Rapid Prototyping for Wearable Computing

Daniel Spelmezan, Adalbert Schanowski, Jan Borchers
Media Computing Group , RWTH Aachen University
{spelmezan,borchers}@cs.rwth-aachen.de
adalbert.schanowski@rwth-aachen.de

Abstract

We present tools for prototyping and for testing wearable computing applications. The hardware platform consists of a mobile phone and a custom-built box, which can be equipped at runtime with different sensors and actuators. Software libraries for signal processing and classification complement the toolkit. Users without expertise in electronics or in signal processing can quickly create fully functional wearable prototypes that sense human motion and trigger tactile feedback as response to specific postures in real-time.

1. Introduction

Despite advances in hardware platforms for wearable computing, we lack tools that allow non-experts to put their ideas into practice. Sensor platforms require expertise in electronics and in microprocessor programming. Limited sets of built-in sensors restrict application areas. Knowledge in signal processing is mandatory. Moreover, activity and context recognition rely on algorithms that train classifiers in advance. These conditions make it difficult to quickly prototype, test, and deploy wearable applications. We would like end-users to participate and shape the wearable computing domain. A standard Java-enabled mobile phone and an inexpensive open-source electronics prototyping platform serve as initial test bed.

2. Related Work

Only few toolkits allow non-experts to experiment with sensors and actuators. Buechley [2] presented a construction kit for electronic textiles. iStuff Mobile [1] helps explore new ubiquitous computing scenarios using a visual programming metaphor and sensor enhanced mobile phones. Exemplar [3] supports designers in authoring sensor-based interactions but does not support conducting experiments in the field.

3. Toolkit Design and Goals

We designed our toolkit to foster experimentation by lowering the threshold [5] and reducing the time required for creating initial wearable prototypes. Users can exchange sensors, such as accelerometers, bend sensors, or force-sensitive resistors, and actuators, such as vibration motors, buzzers, or LEDs, at run-time. Software libraries offer basic signal processing techniques as well as classification algorithms, which set thresholds on sensor signals to classify posture and activity. This approach does not scale to continuous motion but reliably detects transitions between postures, including different levels of weight distribution, joint flexion, or body inclination, and user activities composed of standing, walking, and running. Moreover, setting thresholds on sensor signals is quick and thus promotes more prototyping cycles compared to techniques that require training of recognition models in advance.

3.1 Hardware Platform

A standard mobile phone acts as host device for our custom-built sensor/actuator box (SensAct). Bluetooth allows the host device to control up to seven boxes concurrently. Figure 1 illustrates the tasks of each unit. Each SensAct box (Figure 2) contains an open-source Arduino BT[1]. A custom-built motor controller for the Arduino provides vibrotactile feedback similar to [4].

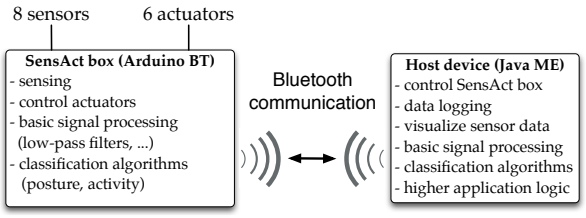

Figure 1: The system architecture of our platform.

[1] http://www.arduino.cc

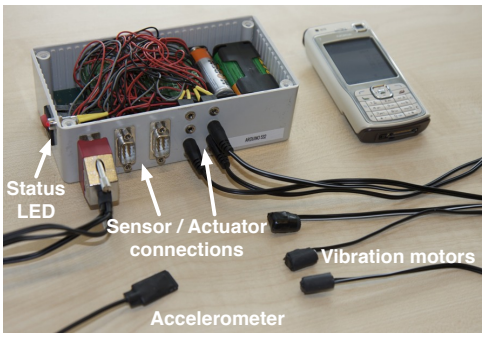

Figure 2. The SensAct box allows replacement of sensors and actuators at runtime.

3.2 Software Tools

We developed software libraries for the SensAct box and the host device. The libraries include algorithms for posture classification and activity recognition used for implementing stand-alone programs. These programs can be freely distributed between several SensAct boxes, if desired. End-users only decide which posture or activity recognition they want to run, and upload the appropriate programs to specific boxes. On the host device, users then link classification results to vibrotactile feedback patterns or to audio and visual feedback produced by the host. All classification algorithms can alternatively run on the host, using raw sensor measurements streamed in real-time from one or from multiple SensAct boxes.

iSenseMobile is the core application that runs on the host device, a Nokia N70 in our case. The host sends control messages to the SensAct boxes. These messages define, for example, the sampling rate for sensors or trigger actuators for rendering vibrotactile feedback. Other control messages start and stop the streaming of raw sensor data or of classification results. Moreover, the host visualizes raw sensor measurements as graphs and as numerical values. This facility supports users in exploring how different body motions affect sensor measurements and allows quickly identifying sensors that got displaced due to movement while testing the prototype in the field.

iSense (Figure 3) is the offline desktop companion to iSenseMobile and supports refinement of initial wearable prototypes. Users can inspect logged sensor measurements or classification results and synchronize these data with video footage recorded during field tests. This tool provides the same signal processing techniques and classification algorithms that are implemented on the mobile platform. iSense offers arithmetic operations that combine sensor signals and it allows to experiment with different filter and threshold parameters for posture and activity recognition.

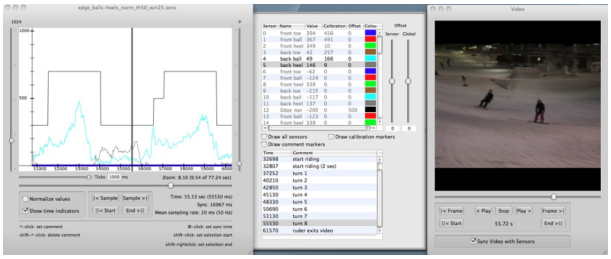

Figure 3. Analyzing sensor recordings and video synchronization with iSense.

4. Conclusions and Future Work

To advance the field of wearable computing, we need to bring new user groups to the table, such as designers, industrials, and artists. These users need appropriate tools that enable them to rapidly prototype and test wearable computing applications with minimal effort. The presented toolkit is a first step in this direction. Our mobile platform allows end-users to experiment in the field with different sensors and actuators on the fly, using a standard mobile phone and inexpensive off-the-shelf components. Classification algorithms yield a posture model, whose results can be linked to real-time feedback. We intend to add new algorithms for posture and motion detection using alternative sensors. Non-experts also need support in designing complex vibrotactile feedback patterns rendered by multiple actuators. Furthermore, end-users expect an interface that allows for quickly defining higher application logic for interactive systems.

5. References

[1] R. Ballagas, F. Memon, R. Reiners, and J. Borchers, "iStuff Mobile: Rapidly Prototyping New Mobile Phone Interfaces for Ubiquitous Computing", *Proc. CHI 2007,* ACM, New York, pp. 1107–1116.

[2] L. Buechley, "A Construction Kit for Electronic Textiles", *Proc. ISWC 2006*, IEEE, pp. 83–90.

[3] B. Hartmann, L. Abdulla, M. Mittal, and S. R. Klemmer, "Authoring Sensor-based Interactions by Demonstration with Direct Manipulation and Pattern Recognition", *Proc. CHI 2007*, ACM, New York, pp. 145–154.

[4] R. W. Lindeman, J. L. Sibert, C. E. Lathan, and J. M. Vice, "The Design and Deployment of a Wearable Vibrotactile Feedback System", *Proc. ISWC 2004*, IEEE, Washington, pp. 56–59.

[5] B. Myers, S. E. Hudson, and Randy Pausch, "Past, Present, and Future of User Interface Software Tools", *ACM TOCHI 7 (1)*, ACM, New York, 2000, pp. 3–28.