Pers Ubiquit Comput (2005) 9: 169–185
DOI 10.1007/s00779-004-0328-1

ORIGINAL ARTICLE

Christian Sandor · Gudrun Klinker

A rapid prototyping software infrastructure for user interfaces in ubiquitous augmented reality

Received: 1 June 2004 / Accepted: 22 October 2004 / Published online: 11 January 2005

Abstract Recent user interface concepts, such as multimedia, multimodal, wearable, ubiquitous, tangible, or augmented-reality-based (AR) interfaces, each cover different approaches that are all needed to support complex human–computer interaction. Increasingly, an overarching approach towards building what we call ubiquitous augmented reality (UAR) user interfaces that include all of the just mentioned concepts will be required. To this end, we present a user interface architecture that can form a sound basis for combining several of these concepts into complex systems. We explain in this paper the fundamentals of DWARF's user interface framework (DWARF standing for distributed wearable augmented reality framework) and an implementation of this architecture. Finally, we present several examples that show how the framework can form the basis of prototypical applications.

Keywords Augmented reality · Ubiquitous computing · Tangible user interfaces · Multimodality · Software architectures · Frameworks · Mobile systems

1 Introduction

One of the major challenges of current computer systems is to provide users with suitable means to plan, model, and control complex operations that are composed of many inherently interdependent processes. For example, control rooms of industrial plants, surgery preparation rooms, cockpits of airplanes, and consoles of modern cars are typically equipped with many different physical or electronic input and output devices. Recent user interface concepts, such as multimedia, multimodal, wearable, ubiquitous, tangible, or augmented-reality-based (AR) interfaces, each cover different approaches. We believe that all of these approaches are necessary to support increasingly complex human–computer interaction. Increasingly, an overarching approach towards building ubiquitous augmented reality (UAR) user interfaces that include all of the just mentioned concepts might be required. An instance of a UAR user interface can be seen as an aggregation of these conceptual aspects of interaction design. But which aspects should be used for a task at hand? To allow interaction designers to quickly change the aspects that are used, we implemented a software infrastructure that allows the rapid exchange of interaction styles.

1.1 Current user interface paradigms

Current user interface research addresses several different issues of human–computer interaction: multichannel communication between users and computers, user mobility, and the three-dimensional combination of virtual and real worlds. Each of these issues describes a dimension in a design space of future human–computer interfaces.

1.1.1 Multichannel communication

To provide human–computer interaction beyond traditional WIMP-based (windows, icons, menus, and pointing devices) user interfaces [45], various communications channels are being explored that correspond more naturally to the human visual, auditory, and tactile senses, both for gathering user input (speech, gestures, special three-dimensional input devices) and for providing output (sound, graphics, haptics) to the user. According to Nigay et al. [29], multimodality is the capacity of a system to communicate with a user along different types of communication channels and to

C. Sandor (✉) · G. Klinker
Institut für Informatik, Technische Universität München, Munich, Germany
E-mail: sandor@in.tum.de
E-mail: klinker@in.tum.de

extract and convey meaning automatically. Both multimedia-based and multimodal systems use multiple communication channels. Research on multimedia-based user interfaces focuses on handling the vast amount of data that is required to gather raw input streams and to generate raw output streams in real time. Multimodal systems, on the other hand, are defined at a higher level of abstraction. They strive towards associating semantic meaning with media streams. They are expected to help users control systems more easily by combining several interaction modalities into more powerful interaction paradigms than any single modality would be able to provide on its own [35, 54]. There are four different types of multimodality (exclusive, alternate, concurrent, and synergistic), depending on the combined or independent fusion of several interaction channels, as well as on the sequential or parallel use of multiple modalities [29]. Although multimedia-based and multimodal systems have much in common, they cannot be described as one being a subset of the other. Many of today's Internet browsers and email systems provide multimedia-based functionality without being multimodal. On the other hand, multimodal systems focus more on the synergistic high-level interpretation of a few combined and parallel input tokens.

1.1.2 Mobility

Current trends towards mobile systems enable users to communicate with their computers while they are far away from their desks. Such mobility requires lightweight and untethered solutions that allow users to roam freely in a wide area. There are two approaches that address these requirements. Wearable user interfaces [44] strive towards providing users with lightweight, portable, or wearable computer equipment that can become part of their daily attire. Wearable functionality can be provided on palm-based computers or mobile phones. It can be attached to a belt or woven into users' clothes. Using a personal area network (PAN, http://grouper.ieee.org/groups/802/15/), computing power can be connected to wearable gadgets, such as headsets, microphones, head-mounted displays (HMDs), 3D mice, and portable keyboards. Ubiquitous [55, 26], ambient [10], and pervasive [14] interfaces to computers have been proposed by Weiser and others with the goal of providing computing power to people in such a pervasive manner that the computer in itself becomes a secondary (virtually invisible) issue. Large-scale environments, such as buildings, are equipped with networked computers and multichannel user interfaces such that users are always surrounded by them. Research in this field focuses on developing proper system layouts for such large-scale computer networks, requiring high data bandwidths and system adaptivity to changing user demands. Ad hoc interoperability of services is needed [23] in order to build context-aware smart spaces into which wearable, smart appliances can be integrated to provide users with personalized and context-adapted information. Wearable and ubiquitous computing are two different approaches along a continuum of options in the mobility spectrum. They are not mutually exclusive, but they tend to favor different trade-offs between the low-power, individualized computer use on a wearable computer and high-performance computing in a server-like, community-based stationary environment. Current trends begin combining both approaches.

1.1.3 Interaction embedded within the real world

User mobility provides the unique opportunity to let users communicate with their computer system while remaining involved in their three-dimensional, real-world environment. This provides the chance for users to communicate with computers via interaction with position-tracked real physical objects. To this end, two complimentary approaches have emerged: tangible user interfaces (TUIs) and augmented reality (AR). TUIs build on the observation that century-old, very well designed physical tools exist, e.g., in craftsmanships, that have been fine-tuned for years towards very specific usage. Based on humans' spatial and motor skills, each such tool is directly suited towards fulfilling specific tasks. The purpose of each tool is immediately obvious to a trained craftsman. It represents a unique combination of input and output behavior that is directly suited to the task it has been designed for. The TUI community strives towards providing similarly powerful TUIs for computers and their interaction with virtual information. Ishii's work with the MIT Tangible Media Group has produced a large number of creative TUIs, e.g., [50]. A formal model of TUIs is provided in [51]. AR focuses on presenting information in three dimensions with respect to the user's current position. Users can, thus, see, explore, and manipulate virtual information as part of their real-world environment. In his classical definition of AR, Azuma [2] states three requirements: real-time performance, user registration in three dimensions, and a combined presentation of both virtual and real information. In Milgram's taxonomy [28], AR is seen as a subset of mixed reality. TUIs and AR overlap considerably with respect to the manipulation of real objects. Yet, for AR, this may be a pure consequence of augmentations issuing instructions provided by an AR application, e.g., for machine maintenance or repair, while TUIs consider such tracked object manipulation as the main means for users to control the system. On the other hand, such TUI-based interaction does not necessarily have to result in geometrical (visual or aural) augmentations of the three-dimensional world. Illustrations may be provided in a lower-dimensional space, such as on a two-dimensional table top or on a wall. AR, on the other hand, focuses on analyzing and presenting information in three dimensions.

1.2 Convergence of paradigms

Many of the above-mentioned research directions are currently broadening their spectra to include important aspects of other concepts, thereby generating a confluence of the individual fields. Examples of those broadened approaches are tangible augmented reality [20] and multimodal augmented reality [19]. We refer to this emergent area of user interfaces as ubiquitous augmented reality (UAR). The idea to focus on the overall user experience by selecting the right interactions and visualizations has already been proposed by Buxton [8]. He calls this approach holistic, referring to the idea of the whole being more than the sum of the parts. Obviously, this overarching, holistic approach introduces new challenges to both the interaction designers and the supporting software infrastructure. Interaction designers have to choose from a broader variety of possible interaction styles to use. On the other hand, a supporting software infrastructure has to be capable of dealing with a wide range of interaction styles. Also, it has to enable the interaction designer to quickly try out new interaction styles. Additionally, a good software infrastructure should be extensible so that new interaction styles can be added later on. Apart from the interaction style mentioned in the last section, several other styles are currently being incorporated into our infrastructure: zoomable user interfaces [36], attentive user interfaces [30, 53], and perceptive user interfaces [49].

1.3 A supporting software infrastructure

To progress in this direction, we have built a framework for UAR user interfaces, which is based on a tangible tool metaphor and allows the multimodal construction of higher-level interaction metaphors. The framework offers a tool chest containing a set of tools, each providing a specific functionality to the user. By composing more complex tools out of the simple basic toolset. higher-level functionality can be achieved. This allows users to manage any complex, inter-related processes, by using a number of physical objects in their surroundings. The framework can be used for single-user as well as multi-user applications. The system state is presented as a three-dimensional augmentation that is embedded within the real environment of a user. Scene views are provided via several both personal and ubiquitously available displays, accounting for different options in user mobility and privacy. Some views are common to all users (e.g., in the form of projections on a table or wall), whereas others are restricted to subgroups of users (shown on portable display devices) or to a single user (shown on an HMD). The displayed content depends on the current position of a display in the scene, representing its current viewpoint. Users are provided with several tangible objects which they can manipulate together in a group or individually to influence the system state. Some of these objects are owned by specific users, and the interaction style with these objects can also be tailored to the preferences of individual users.

1.4 Organization of the paper

The main design goal of our user interface framework is rapid prototyping and the collection and reuse of different interaction elements. The following sections explain how this is achieved. Section 2 provides an overview of our approach, introducing the technical requirements for UAR user interfaces, the DWARF framework, and our user interface architecture that lays the foundations for our work. Section 3 presents four prototypical examples showing increasingly complex combinations of user interaction concepts. Section 4 provides a summary and discusses future directions.

2 Our approach

In the last section, the three main characteristics for UAR user interfaces were presented: mobility, multichannel communication, and interactions that are embedded in the real world. Based on these characteristics, we now discuss the implications for a supporting software framework that addresses UAR user interfaces. First, we discuss the technical requirements in more depth and build an analysis model on the software level to have a reference frame for further explanations. Then, we point out how our DWARF framework (distributed wearable augmented reality framework, http://www1.in.tum.de/DWARF/) generically addressed UAR user interfaces. Next, more details on the specific user interface concepts within DWARF are presented. At the end of this section, we briefly discuss the benefits and limitations of our approach. Probably the most similar approach to ours is the iRoom project from Stanford University [18]. Their work shares some common concerns. They have built a highly dynamic software infrastructure for mobile and distributed user interfaces based on tuplespaces [17]. However, they do not address tangible interactions and AR interfaces. Instead, they focus on more conventional input, like pen-based interactions, whereas the output they are mainly concerned with is wall-sized displays. As a result, iRoom does not couple real-world objects to visualizations and, thus, does not provide a data flow framework for continuous integration, as introduced in the next section.

2.1 Technical requirements for ubiquitous augmented reality user interfaces

It is common practice in software engineering to build an analysis model of a problem domain. After the main requirements are established and the analysis model has been formulated, this model serves as a reference frame

for discussions. The generic functional decomposition that is presented here is the result of our analysis of the problem domain:

- Mobility: The main requirement for a mobile system running in ubiquitous environments is flexibility. Resources in the environment have to be connected dynamically to resources a user might carry around, e.g., palmtop computers or wearables like MIThril (http://www.media.mit.edu/wearables/mithril/). This implies a highly modular architecture, whose components should be dynamically reconnectable.
- Multichannel communication: To address multimodality, a system has to be able to deal with several input channels. The user intention has to be extracted from the input that is received over these channels. Complementarily, a multimedia-based system has to have a coordination instance that distributes the content to be presented to the user, leveraging the available output channels.
- Interaction: Embedded within the real world for AR user interfaces and TUIs, a proper three-dimensional registration between real and virtual objects has to be established. To this end, mobile real-world objects have to be tracked, i.e., their position has to be determined continuously. A typical example for AR involves tracking a user's head and using its pose to set the viewpoint of a virtual scene, as seen in an HMD. Similarly, TUIs often couple real-world objects to virtual representations of objects.

Based on these requirements, we propose a generic functional decomposition of UAR user interfaces. A large number of AR frameworks have recently been analyzed (see [40]). As most of these frameworks also support multichannel communication and mobile systems, the findings made in that analysis paper helps establish a foundation for an analysis model covering UAR user interfaces. We have consolidated the recurring patterns into this generic functional decomposition. Fig. 1 (inspired by [27]) shows the relevant subsystems and components within them. It is important to note that the subsystems (input devices, media analysis, interaction management, media design, and output devices) are general purpose and generic; however, the components within them are just examples. Similarly, DWARF is one possible implementation of a framework enabling UAR user interfaces. Other implementations adhering to the functional decomposition in Fig. 1 would be possible. The input devices subsystem contains input devices that are used to receive commands from the user. Each of these devices offers an individual input modality to be evaluated by the multimodal user interface. The output devices subsystem renders the signal on the specified output devices. For multimedia-based systems, several output devices are used at the same time. Media analysis is the process of turning physical user input into abstract tokens [34] handed over to the subsequent parts of the system—this can be compared to the task performed by the lexical analysis of a compiler. Separate classes, such as gesture analysis, speech analysis, and tangible input analysis, deal with the specific properties of different input modalities of the input devices. The software components that present content to the user over any of the cognitive channels, e.g,. visual and aural, are contained within the media design subsystem. The interaction management subsystem determines which output is presented to the user. Current challenges for interaction management are performance, flexibility, adaptivity, usability, and efficient error management. The media fusion component takes the tokens of several input channels and infers user intention from them. In this component, two different ways for combining different input channels under their respective

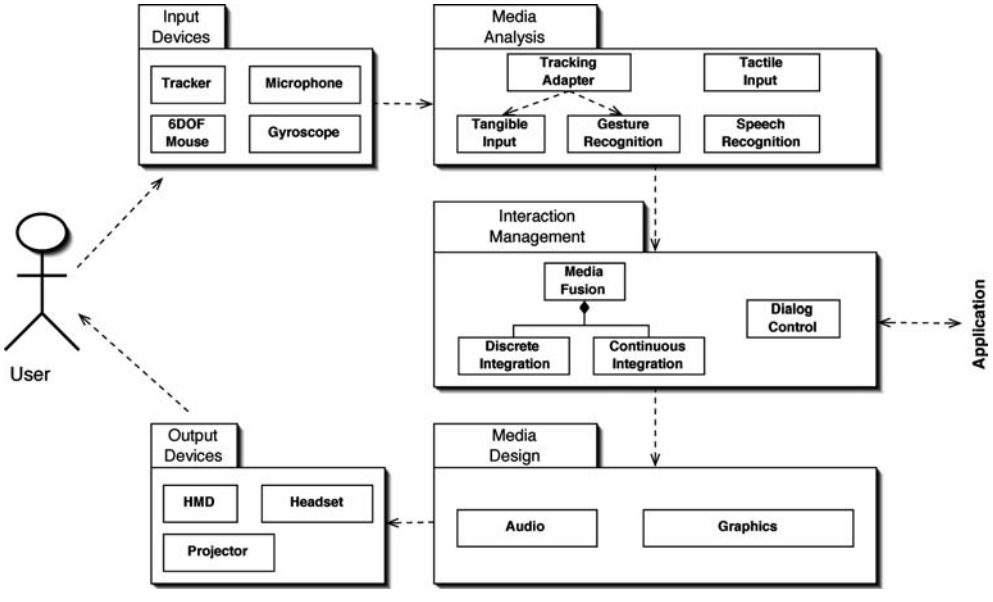


Fig. 1 A generic functional decomposition of UAR user interfaces

boundary conditions are considered. Continuous integration combines tokens that can take real values in a certain range, e.g., rotations can take an infinite number of different values between 0° and 360°. Example input devices that deliver these kinds of tokens are mice, trackers, and gyroscopes. A variety of frameworks [32, 39, 41] already exist that ease the task of building data flow networks for continuous integration. However, they do not take into account the distribution over several hosts and the continuous dynamic reconfiguration that is necessary for UAR. Discrete integration refers to the integration of devices such as speech recognition, which deliver only discrete values, like the word that was recognized. The most similar approach to our model for discrete integration is [15]; however, in contrast to our framework, they focus on static setups. Finally, the dialog control component selects the presentation medium and what to present through it.

2.2 The distributed wearable augmented reality framework (DWARF)

For the past few years, we have been building a general, reusable, and easily (ad hoc) configurable distributed wearable augmented reality framework called DWARF [3, 4]. DWARF describes different contributing system parts as separate components that are able to connect with one another across a dynamically configurable peer-to-peer network of distributed processes. Whenever new components are within reach, they are connected automatically into the network of communicating components. The connectivity structure of components is not fixed at start-up time. In fact, it can be changed arbitrarily at run-time. DWARF is suitable for building highly dynamic, flexible system arrangements within which mobile users, who carry mobile sensors and devices, can be connected on demand to stationarily available, ubiquitous resources that are provided within intelligent environments of the future. DWARF makes it possible to develop mobile applications because of the following features:

- Flexible architecture: Because of the fine granularity of components and the loose coupling between them, DWARF systems are highly flexible.
- Fast: Several communication protocols are implemented for the communication between components. Some of them are especially well suited for real-time applications, e.g., shared memory and CORBA events.
- Distributed: The components that form a DWARF system can be a combination of local and remote devices. The distribution is completely transparent to the components.
- Adaptivity: With the inherent options for ad hoc connections and the reconfiguration of components, DWARF systems are also inherently adaptive.
- Operating system independent: To allow deployment among a variety of devices, DWARF has been designed to be independent of a specific operating system. We have successfully implemented DWARF systems on Linux, Window, and Mac OS X platforms.
- Programming language independent: Similarly, DWARF supports three programming languages so far: Java, C++, and Python.

We have already built around ten systems based on DWARF. For a thorough list, please refer to our projects Web page at http://www1.in.tum.de/DWARF/ProjectsOverview). Based on DWARF's flexible nature, we have developed a user interface architecture with a supporting set of components that are described in the next section.

2.3 The user interface architecture in DWARF

DWARF was designed as a research platform combining wearable systems with ubiquitous environments. Its component model and architecture can be used in several different research areas. In this section, we explain several architectural principles and components that make up the user interface framework within DWARF. An overview of the architecture can be seen in Fig. 2. An important distinction for communication channels is the frequency with which messages are passed. Discrete events are typically sent every few seconds, whereas continuous events, such as tracking data, are sent at very high frequencies.

2.3.1 Layering and device abstraction

We arrange the user interface components into three layers according to Fig. 2. Most data flows linearly from the media analysis layer, which contains input components to the interaction management layer, where the tokens are interpreted. From there, the data flow continues to the media design layer, where the output components reside. We have developed a standardized format for tokens that are sent from the input components to the interaction management layer. Input tokens can be decomposed into four different types: analog values that can be either within a limited range (e.g., rotations) or an unlimited range (e.g., translations), and discrete values that can be either Boolean (e.g., pressing a button) or text strings (e.g., the output of a speech recognition process). Due to this standardized format, we can exchange one input device for another—as long as they emit the same type of tokens. For example, a speech recognition component listening for a set of words could be interchanged transparently with tangible buttons with the same set of labels. Similarly, the interaction management layer sends commands to the media design layer. This setup corresponds to the command pattern described by Gamma et al. [13]. The commands consist of actions that have to be executed by

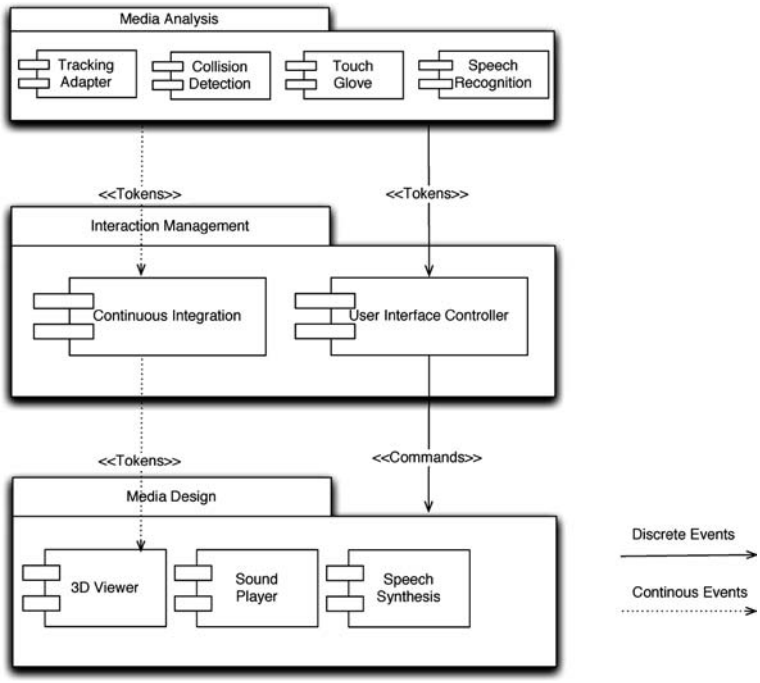


Fig. 2 Functional decomposition of DWARF-specific user interface components

the output components (e.g., by presenting the question "yes or no?" to the user). One can interchange output components in the same way as with input components. Due to this flexible DWARF component model, the exchange of I/O components works, even at system run-time.

2.3.2 Efficient coupling of real and virtual worlds

One of the recurring input modalities used by all tangible objects is their location within the scene. For example, mobile displays present scene views depending on their current position. To ensure the most efficient transfer of pose data from trackers to three-dimensional viewers, our system allows us to establish direct connections between tracking and viewing components whenever no filter has to be involved. For more complex visualizations, e.g., non-linear input processing [37, 46], a pipe-filter component tree preprocesses the pose data emitted by the trackers before feeding it to the viewer. This approach has already been implemented by several other frameworks. In our approach, we add to this approach the concept that the filter components can be arbitrarily distributed and interchanged at run-time, once again using the flexible DWARF component model, allowing us to quickly experiment with different arrangements during system run-time. This feature turns out to be very beneficial to user interface development whenever we need to dynamically experiment with different options to interpret the tracking data [25]. In contrast, if the setup is known in advance, quite a few systems exist that are able to process and forward multisensory input data to a display system. An example is the OpenTracker system of the Technical University of Vienna [41]. In a joint collaboration [5], we have shown that the OpenTracker system can be easily integrated into DWARF by plugging the associated transformation hierarchy of the OpenTracker framework into the DWARF system. Similar approaches can be pursued for other systems.

2.3.3 Central coordination with Petri nets

Inside the interaction management layer, we decided to move all functionality into the DWARF user interface controller (UIC) component, thereby combining the functionalities of dialog control and discrete integration. It combines input tokens sent by the media analysis components and then triggers actions that are dispatched to components in the media design package. Note that the UIC must have an internal model of the state of the user interface, otherwise, context-sensitive commands could not be interpreted correctly. The internals of the UIC are explained using the example of the SHEEP game in Sect. 3.3. An interesting point here is that we chose Petri nets [16] to specify the behavior of a UIC instance because the specification of multimodal interactions can be mapped very conveniently to Petri nets. The UIC is based on the Petri net framework Jfern (http://sourceforge.net/projects/jfern), which provides large parts of the functionality needed for this component: the Petri net that models the multimodal interactions for a user are written in XML. From these descriptions, Java classes are generated. Jfern also allows the graphical display of the Petri net and its current state. This is very useful during program development and also for demonstrations because people can always see immediately the current state of the user interface.

User input is modeled as tokens (not to be confused with the tokens sent by the media analysis components) that are placed into the Petri net. The rule-based evaluation of the user input is encapsulated into guards that check whether a transition is legal. Whenever a transition is triggered, events are sent to the media design components, which then adds, removes, or changes the properties of parts of the user interface. Figure 3 shows the flow of events within a Petri net (bottom row in Fig. 3 is from the SHEEP game (see Sect. 3.3)) and the according inputs to and outputs from the system (top row). The UIC receives two tokens from the media analysis package: one that represents the collision of wand and table and one for the ''insert'' speech command. These tokens are placed onto places in the Petri net. After all places on incoming arcs of a transition are full, the transition is triggered, which results in a creation of a new sheep.

2.3.4 Interactions and Petri nets: one-to-one mapping

To conform to our lightweight and distributed approach, we model each interaction in its own Petri net. For example, all the interactions found in Sect. 3 are each realized as individual Petri nets. When thinking about tangible interactions, this couples the functionality of each tangible object to a specific instance of a UIC. As a result, the visualization of the Petri net's state at run-time tells us the state of the coupled tangible interaction. Additionally, this approach fits very well with the tool metaphor for interactions [4].

Fig. 3 Realization of a multimodal point-and-speak user interface with a Petri net within the SHEEP game

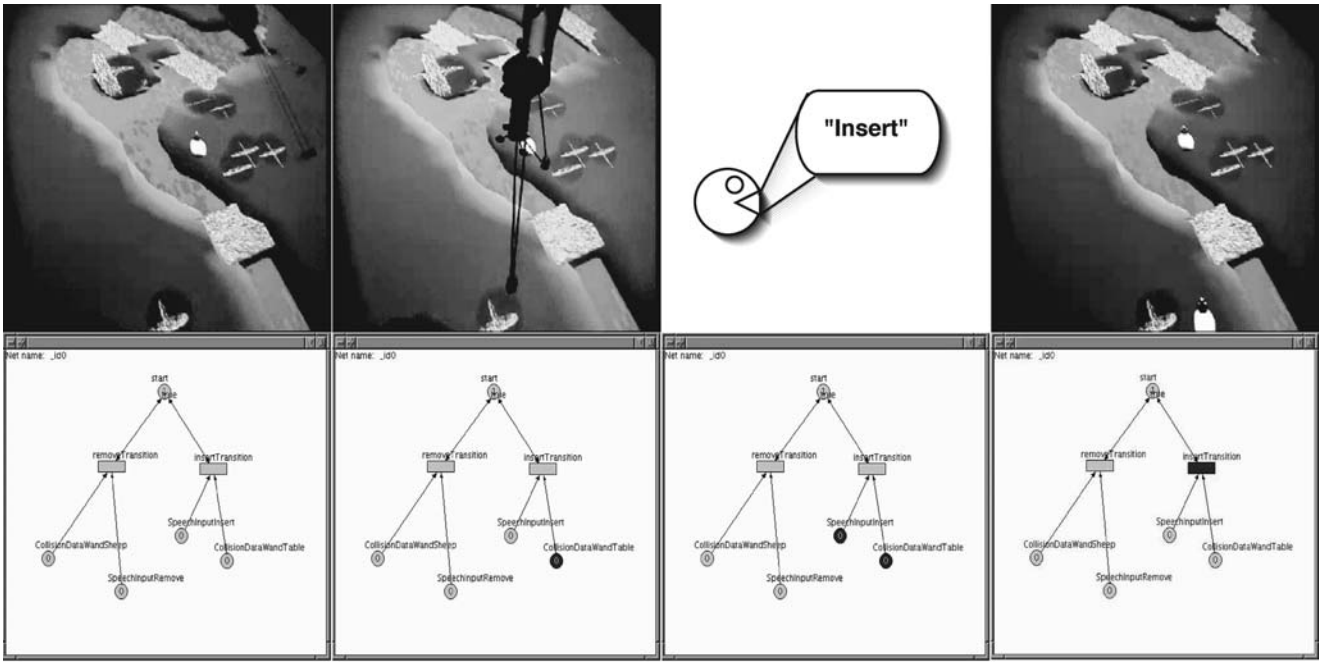
2.3.5 Lightweight and stateless I/O components

To address the flexibility requirement of user interfaces, we chose to keep as much state information as possible in the interaction management layer (to be explained in the next section). As a consequence, the I/O components were designed to keep as little state as possible. This allows us to add and remove I/O components conveniently at system run-time. However, this is not possible for some components. Currently, we are working on a persistence layer to be able to pass states between components.

2.3.6 Set of reusable I/O components

The available set of I/O components increases continuously. It is important to notice that reusing these components includes no programming of code because they are generic and are meant to be reused among different applications. To tailor components to a specific application, the components are configured via an XML file. Here is a short list of the most important I/O components:

- Speech recognition: For commands that address the control of the system, usability studies have shown that a command language is preferable to tactile input [33]. We provide a speech recognition component in the framework. Especially in applications that require hands-free working (e.g., maintenance), this type of interaction gains importance. Internally, speech recognition is a word spotter. It is configured via a context-free grammar.
- TouchGlove: The TouchGlove is a special-purpose input device developed at Columbia University [6]. It is explained in some more detail in Sect. 3.4. Input



tokens that are emitted by this device can be fed into a DWARF user interface. Interestingly, this device can emit both continuous and discrete input data. It can, thus, be connected to both the UIC and directly to the viewer.
- Collision detection: The pose data emitted by the tracking components is used by the collision detection component to detect collisions between objects. This includes collisions of real objects with virtual objects, real objects with real objects, and virtual objects with virtual objects. The first two types of collision must be considered to capture user input that is provided via the movement of tangible objects. A common practice in TUIs is to couple real-world objects with virtual objects. This explains the need for the last type of collision, that is, virtual objects with virtual objects. Consider a virtual slider that moves according to the tracked hand of the user. Whenever the slider is at its maximum value, a collision between the slider and the frame in which it is displayed is triggered.
- Sound player: The sound player component is a simple Java application configured via an XML file. It contains the mapping between incoming commands and sound files that are to be played.
- Three-dimensional viewer: The three-dimensional viewer component displays specific views of the virtual world according to the current location and function of the tangible object. An important design goal is the ability to update the virtual parts of a three-dimensional scene in real time. This component turned out to be quite difficult to design and implement.

Our current version of the framework accepts all important commands that are necessary for the display of dynamic three-dimensional scenes in real time. Additionally, several viewing modes are supported: video background for video see-through displays or visualization of AR scenes (e.g., Fig. 13b is a screenshot taken from the viewer), or support for a variety of stereo modes for different stereoscopic displays. Furthermore, the display of head-fixed content (see [12]) is possible. The underlying technology is an open source implementation of Open Inventor [47]: Coin3D (http://www.coin3d.org).

2.4 Summary

A high-level description of the DWARF user interface framework is that it is a hybrid of the relatively old idea of a user interface management system (UIMS) [31] and a component-based toolkit for data flow networks. It is an UIMS approach because of the application of a formal model (Petri nets in the UIC component), clear layering, and well defined tokens. It also has a toolkit character regarding the flexibly connectable filters that form data flow networks. Graphical widgets that are encapsulated in Open Inventor nodes can easily be reused and, thereby, are another feature of a lightweight toolkit. The combination of these two aspects fosters the advantages of both approaches. UIMSs have nice properties regarding reusability and rapid prototyping. However, they did not catch on [9] because of their limits regarding execution speed, difficulties to extend them for new interactions, and their tendency to force programmers to use a certain specification model. The speed of interpretation of a formal model (in our case, Petri nets) is hardly an issue any more these days. The last two concerns we hope to overcome by using a lightweight, component-based approach. Actually, for rapid prototyping, it is possible to run a DWARF user interface entirely without using Petri nets and replacing them with simple Python components. After fine-tuning the parameters in the Python components, the logic can be easily ported back into the Petri net model. Our user interface approach has successfully been used in various systems (http://www1.in.tum.de/DWARF/Projects-Overview). During our experiments, we have observed that most user interface developers felt quite comfortable using Petri nets to model their interactions. This is the first step towards developing a more general user interface based upon our approach. Once the functionality of the interaction has been specified, developers can experiment with different I/O devices to find the best setup [25]. The DWARF framework bears a lot of potential towards collaboration with other AR-based research activities and generating mutually integrated approaches. As a first step, we have worked towards integrating the Studierstube system of the Technical University of Vienna with DWARF [5], thereby extending our space of available DWARF components. The results are very encouraging. However, open questions remain. We have been able to achieve full flexibility for exchanging input devices at run-time. For output devices, however, we have only been partially successful. The underlying problem has been rather complex. It turned out that it is very difficult to define the semantic expressiveness of an output component, i.e., which auditory interfaces can be mapped to GUIs and which cannot? For input components, the definition of expressivenes was relatively simple because the receiver of emitted tokens is a computer. For output components, the receiver of content is a human. Perception and cognitive processing of information within the human mind are not understood well enough to come up with an ontology of output devices yet.

3 Example systems

In this section, we present several systems that we have built so far. An overview of the types of user interfaces for these applications is given in Fig. 4. It displays various exemplary demonstration systems according to the classification of interface dimensions (multichannel systems, mobility, integration into a three-dimensional environment) that was suggested in the introduction.

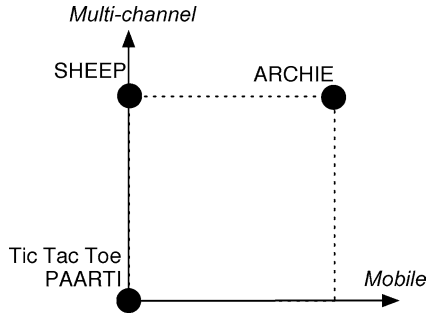


Fig. 4 Overview of the types of user interfaces realized in the example systems

Since all example systems are AR and TUI systems, the dimension interactions embedded within the three-dimensional real world has been omitted. The abilities of the systems are shown on the two axes, multichannel and mobile.

We present the example systems in an order of increasing complexity and maturity. We start with the Tic-Tac-Toe system in Sect. 3.1, which provides a tangible interface for a board game and accepts simple user gestures. Then, we explain the practical application of augmented reality in technical integration (PAARTI) system (Sect. 3.2). This is a very good example of extending a real worker's welding tool into a tangible user interface. Adding multichannel presentation schemes, we proceed to SHEEP (Sect. 3.3) and finally discuss the partly mobile augmented reality collaborative home improvement (ARCHIE) system in Sect. 3.4.

3.1 Tic-Tac-Toe

3.1.1 System overview

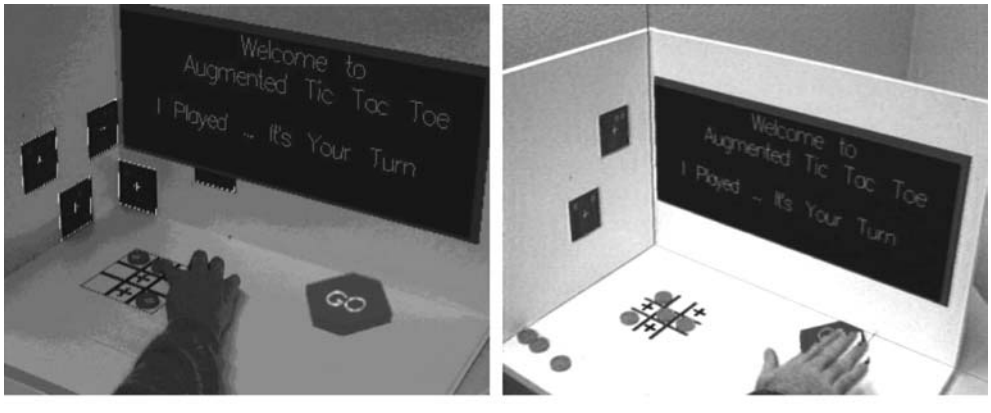
An early example from our work is the Tic-Tac-Toe game [22, 48]. It was developed while the second author of this paper was at the Fraunhofer Institute for Computer Graphics, Germany, prior to the start of the AR research group at the Technical University of Munich, Germany. It set some of the conceptual bases for the later development of distributed tracking concepts [21], the DWARF framework, and our approach to tangible and UAR user interaction. The user sits in front of a Tic-Tac-Toe board with some chips. A camera on a tripod behind her records the scene, allowing the AR system to track user motions while also maintaining an understanding of the current state of the game. The user and the computer alternate placing real and virtual chips on the board. The augmented scene is shown to the user on a nearby display.

3.1.2 Interactions

The user places a real chip onto the board (see Fig. 5a). When she has finished a move, she waves her hand past a three-dimensional "Go" button (Fig. 5b) to inform the computer that she is done. It is important for the computer to wait for such an explicit "Go" command rather than taking its turn as soon as the user has moved a chip. After all, the user might not have finalized her decision. When told to continue, the computer scans the image area containing the board. If it finds a new chip, it plans its own move. It places a virtual cross on the board and writes a comment on the virtual message panel behind the game. If it could not find a new chip or if it found more than one, it asks the user to correct her placement of chips.

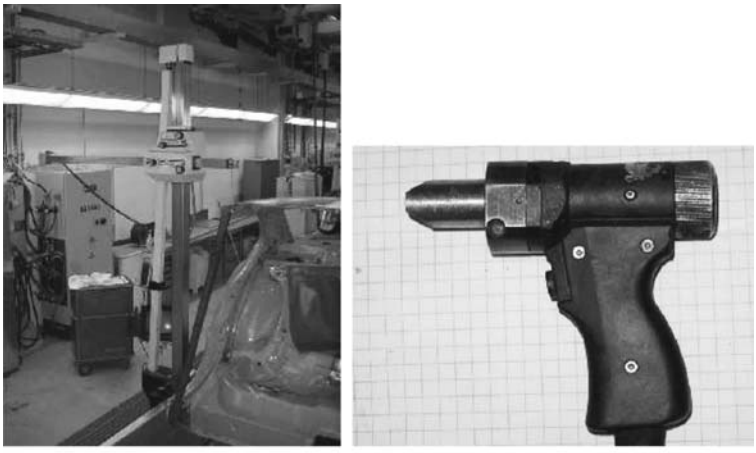
3.1.3 Discussion

AR Tic-Tac-Toe is an early, and quite simple, example illustrating the concepts behind our work. It does not address multichannel communication but, rather, focuses on the ease of use of tangible objects instead. The users generally liked the easy-to-understand and natural interactions within Tic-Tac-Toe. Mobile aspects have not been examined since board games are inherently located in fixed locations. When the Tic-Tac-Toe system was developed, the DWARF system did not yet exist.



(a) Placement of a new stone **(b)** Signalling the end of user action

Fig. 5a, b AR Tic-Tac-Toe. **a** Placement of a new stone. **b** Signaling the end of user action



(a) The old measuring device (b) The original welding gun

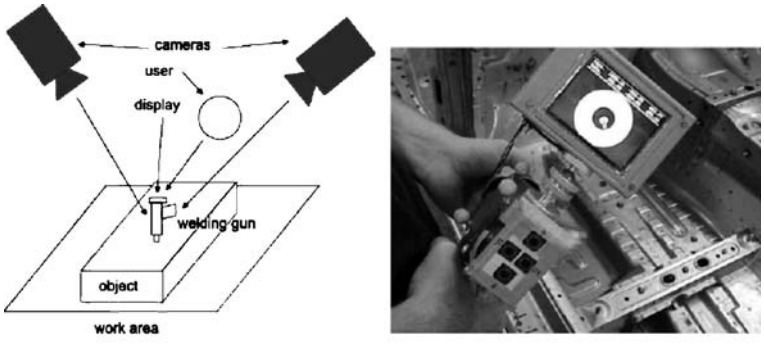
Fig. 6a, b The old setup for stud welding. **a** The old measuring device. **b** The original welding gun

Thus, this system is a good prototype for the traditional way of developing a system. Focused on the main event loop of a C program, the system used specially developed functions to use and interpret new video data, as well as event-oriented action items. Depending on the thereby resulting internal system state, the Tic-Tac-Toe game then proceeded to analyze the current viewer position, as well as the current state on the physical board game to decide upon its next move. The resulting motions of chips on the board were then transmitted to the users in an augmented, three-dimensional way.

3.2 PAARTI

3.2.1 System overview

The practical application of augmented reality in technical integration (PAARTI) system [11] was a collaboration between our research group and BMW. It helps welders shoot studs with high precision in experimental vehicles. A prototype was built within 6 months to successfully demonstrate to BMW that it was possible to automatically guide a worker to place his welding gun with high precision at the correct locations on a car frame. The system has been tested by a number of welders. It shows significant time improvements over the traditional stud welding process. Our prototype is currently in the process of being modified and installed for production use. A common task in the automotive industry is to build prototype cars. A time consuming part of this process is the manual shooting of studs into a car frame. The studs have to be placed very accurately with required precisions in the millimeter range. The former process was to measure the points where the studs have to be placed with a high precision position locator (see Fig. 6a) and after that, to shoot the stud with the welding gun (Fig. 6b). After examining the problem systematically (as described in more detail in [11]), we identified several different options for positioning the mobile, AR-equipped display: on the user (in form of an HMD), arbitrarily within the environment (e.g., on a nearby wall), or on the tool performing the welding operation. Within the given context, the third solution appeared optimal with respect to the main issues, such as achieving well specified and guaranteed levels of precision, as well as being intuively usable. We thus decided to use the setup illustrated in Fig. 7a. It builds upon the concept of an intelligent welding gun, a regular welding gun with a display attachment, a few buttons for user interactions, and reflective markers to track the gun position from stationary cameras (see Fig. 7b).



(a) Sketch of the system (b) The intelligent welding gun

Fig. 7a, b The new setup as realized in PAARTI. **a** Sketch of the system. **b** The intelligent welding gun

For tracking, we used the commercial ART system (http://www.ar-tracking.de), which is a high-precision (accuracy better than 1 mm) optical tracking system. Several cameras are mounted around the area that contains the tracked items. Near the cameras, there is an emitter for infrared light. The infrared rays are reflected from the targets, which allows the ART system to locate their positions. A target is composed of some retro-reflective spheres. Several targets can be tracked at the same time. We used the same tracking system as for ARCHIE and SHEEP. Pictures of the targets can be seen in Sect. 3.3, Fig. 9.

3.2.2 Interactions

While welders operate and move the gun, the display shows three-dimensional stud locations on the car frame relative to the current gun position (Fig. 8). Navigational metaphors, such as notch and bead and a compass, are used to help welders place the gun at the planned stud positions with the required precision. When a stud has been welded onto the car frame and the welder is satisfied with the result, he presses a button on the gun as confirmation. Then. he will be guided by the display on the gun to the next stud location. This process continues until the car frame is fully equipped with studs.

3.2.3 Discussion

PAARTI is a stationary system, as the car frame and the intelligent welding gun have to be located in a high-precision tracking environment, which are, so far, only available as stationary systems. The interactions within PAARTI have been tested with several welders from BMW. The evaluation showed that the tangible interactions with the intelligent welding gun were liked very much by them. Multichannel interactions were not necessary because the relatively simple process of welding does not need any more information than the one delivered by the intelligent welding gun. We think that an interesting lesson that can be learned from PAARTI is the way in which the tangible user interface was designed. A tool that was already used by the welders was enhanced with additional information. This pattern of augmenting already existing tools with additional information could be used in several other scenarios as well.

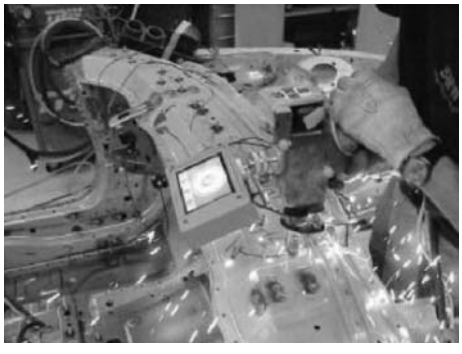


Fig. 8 The improved welding process

3.3 SHEEP

3.3.1 System overview

One of the major challenges of current computer systems is to provide users with suitable means to plan, model, and control complex operations that consist of many inter-related processes and dependencies. Multimodal, multi-user interaction schemes are needed to provide adequate control metaphors. It is our hypothesis that TUIs provide particularly intuitive means for controlling complex systems.

To demonstrate the potential of TUIs to dynamically visualize, manipulate, and control inter-related processes, we have built SHEEP. SHEEP is a multiplayer game centered around a physical table with a pastoral landscape that contains a herd of virtual and real sheep. The landscape and virtual sheep are projected from a ceiling-mounted projector. Players can assume one of several roles. According to their different roles, players use different input devices and interaction technologies to interact with the game. Within this system, every sheep is an independent process, communicating with other processes to attract or repel each other. SHEEP processes can be created or deleted. They can be visualized and manipulated using various modalities that are packaged into tangible units. The system was first demonstrated at the 2002 International Symposium on Mixed and Augmented Reality (ISMAR 2002) [43] and formally presented at the 2003 International Symposium on Mixed and Augmented Reality (ISMAR 2003) [25].

3.3.2 Interactions

The SHEEP game contains numerous interactions which are explained below:

- Coloring sheep: Figure 9a shows the view through an HMD onto the table. A player wearing an HMD can pick up sheep with his tracked hand (note the marker that is attached to the player's hand) and color them by moving them into color bars shown inside the HMD. After that, he can drop the sheep back onto the table again.
- Attracting sheep: The scene also allows for real, tangible sheep. One such object, a small Lego toy, is shown in Fig. 9b. Since all sheep are programmed to stay together, the entire herd can be controlled by moving the tangible sheep—thereby making it be the leader of the herd. Moving the sheep around constitutes a tangible interaction that was very popular

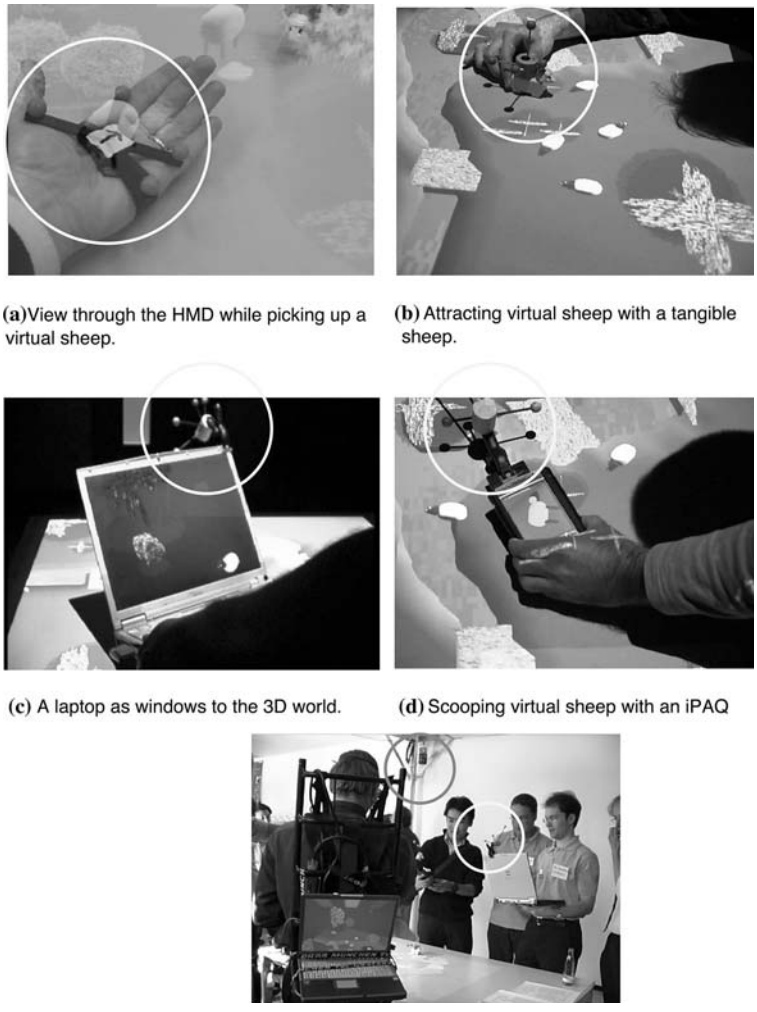


Fig. 9a–e An overview of the various interactions realized in SHEEP. Additionally, infrared-reflective markers (*light circles*) and the camera that is detecting them are highlighted (*dark circle* in **e**). **a** View through the HMD while picking up a virtual sheep. **b** Attracting virtual sheep with a tangible sheep. **c** A laptop as a window to the three-dimensional world. **d** Scooping virtual sheep with an iPAQ. **e** Demonstration of SHEEP at ISMAR 2002

among users because of its immediate comprehensibility.
- Exploring the three-dimensional world: A separate laptop can be used to view the scene on the table in three dimensions from arbitrary vantage points (Fig. 9c). This constitutes a tangible interaction, with the metaphor of moving a window about in the three-dimensional world similar to the active lens of the metaDESK [50].
- Creating and removing sheep: By putting on a headset with a microphone and grabbing a tracked magic wand, a player can create new sheep and remove sheep from the table. This is done by multimodal point-and-speak input. The technical realization and visual sensation for the users has been shown in Sect. 2.3 (Fig. 3).
- Scooping sheep: Players equipped with a tracked iPAQ (Fig. 9d) can use it to scoop sheep up from the table. Scooped sheep can be dropped back somewhere else on the table. During the scooping operation, the scooped sheep is displayed on the palm-sized computer. The entire interaction is illustrated in Fig. 10. This interaction is similar to pick-and-drop [42], however, we use a PDA to pick up virtual objects instead of a stylus.

3.3.3 Discussion

SHEEP is a stationary system that extensively uses tangible interactions. Synergistic multimodal input was used to improve the usability of the system. Coordinated multimedia output was deployed to enhance the immersivity of the user's experience. Audio feedback was used whenever possible and a large number of displays was used: tracked laptop, iPAQ, projected landscape on table, and an HMD. This system was the first test of the multichannel abilities of the DWARF user interface framework. On a technical level, we were successful in validating our claims about the benefits of our framework. The three-dimensional viewer had to be reimplemented, as we had several problems with its implementation for SHEEP (details can be found in [25]). The improved version of the three-dimensional viewer was successfully deployed in ARCHIE, which is described in the next section.

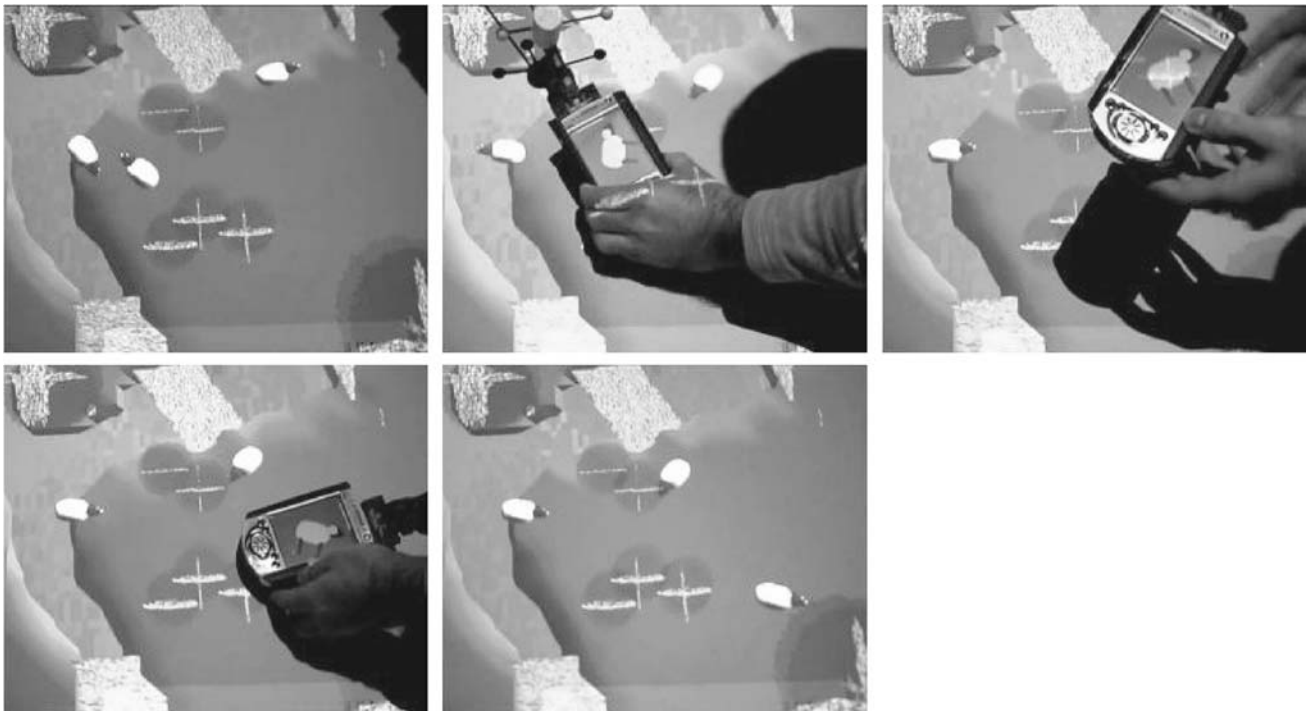


Fig. 10 Scooping a virtual sheep with an iPAQ that acts as tangible display

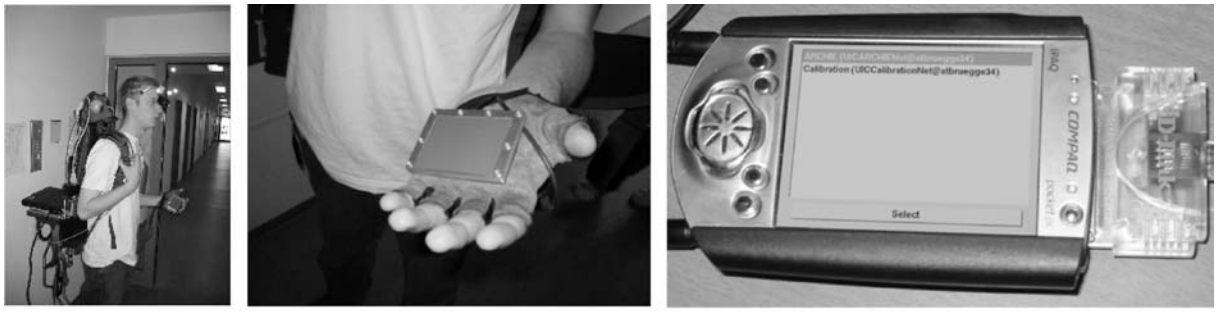
3.4 ARCHIE

3.4.1 System overview

The augmented reality collaborative home improvement (ARCHIE) system was developed as a team effort by eight Masters students within half a year. The goal of the project was to develop a system to support architects in constructing buildings. Usually, a number of people with different interests are involved in the development process of a building. The potential buyer has to mandate an architectural office to initiate the building process because the process is too complex to handle for himself. A mediator is responsible for representing the of the later building owner's interest towards the architect. The architect assigns work to specialized persons, for example, to technical engineers for designing plans of the wiring system. Although the later building owner is the contact person for the architect's office, he is only one of the many stakeholders interested in the building. Furthermore, landscape architects have to approve the geographic placement of the new building in the landscape. Last but not least, a building company has to be involved as well. End users should be given the option to give feedback during the design phase too. So, the architect's office receives feedback from many participants about their plans. The system we developed to support the collaboration between these groups of users is described below: An architect who is equipped with a wearable computer enters the office of the future. He can now select which of the two applications to start: modeling or presentation. Modeling lets the architects work collaboratively on a virtual model that is placed on a table. They have various tangible tools to modify the model. Presentation lets the architect present the modeled building to customers. There have already been a variety of similar systems [38, 52, 1, 7]. With ARCHIE, we had a different focus compared to those systems. Instead of focusing on the overall usability by the end users, we were more trying to test drive our framework and evaluate how it can deal with such a complex scenario.

3.4.2 Interactions

- Mobile interactions: The architect enters the office with his wearable device (see Fig. 11a). The wearable device is equipped with an HMD and a custom input device that was developed by Columbia University's Computer Graphics and User Interfaces Lab [6]. Basically, it is a touch pad fixed inside a glove, as can be seen in Fig. 11b. When the user enters the room, the involved components of the DWARF framework detect that they are ready to start up the modeling or the presentation application. An appropriate GUI is displayed to the user. Depending on the actual configuration of the wearable, there are two possible GUIs and dependent interactions; either the GUI is displayed on an iPAQ (see Fig. 11c) and the user can select the desired application by touching the respective menu entry, or when the architect wears an HMD, the GUI is displayed inside the HMD and the selection is made via the TouchGlove.
- Modeling: The setup for this part, which is similar to SHEEP, can be seen in Fig. 12a. An architect can take a tangible wall, move it somewhere, and create a new



(a) The wearable system (b) The TouchGlove input device (c) Selecting functions with the iPAQ.

Fig. 11a–c Mobile parts of ARCHIE. **a** The wearable system. **b** The TouchGlove input device. **c** Selecting functions with the iPAQ

virtual wall object by pressing the create button on his input device or using speech input. Another architect can help by moving around a tangible sun to simulate different lighting conditions, which is very important for architectural design. Both continue their work and build an approximate outer shape for a new building (see Fig. 12b).

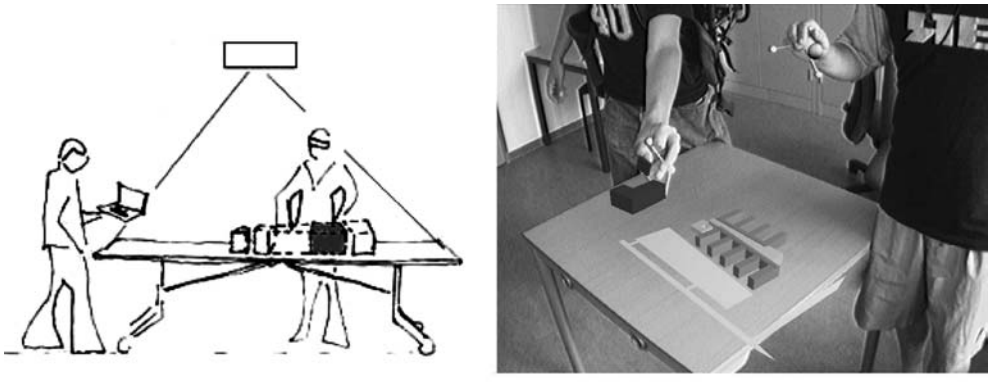
– Presentation: When the architect selects the presentation application, a video view is started, which is projected on a wall instead of using an HMD. Instead of the previously used HMD, a projected view is started, providing scenes as seen from a tangible camera object, as seen in Fig. 13c. The audience can now get a spatial understanding of the proposed building that is displayed on the projection screen. The model to be presented is rendered stereoscopically in anaglyphic red-cyan (Fig. 13b); for a realistic three-dimensional view, the visitors need to wear the corresponding red-cyan glasses (Fig. 13a).

3.4.3 Discussion

ARCHIE is one of the most complex applications we have built so far. Its mobile part shows the abilities of DWARF for the combination towards wearable and ubiquitous computing. Multichannel and tangible interactions were implemented similarly to in SHEEP and, once again, validated the claims regarding the expressiveness of our framework. ARCHIE was presented to architects who responded to it very well and liked it a lot. They suggested that an integration with already existing tools used in architecture (e.g., three-dimensional Studio Max) would increase acceptance among architects. The modeling within ARCHIE is still very simple and will be enhanced in future versions by features like selection of different materials or group selections and manipulations of virtual objects. Most importantly to us, ARCHIE provided new tools towards automatically setting up and testing new user interfaces for architects. While such customers were unaware of discussing topics related to the architectural design of a building at hand, we were also able to collect spontaneously provided usability data regarding the preference of users to using various interaction tools, as well as the time they needed to use such tools to respond to system requests [24]. This intermediate level of system use, as well as system evaluation, is what we are striving for. We are gathering increasing amounts of evidence that DWARF is a very suitable toolkit for this purpose.

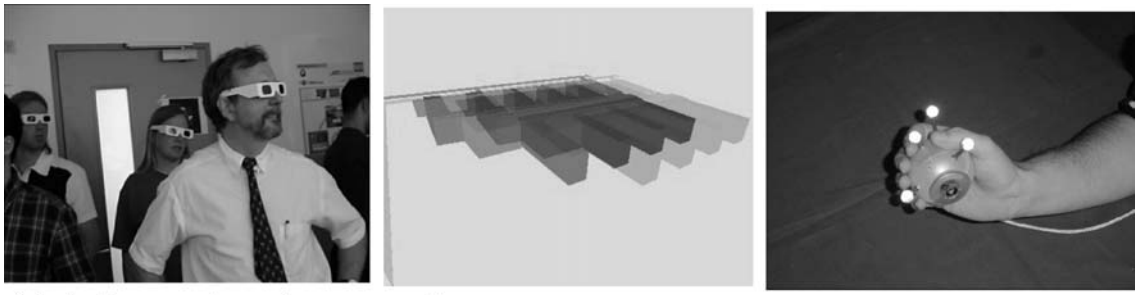
4 Summary and future work

In this paper, we have postulated a holistic approach to user interface development. Based on the observation that a confluence between multimedia-based, multimodal, wearable, ubiquitous, tangible, and augmented



(a) Conceptual drawing of the setup (courtesy of Manja Kurzak) (b) Tangible objects for modelling and visualization: sun and wall

Fig. 12a, b Collaborative modeling of a building. **a** Conceptual drawing of the setup (courtesy of Manja Kurzak). **b** Tangible objects for modeling and visualization: sun and wall



(a) Audience wearing red-cyan glasses

(b) View rendered in red-cyan

(c) The tangible camera controlling the viewpoint for the presentation

Fig. 13a–c Presentation of a planned building to an audience. **a** Audience wearing red-cyan glasses. **b** View rendered in red-cyan. **c** The tangible camera controlling the viewpoint for the presentation

reality (AR) based user interfaces exists, we have set out to describe the overall technical requirements for an overarching architecture. Our distributed wearable augmented reality framework (DWARF) is able to fulfill the central requirements and has recently been extended with interaction elements from zoomable and attentive user interfaces (http://www1.in.tum.de/DWARF/CarMovie. On its basis, we have proposed a user interface architecture for ubiquitous augmented reality (UAR). A number of prototype demonstration systems were shown to fit well into this framework. One of the major goals of our research is to provide a rapid prototyping environment within which new user interface ideas can be prototyped and tested easily. To this end, the current setup has already proven to be suitable for joint, online development, testing, and enhancement of both individual interaction facilities and multimodal, ubiquitous combinations thereof. The SHEEP game was partly developed in such joint sessions, which we called Jam sessions [25]. Furthermore, Kulas demonstrated within the augmented reality collaborative home improvement (ARCHIE) system that usability evaluations can be seamlessly integrated into the live development and testing process [24]. To this end, user evaluation processes can be created to automatically inspect and evaluate the data streams flowing between the individual interaction devices, tangible objects, users, displays, etc. As a next step, the user interface architecture (Fig. 2) can serve as the basis to dynamically integrate further, in-depth enhancements to the analysis and interpretation of user input. By including tools to track a user's gestures or mimics (e.g., by eye tracking), a cognitive model of the user can be accumulated. Combined with further sources of environmental context data, the user interface controller (UIC) can be extended to react adaptively to changing situations. By extending the information presentation primitives of the three-dimensional viewer, new presentation metaphors, as well as context-dependant layouts of information within an augmented environment, can be provided. Note that all of these enhancements can be included, tested, modified, and/or rejected online and non-exclusively. The result is a live, dynamically changeable, and also dynamically adaptive, development environment for UAR user interfaces. The environment can, thereby, provide us with the opportunity to easily explore, combine, and test different concepts that are currently emerging, while also providing increasing degrees of automatic adaptation by the tools themselves. We expect this flexible, dynamically changeable setup to be able to provide us with the tools to generate and test new concepts much more rapidly and flexibly.

Acknowledgements Special thanks go to our students Daniel Pustka, Franz Strasser, Gerrit Hillebrand, Marco Feuerstein, Ming-Ju Lee, Otmar Hilliges, Chris Kulas, Manja Kurzak, Felix Loew, Marcus Tönnis, Johannes Wöhler, and Bernhard Zaun for developing many useful components and tools; without these, this work would not have been possible. The Ph.D. students who designed and developed most parts of DWARF are: Thomas Reicher, Martin Bauer, Martin Wagner, and Asa MacWilliams. The Columbia University's Computer Graphics and User Interfaces Lab provided us with two prototypes of the TouchGlove; we would like to thank Steven Feiner and Gabor Blasko for their support. The tracking system used for SHEEP was partially on loan from BMW (TI-360). This work was partially supported by the High-Tech-Offensive of the Bayerische Staatskanzlei, Germany (Prof. Brügge).

References

1. Anderson D, Marks J, Agarwala A, Beardsley P, Leigh D, Sullivan E, Yedidia J, Frankel J, Hodgins JK , Ryall K (2000) Tangible interaction and graphical interpretation: a new approach to intuitive 3D modeling. In: Proceedings of the 27th ACM annual conference on computer graphics (SIGGRAPH 2000), New Orleans, Louisiana, July 2000
2. R Azuma (1997) A survey of augmented reality. Presence–Teleop Virt 6(4):355–385
3. Bauer M, Bruegge B, Klinker G, MacWilliams A, Reicher T, Riss S, Sandor C, Wagner M (2001) Design of a component-based augmented reality framework. In: Proceedings of the IEEE and ACM international symposium on augmented reality (ISAR 2001), New York City, New York, October 2001, pp 45–53
4. Bauer M, Bruegge B, Klinker G, MacWilliams A, Reicher T, Sandor C, Wagner M (2002) An architecture concept for ubiquitous computing aware wearable computers. In: Proceedings of the 2nd international workshop on smart appliances and wearable computing (IWSAWC 2002), Vienna, Austria, July 2002

5. Bauer M, Hilliges O, MacWilliams A, Sandor C, Wagner M, Newman J, Reitmayr G, Fahmy T, Klinker G, Pintaric T, Schmalstieg D (2003) Integrating Studierstube and DWARF. In: Proceedings of the international workshop on software technology for augmented reality systems (STARS 2003), Tokyo, Japan, October 2003
6. Blasko G, Feiner S (2002) A menu interface for wearable computing. In: Proceedings of the 6th international symposium on wearable computers (ISWC 2002), Seattle, Washington, October 2002, pp 164–165
7. Broll W, Meier E, Schardt T (2000) The virtual round table—a collaborative augmented multi-user environment. In: Proceedings of the 3rd international conference on collaborative virtual environments (CVE 2000), San Francisco, California, September 2000, pp 39–45
8. W Buxton (1991) The three mirrors of interaction: a holistic approach to user interfaces. In: Proceedings of the Friend21 international symposium on next generation human interfaces, Tokyo, Japan, November 1991
9. Caroll JM (2002) Human–computer interaction in the new millenium. Addison-Wesley, Reading, Massachusetts
10. Davies JM (1998) An ambient computing system. Masters thesis, Department of Electrical Engineering and Computer Science, University of Kansas, Kansas
11. Echtler F, Sturm F, Kindermann K, Klinker G, Stilla J, Trilk J, Najafi H (2003) The intelligent welding gun: augmented reality for experimental vehicle construction. In: Ong S, Nee A (eds) Virtual and augmented reality applications in manufacturing, chap 17. Springer, Berlin Heidelberg New York
12. Feiner S, MacIntyre B, Haupt M, Solomon E (2003) Windows on the world: 2D windows for 3D augmented reality. In: Proceedings of the 6th annual ACM symposium on user interface software and technology (UIST'93), Atlanta, Georgia, November 1993. ACM Press, New York, pp 145–155
13. Gamma E, Helm R, Johnson R, Vlissides J (1995) Design patterns: elements of reusable object-oriented software. Addison-Wesley, Reading, Massachusetts
14. Garlan D, Siewiorek D, Smailagic A, Steenkiste P (2002) Project Aura: toward distraction-free pervasive computing. IEEE Pervas Comput 1(2):22–31
15. Jacob RJK, Deligiannidis L, Morrision S (1999) A software model and specification language for non-WIMP user interfaces. ACM Trans Comput–Hum Interact 6:1–46
16. Jensen K, Rozenberg G (1991) High-level Petri nets: theory and applications. Springer, Berlin Heidelberg New York, ISBN 3-540-54125 X
17. Johanson B, Fox A (2004) Extending tuplespaces for coordination in interactive workspaces. J Syst Softw 69:243–266
18. Johanson B, Fox A, Winograd T (2002) The interactive workspaces project: experiences with ubiquitous computing rooms. IEEE Pervas Comput 1:67–74
19. Kaiser E, Olwal A, McGee D, Benko H, Corradini A, Li X, Feiner S, Cohen P (2003) Mutual disambiguation of 3D multimodal interaction in augmented and virtual reality. In: Proceedings of the 5th international conference on multimodal interfaces (ICMI 2003), Vancouver, British Columbia, November 2003. ACM Press, New York, pp 12–19
20. Kato H, Billinghurst M, Poupyrev I, Tetsutani N (2001) Tangible augmented reality for human–computer interaction. In: Proceedings of the 17th Japanese conference of computer graphics (NICOGRAPH 2001), Nagoya, Japan, November 2001
21. Klinker G, Reicher T, Bruegge B (2000) Distributed tracking concepts for augmented reality applications. In: Proceedings of the IEEE and ACM international symposium on augmented reality (ISAR 2000), Munich, Germany, October 2000
22. Klinker G, Stricker D, Reiners D (1999) An optically based direct manipulation interface for human–computer interaction in an augmented world. In: Proceedings of the 5th EUROGRAPHICS workshop on virtual environments (EGVE'99), Vienna, Austria, June 1999
23. Kortuem G, Schneider J (2001) An application platform for mobile ad-hoc networks. In: Proceedings of the workshop on application models and programming tools for ubiquitous computing (UbiTools 2001), Atlanta, Georgia, September 2001
24. Kulas C (2003) Usability engineering for ubiquitous computing. Masters thesis, Technische Universität München, Munich, Germany
25. MacWilliams A, Sandor C, Wagner M, Bauer M, Klinker G, Bruegge B (2003) Herding SHEEP: live development of a distributed augmented reality system. In: Proceedings of the 2nd IEEE and ACM international symposium on mixed and augmented reality (ISMAR 2003), Tokyo, Japan, October 2003
26. Mattern F (2001) Pervasive/ubiquitous computing. Informatik-Spektrum 24:145–147
27. Maybury M, Whalster W (eds) (1998) Readings in intelligent user interfaces. Morgan Kaufmann, San Mateo, California
28. Milgram P, Kishino F (1994) A taxonomy of mixed reality visual displays. IEICE Trans Inform Syst E77-D(12): 1321–1329
29. Nigay L, Coutaz J (1993) A design space for multimodal systems: concurrent processing and data fusion. In: Proceedings of the joint conference of ACM SIGCHI and INTERACT (InterCHI'93), Amsterdam, The Netherlands, April 1993. IOS Press, Amsterdam, The Netherlands, pp 172–178
30. Novak V (2004) Attentive user interfaces for DWARF. Masters thesis, Technische Universität München, Munich, Germany
31. Olsen D (1992) User interface management systems: models and algorithms. Morgan Kaufmann, San Mateo, California
32. Olwal A (2002) Unit—a modular framework for interaction technique design, development and implementation. Masters thesis, Royal Institute of Technology (KTH), Stockholm, Sweden
33. Oviatt SL (1999) Ten myths of multimodal interaction. Commun ACM 42:74–81
34. Oviatt SL (1999) Mutual disambiguation of recognition errors in a multimodel architecture. In: Proceedings of the ACM SIGCHI conference on human factors in computing systems (CHI'99), Pittsburgh, Pennsylvania, May 1999. ACM Press, New York, pp 576–583
35. Oviatt SL (2000) Multimodal interface research: a science without borders. In: Proceedings of the 6th international conference on spoken language processing (ICSLP 2000), Beijing, China, October 2000
36. Perlin K, Fox D (1993) Pad: an alternative approach to the computer interface. Comput Graph 27:57–72
37. Poupyrev I, Billinghurst M, Weghorst S, Ichikawa T (1996) The go-go interaction technique: non-linear mapping for direct manipulation in VR. In: Proceedings of the 9th annual ACM symposium on user interface software and technology (UIST'96), Seattle, Washington, November 1996. ACM Press, New York, pp 79–80
38. Rauterberg M, Fjeld M, Krueger H, Bichsel M, Leonhardt U, Meier M (1998) BUILD-IT: a planning tool for construction and design. In: Proceedings of the ACM SIGCHI conference on human factors in computing systems (CHI'98), Los Angeles, California, April 1998. ACM Press, New York, pp 177–178
39. Regenbrecht H, Wagner M (2002) Interaction in a collaborative augmented reality environment. In: Proceedings of the ACM SIGCHI conference on human factors in computing systems (CHI 2002), Minneapolis, Minnesota, April 2002
40. Reicher T, MacWilliams A, Bruegge B, Klinker G (2003) Results of a study on software architectures for augmented reality systems. In: Proceedings of the 2nd IEEE and ACM international symposium on mixed and augmented reality (ISMAR 2003), Tokyo, Japan, October 2003
41. Reitmayr G, Schmalstieg D (2001) OpenTracker—an open software architecture for reconfigurable tracking based on XML. In: Proceedings of the IEEE virtual reality conference (VR 2001), Yokohama, Japan, March 2001, pp 285–286

42. Rekimoto J (1997) Pick-and-drop: a direct manipulation technique for multiple computer environments. In: Proceedings of the 10th annual ACM symposium on user interface software and technology (UIST'97), Banff, Alberta, Canada, October 1997. ACM Press, New York, pp 31–39
43. Sandor C, MacWilliams A, Wagner M, Bauer M, Klinker G (2002) SHEEP: the shared environment entertainment pasture. In: Demonstration at the IEEE and ACM international symposium on mixed and augmented reality (ISMAR 2002), Darmstadt, Germany, September/October 2002
44. Schmidt A, Gellersen H-W, Beigl M, Thate O (2000) Developing user interfaces for wearable computers—don't stop to point and click. In: Proceedings of the international workshop on interactive applications of mobile computing (IMC 2000), Rostock, Germany, November 2000
45. Shneiderman B (1997) Designing the user interface. Addison-Wesley, Reading, Massachusetts
46. Song D, Norman M (1993) Nonlinear interactive motion control techniques for virtual space navigation. In: Proceedings of the IEEE virtual reality annual international symposium (VRAIS'93), Seattle, Washington, September 1993, pp 111–117
47. Strauss P, Carey R (1992) An object-oriented 3D graphics toolkit. In: Proceedings of the 19th ACM annual conference on computer graphics (SIGGRAPH 1992), Chicago, Illinois, July 1992, vol 26, issue 2, pp 341–349
48. Stricker D, Klinker G, Reiners D (1998) A fast and robust line-based optical tracker for augmented reality applications. In: Proceedings of the 1st IEEE international workshop on augmented reality (IWAR'98), San Francisco, California, November 1998. AK Peters, Wellesley, Massachusetts, pp 129–145
49. Turk M, Robertson G (2000) Perceptual user interfaces (introduction). Commun ACM 43:32–34
50. Ullmer B, Ishii H (1997) The metaDESK: models and prototypes for tangible user interfaces. In: Proceedings of the 10th annual ACM symposium on user interface software and technology (UIST'97), Banff, Alberta, Canada, October 1997. ACM Press, New York, pp 223–232
51. Ullmer B, Ishii H (2000) Emerging frameworks for tangible user interfaces. IBM Syst J 39(3–4):915–931
52. Underkoffler J, Ishii H (1999) Urp: a luminous-tangible workbench for urban planning and design. In: Proceedings of the ACM SIGCHI conference on human factors in computing systems (CHI'99), Pittsburgh, Pennsylvania, May 1999. ACM Press, New York, pp 386–393
53. Vertegaal R (2003) Attentive user interfaces. Commun ACM 46(3):40–46
54. Waibel A, Vo MT, Duchnowski P, Manke S (1995) Multimodal interfaces. Artif Intell Rev 10:299–319
55. Weiser M (1993) Hot topics: ubiquitous computing. IEEE Comput 26:71–72