Simulation of Tangible User Interfaces with the ROS Middleware

Stefan Diewald [1], Luis Roalter [1], Andreas Möller [1], Matthias Kranz [2]

[1] Technische Universität München
Distributed Multimodal Information Processing Group
Arcisstraße 21
80333 Munich, Germany
E-Mail: stefan.diewald@tum.de, roalter@tum.de,
andreas.moeller@tum.de

[2] Universität Passau
Lehrstuhl für Informatik mit Schwerpunkt Eingebettete Systeme
Innstraße 43
94032 Passau, Germany
E-Mail: matthias.kranz@uni-passau.de

Author Keywords

TUI prototyping; middleware; virtual TUI; Gazebo; ROS

ACM Classification Keywords

H.5.2. Information interfaces and presentation (e.g., HCI): User Interfaces: Prototyping.

TEI 2014, Feb 16 – 19, 2014, Munich, Germany.

Abstract

Prototyping is an important part in research and development of tangible user interfaces (TUIs). On the way from the idea to a working prototype, new hardware prototypes usually have to be crafted repeatedly in numerous iterations. This brings us to think about virtual prototypes that exhibit the same functionality as a real TUI, but reduce the amount of time and resources that have to be spent.

For that reason, we have created a toolkit that can be used for developing and testing fully functional implementations of a tangible user interface as a virtual device. The entire interaction between the TUI and other hardware and software components is controlled by a middleware, while the human interaction with the TUI can be explored using a 3D simulator and 3D input/output technologies. We argue that by simulating parts of the hardware-software co-design process, the overall development effort can be reduced.

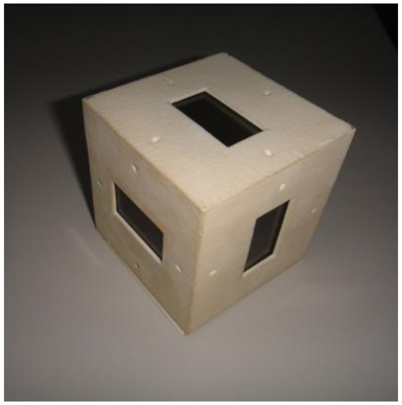

Figure 1. Real TUI with a display on each side. Inside the cube are an accelerometer and a communication device.

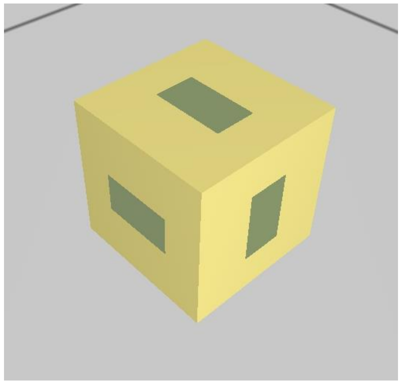

Figure 2. Virtual TUI simulated with Gazebo in ROS. The displays are fully functional. The simulated and the real cube (Fig. 1) look identically. The only difference of the cube is in its colors.

Introduction

The development and evaluation of prototypic tangible user interfaces (TUIs) [3] consumes a lot of effort and time due to iterative design and debug processes on some kind of hardware. Starting from I/O cubes [13], to tabletops [9] and various augmented everyday objects [1], each TUI consists of individual hardware that has often to be built from scratch. In order to reduce the development time for initial prototypes, developers use hardware frameworks, such as *Blades & Tiles* [12], or *Pin & Play* [14]. However, still a lot of work has to be spent on the hardware before a running prototype can actually be used for evaluating user interaction and HCI-related aspects.

Hence, a prototyping approach allowing the simulation of tangible user interfaces at an early stage, e.g. to evaluate novel interaction concepts, before building any kind of hardware could extremely shorten the overall development time. This is especially the case for TUIs that are based on novel hardware that is not yet available or cannot be realized within reasonable expenditure.

In this paper, we introduce a toolkit for TUI simulation that allows shifting the early prototyping process into a high-fidelity 3D virtual environment. That way, shapes of objects for a new TUI and/or new interaction concepts can be evaluated before an actual hardware prototype needs to be built. The proposed toolkit is based on a middleware that can be used for virtual as well as for real TUIs. For that reason, interactions between real *and* simulated components are possible. Moreover, the transition from the virtual to the real prototype does not entail significant changes from software side.

The paper is structured as follows: We first develop and present the requirements for a TUI simulation environment. Based on these requirements, the designed TUI simulation toolkit is introduced. In a comparison between the development process of a real prototype and a virtual prototype, the working method with the proposed solution is presented. Subsequently, we portray the challenges that need to be overcome for being able to assess interactive as well as tangible aspects of a virtual TUI with the proposed toolkit. The paper concludes with a summary of achievements and describes directions for future work.

Prototyping with Virtual Hardware

Since all of the currently available TUI prototyping toolkits and methods need some kind of dedicated real hardware to test the functionality of the TUI, we have considered a prototyping platform that can be used without any real hardware components. Based on the results from Kranz et al. with intelligent and smart environments [7], we created a toolkit that allows a complete virtual representation of a TUI. As presented in Fig. 1 and Fig. 2, we can create high-fidelity virtual prototypes that look almost identical to subsequently developed real prototypes.

Prototyping with Virtual Hardware

A simulation environment for TUIs needs to fulfill several requirements in order to enable a complete evaluation of the system at an early stage and to fulfill the key properties of TUIs as described by Kim and Maher [4]. Since tangible user interfaces are based on the linkage of the virtual and physical domain, it is important that the simulation can simulate any kind of physical object, especially rigid body objects that are commonly used in many activities of daily living (ADL).

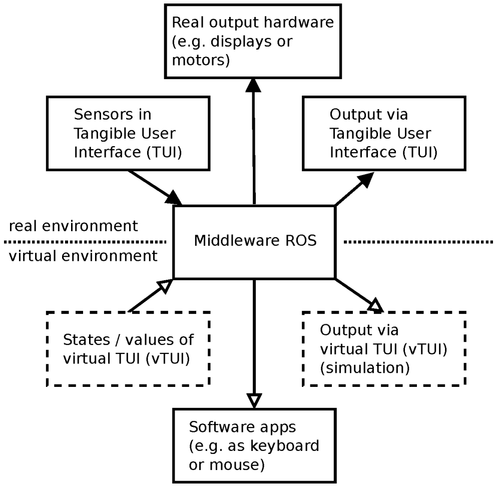
Real output hardware (e.g. displays or motors)
Sensors in Tangible User Interface (TUI)
Output via Tangible User Interface (TUI)
real environment
virtual environment
Middleware ROS
States / values of virtual TUI (vTUI)
Output via virtual TUI (vTUI) (simulation)
Software apps (e.g. as keyboard or mouse)

Figure 3. Up to now, TUIs consisted always of a physical part that only influenced the virtual environment by controlling software applications running on a computer. The proposed solution introduces virtual TUIs and proposes a middleware for connecting elements in real and virtual environments.

A physics engine has to ensure that the virtual objects behave like real physical ones. Besides the support for modeling objects of any arbitrary shape, it should further support realistic textures. Simple geometric shapes as well as complex TUIs should be supported. In order to allow an intuitive evaluation by the user, it has to offer an intuitive 3D interface with the possibility to interact with the simulated objects and to explore the virtual environment.

Another important factor is the connection of the simulation environment to a middleware that can actively support the TUI development. It would be useful to choose one that can be used for the virtual simulation as well as afterwards for a real implementation.

The ROS Middleware as TUI Middleware

Based on former research on the simulation of intelligent environments [7], we have chosen the *Robot Operating System* (ROS) as middleware. ROS is one of the major middleware systems in the domain of robotics. Thus, an advantage of this middleware is that a huge set of drivers and applications (mainly for robotic systems) is already available. ROS has also been used on immobile robots *(ImmoBots)* such as intelligent environments [11]. As argued in this work, intelligent objects behave somewhat like robots as well. This allows us to deal with TUIs as if they were robots, implying that we are able to use any robotic simulation method equally for TUIs.

As presented in Fig. 3, the middleware brings different hardware and software concepts together, creating the possibility to interconnect real TUIs with virtual TUIs (vTUIs).

Using the same toolkit for intelligent environments, robots and TUIs, a common middleware reduces the amount of code that has to be written to establish the communication between these kinds of systems. The middleware already provides us with basic messages and communication protocols to transfer any kind of data between different nodes. For the virtual development process, ROS does not depend on existing hardware. The developers are completely free in designing the communication with other TUIs and *Smart Things*.

3D Simulation of Physical Objects with Gazebo

The 3D simulation is performed with Gazebo [5], a 3D robot simulator. Gazebo is a complete physical simulation of robots including shapes, joints, contacts, collisions, and friction. Gazebo utilizes the 3D framework *OGRE (Object-Oriented Graphics Rendering Engine)*[1] for rendering the environment and objects. It uses the *Open Dynamics Engine (ODE)*[2] library as physics engine that can simulate rigid body dynamics. During the simulation, Gazebo publishes the models' states and behaviors via the ROS middleware's communication infrastructure so that all other nodes can uses these parameters for triggering certain actions.

[1] http://www.ogre3d.org/, last accessed Dec. 11, 2013.

[2] http://ode-wiki.org/wiki/, last accessed Dec. 11, 2013.

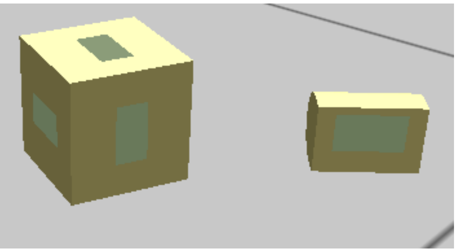

Figure 4. Two virtual TUIs are combined in the simulation through the ROS middleware: an I/O cube and a ubiquitous presence system [6].

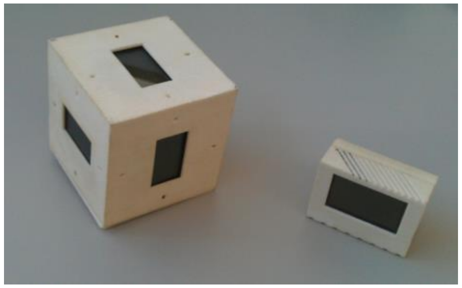

Figure 5. The real counterparts to the virtual I/O cube and the presence system vTUI depicted in Fig. 4. Due to the virtual hardware driver concept of the ROS middleware, the same software can be used for the virtual and the real TUI, only the device drivers have to be replaced.

Gazebo uses *URDF (Unified Robot Description Format)*[3] files for the description of the models. The physical elements for the simulation can be modeled with all common 3D modeling tools, such as *Blender* or *Cinema4D*. By connecting the different objects via different types of joints and defining a mass, inertia and friction values for the elements, the physics engine can simulate the dynamic behavior in a realistic way.

Originally developed as outdoor robotic sensor simulator, a specialty of Gazebo are virtual sensors and actuators that can be assigned to any object in the simulation. Examples of available sensors are cameras, laser scanners, contact switches, force sensors, or inertial measurement units (IMU). It is even possible to simulate a simple battery unit that can be loaded and drained which is another important factor for modeling wireless active components for TUIs. Diewald et al. have presented a more extensive list of available sensors and actuators for Gazebo [2].

Gazebo's functionality can be easily extended through a well-documented API. For example, we have added the support for touch-sensitive virtual displays. Based on this extension, we are able to simulate complete tabletop TUIs or TUIs with embedded displays such as the *Display Cube* [8] (see Fig. 1 and Fig. 2) which is used for comparing the development process of a real TUI to a simulated TUI in a later section of this paper.

For testing and evaluating a TUI system, users can interact with the physical objects through a GUI. They can apply rotational and translational force to any object in the simulation.

Combination of Real and Virtual TUIs

Hardware abstraction allows using the ROS as middleware for real as well as for simulated TUIs. A common abstract hardware layer is used for hiding the actual implementation and for handling the exchange of states and values. Recording and afterwards playing back the exchanged messages allows the middleware to simulate individual objects and larger parts of the environment or setup, without the need of performing input actions repeatedly. By using high-level hardware layer bindings, the injection of hardware messages is possible with little effort. Connecting virtual and real TUIs to the same middleware network transparently joins both virtual and real hardware together. They are indistinguishable for other nodes.

Comparison of the Development Process of a Real and a Simulated Tangible User Interface

Most of the differences between performing rapid prototyping on real hardware and modeling a system virtually emerge during the early prototyping phase. We illustrate the advantage of virtual modeling over conventional modeling by comparing the development process of a cube with a 3D accelerometer and six displays as an example for a TUI. For the real prototype, one needs to choose the sensors and hardware components which fulfill the needs of the developer. This process is time-consuming. Using a virtual TUI, the developer only needs to specify the parameters s/he needs to get from the TUI, such as

[3] http://www.ros.org/wiki/urdf, last accessed Dec. 12, 2013.

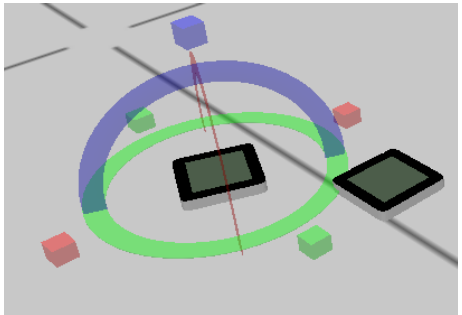

Figure 6. Two *Sifteo cubes*. The left vTUI is currently manipulated through the GUI. Gazebo allows for manipulation in all 6 degrees of freedom by applying force to the object.

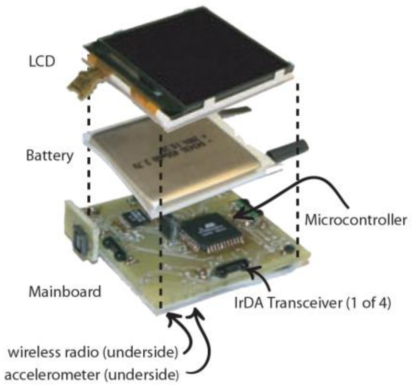
LCD
Battery
Microcontroller
Mainboard
IrDA Transceiver (1 of 4)
wireless radio (underside)
accelerometer (underside)

Figure 7. A view on the complex hardware setup of the real *Sifteo cubes*.
Image Source: Merrill et al. [10]

pose, location etc. In the simulation, the desired actions can be attached to these parameters, so that the simulator can be used for evaluating the model and the designed behavior. This approach helps finding the necessary and proper parameters before the real prototype is implemented. A last goal of such an approach is to separate physical development from the software development.

Hardware development follows often an iterative development approach. It is barely possible to meet all requirements for the actual implementation with the first prototype. Hence, the cycle needs to be traversed multiple times to iteratively improve the prototype. Creating and refining virtual models is usually significantly faster than creating physical prototypes, which allows for faster iterations in the development cycle of prototyping and testing. Often, detected deficiencies after testing result in the creation of an entirely new model. In the virtual representation, the object of the last iteration can much easier be reused by e.g. modifying the shape or texture. Using the ROS for connecting the devices, one can simultaneously develop the software for the virtual device and for the final prototype.

In currently available toolkits and TUIs, a communication protocol to connect heterogeneous hardware has to be created explicitly. The ROS middleware simplifies this process by reducing the code that has to be written just to a device driver that connects the device to an actual ROS node. With increasing processing power of microcontroller platforms, the ROS node could in the future reside in the TUI itself.

More examples of vTUIs are shown in Fig. 4 and Fig. 6 with their real counterparts being depicted in Fig. 5 and Fig. 7 respectively.

Challenges Towards the Simulation of Tangible Aspects

For shifting the whole prototyping phase of a TUI into simulation, users need to be able to evaluate interactive as well as tangible aspects. However, with the currently available input/output systems, the simulation is – for the most part – limited to interactive aspects. This is mainly due to the lack of haptic feedback. Many tangible interactions, such as squeezing a ball or feeling the structure of a surface, cannot be experienced in the 3D environment. This often restricts the evaluation of "look and feel" to the "look" part. Although our experiences have shown that advanced 3D simulation users can also get a good impression of the "feel" component over time, the assumed "feeling" from advanced users cannot replace the user evaluations of TUIs.

Conclusion

The proposed simulation approach based on the ROS middleware has several advantages compared to classical prototyping approaches. For most developers, the time savings will be the most important one. The possibility to simulate tangible user interfaces with new and not yet realizable technologies is another benefit. The effort in terms of costs and time to explore design alternatives is significantly reduced. The interaction between real and simulated devices allows extending available systems with novel devices. Repeatable and easily modifiable test scenarios enable objective comparability of different systems. Time and resources can also be saved for multi-device scenarios, since an object can simply be spawned multiple times in the virtual environment.

We extended the ROS middleware by several components that provide functions necessary for TUIs. For example, we have developed components for display outputs and various sensors, such as a touch sensor. By implementing selected existing, previously published research prototypes of TUIs and a commercial platform, we confirmed the feasibility and function of the proposed approach.

Due to the limitations of the currently available off-the-shelf input/output devices for 3D exploration, the proposed solution is not yet intended to fully replace a physical prototype, but to minimize the time-consuming and costly iterations for creating a working physical prototype. Future work includes finding better suitable interfaces for exploring the 3D scene [15] and manipulating the virtual objects.

References

[1] Chung, K., Shilman, M., Merrill, C., and Ishii, H. OnObject: Gestural Play with Tagged Everyday Objects. *Adj. Proc. UIST '10*, ACM (2010), 379–380.

[2] Diewald, S., Roalter, L., Möller, A., and Kranz, M. Towards a Holistic Approach for Mobile Application Development in Intelligent Environments. *Proc. MUM '11*, ACM (2011), 73–80.

[3] Ishii, H., and Ullmer, B. Tangible Bits: Towards Seamless Interfaces Between People, Bits and Atoms. *Proc. CHI '97*, ACM (1997), 234–241.

[4] Kim, M. J., and Maher, M. L. The Impact of Tangible User Interfaces on Spatial Cognition During Collaborative Design. *Design Studies 29*, 3 (2008), 222–253.

[5] Koenig, N., and Howard, A. Design and Use Paradigms for Gazebo, an Open-Source Multi-Robot Simulator. *Proc. IROS '04*, IEEE (2004), 2149–2154.

[6] Kranz, M., Holleis, P., and Schmidt, A. Ubiquitous Presence Systems. *Proc. SAC '06*, ACM (2006), 1902–1909.

[7] Kranz, M., Möller, A., and Roalter, L. Robots, Objects, Humans: Towards Seamless Interaction in Intelligent Environments, *Proc. PECCS '11*, (2011), 163–172.

[8] Kranz, M., Schmidt, D., Holleis, P., and Schmidt, A. A Display Cube as a Tangible User Interface. *Adj. Proc. UbiComp '05*, (2005).

[9] McAdam, C., and Brewster, S. Multimodal Feedback for Tabletop Interactions. *Proc. ITS '11*, ACM (2011), 274–275.

[10] Merrill, D., Sun, E., and Kalanithi, J. Sifteo Cubes. *Ext. Abstracts CHI '12*, ACM (2012), 1015–1018.

[11] Roalter, L., Möller, A., Diewald, S., and Kranz, M. Developing Intelligent Environments: A Development Tool Chain for Creation, Testing and Simulation of Smart and Intelligent Environments. *Proc. IE '11*, ACM (2011), 214–221.

[12] Sankaran, R., Ullmer, B., Ramanujam, J., Kallakuri, K., Jandhyala, S., Toole, C., and Laan, C. Decoupling Interaction Hardware Design Using Libraries of Reusable Electronics. *Proc. TEI '09*, ACM (2009), 331–337.

[13] Schiettecatte, B., and Vanderdonckt, J. AudioCubes: A Distributed Cube Tangible Interface Based on Interaction Range for Sound Design. *Proc. TEI '08*, ACM (2008), 3–10.

[14] Van Laerhoven, K., Villar, N., Schmidt, A., Gellersen, H.-W., Hakansson, M., and Holmquist, L. Pin&Play: The Surface as Network Medium. *IEEE Commun. Mag. 41*, 4 (2003), 90–95.

[15] Wimmer, R., Holleis, P., Kranz, M., and Schmidt, A. Thracker – Using Capacitive Sensing for Gesture Recognition. *Workshop Proc. ICDCS '06*, (2006), 64–69.