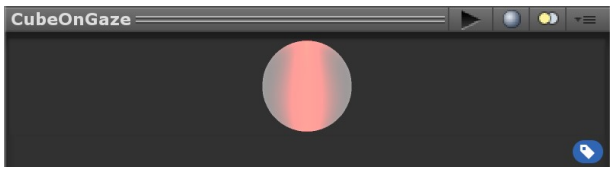
This chapter deals with the virtual objects which the user sees in the application. These are created and programmed to meet the essential requirements discussed above (see Chapter 4.3.2).

6.8.1 Current Order

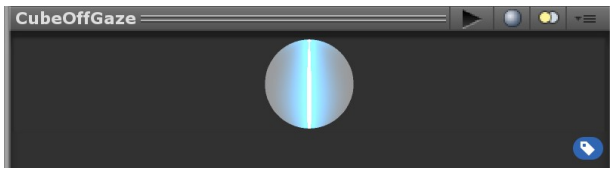
According to the requirement analysis (see Chapter 4.3.2), the app should provide the possibility to get information about the current order. The order ID and the object properties of all objects contained in the current order should be displayed.

Transform the Object and Assign a Material

In the “Hierarchy” panel, a 3D object, a cube, is created. Two materials, “CubeOffGaze” and “CubeOnGaze“ (see Fig. 6.9), are created.



(a) “CubeOnGaze” material



(b) “CubeOffGaze” material

Figure 6.9: Materials of “Current Order”

Depending on the situation, the game object should be rendered with one of these two materials. First the material “CubeOffGaze” is assigned to the object by drag and drop. In the later course of this chapter, changing the material depending on the situation will be implemented. The following transformation (see Fig. 6.10) is chosen for the cube, with the position chosen arbitrarily. The only important thing is that the object is placed near the position of the camera (0,0,0) so that it is close to the user when the app is started for the first time. While using the app the position will change. Crucial, however, is the rotation and the scale.

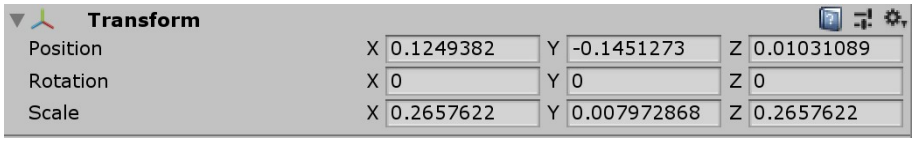


Figure 6.10: Transformation of current order object

Implementing the Ability to Interacting with Objects

The object should offer the user the following possibilities of interaction: The user should be able to move the object. In addition, the user should receive information about the order when gazing the object.
To give the user the ability to move the object, the “Tap To Place” script from the HoloToolkit is added to “Current Order”.
In order to obtain information from the object via gaze, the script “Interactable” is written.
The game objects “Description” and “Message” are declared. The materials “OffGaze” and “OnGaze” are also declared. All declarations are public. This type of protection makes it possible to hand over two game objects and two materials to the script.

```
1  using HoloToolkit.Unity.InputModule;
2  using UnityEngine;
3
4  public class Interactible : MonoBehaviour, IFocusable
5  {
6      public GameObject Description;
7      public GameObject Message;
8      public Material OffGaze;
9      public Material OnGaze;
```

Listing 6.3: Make object interactable part 1

Both, materials and objects can then be used in the script. The script uses the two functions “OnFocusEnter” and “OnFocusExit”. These two functions come from the interface “IFocusable” from the “HoloToolkit”, from which this script inherits. “OnFocusEnter” is called as soon as the user gazes “Current Order”. In this function, the game object “Current Order Message” is set as active. This makes it visible to the user. The game object “Current Order Title” will be disabled. So it becomes invisible to the user. In addition, in this function, the material of “Current Order” is set as the material “OnGaze”.
The “OnFocusExit” function is called as soon as the user leaves “Current Order”. In this function, “Current Order Message” is deactivated and “Current Order Title” is activated. Likewise, the material “OffGaze” is loaded.

```
1  void IFocusable.OnFocusEnter()
2  {
3      // hide title text
4      Description.SetActive(false);
5      // show data from database
6      Message.SetActive(true);
7      // set material of the object to "OnGaze"
8      Renderer rend = GetComponent<Renderer>();
9      rend.material = OnGaze;
10 }
11
12 void IFocusable.OnFocusExit()
13 {
14     // show title text
15     Description.SetActive(true);
16     // hide data from database
17     Message.SetActive(false);
18     // set material of the object to "OffGaze"
19     Renderer rend = GetComponent<Renderer>();
20     rend.material = OffGaze;
21 }
```

Listing 6.4: Make object interactable part 2

The two materials “CubeOnGaze” and “CubeOffGaze” are handed to the script in the “Inspector” panel. The game objects handed to the “Interactable” script will be discussed in the next parts.

Current Order Title

“Current Order Title” is a 3D text game object. The text, which is displayed, is “Gaze to get information about the current order”. The position of the text is above “Current Order”.

“Current Order Title” is passed to “Interactable”. Thus, the text is readable to the user of the application as long as he does not gaze “Current Order”.

Current Order Message

The game object “Current Order Message”, is like the game object “Current Order Title”, passed to the “Interactable” Script in “Current Order”. Thus, it is activated as soon as the user gazes “Current Order” and only until he averts his gaze again.
“Current Order Message” shows the user, if activated, the order ID and color values of the current order. Since an order can contain several workpieces, the output of the data from the database must be adapted to the length of the data to be displayed. Since the information from the database is displayed only when the user is viewing “Current Order”, the user’s view is limited while reading the information. For a long text, the user may not see all the text in the display, depending on his position. If he then changes his position or his gaze to read the text that is outside his view, “Current Order” might no longer be gazed. Thus, no text would be displayed. In addition, the position of the text must be flexible depending on the length of the text, since the end of the text should always be above “Current Order”.
To read data from the database and then display it dynamically in terms of position and text size, the script “Get Current Product“ is used.
Four 3D text game objects are handed over to the script. In these, the read data are stored at the end. A list of strings called “names” is also declared.

```
1  public class GetCurrentProduct : MonoBehaviour
2  {
3      private List<string> names = new List<string>();
4      public TextMesh shortText;
5      public TextMesh mediumText;
6      public TextMesh longText;
7      public TextMesh minimumText;
```

Listing 6.5: Get current product code part 1

In “ReadingName()”, the PHP script on the database server first accesses the database and performs a SQL query, to get the order ID and the color values of the workpieces.

```
1  SELECT order_id, keywords FROM `factory_order` WHERE done = 0
2  ORDER BY id DESC
```

Listing 6.6: Get current product SQL code

“ReadingName()” is a coroutine, which provides the possibility to pause the script via “yield return”. “yield return www.send ();“ pauses the script until the query from the web request

is finished. Only after finishing the request, the script will continue to run. If no connection can be established, the script ends. If the connection is successful, the data read are stored in the string “dataStream”. In the PHP script, all the data read by the SQL query from the database are strung together, separated from a “|” character. The split function divides the string between each “|” and stores each item in “names”. The data structure list is chosen because the length of the list is dynamic, as is the amount of data from the database that can change at any time. At the end of this coroutine, all data from the database are stored individually in the list “names”.

```
1  IEnumerator ReadingName()
2  {
3      string dataStream;
4
5      using (UnityWebRequest www = UnityWebRequest.Get("192.168.2.177/
           phpmyadmin/BachelorarbeitNico/GetActualProduct.php"))
6          {
7          // waint until read from database
8          yield return www.Send();
9
10         if (www.isNetworkError || www.isHttpError)
11         {
12             Debug.Log(www.error);
13         }
14
15         else
16         {
17             //safe information from databank in dataStream
18             dataStream = www.downloadHandler.text;
19             //split the string by | and store data in a list
20             names = dataStream.Split('|').ToList<string>();
21             //prepar list for further usage
22             names.Remove(names.Last());
23         }
24     }
25 }
```

Listing 6.7: Get current product code part 2

The “Update()” function is called every 20ms. In this function the coroutine “ReadingName()” is called. As a result, the data is queried from the database every 20ms. All items in the list “names” are stored in a string “namesBuffor”. The “namesBuffor.Length” function checks the length of the string. Depending on the length, the data is stored in one of the 3D text game objects, while the contents of the other 3D text game objects are deleted.

```
1  void Update()
2  {
3      StartCoroutine(ReadingName());
4      // reset Buffor
5      string namesBuffor = "";
6      // if elementes in list < 0 (data read from databank)
7      if (names.Count() != 0)
8      {
9          //prepar data output
10         foreach (string name in names)
11         {
12         namesBuffor = namesBuffor + name + "\n";
13         }
14         // case length of datastream is short
15         if (namesBuffor.Length > 130)
16         {
17         shortText.text = namesBuffor;
18         minimumText.text = null;
19         mediumText.text = null;
20         longText.text = null;
21         }
22         // case length of datastream is medium
23         if (namesBuffor.Length > 265)
24         {
25         shortText.text = null;
26         minimumText.text = null;
27         mediumText.text = namesBuffor;
28         longText.text = null;
29         }
30         // case length of datastream is long
31         if (namesBuffor.Length > 400)
32         {
33         shortText.text = null;
34         minimumText.text = null;
35         mediumText.text = null;
36         longText.text = namesBuffor;
37         }
38     }
39     // if elementes in list = 0 (no data read from databank): case
           length of datastream very short
40     else
41     {
42         shortText.text = null;
43         mediumText.text = null;
```

```
44            longText.text = null;
45            minimumText.text = "plant does not produce";
46        }
47    }
```

Listing 6.8: Get current product code part 3

Create the Text Format for Output

Four 3D text game objects, “Short Text”, “Medium Text”, “Long Text” and “Minimum Text” are created. These text objects define the format of the displayed data from the database. The position of “Long Text” for example is higher than that of “Medium Text”, since its text content is longer. The end of all text streams, regardless of the length of the text, are supposed to end above “Current Order”. Likewise, the font size becomes smaller with the length of the content. The four game objects are handed over to the “Get Current Order” script, in which their text gets defined.
The 3D text objects are subordinated to “Current Order”. Thus, their position change as soon as the position of “Current Order” changes. The position of the text objects is always above “Current Order”.

6.8.2 Stations

According to the requirement analyses (see Chapter 4.2), the user should get information about the status of the stations. First, the status “processing”, that is, if a station is currently producing, second , “maintenance” , for example, when the station is out of stock, or third, “error”, for example, if no carrier gets to the station, should be displayed to the user. Otherwise the stations have the status “waiting”. These states are signaled to the user via color of a sphere: green for “processing”, yellow for “maintenance”, red for “error” and “grey” for anything else. For each IMS station there is a ball which can change the color according to this color-coding, with the name of the IMS station above it. The balls can be moved by the user of the app.

Transform the Object and Assign a Material

The ball is a 3D sphere game object. Like all other game objects, the rotation and scale of transformation (see Fig. 6.11) is important as they determine the shape of the object. The position, on the other hand, is not particularly important because it can change constantly when using the app.

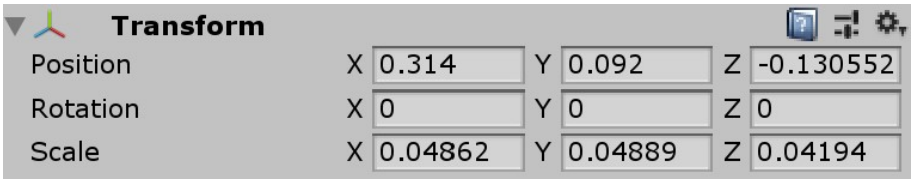
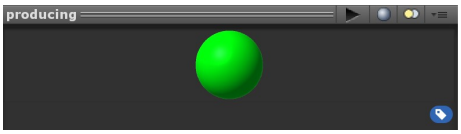
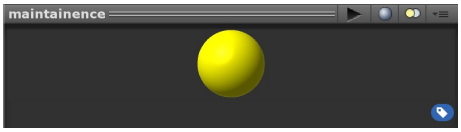


Figure 6.11: Transformation of IMS1

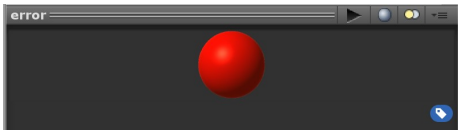
The four states of an IMS station are signaled to the user by the color of its ball. For this purpose, four materials are created (see Fig. 6.12), which are assigned to the ball depending on the status.



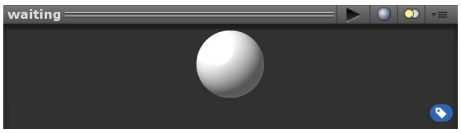
(a) Producing material



(b) Maintainence material



(c) Error material



(d) Waiting material

Figure 6.12: Materials of “IMS1” and all other stations

Implementing the Ability to Interact

To allow the user in the application to move the IMS stations, the “Tap To Place” script is added to each IMS station game object.
In order to get the status of a station from the database and to set the material adapted to the status, a script is added to each IMS station game object. The script “IMS 1” for the game object IMS 1 is explained below, representing all scripts for all IMS stations. In the first part of the script, four materials are declared. These are declared as public and allow the four materials of the sphere (see Fig. 6.12) to be handed over to the script from the “Inspector” panel. A list of the data typ string is also declared, in which the read data from the database is stored during the course of the script.

```
1  public class GetActualProduct : MonoBehaviour
2  {
3      // create list for store information from database
4      private List<string> listFromDatabase = new List<string>();
```

```
5
6       //variables for different colours
7       public Material waiting;
8       public Material producing;
9       public Material maintainence;
10      public Material error;
11  }
```

Listing 6.9: IMS1 script part 1

The coroutine “ReadingName()” is called from the “Update()” function every 20ms. It reads the data out of the database and stores it in a list. This is done in exactly the same way as with “Current Order”, and an explanation for this process can be read in more detail in the Chapter 6.8.1. The SQL code in the PHP script, which is called “in ReadingName()”, queries all states of the respective IMS station. The “id” in the code filters according to the respective station. In this case, the “id = 8” filters the information for IMS 1. For example, the “id = 7” filters the information for IMS 3A. The difference between the scripts of the different IMS stations is only the PHP script, and thus, also the SQL code (see listing 6.10).

```
1  SELECT is_production, is_maintainence, is_error FROM factory_segment
2  WHERE id = 8
```

Listing 6.10: IMS1 SQL query in PHP script

Depending on which status is read out of the database, the material of the ball is set in the following code example (see listing 6.11). Should a station have the status “maintenance” and “error” at the same time, the ball will load the material “error”. Should the station produce and simultaneously be in maintenance, the ball would get the material “maintenance”. If there is no status, the station waits and the ball gets the material “waiting” (see listing 6.11).

```
1  Renderer rend = GetComponent<Renderer>();
2  // if station has an error -> set color of cube to red : error
3  if (listFromDatabase[2] == "1") rend.material = error;
4  // if station is maintainence but not error -> set color of cube to
       yellow : maintainence
5  else if (listFromDatabase[1] == "1" && listFromDatabase[2] != "1")
       rend.material = maintainence;
6  // if station is producing but not maintainence -> set color of cube
       to green : producing
7  else if (listFromDatabase[0] == "1" && listFromDatabase[1] != "1")
       rend.material = producing;
8  // else the plant is waiting -> set color of cube to grey
9  else rend.material = waiting;
```

Listing 6.11: IMS1 script part 2

IMS Station Text

Each ball is assigned a text with the name of the station. This text is realized by adding a 3D text game object to each IMS station. The position of the game object is above the respective station. The game object is subordinated to the station by drag and drop. So, the position depends on the location of the station and changes as soon as the IMS station is moved via “Tap To Place”.

6.8.3 Error Workpieces

As mentioned in the requirement analysis (see Chapter 4.3.2) the app should display the order ID and color values of the workpieces, which are taken from the production process by the robot for testing purposes.
The robot can sort out up to three workpieces and place them side by side on a storage place. On the panel of the plant, a button is implemented, which resets the counter of the rejected workpieces. This button is also implemented in the AR application. So that the user, in addition to the button on the panel, must press the virtual button in the app to reset the storage of out sorted workpieces. After resetting, the storage ID and color values of the next three out sorted workpieces can be displayed. The button changes its color to blue as soon as at least one workpiece is sorted out. Thus, the user knows whether the counter is reseted or not.
The information about the workpieces and the reset button can be moved by the user of the app.

Transform the Object and Assign a Material

The error workpieces are represented by three 3D sphere game objects, called “Error Workpiece”. Their position is not interesting because the objects can be moved in the app at any time. Their scale and rotation is shown in Fig. 6.13.

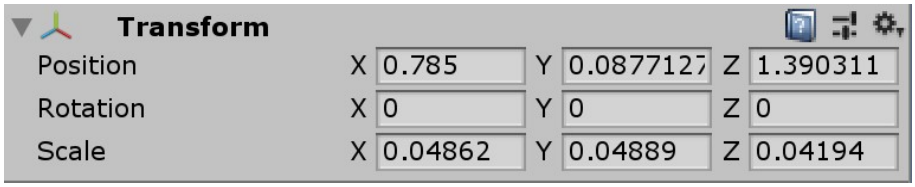


Figure 6.13: Transformation of “Error Workpiece 1”

The game objects will be assigned a material (see Fig. 6.14).

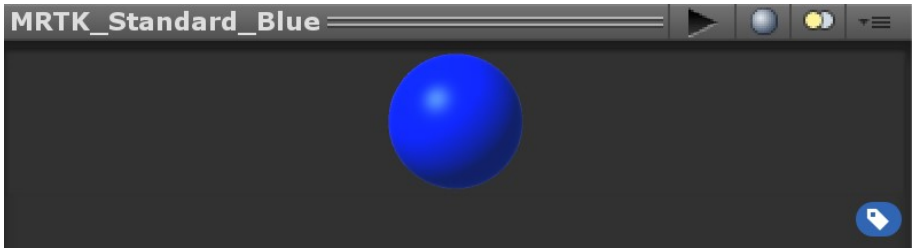


Figure 6.14: Material of the “Error Workpiece”

Implementing the Ability to Interact

To provide the user in the app with the opportunity to move the object, the “Tap To Place” script is added to the inspector panel of each error workpiece object via drag and drop.
A 3D game object, a cube, is created. This object, “Button Error Workpiece”, is the button that resets the workpiece counter. This game object is subordinate to “Error Workpiece 1”. Thus, a change in position of the object “Error Workpiece 1”, leads also to a change in position of the object “Button Error Workpiece”.
The button is placed on the left of “Error Workpiece 1”. The button’s transformation can be seen in Fig. 6.15.

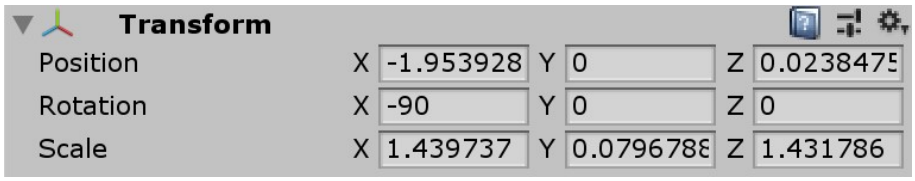
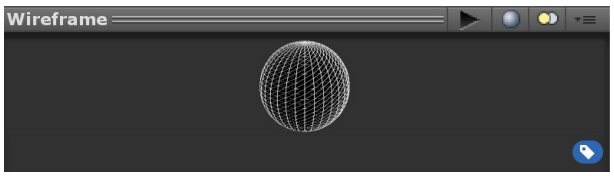
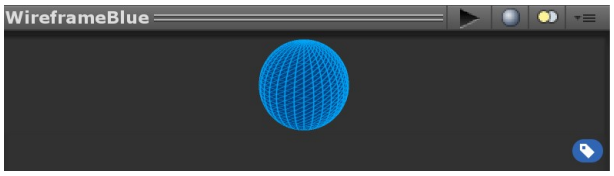


Figure 6.15: Transformation of “Button Error Workpiece ”

The button should be assigned to two materials depending on the counter. These are shown in Fig. 6.16.



(a) First Material of “Button Error Workpiece”



(b) Second Material of “Button Error Workpiece”

Figure 6.16: Materials of “Button Error Workpiece”

The “Button Error Workpiece” implements the script “Get Error Workpieces”. This script reads the data from the database, changes the color of the button, and resets the counter of the error workpieces.
In the script (see listing 6.12), two materials and three 3D text objects are defined as public. Therefore, two materials (see Fig. 6.16) and three 3D text objects (see Fig. 6.8.3) can be handed over to the script from the “Inspector” panel. Further, a variable for the counter is defined, as well as a list of the data type string to store data read from the database.

```
1  public class GetErrorWorkpieces : MonoBehaviour, IInputClickHandler,
       IInputHandler
2  {
3      private List<string> listFromDatabase = new List<string>();
4      private string dataStream = "";
5
6      int counter = 0;
7      string dataStreamBuffor = "";
8      public Material reset;
9      public Material gotAWorkpiece;
10     public TextMesh FirstObject;
11     public TextMesh SecondObject;
12     public TextMesh ThirdObject;
13 }
```

Listing 6.12: Error workpiece script part 1

The “Update()” function is called every 20ms. In “Update()” the coroutine “ReadingName()” is called. It reads the data from the database. This process is the same as for the game object “Current Order”. The exact procedure can be read in Chapter 6.8.1. The SQL code in

the PHP script (see listing 6.13), which is called “in ReadingName()”, queries if a workpiece is assembled wrong.

```
SELECT id, keywords FROM `factory_order` WHERE done = 0 AND valid =
    0
```

Listing 6.13: Error workpiece SQL query in PHP script

In the “Update()” function, the read data, order ID and color values, are depending on the counter assigned to one of the 3D text objects, and thus displayed to the user in the app (see listing **??**).

```
if (dataStreamBuffor != dataStream && dataStream!= "0 results")
{
    if (counter == 0) FirstObject.text = listFromDatabase[0] + "\n" +
        listFromDatabase[1];
    if (counter == 1) SecondObject.text = listFromDatabase[0] + "\n"
       + listFromDatabase[1];
    if (counter == 2) ThirdObject.text = listFromDatabase[0] + "\n" +
        listFromDatabase[1];

    counter++;
    dataStreamBuffor = dataStream;
}
```

Listing 6.14: Error workpiece script part 2

The function “OnInputDown()” is provided by the “HoloToolkit”. It is called as soon as the game object is clicked, for example with an “Air Tap”. This is the case when the user presses the button in the application. The function resets the counter. Likewise, the text of the 3D text objects (see Chapter 6.8.3) gets deleted. As a result, the text is no longer displayed (see listing 6.15).

```
public void OnInputDown(InputEventData eventData)
{
    // Button was pressed -> delete all data strings and set counter
        = 0
    FirstObject.text = "";
    SecondObject.text = "";
    ThirdObject.text = "";
    counter = 0;
}
```

Listing 6.15: Error workpiece script part 3

The “ChangeMaterial()” function is called every 20ms by the “update()” function. This function checks the value of the counter. If it is zero, the material a) shown in Fig. 6.16 will be loaded. Otherwise, the material b) will be loaded (see listing 6.16).

```
1  public void OnInputDown(InputEventData eventData)
2  {
3      public void ChangeMaterial()
4      {
5      // if counter = 0 -> Button was pressed -> Set material to black
6      if (counter == 0)
7      {
8          Renderer rend = GetComponent<Renderer>();
9          rend.material = reset;
10     }
11     // if counter is not 0 -> Button was  not pressed -> Set material
           to blue
12     else
13     {
14         Renderer rend = GetComponent<Renderer>();
15         rend.material = gotAWorkpiece;
16     }
17 }
```

Listing 6.16: Error workpiece script part 4

Object Title

For each of the game objects, a 3D text object is created to display the title. For the first object, the title is “First error workpiece”. This text object is placed next to the ball. This also moves when the ball is moved by “Tap To Place”, because it is subordinated to it.

Object Message

A 3D text object is created for every error workpiece object. This is positioned in such a way that it is placed under the respective ball. The text objects are handed over to the “Get Error Workpieces” script in the inspector panel of “Button Error Workpiece”. As a result, the content of the text object is determined in this script. The objects also moves when the respective ball is moved by “Tap To Place”, because it is subordinated to it.

6.8.4 My Elements

The game object “My Element” is a game object without text or geometric shape. All other game objects are subordinated to it. In this game object, the script World Anchor Manager provided by the “HoloToolkit” is loaded. “Persistent Anchors” are enabled in the “Inspector” panel of this object. Thus, the position of the game object remains present by ending and repeatedly starting the app. Since all game objects are subordinated to this game object, their positions are also persistent.

6.9 Creation of an Executable File for the HoloLens

After programming the app for the HoloLens, an executable file is created in this chapter. Using this file, the user can open the application with the HoloLens.
Therefore, a Visual Studio Project is created in Unity under “File” and “Build Settings”. Once the steps in Chapter 6.2 have been completed, the following settings (see Fig. 6.17) should be automatically selected, otherwise they must be set manually.

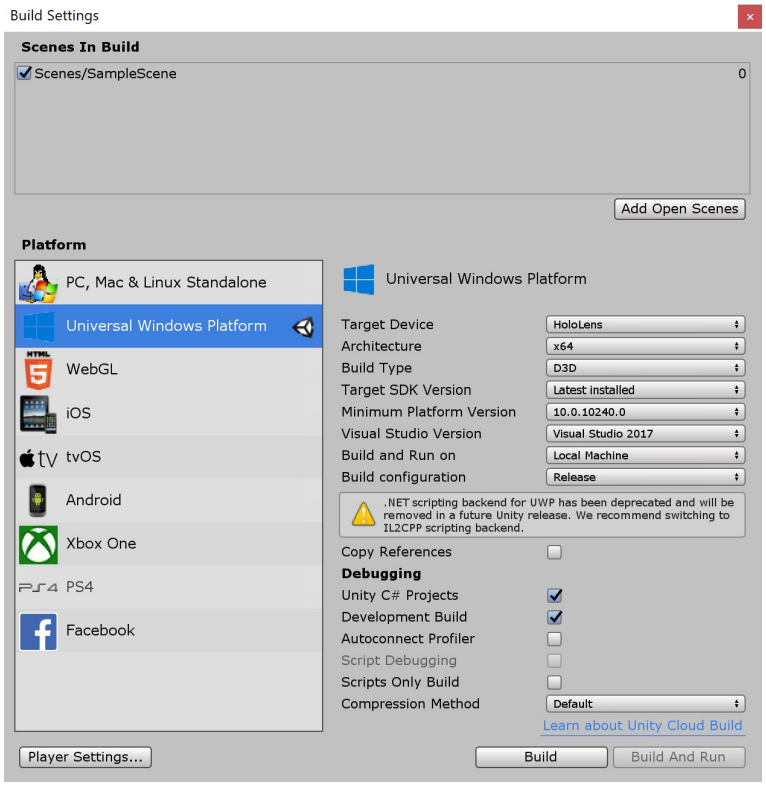


Figure 6.17: Build settings in Unity to create an executable file

To create the Visual Studio project, a new folder called “App” is created in the Unity project folder.
The project is opened in Viusal Studio. The HoloLens is connected to the development computer via USB. In the menu bar of Visual Studio, the build mode is set to “Master”, “x86” and “Device”. The executable file is then created by clicking “Start without Debugging”.

7 Functional Tests

To see if indeed all requirements of the requirement analysis (see Chapter 4.3.1) are fulfilled, the programming is functionally tested.

7.1 Current Order

As soon as the user gazes “Current Order”, it should display the ID and color values of all objects from the current order. The output should be formatted depending on the number of objects that have been ordered, so that the text is always readable and its position is always the same.
In Fig. 7.1, “Current Order” is shown how it looks like without interaction with the user. This is the case if he does not gaze at “Current Order”. The material of “Current Order” is blue. The static text “gaze to get information about the current order" is displayed.

Figure 7.1: “Current Order”, if it is not gazed

As soon as the user gazes “Current Order”, the material changes from blue to red. In addition, order ID and color values are displayed. The position of the data output is chosen in such a way that the output ends above “Current Object”. This is shown in Fig. 7.2.

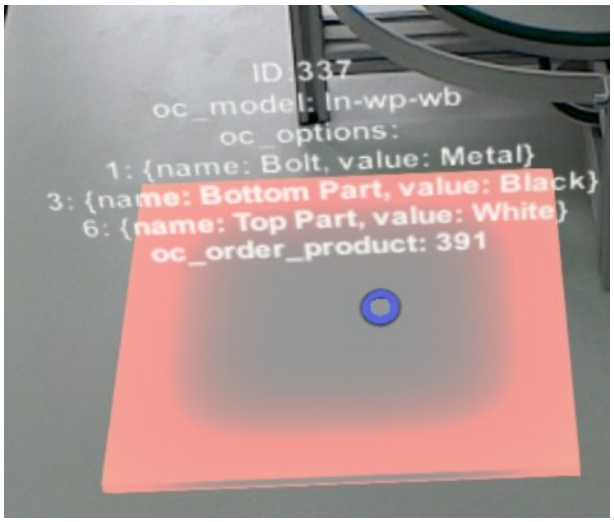


Figure 7.2: “Current Order”, if it is gazed and one object is ordered

The font size and position of the text are adjusted to the length of the output. The last text line of the text output always ends above “Current Order”. This shows the comparison of Fig. 7.2 and Fig. 7.3

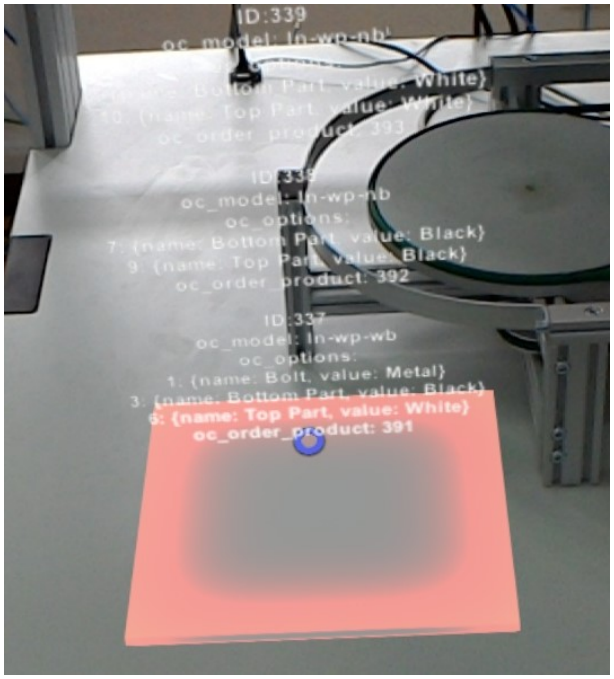


Figure 7.3: “Current Order”, if it is gazed and three objects are ordered

If more than three objects are ordered, the requirement that the text always ends above “Current Order” can no longer be met because otherwise the text size would be too small to read. In addition, the field of view through the HoloLens does not allow further lifting of the position, as the upper part of the text would then no longer be seen. Thus, font size and position of the text are not formatted differently for more than three workpieces than for three workpieces.

7.2 Stations

Each ball should display the status of its IMS station using a color code. If the station has an error, the color of the ball is red, which can be seen in Fig. 7.4.

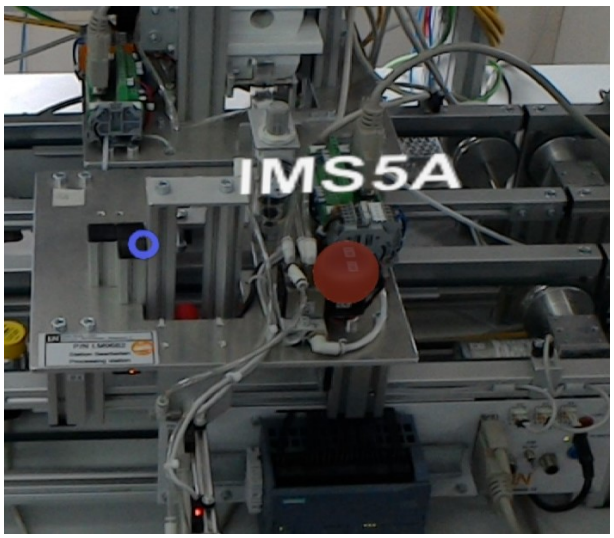


Figure 7.4: IMS station, if it detects an error

In case that the IMS station is in maintenance, the ball is colored in yellow (see Fig. 7.5).

Figure 7.5: IMS station, if it is in maintenance

The color of IMS Station, when processing flawlessly an order, is green (see Fig. 7.6).

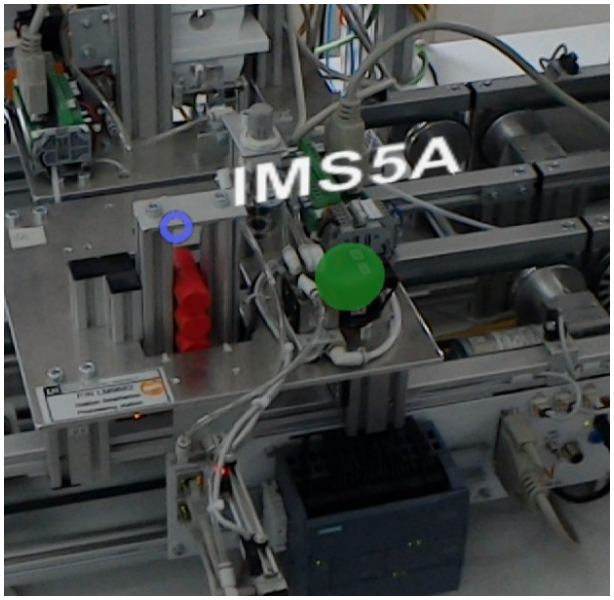


Figure 7.6: IMS station, if it is processing an order

If the IMS station does not produce, has no fault and is not in maintenance, the ball of the IMS station is not colored (shown in Fig. 7.7).

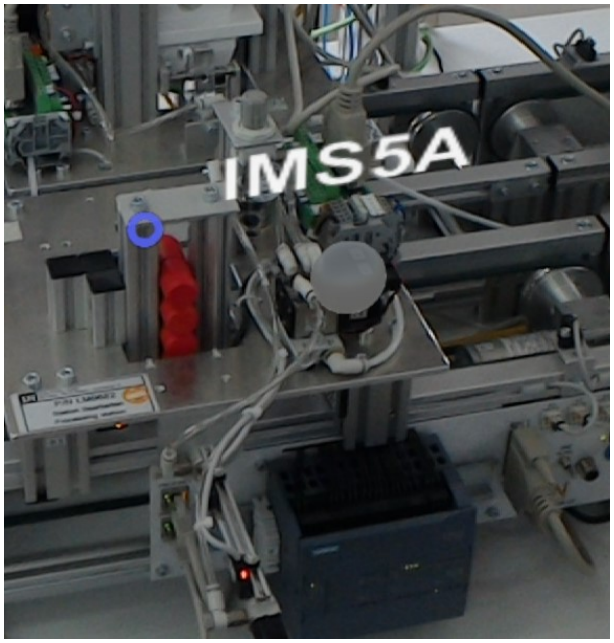


Figure 7.7: IMS station, if it is waiting

Determining the materials of the IMS stations depending on the status “processing”, “maintenance”, “error” and “waiting” causes the application to crash from time to time. The status of the IMS stations are first displayed with a delay. After that, the application crashes. This error could not be resolved within the scope of this work. In order to ensure that the application runs correctly, the IMS stations are therefore permanently assigned the “waiting status. Possibilities of an appropriate solution in future work are discussed in the Chapter 8.1.

7.3 Error Workpiece

Not more than the information of three error workpieces can be displayed at the same time. Each of the possible three error workpieces objects is supposed to display the order ID and the color values of the error workpieces. The output is positioned depending on the error workpiece’s position. This can be seen in Fig. 7.9, in which only one error workpiece is detected, as opposed to figure 7.10, in which three error workpieces are detected.
Similarly, the differences in material of the buttons can be seen comparing Fig. 7.8 and Fig. 7.9. In Fig. 7.8, no workpiece has been sorted out yet. The edges of the button are visible, but no color is assigned to it. In Fig. 7.9 one workpiece has already been sorted out. The button is now blue.

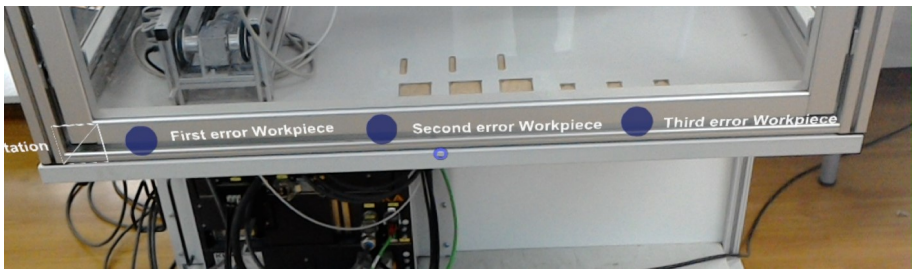


Figure 7.8: Output of “Error Workpiece”, if no error workpiece is detected

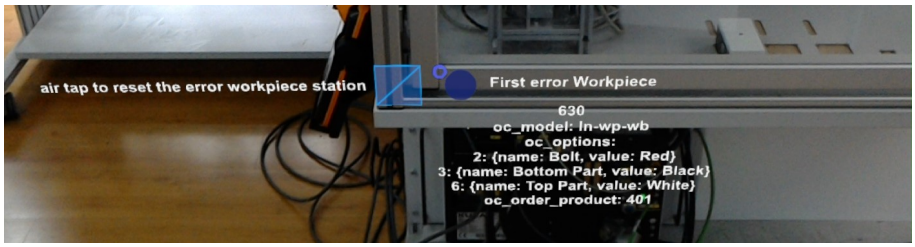


Figure 7.9: Output of “Error Workpiece”, if one error workpiece is detected

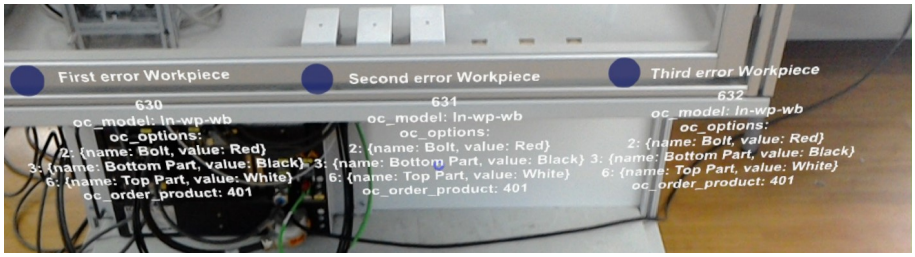


Figure 7.10: Output of “Error Workpiece”, if three error workpieces are detected

8 Summary

8.1 Conclusion

In this thesis an AR application for smart glasses was designed for an existing industry 4.0 plant. First, the market was analyzed and explored for suitable data glasses. The HoloLens from Microsoft was selected as the only appropriate product with extensive documentation, whose manufacturer is capable of providing support and which was already frequently used in industry.
In order to get constantly the latest data from the database of the plant, displayed on the HoloLens, a communication option between the HoloLens and the database was established. In this case, a PHP script, which is located on the database server and queries data from the database via SQL, can be called from the HoloLens development environment. The data read in from the database are thus always up-to-date and are displayed to the user of the application in real time.
When programming the AR application, it turned out that adding the “Mixed Reality Toolkit” could save the programmer a lot of effort. Capabilities, such as spatial mapping, remembering the coordinates of objects or aiming and clicking on objects and thus the ability to interact with virtual objects, were made available to the user by adding prefabricated program codes, if required. The programmer has thus the possibility to insert the basics and only has to program the contents of the application.
The AR application, which was created for this bachelor thesis, gives the user of the app information about the current orders in the ERP system of the plant. Here, ID and color values of all workpieces contained in the order are indicated.
Ideally, the different states of the IMS stations should be displayed to the user. Therefore, the user of the app should know at any time in which state the plant is. If the system detects an error, the user of the app should know immediately in which IMS station the error was detected. The user should also get informed directly if the inventory of a station is exhausted. This in turn should be indicated by balls above the stations, which change the color depending on their status.
However, as described in Chapter 7.2, displaying the status of the IMS stations causes the app to crash. Due to this failure, which cannot be solved within the scope of this work, the appropriate adaptation leads to the situation that only the objects for the IMS stations are

implemented in the application, but they do not change the material depending on the status. Since displaying the status of the IMS stations is first delayed before the app crashes completely, one explanation for the problem may be that too much data is being queried too often. For each IMS station, data from the database are queried in the respective PHP script. As the information about the status of each of the eight stations is in the same table, this table in the database is queried by eight different PHP scripts at the same time. This query is done every 20ms. Thus, one solution would be to query each status of all IMS stations in one PHP script and store it in an array. The array could then be evaluated in Unity. Based on this evaluation, the materials of the IMS stations could then be set. Another alternative would be to call the PHP scripts for the IMS stations created in this work not via eight different scripts in the respective update function, but in one script after another.
Defectively produced workpieces are ejected from the production process of the Industry 4.0 system. Information about these incorrectly produced workpieces is displayed to the user by displaying the order ID and the color values. By comparing the physical object that was sorted out and the color values that are displayed, the user always knows instantly why the product was rejected.
The tracking of objects in the application is markerless. The HoloLens continuously scans the environment and creates a room plan based on this scan. So, virtual objects can be inserted into the real world based on these scans. The advantage is that no markers have to be used to track objects. The application enables the user to display the information at any location because he can easily move the objects. These objects interact with the environment by being placed on real surfaces.

8.2 Future Work

To solve the detected problem with the app, first, the IMS stations or the setting of their materials could be implemented flawlessly. This was not achieved in this work as mentioned in Chapter 7.2. Possible solution approaches can be found in Chapter 8.1.
Amongst others, one selection criterion for the data glasses was that future AR applications could be implemented. This is indeed easily possible with the HoloLens. For example, instead of displaying the color values of the error workpieces the data from the database can be used to create a real 3D model of the workpiece. The user then does not compare the values with the real object, but the real object with a virtual object. Further, voice commanding can be used to retrieve data from the database. More complex holograms, for example a human, can be created. This can bring the app closer to the user and serve as a kind of operating manual. Here, spatial sound could also be used for the HoloLens' ability to play a sound depending on distances and directions. Extensions of future AR applications are therefore possible.

8.3 Practical Implications

The use of the app has already shown in test phases how AR can considerably simplify and accelerate production processes. For example, in the event of a system error, it was possible to immediately identify which IMS station had an error, regardless of the user's position in the room. Thinking one step further and considering how many processes can be made more productive by AR, it becomes obvious how much AR can change the industry in the future.

Bibliography

[1] *Epson Moverio BT-300 vs Microsoft HoloLens*. https://productz.com/en/epson-moverio-bt-300-vr-headset-vs-microsoft-hololens-vr-headset,

[2] *MR Content Types, Interactions & Gestures, Interfaces & Spacingitout*. https://arvrjourney.com/research-mr-content-types-interactions-gestures-interfaces-spacing-

[3] *Solids Process automation*

[4] ANDELFINGER, Volker P. ; HÄNISCH, Till: *Industrie 4.0: Wie cyber-physische Systeme die Arbeitswelt verändern*. Springer, 2017

[5] AZUMA, Ronald T.: A Survey of Augmented Reality. In: *Presence: Teleoperators and Virtual Environments* 6 (1997), Nr. 4, 355-385. http://dx.doi.org/10.1162/pres.1997.6.4.355. – DOI 10.1162/pres.1997.6.4.355

[6] BÜHLER, Peter ; SCHLAICH, Patrick ; SINNER, Dominik: *Webtechnologien: JavaScript–PHP–Datenbank*. Springer-Verlag, 2018

[7] BURNUS, Heinz: *Datenbankentwicklung in IT-Berufen: eine praktisch orientierte Einführung mit MS Access und MySQL*. Springer Science & Business Media, 2007

[8] COPE, Kevin: *What is the Automation Pyramid?* https://realpars.com/automation-pyramid/#, . – Accessed Juni 11, 2018

[9] CURLE, Peter: *The key of Augmented Reality in Industry 4.0*. https://vrfutr.com/2018/11/09/the-key-of-augmented-reality-in-industry-4-0/. Version: 2018

[10] DÖRNER, Ralf ; BROLL, Wolfgang ; GRIMM, Paul ; JUNG, Bernhard: *Virtual und Augmented Reality (VR / AR) - Grundlagen und Methoden der Virtuellen und Augmentierten Realität*. 2013. Aufl. Berlin Heidelberg New York : Springer-Verlag, 2014. – ISBN 978–3–642–28903–3

[11] FEINER, Steven ; MACINTYRE, Blair ; HAUPT, Marcus ; SOLOMON, Eliot: Windows on the world: 2 D windows for 3 D augmented reality. In: *ACM Symposium on User Interface Software and Technology*, 1993, S. 145–155

[12] GRETZINGER, Nick: *Industrie 4.0 ? Grundlagen und aktuelle Entwicklung*. https://www.ingenieur.de/technik/fachbereiche/industrie40/industrie-4-0-grundlagen-und-aktuelle-entwicklung. Version: 2018

[13] HÖHL, Wolfgang: *Interaktive Ambiente mit Open-Source-Software : 3D-Walk-Throughs und Augmented Reality fu?r Architekten mit Blender 2.43, DART 3.0 und ARToolKit 2.72*. Vienna : Springer Vienna, 2009. – ISBN 978–3–211–79171–4

[14] KÖNIG, Marie-Christine ; JALALZADA, Mohamad H.: *Entwicklung einer Augmented Reality Anwendung für eine AR-Brille auf Basis von ArUco-Markern*. – Accessed 2018

[15] LUCAS-NÜLLE-TEAM: *CSF 4: ERP Lab for Smart Factory 4.0*. 2019

[16] LUCAS-NÜLLE-TEAM: *Technical documentation for offer: 1762PRC44*. 2019

[17] LUETH, Knud L.: *Will the industrial internet disrupt the smart factory of the future?* https://iot-analytics.com/industrial-internet-disrupt-smart-factory/. Version: 2015

[18] MEIER, Andreas ; KAUFMANN, Michael ; KAUFMANN, Michael: *SQL-& NoSQL-Datenbanken*. Springer, 2016

[19] MICHAEL RÜSSMANN, Philipp G. Markus Lorenz L. Markus Lorenz: *Industry 4.0 - The Future of Productivity and Growth in Manufacturing Industries*. The Boston Consoulting Group, 2015

[20] MICROSOFT: *Microsoft reveals HoloLens hardware specs*. https://www.theverge.com/2016/2/29/11132090/microsoft-hololens-hardware-specifications-developer-kit. Version: 2016

[21] MICROSOFT: *HoloLens (1st gen) hardware details*. https://docs.microsoft.com/en-us/windows/mixed-reality/hololens-hardware-details. Version: 2018

[22] MICROSOFT: *Install the tools*. https://docs.microsoft.com/en-us/windows/mixed-reality/install-the-tools. Version: 2019

[23] MICROSOFT: *Mixed Reality documentation*. https://docs.microsoft.com/en-us/windows/mixed-reality/. Version: 2019

[24] MILGRAM, Paul ; KISHINO, Fumio: A taxonomy of mixed reality visual displays. In: *IEICE TRANSACTIONS on Information and Systems* 77 (1994), Nr. 12, S. 1321–1329

[25] REINHEIMER, Stefan: *Industrie 4.0: Herausforderungen, Konzepte und Praxisbeispiele*. Springer Fachmedien Wiesbaden, 2017

[26] SAIRIO, Mikko: *Augmented Reality, Helsinki University of Technology*. 2001

[27] SCHECHTER, Sonia: *What is markerless Augmented Reality? | AR Bites*. 2014

[28] SCHICKER, Edwin: Datenbanken und SQL. In: *Auflage, Teubner-Verlag* (2000)

[29] SCHMITT, Karola: *Top 5 Reasons Why Industry 4.0 Is Real And Important*. https://www.digitalistmag.com/industries/2013/10/15/top-5-reasons-industry-4-0-real-important-0833970. Version: 2013

[30] SCHRÖDER, Bernd: *Steuerungstechnik für Ingenieure - Ein Überblick*. 2014. Aufl. Berlin Heidelberg New York : Springer-Verlag, 2014. – ISBN 978–3–658–06643–7

[31] SMITH, Sherri L.: *Meta 2 is the Augmented Reality of Tomorrow, Today*. https://www.tomsguide.com/us/meta-2,news-24662.html,. – Accessed March 14, 2017

[32] TÖNNIS, Marcus: *Augmented Reality - Einblicke in die Erweiterte Realität*. 2010. Aufl. Berlin Heidelberg New York : Springer-Verlag, 2010. – ISBN 978–3–642–14179–9

[33] UWE DOMBROWSKI, Philipp K. Thomas Richter R. Thomas Richter: Interdependencies of Industrie 4.0 & Lean Production Systems: A Use Cases Analysis. In: *Procedia Manufacturing* 11 (2017), 1061-1068. http://dx.doi.org/10.1016/j.promfg.2017.07.217. – DOI 10.1016/j.promfg.2017.07.217

[34] UWE DOMBROWSKI, Thomas R.: *Supplementing Lean Production Systems with Information and Communication Technologies*

[35] ZANDER, Hans-Joachim: *Steuerung ereignisdiskreter Prozesse - Neuartige Methoden zur Prozessbeschreibung und zum Entwurf von Steueralgorithmen*. 2015. Aufl. Berlin Heidelberg New York : Springer-Verlag, 2015. – ISBN 978–3–658–01382–0

A Appendix

The appendix of this thesis contains amongst others, the program codes, and is stored on a CD. This storage medium is handed over to Prof. Dr.-Ing. Florian Wenck and Prof. Dr.-Ing. JianQiang Shen. It can be viewed at any time.

Versicherung über die Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Arbeit im Sinne der Prüfungsordnung nach §16(5) APSO-TI-BM ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen habe ich unter Angabe der Quellen kenntlich gemacht.

Hamburg, July 1, 2019
Ort, Datum

Unterschrift