Online support for shop-floor operators using body movements tracking and augmented reality

Bachelor's Thesis

Mathias Andréasson
Martin Henoch
Sarar Innab
Tony R Jaakkola
Christoffer Lindvall
Emelie Widegren

Department of Signals & Systems
Chalmers University of Technology
Gothenburg, Sweden 2015

Bachelor's Thesis 2015:1

Online support for shop-floor operators using body movements tracking and augmented reality

MATHIAS ANDRÉASSON
MARTIN HENOCH
SARAR INNAB
TONY R JAAKKOLA
CHRISTOFFER LINDVALL
EMELIE WIDEGREN

Examiner: Petter Falkman
Supervisor: Knut Åkesson

Chalmers University of Technology
Department of Signals and Systems
SE-412 96 Gothenburg
Sweden

Department of Signals and Systems
Gothenburg, Sweden June 2015

Abstract

Training operators in factories can be both a long and expensive venture, especially due to errors in the production. In this thesis an assembly scenario in which an operator is equipped with a prototype platform consisting of AR-glasses (Augmented Reality) in combination with a motion tracking system is explored.

The hypothesis was that the use of this technology could lead to the reduction of errors that occur during training sessions and thereby achieving a more time-efficient and productive training method.

In order to be able to evaluate the technology an experimental platform has been developed and usability tests have been performed to evaluate the use of the technology to see if providing the operator with real-time instructions through use of AR-technology enhances the learning experience and if improvements could be detected.

After evaluation, it was concluded that using the prototype did not show a detectable improvement, neither in time expenditure nor number of errors encountered versus conventional methods.

The evaluation also showed that improvements to the experimental platform are very possible and continuing work would likely remarkably enhance the results.

Sammanfattning

Att träna montörer inom industrin kan vara en både tidskrävande och kostsam process, särskilt på grund av fel som uppstår i produktionen. I denna rapport redovisas en lösning där montörer utrustas med en prototyp bestående av AR-glasögon (Augmented Reality) i kombination med motion capture-teknik.

Vår hypotes var att användningen av den utvecklade tekniken skulle leda till en reduktion i felfrekvens under montering, och därmed leda till en mer tidseffektiv träningsmetod.

För att utvärdera prototypen utvecklades en experimentell plattform och användningstester genomfördes, för att undersöka om presentationen av realtidsinstruktioner genom AR-teknologi ledde till en mätbar förbättring i prestanda.

Efter utvärderingen drogs slutsatsen att den använda prototypen inte visade någon direkt fördel, varken i hur lång tid en session tog eller i antalet fel jämfört med konventionella metoder.

Utvärderingen visade också att det finns stora möjligheter till att utveckla den experimentiella plattformen och att fortsatt arbete skulle kunna ge signifikanta förbättringar av resultaten.

Acknowledgements

Knut Åkesson, for the support, patient guidance and advice.

Knut Åkesson, Julien Provost and Amir Hossein Ebrahimi for the idea and for letting us continue on your work.

Björn, Michael, Vera, Marco, Max, Dimitri, Robert, Christian and Edvard for helping us complete the usability tests.

Christian Ulvemyr for his help, materials support and general sponsorship.

This work has been supported by the Know4Car project, grant agreement number 284602, funded by the EC Seventh Framework Programme theme FoF-ICT-2011.7.4

Group 14, Gothenburg 2015-06-07

Contents

Glossary

angularJS Is a web application framework.

apache camel A Java framework that provides advanced routing features such as routing messages based on the contents of the application layer in the Internet protocol. .

bootstrap A collection of tools for creating websites and web applications.

Head up Display Transparent display that visualize data within the users field of view.

HTML Is the standard markup language used to create web pages.

javaScript Is a dynamic programming language.

javaScript Object Notation An alternative to XML used to transfer data from a database to a website in a text format.

mySQL Is a relational database management system.

previous project *Online support for shop-floor operators using body movements tracking [1].*

servlet A Java feature that is responsible for serving a portion of requests in a server that are logically grouped. A common type of Servlet is the web Servlet that basically is responsible for serving a resource in a web server.

unity3D A cross-platform game engine developed by Unity Technologies.

Acronyms

API Application Programming Interface.

AR augmented reality.

CSS Cascading Style Sheet.

EER Enhanced Entity–Relationship model.

GUI Graphical User Interface.

HUD heads-up display.

JPA Java Persistence API.

JSF JavaServer Faces.

OHMD Optical Head Mounted Display.

ORM Object-Relational Mapping.

POJO Plain Old Java Object.

RPC Remote procedure call.

SDK Software Development Kit.

UML Unified Modelling Language.

XML Extensible Markup Language.

1

Introduction

A challenge in the manufacturing industry is to ensure that the operator has access to the right information at the right time for effective knowledge management. New technology using 3D cameras makes it possible to, in real-time, follow what happens in an assembly station via motion capture, as in the project*Online support for shop-floor operators using body movements tracking* [1]. This thesis explores the integration of human operators in information management systems further by using augmented reality as tool for displaying information in combination with already developed technology to follow the operators movements and evaluate the use of it.

Training operators can be both time-consuming and expensive, today this is often done directly on the assembly line. An error in one of these processes can be extremely costly because it can result in a faulty product. Therefore a system which gives an instant response on an incorrect action can help repair the fault before the product moves further down the assembly line. In factories of today hundreds of different configurations of products are assembled. To assist and speed up the learning of these tasks virtual training can be applied, for example, see [2].

"The ability to overlay computer graphics onto the real world is commonly called augmented reality" [3]. Rather than presenting information on traditional monitors the operators can be equipped with AR glasses that are projecting information directly in front of them. The technology is similar to a heads-up display that can be seen in vehicles [4].

Augmented reality is interesting since it allows the user to see information that the user normally cannot see with its own senses. Unlike virtual reality, where the real world is replaced with an artificial one, augmented reality allows the user to interact with the real world and at the same time see both the real and virtual world, see Figure 1.1 to see a reality-virtuality continuum. We believe the technology can help close the gap between product development and manufacturing because of the ability to reuse and reproduce digital information and knowledge while supporting the operators.

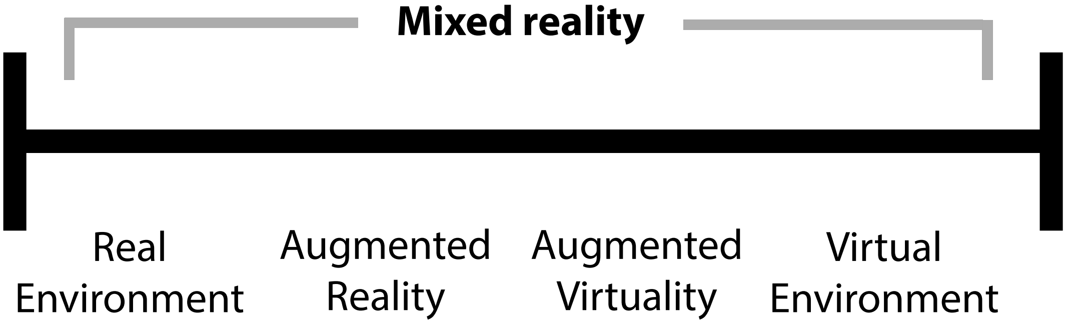


Figure 1.1: Reality–virtuality continuum.

Previous studies have investigated the use of AR in an assembly environment [5], [6], [7] as well as the usage of virtual reality to assist operators [8]. However, those studies either use virtual reality or tracks the assembly object itself for determining progress while this thesis tracks the operator's activities in order to determine progress.

1.1 Aim

The aim is to evaluate the advantages and disadvantages of human operators using AR glasses in information management systems with the aim of improving learning, minimize errors and simplify the management of information to the operators.

1.2 Scope

Due to the fact that the scope of this project is time limited to the first half of 2015, from January through June many delimitations had to be set on the project to limit it from subject areas that were not deemed necessary for evaluating the use of AR.

The project will not include development of algorithms for motion capture and imaging or animations of 3D models. No special consideration will be taken towards ergonomics more than setting up a viable test environment. Because of the time constraint the system will be integrated with established libraries when possible instead of creating tools and writing code from the bottom up. Advanced graphics and animations are outside the scope of this project, both due system performance limitations and time. Some choices, especially considering the layout of the test station and the motion capture were already decided from the beginning since this project is based upon a previous project.

1.3 Structure

Chapter 2 describes the problem and gives information about the various technologies used in this project in order to supply the reader an understanding of the field. In chapter 3 the setup of the experimental platform is described. Chapter 4 describes how the platform works with specification of the underlying modules. Chapter 5 contains the methods and theory concerning evaluation of the project as well as the structure of the usability tests. Then the results of those tests are presented and explained and discussed in Chapter 6. Finally, the conclusion that wraps up this entire report is presented in chapter 7.

2

Problem

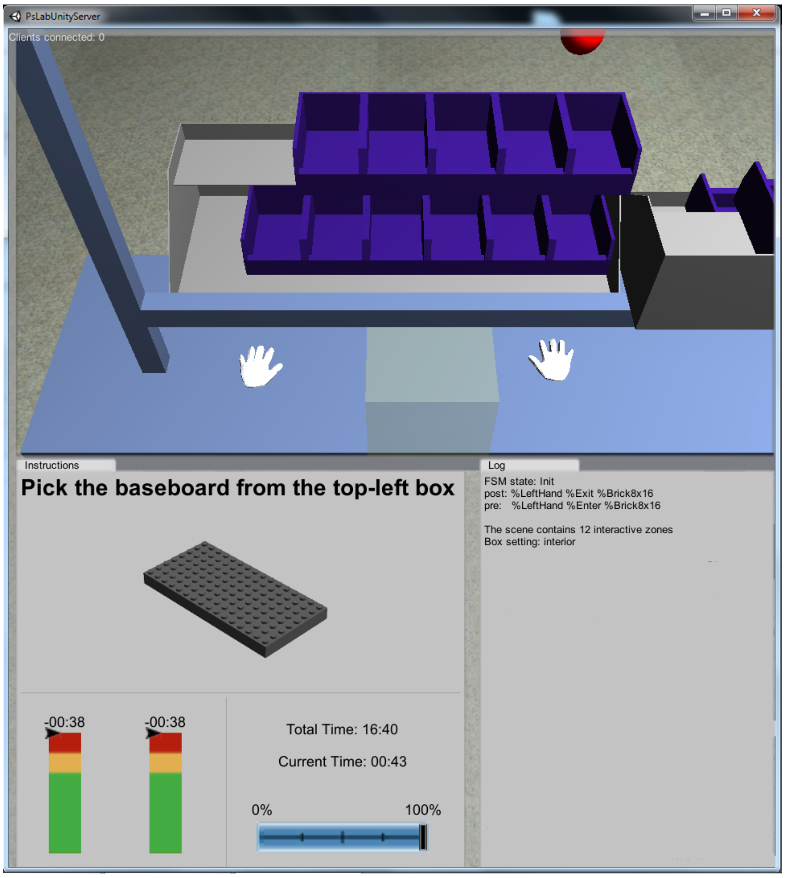
To successfully accomplish the goal to evaluate AR-technology in manufacturing industry, several problems need to be overcome. An AR platform must be developed. A scenario is needed to be set up to in which the AR platform can be tested. And lastly a method of evaluating the performance of the platform has to be devised.

The point of interest in the evaluation is to find out how the performance of an operator changes when using AR assistance instead of a conventional instruction list in an assembly assignment.

The scenario that will be used in this purpose is analogous to a training exercise used when instructing production engineers at a known automotive manufacturing company. In this scenario the operator is instructed via a visual medium to assemble a LEGO construction resembling a vehicle or parts of a vehicle.

2.1 Background

In the project *Online support for shop-floor operators using body movements tracking* [1] a tracking system for this same scenario is developed at the Department of Signal & Systems at Chalmers University of Technology. In the project, from now on referred as the *previous project*, a platform is developed with the purpose to, in real-time, track an operator in an assembly environment. The platform tracks an operator's hand movements with a 3D-camera in order to determine the operator's progress during an assembly session and thus add feedback to the system so it can detect when an instruction is completed and the operator can be provided with the correct instructions at correct time. The instructions given to the operator is displayed on a screen monitor in the form of text based assignments and a graphical representation of the boxes in which the parts for the assembly are stored.



Pick the baseboard from the top-left box

```
FSM state: Init
post: %LeftHand %Exit %Brick8x16
pre:   %LeftHand %Enter %Brick8x16

The scene contains 12 interactive zones
Box setting: interior
```

Total Time: 16:40

Current Time: 00:43

Figure 2.1: A screen shot from the previous project during an assembly session.

2.2 Contribution

In this thesis, the platform from the previous project is extended and modified to display instructions and visible markers on the part boxes on the AR glasses instead of a traditional monitor.

Additional modifications include a database to manage and store data and a front-end application to view statistics, track the operators progress and manage an assembly session. The contribution also includes communication modules for integrating all the different parts of the platform.

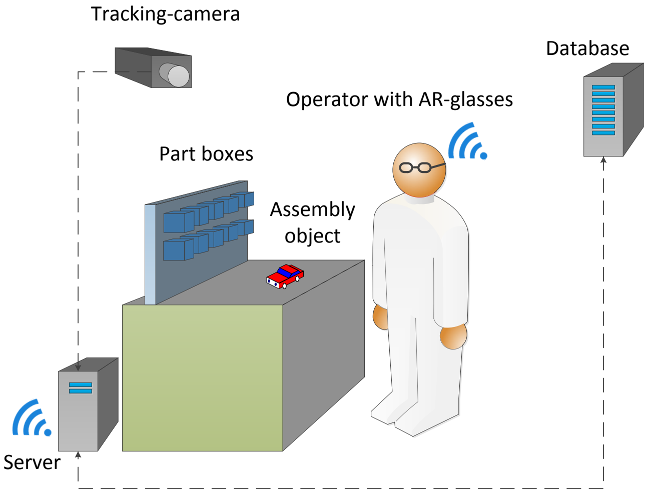


Figure 2.2: A simplified overview of the system.

The hypothesis of this research is that the new modifications to the system will result in an assembly guidance platform superior to the conventional assembly training methods such as using instruction lists on paper or viewing instructions on traditional monitors. The parameters which is expected to show improved results are; lower rate of errors and a shorter assembly time.

3

Setup of the Experimental Platform

The development of the platform used in the evaluation were divided into four key modules with distinctly different functionality. As mentioned in the previous chapter some parts of the modules were inherited from the previous project.

The key modules were the following;

- A module for displaying information to, and receiving optical input from the operator. (*Visual AR*)
- A module for monitoring the operator movements. (*Motion capture*)
- A module for handling instructions (*Instruction handling*)
- A module for storing and retrieving operator data (*Progress recording*)

All these modules also needed a way of communicating. In the following sections the problems these modules address will be described along with a description of the implementations and integration of said modules.

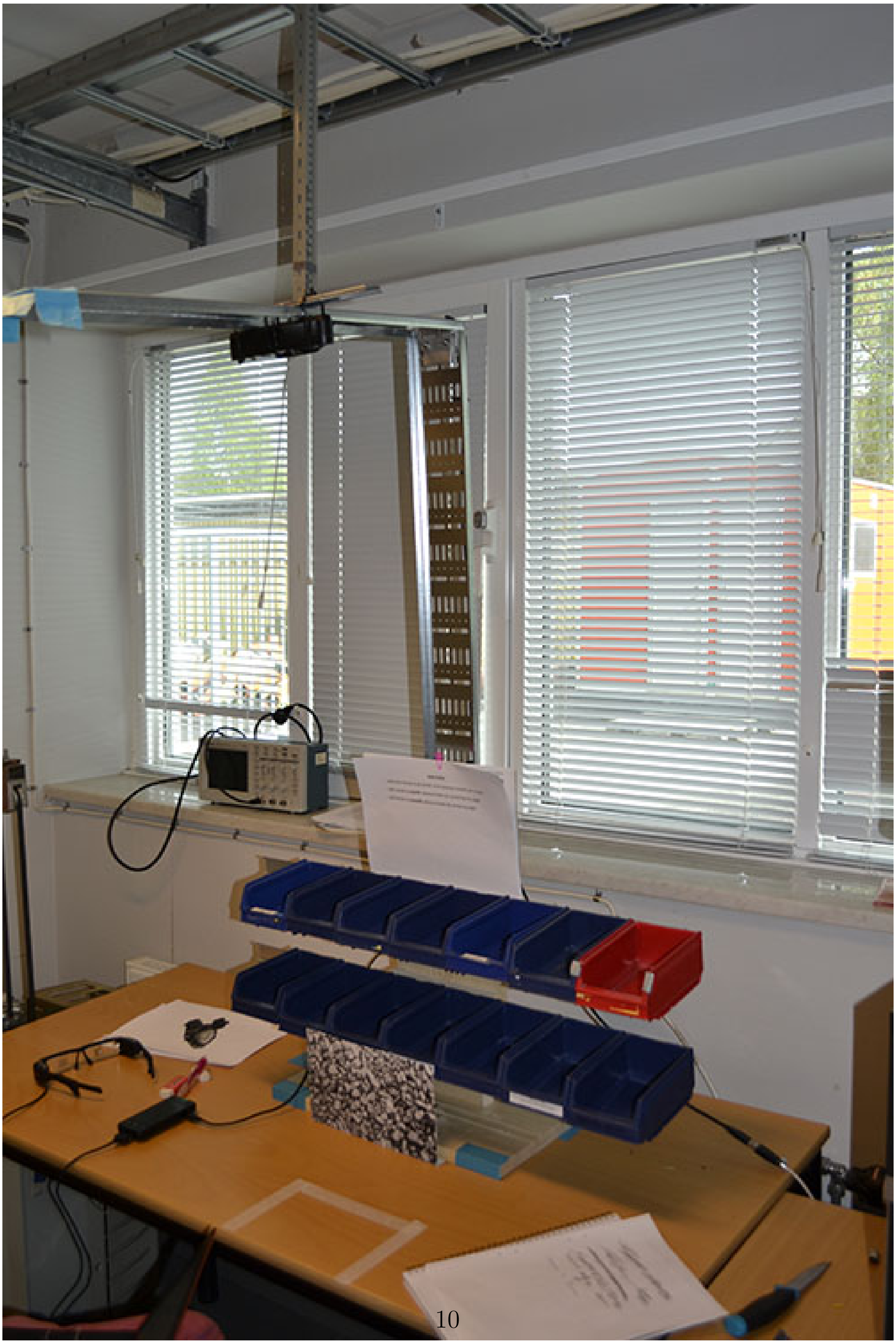


Figure 3.1: The experimental platform showing the 3D-camera, part boxes and the AR-glasses with the accompanying touchpad

3.1 Visual AR

The main purpose of the AR module is to display the assembly instructions to the operator in such a way that the needed information is easily accessible without disturbing the operator. To accomplish this, two crucial component is necessary; first a hardware device capable of displaying the Graphical User Interface (GUI) in the operators field of view, secondly a software environment in which the GUI program is run.

3.1.1 AR-Glasses

To be able to place visual markers and relate graphical object to actual real world positions the device needed to have a camera input. Other vital requirements can be seen in table 3.1

Table 3.1: Our requirements for the AR glasses

Requirements	
Must have/be	Should have/be
Binocular OHMD	Unhindered vision
Price under 20,000 SEK	Accelerometer, gyroscope
Camera	Fully portable
Support for Unity3D	Customizable to the user
Fully portable	At least 6 hours of battery life

After some research the following devices which met the requirements were found.

Table 3.2: Specifications of the different AR glasses in consideration

Manufacturer – model	Features	Price ($)
Epson -		
Moverio BT-200	23° FOV: 60Hz: 88g: 960×540px Support for Unity3D: Accelerometer Camera: Microphone: 6hrs. battery life Magnetometer: 24 bit-color: Gyroscope Wireless	700
Vuzix -		
Wrap1200DXAR	35° FOV: 30Hz: 85g: 852×480px Accelerometer: Gyroscope: Camera 4.5 hrs. battery life: Accelerometer 24 bit-color: Magnetometer: Gyroscope	1500
Google - Glass	35° FOV: 30Hz: 43g: 640×360px Accelerometer: Gyroscope: Camera Wireless: Magnetometer	1500

While all three of these can be used as an AR-device, the Epson glasses have several properties that makes it the better choice, such as transparent screens, Stereoscopic view mode and touch pad input capabilities and was consequently chosen for this module.

The Moverio BT-200 glasses from Epson [9] is an Android powered device consisting of a head mounted display/sensor unit connected to a combined hand-controller/battery-pack unit.

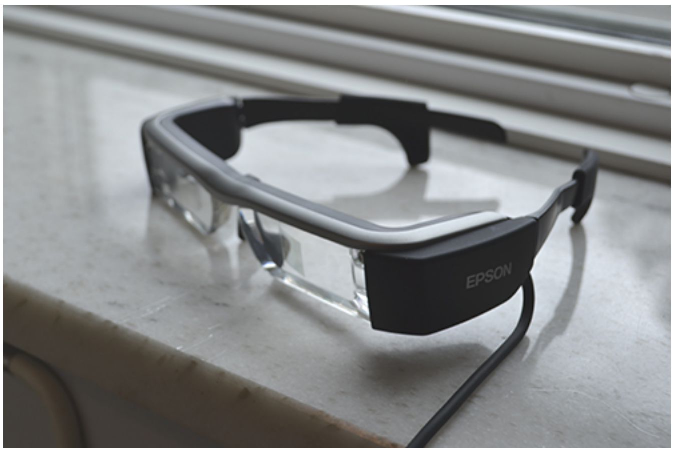


Figure 3.2: Epson Moverio BT-200.

The AR-glasses uses two transparent screens embedded in the glasses lenses, one for each eye, to display graphics. These can be used in three different display modes; Stereoscopic 3D, Stereoscopic 2D or individual projection. The use of stereoscopic 3D is preferable for the intended purpose in this project, on the account of the 3D aspect making it easier for the operator to interpret the graphical instructions.

3.1.2 Software for AR-Glasses

Due to the large amount of real-time critical data that needs to be processed by the AR driving applications, as well as the limited processing power of the see-through glasses, a client/server approach is used. This allows for a more modest use of processing needed to be performed on the glasses themselves (client) and the real time critical and larger parts of data can be handled by a more powerful modern computer (server). As the system for processing of the hand-tracking data and the operating sequences were developed in a previous project [1], the new platform makes use of much of the existing infrastructure along with new software methods for extracting interesting parts of data from the system in the previous project. Using this extracted information the new system builds a virtual scene that holds information about all virtual objects created by the system. This information in pertinent parts, is then relayed to the client application via wireless communication and is used to drive the visualization on the client.

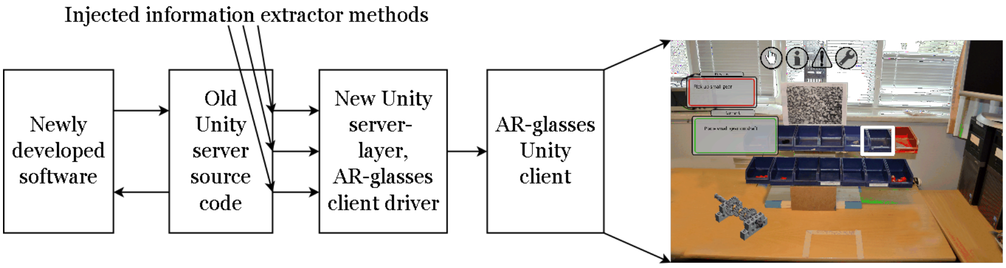

Injected information extractor methods
Newly developed software
Old Unity server source code
New Unity server-layer, AR-glasses client driver
AR-glasses Unity client

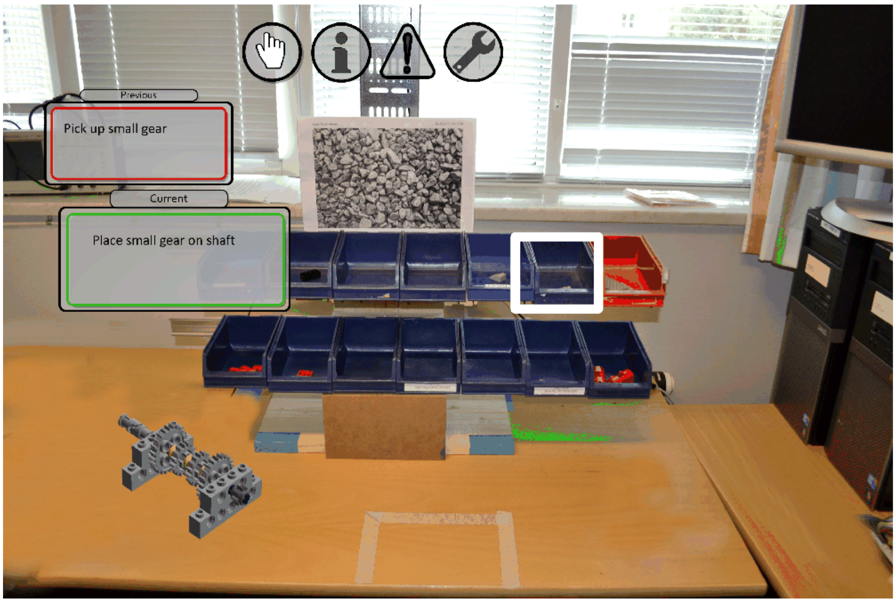
Figure 3.3: Overview of the information connections of the server/client system

The client application in the glasses is then responsible for displaying the overlaying of virtual 3-dimensional objects into a real-world scene as well as textual and graphical information to the operator.

3.1.3 AR User Interface

The information presented to the operator via the glasses can be divided into three categories, operational information, device status and graphical instructions. Device status refers to information related to the AR-glasses, such as battery status and WIFI signal. These can be displayed on static positions on the Head up Display (HUD). Operational information refers to textual and graphic based information gathered from a peripheral database containing assemble instructions and images of components. Graphical instructions refers to the graphical elements that are used to identify the correct components to assemble.

On the HUD four separate views can be displayed, the standard of which is the ”Action view” that is used while performing the assembly. As seen in figure 3.4 the action view displays non-view-obtrusive text based instructions on the left side of the HUD, along with a graphical representation of the actual part. A white square indicates which box the part can be found. After several iterations and test, a HUD design that is semi translucent and sparsely uses bright colors works best for this view.

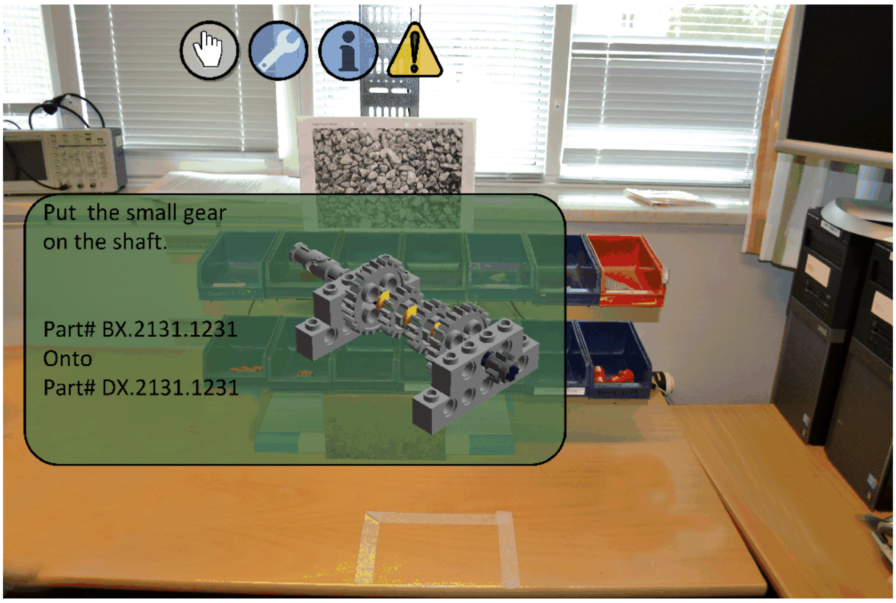


Pick up small gear

Place small gear on shaft

Figure 3.4: Action overview.

If more detailed information is needed by the operator there is an option to switch view screen to "Information view" for more comprehensive assembly instructions and a larger graphical view.

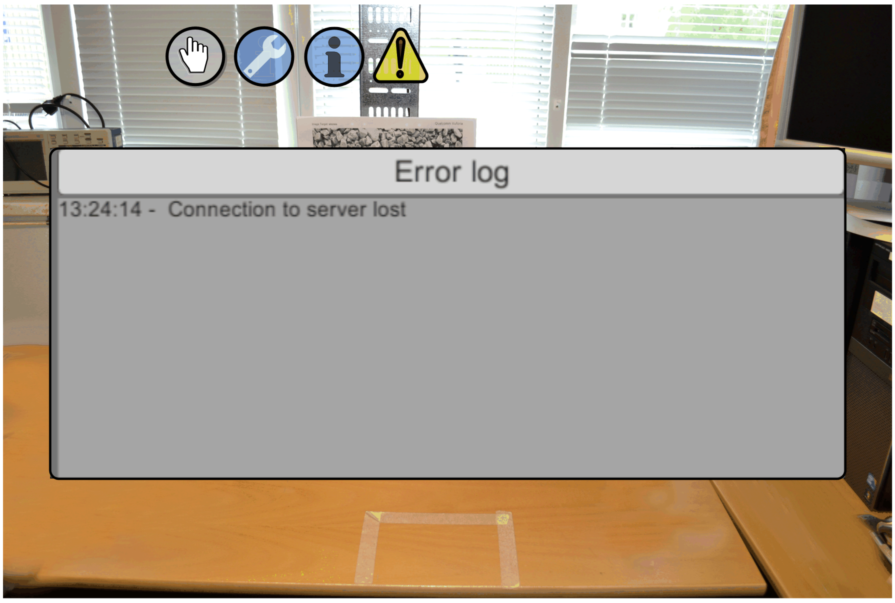


Put the small gear on the shaft.

Part# BX.2131.1231
Onto
Part# DX.2131.1231

Figure 3.5: Information overview.

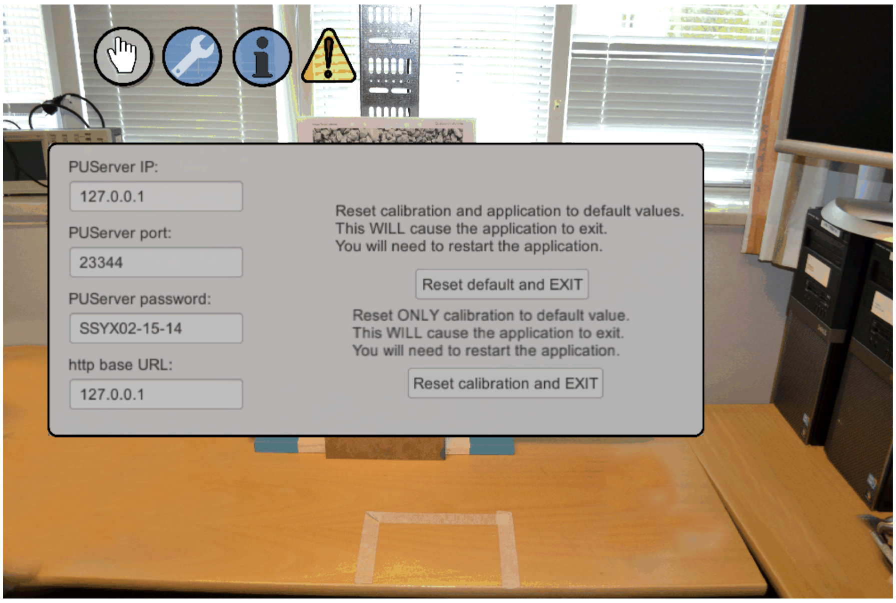
The ”Device view” is used for displaying error/alarm log. The functionality of this view is limited in the current edition of the platform but could easily be expanded in the future.


Error log

13:24:14 - Connection to server lost

Figure 3.6: Device view

And finally a view to set up communication parameters and calibrate the optical reference.



PUServer IP: 127.0.0.1

PUServer port: 23344

PUServer password: SSYX02-15-14

http base URL: 127.0.0.1

Reset calibration and application to default values.
This WILL cause the application to exit.
You will need to restart the application.

Reset ONLY calibration to default value.
This WILL cause the application to exit.
You will need to restart the application.

Figure 3.7: Setting view

3.1.4 Overlaying Virtual Objects Using Sensor Data

The aim to overlay virtual 2- and 3-dimensional objects on top of real world objects requires a knowledge of the position of the head of the user, or rather the position of the glasses used, in relation to real world objects. This can be achieved in various ways and one part of it is knowing the rotation of the head. If the head rotation is known at all times then the virtual scene can be rotated around the head to preserve an illusion of static or moving virtual objects perfectly aligned with the real world. Knowing the rotation of the head is not enough though if the head moves spatially so knowledge of this movement will be needed as well.

The glasses have various sensors and among them are gyroscopes, accelerometers, magnetometers and video camera. Using a combination of these sensors, sufficient information can be attained to reliably determine the relative position of the glasses.

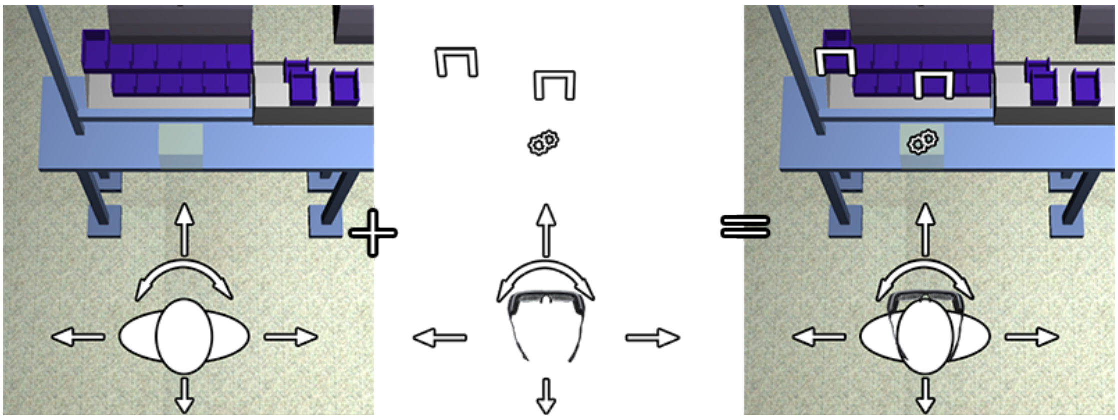


Figure 3.8: Left: Operator without glasses, Middle: Glasses visualizing virtual objects controlled by sensors, Right: Operator using glasses, seeing the real world with virtual objects overlaid.

3.2 Motion Capture

The need for capturing of the operators movements arose in order to tell the system when the operator had completed the given instruction. By tracking the hands of the operator, the system can know when the correct parts have been picked up by the operator. This module is essential to the system because it provides a way to proceed in the instruction sequence without the operator being forced to manually acknowledge every action, but it is not in itself a technology aspect that is in the scope of this thesis.

3.2.1 Hand Tracking

The hand tracking system used in this project was as previously mentioned developed in a previous project [1], the system allows getting reliable input of the operators hand movements. The hand tracking works by using a depth-sensor camera [10] positioned above the work area. The sensor data from the camera is fed into a recognition system that matches the sensor data to the human hand skeletal structure and thereby recognizes where they are and how they are positioned. The system allows for recognizing position as well as gestures performed using finger movements and hand rotations.

Using the positional data of the hands and their movements, decisions can be made by other modules of this project on how to proceed during a work sequence.

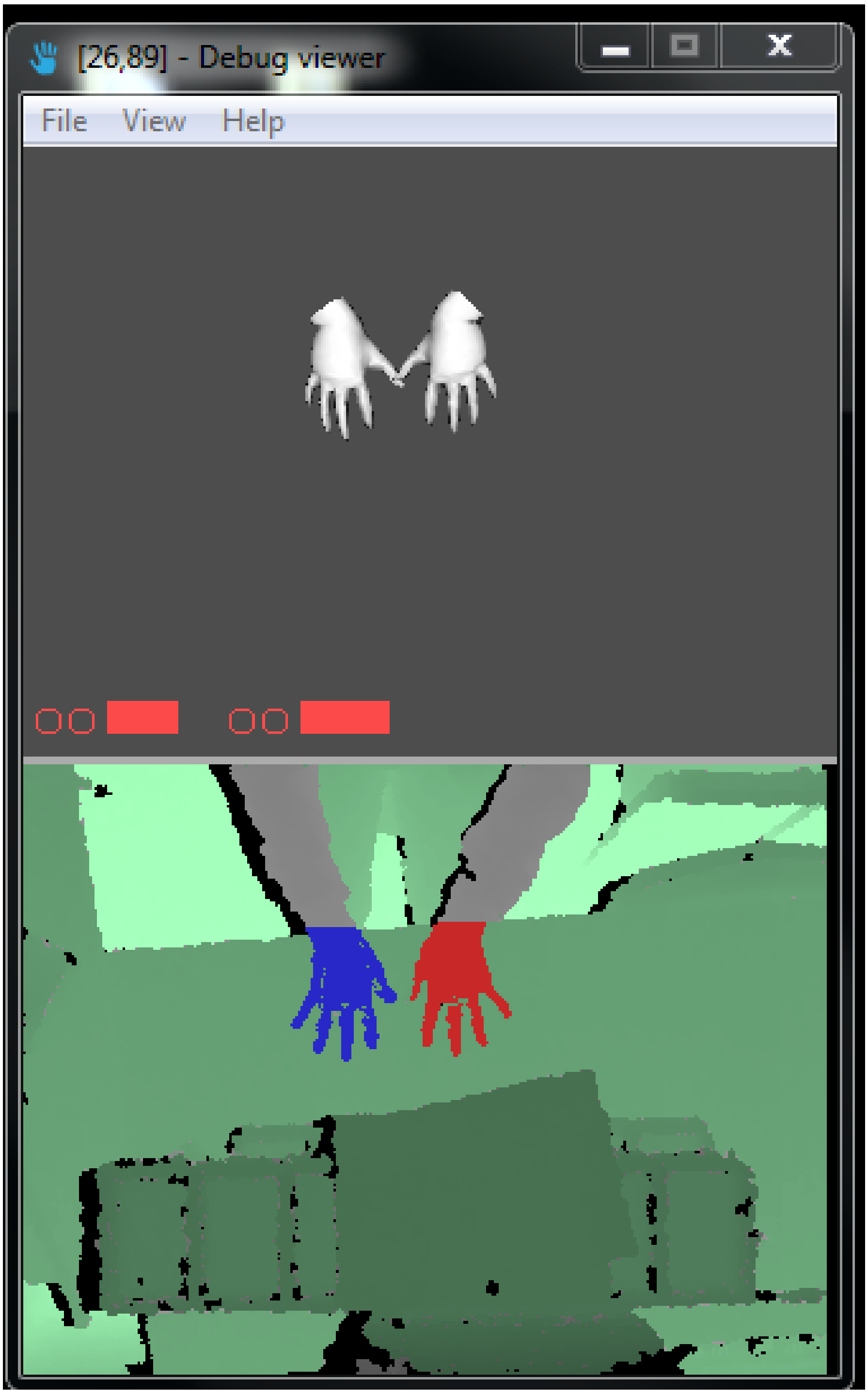


Figure 3.9: The lower image displays the users hands when they are identified by the recognition system, and the upper shows a 3D model built on hand and finger position and rotation data.

3.3 Instruction Handling

This module had the purpose of handling the sequence of instructions given to the operator. To do this, it had to have a way to store or access all the possible instructions and in which order they are to be given. By communicating with the hand tracking system the module could tell when an instruction was completed.

3.4 Operator Data

In order to store the operator data i.e. information on how long it takes to complete an instruction sequence and how often an user error occur, a system for handling users and interpret data from the supervising agent was needed.

3.4.1 Progress Recording

During use of the system, the data is analyzed and collected. Essentially out logs of successful and unsuccessful operations by the operator with timestamps as well as the user's interaction with the interface is logged. Together, a number of contextual factors are stored, such as operator ID and glasses position and direction through its sensors.

The reason for collecting this information is to allow for later analysis of the operators behavior during the assembly session. It is primarily interesting to examine the error rate and efficiency of the operator. The pairing of the data to an operator ID also allows for studying operator behavior over multiple sessions which could be useful to determine learning curve.

3.4.2 Data Storage

The platform uses persistent data (data that is stored in non-volatile memory such as hard drives) frequently for retrieving assembly details, storing task performance details and measurements. The following information is persistent:

Assembly sequence that forms assembly tasks in a way that supervising is possible.

Instructions for guiding an operator with graphics and text.

Measurements for evaluating the platform and the AR technology associated with it.

Assembly sessions to link measurements with performed assembly tasks.

Graphics for instructions are separated from the database and located in a web server. The reasoning behind this will be resolved in a relevant section on the **technical details** chapter.

Data for instructions and for tracking specific assembly actions (such as taking an assembly object) are managed by the system efficiently so that time delays do not interfere with the assembly task. Furthermore, the system manages different assembly tasks that can be updated (to better illustrate an instruction) or new ones added which results in a dynamic behavior of the data. Moreover, measurements are stored in a way that relations between measurements are maintained and that statistic analysis can easily be done. A database achieve these requirements and therefore a database system was used for maintaining this data.

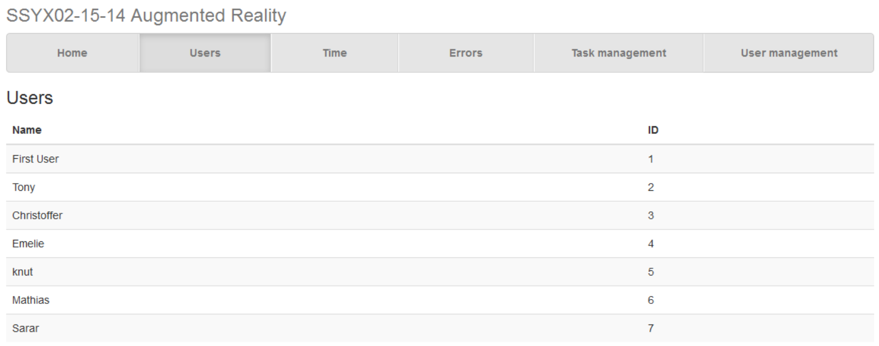
3.5 Front-End Application

The front-end application is mainly developed in order to view the measurement data stored in the database but it also has functionality for starting/canceling an assembly session and creating users which gets stored in the database this is displayed in a web application.

The data deemed relevant for the evaluation of the platform are the errors made by the user and time elapsed per test session.

Users Tab

The users tab lists all of the users currently in the database as well as their ID, see 3.10



SSYX02-15-14 Augmented Reality

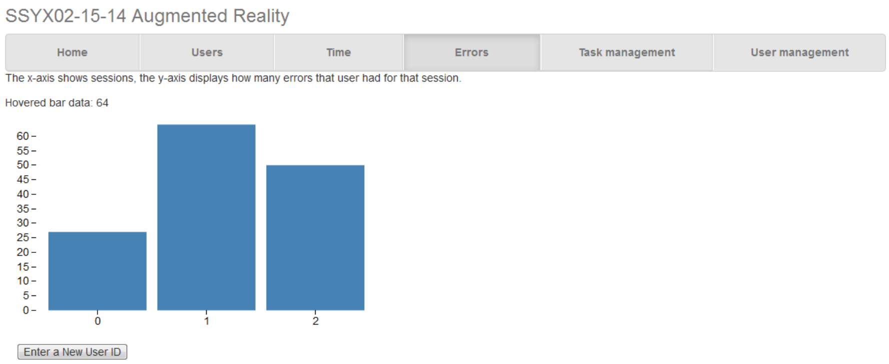
Users

Name	ID
First User	1
Tony	2
Christoffer	3
Emelie	4
knut	5
Mathias	6
Sarar	7

Figure 3.10: The web application displaying the users in the database.

Errors Tab

When entering the errors tab a prompt will pop-up and ask the user to enter an ID of a user to view the total number of errors per session in a bar chart, see 3.11.



SSYX02-15-14 Augmented Reality

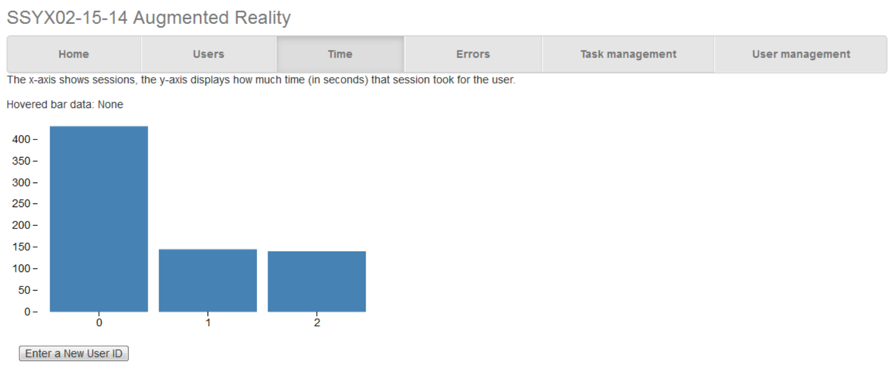
The x-axis shows sessions, the y-axis displays how many errors that user had for that session.

Hovered bar data: 64

Figure 3.11: The web application displaying the amount of errors a user has for each session.

Time Tab

When entering the errors tab a prompt will pop-up and ask the user to enter an ID of a user to view the time taken per session in a bar chart, see Figure 3.12.



SSYX02-15-14 Augmented Reality

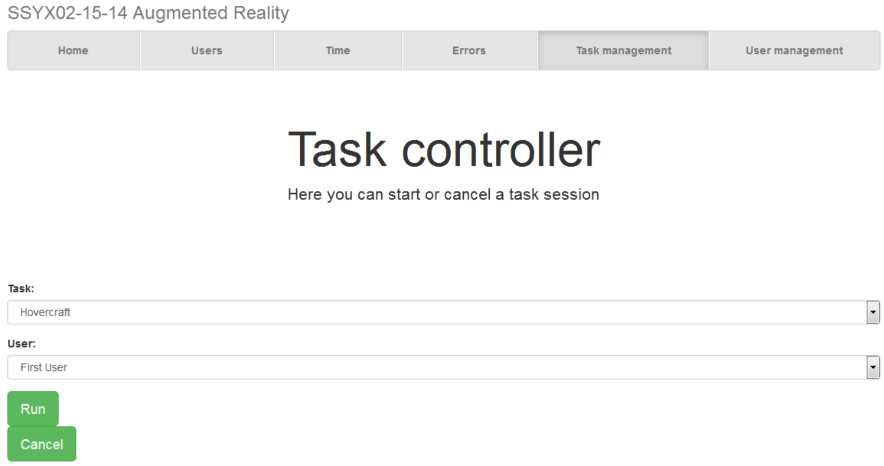
The x-axis shows sessions, the y-axis displays how much time (in seconds) that session took for the user.

Hovered bar data: None

Figure 3.12: The web application displaying the time taken for each session.

Task Management Tab

Here various options can be set before running the program. A pre-created user can be selected, as well as a task to perform. Two buttons are also available, they can either start a test session with the Run button, or cancel an already running session, see Figure 3.13, Figure 3.14 and Figure 3.15.



SSYX02-15-14 Augmented Reality

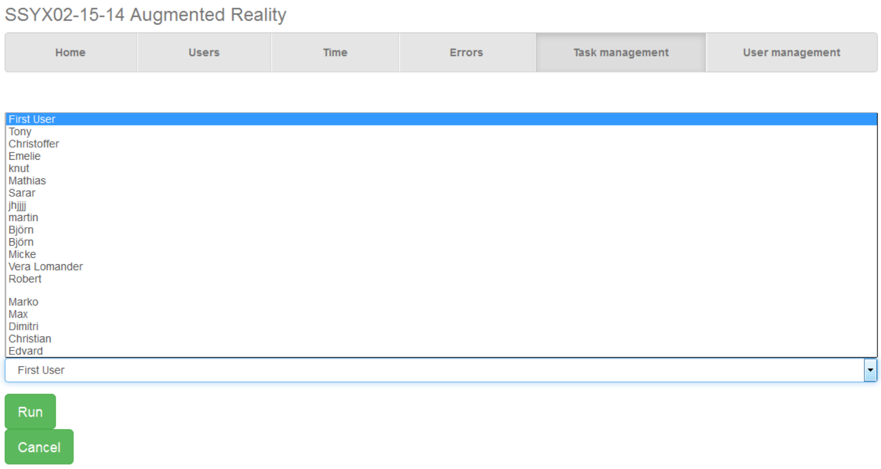
Task controller

Here you can start or cancel a task session

Task: Hovercraft

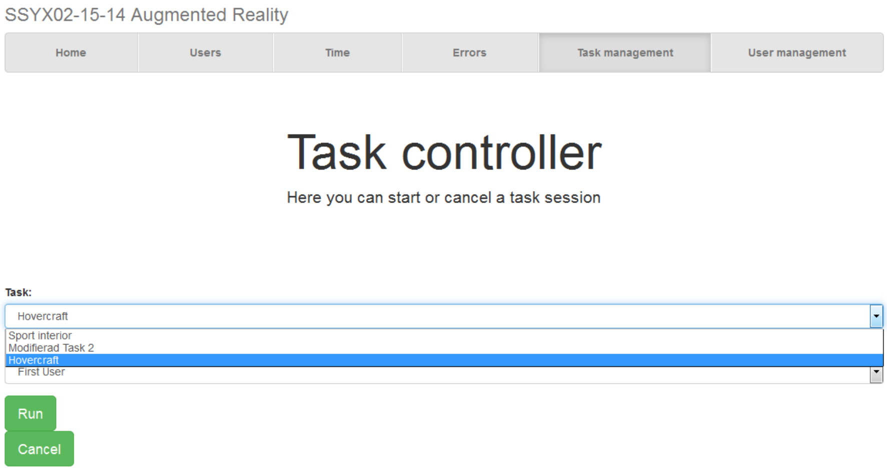
User: First User

Figure 3.13: The web application displaying an overview of the Task management tab.



SSYX02-15-14 Augmented Reality

Figure 3.14: The web application displaying the menu for choosing which user to use for the session.



SSYX02-15-14 Augmented Reality

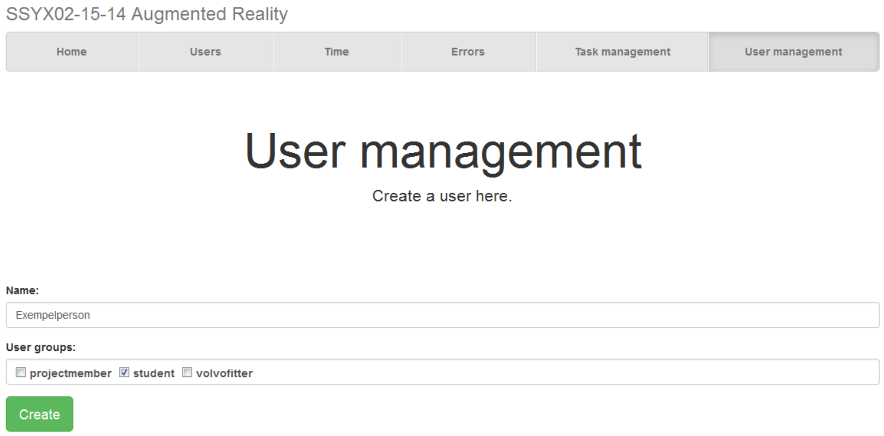
Task controller

Here you can start or cancel a task session

Figure 3.15: The web application displaying the menu for choosing the different assembly tasks.

User Management Tab

Here new users can be created, they can also belong to one of three groups, to assist in easier navigating the different users, see Figure 3.16.



User management

Create a user here.

Name:
Exempelperson

User groups:
□ projectmember ☑ student □ volvofitter

Figure 3.16: The web application displaying the different options for creating a user with name and user group.

4

Implementation

THE following chapter will describe the methods and tools used for implementing the modules formulated in the previous chapter as well as technical details of the platforms hardware.

4.1 Software for AR-glasses

Since the source code for the previous project was available, injecting newly developed methods into the old system code allowed data extraction for interesting parts of information that were needed to drive the new applications. These new methods extracted the following: placement and existence of virtual objects in the computer generated scene, textual instructions given to the user, expected timing of the instructions to be carried out, and information about images being used. This extraction happens in real-time while the application is running.

4.1.1 Server component specifics

The project uses Unity3D as a visualization platform. Working with this allows modeling the virtual geometry to fit real world measurements which is a key component of performing a believable overlaying of virtual objects into the users' field of view. The server component uses the aforementioned extracted data to keep an exact copy of the virtual world geometry that has been created by the work process sequence. This world or parts of it is relayed to the client in the correct way depending on the client specifics and its' capabilities.

The server uses networking components from the Unity3D SDK to communicate with the client application. The networking components allow for exact mirroring, of all actions, between server and client. All created objects can and should be assigned a network ID number that will uniquely identify that object if network manipulation of said object is desired. All IDs can be assigned to groups if similar behavior is needed for multiple objects. Using these ID numbers or group numbers, object and geometry transforms are sent via reliable delta compressed streams over the network.

Remote procedure call (RPC) (RPC) is used for inter-process communication. In this project between the AR-glasses client and the server. Single- and multi-target RPCs are used for invoking any wanted or needed method calls both from server to client and vice versa.

Since the server does all decision-making regarding the placement and existence of objects it keeps a record of the type of client that is currently connected, since the client type decides which instruction set is used when calling RPCs. The client type is locked during the execution of a work sequence because a change in type during operation could potentially cause scene corruption.

The reason for the corruption possibility is that all RPCs sent from the server are fully buffered during the execution of a sequence. This is a design feature. The buffering allows a client to resume the sequence in case of networking failures. As soon as a client reconnects after a failure, the entire RPC buffer is sent to the client so that it is synchronized with the server. The corruption occurs due to the different RPC methods needed by the different client types.

4.1.2 Client component specifics

Since much of the real-time information processing was offloaded to the server, the network stream between client and server requires very little bandwidth. Measurements performed during normal operation showed that a bandwidth of 500 bytes/s was sufficient to transfer data.

4.1.3 Fiduciary markers

As mentioned earlier, an important aspect of creating a believable overlaying of virtual objects on top of a real world scene requires the virtual components to remain spatially and rotationally synchronized with the real world, i.e. the virtual objects should stay in place vs. the real world if the observer moves around. To try to achieve this in the 2/3-D client component, two primary techniques were used.

The first and in this project the most important one was using so called fiduciary markers for image target recognition, and the built-in camera of the glasses to record the marker positions in the real world.

These markers are placed at strategic locations and in this project were used as position and rotation guides for placing the virtual objects into the scene.

The fiduciary markers, or image targets, that are used, hold certain properties that makes them suited for different image processing algorithms. What they hold in common is that they contain areas that an algorithm can find fairly easy and accurately, most often using a camera image as input.

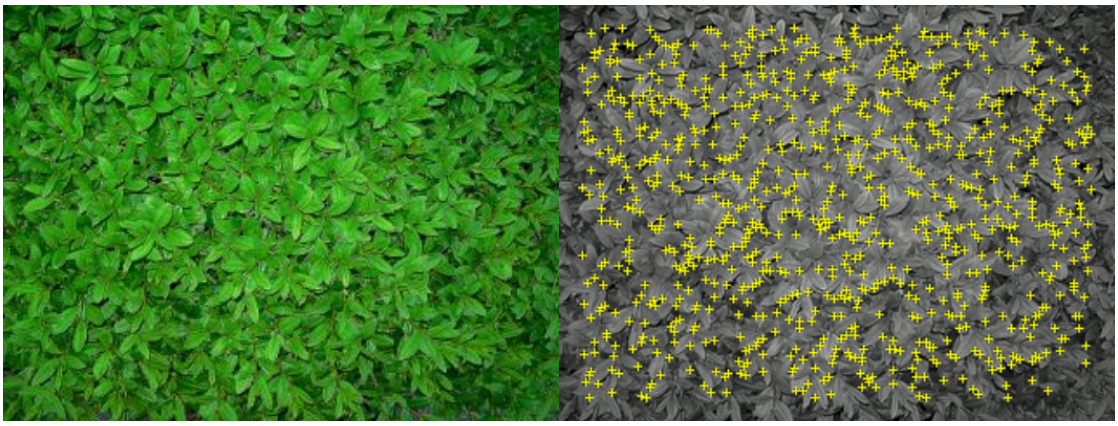


Figure 4.1: Visualization of the properties of an image target (left image). Yellow crosses (right image) represent uniquely identified areas in the image that the image processing algorithm can find.

Several software development kits exist that incorporate image target recognition, for example ARTag, ARToolkit and Qualcomm Vuforia. Qualcomm Vuforia SDK, from now on referred to as the AR-SDK, was chosen as it already had support for Unity3D, mobile solutions, and beginning support for digital see-through eyewear at the time of choosing. This again allowed more focus on the aim of the project and not on developing new technology.

By placing an image target at a known real world location and referencing this image target in in the virtual scene, the two can be merged in the visualization overlay and thereby create a stable reference point for the whole AR experience. This works as long as the image target is in view of the camera built in to the glasses used. If or when the image target goes out of view of the camera then the referencing solution breaks and at the same time breaks the AR illusion. Several image targets could be used to remedy this situation but that solution was not explored fully in this project due to time constraints.

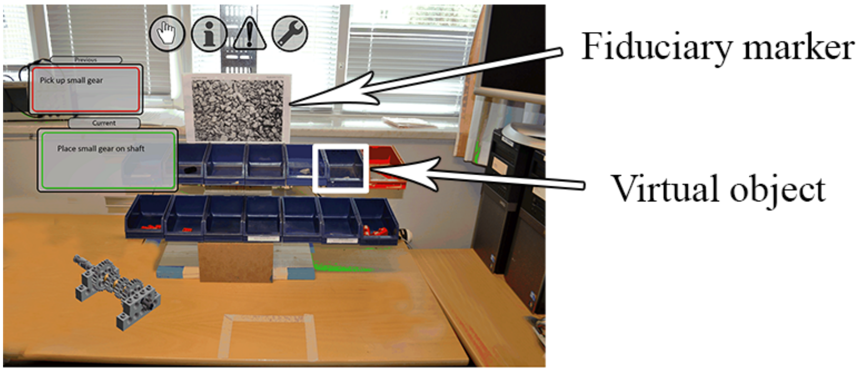


Figure 4.2: An example of a real world workspace with an image target in place and a virtual object referenced to this image target.

4.1.4 Gyroscope and accelerometer

Another possible solution to keeping a stable reference point between real and virtual components is to use the built-in gyroscope and accelerometers of the device to keep an internal reference point of the glasses in sync with real world rotation.

The gyroscope is a device that can measure angular velocity (denoted as ω) very accurately and these can be sampled at a very high rate. Angular velocity is the rate of change in the angle (θ) over time (t).

$\omega = \dot{\theta} = \frac{d\theta}{dt}$

By integrating the angular velocity over time it is possible to get the current angular position against a reference,

$\theta(t) = \int_{t=0}^{t} \omega(t) dt, \theta(0) = 0$

Since we are working with a digital system the angular velocities are sampled and non-continuous, so a small error may be introduced as events happening between the samples are ignored. With a sampling period of T_s this becomes,

$\theta(t) \approx \sum_{t=0}^{t} \omega(t) T_s$

which is an approximation of the real angle. The introduced error is called a drift error and it will increase over time. The drift error can be made smaller over a specified period of time by taking samples more often but will most likely not become zero when the gyroscope is returned to its initial position, and this can become problematic due to the way the angle is used in the project, virtual objects are not projected at the correct spot in the real world when trying to use this θ as the only input which will in turn break the illusion and functionality of the projection. As a conclusion the gyroscope is very reliable and fast in the short term, but less so in the longer term.

A way to combat this is using different types of sensors and combining the information that they give. As mentioned earlier, the device used in the project also contains an accelerometer. The accelerometer measures proper acceleration and as an example an accelerometer at rest, on earth, will measure an acceleration of $g \approx 9.8\ m/s^2$. The accelerometer in itself is an extremely sensitive device and raw data from it can be full of spikes and large anomalies. For this reason the data sampled from the accelerometer was passed through a developed low-pass filter to get rid of some of the anomalies and ”smooth-en” the signal. This smoothing has the drawback of making the accelerometer data very reliable in the long term, but less so in the short term.

Now by combining the measured data from both an accelerometer and a gyroscope the benefits of fast reliable data in short time frames from the gyroscope will be corrected in the longer time frame by the accelerometer data to minimize the drift error. Vertical drift is eliminated by this method but there is still the issue of horizontal drift.

A combination of the two techniques were used for the following reasons: Image target positioning was primarily used because it provided an exact, reliable and fast reference system for both rotational and spatial movement. Gyroscope and accelerometers were used for when or if the image target went out of sight from the camera. This allowed for at least an approximate reference until the image target referencing could be reestablished. The gyroscope solution could not be used for spatial referencing, and not for too long for rotational referencing.

4.1.5 Image target positioning, self-calibration

The image target placement in the operator workspace is extremely important due to being the most precise reference between the real world and virtual scene. Because the operator workspace might be dynamic regarding the placement of materials and tools, and the image target might have to be moved, a self-calibration system was developed to make sure that the reference is in sync.

The calibration works by placing a secondary image target on a known position of both the real world and the virtual scene. For example the placement of a box of parts. Then by making sure that the camera has both the primary or normal image target, and the secondary image target in view, the placement coordinates of the primary target are calculated and saved in the application.

These calculations can be performed because the sizes of the image targets are known, and the AR-SDK can precisely measure the angles and distances between them.

One initial problem that occurred was that the measurements are so precise that small vibrations or movements of the image could influence the measurement.

This was solved by passing the measured distances over a period of time through a low-pass filter which smoothed the measurements. Using this technique the correct placement coordinates could be obtained with a precision of about +/- 1 mm, and a measurement time of about 10 seconds, under normal conditions.

4.1.6 Depth occlusion

During development it was discovered that an important aspect of incorporating virtual objects into a real world scene is to try to make the objects blend into the real world as good as possible. This allows for a less intrusive or jarring visualization, or a more “natural” experience.

If one object occludes the view of another object then the first object is perceived as being closer to the viewer. A method was developed to mimic this perceived depth occlusion, so that real world objects that should be in front of a projected virtual object actually appears as if it indeed is in front of said object.

Simple graphical markers were used to point to a specific area during the execution of a work sequence. These markers, when not using the depth occlusion methods seemed to lay flat and unrealistic in an undefined warped space in front of what they were supposed to point to.

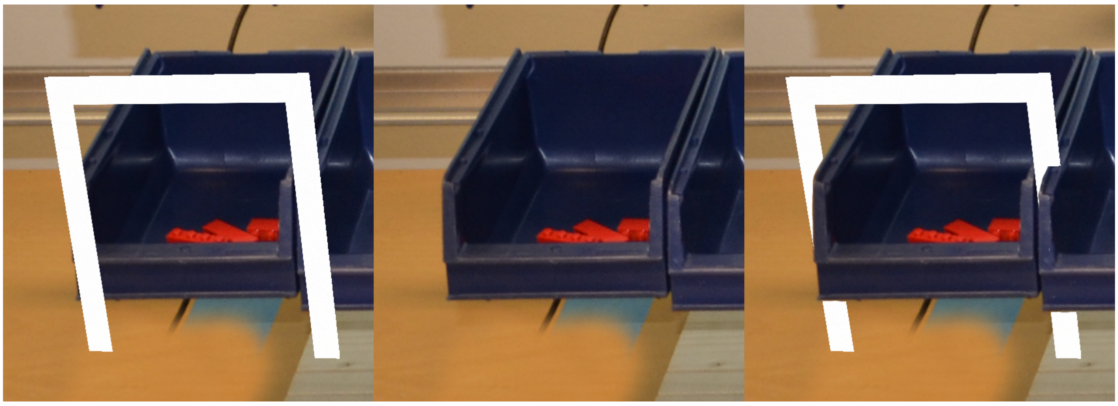


Figure 4.3: Left: Simple marker without depth occlusion applied. Middle: Real world scene, Right: Simple marker with depth occlusion applied.

By applying the developed depth occlusion function, a more realistic and viewer friendly overlay could be performed.

The depth occlusion works by inserting invisible objects into the virtual scene that have the same geometry as the real world object that is supposed to do the occlusion. Though the objects are invisible in the virtual scene their property is changed so that they occlude virtual objects. The technique works well if positions and geometries of real world objects can be modeled into the virtual scene, using said invisible objects.

4.2 Database

The database used for maintaining persistent data is called *MySQL* and is a relational database.

The database scheme contains several entities and relations, therefore only a subset of the entities are shown at a time describing their purpose (for a complete diagram see figure 1 in appendix A). The entities can be divided in the following three main groups:

Model of an assembly task that holds details for the different assembly sequences existing which is crucial for supervising progress in the assembly task and instructing an operator

Assembly session that provides details such as user or the status for an ongoing or performed task.

Measurements which is different time and error measurements for the assembly sessions.

In the following subsections each of the three entity groups will be described individual and in detail.

4.2.1 Model of an assembly task

An Enhanced Entity–Relationship model (EER) diagram over the entities that represent the model of an assembly task will be shown, followed by a ”top-down” description:

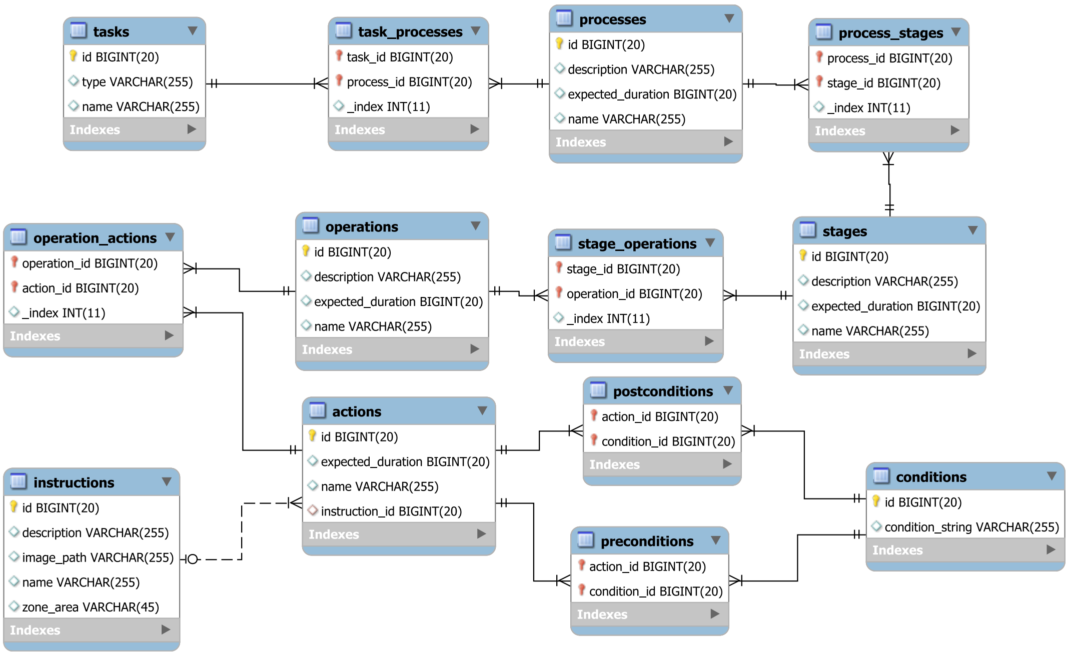


Figure 4.4: EER diagram over the model of an assembly task

Basically, an assembly task is built up of a sequence of *Actions* (which will be described later), where the sequences are divided in a hierarchy with the levels **Operation**, **Stage** and **Process** as seen in figure 4.4. This hierarchy was chosen to follow the structure of the previous system to avoid otherwise potential issues with the communication to the hand tracking module. In the following two subsections, the assembly task model will be examined in the perspective of supervising and then in the perspective of instructing an operator.

An assembly step

The entities relevant for supervising a step in an assembly task can be illustrated in the following figure:

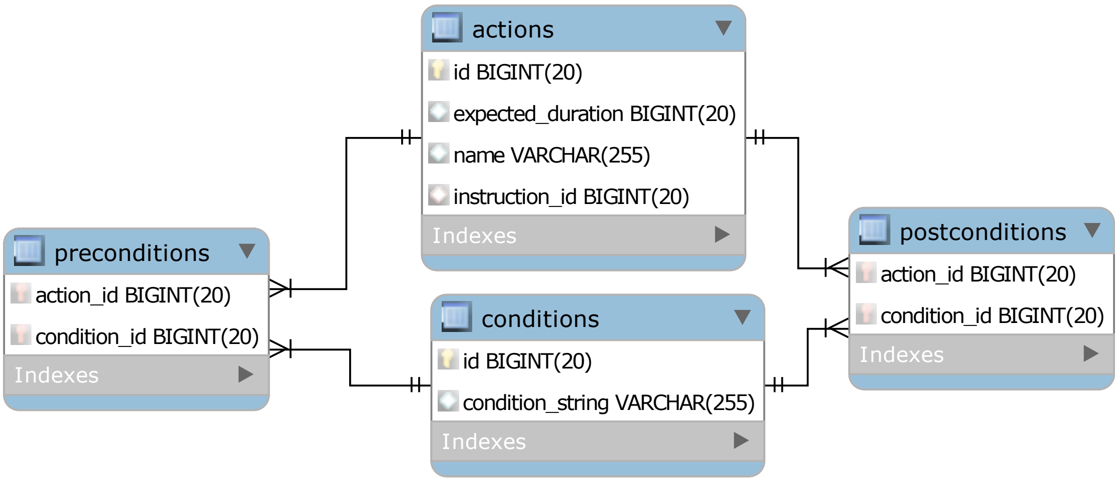


Figure 4.5: EER diagram with entities describing an Assembly step (*Action*)

A *Condition* is the lowest meaningful event being tracked by the hand-tracking module. In the most cases, a *condition* describes an event of a hand combination entering/exiting a zone in the assembly station. However, it exists other special *conditions* such as both hands joining each other (this *condition* is expressed only with one attribute) and this is why the *conditions* entity is not split in more attributes. Instead a string with a special formatting is used to represent the whole *condition*. Moreover, an *action*, describes the events being tracked by the supervisor and is the type of activity reported by the hand-tracking module. Furthermore, a set of *preconditions* and *postconditions* forms an **Action** and is the lowest step in an assembly task.

An operator instruction

The entities interesting for providing an operator with instructions can be seen in the following figure:

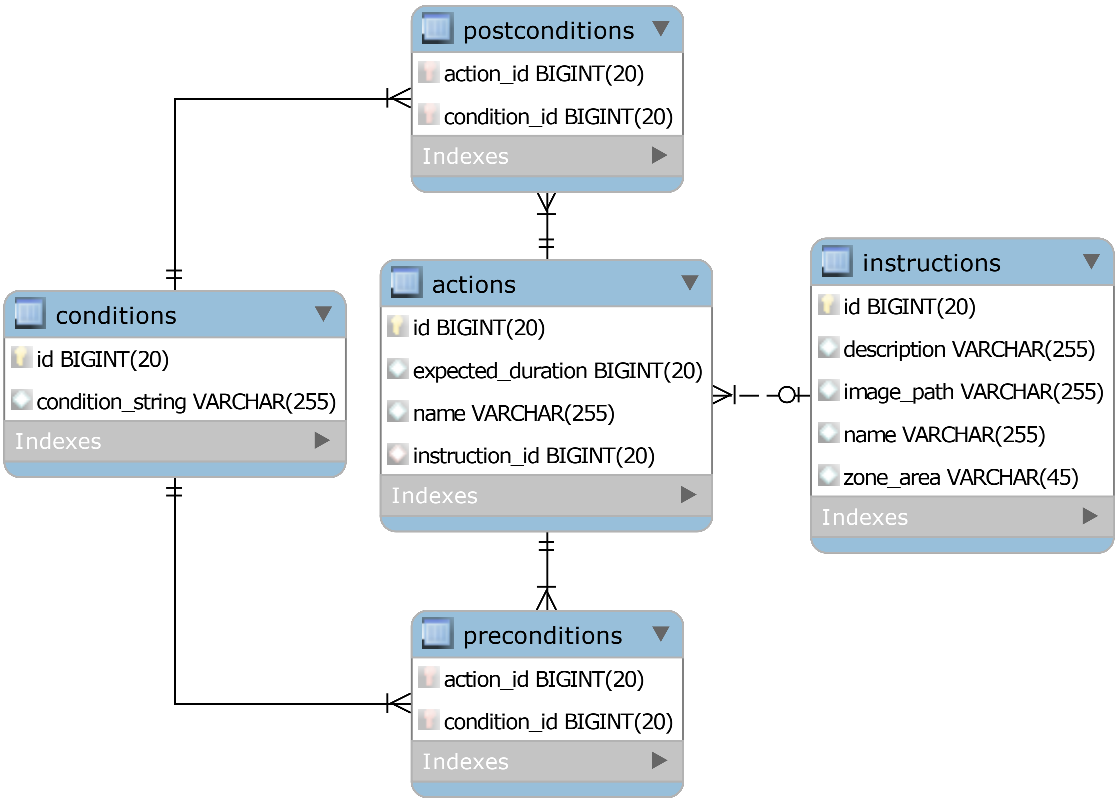


Figure 4.6: EER diagram with entities describing an Instruction

An *instruction* contains both textual and graphical data to instruct an operator for a given *action*. The conditions are in this case of interest to provide the GUI with necessary information on which objects in the virtual world is relevant for the *instruction*. Note that, as mentioned earlier in the previous chapter, that the graphic content itself is not included in the entity. Instead a path of where the graphic resource exists (in a web server) is provided.

4.2.2 Assembly session

An assembly session basically connects a task with a status and a user as following:

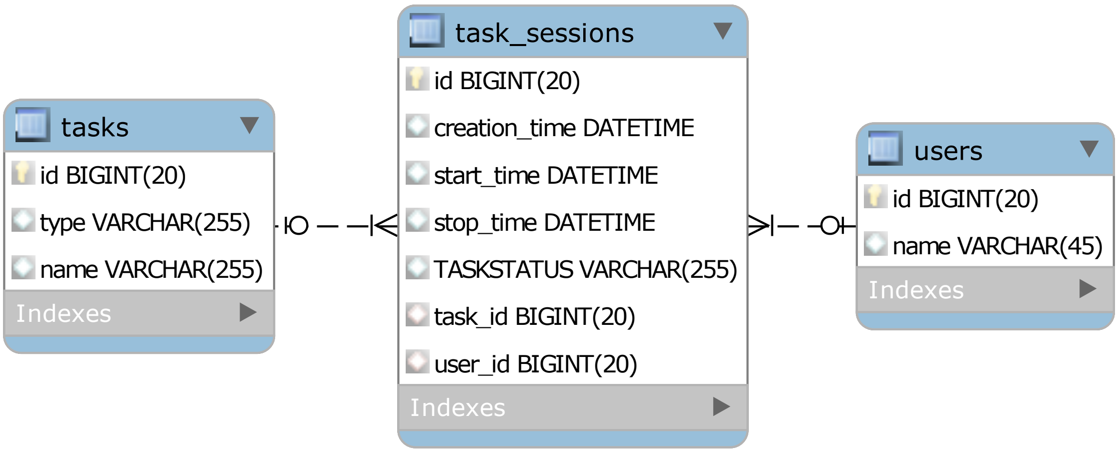


Figure 4.7: EER diagram over an assembly session

A *Task session* differs from the other entities as it is the only entity holding a state. It provides information on the status of the performance of a task (such as *running*, *canceled* or *finished*), which task and what operator. A *user*, in this case, is the operator performing the task.

4.2.3 Measurements

The platform also perform measurements on the task sessions for evaluating purposes. The relevant entities used for measurements are illustrated in the following figure:

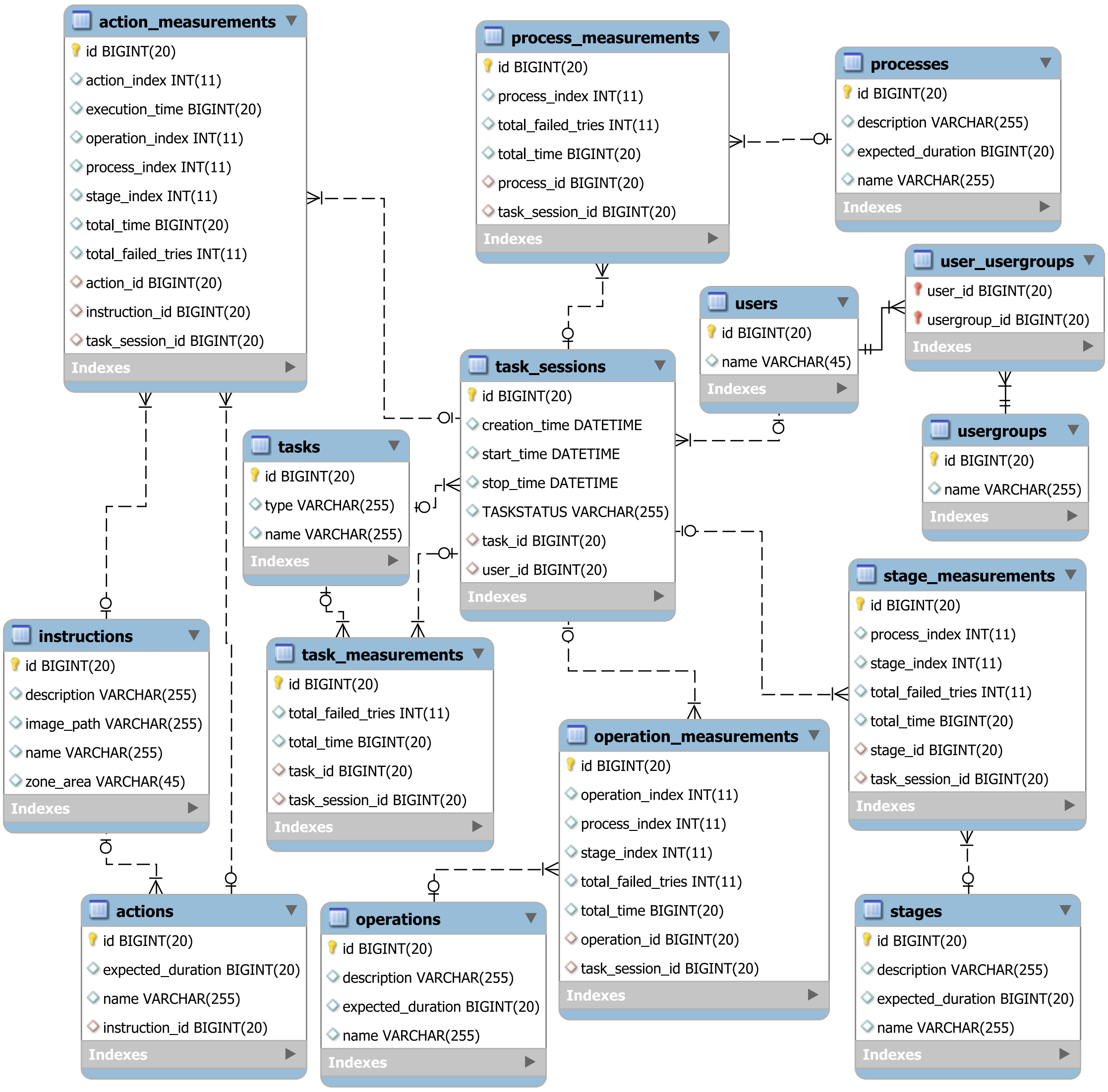


Figure 4.8: EER diagram over measurements

Basically, each level in the assembly task has a corresponding measurement entity that record times and number of wrong activities for a session. Note also that the entities defining an assembly task contains fields for the expected duration which is relevant as it can be compared with measured times. It also exists *User groups* for users as seen in the figure and its purpose is to make analysis on the results based on different *User groups* (such as *experienced operators* or *beginners*) possible.

4.3 Application Server

This system is responsible of controlling the assembly process and presenting statistical results. Its purposes will be illustrated in the following figure (for a more detailed model see figure 4.14) and then described:

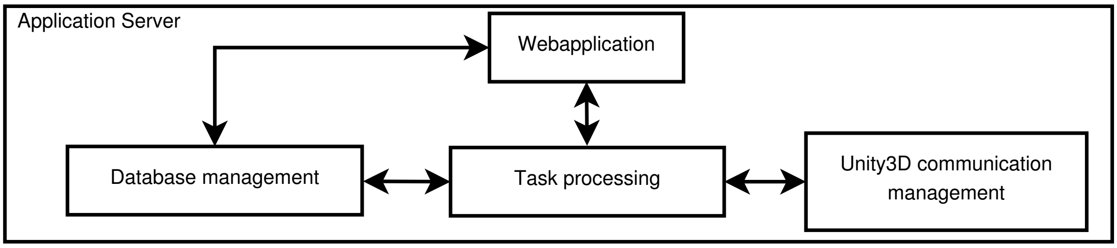


Figure 4.9: Simplified overview of the Application server

The main purpose for this system is to process through an assembly task to provide an operator with instructions. However, displaying instructions and tracking an operator's action occurs, outside this application, by the Unity3D application and therefore a way of communicating with it is crucial.

The Web application seen in figure 4.9 provides an interface for managing users, to initiate or cancel an assembly task and to navigate over different presentations of statistical results. It also works as a resource provider for binaries of grater size which is considered the case for instruction images.

Tasks, users and measurements are persistent data which is maintained by a database outside this system. Therefore, a mechanism for communicating with the database is required in order for the web application and the task processing to function.

All this will be described next and thereafter summarized.

4.3.1 Communicating with database

The communication with the database is done through the Java Persistence API (JPA), an Application Programming Interface (API) that abstract over different database concepts among other things. An advantage of using JPA is that connections to the database is entirely managed by the JPA provider where connection details are provided to the application as configurations. However, the reason

JPA was chosen is because it provides an efficient way of mapping Plain Old Java Object (POJO) to relational data. This type of mapping is commonly refereed as Object-Relational Mapping (ORM).

The relation between a java object and relational data will be described in the next section, followed by another subsection describing how the actual ORM mechanism works in JPA.

The JPA model of the database

Database entities can be represented by Java classes where a Java class, from now on refereed as a JPA entity, is associated with a table(entity) in the database while object instances of a JPA entity are entries(or rows) of the associated table.

The JPA entities has very similar structure of the database entities(illustrated on figure 1 in Appendix A) and to show this, an Unified Modelling Language (UML) class diagram is shown next:

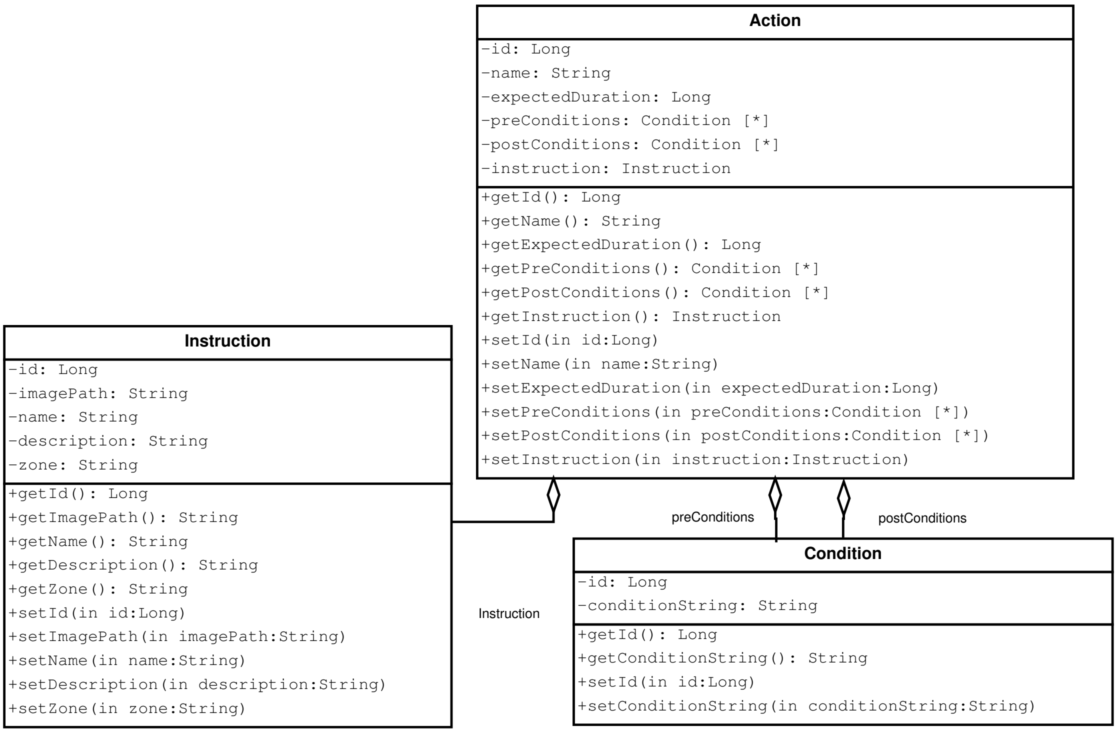

Action
-id: Long
-name: String
-expectedDuration: Long
-preConditions: Condition [*]
-postConditions: Condition [*]
-instruction: Instruction
+getId(): Long
+getName(): String
+getExpectedDuration(): Long
+getPreConditions(): Condition [*]
+getPostConditions(): Condition [*]
+getInstruction(): Instruction
+setId(in id:Long)
+setName(in name:String)
+setExpectedDuration(in expectedDuration:Long)
+setPreConditions(in preConditions:Condition [*])
+setPostConditions(in postConditions:Condition [*])
+setInstruction(in instruction:Instruction)
Instruction
-id: Long
-imagePath: String
-name: String
-description: String
-zone: String
+getId(): Long
+getImagePath(): String
+getName(): String
+getDescription(): String
+getZone(): String
+setId(in id:Long)
+setImagePath(in imagePath:String)
+setName(in name:String)
+setDescription(in description:String)
+setZone(in zone:String)
preConditions
postConditions
Instruction
Condition
-id: Long
-conditionString: String
+getId(): Long
+getConditionString(): String
+setId(in id:Long)
+setConditionString(in conditionString:String)

Figure 4.10: An UML class diagram showing a subset of the JPA entities

As seen by the figure, JPA entities share many similarities with database entities and the need of JPA entities might still not be apparent. One way of seeing the purpose of JPA entities is that they mirror the database entity concept with the goal to abstract over the database itself. In fact, the database entities were generated from the JPA entities automatically (by configuring the JPA provider to do this).

Accessing the database

As the JPA entities mirrors the database entities, they reveal the database structure to the JPA provider but they do not really provide any functionality for accessing the database entries. For this, entity managers are used and can be seen as classes providing methods for navigating over the entities. These methods basically executes database queries and therefore defines the actual database logic.

4.3.2 Communicating with Unity3D

The communication between the Application server and the Unity3D server is done by passing asynchronous messages between them through the network. The protocol and the structure of the message, from now on referred as *Esbmessage*, was designed in the previous project also for communicating with Unity3D. It is based on the TCP/IP protocol using a Client/Server architecture. Furthermore, Apache camel, a routing framework, was used for routing the messages to the correct destination.

This communication protocol was reused in this project to avoid potential issues with writing network protocols and support backward compatibility with the Unity3D application designed in the previous project. The next three subsections will describe the structure of an *Esbmessage*, transmitting *Esbmessages* and routing *Esbmessages* individually in more detail.

Structure of ***Esbmessage***

The *Esbmessage* is made up of two elements *head* and *body*. The *head* element holds information about three important details which are the following:

Sender which tells who sent the *Esbmessage*.

Type that is the topic or the group of receivers for the *Esbmessage*.

Timestamp that is the time when the message was issued.

The body, on the other hand, is the actual content of the message and consist of set of values associated to keys. The value of the key (in the *body* element) named *Type* is the important one as the receivers decides how the message will be processed by this value.

Transmitting an *Esbmessage*

In order to transmit an *Esbmessage* object, it needs to be transformed into a sequence of bytes before it is sent and upon receive transformed back into an *Esbmessage* object. This transformation was done using some interconnected encoders/decoders according to the following figure:

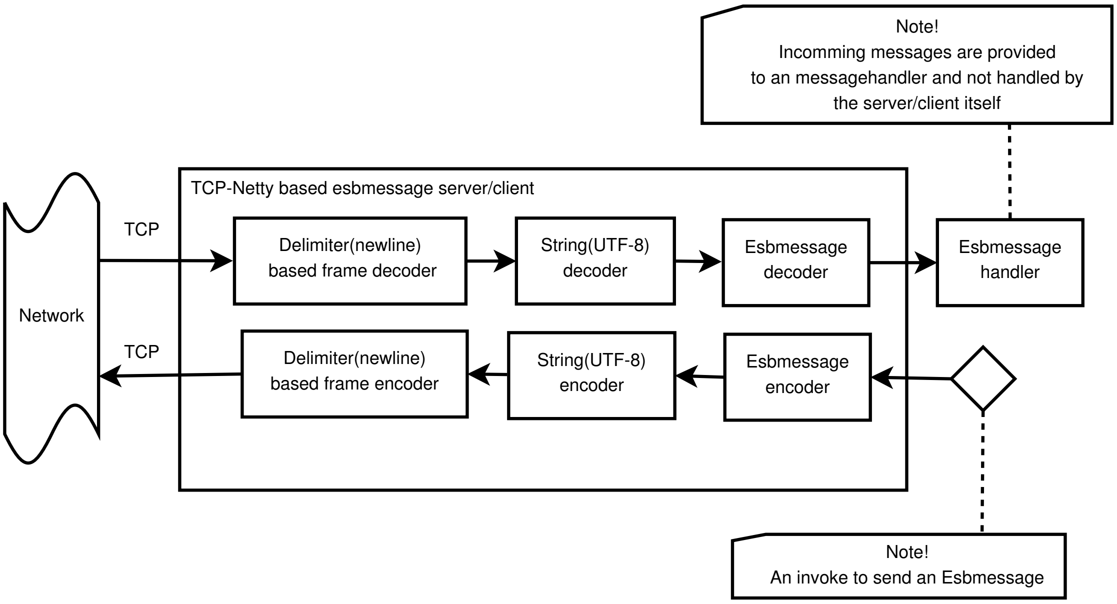


Figure 4.11: Esbmessage server/client overview

The delimiter based frame decoder is responsible for splitting incoming data into frames limited by a delimiter (newline character in this case) maker while the delimiter based frame decoder simply adds the delimiter marker at the end of each message. The string encoder/decoder simply decodes/encodes a java string according to an encoding format (UTF-8 in this case). Furthermore, the *Esbmessage* encoder transforms an *Esbmessage* into an XML document while the *Esbmessage* decoder parses an XML document into an *Esbmessage* instance.

Routing

For routing, as mentioned earlier, Apache camel is used and every *Esbmessage* is sent to this module which determines the destination and forwards the message. In Apache camel, routes can be configured in different ways and one way of doing it is to have routes with network destinations (Apache camel can also communicate with Java beans) based on a specific field in a XML document. As *Esbmessages* are encoded as XML documents, Apache camel was configured with routes where the destination was determined by the field *type* in the *head* element of the *Esbmessage*.

However, routing of *Esbmessages* to the Unity3D application cannot be done directly as that application has no server for receiving *Esbmessages*, Instead an Unity3D client connects to an proxy or bridge that consist of two servers where the first server forwards messages from Unity3D to the routing module while the other forwards messages from the routing module to the Unity3D client. This is illustrated with the following figure:

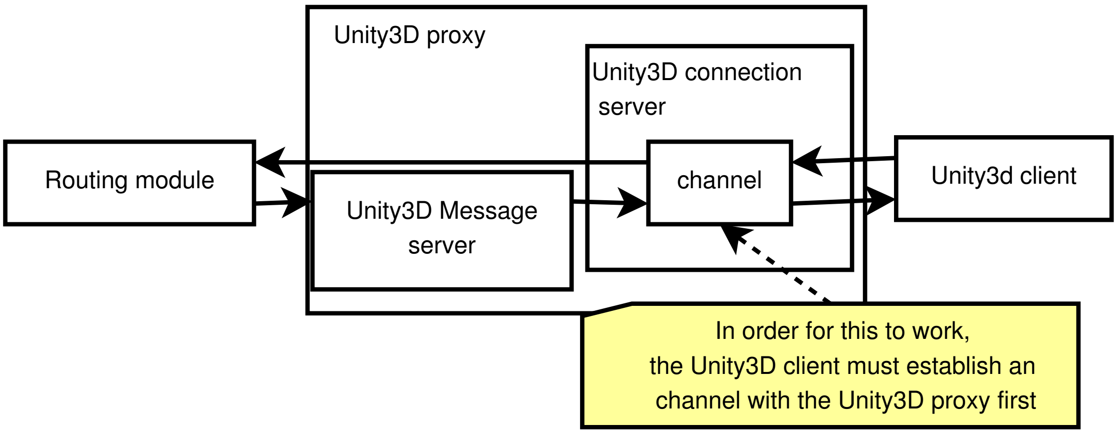

Unity3D proxy
Unity3D connection server
Routing module
channel
Unity3d client
Unity3D Message server
In order for this to work, the Unity3D client must establish an channel with the Unity3D proxy first

Figure 4.12: Illustration of how the Unity3D proxy works

4.3.3 Performance of an assembly task

When an operator performs an assembly task, the platform processes a virtual model of that task, found in the database, to keep track of the assembly progress so that the operator can be provided with correct instructions at correct occasions. However, in order for this to work, the virtual task needs to be synchronized with the actual task and therefore it needs a way of being notified when the operator completes an instruction.

Task processing

The processing of an assembly task basically works like a state machine where the state represent the current assembly step being performed and the virtual task itself forms the state machine by defining the possible states and how they are linked together. The state machine changes state from one assembly step to the next step in the assembly task or it cancels and changing to a state representing the end. Furthermore, this state machine has a set of supervisors associated with it and these supervisors will be notified with event messages when any state transition occurs. This is illustrated with the following example sequence diagram illustrating the state transitions:

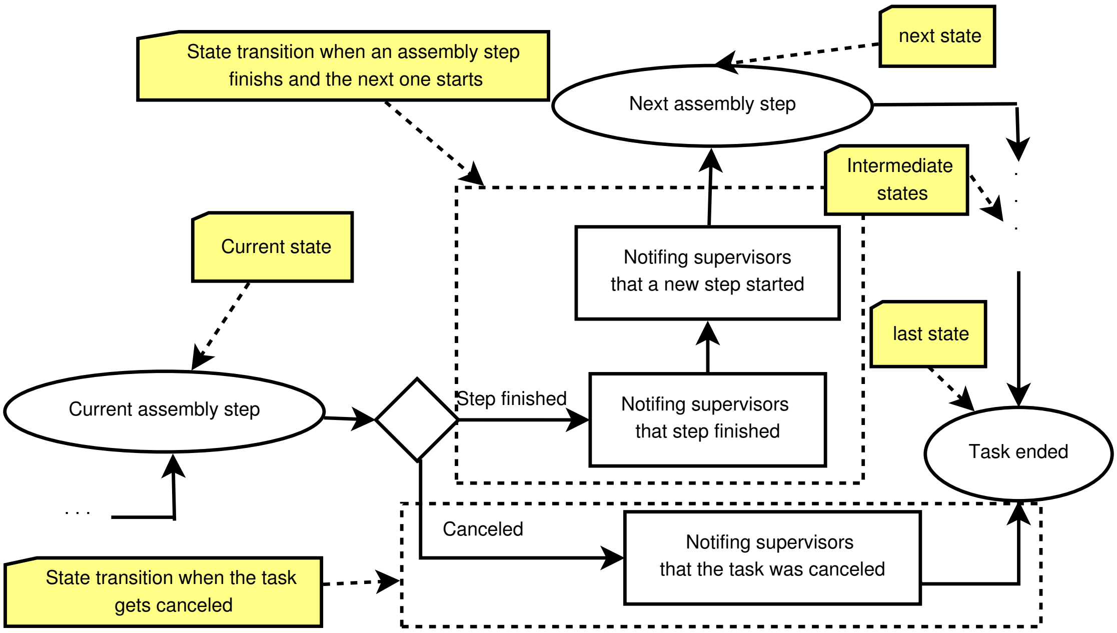

State transition when an assembly step finishs and the next one starts
next state
Next assembly step
Intermediate states
Current state
Notifing supervisors that a new step started
last state
Current assembly step
Step finished
Notifing supervisors that step finished
Task ended
. . .
Canceled
Notifing supervisors that the task was canceled
State transition when the task gets canceled

Figure 4.13: Sequence diagram illustrating how state transitions occurs on an assembly task

Task feedback and operator instructing

As mentioned earlier, in order to provide feedback to the virtual task, the system needs a way of being notified when the operator completes an instruction. The Unity3D system is responsible for tracking operator activities and sending these notifications. But it is also responsible for displaying instructions to the operator and therefore needs a way of receiving instructions. Furthermore, in order for the Unity3D system to know when an instruction is complete, it needs to know what action completes that instruction. The instruction sent to Unity3D already includes this action as it is needed for the GUI (to determine the virtual objects associated with the instruction). A supervisor observing the virtual task is used to send the instructions and furthermore, this same supervisor also receives the notifications about instruction completion to provide the feedback by changing the state of the virtual task to the next state.

Measuring on the task session

For doing measurements a supervisor observing the virtual task is used to calculate the times taken for completing the assembly steps. However, it also counts the number of wrong actions done by the operator. Therefore, this measurement supervisor needs notifications on all actions performed by the operator during the session in order to determine the number of errors. Whenever the operator performs an action, the Unity3D system also sends notification about this as it tracks the operator activities. The measurement supervisor also receives these notifications to calculate the errors and the results are persisted.

4.3.4 Front-End Application

When developing the web application our supervisor suggested us to use AngularJS and after some research that's what we went with. The front-end development is developed using AngularJS, the front-end framework Bootstrap and HTML. Bootstrap is a framework that aims to ease web development consisting of HTML- and CSS-based design templates and optional JavaScript extensions. To display the dynamic data from the database a bar chart written in D3.js was chosen, based on its' many already finished templates.

A Java Servlet can be seen as a component running in a web server serving a resource for clients and these Servlets share many similarities with dynamic pages such as PHP. However, the programming environment for a Servlet makes it preferred when most of the content in the response are dynamic and not static. Servlet are used in the web application to serve JavaScript Object Notation objects containing results queried from the database. The following Servlets exists:

TaskTimeByUser when this resource is requested with a given user, it queries the database for all of the users task session, selecting total time and then responds the client with the result(the users) as an JavaScript Object Notation object with the result(the users) in an JavaScript Object Notation object

getActionsTimeAverage when this resource is requested, it queries the database for all actions selecting different measured times and then responds the client with the result as a JavaScript Object Notation-object

findAllUsers when this resource is requested, it queries the database for all users through the entity managers and then responds the client with the result (the users) as a JavaScript Object Notation-object

errorsByUser when this resource is requested with a given user, it queries the database for all of the users task session, selecting total errors and then responds the client with the result(the users) as an JavaScript Object Notation object

The data from the database is received as JavaScript Object Notation-objects and parsed to usable data to the bar charts from there.

Task controlling and user management

The web application also provides an interface for creating users, initiating and canceling an assembly task. Designing such web interface involves mapping user and task instances into elements. Furthermore, it involves communicating with other modules in the system to navigate the database and to start/cancel a task. To develop this web interface with Servlets and pages might be time consuming. Luckily, there exists a feature named JavaServer Faces (JSF) which is efficient in such cases. A JSF page is a dynamic page where the programming environment provides an easy way to manage the mapping between Java lists (or Java objects) and different components. Furthermore, JSF beans, also part of the JSF feature, can be used in JSF pages to communicate with other modules in an easy manner.

4.3.5 Detailed overview and internal integration

The following figure is shown to give a detailed overview of the inner workings and integration between the modules in this system:

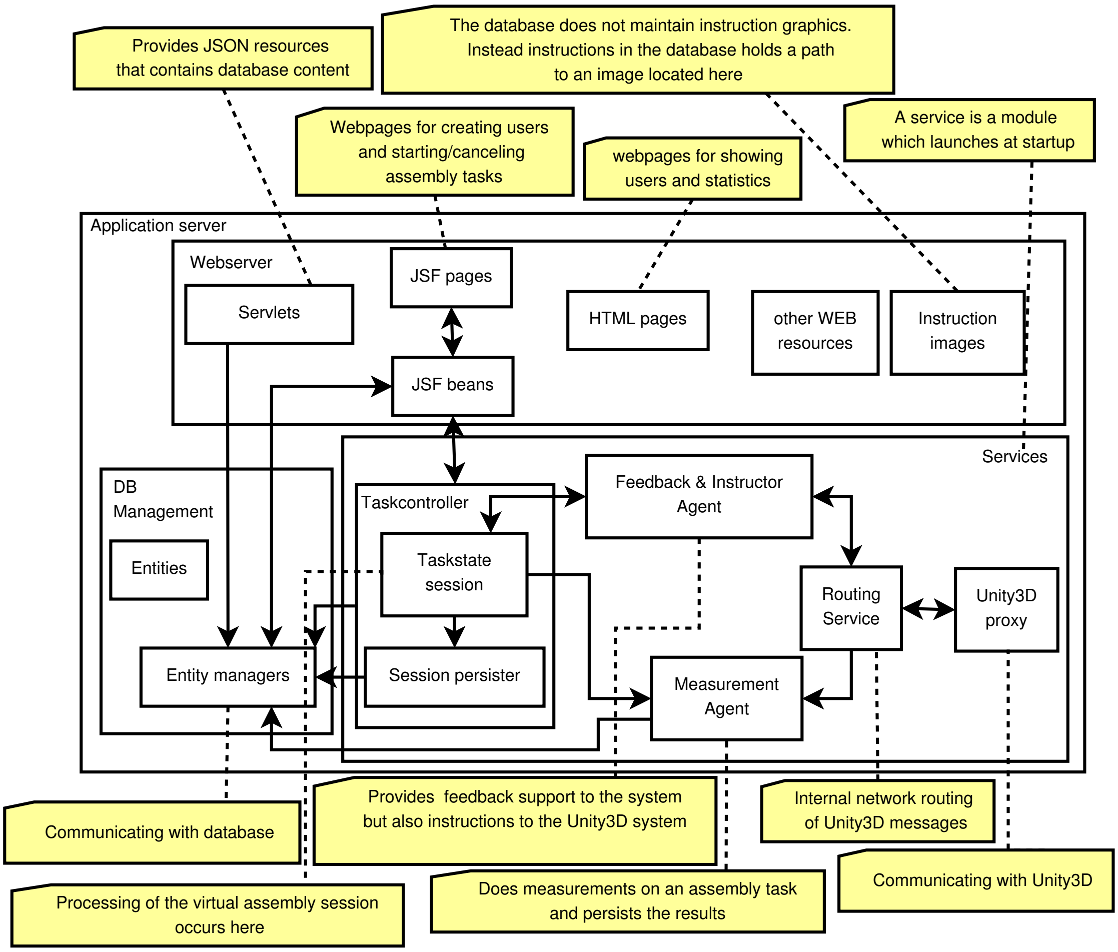


Figure 4.14: Architecture of the application server

Each subsystem or module except the web server provides all its functionality through methods and in this way an interface for communicating with it. The communication that occurs through these interfaces is basically what constitutes the integration. The web server do not actually need to provide an interface for its functionality as the other subsystems do not communicate with the web server in order to complete an operation. Moreover, routing messages (which is done locally) involves network communication of course but this logic is actually as mentioned encapsulated in methods and therefore abstract over the actual network communication.

5

Usability Testing

The main objective of the project was to explore the potential for the usage of Augmented Reality applications in industrial production. To reach this objective the use of usability research to evaluate the application was a key component.

5.1 Usability research

The purpose of usability research as a field is to evaluate different aspect of usability for an application, such as learnability, error frequency and user satisfaction, and to aid in the promotion of such qualities in the further development of the application.

The rigorous study of usability, and the development of methods and tools used in usability research has been largely developed during the last few decades, driven by the motivation to make increase the usability of web applications.

The standard ISO 9241-11 first introduced in 1998[11] defines usability as a concept, and is a reflection of the establishment of the formal study of usability. There is a broad array of different methods used in usability research with different advantages and disadvantages, usability testing is one of them.

5.1.1 Usability research methods

Usability.gov defines usability testing as follows. "Usability testing refers to evaluating a product or service by testing it with representative users"[12]. Typically, during a test, participants will try to complete typical tasks while observers watch, listen and take notes. The goal is to identify any usability problems, collect qualitative and quantitative data and determine the participant's satisfaction with the product. During usability tests the "think aloud protocol" is used, in which the test participants are encouraged to think aloud during the tests.

Another method that can be utilized in usability research is A/B-testing [13]. In A/B-testing, different versions of the software, with one or more variations between them, are provided to different users. Then KPI (key performance indicators) such as conversion rates are compared between the users.

Some advantages of A/B-testing is that it will measure actual behavior in real world conditions. And it's easy to implement new tests once the application has a sufficient users and the proper infrastructure is set up. And A/B testing can be used to resolve issues and questions that arise from other forms of usability research. For example.

For our purposes, usability tests were the most appropriate and effective method to use. There are methods that are suitable to use during the initial design phases, but since the basic tools and premises of our application were given from the start of the project, there were little value in using usability research methods focused on the early stages of the development process. These kinds of methods, such as participatory design [14] are used to inform early design choices, and since these were already given, the value of conducting such studies were diminished.

In a purely industrial project rather than an academic one, it might had made more sense to carry out early usability research to better inform early design choices to cater to customer benefit, before making the decision to implement a system based on any specific technology, such as augmented reality.

Other methods, such as A/B-testing, are more suitable for more mature products in a later development stage. Since it requires an established user base, and an infrastructure to effectively create and try out variations.

5.1.2 Usability Test Sample Size

Because of the aforementioned reasons, usability tests were chosen as a method to evaluate the application. One consideration to take into account is the number of test participants. How many test participants that is engaged in a usability test is a balancing between cost and time-efficiency and making sure to capture a satisfying proportion of the total number of usability problems that can be discovered. A test with too few test participants will likely not find all the usability problems, while the marginal value of additional test participants quickly diminishes for usability tests using too many test participants. In [15] Jakob Nielsen presents a statistical model for discovery of usability problems, and methods that can be used to determine optimal number of test participants. Nielsen derives the following formula for modeling number of unique problems found

$$\text{Number of unique problems found} = N(1 - (1 - p)^n) \tag{5.1}$$

Where p is the problem discovery rate, N is the total number of problems that can be found, and n is the number of subjects. In the article, Nielsen refers to an observation made by Hertzum and Jacobsen (2001) that small-sample estimates of p are almost always inflated. A method presented, combining a normalization procedure and the Good-Turing (GT) discounting procedure, produced very accurate adjustments to the initial estimate of p.

The GT adjustment is shown in (5.2), where p_{est} is the initial estimate of p, from the raw data, $E(N_1)$ is the number of usability problems detected by a single user, and N is the total number of unique usability problems discovered.

$$p_{GT-adj} = \frac{p_{est}}{1 + \frac{E(N_1)}{N}} \tag{5.2}$$

The adjustment from the normalization procedure is presented in (5.3), where p_est as before is the initial estimation of p, and n is the total number of users.

$$p_{Norm-adj} = \left(p_{est} - \frac{1}{n}\right)\left(1 - \frac{1}{n}\right) \tag{5.3}$$

These two adjustments combines into the adjusted value p_{adj} in equation (5.4).

$$p_{adj} = \frac{1}{2}\left(p_{GT-adj} + p_{Norm-adj}\right) \tag{5.4}$$

5.2 Test conductors and participants

During the usability tests, the following three roles were active:

- Test participant - performing the test scenarios and 'thinking aloud' throughout the process
- Supervisor - giving the test participants instructions, and guidance when needed to let participant complete the tests.
- Observer - Taking notes of the behaviour and comments by the test participants. And to document any other events of interest during the scenarios.

It was crucial for the tests for the supervisor to encourage the test participants to keep talking during the tests, without interfering too much in the problem solving process. In order to get as much valuable insights into the usability of the platform, while minimizing bias.

From the answers to the background questionnaire, it was evident that the test participants as a group was very homogeneous. All test participants were engineering students aged $20-24$. As a profile they had significantly more technological competency than the general population. This is the context that the tests were made in, and it should be taken into consideration when studying the results.

5.2.1 Performed tests

Two different scenarios were carried out by the test participants. Both consisted of the assembly of a LEGO model consisting of 27 pieces, 14 distinct.

In scenario A, the AR-platform were used to give instructions to carry out the assembly. In scenario B, plain print paper instructions were given to the test participant to carry out the assembly. The instructions used in scenario B are shown in Appendix B.

Counterbalancing were used during the tests. Which means that the ordering of scenario A and scenario B were interchanged between different tests. This was important to do since test participants memorized the assembly to some degree, and got faster completion times for the scenario that was performed last. Therefore it was important to mix the ordering in which the scenarios were performed.

The observer were instructed to specifically be attentive to different events during the tests. Some examples of there were:

- Incorrect component mounted onto the rig
- Repeated motions by the users to
- Unnatural or uncomfortable movements by the user.
- Correct component incorrectly mounted onto the rig
- Test participant unable to find the correct component
- Malfunction in hand tracking or orientation of AR-world.
- Failure to progress in a scenario without the aid of the supervisor
- Any critical hardware or software failure

5.3 Surveys and interviews

After all the different scenarios were carried out the test participants got to fill out a questionnaire. The first part of the questionnaire contained questions about task participant background, to establish the context in which the tests were performed, such as education levels and previous experience with motion capture technology. The second part of the questionnaire dealt with the users satisfaction and experiences in using the platform. Such as grading the perceived responsiveness of the hand tracking system.

Following the questionnaire, interviews between the supervisor and observer and the test participant were conducted. The interview consisted of open-ended questions allowing input from the test participant, not covered by the survey, to be collected.

It was important for the surveys to be conducted before the interviews. To allow the test participants to reflect on their experience before the discussion with the conductors of the test. This in order to decrease bias caused by influence by the interviewer. Affecting the quality both in the answers to the surveys and the interviews.

5.4 Analysis

After the usability tests were completed, all the collected data were considered in identifying usability problems, and their frequency and severity. The data that was analyzed included the notes taken during the tests by the observer, answers to survey questions, notes from interviews, data logs from the system and video recording of the test sessions.

5.5 Results

From the usability tests a multitude of forms of data were collected, including notes from the observer, system logs, interviews and video recordings. From these data collections a comparison between our system and the control plain paper solution could be made. More importantly, problems could be identified in the usability of the system.

In table 5.2 data from the test scenarios is presented. Total completion times are presented for all the test sessions. For the sessions using the AR-platform the total number of recorded errors are also displayed. The number of errors presented were logged by the system when the hand tracking system detected a hand entering the wrong box.

It is clear from the completion times for the sessions using the AR-platform that these were on average significantly higher than those using paper instructions. The only exception was user number 2, which partly can be explained by the ordering of the sessions for the user. The first session of user number 3 had to be discarded due to a technical error, caused by the glasses running out of batteries.

From the analysis of the observations made during the usability test sessions, the survey answers and interviews the usability problems in table 5.3 were identified. The assigned severity of the problems are based on an assessment of how great an impediment the problems were to the stated goals of the application. How much confusion were caused, how much time delay, how much frustration by the user and so on. The frequency rating refer both to how frequent the problem was between different test participants, and how frequent the problem was during individual tests.

For example, one problem presented in table 5.3 is false positives detected by the hand tracking module when a component was being mounted (problem number 4). The system was constructed to interpret the hands of the operator going into the assembly zone and out again as a component being mounted. The detection of that event was too easily triggered, which led to the false positives when the system prematurely read the assembly of a component to be completed. The operator then lost the assembly instruction, and was instead instructed to pick the next piece. The issue was a source of confusion and time delay.

Another problem was that the operator could not see the highlight, or virtual marker, that marked the box of the next component to pick when it was out of view (problem 5). The operator had to "scan" for the correct box by turning his or her head when the problem arose. Since the field of view for the AR-glasses was limited, this issue caused some time delay.

In table 5.1, which usability problems were exposed for which user is presented. Which gives the estimation $p_est = 0.4778$ for the problem discovery rate. Which can be used to evaluate the validity and coverage of the usability test. The coverage being the percentage of the total usability problems that was uncovered during the tests.

Table 5.1: The exposed usability problems for different users. The estimation of the problem discovery rate is given by $p_{est} \approx 0.48$.

	Problem											
Subject	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Count**	p
1	0	1	1	1	1	1	1	1	0	0	7	0.7
2	1	0	1	0	1	0	1	0	1	1	6	0.6
3	1	1	1	1	1	1	1	1	0	0	8	0.8
4	0	0	1	1	1	0	1	0	1	0	6	0.6
5	0	0	1	1	1	0	1	1	0	1	6	0.6
6	0	0	1	1	0	0	0	0	1	1	4	0.4
7	0	0	1	1	0	0	1	0	0	0	4	0.4
8	0	0	0	1	0	0	0	0	0	0	2	0.2
9	0	0	1	1	0	0	1	0	0	0	3	0.3
Count	2	2	8	8	5	2	7	3	3	3		
P	0.22	0.22	0.89	0.89	0.55	0.22	0.78	0.33	0.33	0.33		0.4778

Table 5.2: The test results from the usability test sessions.

User	Test Type	Test nr	Errors	Time (min)
User 1	The system	1	36	09:50
	The system	2	32	06:30
	Paper instruction	3	-	06:30
User 2	Paper instruction	1	-	06:21
	The system	2	28	06:02
	The system	3	19	05:27
User 3	The system	1	N/A	N/A
	The system	2	62	06:01
	Paper instruction	3	-	03:41
User 4	Paper instruction	1	-	05:27
	The system	2	28	07:18
User 5	The system	1	58	19:19
	The system	2	39	11:06
	Paper instruction	3	-	05:42
User 6	Paper instruction	1	-	05:45
	The system	2	40	08:57
User 7	The system	1	38	12:35
	Paper instruction	2	-	04:14
User 8	Paper instruction	1	-	03:10
	The system	2	42	04:11
User 9	Paper instruction	1	-	03:44
	The system	2	25	06:50

Table 5.3: Usability problems

Problem number	Description	frequency	severity
1	Lack of detection by the hand tracking when taking a component from the correct box	low	medium
2	Lack of detection by the hand tracking system when taking a component into the assembly zone	very low	medium
3	Pictures too small to accurately distinguish details	high	medium
4	False positives by the hand tracking system in detecting a component being mounted	high	high
5	User being unable to find box highlight when being out of view	high	low to medium
6	Inability to revert a progression in a task / inability to step back the instructions	high	high
7	Loss of orientation of the AR-world due to Vuforia marker going out of view	high	medium to high
8	User unable to distinguish current component from instruction image	high	medium
9	Ambiguousness from instruction image due to poor perspective/orientation	high	medium
10	Users not using the textual information in the instructions	high	low

6

Discussion

The results from the usability tests, gives us a foundation to reason about the issues of our system, and the methodology used to evaluate it. And beyond, to reason about the problems and potential of the use of AR-technology in general.

6.1 Usability Tests

An assessment of the validity of the results given by the usability tests can be made using the methodology presented in section 5.1. The question is if the usability tests uncovered a satisfying proportion of the total number of usability problems, or left too many undiscovered.

From the raw data of discovered usability problems in table 5.1 the estimate $p_{est} = 0.4778$ can be read. Using (5.2), (5.3) and (5.4), given the expected number of problems discovered by the first user $E(N_1) = p_{est} \cdot N$ and the total number of discovered problems $N = 10$, the result (6.3) is acquired for the adjusted problem discovery rate.

$$p_{GT-adj} = \frac{0.4778}{1 + \frac{0.4778 \cdot 10)}{10}} \approx 0.32 \tag{6.1}$$

$$p_{Norm-adj} = \left(0.4778 - \frac{1}{9}\right)\left(1 - \frac{1}{9}\right) \approx 0.33 \tag{6.2}$$

$$p_{adj} = \frac{1}{2}\left(0.3233 + 0.3259\right) \approx 0.33 \tag{6.3}$$

The expected value of the percentage of the total number of usability that was discovered can now be calculated by (6.4). Which indicates that the tests had good coverage and shows us that probably the majority of the usability problems in the application were discovered during the tests.

$$1 - (1-p)^n = 1 - (1 - 0.3246)^9 = 97\% \tag{6.4}$$

Of course, these calculations are completely dependent on the subjective analysis presented in table 5.3 of what constituted a usability problem. In [15], Nielsen refers to a study where four different labs independently evaluated a calendar system and prepared reports of the usability problems they discovered. The number of unique problems identified by each lab ranged from four to 98, and only one usability problem was reported by all four labs. So it is clear that without clear context and goals for an application, the quantitative analysis of the coverage of a usability test has very little objective value.

Using the same calculations, we get the result that even with four users 79% of the usability problems would be discovered. It is clear that it would have been more cost-efficient to carry out multiple develop-test-cycles with small sample sizes of 4 or 5 test participants.

The number of errors presented in table 5.2 proved to be a poor metric in assessing the effectiveness of the system. There were a lot of false positives in detecting the user taking the wrong component. When adjacent boxes were incorrectly triggered by the user actually picking the correct component.

6.1.1 Recommendations

In table 6.1, different alternative solutions are presented to the usability problems identified during the usability tests, which are presented in table 5.3. Some of the problems have several different proposed solutions, ranging from less to more extensive in implementation. Some of the proposed solutions were planned, but were never fully implemented.

Table 6.1: Proposed solutions for identified problems from table 5.3. Some problems have several different alternative solutions.

Problem number	Problem description	Proposed solutions	Comment
1	no detection pick from box	-	need further investigation
2	no detection assembly zone	-	need further investigation
3	images too small	Alt. 1: Enlarge the images Alt. 2: Replace instruction image with 3D-projection of model onto physical token with fiduciary marker.	-
4	false positives assembly zone	Alt. 1: Adjust sizes of the triggering boxes in Unity. Use one trigger box for detecting entrance into zone, and one trigger box for exit from the zone, to make it less sensitive and less prone to false positives. Alt. 2: Combine instructions to pick and assemble components, and remove detection of assembly. Alt 3: Develop some other form of verification of assembly, rather than hand entering and leaving assembly zone. For example use computer vision and camera output from glasses to confirm correct assembly.	-
5	highlight out of view	When the highlighted box is out of view, indicate where to look for it by an arrow in the periphery of the GUI of the glasses.	planned feature

Continued on next page

Problem number	Problem description	Proposed solutions	Comment
6	no undo command	Use one of the function keys on the controller. For example 'Function' or 'BACK'.	-
7	loss of AR-orientation	Alt. 1: Complement orientation with gyroscope and accelerometer, when fiduciary marker is out of view. Alt. 2: Use complementary fiduciary markers. Alt. 3: Use both alt. 1 and alt. 2.	planned feature
8	current component indistinguishable	Highlight the new component in the instruction image. For example by using an exploded-view-drawing-style image.	-
9	bad perspective	(See solution alternative 2 to problem 3.)	-
10	text not used	Alt.1: Don't make adjustment Alt. 2: Remove the text Alt. 3: Make the text more prominent.	-

6.2 The Future Potential of AR

As AR-technology keep maturing, the use of AR-assistance in manufacturing and assembly will probably be a more commonly used practice since it shows many signs of being a technology that can reduce mental workload of operators and reduce errors in assembly [16]. But before AR-assisted workers will be a common sight on the factory floor the technology need to overcome some of the current limitations.

6.2.1 Limitations

Like with any relatively new technology, augmented reality has numerous limitations which keep it from reaching its full potential. What follows is a summary of the most important ones in respect to use of AR in manufacturing.

Hardware

The usability of an AR-platform is highly dependent on well functioning hardware, designed for AR-purposes. This is a point that became very obvious while conducting tests with the platform used in this project. While the Epson glasses are well suited for watching media content or browsing Web pages, its lack of wide angle screens and poor camera quality put tight restraints on what could be accomplished when the device is used as a part of an AR platform. Another part is the mobility issue, while some assembly tasks may be suitable for a stationary setup, a much larger scope of situations that would benefit from AR-assistance could be reached when battery and sensory technology allowed for an AR-platform to be completely wearable.

Interface

Beyond having a suitable hardware setup, an AR-platform needs to have an intuitive and easy to use user interface. The method of using hand tracking to interact with a AR-system have been under development long time and is getting better, but often when using AR in a manufacturing process the operator needs to use one or both of the hands for completing the assembly assignment, so an alternative mode of interaction that does not require hand movements would highly strengthen the usability of the platform.

7

Conclusion

Enabled by the process of design, development and consequent evaluation of the platform we can now address to the original problem statement and goals of the project.

The purpose of the application was mainly threefold:

- Present instructions to the operator.
- Track the progress of the assembly process, and to use that information to feed the operator the correct information.
- Collect data over user behaviour to gain insight.

Our contribution to the application, building on the previous project, consisted mostly of two parts: To adapt the system to present instructions through smart glasses rather than a monitor, and to make use of augmented reality to present instructions more effectively.

From the results of the usability tests and consequent analysis we can conclude that we have not succeeded in constructing a solution that reduces assembly time and errors, compared to our conventional control solution.

We have however been able to identify strengths and weaknesses of our solution. These insights can be used in further development of industrial applications of AR-technology.

The identified problems, which are described in table 5.3, can be summarized in three points:

- The system was not effective in tracking and/or assuring correct assembly
- The orientation of the AR-world was not robust when taking into account user behaviour.
- Instruction images were of poor quality.

Appendices

A. Diagrams

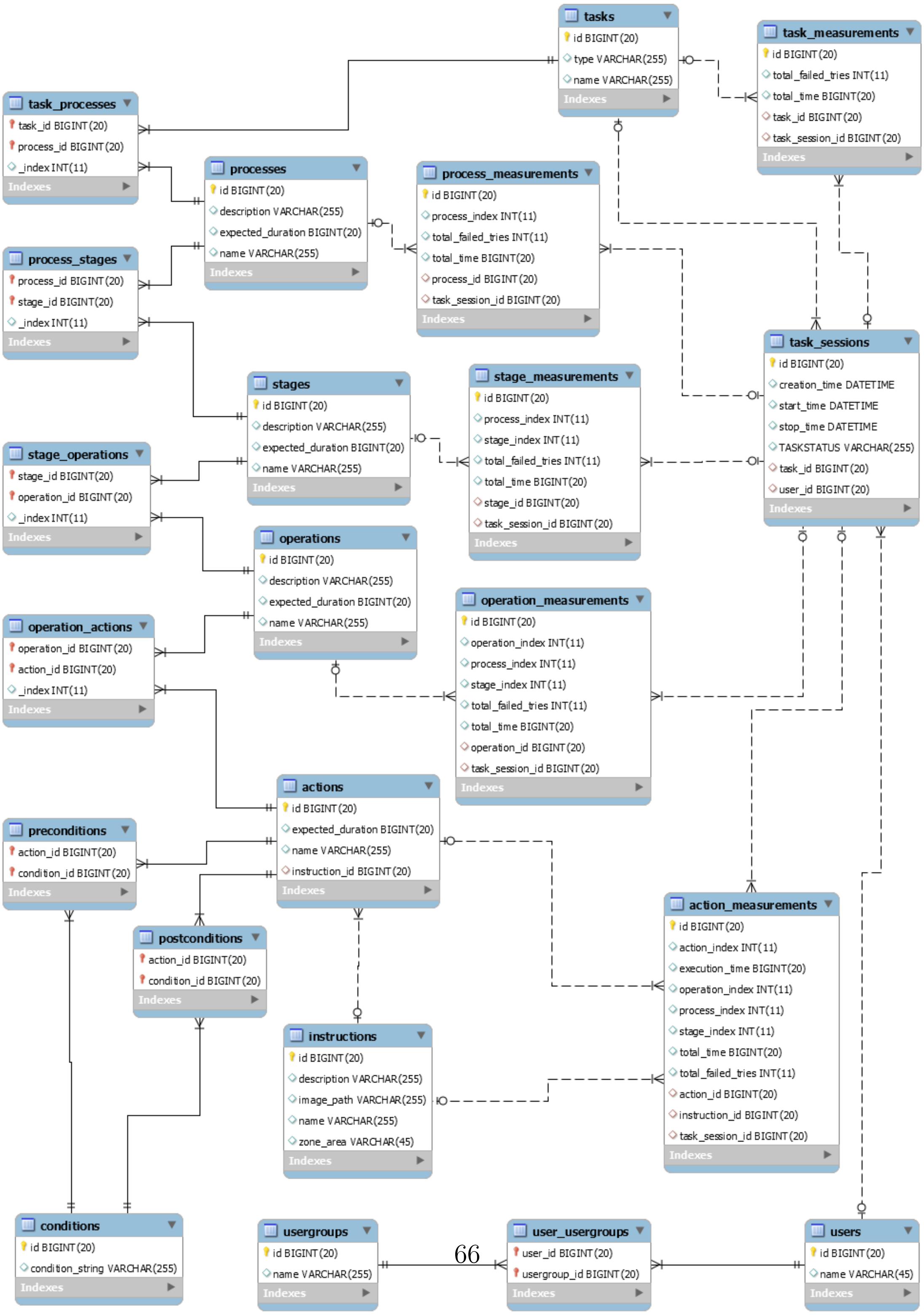
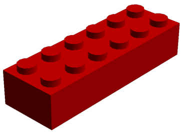
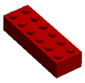
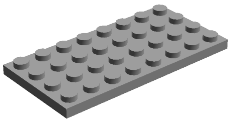
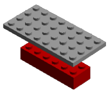
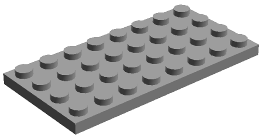
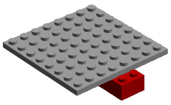
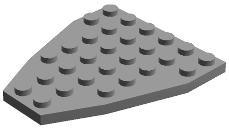
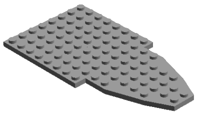
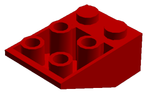
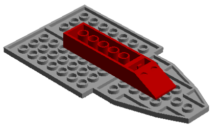
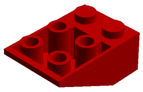
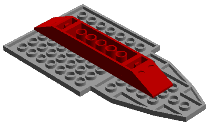
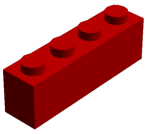
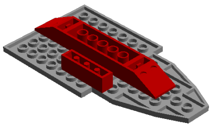
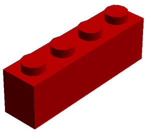
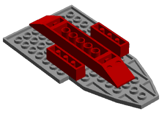
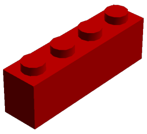
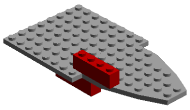


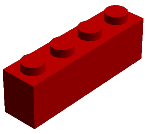
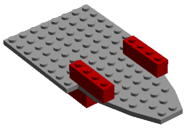
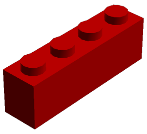
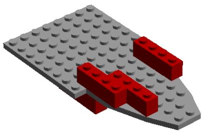
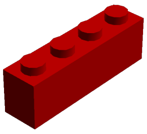
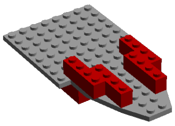
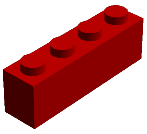
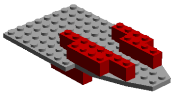
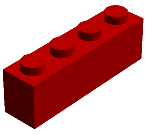
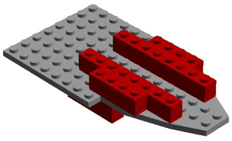
Figure 1: Complete EER diagram over database scheme

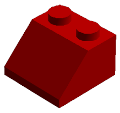
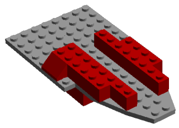
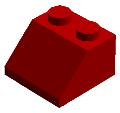
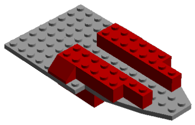
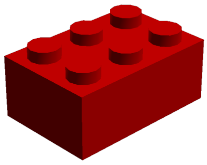
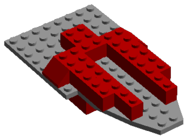
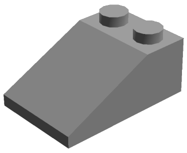
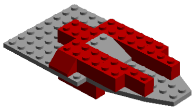
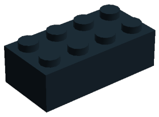
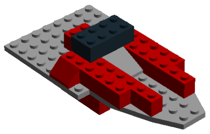
Plain paper instructions for test scenario B

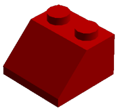
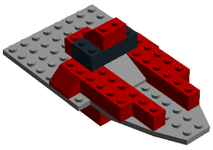
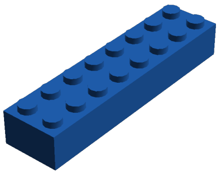
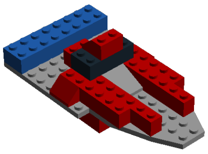
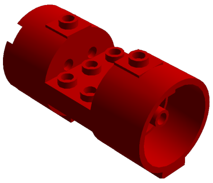
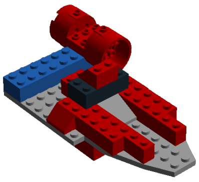
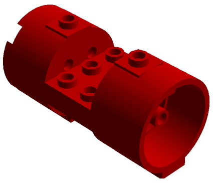
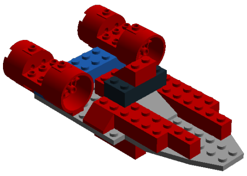
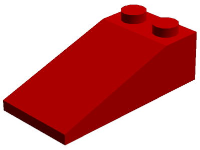
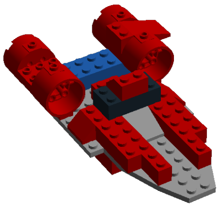
Assembly instructions hovercraft

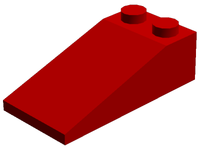
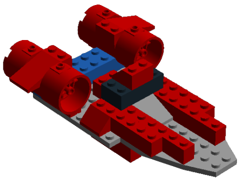
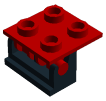
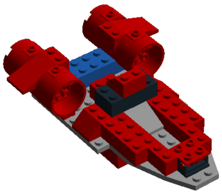
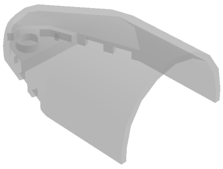
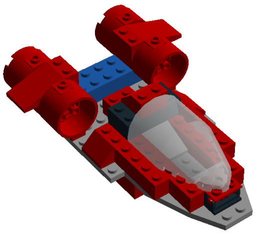
Next piece to pick	Next assembly step
	 (1/27)
brick1x4-red	
	 (2/27)
flatPlate4x8-lightGrey	
	 (3/27)
flatPlate4x8-lightGrey	
	 (4/27)
stemPlate-lightGrey	

 **console2x3-red**	 **(5/27)**
 **console2x3-red**	 **(6/27)**
 **brick1x4-red**	 **(7/27)**
 **brick1x4-red**	 **(8/27)**
 **brick1x4-red**	 **(9/27)**

 brick1x4-red	 (10/27)
 brick1x4-red	 (11/27)
 brick1x4-red	 (12/27)
 brick1x4-red	 (13/27)
 brick1x4-red	 (14/27)

 roofTile2x2-red	 (15/27)
 roofTile2x2-red	 (16/27)
 brick2x3-red	 (17/27)
 roofTile2x3-lightGrey	 (18/27)
 brick2x4-black	

	(19/27)
	 (20/27)
roofTile2x2-red	
	 (21/27)
brick2x8-blue	
	 (22/27)
engine	
	 (23/27)
engine	
	
roofTile2x4-red	

	(24/27)
	
roofTile2x4-red	(25/27)
	
hinge	(26/27)
	
cockpit	(27/27)

Bibliography

[1] J. Provost, A. H. Ebrahimi, K. Åkesson, Online support for shop-floor operators using body movements tracking, in: 12th IFAC/IFIP/IFORS/IEA Symposium on Analysis, Design, and Evaluation of Human-Machine Systems, 2013, pp. 102–109.

[2] A. Stork, N. Sevilmis, D. Weber, D. Gorecky, C. Stahl, M. Loskyll, F. Michel, Enabling virtual assembly training in and beyond the automotive industry, in: Virtual Systems and Multimedia (VSMM), 2012 18th International Conference on, 2012, pp. 347–352.

[3] M. Billinghurst, Augmented reality in education, New Horizons for Learning 12.

[4] S. Cheng, A. Doshi, M. Trivedi, Active heads-up display based speed compliance aid for driver assistance: A novel interface and comparative experimental studies, in: Intelligent Vehicles Symposium, 2007 IEEE, 2007, pp. 594–599.

[5] N. Pathomaree, S. Charoenseang, Augmented reality for skill transfer in assembly task, in: Robot and Human Interactive Communication, 2005. ROMAN 2005. IEEE International Workshop on, 2005, pp. 500–504.

[6] N. Vignais, M. Miezal, G. Bleser, K. Mura, D. Gorecky, F. Marin, Innovative system for real-time ergonomic feedback in industrial manufacturing, Applied Ergonomics 44 (4) (2013) 566 – 574.
URL http://www.sciencedirect.com/science/article/pii/S0003687012001858

[7] S. Makris, G. Pintzos, L. Rentzos, G. Chryssolouris, Assembly support using {AR} technology based on automatic sequence generation, {CIRP} Annals - Manufacturing Technology 62 (1) (2013) 9 – 12.

URL http://www.sciencedirect.com/science/article/pii/S0007850613000966

[8] P. Carlson, A. Peters, S. Gilbert, J. Vance, A. Luse, Virtual training: Learning transfer of assembly tasks, Visualization and Computer Graphics, IEEE Transactions on 21 (6) (2015) 770–782.

[9] http://www.epson.se/se/sv/viewcon/corporatesite/products/mainunits/overview/12411, Epson moverio bt-200 = ”http://www.epson.se/se/sv/viewcon/corporatesite/products/mainunits/overview/12411”, [Online; accessed 12-May-2015] (2015).

[10] https://www.asus.com/Multimedia/Xtion_PRO/, Asus xtion pro = ”https://www.asus.com/Multimedia/Xtion_PRO/”, [Online; accessed 15-May-2015] (2015).

[11] Ergonomic requirements for office work with visual display terminals (VDTs) – Part 11: Guidance on usability (1998).

[12] Usability testing, http://www.usability.gov/how-to-and-tools/methods/usability-testing.html, accessed: 2015-05-19.

[13] Putting a/b testing in its place, http://www.nngroup.com/articles/putting-ab-testing-in-its-place/, accessed: 2015-05-19.

[14] Participatory design, http://www.usabilityfirst.com/usability-methods/participatory-design/, accessed: 2015-05-19.

[15] J. R. L. Carl W. Turner, J. Nielsen, Determining usability test samplesize, International Encyclopedia of Ergonomics and Human Factors 3 (2) (2006) 3084–3088.

[16] L. Hou, X. Y. Wang, M. Truijens, Using augmented reality to facilitate piping assembly: An experiment-based evaluation, Journal of Computing in Civil Engineering 29 (1), hou, Lei Wang, Xiangyu Truijens, Martijn.