

Augmented maintenance of powerplants: A prototyping case study of a mobile AR system

Gudrun Klinker, Oliver Creighton, Allen H. Dutoit, Rafael Kobylinski, Christoph Vilsmeier,
Bernd Brügge

*Technische Universität München, Fachbereich Informatik, Lehrstuhl für angewandte
Softwaretechnologie, Arcisstr. 21, D-80333 München
{klinker,creighto,dutoit,kobylins,vilsmeie,bruegge}@in.tum.de*

Abstract

Augmented reality (AR) research has progressed in great strides over the past few years. Most current demonstrations focus on providing robust tracking solutions since this is the most critical issue when demonstrating AR systems. An issue that is typically neglected concerns the online access, analysis and visualization of information. The information required by AR demonstration systems is kept to a minimum, is prepared ahead of time, and is stored locally in form of three-dimensional geometric descriptions. In complex mobile settings, these simplifying assumptions do not work.

In this paper, we report on recent efforts at the TU Munich to analyze the information generation, retrieval, transmission, and visualization process in the context of maintenance procedures that are performed in nuclear powerplants. The use of AR to present such information online has significant implications on the way information must be acquired, stored, and transmitted. This paper focuses on pointing out open questions, discussing options for addressing them, and evaluating them in prototypical implementations.

1. Introduction

Augmented reality research has progressed in great strides over the past few years. By now, we have become accustomed to seeing spectacular demonstrations at events such as Siggraph, ISAR, or the Hanover Fair, showing, for example, how to play memory games and telecommunicate by presenting video streams on physical cardboards [7], how to play virtual AirHockey and Borderguard games [12], and how to visualize city planning projects and the laws of optical reflections and refractions on a table [14]. Further demonstrations present the use of AR to assist a skilled mechanic in a disassembly procedure [4] and how to insert a door lock into a car door [8].

For any such demonstration to work robustly and reliably on a show floor, it has to be well tuned; many simplifications have to be made. User and mobile object tracking are typically the central issues that decide to a large extent whether the demonstration can be presented successfully or not – i.e., whether users feel immersed in a real environment while wearing a head mounted display (HMD).

An issue that is typically neglected concerns the online access, analysis and visualization of information. The information required by AR demonstration systems is kept to a minimum, is prepared ahead of time, and is stored locally in the form of three-dimensional geometric descriptions so that it can be inserted quickly into the spectator's view. In complex mobile settings, these simplifying assumptions do not work. One of the essential aspects of such applications consists of being able to access relevant information from one or more large servers as suitable for the current location and work context, to transform it into suitable presentation formats (metaphors, glyphs), and to render it on available multi-media output devices – all in due time and resolution.

Feiner et al. have presented several systems addressing such issues. In particular, the Touring machine demonstration of a “Mobile Journalist’s Workbench” shows how visitors at the campus of Columbia University can visualize past historic events exactly where they took place. To this end, they have built in collaboration with the department of journalism a database of audio and video documents describing such events [6]. Curtis et al. also report requiring large amounts of detailed wiring information for their wire bundle assembly application [3]. They had to combine and transform line drawings from several 2D and 3D descriptions into meaningful sequences of cable layout instructions. Molineros and Sharma have presented a system that uses an automatic assembly planner to generate machine assembly instructions, visualized online with an AR-module while a worker performs the assembly task [9].

Our long-term goal is to provide an augmented reality framework that would enable information systems

developers to reuse and adapt AR components to their problem domain. Our approach has been, on the one hand, to define an application independent augmented reality framework [1], and, on the other hand, to conduct case studies for specific applications. In this paper, we report on a case study recently conducted at TU Munich in the domain of the maintenance at nuclear powerplant. At the chair for applied software engineering, we regularly conduct semi-annual project courses with senior students to analyze the information processing cycle in a specific industry-related context. Courses are conducted in tight conjunction with the Human Computer Interaction Institute of Carnegie Mellon University, with students in both institutions working collaboratively on separate aspects of the problem [2]. Typical results of such courses are a detailed study of an industrial context, including a requirement analysis and application scenarios, followed by a general system architecture suited to address a selected set of problems and a prototypical implementation of some of the suggested solutions.

With regard to the plant maintenance scenario, we are addressing the process of semi-automatically generating electronic work procedures from paper manuals, placing them into a database, and then retrieving them via an unreliable network from the database according to suitable user interactions and the current user location and work context. Use of augmented reality technology to view the information has important implications relating back to the tasks of generating, storing, retrieving, transmitting, and visualizing electronic instruction manuals.

These implications are of general value to the process of generating or adapting large information and communication systems for the purposes of mobile AR presentations. As such, the issues are probably addressed by researchers in other projects as well [3][6][9]. Yet, they are typically not emphasized. This paper focuses on pointing out open questions, discussing options for addressing them, and evaluating them in prototypical implementations. In the next sections, we present the application domain of the system under development, we then discuss the general architecture and long-term design goals, we then describe the current status of the project (not all subsystems were completely implemented during the project course), and, finally, we conclude with the lessons learned.

2. Maintenance in nuclear powerplants

2.1. Current situation

The maintenance of nuclear powerplants is expensive. The complexity of nuclear powerplants results into a voluminous design and procedural documentation. Economic factors force operators to minimize the down time of the powerplant, thus putting pressure on the time required for maintenance. Finally, safety regulations require operators to perform periodic inspection and

maintenance of critical subsystems as well as account for the execution and outcome of these activities.

To address these issues, inspection and maintenance is dictated by a rigid collection of procedures that describe every step that technicians need to perform. The detail level of each procedure is such that it requires minimum training from the technician and minimizes the probability of errors. Figure 1. describes a simple example of such a procedure.

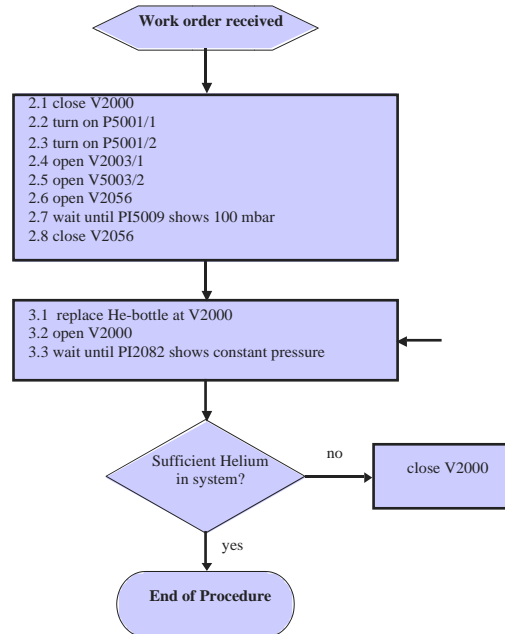


Figure 1. Helium flushing system procedure (simplified)

As most nuclear powerplants are several decades old, inspection and maintenance are typically supported by a paper-based system. Technicians assigned to a procedure are handed a paper checklist describing all the steps they need to perform. As the technician executes a step and observes the result, he marks the checklist accordingly. The technician communicates with the supervisor via a portable radio.

While this system meets the complexity and safety constraints associated with nuclear powerplants, it is not cost effective and can be error prone. The technician must go back and forth between his work and the checklist, thus interrupting the workflow. The information available to the technician is restricted to the checklists and manuals that he brought with him. The communication between the technician and the supervisor is low bandwidth, while sufficient for supporting routine procedures, it is not ideal for dealing with unexpected problems. These considerations lead to considering mobile augmented reality systems as an alternative to paper-based systems.

2.2. Visionary system

The system under consideration in the paper (STARS, for Sticky Technology for Augmented Reality Systems) replaces the paper checklist and documentation with Interactive Electronic Technical Manuals (IETMs). An IETM is an electronic version of the paper manual, which can be executed like a software program. The IETM encodes all possible paths through a maintenance procedure and describes to the technician the next step to be accomplished. Based on input from the technician, the system acknowledges the completion of the current step selects the next relevant step from the IETM, and describes it to the technician. When the procedure is completed, the technician can move on to a different location in the powerplant and proceed with the next maintenance procedure.

STARS technicians are equipped with a wearable computer with a head-mounted display (HMD). The wearable computer has a second display, an LCD plate that can be held by the technician or stowed away in his belt (Figure 2). The HMD is used for displaying augmentations in the technicians' environment (e.g., labels and arrows identifying the devices the technician operates), while the LCD plate is used to display rich information items, such as longer text descriptions, blue prints and prerecorded videos illustrating complex procedures. The technician interacts with the system via a speech interface.

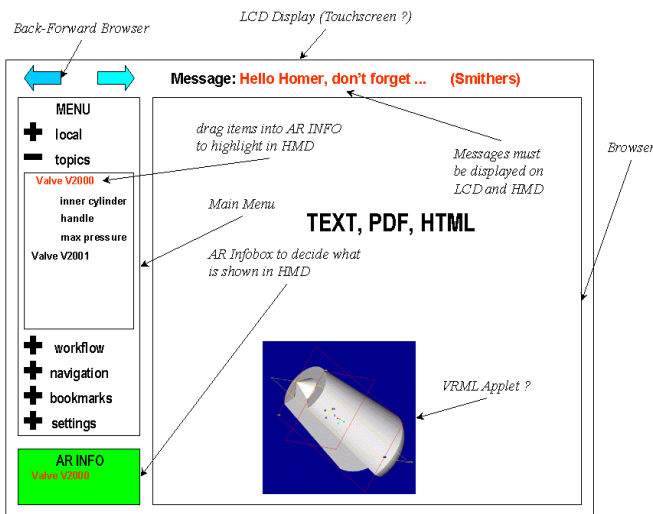


Figure 2. LCD plate with a two-dimensional view

The technician uses access credentials (e.g., a smartcard or an iButton) to open doors to reach the area where the maintenance procedure takes place. The system takes advantage of these access mechanisms to track the current room location of the technician. A camera attached to the

technician's head captures what the technician sees. This camera is used for fine-grained tracking of the position of the technician (so that the augmentations are correctly registered in the environment) and for monitoring by the supervisor. The wearable computers are connected to central servers via a wireless network. Given the powerplant environment, the quality of service and reliability of the wireless network may vary. The video feed from the head-mounted camera can also be transmitted on the wireless network when needed. Figure 3 is an example of the view on head mounted display we would like to be able to provide during the helium flushing system procedure described above.

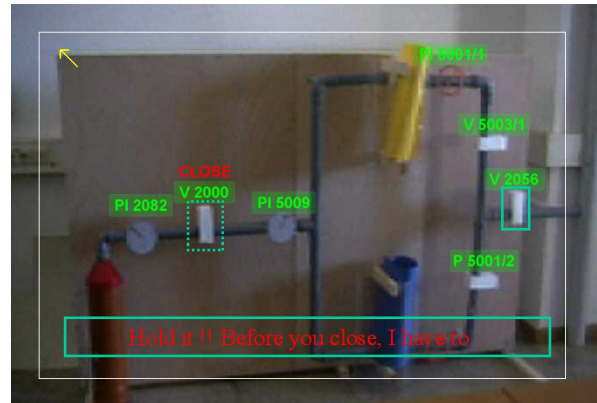


Figure 3. Idealized HMD view of the helium flushing system (mockup) , as conceptualized in the visionary system

2.3. Goal and scope of the STARS project

The goal of the STARS project was to investigate architectural and technical issues raised by the augmented reality system described above. While the use of augmented reality addresses usability and cost effectiveness issues during maintenance, the realization of this system presents several challenges, including:

- *IETM*. Information currently available in paper manuals and blueprints needs to be digitized and cross-linked. The digital form of these manuals, called IETMs (Interactive Electronic Technical Manuals), can then be served to the user as a sequence of steps that depends on the user's context. The IETMs, however, also need to be linked to the relevant elements of the 3D world model, for the augmentations to be useful.
- *Network connection*. The network connection between the wearable and the servers is episodic and unreliable, hence the wearable has to function as a self-contained system for periods of time.
- *Multi-modal user interface*. The user interface needs to accommodate several modalities, such as speech, augmented reality, 2D and 3D presentations, so that the technician and the supervisor can best accomplish their work.

The STARS project adopted a prototyping approach in which several options were evaluated and the most promising ones implemented in a demonstration prototype. In a first step, the project focused on issues introduced by augmented reality, and, hence, focused on the technician's interface. However, the STARS architecture was designed to accommodate the addition of the supervisor interface at a later date. The next section describes the STARS architecture.

3. System architecture

STARS includes one or more server (a workstation serving IETM data) and any number of mobile clients (a wearable attached to a technician's belt) connected via a wireless network. The server stores all the IETMs necessary for the maintenance of the powerplant, while the mobile clients retrieve individual IETMs on demand. The IETMs are then executed on the mobile client. An AR component tracks the user's position and displays the augmentations in the HMD as a function of the current IETM step and of the user preference.

While network service is typically unreliable, some areas of the nuclear powerplant (called 'data stations') are easily recognizable by the technicians and guarantee a reliable network connection. Moreover, when in a 'data station', technicians may temporarily connect their wearable via a fixed high-bandwidth network to prefetch needed IETMs before the maintenance procedure starts.

The STARS software is divided into three main subsystems (see Figure 4): the IETM subsystem (depicted in gray), Network subsystem (depicted in blue), and the User Interface subsystem (depicted in white).

IETM. The IETM subsystem includes an IETM interpreter localized on the mobile client and an IETM database on the server. The IETMs are retrieved transparently through the Network subsystem. Once the mobile client retrieves a complete IETM and initiates the maintenance procedure, the IETM interpreter does not need the network connection. The IETM subsystem is described in more detail in Section 4.

Network. The Network subsystem includes a Network proxy component on the mobile client and a Network interface component on the server. The Network proxy is responsible for monitoring the quality of service of the network and tracking the coarse-grained location of the user indicated by the access control mechanisms. The Network interface keeps statistics on the downloading of IETM elements. Together, both components prioritize the prefetching of items for a particular mobile client. The Network subsystem is described in Section 5.

User Interface. The user interface manages all modes of the user interface, including processing speech commands and displaying information on the LCD plate. The user interface subsystem includes an AR component that is dedicated to tracking the user's head and displaying augmentations in the HMD. The User Interface subsystem

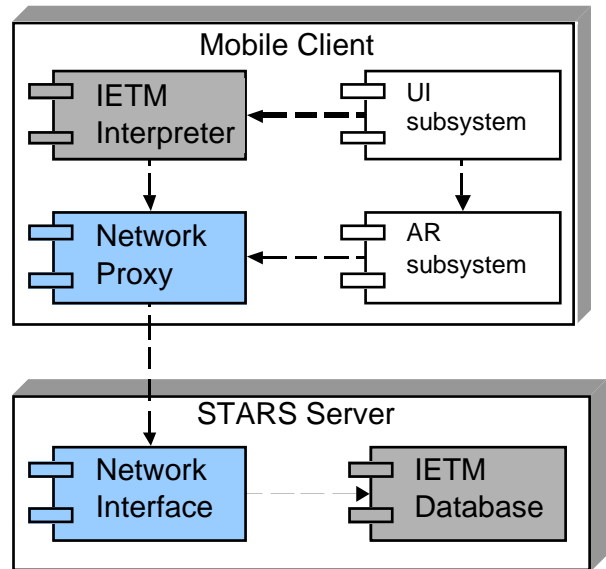


Figure 4. High-level STARS architecture.

is written in such a way that it can operate in standalone mode, without network connection. The User Interface subsystem is described in Section 6.

The communication among subsystems on the mobile client is managed by CORBA middleware. This enables the dynamic configuration of different components based on the current wearable available and the current task. The communication between the mobile client and the servers is entirely managed by the Network subsystem.

Next, we describe each subsystem in more detail, starting with the IETM subsystem.

4. IETMs

To replace paper-based manuals, maintenance manuals have to be converted into Interactive Electronic Technical Manuals (IETMs, [16]). One of the major components of this conversion process involves parsing and interpreting the documents.

Each IETM represents a significant component of a nuclear powerplant, such as the entire Helium Flushing System. The IETM includes a list of subparts of the component, and for each component, a list of work orders, such as the Bimonthly Check of the Helium Flushing System. Each work order is structured as an ordered sequence of steps, which can be repeated as loops, if necessary. Within each sequence, individual steps list physical actions to be performed by workers on well-specified physical objects, such as closing valve V2000. Figure 5 shows an example IETM describing the work order for the maintenance procedure shown in Figure 1.

Since procedures for maintaining or repairing complex systems are inherently algorithmic, the sequences and looping construct, in combination with branching and

subroutine constructs are enough to describe virtually any technical manual. Objects that are represented in IETMs each correspond to specific physical entities with unique IDs. Loops can refer to groups – and even entire hierarchies–of similar physical objects that are specified by containers of identities. Actions are explained to the user with text strings. In typical mobile IETM applications, such strings are shown to the user on a mobile LCD screen. In STARS, we also provide facilities to read the instructions out loud by a speech synthesis system and to present selected aspects as augmentations in an HMD. As additional help, video or audio streams can be included into an IETM and presented on the LCD by user request.

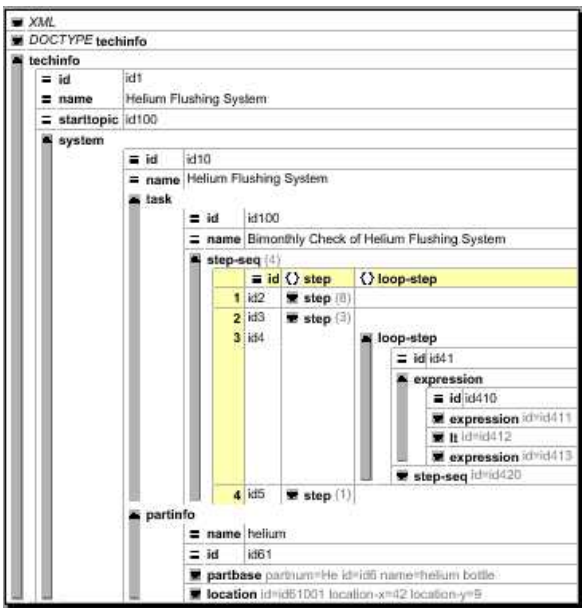


Figure 5. Sample IETM describing the Helium Flushing Procedure.

When converting existing manuals into an IETM structure, we currently are able to generate correct syntactic descriptions. But we cannot yet add semantic knowledge, such as defining the exact set of objects to which a maintenance action has to be applied ("close all valves in that part of the room"). To this end, we currently provide an interactive environment in which the flow of control is determined by the interaction with the user and the current state of the work order.

The result is an IETM system dealing with data, behavioral and state information about maintenance procedures. Data are stored in XML format, behavior as control directives in the IETM structures, and state information as global variables maintained by the IETM Interpreter while executing an IETM. All information is encapsulated in a relational IETM database which exposes a high-level interface for retrieving and searching the data. A typical IETM database consists of several hundreds of thousands of pages. The access function is optimized for

the retrieval of individual XML nodes, as this is the most common case when interpreting an IETM. An IETM can represent several gigabytes of data.

5. Network access to IETM database

As stated in previous sections, one of the essential aspects of STARS consists of accessing relevant information from one or more large servers as suitable for the current location and work of the mobile user. The information presented to the user include text documents, graphics, and videos illustrating complex procedures. It is not known in advance which documents will be accessed by the user, because this depends on external factors, such as user education and experience level. It is not possible to put all potentially relevant information on the mobile device because this would exceed the storage space of today's devices. As the users have to be mobile, all data must be transmitted over wireless networks. However, we assume that across the powerplant area, fixed Ethernet network stations ('data stations') are installed and easily accessible by users for faster service.

5.1. Ideas and concepts

The low bandwidth of the wireless network requires the implementation of a caching subsystem. Loaded documents are stored in a local cache and further requests for the same document are served from the local cache instead of being loaded from a server machine, as already supported by many popular web browsers [5]. For coping with network latency and temporary network outages, documents should be loaded in the cache before the user requests them the first time [10]. The fitness of the set of documents to be prefetched is critical to ensure efficiency. Prefetching too much data results in a waste of network bandwidth. Prefetching not enough data or prefetching the wrong data results in a high miss rate. Therefore, it is critical to prefetch exactly those documents that are needed next by the mobile user.

In STARS, we use access statistics and context information to decide which documents should be preloaded on the mobile device. We analyzed three context criteria: The current location of the mobile user, its work order, and the list of the documents the user has requested so far.

Context-based statistical prefetching records user requests, annotates them with context information, and stores them into a database for later retrieval. When this is done over a longer period of time, it should be possible to predict which documents are interesting for a user in specific contextual situations.

Another requirement addresses the problem of temporary network outages. In today's network protocols (e.g. HTTP, IIOP, Java RMI) a connection breakdown is interpreted as an error condition and must be handled by the application developer (e.g., the developers of the

IETM subsystem). However, in STARS, temporary network outages are a normal situation and should be handled transparently. In the STARS architecture, the Network subsystem is responsible for handling such outages and shielding them from the IETM subsystem. Here, we looked at two approaches. First, while transmitting data from server to client, we insert synchronization points into the transmitted byte stream. After a connection breakdown, the transfer can then be restarted from the last synchronization point and need not start from the very beginning. Second, when a network connection goes down unexpectedly, we attempt to establish an alternative network connection, possibly over another network access medium. If, for example, a WaveLAN connection is interrupted, we can try to establish a network connection over Bluetooth and continue the data transfer over the new connection until a new WaveLAN connection can be resumed. Obviously, this approach influences the way the network subsystem is designed. The network subsystem has to be independent of network technology, so that it is possible to integrate a number of low level network access technologies, such as, Ethernet, WaveLAN, Bluetooth, GSM, UMTS, or Infrared.

5.2. Implementation

We have implemented the context-based statistical prefetching component, using the document history as context information. The prefetching component records each document request that was made by the application and stores the identifier of the requested document in a server side database, along with a timestamp. For prediction of future document requests, these database entries are read and its time dependencies are analyzed. Then a dependency graph is generated that represents the dependencies among document requests. While this is done, younger entries are weighted more heavily than older entries. This procedure assures that the system adjusts itself to changes in the documentation structure of the powerplant or its technical documentation.

We also implemented the handling of network outages. We chose to integrate the synchronization point approach described above, where one synchronization point is inserted every 128 data bytes. Consequently, after a connection breakdown, at most 128 bytes must be retransferred a second time, no matter how large the transferred document is.

The implementation of the network subsystem is transparent to the application, as we took the already existing IETM access component and implemented a client proxy for use by the application installed on the client device. This proxy implemented the same interface as the IETM access component so that it was possible to switch transparently from the real IETM access implementation to the proxy. The client proxy forwards incoming document requests over the network to the server IETM access implementation. It is also responsible

for prefetching documents and storing them in a cache on the client side. Moreover, the client proxy monitors the network connections and handles the repetition of data transfers in case of network outages. This all happens transparently to the application.

6. User interface and augmented reality subsystem

From a user's point of view, the user interface is the central component of the system. According to user input, as well as to changing user positions and work progress, it uses the (invisible) networking facilities to request IETMs from the central server. Upon arrival, it parses the new work order and steers several output modalities to present the information in suitable formats to the user. As described in section 3, the STARS user interface consists of several components that handle various input (speech recognition, keyboard input, access card, iButton) and output (HMD, LCD plate, speech synthesis) modalities.

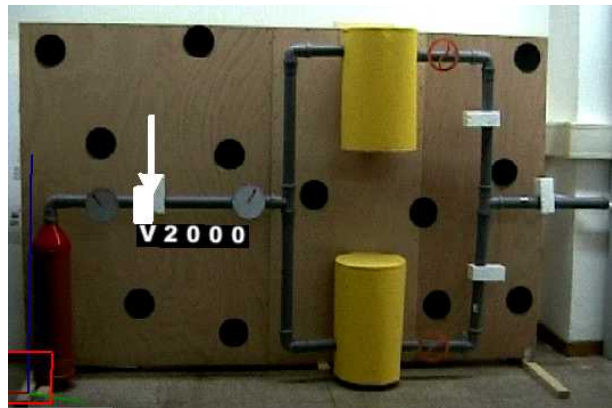


Figure 6. Augmented scene, highlighting valve 2000 which is the focus of the worker's current repair instruction.



Figure 7. Photo of a video clip instructing workers how to close a valve (featuring a student of the praktikum, Martin Thiede).

As a result, a typical communication between the user and the STARS system can look as illustrated in the following scenario:

The user requests to see the next maintenance step saying “Computer, next step”. A computer-generated voice responds “Close valve V2000”. At the same time, valve V2000 is highlighted in the worker’s head mounted display (Figure 6). The LCD screen shows the first picture of a movie how to perform the closing procedure (Figure 7), as well as an overview of the work order.

The user interface component of STARS realizes all application specific behavior while relying on the AR components for tracking the user position and the rendering of the augmentations. In this section, we focus on the augmented reality component and its relationship to the rest of the system.

The AR subsystem consists of three components which are needed to render information in the user’s field of view: a tracking component to determine the user’s current position, a rendering component to display relevant information, and a communication component to track the user’s work progress.

6.1. Communication between the augmented reality and user interface subsystems

To generate augmentations of the real world, we need a model describing the real objects involved. Augmentations will involve subcomponents of this scene, highlighting them in specific colors, as well as with arrows and labels. To dynamically adapt such augmentations to the user interaction state we need to divide competences between the AR and UI subsystems.

- At one extreme, the UI subsystem sends 3D display lists to the AR subsystem describing the entire set of graphical primitives that should currently be displayed. In this configuration, the UI subsystem has to maintain the entire geometrical scene description and generate its own visualization techniques as needed to visualize the current work instruction. The AR subsystem acts as a user tracking and data overlay server, showing whatever the UI system sends.
- At the other extreme, the AR subsystem owns the geometric model, combined with a detailed state model that relates the current work context to appropriate augmentation strategies. The UI subsystem then merely informs the AR subsystem of state changes, and the AR subsystem changes its augmentation primitives accordingly.

Both approaches have drawbacks. In the first option, the UI system has to implement a lot of visualization expertise, e.g. how to highlight or label an object using suitable glyphs, colors and annotation positions. Furthermore, communication channels will be cluttered with repetitive messages containing geometric data. In the latter case, the AR subsystem has to mirror much of the

UI’s state model. When new concepts are added to the scope of user interaction functionalities, both subsystems need to be modified. Neither approach optimally modularizes both subsystems so that they can be reused for other tasks.

Rather than pursuing any one of the extreme approaches, we have set out to determine a set of communication primitives that separate visualization expertise from controlling the user’s interaction state. In the context of the plant maintenance task, we have identified the following set of primitives:

- Highlight object (obj, form)
- Label object (obj, text)
- Display information (text, screenpos): id
- Clear information (id)
- Edit information (id)
- Set compass (object, form)
- Is visible (obj): bool

In this approach, the AR subsystem owns the geometrical scene model. It also maintains its own set of glyphs (e.g., arrows) and visualization strategies. The UI subsystem, on the other hand, maintains the current user state. According to state changes, the UI tells the AR subsystem, which geometric objects to highlight or label in what form (i.e., using which visualization strategy). The UI can also request that special messages (e.g.: warnings) be shown on the screen independently of the user’s viewing direction. Such information can be edited or cleared according to the evolving work context. Finally, we have adopted Boeing’s metaphor of a compass [3] – a screen-based glyph that tells the user which direction to turn his head when the current augmentations are outside the field of view. In principle, the compass could also be complemented by voice commands “turn left”, “turn right” etc. However, due to lack of time, the compass concept wasn’t fully implemented.

6.2. Rendering

Our geometric scene model consists of a list of objects such as pipes, valves, etc. (Figure 8). Each object refers to a distinct part of the scene. It is represented as a distinct entity with its own

- a unique identifier,
- geometric and photometric description and transformation matrix relative to a global coordinate system,
- an optional labeling string with an associated transformation matrix to position the string relative to the object,
- a further transformation matrix to position a 3D arrow, and
- modal parameters to describe the current drawing style of the object.

The modal parameters of each scene object are set according to the commands sent from the UI. Currently, we provide the following different drawing styles: an

object is either drawn normally, highlighted, or not drawn at all. If it is drawn normally, it can be shown in full, or as a wire frame. If it is highlighted, it is always shown as filled white polygons with a white 3D arrow pointing at it from a specified position. Optionally, we also present a white label at a prespecified position – relative to the object position.

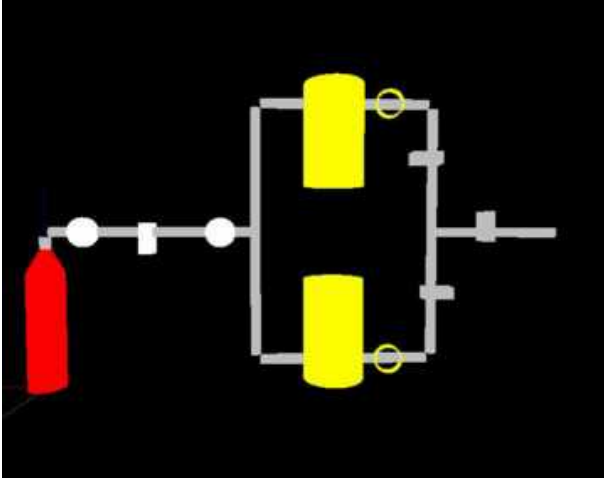


Figure 8. Geometric Model of the real scene (mockup of a helium flushing system)

6.3. Open issues regarding the automatic generation of dynamic world models from existing data sources

Quite a few geometric file formats exist that allow us to describe complex object shapes, including options for specifying various levels of detail as well as complex motion paths. We can expect scene models to be generally available in some such file format, ready for the AR subsystem to be read.

It is an open issue to determine how to combine such “legacy” file formats with additional information that we need to provide flexible rendering modalities which go beyond showing levels of detail. To this end, we need to be able to group geometric descriptions into units to which a common visualization technique should be applied, such as a change of color, or a change of polygon filling style. Ideally, such functionality should be applicable to existing “legacy” geometric descriptions without the need for major manual modifications. Furthermore, the set of modifiable parameters should be extensible such that new visualization styles can be included whenever they are deemed suitable for a particular augmentation context. Currently our rendering variations are implemented on the OpenGL level.

Another open issue involves the semantic association of a geometric scene model with an IETM database. Semantic references to objects, such as “valve V2000” have to be associated with geometric descriptions of physical entities. When reading a manual, workers are currently asked to make this association mentally – typically by

referring to a geometric drawing of the plant. Since many similar objects (pipes, push buttons, etc.) are often placed in close vicinity—some with labels, but most without—workers lose a lot of time correctly identifying which physical object they are expected to work on. This is an error-prone situation which augmented reality technology can potentially improve.

Providing (semi-) automatic tools to generate such mixed semantic-geometric work instructions in which every object name is associated with a geometric entity is an immense data analysis problem, requiring that physical manuals be parsed, understood and then cross-linked with geometrical drawings or 3D models of the respective plant. Curtis et al report of a similar problem when generating electronic wiring instructions for the wire bundle assembly task [3]. In our current prototype, we have manually established semantically correct associations between object names in the IETM database and in the world model. We discuss this issue further in section 7.1.1.

6.4. Tracking

Since tracking was not a central focus of this project, we have implemented only basic functionality so far. A color camera (currently installed on a tripod) records a scene that contains a set of black circles at known locations (see Figure 8). To initialize the tracker, the locations of the circles currently have to be identified interactively by an operator. The system then tracks the marks automatically using normalized cross correlation to track feature templates from image to image.

Once six or more circles have been identified, we compute the intrinsic and extrinsic camera parameters using Tsai’s calibration algorithm [13]. Further on, only the extrinsic camera parameters are recomputed from the tracked feature locations whereas the intrinsic parameters are assumed to remain constant. We do not yet account for occluded or disappearing features.

7. Results and lessons learned

During the winter semester 2000/2001, we designed the architecture described in previous section based on requirements from a real client and implemented a subset for the purpose of demonstration. Given our limited resources and that the focus on this case study was on mobile AR systems, a substantial amount of effort was spent on the AR and the Network subsystems. Other aspects of the systems were not implemented or only simulated (e.g., the speech recognition subsystem was replaced with a keyboard, the tracking camera was set on a tripod, the IETMs stored in the system were limited to the example depicted in Figure 1., the iButton access mechanism was not implemented since we only dealt with one room). However, the resulting prototype was sophisticated enough that we were able to give a full hour demonstration to the client using prerecorded video

sequences. In building this prototype, we were also able to investigate several architectural and technical issues relevant to the augmented reality framework cited in the introduction.

In this section, we summarize these issues we encountered and discuss their general implications. Here, we only focus on four issues that are relevant to the augmented reality and mobile aspects of the system:

- the linking of geometrical and information models,
- the definition and reuse of generic AR components,
- the multimodality of user interfaces and its implications on the information model, and
- the mobility and unreliability of the network service.

7.1.1. Linking geometrical and information models

STARS deals with a legacy situation: nuclear powerplant maintenance is currently supported by a paper-based system. Hence, STARS would require the digitization and validation of a voluminous documentation. While digitization of legacy documents is a difficult issue to the scale of the problem, it has been investigated and several off-the-shelf products have been developed to cope with this problem. However, an augmented reality system such as STARS imposes additional requirements on the information model.

To provide added value, STARS needs semantic links between IETMs and the 3D geometric representation of the corresponding device. In the example of Figure 1., there must be a link from the information model representing the work order step closing valve 2000 and the geometric element V2000. Hence, conversions and authoring systems for dealing with legacy paper-based information will have to enable the creation and validation of such links. While this issue may seem trivial, the scale of the task is such that alone it could prevent the deployment of augmented reality systems in a realistic context.

To deal with this issue, we propose that authoring systems could also become augmented reality systems. The author of the maintenance procedure, in this case, would simulate the execution of the maintenance procedure with a HMD and insert the semantic links between the IETMs and the 3D model by pointing at relevant devices (since IETMs and 3D models are usually available in powerplants, only the semantic links are missing).

In general, such augmented reality authoring systems (behaving the same way as the target system but enabling its user to modify the geometric and information models) will be useful in other domains as well for fine tuning and validation.

7.1.2. Generic augmented reality components

In section 6.3, we discussed several issues related to the distribution of responsibilities among the user interface, and the AR rendering component. To maximize reuse, the

AR rendering component must not embed nuclear powerplant maintenance knowledge while providing high levels of functionality. This results into the user interface component embedding all of the application specific knowledge while relying on the AR component for all AR related functionality.

A consequence of this division of labor is that the AR component must offer a set of communication primitives for manipulating and viewing the geometrical model selectively. Most current rendering components fall short of this as they assume the geometrical model is mostly static.

Instead, to effectively support this type of application, the AR rendering component must evolve similarly as today's 2D user interface libraries which by now have matured to provide reusable interfaces enabling the dynamic combination, resizing, creation, and destruction of a variety of 2D widgets. AR rendering components need to provide similar functionality for "AR widgets" which will be used to illustrate maintenance instructions embedded in real settings.

7.1.3. Multi-modal interface and information models

Mobile AR systems will be bound to include multi-modal user interfaces to enable users to select the mode of interaction that best suits their work (e.g., speech, LCD plate, AR, etc.) [11]. In this context, AR can be seen as one mode of interaction between the user and the wearable.

Different modes of interactions require different structuring of the information. For example, when steps are displayed on the HMD, they are shown one at a time, a brief description is printed at the bottom of the screen, and the devices that are relevant to the step are highlighted as 3D augmentations. Conversely, when the maintenance procedure is displayed on the LCD plate, the complete procedure can be described in one screen, the current step is highlighted, and links to other information such as demonstration videos are offered. This layering of information by mode of interaction requires the information model to be correspondingly tagged and structured for this purpose, in a similar way that VRML models can be structured in terms of levels of details [15]. This issue, similar to the linking of geometrical information in section 7.1.1, is bound to be time consuming and difficult to resolve in scale.

7.1.4. Mobile and unreliable network services

In section 5, we designed and implemented a prototype application transparent network subsystem. The application transparency has the advantage that both application and network subsystem can be implemented and tested in parallel. However, a drawback of this approach is that there is no possibility for the application to influence the behavior of the network subsystem. For some aspects, this would have been advantageous. A user

could for example trigger the prefetching of needed documents in advance if he or she was in front of a 'data station'. This would have been especially advantageous in the learning phase of the prefetching subsystem, when the prefetching statistic is not yet filled with sufficient information to predict document requests.

In practice, we saw remarkable performance gains when running the application with the implemented network subsystem. Large data items, as for example videos and graphics, were accessible nearly immediately by the users when prefetched in advance. On the other side, we saw that prefetching is a bandwidth consuming technique. This is especially true if a group of users are using a shared medium for client-server communication, for example, Wave LAN. Therefore, we are investigating on two further steps. First, we want to integrate more context information in our prefetching system in order to improve the accuracy of our prefetching statistic. Second, besides client-to-server communication, we want to support client-to-client data communication to save shared medium bandwidth in cases where users are located near to each other, access the same documents and are equipped with client devices that can communicate with each other.

8. Summary

In this paper, we have reported on our recent efforts to design an AR system, STARS, for a powerplant maintenance scenario. The work was carried out as a three-month project with senior students of computer science. Although not all aspects of the system could be implemented within the short time frame, several general issues regarding the processes to generate, access, and transmit information in "AR-ready" formats in a wide-scale, maintenance-oriented scenario were identified and discussed. We thus expect this project to have laid part of the foundations for further research activities in providing mobile, immersive access to large quantities of work instructions.

9. Acknowledgments

The paper is based on the dedicated project work of all students participating in the STARS-3 and AR Praktika both at the TU Munich and at Carnegie Mellon University. We would like to thank all students, coaches, and the people providing the system administration support. Furthermore, we would like to thank Mr. Erwin Rusitschka from Framatome ANP and Mr. Dick Martin from Inmedius for discussing the overall application scenario with us.

10. References

[1] M. Bauer, B. Brügge, G. Klinker, A. MacWilliams, T. Reicher, S. Reiß, C. Sandor, & M. Wagner. Design of a Component-Based Augmented Reality Framework. ISAR '01.

- [2] B. Brügge, A.H. Dutoit, R. Kobylinski, & G. Teubner. Transatlantic Project Courses in a University Environment. *7th Asia-Pacific Software Engineering Conference*. Singapore. Dec. 2000.
- [3] D. Curtis, D. Mizell, P. Gruenbaum, and A. Janin. Several Devils in the Details: Making an AR App Work in the Airplane Factory. In R. Behringer, G. Klinker, & D.W. Mizell, eds. *Augmented Reality—Placing Artificial Objects in a Real Scene*, A.K. Peters, 1998, pp. 47-60.
- [4] D. Curtis, C. Esposito, P. Gruenbaum, A. Janin, and D. Mizell. The Boeing Portable Maintenance Aide. *Demonstration at the 1st International Symposium on Augmented Reality (ISAR'00)*, Munich, Oct. 2000.
- [5] R. Fielding, et al., "Hypertext Transfer Protocol -- HTTP/1.1", *RFC 2616*, June 1999.
- [6] T. Höllerer, S. Feiner, and J. Pavlik. Situated Documentaries: Embedding Multimedia Presentations in the Real World. *Proc. 3rd International Symposium on Wearable Computers (ISWC'99)*, San Francisco, Oct. 1999, pp. 79-86.
- [7] H. Kato, M. Billinghurst, I. Puoppyrev, K. Imamoto and K. Tachibana. Virtual Object Manipulation on a Table-Top AR Environment. *Proc. 1st International Symposium on Augmented Reality (ISAR'00)*, Munich, Oct. 2000, pp. 111-119.
- [8] G. Klinker, D. Stricker, and D. Reinert. Augmented Reality: A Balance Act between High Quality and Real-Time Constraints. In Y. Ohta & H. Tamura, eds. *Mixed Reality—Merging Real and Virtual Worlds*, Ohmsha & Springer Verlag, 1999, 325-346.
- [9] J. Molineros, V. Raghavan, & R. Sharma. AREAS: Augmented Reality for Evaluating Assembly Sequences. In R. Behringer, G. Klinker, & D.W. Mizell, eds. *Augmented Reality—Placing Artificial Objects in a Real Scene*, A.K. Peters, 1998.
- [10] V.N. Padmanabhan, J.C. Mogul. Using Predictive Prefetching to Improve World Wide Web Latency. *ACM SIGCOMM Computer Communication Review*, Vol. 26, No. 3, July 1996.
- [11] C. Sandor & T. Reicher. CUIML: A Language for the Generation of Multimodal Human-Computer Interfaces. *Proc. European User Interface Markup Language Conference (UIML)*. Paris, France, Mar. 2001.
- [12] H. Tamura. What happens at the Border Between Real and Virtual Worlds – The MR Project and Other Research Activities in Japan. *Proc. 1st International Symposium on Augmented Reality (ISAR'00)*, Munich, Oct. 2000, pp. xii–xv.
- [13] R.Y. Tsai. An Efficient and Accurate Camera Calibration Technique for 3D Machine Vision. *Proc. CVPR*, pp. 364-374, 1986. See also <http://www.cs.cmu.edu/rgw/TsaiCode.html>
- [14] J. Underkoffler, B. Ullmer, & H. Ishii. Emancipated Pixels: Real-World Graphics In The Luminous Room. *Proc. Siggraph'99*, Los Angeles, Aug. 8-13, 1999, pp. 385-392.
- [15] The VRML 2.0. Specification. International Standards Organization (ISO) ISO/IEC 14772. See also: VRML repository at <http://www.edsc.edu/vrml/>
- [16] MIL-M-87268: Interactive Electronic Technical Manuals: Content, Style, Format, and User-Interaction Requirements. Department of Defense. United States Gov. <http://www.ietm.net/>