3DSim: Rapid Prototyping Ambient Intelligence

Ali A. Nazari Shirehjini and Felix Klar

Fraunhofer-IGD
Departments “Interactive Multimedia Appliances” and “Communication and Cooperation”
Fraunhoferstr. 5, 64283 Darmstadt, Germany
{ali.nazari, felix.klar}@igd.fraunhofer.de

Abstract

The present paper introduces 3DSim, a tool for rapid prototyping *Ambient Intelligence* applications. A major feature of this work is the use of a 3D-based virtual environment to represent an intelligent meeting space allowing for prototyping a number of components such as lights, blinds, SMART Boards™, large display walls, projectors, *aware chairs* and humans as well as human activity animation. As a result of using standardized interfaces, any UPnP control point may be used to invoke actions on *virtual* UPnP devices. Such devices are dynamically inserted to the environment and can be removed at run-time. Data delivered by sensors is interpreted by an environment monitoring component which routes higher-level atomic context information to virtual objects. As a result of invoked device actions and gathered context information, resulting state changes (e.g., switching light states) are visualized within the scene.

1. Introduction

Ambient Intelligence (AmI) [1, 2] is a paradigm of information technology, in which informationalized objects are introduced to everyone’s physical environment. These intelligent everyday objects, as well as other embedded systems, have the common goal to adapt to the user’s needs and situations, in order to support him on common everyday tasks.

Research and investigation of different aspects of AmI requires fully developed physical spaces including intelligent actuators, sensors or context management systems. Developing other orthogonal AmI *building blocks,* such as adaptive user interface, reactive agent systems, multi-modal dialogue management systems or multimedia output coordination requires to deploy those building blocks to several environments in order to validate them under various conditions and context domains. This requires access to various real AmI spaces such as intelligent homes, aware meeting rooms, adaptive classrooms, or reactive office environments. Building up such environments can become very cost-intensive and time-consuming.

In this paper we introduce a photo-realistic, 3D-based rapid prototyping and simulation environment system named *3DSim*. It allows to develop above-mentioned *orthogonal* AmI *building blocks* without demanding to set up an underlying physical environment already in the early development phases (see Fig. 1).

However, 3DSim doesn’t replace systematic evaluations under real life conditions within physically existing intelligent spaces. Rather it simulates a fully equipped environment (e.g., a meeting room) by providing a set of available devices and actuators which can be dynamically integrated within the environment, as well as the possibility to simulate persons which are represented by avatars. 3DSim also provides components simulating various sensor entities as well as an environment monitoring subsystem. Provided devices offer full UPnP support (as will be described later on), thus building the basic execution layer for developing higher level AmI building blocks such as reactive-agent systems. By providing a photo-realistic environment visualization, UPnP devices and several sensors, 3DSim supports the development life cycle for above-mentioned orthogonal AmI building blocks.

3DSim offers the possibility to integrate new devices and sensor components as well as to interact with those devices and sensors, i.e. to invoke actions on available devices to gather raw sensor data or to query context data managed by the environment monitoring subsystem. Therefore, AmI building blocks (e.g., situation-recognition, goal-based interaction [3]) built upon 3DSim could be easily deployed to real physical environments which support the same standards and interfaces such as the UPnP standard.

After introducing use cases and functional description of 3DSim, the present paper describes the underlying architecture and finally discusses future work and additional use cases for 3DSim.

2. Functional Description of 3DSim

From a functional point of view, 3DSim supports AmI developers targeting

- human-ambient-interaction solutions
- dialogue management and multimedia output coordination systems
- situation-recognition components
- strategy components and re-active agent-systems allowing to adequately react on specific user situations

2.1 Human-Ambient-Interaction Solutions

Through the photo-realistic and interactive 3D visualization of a virtual AmI domain, developers of user interface components get a nearby realistic impression of the visualized domain. This allows to get a first judgement of the usability of the interaction system and developed metaphors for the targeted domain. As an example, remote environment control systems developed especially for home environments (such as the Philips iPronto [4]) can be deployed to meeting environments using 3DSim in order to test their suitability to new domains which are composed of different device types thus requiring different interaction metaphors. Considering smaller research groups which mostly carry on only one domain (e.g., home or work) as a testbed for their interaction solutions, efficient development and copious testing (e.g., for adaptive user interfaces) becomes possible for *several* domains – even domains that have no physical devices, yet.

2.2 Dialogue Management

Within the virtual environment, persons can use 3D gestures (3D pointing metaphors) to point to devices and room objects. In contrast to camera based 3D gesture tracking systems, 3DSim provides reliable information on user gestures, thus allowing to develop dialogue management systems.

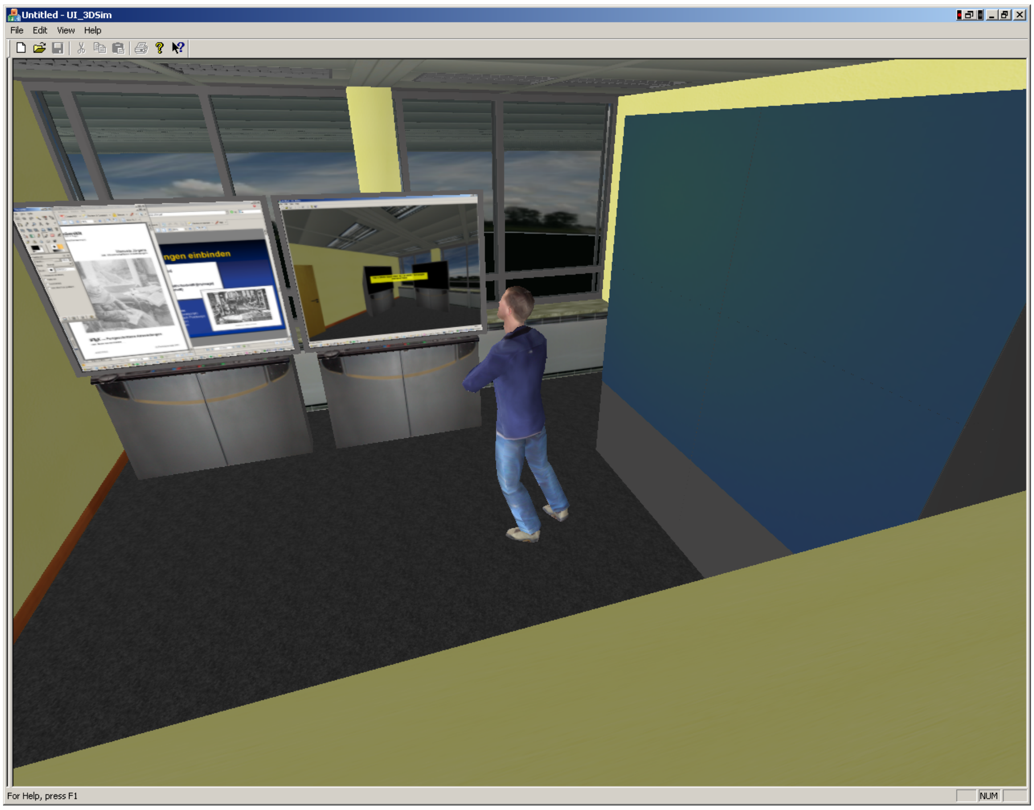

Figure 1: 3D-based rapid prototyping environment with three displays inserted and an avatar looking at a SMART Board.

2.3 Situation-Recognition

3DSim has an integrated *environment monitoring system* (EMS), which is responsible for analyzing sensor data (i.e., performing a situation recognition) and *detecting* higher level environment state transitions. This data can arrive from real physical sensors connected to 3DSim or from virtual sensors that can be attached to 3DSim by using its underlying plugin architecture described in section 4. After recognizing a new state, the EMS will interact with the involved objects of the 3D world, resulting in the visualization of the new state. In its basic configuration, 3DSim is able to process the following sensor data:

Location Information for each object, device and avatar.
Light Settings such as the intensity and the color of a light.
Device States for projector devices (stop / playing).

By providing above mentioned context data, more complex situation-recognition systems can be developed on the basis of 3DSim sensors and the EMS.

2.4 Strategy Components and Re-active Agent Systems

3DSim allows to validate and test the dynamic behaviour of:

- strategy components [3]
- planning components for multimedia output coordination
- *adaptive* user interfaces [5]
- device composition platforms [1]

This gets possible by *manipulating* ambient settings, environmental conditions or device states by providing virtual sensor data, as these are the most relevant input parameters determining the behaviour of mentioned components. These manipulations are done within the virtual 3D world, for example by adding new devices to the environment, by repositioning objects and avatars or by interacting with actuators and devices such as lights, blind shutters or displays. Especially media types such as pictures, sound and movies can be rendered on any available display device.

By doing so, a broad field of possibilities becomes available for testing the mentioned AmI building blocks. In contrast, testing such components within a real physical environment requires the physical availability of all those devices and the possibility to reposition them, to change their states, and manipulate arbitrary environmental conditions such as light or temperature. Considering the static nature of large wall display installations, the wheight of *mobile* back projection systems or the effort and limitations to get certain temperature and light settings in a room, the benefits offered by the presented rapid prototyping system get more visible.

2.5 Actuator and Sensor Integration Interface

UPnP actuators and sensor components triggering changes within the 3D scene are integrated into the system using TCP/IP-based eventing interfaces (see Fig. 2). Raw sensor data is sent to the EMS which interprets several sensor inputs and generates higher level atomic context information, such as the position of objects or avatars in a room.

Above-mentioned raw sensor data is provided either by 2D or 3D interaction or by external subsystems which use the event-based communication interface of 3DSim to deliver their raw sensor data to the EMS (see Fig. 2). Especially re-positioning of objects and persons, changing avatar animation modes (human activity such as writing, presenting or siting) is done by using 3D interaction means. For a set of specific objects more complex tracking information with object speed consideration can be simulated using third party sensor simulation software such as Ubisense Tracking Simulator [6].

UPnP actions can be invoked using arbitrary UPnP control points such as a generic control point coming with the Intel UPnP SDK. UPnP devices are integrated within the system

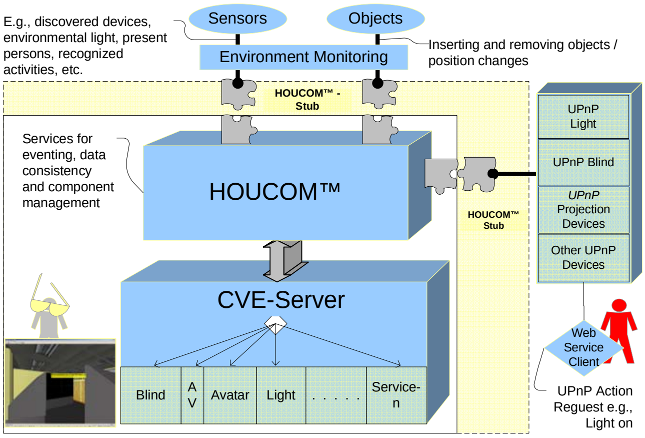

Figure 2: This figure shows the internal structure of 3DSim's rendering components. By providing TCP/IP-based open interfaces and the UPnP standard, 3DSim allows for easy integration of additional sensors and UPnP actuators.

by launching corresponding device stubs (as described in section 4.2).

As shown in Fig. 2, the eventing system of 3DSim is based on the HOUCOM system [7, 8]. Events provided by the EMS and UPnP components are routed to the rendering components using HOUCOM stubs. These stubs translate UPnP events and EMS messages into the specific ontology of a *CVE-Server*. Receiving these messages, the CVE-Server will update corresponding entities of the virtual environment which triggers a 3D user view change in a 3DSim-client.

3. Context Visualization

In its context visualization mode, 3DSim visualizes sensor events received from real sensor components. This allows to visualize and animate context events thus testing the accuracy of context awareness systems such as object positioning, user activity recognition, pressure sensors, intelligent chairs reporting their state, shutters and light sensors. By doing so, a smart environment designer can monitor the *awareness* of the environment he is currently setting up. For instance, 3DSim repositions objects (see Fig. 3) as soon as location information for an object have been made available by corresponding sensor systems.

From a technical perspective, this mode differs from the simulation mode, where sensor data is simulated by virtual sensors, only in the way sensor data is triggered. In the simulation mode, sensor data is provided by a 2D GUI and 3D gesture elements. In the context visualization mode, sensor data is provided by real sensors. In this mode, a physical smart environment is usually existing. The aim of a user employing 3DSim is to get a visualization of atomic context information provided by available sensors thus allowing to validate sensor data, sensor fusion algorithms or the results of context reasoning (activity-recognition) systems.

3.1 Visual Overlapping

3DSim enables testing the accuracy of a positioning sensor. Therefore the virtual representation of a real-world object is visualized within the scene at the right position and orientation. The tracked position and orientation of the device is used to visualize other instances of the same object. This results in a scene showing the object's desired and tracked position and orientation. This *overlapping mode* allows to visually determine the accuracy of received location information.

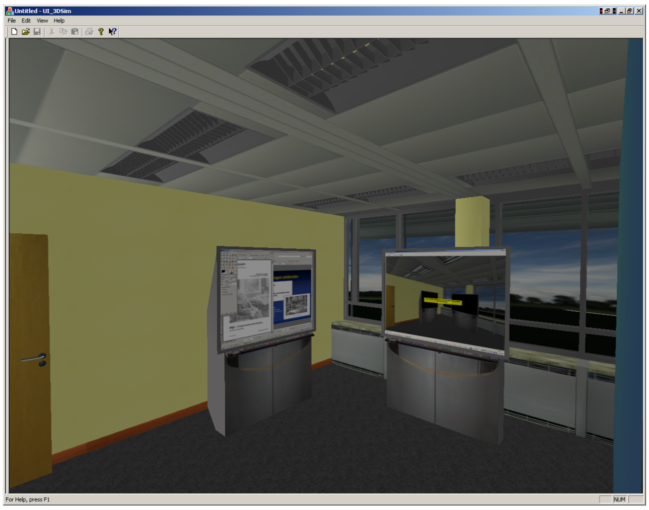

Figure 3: This figure shows two SMART Boards, re-positioned by 3DSim after receiving corresponding context information.

3.2 Human Activity Animation

Various avatar animations (cleaning whiteboard, annotating on a display, walking, sitting, writing, presenting, . . .) show the developer what activity is currently recognized by the EMS. For example, when a sensor is recognizing a real person that is writing something on a specific white board, 3DSim animates an avatar writing on that white board within the virtual environment.

3.3 Environmental and Device States

Environmental states (such as light) are visualized by adjusting 3D scene settings. Device states (moving up/down blinds, opening and darkening blinds, opening door and windows, switching on/off lights and display devices) are visualized when they are reported by the EMS. Corresponding lighting and shadowing effects are considered by the rendering engine in order to achieve a realistic visualization.

4. General Architecture of 3DSim

3DSim's core consists of a CVE-server which manages the 3D-environment and a 3DSim-client which provides a user interface that visualizes and allows interaction with the 3D-environment (see Fig. 4). The server holds the 3D-environment's state whereas a client operates on the 3D-environment.

3DSim is extendible by 3DSim-services, which are modules that plug into the 3D-environment, extending it with new 3D objects that simulate real-world objects. 3DSim-services may be added or removed during runtime, allowing to extend or contract the 3D-environment dynamically. Interaction with 3D objects is realized through the HOUCOM eventing-mechanism. If an object receives an event (e.g., from the EMS) it will modify its state depending on the received event data.

4.1 Visual Capabilities and Dynamic Component-Based Extensions

A 3DSim service developer will use the CVE-API to operate on a 3D object. Position, orientation and visibility state are the basic values that can be adjusted. CVE furthermore provides routines for collision detection, animation of objects and reaction on user input. It is also possible to define states that will periodically be observed during the simulation. If a predefined constellation is recognized, predefined reactions are triggered in the environment.

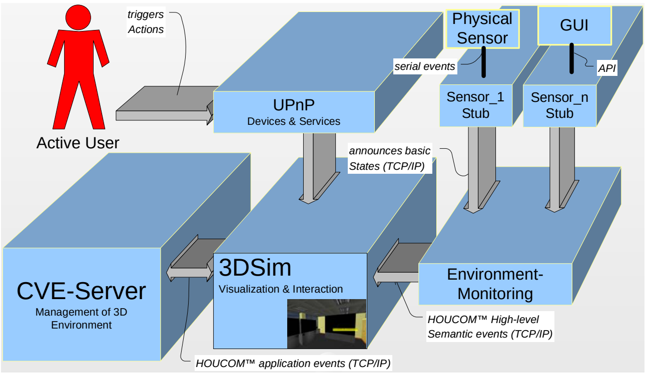

Figure 4: 3DSim system overview: elements and interaction between them

4.2 The Collaborative Virtual Environment Sub-System

CVE (Collaborative Virtual Environment) is a distributed 3D-environment that makes use of an underlying plugin architecture. It has been successfully deployed in EU-funded projects ELIN [9] and GlobalIT [8]. CVE is based upon HOUCOM [7] which provides TCP/IP based event-distribution and component life cycle management. A plugin architecture is realized by dynamic components (so called *services*). These services correspond to executables (DLLs) with known *interface implementation* which provide well-known callback-functions [7]. For rendering purposes the *RenderWare platform* [10] for game development is used. However, the CVE platform provides an abstraction layer allowing to deploy any rendering engine that supports a certain amount of functionality.

CVE allows building up a world with high level objects that are packaged in a service. A service can be loaded by the HOUCOM framework during runtime. A service holds the logic of a 3D object for a client and a server, handling-routines for events received by HOUCOM and routines that transfer the state of a 3D object between server and client. The CVE-system will route appropriate incoming HOUCOM events to the corresponding 3D object.

A 3DSim-service is a specialized CVE-service. It additionally provides a HOUCOM stub that translates events sent by UPnP-services or the EMS into HOUCOM events, in order that they will be understood by a 3D object. At startup-time each 3DSim-service (representing a device) will start the corresponding UPnP-device implementation on the client side, which provides interaction routines with the 3D object representing this device.

5. Conclusion

In previous sections, we presented the 3DSim system. Based on its open interfaces, it allows to integrate a various number of external sensors which usually are triggered by humans using GUI elements. Moreover, 3DSim comes with a set of UPnP devices which can be easily added to the system by dragging a corresponding device profile to a location within the scene. Received sensor data is interpreted by an *environment monitoring system* which sends higher-level atomic context information to a specific 3D object resulting in scene updates. Components developed using 3DSim can be easily adapted to physical environments which are developed using the same standards (currently only UPnP) as available within the virtual environment. The flexible architecture of 3DSim allows for easily extending the system by adding new components, which makes new device types available, e.g., new objects or just rather animations (activity simulation) for an avatar. Interactions are done using standard 2D GUIs and 3D metaphors.

Providing an almost real impression of the targeted domain becomes important to AmI-developers. In 3DSim this is achieved by providing a set of devices with a photo-realistic 3D model and with their actual measurements. Furthermore, the *interaction designer* is supported by providing 3D environments which are modeled using real life construction plans. As an important feature, 3DSim also allows to visualize multi-media documents on displays.

6. Related Work

MavHome ResiSim [11] provides an *abstract* virtual home allowing to simulate system behaviour. ResiSim has been developed with the focus to investigate behaviour of agent systems controling an intelligent home.

UbiWise [12] developed at HP Research Labs is a 3D-based simulator allowing users to create arbitrary environments. There, the user can include a set of devices. UbiWise simulates prototypes of new devices and protocols using a Java program. The UbiWise simulator concentrates on computation and communications devices.

In contrast, 3DSim allows to include any UPnP device into the environment and allows arbitrary UPnP control points to interact with those devices. UbiWise focuses on supporting new appliance development. Thus device representations are of low visual quality and do not provide a photo-realistic impression. In contrast, 3DSim aims to support the development of human-ambient-interaction systems such as PDA based control systems, adaptive user interfaces, multimedia output coordination or goal-based interaction systems.

7. Future Work

The current version of 3DSim provides already implemented mechanisms for

- to add / remove objects, devices and humans to the scene
- to change position and orientation of objects and humans
- to change device states by invoking UPnP actions using any UPnP control points
- to render well-established MIME types on any – even mobile – display device.
- to start animations (walking, sitting, standing up, writing, annotating a display, presenting, cleaning a display) for avatars
- to animate moving up and down blind shutters

Work in progress focuses to provide devices supporting the EIB and LON standards. Real-time rendering (radiosity) of the room light settings and shadowing effects – especially as a result of blind movement and rendering different media types on room displays – is one of main topics of the future work.

In addition to the current meeting environment, we intend to integrate several office spaces, classrooms and also home environments in the near future. Basic components of the presented system (CVE-Server and rendering client components) are based on the RenderWare engine. Currently we are working on a sub-version based on OpenSource engines in order to make 3DSim freely available for research purposes. Another issue is the young nature of the UPnP standard. Currently, there are no final versions of UPnP standards for blind shutters and smart projectors available. We proposed a UPnP design for *presentation* devices based on Intel UPnP AV architecture. However, we look forward to get UPnP standards for mentioned devices and to consider possible updates within 3DSim.

8. References

[1] T. Kirste, Smart environments and self-organizing appliance ensembles, in: In Aarts E, Encarnação J L (eds): True Visions, Springer, 2005.

[2] E. Aarts, R. Harwig, M. Schuurmans, Ambient intelligence, McGraw-Hill, Inc., New York, NY, USA, 2002, pp. 235–250.

[3] J. L. Encarnação, T. Kirste, Ambient intelligence: Towards smart appliance ensembles, in: From Integrated Publication and Informations Systems to Virtual Information and Knowledge Environments (2005), Springer.
URL `http://www.informatik.uni-rostock.de/mmis/paper.pdf`

[4] iProntoPro, Prontopro next generation - home theater control panel (2004).
URL `http://www.remotecontrol.philips.com`

[5] A. A. N. Shirehjini, A novel interaction metaphor for personal environment control: Direct manipulation of physical environment based on 3d visualization, in: Computers & Graphics, Special Issue on Pervasive Computing and Ambient Intelligence, Vol. 28, Elsevier Science, 2004, pp. 667–675.

[6] U. Ltd., Ubisense visual developer, www.ubisense.net (2004).

[7] A. A. N. Shirehjini, H. Guddat, S. Noll, N. Schiffner, Platform for distributed multimedia environments supporting arbitrarily nested team structures, Vol. 5241, 2003.

[8] F. Morbitzer, U. Holzmann-Kaiser, S. Noll, Global IT-based Qualification based on an open service platform, Tech. rep., Fraunhofer Institut for Computer Graphics, BMBF 01AK910A (2003).

[9] M. Einhoff, J. Casademont, F. P. i Sapiña, S. Noll, ELIN: A MPEG Related News Framework .

[10] Renderware., http://www.renderware.com.

[11] D. Cook, M. Youngblood, E. Heierman, K. Gopalratnam, S. Rao, A. Litvin, F. Khawaja, Mavhome: An agent-based smart home (2003).
URL `citeseer.ist.psu.edu/cook03mavhome.html`

[12] C. S. Design, Ubiwise, a simulator for ubiquitous.
URL `citeseer.ist.psu.edu/701349.html`