Human Machine Interface Visualization Enhancement of an ABB Quality Control System

Nils Johansson and Filip Williamsson

October 25, 2009
Master's Thesis in Computing Science, 2*30 ECTS credits
Supervisor at CS-UmU: David Sjölie
Examiner: Per Lindström

Umeå University
Department of Computing Science
SE-901 87 UMEÅ
SWEDEN

Abstract

Today's increasingly powerful computers grant the ability of visualizing information in stunning graphics. However, this has so far not been taken advantage of much in industrial applications. This thesis has explored the market of 3D visualization software, and evaluated six commonly used technologies to see which one is best suited for developing a human-machine interface (HMI) for a Quality Control System produced by ABB. The thesis has created a number of design concepts to improve the usability of the HMI. These were based on usability issues found in a user study and a heuristic study of the current HMI. They were then implemented in a technology demonstrator using the recommended 3D visualization software.

Contents

List of Figures

Chapter 1

Introduction

Visualizing and presenting information to users is a significant part of product development. Today's graphics in ABB's software products are mainly in 2D. Standard GUI technologies such as Microsoft WPF are often used in the development of product's user interfaces.

The goal of this master thesis is to explore the market for technologies used in 3D graphics development, evaluate these technologies and recommend a suitable tool. The final part was to develop a demonstrator, using the selected technology, to demonstrate a vision of how one of ABB's products could present information in the future.

The tool in focus for the project was a Quality Control System from ABB. The product uses a specific human-machine interface (HMI), currently implemented using 2D graphics, to display information about the process. A part of the project was to propose concepts on how 3D graphics could solve some of the issues with this interface.

The 3D development process consists of three major steps; building graphics using modeling tools, programming the application to use the graphics, and finally compiling and deploying the application. This process in generalized but is applicable for most 3D development processes. The evaluation of 3D tools focuses mainly on the second step: programming the application, and thus the software evaluated was different types of 3D engines.

The evaluation of the engines was performed by first gathering general information on commonly used software for 3D development. The next step in the process was to narrow down the selection to the six technologies best fitting to ABB's purposes according to nine criteria. These six technologies were then evaluated by one person for two weeks each and compliance to the criteria was tested. After the period ended the recommended technology was Unity 3D. It was chosen mainly for its high productivity and powerful library of functionalities.

Before and during the project a user study was conducted on users of the Quality Control System. The study sought to find out what the requirements were on a future HMI, and what kind of issues the users of the current HMI experienced. The results from this study in combination with a heuristic evaluation of today's HMI and a brainstorming session resulted in a number of design concepts for a future interface. These design concepts aims to resolve the issues found in the study.

The final part of the project was to implement a concept demonstrator in the recommended technology (Unity 3D). The concept demonstrator is an application built entirely using 3D graphics and it implements some of the suggested design concepts.

The purpose of the application is to propose how 3D graphics can be used in the future for visualizing data in an industrial process.

1.1 Purpose

This study was conducted upon request from ABB; the purpose was to evaluate how the current HMI of one their Quality Control Systems could be improved using 3D graphics.

Tons of different engines can be found for creating 3D applications and some are more or less useful, the challenge was to identify one that was powerful and productive enough to meet the requirements of ABB's production standards.

As one part of this project was to propose concepts on how to solve some issues with the current user interface, the study provides a concept demonstrator. This demonstrator is an example of a future HMI in 3D, built in the recommended technology. All concepts are based on issues found in a user study that was realized within the scope of this project. Requests from ABB were also weighted into the design proposals.

1.2 Process

This is the initial time plan for the thesis work, it is shown in figure 1.1.

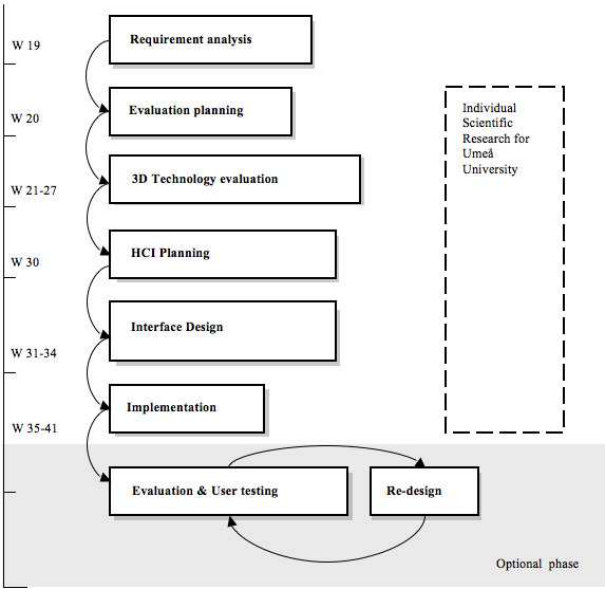

Figure 1.1: Overview of project activities

In the first phase, background information was gathered. Information was details on the Quality Control System and its HMI and the 3D application development process.

In the next phase, the evaluation planning, a large list of 3D development technologies was created and nine different criteria for evaluating them were decided.

During the 3D technology evaluation phase six technologies were evaluated according to the criteria in a deeper study. Once this phase was done, one technology was chosen for recommendation.

The HCI planning phase consisted of analyzing the current HMI, the user study and analyzing the method of the user study.

The interface design phase consisted of analyzing the results from the HCI planning phase, and creating design concepts for the upcoming technology demonstrator.

The final phase of the project consisted of creating a technology demonstrator in the recommended technology.

1.3 Outline

This is a description of the contents in each chapter of this report.

Quality Control System This chapter describes one part of the background research. It contains information about the specific ABB Quality Control System; its purpose, how it is used and its current graphical user interface.

The 3D development process This chapter describes the necessary steps in the process of creating a 3D application.

Data visualization in 3D environments This chapter holds information of information visualization systems and how usability should be considered when creating a technology demonstrator for this project.

Augmented reality A general overview of the concept of Augmented Reality and an in-depth survey on how it could be applied in a future HMI of the Quality Control System.

Method This chapter covers the methods used in the various phases of the project. This includes the user study, brainstorming session, heuristic evaluation and the development of the technology demonstrator.

Results This is the results of the project including results from the 3D technology evaluation, the heuristic evaluation, the user study and a number of design concepts.

Conclusion This chapter is a conclusion of the results. It answers the research questions from the introduction and presents future work and limitations.

Chapter 2

Quality Control System

CENSORED BY ABB CORPORATE RESEARCH.

Chapter 3

The 3D development process

A part of the background research consisted in finding out how to develop 3D applications. This chapter describes important steps and the general procedure of creating such software.

Developing a 3D application consists of three major steps; creating 3D graphics, programming, and deploying into a runnable application (figure 3.4).

The graphics are most commonly created in 3D modeling software such as Autodesk Maya, Blender, Autodesk 3Ds Max, NewTek LightWave or Autodesk SoftImage. In these applications models are created, materials and textures are applied, lighting are added. Other uses can be to create animations and also characters with bone structure.

The 3D modeling tool can save the graphics into 3D model files. These files can then be imported by 3D engines to make use of the stored graphics. Some of the file formats are FBX, 3ds and COLLADA. However, the engines differ in that they might not support all of the features that these different formats provide.

The next step in the process is to use a so-called 3D engine where applications can be programmed to use the graphics dynamically. A 3D engine is a library of functionalities for handling the graphics in real-time. It often provides functionality for representing a hierarchy among graphic objects (so called scene-graphs), controlling position, size, orientation, etc. This is the step where programming logic are added into the application.

The 3D engine uses the hardware for output of the graphics; this is done using low-level graphics APIs. The most commonly used low-level graphics API are Direct3D (for Microsoft systems only) and OpenGL (an open source API available for most operating systems). The aim of these APIs is to abstract the communication between a graphics application and the graphics hardware drivers.

Finally, when the application is completed it is compiled into an executable file.
This process may be simplified into three steps:

1. Create graphics (figure 3.1)
2. Programming (figure 3.2)
3. Deploy and run (figure 3.3)

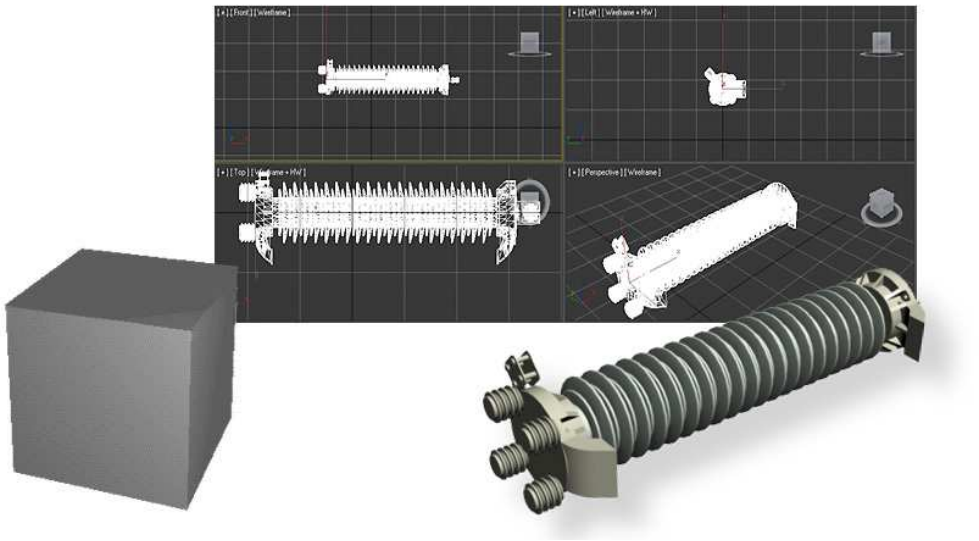

Figure 3.1: First create static graphics in 3D modeling software

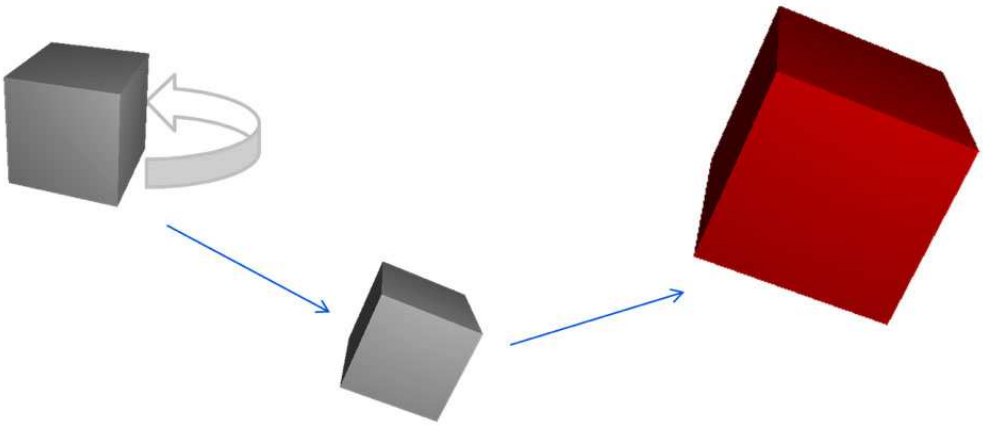

Figure 3.2: Program the graphics in a 3D engine to make the environment alive

Figure 3.3: Compile, distribute and run

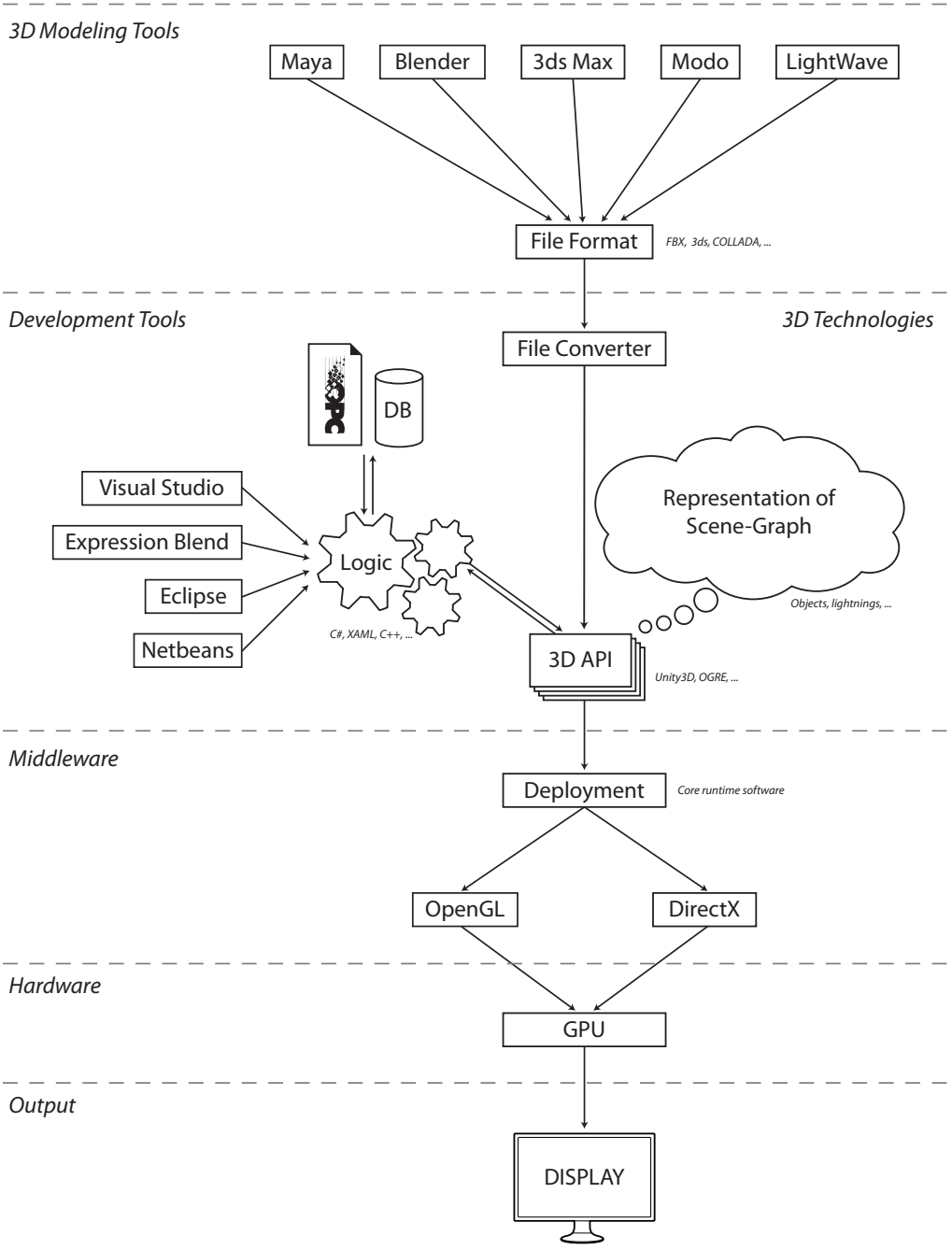

Figure 3.4: Advanced overview of the 3D development process

Chapter 4

Data visualization in 3D environments

When building an artifact that in some way interacts with a human, the usability of the interaction is important in order to avoid usage problems with annoyance, time-consuming tasks and in some cases safety issues [18]. This does of course also apply to computer software with graphical user interfaces. In this thesis we are interested in building a user interface that is based on 3D graphics. The purpose of this section is to find out how the system could be designed in order to provide good usability. Working with 3D graphics gives the designer a different kind of toolset for providing functionality adapted to users' needs than with compared to 2D graphics, so. So what can this toolset bring to the table in terms of usability? This short review attempts to find interesting examples of usable functionality that can be achieved in 3D environments, and also examples and recommendations on how 3D graphics can provide better usability in general in a graphical user interface.

A goal of this thesis is to design a conceptual system that visualizes data, so the first step was to find fitting guidelines on what is important when designing such a system. A commonly adopted set of guidelines is Schneiderman's taxonomy for data visualization [29]. He describes seven important tasks that a user will perform when interacting with a system for data visualization.

It is desirable to provide good usability in the interface and therefore an article study was made to find fitting functionality or guidelines that could be used in the system in order to fulfill the requirements of these tasks.

4.1 Visualizing data

An article by Schneiderman, 1996 discusses how to visually represent data of different types and what methods of representation that is commonly used and are suitable for each type [29]. He claims that there are seven different tasks that are important when working with data; Overview, zooming, filtering, details-on-demand, relations, history and extraction. Each of these tasks is important for the user's performance and for the understanding of a system, and therefore the system's support for performing these tasks should be well designed.
A short description of each of these tasks follows.

Overview is the task of gaining an overview of a collection of items. Along with an overview there is need for navigation tools to navigate through the data.
A *Zoom* function is useful for getting a detailed view of items of interest. Smooth zooming helps users preserve their sense of position and context.
A *filter* for filtering out uninteresting items. Allowing the user to eliminate unwanted items will reduce the risk of information overload.
Details-on-demand. Details should be available when the user needs them.
Relations. The relations between different items displayed should be apparent.
History. A history of actions should be kept, in order to support undo, redo and progressive refinement.
Extraction. It should be possible to extract data for uses outside of the system, such as saving to a file for emailing, printing etc.

In order to make sure that software for data visualization provides good usability, all these tasks should be supported with carefully designed functionality. As this thesis aims to make a design proposal of such a system, each of these tasks will be considered in the design. The rest of this study attempts to find examples of functionality for 3D systems that can be used to fulfill each of these tasks.

4.2 Overview and Zoom

Overview and zoom are both concerned with the navigation in an information visualization system. A number of suggestions are described below that show how the navigation could be handled in terms of both overview and zoom, which is why they both have been combined into a single chapter.

One problem when navigating through a 3D interface is that the wrong combination of navigation operations can make the user lose orientation. For example, when an object is zoomed in, the pivot point for rotation can be outside of the current view. If the user then uses an orbiting tool, the camera moves around the pivot point and thus causes the object to move out of the view. These kind of problems are common for novices, but are sometimes also encountered by more experienced users [23].

Some approaches to solve these issues are presented in an article by (Fitzmaurice, 2008). A number of tools for navigation in CAD software have been developed, and some of the classical tools have been redesigned in order to provide "safe" 3D navigation that keeps the user away from losing orientation. The tools were developed over a time of two years, and have been thoroughly tested. The end result of the study show that the use of these tools can help new users in learning to use 3D software, as well as improving the usage experience for expert users.

There are three different toolsets for 3D navigation described in the article. A short description of the functionality of each tool is provided below

The *View Object tool* has four functions: *Center, Zoom, Orbit* and *Rewind*. The Center function sets the pivot point of the model with a single mouse click; this pivot point is used by orbit and zoom. There is no point zoom, i.e., zooming in towards the location of the mouse pointer. Such a function could cause the user to zoom in beside the object and thus making it disappear from view, instead zooming always focuses on the pivot point. The orbit tool lets the user circle around the object using the mouse, this function occupies a larger space in the control bar than the other tools, in order to show that it is more important and more frequently used than the others.

The *Rewind tool* is available in all toolsets, and lets the user see a history of camera actions. The history is displayed through thumbnails of old camera states. The user

can reset the camera to a particular state by clicking it. This feature was appreciated by both beginners and experienced users in the tests.

The *Tour building tool* also has four functions; *Forward, Look, Up/Down* and *Rewind*. The forward tool is used to move the camera closer to an object. One mouse click moves the camera 50% closer to the point chosen on the object's surface with a 0.5 sec animation. If the mouse button is held down, a slider appears that can be used to move towards and from the object. The closest the camera can be moved towards the object is 95% of the distance between the starting point and the object; this is to prevent the user from seeing through the graphics. The Look tool lets the users turn the camera to look around, like they would turn their own head. The Up/Down tool acts like an elevator and provides a graphical slider that allows the user to adjust their height in the scene. The forward tool is the largest of the tools in this toolset, again emphasizing important and frequently used tools.

The *Full toolset* has all the previously described tools available with a few changes. Instead of Forward it has a Walk function that lets the user move forward at the same time as changing the viewing direction. The zooming function in this toolset is a point zoom, which means that the zoom function centers on the position of the mouse pointer in the view when zooming in.

In the article, some users (some of which were experienced Photoshop users) had trouble with orienting themselves in a 3D environment. When the view changed without any animated camera rotation or movement they lost their orientation. This problem could probably be avoided by illustrating the camera movement with an animation when views change in the 3D application, and thus make the users understand how the different views are related. This is shown to reduce the cognitive load of understanding the transitions between pre-zoomed and post-zoomed state in [14].

The article makes a comparison between graphical icons and text. When there are many graphical icons with no text, a new user of the system can get confused and find it difficult to understand the purpose of each icon. To resolve this problem, text was used instead of icons to describe functionality in the system. This removes the need for inexperienced users to learn the meaning of several new graphical icons.

Error prevention is a part of usability in an artifact, by preventing errors time, effort and annoyance can be saved. Some examples of error prevention are that the user was not allowed to move forward through an object and not moving too far from it. Another example was cursor wrapping; when the cursor were at the far end of the screen, the position was reset at the other end. This lets the user for example use the Orbit tool without the screen limiting the amount of movement that can be performed.

4.3 Filtering

Filtering is an important issue in information visualization systems; it lets the user filter out irrelevant data and focus on important information. If there is no filtering function available and the amount of data is large, the user can experience information overload [25]. Another important reason for having filtering is the occlusion problem. This is a phenomenon that occurs when objects in the foreground cover (occlude) objects behind them, further back in the environment.

There are many attempts to solve this problem, and one of them is called the BalonProbe [15]. The principle of this technique is that the user selects a point in 3D space, preferably in an area with a high population of objects; this point is the center of

the balloon and the objects are then "exploded" out to the surface of the balloon, thus giving the user a better overview of what objects that are available.

Other common approaches are transparency and layers. An attempt with semi-transparency was made by Marcus (2003) in order to provide filtering. The intention was to make irrelevant objects in the foreground see-through to shift the focus to more important objects in the background. The article claims to have overcome many of the shortcomings of traditional 3D data visualization systems such as occlusion.

In the recommended 3D development software it is possible to add layers to a scene in a 3D application [10]. Layers are commonly used in 3D engines, and are often used by cameras to only render parts of a scene, and by lights to selectively illuminate objects in the scene. It is also used in cases where ray casting is used for collision detection; the objects in certain layers of the scene are not taken into account when detecting collisions. Layers contain graphical objects and can be turned on and off in cameras etc. during runtime using code.

4.4 Details-on-demand

When the user has found information using the overview, zoom and filtering functions, he/she will want to see the details of the selected items. Detailed information should not be provided at all times, but the user should always have the ability to access information when it is needed in order to reduce the risk for information overload [25].

The display of detailed information about current selections could be provided in a window on the side. This was used to provide information in Marcus, 2003 [23]. Some of the users in the study (chapter 2) suggested that help could be provided about some parts of the interface by using a combination of pressing keys on the keyboard and clicking a component of the interface with the computer mouse. The users did not want to read the manual of the HMI, but they believed that if the information was available directly in the interface they would be more likely to use it.

4.5 Relations

Relations between information and objects should be displayed in such a way that a user can see the connection. In most articles reviewed here, the primary ways of dealing with inter-object relations were to use colors, size, focus and grouping, when displaying data, such as in Marcus (2003) [23]. E.g., this could be used to map out which objects in the environment are related, by organizing them in groups where they are in close proximity, or displaying them in the same color when the user needs to know which ones that are related.

This is used in Chang, D 2002 [13] to improve learning in an interface. Some of the laws of gestalt theory are the law of balance, continuation, closure, proximity and similarity. These laws could be used to organize and design the interface in order to provide information about relation among objects. Related objects should be kept in close proximity.

4.6 History

Functionality that covers history-related tasks is important in order for users to have control over their work in the software. History related tasks are for example undoing

and redo functionality that allows a user to revert from a faulty decision. The "rewind" function described in the overview and zoom chapter, as part of the overview function, is one example. This functionality allows the user to see a history of camera states and revert to those states by clicking a thumbnail picture.

Besides navigation in 3D space, there are few tasks that could implement undo and redo functionality. Almost all changes and inputs made in the HMI cause actions in the physical system and can therefore not be undone.

4.7 Extraction

At this stage the design is focused on representation and user interaction. Extraction features will be added in the future. Examples of extraction for this application could be CENSORED BY ABB CORPORATE RESEARCH.

4.8 Discussion

When looking at the amount of research found on each of Schneiderman's [29] tasks related to usability in 3D, it can be noted that the most research has been performed on navigation in 3D environments and how to keep the user from losing his/her orientation. However, this is not very strange, as overview is the least abstract and context-dependent task of those listed in the taxonomy [29]. For example, it is hard to find general concepts on how to provide details-on-demand and filtering that can be applied to any system, as they have to be very system specific. It can be differences in what type of data is displayed, how it is measured; bars, graphs etc. and therefore these tasks must be designed individually for each system and then tested for usability and evaluated instead of using concepts from other systems.

4.9 Summary

The literature that was reviewed in this study suggests that an interface built with 3D graphics can benefit from a number of concepts. The tasks for viewing data in a system should be divided in the categories *Overview, Zoom, Filter, Details-on-demand, Relations, History* and *Extraction*. And for most of these categories, examples of how the interaction can be designed have been found in the study.

Overview & Zoom Concepts that could be used for supporting this kind of task are ones described under the overview chapter for users new to 3D navigation, for example the *View object* and the *Tour building* toolsets. These are designed to keep the users from losing their orientation which could be useful for a future HMI for the Quality Control System. The users of the HMI should not need to be experienced with 3D navigation to be able to use the interface.

Filter There are at least two different scenarios in the design proposal of this thesis that could require filtering: CENSORED BY ABB CORPORATE RESEARCH.

The second would be when the user wants to monitor a particular function of the system; in this case the data from all the other functions would be irrelevant and thus should be filtered out.

In the Filter chapter, a layering system in Unity 3D is described. This could be used for filtering by creating a number of layers in the application and add related objects to each one. Then the user will only see the currently relevant layers while using the application.

Details-on-demand To provide details-on-demand in the design, detailed information could be provided on the side of an object when it is selected. Another solution would be to use a search function in the interface that finds information from the manual, and all sources of information that could be relevant for the user. Search results could be settings, visible objects in the view and manual information.

Relations In the design proposal, relations are displayed by providing preset views that can be automatically zoomed in on. When a view is selected, the camera zooms in and displays only objects relevant for that view. These objects are placed in close proximity and irrelevant objects are filtered out.

History By providing the users with a "rewind" function in the interface, they can step back through history and reset camera angles and positions. This would give users an additional degree of freedom in the use of the system, as they can enter any view of their choice without the need to worry about whether or not they can return from a view.

Extraction As stated in the Extraction chapter, this is not a relevant issue at this time for the system. Possible extraction functionality could be CENSORED BY ABB CORPORATE RESEARCH.

Chapter 5

Augmented reality

Virtual Reality (VR) and 3D visualization is one approach for visualizing information in a way that is natural for the user. By making information appear in a familiar way, things become easier to understand [17]. Augmented Reality (AR) is a way of creating a virtual reality which augments the real world and creates a mix of reality and computer-generated images. AR allows real-time overlaying of visual objects on the real world, providing an augmented knowledge about the surrounding world.

Today AR systems are commonly seen using the combination of sense of sight and visual virtual information, but in fact; AR is based on any sense. Examples of AR could be speed information about passing cars projected onto the wind shield for a driver, or a real-time language translation system built into a pair of headphones.

This study focuses on vision and investigates if it's possible to apply this approach to improve ABB's Quality Control System. What issues must be taken into account when building such industrial AR system?

This chapter is structured as follows. First, we will look into AR, mixed reality and common AR displays. Second, an industrial research study will be reviewed. Based on the article studies, a conclusion will be drawn concerning implementation for ABB's Quality Control System.

5.1 Background

In 1964 Ivan E. Sutherland invented the first see-through head-mounted display (HMD) system [11]. The HMD was primitive and the user's virtual environment were built with simple wireframes, displayed in two miniature CRT's. Since then, the AR area has grown significantly. Thanks to today's development tools [3] and computer performance the availability of technologies for AR has increased dramatically. AR is used in a wide range of areas, both research and profitable applications. The systems used nowadays are far more complex and diverse.

When research was started, AR was lacking a definition. Soon both Azuma and Milgram made great effort to establish such criteria. Azuma et al. describes AR as based on any sense if the following criteria are fulfilled;

- combine real and virtual objects in a real environment;
- run interactively in real-time;

– and register real and virtual objects with each other [24].

The criteria denoted that AR applies to any sense, including hearing, smell, and touch. Milgram et al. on the other hand, coined the Reality-Virtuality Continuum which defined AR as part of a real-to-virtual environment, called Mixed Reality (MR) [9]. The taxonomy extends from the completely real to the completely virtual environments (figure 5.1).

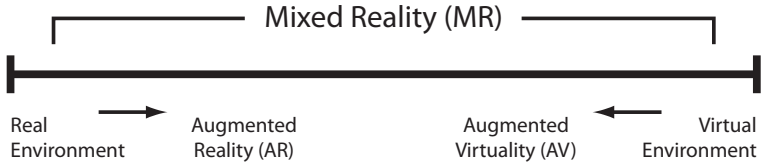

Figure 5.1: Reality-Virtuality (RV) Continuum

The purpose was to present an ordered classification, so that research, evaluations, etc, could be easily compared. For a better understanding of this taxonomy we can consider the differences between its subareas. At one end, one step from normal reality, visual AR technology allows its users to view the real world with virtual objects. The virtual objects can both be overlaid or mixed within the real world. On the opposite side, VE technologies fully immerse a user inside a complete virtual world. While the user is immersed, input from the real world is not accessible. In contrast to AR, Augmented Virtuality (AV) technologies are those when a virtual world is being enhanced by a real world object. For example, the user might see his/her own body in a virtual environment display, instead of a generic avatar, often used in VR technologies. In other words; this taxonomy extends from the completely real to the completely virtual environments.

5.2 Displays

When developing a visual AR application, the display is of great importance to the user. Researchers and companies have come up with a number of categories for these displays; this section covers some of those available today.

Based on the user's abilities to orientate the world, Billinghurst et al. presents a categorization for AR application displays [12].

– Head-stabilized

– Body-stabilized

– World-stabilized

Head-stabilized information can be described as images from a fixed camera, with this approach virtual layers that could be added on top of the reality are for example a timestamp for a security camera. Body-stabilized information can be described as a camera fixed to a point in space and that can rotate around all its axes and thus track the user's orientation. World-stabilized information can be described as a camera attached to a user's head, it tracks both position and orientation of the user and therefore it annotates the real on a higher level than head- and body stabilized displays [12]. This

type can be used for giving additional information to a fighter pilot inside their helmet etc.

When it comes to devices itself, Azuma et al. distinguishes three types: head-worn displays (HWD), handheld displays and projection displays. HWD displays are mainly presented in front of the user's eyes (called HMD when used in VR) and are used in several industrial AR projects today [28]. There are two common see-through approaches for HWD's. Video see-through captures the reality environment with help of a camera and displays it with an LCD, or any equivalent displaying technique. The optical approach, on the other hand, eliminates the video part by adding an optical layer, known. This means that the users see the reality with their own eyes, but through a transparent layer that acts as a display. This approach is commonly used in military flight cockpits.

Handheld displays are of a type that is becoming increasingly more common. This derives from the modern mobile phones and wearable computers available for customers nowadays. Modern phones and computers are often shipped with built in cameras, which makes it easy to present a video see-through system. Commercial applications for Apple iPhone for instance are under development that is intended for navigation [6]. The application recognizes locations by using the phone's built in camera, GPS and compass and displays information such as distance and names.

Finally, projection displays shows the desired information right upon the real world object One interesting approach is laser head-up displays. These displays are available for vehicles, and work by projecting information directly onto the wind shield. Drivers get valuable information such as driving assistance and speed information, by overlaying the real world.

5.3 Common pitfalls and design issues

This section presents common pitfalls and design issues that researches have shown are significant.

5.3.1 Object identification

Tracking mechanisms in AR systems are of great relevance for obtaining a highly functional environment. For example, positioning of real-world objects requires correct registrations for displaying proper information onto it. However, registration errors are often unavoidable [11]. Under such conditions, error estimation is useful. By estimating virtual object's position in real-world, changes in view may give enough info for keeping an acceptable position tracking. One common approach is to add artificial or physical markers to the environment, often black and white symbols in a very specific pattern. Marker tracking is widely applied in both commercial and non-commercial systems. These solutions are straight-forward, less time consuming, and cost-effective in manner of performance [20] [3]. Studies show that deterministic environments (such as indoor usages) are functioning better for tracking mechanisms [9].

5.3.2 Object occlusion

Several AR systems use virtual objects to overlap the real world. These systems are easy to implement and straight-forward. However, important information of the real world risk getting occluded using overlaid virtual objects. Correct occlusion is needed

to provide an authentic real environment superimposed with virtual objects for the user. This is vital for achieving a correct view of objects' position [31].

Numerous methods for estimating correct object occlusion involving real and virtual objects are available. Zhu et al. uses model registration to construct a virtual representation of the real environment [31]. In order to estimate depth, two different camera angles are used. Results show that this approach can efficiently and correctly register both virtual and real objects in an AR environment.

5.3.3 User interaction

Since AR present a way of displaying information next to the real world, researchers often claim that the interaction techniques also need to be revisited. 2D graphics can be difficult to present in such real-world environments. Interaction design and usability have been mainly focused on 2D graphical interfaces and virtual environments. AR systems can be very different from time to time as the physical environment change; positions of objects, angles etc is not always the same. An approach to interacting with such a system is by using tangible interfaces [20] [11] [22]. Tangible interfaces present the user interfaces as a virtual tool in the real environment, instead as for example a static 2D button. Interaction with objects is designed to imitate physical manipulation of objects. By using such an approach the interaction becomes more integrated with the reality, which is shown in Looser et al.'s [22] work.

5.3.4 User acceptance

According to Nilsson and Johansson [26] attention must be put on creating user acceptance towards AR technology. By providing a smooth integration between virtual and real world objects, Liarokapis et al. claims that a high accuracy and rendering quality is key elements for a successful system, that is, the system must appear as believable to the user. They claim that tracking of light sources and correct reflections applied to the virtual objects is one solution. Also realistic shading and shadowing of those objects are of importance for the users to accept the system.

5.4 Industrial case study

A study by Träskbäck and Haller [30] illustrates some important issues to address when designing an AR system for an industrial environment. They introduce AR into a safety-critical environment; an oil refinery. Oil refineries are demanding environments and require highly trained personnel. The purpose of their study was to train employees in a specific refinery process with the help of AR. The ultimate goal was to develop a usable training application which could be used during processing.

In order to reveal user requirements for such an AR system a field observation was conducted in the refinery. The study revealed a number of interesting requirements, which the application needs to meet in order to be efficient and usable:

- The AR tool should not draw the user's attention while the trainee moves around in the refinery.

- It is of great importance that the AR system is wearable, accurate, and can work in harsh environmental conditions.

– The user interface must offer information on the flow direction in the pipes.

A safety-critical and large environment needs to take these requirements into account in addition to basic user requirements.

5.5 Summary

In this article study we have seen that there are multiple displays and ways of orientating the user in an AR system. AR enhances a user's perception of the real world. The virtual objects display information that the user cannot directly detect with own senses and helps a user to perform real-world tasks.

The literature study does also show that AR applications have multiple error sources. Common pitfalls are object identification, interaction, occlusion avoidances of real world objects, etc. When it comes to the scope of preventing tracking errors, the article study shows that fixed indoor environments is well suited for AR applications. This will limit tracking mechanism issues and more. Other benefits of having a static or semi-static environment are the ability to reach realistic lightning and reflections on visual objects to obtain a realistic view. Imagine an operator or engineer receives an alarm, instead of stopping the whole production the information can be visualized in relation to the specific real object using an AR interface. For instance, information that is encapsulated behind covers etc can be viewable for the operator.

When designing AR applications, it is important to consider how to make the integration of real and virtual as seamless as possible. Each application must choose the best combination of techniques for detecting information from the real world and presenting electronic information to the user.

CENSORED BY ABB CORPORATE RESEARCH.

In future, operators might find the window in the control room as a perfect spot for both receiving and interacting with information. For example, using a projection display would allow see-through capabilities.

We have seen an interesting head-up laser display for in-vehicle use. This kind of non-mounted solution is appealing for use on the control room window, but there are multiple limitations with this approach. For example, how should the system verify the operator's orientation to achieve appropriate alignment of virtual objects? Thus, how should two operators be able to use this technique accurately? Probably alarm systems, etc would be easier to use instead of positioned virtual contextual data.

However, the control room is an ideal spot to begin looking into AR for solving both usability and monitoring issues for the future.

Chapter 6

Method

6.1 User study

Prior to our arrival in ABB's project a number of field studies and interviews with users of the Quality Control System interface were performed [27]. The results from this study were made available for the authors of this report.

The purpose of using these results is to identify important issues with usage of the system. This information is also important in order to know which areas of the graphical user interface that should be in focus.

A field study is conducted in the user's natural environment, where the system in question is used. This type of study can reveal *tacit knowledge*, which is knowledge that can only be revealed in an environment that the user is familiar with. These are things that the users themselves aren't aware of, for example habits, and company culture.

Other pros with this approach are that the user can feel more comfortable when answering questions in a well known environment. This provides more accurate results.

6.1.1 Procedure

CENSORED BY ABB CORPORATE RESEARCH.

6.2 Technology evaluation

6.2.1 Market scan

The first step in the technology evaluation was to perform a market scan. The initial study (or *market scan*) was performed by gathering data on 3D development software to get a large list of names with a brief description of relevant information for each. This was done by searching on the Internet, in forums and on relevant sites. When the list was done, six technologies were chosen for deeper evaluation.

The following criteria were established in cooperation with the project group at ABB Corporate Research, they were used to select technologies for deeper evaluation.

Support Support should be provided for the developers. Documentation such as manuals, tutorials and examples must be included either in the tool or be easily accessible online.

Licensing The license for a technology is optimally free of cost for development and distribution. Source code should be included with the license.

Productivity It should be time efficient to develop applications in the tool. The software should give the developer good feedback. Examples are error, and graphics handling (position, rotation, etc).

Established Technology The technology should be commonly used in modern applications. The company or organization that produces the tool must have existed for several years, with a stable history.

Performance The technology should make use of the computers' hardware for accelerating the graphics.

Installation The installation must be easy to perform and deployable applications must be stable for future system updates. There should not be any need for a server for running the software.

Deployment The technology should preferably be deployable on all computer platforms (Windows, Linux and Mac) and also in some kind of web format that can be run on all popular internet browsers.

Development procedure Development should be done in a modern and commonly used high level language, e.g. C++, C# or Java that has an official API easily accessible. It should be possible to write the code in a powerful editor such as Microsoft Visual Studio 2008 or Eclipse.

3D Visualization Capabilities The technology has to support 3D graphics in a non-limiting way. There should be libraries for graphical features without the need for creating them yourself. It is important that the engine can import the most commonly used graphic formats (3ds Max, Maya, Blender, etc).

6.2.2 Deeper evaluation

The in-depth studies of the technologies chosen in the market scan were performed by one person for two weeks per technology. Trial versions of each technology was obtained by downloading them from their respective websites, and installed on a Windows computer. The procedure was to explore the possibilities with the technology and evaluate it further with the criteria from the market scan.

The explored issues were: support for importing graphics from modeling software, adding lighting to the scene, implementing movement and control of objects and the camera (for example: spinning around the object with the mouse), accessing properties of objects and using external libraries.

A grading system was used to rate the technology's compliance with each criterion. A number ranging from 1 - 5 is set for each of the criteria described above, for each technology. The grades are then summarized to represent an overall grade of the technology. As there are 10 criteria the maximum grade is 50 for a single technology.

The grading scale is described below.

Score 1: Very low or no compliance with this criteria

Score 2: Low compliance with this criteria

Score 3: Average, the expected level

Score 4: Better than expected

Score 5: Impressing and beyond expectations

The 3D technology with the highest grade was recommended and used for an implementation of a conceptual future HMI.

6.3 Activity checklist

When designing systems, it is important to have an understanding of the context in which computer-related tasks are performed. This knowledge can reveal what users do, and how they most effectively can make use of a technology.

This kind of understanding should be gained before the design process has progressed too far, while it still is easy to modify and improve the design [19].

This type of contextual knowledge can be gained using activity theory [19] and the activity checklist [19]. Activity theory is a framework of theories for understanding the human mind, claiming that the human mind can only be understood in the context of human interaction with the world, and that activities are socially and culturally determined.

The activity checklist is a tool for analyzing an existing system or for supporting developers in designing a new system. It helps the evaluator comprehend how the users use the system. The structure of the checklist reflects five basic principles of the activity theory, and by using it one can ensure that most aspects of the theory have been covered in the analysis of the system.

In this thesis the activity checklist is used for evaluating a data gathering method. This method is used in the user study, described in chapter 6.1.

The activity checklist consists of four different categories of how context could influence the way users interact with the system. Each category covers various aspects of how the target technology supports human actions. Aspects could be the physical environment, purpose of the system, how usage evolves with experience and also how social factors affect the usage.

Means and ends This section covers how the system supports the users in achieving their goals. Does the system create conflicts between different goals?

Social and physical aspects of the environment How does the context affect the usage of the system?

Learning, cognition, and articulation Does the system reflect the users' mental representation on how things are done? Is users' learning process stimulated by the technology?

Development This area covers development of users' usage of the system. What factors affect users' behavior with the system? For example: what makes a user suddenly use CTRL + C instead of the right click + copy text?

6.4 Heuristic evaluation

Heuristic evaluation means having a small set of evaluators examine an interface and judge its compliance with recognized usability principles (the "heuristics"). The goal is to identify and isolate usability problems in the design and through these improve the user interface to enhance usability [21]. The heuristics used in this thesis is based on Nielsen's ten heuristics [17] from 1994, which is the most commonly used evaluation method today.

Visibility of system status The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.

Match between system and the real world The system should speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.

User control and freedom Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

Consistency and standards Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.

Error prevention Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.

Recognition rather than recall Minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

Flexibility and efficiency of use Accelerators – unseen by the novice user – may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

Aesthetic and minimalist design Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

Help users recognize, diagnose, and recover from errors Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

Help and documentation Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

6.4.1 Procedure

Before the analysis, the evaluators need to have information about the purpose of the system and who the intended users are. In this case the ABB Quality Control user interface and the operators which use the system.

When performing a heuristic evaluation a proper list of guidelines is chosen. In this case Nielsen's ten heuristics were chosen. The analysis was performed by two persons with human-computer interaction experience.

The procedure was as follows; the evaluators walked through the entire user interface briefly to get an overview and collect general usability issues. Next, the evaluators walked through the user interface once for each item in the list of guidelines. The data gathered were noted during the process and finally documented as a list of identified results.

6.5 Brainstorming

The brainstorming method used in this work is a variant of brainwriting. It is a method for rapidly generating ideas about products or processes by asking participants to write their ideas down rather than shouting them out as they would in a traditional group brainstorming session. Studies have shown that brainwriting can generate approximately 40% more ideas than traditional brainstorming [16]. This session was conducted with two persons for one hour in a private room. The participants wrote their ideas on post-it notes in silence, and after 40 minutes the ideas were presented. The results were compared and combined into a list of ideas for improving the current ABB Quality Control System HMI.

Collaboration is an important part of brainstorming session when finding new creative ideas. Participants may generate new and unexpected ideas by combining them with others. The aim of the method is to produce a large number of ideas [16]. To achieve a successful brainstorming, C. E. Wilson writes about the two most basic principles; quantity prior to quality and no criticism of ideas. This is something that was employed in this brainstorming session to find creative solutions for the design concepts.

Data from the user study and the heuristic evaluation of the current user interface were used as a basis for discussion in the brainstorming session. This basis for the discussion was used in order to make sure the participants produced ideas that fulfill the usability demands from the user study and the requirements from ABB.

6.6 Design concepts

A number of design concepts for a future HMI for the Quality Control System were developed. The results from the user study and the heuristic evaluation were taken into account while brainstorming for these concepts. The results from the brainstorming were then sorted, combined and further developed into a number of applicable functionalities for the upcoming concept demonstrator. The functionalities were collected and drawn by hand to sketches of a potential system, which were then used as blueprints for the implementation. The concept demonstrator contains a number of the design concepts presented in the results chapter 7.7; some of the concepts were not implemented due to time limitations.

6.7 Technology Demonstrator

The final part of the project involved implementing a concept demonstrator in the recommended 3D technology. The focus was on implementing the concepts that illustrate the 3D capabilities of the tool.

The graphics used in the application were originally used for marketing, such as brochures and ads. They were built in the modeling tool Cinema 4D and were optimized for real-time use in 3ds Max. Microsoft Visual Studio 2008 was used for coding and the language used was mainly C# on top of the Mono platform.

Chapter 7

Results

7.1 Technology Overview

The following technologies were a part of the initial overview study. The six technologies at the bottom of figure 7.1 were chosen for further evaluation. In Appendix A additional information can be found on each of the technologies from the overview study.

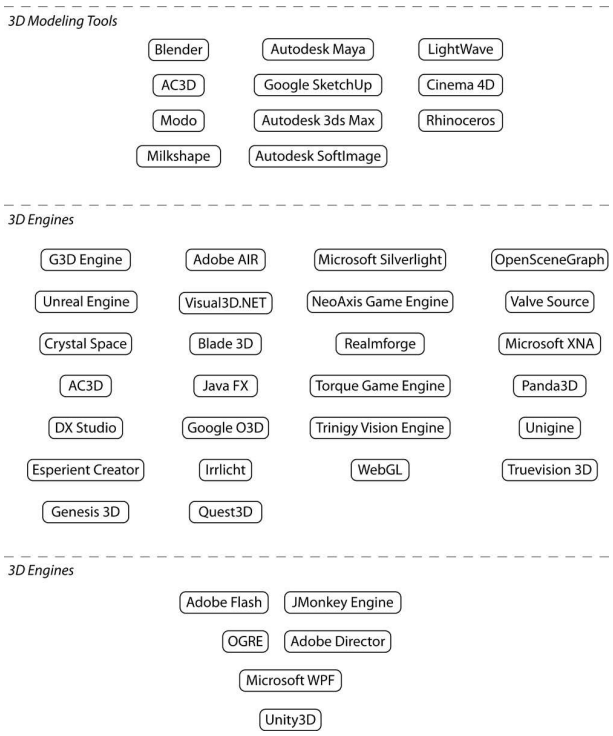

Figure 7.1: Evaluated technologies

7.2 Technology Evaluation

This section shows the result of the in-depth 3D technology evaluation. Each technology has a list showing the score and motivation of every criteria.

7.2.1 Adobe Director

Adobe Director is a developing tool for creating rich internet applications (RIA) [1]. It exports into a format called Shockwave which runs on web browsers primarily in Windows and Mac environments. It is not primarily intended for 3D development but can still offer decent 3D effects.

Evaluation

Support: 4
The documentation is good. Online documentation exists which contains info on all elements of the software and the scripting language.

Licensing: 4
The software development kit has a price of $999. Distribution is royalty-free.

Productivity: 4
Programming interaction in 3D is quite easy to accomplish with Adobe Director. Operations such as moving cameras and objects, handling input from mouse and keyboard, creating, grouping and removing objects, handling collisions, etc., were all quite easy to use.

Deployment: 3
The only output format is Shockwave, which is a RIA format It supports hardware 3D graphics and can be compiled to standalone applications in Windows and Macintosh environments and also be played in internet browsers that has the Shockwave plugin.

Performance: 5
Adobe Director uses hardware acceleration

Installation: 5
The install is a simple download from the developer's website.

Established technology: 3
Adobe Director is the predecessor of Flash, and has been around in the RIA scene since the early 90's. The latest version of the software was released in March, 2009. It is mostly used for presentations, mockups for GUI's and simple game development.

Development procedure: 2
All development is done in either Lingo or JavaScript. Lingo is scripting language mainly developed for Director. Applications built in Adobe Director can be developed on Windows and Mac OS platforms.

3D visualization capabilities: 2
The engine has problems importing graphics without losing important information such as textures. It does not support advanced textures or lighting. It only has very basic 3D support.

Summary

Because of the programming environment, the limited 3D capabilities and the lack of traditional GUI components, Adobe Director would not be a very good choice for a large project such as the Quality Control System HMI.

CENSORED BY ABB CORPORATE RESEARCH.

7.2.2 Adobe Flash

Adobe Flash is the most used multimedia plug-in for Internet web browsers today. The technology aims to enrich the experience for web users. Common usage areas are animations, movies, etc [2].

First version of Flash was released in 1997 by Macromedia. In 2005 Adobe Systems acquired Macromedia and have continued the developing process ever since.

The 3D support itself is limited in Adobe Flash Player 10. The technology supports basic 3D effects, not intended to interact with a scene-graph. Third party 3D libraries has therefore been created, for example Sandy 3D, Away 3D, Alternativa 3D and Papervision 3D. These libraries are commonly used for games, advertisement campaigns and further interactive 3D visualizations for the web.

Evaluation

Support: 5
Flash is a widely used technology. A tremendous amount of tutorials, documents and more are available online. There are over one million designers and developers within communities such as Adobe Developer Connection. Professional support is available for enterprises at a cost.

Licensing: 4
Flash as a technology and SDK is free to use under Adobe product license. The developer tool Flash Builder Professional costs US $699 for one license.

Productivity: 3
The available 3D libraries are easy to use and supply the developer with a lot of readymade functionalities. Compiling is quick, which makes it easy to see results.

Deployment: 4
Flash Player supports Windows, Mac OS X, Linux and Solaris platforms. Standalone versions of Flash are available by using Adobe AIR technology.

Performance: 1
Flash Player 3D graphics is not fully hardware accelerated, this means that the average performance of the graphics are slow, and high-level effects such as pixel lighting and particle systems cannot be used.

Installation: 3
Installing Flash Builder is easy, but third party libraries make the process more complicated.

Established technology: 2
Adobe has existed since 1982 and Flash since 1997. The 3D libraries on the other hand are all new and developed and maintained by third parties, which makes the available technology unreliable.

Development procedure: 3

Programming is mainly done in ActionScript using the Adobe Flash Builder editor for developers. Flash Builder is built upon the open-source editor Eclipse.

3D visualization capabilities: 1

Because of the lack of hardware accelerated graphics, the libraries functionalities are limited.

Summary

Software acceleration makes it impossible to create high performance 3D visualization systems. None of the third party 3D libraries provides the developer with a robust 3D graphics editor. The compiler is on the other hand quick, which makes it possible to get visual results within a short amount of time.

CENSORED BY ABB CORPORATE RESEARCH.

The developing tool Adobe Flash Builder (formerly Adobe Flex) is easy to use and supplies the developer with code completion. Flash is a superior technology for Internet multimedia distribution, but the lack of essential 3D libraries makes it difficult to use. The software acceleration limits the third party libraries to further extend and add powerful 3D functionalities. In the future, WebGL might be used to add hardware acceleration for Flash Player on the web.

7.2.3 Microsoft Windows Presentation Foundation (WPF)

WPF is a graphical subsystem for Microsoft's .NET framework. WPF contains a huge built-in graphical user interface (GUI) library and it also comes with a built-in 3D visualization library, called Viewport3D [5].

Microsoft Visual Studio is the recommended developing tool. Logic programming can be done in procedural (C#, VB.NET) or mark-up (XAML) code. The results are displayed in a viewport by Visual Studio.

WPF itself does not support 3D model importing. This means that some 3rd party vendor applications are needed when developing 3D applications for this system. Blendables, Right Hemisphere Deep Exploration and ZAM 3D are three proprietary exporters available on the market. The most widely spread, ZAM 3D, supports creation of models and exporting functionalities for 3ds and dxf files.

Evaluation

Support: 5

Developers can share their thoughts together in Microsoft's online community, MSDN. Documentation about the 3D environment and how to control Viewport3D is available with multiple examples. The API is well documented. ABB also already has agreements for direct support from Microsoft today.

Licensing: 4

Developer licenses are needed. Developing is done under Microsoft Visual Studio and Microsoft Expression Blend. Application distributions are royalty-free.

Productivity: 4

It is quick and easy to implement 3D functionalities using Viewport3D in Visual Studio. A design mode can be used to display scene changes etc.

Deployment: 3

A WPF application runs on Windows XP SP2 or newer as standalone versions. Applications can also be deployed for the web, as a XAML Browser Applications (XBAP). However, these applications run only in browsers using the Windows platform.

Performance: 5

WPF Viewport3D uses hardware accelerated graphics.

Installation: 4

All Windows versions above XP SP2 are prepared for using WPF and Viewport3D applications. Visual Studio is easy to install.

Established technology: 4

Today WPF is mainly used for developing Windows GUI applications and is well established as such. However, the 3D part of WPF cannot yet compete with technologies primarily focused on 3D development. Viewport3D has existed since .NET version 3.0, released in 2006.

Development procedure: 5

Viewport3D can be developed in both procedural and mark-up based code, such as C#, VB.NET and XAML. The entire process of developing can be done in either Microsoft Visual Studio or Microsoft Expression Blend.

3D visualization capabilities: 3

Viewport3D is not fully loaded with 3D visualization functionalities. It includes only the basics.

Summary

Developing for WPF Viewport3D is straight forward. Mark-up makes it easier and faster for developers to build the scene-graph and resources. Separation between code and models are easily done by adding models to the resource folder as DynamicResources.

One other interesting feature in WPF is UIElement3D. The UIElement3D class library makes it possible to wrap interactive 2D content on 3D surfaces in WPF. Many times like a texture being mapped over a mesh.

This evaluation has focused on Microsoft Expression Blend 2. Compared to previous version, Expression Blend 3 has been updated with more 3D capabilities which make it easier to use.

CENSORED BY ABB CORPORATE RESEARCH.

7.2.4 OGRE

OGRE is one of the most popular open-source 3D engines available today [7]. Thousands of developers are collaborating and sharing their thoughts together at OGRE's online community. A tremendous amount of add-ons and ready-made scripts are also retrievable online.

OGRE deploys on multiple platforms such as Windows, Linux and Mac. Underlying graphical low-level libraries are DirectX and OpenGL and developing is done by coding in C++.

Evaluation

Support: 3
The online community is both huge and active. Currently OGRE has five team members working. Purchasable consultations are available through the founders' companies.

Licensing: 5
OGRE is free to use under their open-source license. Alternative license without any restrictions is available for an unknown amount.

Productivity: 2
OGRE is mainly a 3D engine without any additional support. By using third party add-ons, productivity may increase. During evaluation many third party projects were unreliable.

Deployment: 5
OGRE supports both OpenGL and Direct3D which makes projects deployable on multiple platforms: Windows, Mac and Linux.

Performance: 5
OGRE is hardware accelerated using both Direct3D and OpenGL.

Installation: 3
The installation is relatively easy when using the installer version. Due to OGRE's nature with a community of multiple developers; add-on tools etc are all installed individually.

Established technology: 3
The team is constantly under change and they are all working on other projects next to developing the engine, which makes it future unreliable.

Development procedure: 2
The OGRE installer makes it easy to compile and deploy within Microsoft Visual Studio.

3D visualization capabilities: 4
OGRE supports a wide range of 3D visualization functionalities.

Summary

The scene-graph and API are both easy to understand and quick to start working with, compared to other code-based engines. However, OGRE lacks visual feedback and it forces the developer to recompile for every running. There are third party graphical scene editors, but many of these are developed by independent persons causing the tool to only contain a limited amount of functionalities. The documentation is well written, but there is a huge amount of poorly organized information that is difficult to navigate and read. While OGRE third party tools offer support for almost everything, the quality varies. Some of the tools crashed repeatedly during evaluation.

In conclusion, OGRE supports a high range of 3D functionalities and makes almost everything possible to develop with its add-on library, but the lack of consistent quality makes it unreliable, non-productive and time consuming.

7.2.5 JMonkey Engine

The JMonkey Engine is a Java library for programming 3D games and environments [4]. It is very popular in the independent game development community and is commonly used for smaller game projects. It is an open source technology and is thus developed by independent developers and fans. It is a well recognized technology and Sun plans to use it in their project: "Project Wonderland", a toolkit for building collaborative 3D worlds. Sun is currently using this toolkit for building software that their remote employees can use to work and collaborate together with in a 3d environment.

Evaluation

Support: 2
There are Javadocs on the classes and a small wiki on the project's website. These contain some helpful information, but since it is community driven, not everything is documented properly. There are very few comments in the javadoc about functionality of library classes and the user's guide with instructions is not complete. The forums are active and give fast responses.

Licensing: 5
The engine is open source under the BSD License.

Productivity: 1
The concept of the workflow with this engine is simple; write code, compile, see results and repeat. This workflow gives low feedback to the developer and thus when there is a problem it takes a lot of time to figure out how to resolve it.

Deployment: 4
The projects can be deployed on any platform that has the Java Runtime Environment and supports the LWJGL (Lightweight Java Graphics Library).

Performance: 5
JMonkey engine uses hardware acceleration through OpenGL for its graphics.

Installation: 2
The install consists of downloading a number of Java libraries and importing them into a project in an appropriate Java editor. There is no editor for the JMonkey engine.

Established technology: 3
The JMonkey Engine has a big community of users, and has a big forum. It has existed since 2003. However, it is currently a community-driven open source project with no supporting company.

Development procedure: 4
All development is done in Java. Thus, all functionality that can be implemented in Java can be implemented in the applications produced. The engine is written in Java, and all coding is done in the same language. Therefore, the engine can be developed on all platforms with the Java Runtime Environment.

3D visualization capabilities: 3
The engine comes with a number of converters that can be used for importing graphics. External converters can also be found. There are examples of games

made in this engine that makes use of almost all modern graphical features available for OpenGL.

Summary

This engine has produced a number of powerful 3d graphic applications, but has a steep learning curve and is poorly documented. It is primarily a game engine and its functionality is optimized for that purpose. It could probably be used for building a 3D HMI but there are better options available.

7.2.6 Unity 3D

Unity is a 3d game development environment [8]. It has a graphical user interface and allows the user to represent the 3d scene graphically before the code is compiled. It uses an "asset library" which contains all elements of the application. The assets can be models, materials, scripts, cameras, lights etc. These assets can easily be added to the scene using drag-and-drop technology and programmed using C#, JavaScript and Boo. All game logic runs on the open source .NET platform Mono project.

Evaluation

Support: 5
The support is very good, the entire API can be found on the website. There are sections both for the scripting and for more general information describing the structure of the software and all the different functions in the editor. The engine has a big active community and bug reports are handled quickly.

Licensing: 5
The standalone runtime may be redistributed royalty-free, as long as it is produced using a licensed installation of the Software. A license for Unity Pro costs $1499. A collaboration system "Unity Asset Server" cost $499. It is possible to buy licenses for deploying to both Apple iPhone and Nintendo Wii. Source code license is also available for an unknown cost.

Productivity: 5
Unity has a graphical editor in which all available graphics, sounds and assets can be accessed easily. Applications are designed in a graphical editor; objects, scripts and graphics can be placed using drag-n-drop. This is an important advantage compared to engines based only on code. The remaining coding can be done in an external editor (such as VS2008) that provides code completion.

Deployment: 4
Unity can export to a web format. This web format is played using unity's own plugin for browsers; this can be downloaded from their website. It can also be distributed as a streaming service, a standalone application for Windows and Mac OS X, and also as an OS X Dashboard Widget. Unity can also deploy applications for the Apple iPhone and Nintendo Wii. It cannot produce standalone applications for Linux.

Performance: 5
Unity uses hardware acceleration for its graphics.

Installation: 4

Unity is installed by downloading it from the website. The asset server for version control and collaboration must be run on a Linux, Unix or Mac server.

Established technology: 3

Unity has been used by several commercial game projects. The editor seems very popular in the game development community. It is developed by Unity Technologies, Denmark.

Development procedure: 5

Scripting is done with either JavaScript or C#. C# can be edited in external editors. The engine for compiling C# is Mono. The Unity editor can be run at either Mac OS X or in Windows environments. It can deploy to Mac, Windows and for browsers. Details can be found under the "deployment" requirement above.

3D visualization capabilities: 5

Unity has support for importing the most popular graphics formats. It has support for almost all graphical features that are used in commercial games.

Summary

Unity's biggest strength is that it provides a graphical interface for programming the applications, in which the user can easily design the application visually. This enables higher productivity than purely code-based graphic engines. Other advantages are its ability to import most popular graphics formats, good support and deployment to the web as well as to standalone applications.

7.2.7 Summary

This chapter presents a summary of the evaluated technologies (figure 7.2.7). The highest result got a total score of 40. The score was based on the criteria presented in chapter 6.2.1. The technology with the highest score was Unity 3D, and the biggest reason that it is recommended is its productivity and rich library of 3D visualization features. More details and a motivation of how the requirements were evaluated together with a motivation to the ranking scheme are presented in chapter 6.2.2 and 7.2.6.

Title	Evaluated 3D graphic					
	Unity 3D	Adobe Director	Java jMonkey Engine	OGRE	Adobe Flash	Microsoft WPF
Support	5	4	2	3	5	5
Licensing	4	4	5	5	4	4
Productivity	5	4	1	2	3	4
Deployment	4	3	4	4	4	3
Performance	5	5	5	5	1	5
Installation	4	5	2	3	3	4
Established technology	3	3	3	3	2	4
Development procedure	5	2	4	4	3	5
3D visualization capabilities	5	2	3	4	1	3
Total scores	**40**	32	29	33	26	37

7.3 User Study

CENSORED BY ABB CORPORATE RESEARCH.

7.4 Activity Checklist

In an attempt to verify the user study's validity in accordance to the activity theory an analysis was performed. This analysis was based on the evaluation version of the activity checklist [19] to find out whether the key areas of context (specified by activity theory) were covered in the interviews. The purpose of performing this analysis was to identify areas that might need further attention in this thesis.

The strategy of performing the analysis was to first select areas on the checklist that was relevant for this work. The questions used in the interview were analyzed according to the chosen areas.

CENSORED BY ABB CORPORATE RESEARCH.

7.5 Heuristic Evaluation

The following results describe a heuristic evaluation of the current Quality Control System HMI. The method used is described in chapter 6.4. Results are organized under eight categories corresponding to Nielsen's heuristics, and indicate where and how the current system could be improved in terms of usability.

CENSORED BY ABB CORPORATE RESEARCH.

7.6 Brainstorming Session

The brainstorming session was conducted in a private room were the participants wrote ideas on notes for a limited time (45 minutes). When the time was up each participant presented their ideas and the notes were put on a billboard and the ideas were discussed and combined.
These are the results: CENSORED BY ABB CORPORATE RESEARCH.

7.7 Design concepts

The following section describes a number of design concepts for a future Quality Control System HMI built on 3D graphics. The concepts are all based on a user study [27] and/or requirements from ABB. The procedure of developing these concepts is described in the method chapter 6. For each design concept a motivation is included and a description on what issues the concept could possibly solve.

7.7.1 Process overview

CENSORED BY ABB CORPORATE RESEARCH.

7.8 Technology Demonstrator

A technology demonstrator was created to show some of the previously presented design concepts, this demonstrator was implemented using Unity 3D.

CENSORED BY ABB CORPORATE RESEARCH.

Chapter 8

Conclusion

This report has evaluated a number of technologies based on nine criteria, each technology was graded according to how well it complied with the criteria with a score between 1 - 5 for each. The total score indicate that Unity 3D is the most appropriate technology for the purpose of building a HMI for the ABB Quality Control System out of the evaluated technologies. Unity 3D was chosen primarily because of its high productivity and big library of 3D functionalities.

A few design concepts have been developed. These are described in the results chapter. They are based on issues found in a user study, in a heuristic evaluation of the current HMI and produced in a brainstorming session. The concepts are suggestions on how to solve these issues.

Finally, a technology demonstrator was implemented in the recommended technology. It was built to demonstrate the design concepts in a 3D environment and to show the power of Unity 3D. The demonstrator was presented and received good criticism for Quality Control professionals at five separate occasions within ABB.

8.1 Future Work

In order to cover more important aspects of system usage, the user study conducted in this project could be modified and repeated. It was evaluated in chapter 6.3 with the activity checklist method. The purpose of this was to determine the validity and see how a future user study should be designed to take the context into account. The results of this study show (among other things) that information should be gathered on which operations require most time and effort to learn, it should also investigate how external tools conflicts with the system.

8.2 Limitations

This thesis report has been conducted with the following limitations.

- The design concepts must be evaluated in a proper user study to determine whether they have any value in practice compared to the current solution.
- The evaluation of the 3D development tools only covers six technologies. Therefore, it cannot be guaranteed that the ideal solution has been found.

- All technologies have been evaluated in how well they comply with some chosen criteria and graded thereafter. The grading was done in a subjective way and it can thus not be compared with a similar study with other evaluators.
- The evaluation does only cover performance in that hardware accelerated technologies receives a higher score than software accelerated ones. No other performance has been measured.

Chapter 9

Acknowledgements

We would like to thank **Susanne Timsjö** for being our supervisor and giving us the opportunity of doing our thesis at ABB Corporate Research. She has always been positive and helpful and has helped us a lot with ideas, feedback and structure when it was needed.

Martin Olausson who was our project manager in our time at ABB helped us a lot with structuring the project, arranging meetings and presentations and making sure the project was on the right track.

George Fodor for providing us with visions and ideas on how a future system should work.

Daniel Sjölie for providing us with support from Umeå Universitet and correcting and commenting the final report.

Magnus Larsson for letting us do our master thesis work at ABB Corporate Research.

Dilip Kota for providing us with the opportunity to do our master thesis at ABB.

Markus Holm for teaching us how the physical process was working and sharing his wisdom about regulation.

Christine Mikkelsen for the information about the HMI of the Quality Control System, and teaching us how to use it.

Anders Hanberg for supporting and arranging meetings with XA at ABB.

Daniel Rosendal & **Johan Ziprus** for introducing us to ABB Corporate Research.

References

[1] Adobe Director homepage. http://www.adobe.com/products/director/ (visited 2009-05-15).

[2] Adobe Flash homepage. http://www.adobe.com/products/flash/ (visited 2009-05-12).

[3] ARToolKit is a software library for building Augmented Reality (AR) applications. http://www.hitl.washington.edu/artoolkit/ (visited 2009-09-29).

[4] jMonkey Engine homepage. http://www.jmonkeyengine.com/ (visited 2009-05-12).

[5] Microsoft Windows Presentation Foundation homepage. http://windowsclient.net/wpf/default.aspx (visited 2009-05-14).

[6] Nearest Tube is an Augmented Reality (AR) applications for iPhone 3GS. http://www.acrossair.com/artoolkit/ (visited 2009-08-11).

[7] OGRE homepage. http://www.ogre3d.org/ (visited 2009-05-12).

[8] Unity 3D homepage. http://www.unity3d.com/ (visited 2009-05-13).

[9] Antti Aaltonen and Juha Lehikoinen. Exploring augmented reality visualizations. In *AVI '06: Proceedings of the working conference on Advanced visual interfaces*, pages 453–456, New York, NY, USA, 2006. ACM.

[10] Unity Technologies ApS. Unity 3D web site. http://unity3d.com/support/documentation/Components/Layers.html (visited 2009-06-29).

[11] Ronald Azuma, Yohan Baillot, Reinhold Behringer, Steven Feiner, Simon Julier, and Blair MacIntyre. Recent advances in augmented reality. *IEEE Comput. Graph. Appl.*, 21(6):34–47, 2001.

[12] M. Billinghurst, J. Bowskill, N. Dyer, and J. Morphett. An evaluation of wearable information spaces. In *VRAIS '98: Proceedings of the Virtual Reality Annual International Symposium*, page 20, Washington, DC, USA, 1998. IEEE Computer Society.

[13] Dempsey Chang, Laurence Dooley, and Juhani E. Tuovinen. Gestalt theory in visual screen design: a new look at an old subject. In *CRPIT '02: Proceedings of the Seventh world conference on computers in education conference on Computers in education: Australian topics*, pages 5–12, Darlinghurst, Australia, Australia, 2002. Australian Computer Society, Inc.

[14] Andy Cockburn, Amy Karlson, and Benjamin B. Bederson. A review of overview+detail, zooming, and focus+context interfaces. *ACM Comput. Surv.*, 41(1):1–31, 2008.

[15] Niklas Elmqvist. Balloonprobe: reducing occlusion in 3d using interactive space distortion. In *VRST '05: Proceedings of the ACM symposium on Virtual reality software and technology*, pages 134–137, New York, NY, USA, 2005. ACM.

[16] Elizabeth Gerber. Using improvisation to enhance the effectiveness of brainstorming. In *CHI '09: Proceedings of the 27th international conference on Human factors in computing systems*, pages 97–104, New York, NY, USA, 2009. ACM.

[17] Nielsen J and Mack R. *Usability Inspection Methods*. Wiley, New York City, NY, 1994.

[18] Patrick W. Jordan. *An Introduction to Usability*. Taylor & Francis, London, 1998.

[19] Victor Kaptelinin, Bonnie A. Nardi, and Catriona Macaulay. Methods & tools: The activity checklist: a tool for representing the "space" of context. *interactions*, 6(4):27–39, 1999.

[20] Fotis Liarokapis and Robert M. Newman. Design experiences of multimodal mixed reality interfaces. In *SIGDOC '07: Proceedings of the 25th annual ACM international conference on Design of communication*, pages 34–41, New York, NY, USA, 2007. ACM.

[21] G. Lindgaard. *Usability testing and system evaluation*. London: Chapman and Hall, 1994.

[22] Julian Looser, Raphael Grasset, and Mark Billinghurst. A 3d flexible and tangible magic lens in augmented reality. In *ISMAR '07: Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 1–4, Washington, DC, USA, 2007. IEEE Computer Society.

[23] Andrian Marcus, Louis Feng, and Jonathan I. Maletic. 3d representations for software visualization. In *SoftVis '03: Proceedings of the 2003 ACM symposium on Software visualization*, pages 27–ff, New York, NY, USA, 2003. ACM.

[24] Paul Milgram and Fumio Kishino. A taxonomy of mixed reality visual displays. *IEICE Transactions on Information Systems*, E77-D(12), December 1994.

[25] Ingrid Mulder, Henk de Poot, Carla Verwij, Ruud Janssen, and Marcel Bijlsma. An information overload study: using design methods for understanding. In *OZCHI '06: Proceedings of the 18th Australia conference on Computer-Human Interaction*, pages 245–252, New York, NY, USA, 2006. ACM.

[26] Susanna Nilsson and Björn Johansson. Acceptance of augmented reality instructions in a real work setting. In *CHI '08: CHI '08 extended abstracts on Human factors in computing systems*, pages 2025–2032, New York, NY, USA, 2008. ACM.

[27] CENSORED BY ABB CORPORATE RESEARCH.

[28] Björn Schwerdtfeger, Daniel Pustka, Andreas Hofhauser, and Gudrun Klinker. Using laser projectors for augmented reality. In *VRST '08: Proceedings of the 2008 ACM symposium on Virtual reality software and technology*, pages 134–137, New York, NY, USA, 2008. ACM.

[29] Ben Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *VL '96: Proceedings of the 1996 IEEE Symposium on Visual Languages*, page 336, Washington, DC, USA, 1996. IEEE Computer Society.

[30] Marjaana Träskbäack and Michael Haller. Mixed reality training application for an oil refinery: user requirements. In *VRCAI '04: Proceedings of the 2004 ACM SIGGRAPH international conference on Virtual Reality continuum and its applications in industry*, pages 324–327, New York, NY, USA, 2004. ACM.

[31] Jiejie Zhu and Zhigeng Pan. Occlusion registration in video-based augmented reality. In *VRCAI '08: Proceedings of The 7th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry*, pages 1–6, New York, NY, USA, 2008. ACM.

Appendix A

3D technology overview

This chapter presents all techniques, but excludes six ones chosen for deeper evaluation.

A.1 Modeling tools

In this section all overviewed modeling tools are listed.

A.1.1 Cinema 4D

Cinema 4D is a high end 3D graphics modeling tool. It is produced by MAXON Computer GmbH of Friedrichsdorf, Germany. It has been developed since 1993 and has been used for producing a number of movies such as Surf's Up, Beowulf and The golden compass.

Summary

This software is used for modeling 3D graphics, and not for building 3D applications. Therefore it has not been evaluated in this project.

A.1.2 Google SketchUp

Google Sketchup is a basic 3D modeling tool for creating conceptual design. There are different levels of licenses; Single-user 381 euro, Free version, less functionality and thea pro version available for 8 hours. It is developed by Google.

Summary

This software is intended for - As the name implies - quick sketches or concept illustrations, and not for designing advanced 3D graphics with complex lighting and texturing.

A.1.3 Rhinoceros

Rhinoceros is an advanced 3D modeling tool, that is specialized in NURBS for professional industrial design. The editor runs on Windows, and one commercial single-user license costs 1695 euro. Support and documentation is available at the website. It is

developed by McNeel, USA. The company claims that the software is market leading in Industrial Design.

Summary

The software is intended for professional industrial design and is not software that are very suitable for the purposes of this study.

A.1.4 Modo

Modo is an advanced 3D modeling tool with focus on usability. The editor runs on Windows & Mac OS X. The license alternatives are: new Modo 302 costs $895, and an upgrade to 402 costs $395. It has all documentation and a huge forum at the website.

The company is Luxology, 2001, USA. The software has been used by Pixar Animation studios for creating animated movies. Claims to be easier to use than comparable software (3ds Max, Maya).

Summary

This software is intended for design of 3D models, and not for creating 3D applications, thus it is not evaluated in this study.

A.1.5 Milkshape

Milkshape is a simple 3D modeling software that runs on Windows, the license is shareware and it supports very basic 3D functionality. The main purpose of this application is to edit preexisting graphics for games.

Summary

This is a very basic 3D modeling tool and has no functionalities for creating applications.

A.1.6 AC3D

AC3D was initially developed on Amiga and one of the first graphical modeling tools. AC3D has existed since 1994 and was purchased by the current owner, Invis Limited, in 2002. The application is running on all mayor platforms; Windows, Linux and Mac OS X.

AC3D comes with a great amount of import and export functionalities, which makes it well used for Second Life world modelers. The cost for a single license is $79.95.

Summary

The developer community seems to have disappeared and the gallery does not convince. There is a great lack of "high demand" 3D graphics renderings.

We have chosen to not use this application for modeling purposes in manner of these conclusions and the overall feeling.

A.1.7 Blender

Blender is the most used open-source graphical modeling tool today. The application is built by the non-profit organization The Blender Foundation since 1993 and is supported by a huge user community.

Blender supplies their users with all type of 3D modeling tools, such as texturing, rigging, water simulations, animations, particles and more. Extra functionalities are retrievable as plug-ins and extras.

Blender is compatible with platforms such as Windows, Linux, Mac OS X, FreeBSD, Irix and Solaris.

Summary

Blender is a well common modeling application and consists of a tremendous amount of functionalities. Blender supports even real-time deployment capabilities with their Blender Game Engine tool. The documentation is good, but not logically retrievable and categorized.

When it finally comes to the use of this powerful 3D modeling, things are highly unreliable. Almost every file format is importable, but with multiple limitations. Third party importers and exporters are listed on the webpage, but many of them suffer of children's disease which makes Blender unreliable to use.

Blender's user interface is difficult to start using, and does not implement the operating systems look-and-feel. During our overview evaluation phase, many bugs where showing up within a small period of time. Therefore, Blenders was not chosen to be the main modeling tool for this thesis.

A.1.8 Autodesk Maya

Maya is a high-end 3D modeling tool from Autodesk Inc. It has existed since 1998 and is mainly used by professionals creating animations and architectural scenes.

Maya runs on Windows XP, Mac OS X, Linux and Fedora. The application provides the designers with three API's for scripting capabilities. One single license of Autodesk Maya costs from US $6490.

Summary

Since Autodesk Maya is expensive and mainly used by movie professionals, we decided not to use this tool further in our evaluation phase.

A.1.9 Autodesk SoftImage

SoftImage is another high-end 3D modeling tool from Autodesk. The first version of the application was released in 2000 and was later aquired by Autodesk in 2008. It supports scripting without code and also provides a set of game engine exporters, such as Microsoft XNA, Valve Source, Epic Unreal Engine and more. SoftImage runs only on Windows computers and is purchasable from US $3700.

Summary

SoftImage is a top of the line software from Autodesk. Though, we have decided not to use this tool further in our evaluation phase.

A.1.10 Autodesk 3ds Max

3ds Max is a powerful and well-recognized 3D Modeling tool. It is used for working with 3D models, textures, lighting etc. The license cost is around 4000 USD depending on type (network or standalone). Support is provided through developer forums, telephone "Incident support" ($65 per problem) and an article search function on the website. It is produced by the company Autodesk, USA. It is a well-known technology for creating graphics that is used for creating big titles in both animated movies and commercial games.

Summary

3ds Max is one of the most familiar names there is when it comes to 3D modeling applications. A trial license was used in this project to create, edit graphics and convert graphics for the game engines that was evaluated. It is a powerful tool that has a big community of professional users, and has support for most modern 3D graphics formats

A.1.11 Newtek LightWave

Lightwave is developed by the American company Newtek since 1994. It is a powerful and well-recognized tool that was used to create effects in big projects such as Terminator - The Sarah connor chronicles, Batman - The dark knight and Iron man. Lightwave runs on Windows and Mac OS X computers and costs $995 for version 9.

Summary

Since Lightwave is intended to be used by professional artists for high level graphics, this software has not been used in this project.

A.2 Real-time 3D engines

In this section all overviewed real-time 3G engines are listed.

A.2.1 RealmForge

RealmForge is not under development anymore; Visual3D.NET is its successor.

A.2.2 Visual3D.NET

Visual3D.NET is a relatively new game engine from Realmware Corporation. It is based on Microsoft XNA and was released in 2008. The 3D engine supports a high range of features and possibilities, such as next generation graphics and a physics system.

The technology is integrated with Microsoft Visual Studio for developing purposes, while it also provides an all-in-one scenario and GUI builder. Languages supported are C#, C++, Visual Basic, Java, Lua and IronPython.

The game engine is deployable on Windows computers and soon Xbox360. A single enterprise edition costs from $19500. Evaluation versions are available.

Summary

Visual3D.NET supports many features and is easy to use. The community on the other hand is small and the project has just begun. We have chosen to not evaluate the product further, since Visual3D.NET is still in beta.

A.2.3 Blade 3D

Blade 3D is an all-in-one game engine technology for Windows and Xbox devices. The documentation is well written with multiple tutorials. Easy-to-use toolkits are also provided to make the development work more productive. The built in development environment runs in Windows. Application can be deployed on both Windows and Xbox 360. One single license costs from $999,50 for each year.

Summary

Digini announced on the 30th of July 2009 that stopped working with the technology. Subscriptions will be canceled within 6 months.

A.2.4 Truevision 3D

Truevision 3D is a 3D engine created by Truevision 3D, LLC. The technology is developed and deployed on Windows computers. Truevision 3D supports 3D up to DirectX 9.0c.

The documentation is wiki based and poorly categorized. Support can also be retrieved from a web forum with 6000 other developers. The supported programming languages are C#, C++, Visual Basic and Delphi.

Summary

Because of the lack of high performance 3D functionalities and poor documentation we have drawn the conclusion to not evaluate this product further.

A.2.5 Microsoft Silverlight

Microsoft announced Silverlight as a new competitor to Adobe's Flash technology in 2006. It is an Internet multimedia technology and runs on Windows, Mac OS X and Linux computers.

Silverlight is mainly developed in Visual Studio 2008 and Expression Blend as a .NET technology and is free to use under Microsoft Public License.

The final version of Silverlight 3 was announced on the 9th of July 2009. It presents a lot of new 3D capabilities. A limited number of 3D effects in Silverlight 3 are now hardware accelerated.

Summary

Compared to high performance 3D engines, Silverlight is still far away to fulfill the high demand vision of 3D visualization. The technology, on the other hand, runs in almost every computer as a web technology. Therefore, we have chosen to not evaluate this product further.

A.2.6 WebGL

WebGL is an upcoming new 3D technology for web usages. It will work as a hardware-accelerated open source 3D standard. This will be done without any plug-ins. Instead Khronos Group works on a JavaScript API binding to OpenGL ES 2.0 which can be discovered through the web browsers. The specification will be royalty-free for all developers. Google, Mozilla and Opera will officially support this upcoming industry standard.

Summary

This technology will be a standard for 3D graphics in internet browsers, but is still under development. Therefore it has not been evaluated in this project.

A.2.7 Valve Source

The source engine is a high performance 3D game engine, it can run on Windows platforms and Xbox 360 consoles. It can be developed in Microsoft Visual Studio. This engine is used for top of the line games and license information can only be gained under an NDA. The developers are Valve Corporation, founded in 1996, one of the leaders in the computer gaming industry.

Summary

This is a very powerful engine that is used to develop top-of-the-line games. License information is only available under non-disclosure agreements. Therefore it has not been evaluated in this project.

A.2.8 Google O3D

Google O3D is a browser plug-in for Rich Internet Applications (like Adobe Flash). It runs on Windows XP/Vista, Mac OS X 10.5 and Linux 32-bit x86 distributions. It is developed in JavaScript and is Open Source. Support is provided through system descriptions and manuals and a forum is available at Google Code. It provides hardware accelerated 3D graphics.

Summary

O3D is still in its experimental beta phase and the code is fully exposed because of the use of JavaScript. Therefore it has not been evaluated in this project.

A.2.9 DX Studio

DX Studio is a 3D engine and developing tool that runs on Windows XP / Vista. The license cost is 588 EUR for one Pro Edition license. It provides support through documentation and walk-throughs, a small community based forum and the user base is about 30 000 [Wikipedia]. It is produced by Worldweaver Ltd and they released the first version in 2005. ActiveX technology is used for integrating the 3D engine as a component in other applications.

Summary

The software was released in 2005 and still the user base now is only about 30,000 which means that it is not a popular technology. Therefore it has not been evaluated in this project.

A.2.10 Esperient Creator

Esperient Creator is a 3D engine and developing tool that runs on Windows 2000/XP/Vista. The license cost is US $2400 for the enterprise version. Training etc are offered as consulting services. The forum community is small and not very active.

Many instructional videos are available through the official homepage. The company is Esperient Corporation and it was founded in 2007 and they launched Esperient Creator in late 2008.

Summary

This technology is currently not very established and therefore it has not been evaluated in this project.

A.2.11 Quest 3D

Quest 3D is a 3D graphics engine and developing tool that runs on Windows 2000/XP/Vista.The license cost is 10k EUR for a single VR Edition license

It has a small forum community. The company Act-3D started in 1997 and released the first version of Quest3D in 2000. It can deploy executables and it is also possible to deploy on the web as an ActiveX controller.

Summary

This technology was not part of the technology evaluation part of this project, as there was not enough time to test this software since there were more attractive technologies available.

A.2.12 Microsoft XNA

Microsoft XNA is a developing environment for 3d games for Windows platforms and Xbox 360. The editor runs on Windows & Mac OS X. Applications produced can only be distributed in the XNA Creator's Club. Support for using the software can be found at Creator's Club Online, forums at the website and also at MSDN online. Coding is done in any .NET programming language, and Visual Studio can be used for developing in. It is developed by Microsoft.

Summary

As the applications produced in XNA only can be deployed in XNA Creator's Club it is not a viable choice for this project.

A.2.13 Trinigy Vision Engine

This engine is a professional 3D game engine. Licenses can only be acquired by companies that intend to use the software for commercial game development. It is produced by Trinigy Inc, 1999, Germany. The engine is used by big commercial game developers such as Ubisoft, Firefly & Atari.

Summary

This engine is intended for professional game development.

A.2.14 Torque Game Engine

The Torque game engine is a commercial 3D engine, that can deploy to Windows, Linux, Nintendo Wii, Xbox 360, iPhone, Mac OS X. There are two types of licenses; Independent developer $295 and Commercial $1495. Support is provided through documentation, resources and a big forum at the website. Coding is done in Torquescript (C++). The engine is produced by GarageGames.com, 2000. It supports high quality 3D and is used for several commercial games.

Summary

This is a powerful commercial engine intended for experienced developers.

A.2.15 G3D Engine

G3D engine is a low level 3D engine that runs on Windows, Linux, FreeBSD and Mac OS X. Visual Studio 2008 can be used for developing in. The license is Open source, BSD. Support is provided through a small forum at the website, an online manual, and a forum at sourceforge.net, for registered users. Coding is done in C++. It is produced by an open-source team. It supports high quality 3D, but does not have a built in scene graph.

Summary

Since the engine does not provide a built in scene graph, it requires the developers to create their own graph. This result in more effort to produce an application compared to engines with built-in scene graphs.

A.2.16 Panda 3D

Panda 3D is a game engine that handles graphics, audio, I/O & collision detection. It is open source and runs on all platforms. It can be developed in Python & C++. The license is open source (the BSD license). Support is provided through a well-used forum and an online manual. It is developed by an open source team; The Panda Development Team, since 2002, based in Pittsburgh. It has been used for making Disney's Pirates of the Caribbean Game.

Summary

This engine has many positive attributes; it is well-recognized in the open source game development community, and has been used for creating commercial games.

A.2.17 Unigine

Unigine is a 3D Game engine. It runs in PC and Linux environments, and coding is done in C++. There are two license options; a business license from $24800 and an evaluation kit without C++ API and source code. Support is available through technical support via e-mail, phone and instant messaging and documentation. The package includes a modeling tool. It is produced by the company Unigine Corp, initiated in 2002, based in Russia. The first beta got released in 2004 and is under rapid development.

Summary

This is a professional game development tool that is suitable for large scale 3D game development.

A.2.18 NeoAxis Game Engine

NeoAxis game engine is a 3D game development software that runs in Windows environments. It is developed in .NET and has three types of licenses varying in price from $99 - $9999 Support is provided through documentation and a small forum on the website. The company is Neo Axis Group, LTD, since 2006.

Summary

This technology is a small SDK best suited for independent game developers. At first glance these tools looks useful, but overall not a very impressive technology A small team with a small forum and still in beta phase makes this technology difficult to invest time and money in. On the other hand, they have a nice approach for the future. Mono supports are soon built in which makes the deployment to Mac OS X available. Also, NeoAxis supports Window in Microsofts .NET environment which makes it easy to integrate in WPF applications.

A.2.19 Unreal Engine

The Unreal engine is a professional 3D graphics engine for PC, Xbox 360 & Playstation 3. The engine has been used to produce games such as Mirror's Edge, Gears of War, Unreal Tournament 1-3 and Harry Potter: The Philosopher's Stone. The engine is currently available in versions 1-4. A license for one platform (PC, Xbox or Playstation 2) for version 2 is $350k. An additional platform is priced $50k

Summary

This engine is one of the biggest and most well-recognized commercial engines there is. The engine is very powerful, and the price is very high.

A.2.20 Adobe Air

Adobe Air is a Rich Internet Application with deploy capabilities, it runs on Windows, Linux and Mac OS X. It can be developed in Adobe Flex and its own editor; AIR SDK which supports HTML, XML, ActionScript, Flash, Audio, Video and more. The license for Adobe AIR SDK is free. The development environment Adobe Flex costs from US $249 for one license. There are restrictions when distributing Adobe AIR Runtime. It

is developed by the Adobe Corporation, and has existed since 2007. A stable version was released in 2008. Adobe Air supports basic 3D. Adobe AIR allows rich internet applications (HTML+Flash) to be deployed onto the desktop.

Summary

Adobe Air is primarily a technology for browser usage, such as websites. The 3D support is limited as it is not its primary purpose.

A.2.21 OpenSceneGraph

OpenSceneGraph is a high performance 3D engine toolkit. The toolkit is developed as an open-source project and is free to distribute as a proprietary application under OpenSceneGraph Public License. OpenSceneGraph has existed since 1998, when it started as a hoppy project. Today there are 365 contributors in the project.

Applications developed in OpenSceneGraph can be run under Windows, Mac OS X, Linux and with limited support in IRIX, Solaris and FreeBSD.

Summary

We have chosen to not go further with this technology. OpenSceneGraph does not stand out in the area of 3D technologies. During a short evaluation period this technology turned out to be complicated to install and develop in.

A.2.22 Irrlicht

Irrlicht is a 3D engine that runs on Microsoft Windows, Linux, Mac OS X, Windows CE.

It is licensed under the Zlib license (similar to GNU GPL). Support is provided through a online API for C & .NET, a big developer forum and lots of tutorials at the website.

Coding is done in C++ and .NET and there are also language bindings for Java, Ruby, Lua, Python etc. Irrlicht is open Source and the project was first started in 2002.

Summary

Seems easy to use, and supports a number of languages (with addons) but it is hard to find impressing examples of applications developed in Irrlicht.

A.2.23 Genesis 3D

Genesis 3D is a graphics engine that runs on Microsoft Windows 95/98/XP/2000. It can be developed in environments such as Visual Studio. The license terms are US $10,000 for each title, otherwise free to use for non-commercials. It is supported by a big community and is developed by an open source group called Eclipse Entertainment.

Genesis3D was founded by Eclipse Entertainment in 1998.

Summary

This technology is yet another open source 3D game engine; it does not really provide any features that make it stand out.

A.2.24 Crystal Space

Crystal Space is a 3D game engine that runs on Microsoft Windows, Linux, UNIX and Mac OS X and is licensed under the lesser GNU GPL. Support is provided through a forum, manual and tutorials at the website and the language for developing is C++. It is developed by the open source group The Crystal Space Group and has existed since 1998.

Summary

Just like Genesis 3D, this is another game engine that is very similar to most other open source game engines.

A.2.25 Java FX

Java FX is an internet multimedia technology that deploys a RIA format (Rich Internet Application, like Adobe Flash and Microsoft Silverlight). It can be run and developed on any platforms that support the Java Runtime Environment and Java ME.

The license is free and support can be found on developer forums, code support can be found on the Java Online API. Java FX is produced by SUN Technologies. 3D is currently only supported using external libraries.

Summary

This is a browser technology developed by SUN in order to provide an option for Java developers to create nice looking graphics in their GUI applications. But the technology has currently no hardware support for 3D.