

Towards Implicit Interaction by Using Wearable Interaction Device Sensors for more than one Task

Hendrik Witt
TZI Wearable Computing Lab.
University of Bremen
Am Fallturm 1, 28359 Bremen, Germany
hwitt@tzi.de

Holger Kenn
TZI Wearable Computing Lab.
University of Bremen
Am Fallturm 1, 28359 Bremen, Germany
kenn@tzi.de

ABSTRACT

User interaction and context awareness are key issues in today's wearable computing research. Context-aware wearable devices typically use a multitude of special purpose sensors for this. In this paper, we show for a simple test case that even simple sensors in a wearable human computer interaction device can be used for robust context detection. Furthermore, we describe in detail the recognition of different activities and discuss the results.

Categories and Subject Descriptors

H.4 [Information Interfaces and Presentation]: Miscellaneous; B.4 [Input/Output and Data Communication]: Input/Output Devices

General Terms

Wearable Computing

Keywords

Implicit Interaction, Context Recognition, Wearable Computing, Sensors, Activity Recognition

1. INTRODUCTION

One idea behind wearable computers is that they are continually worn to support their users during work. In particular, wearable computing is often tied up with hands free operation that allows users to focus their attention on primary manual tasks they have to perform. To support this, several challenges related to human-computer interaction (HCI) arise.

Today, the availability of sensor technology (ranging from temperature sensors to more complex acceleration sensors) and the fact that non-computer experts use wearable computers in real working environments require the development of wearable user interfaces that do not bother users with complex interaction sequences for getting application

support but provide them with an intuitive and easy to use interaction.

Interaction techniques can be distinguished into *explicit* and *implicit* interaction. Explicit interaction is performed if the user tells the computer what to do. Examples for explicit interactions are command-line interfaces, WIMP user actions like point-and-click, gestures or speech input. In contrast, implicit HCI "is an action, performed by the user that is not primarily aimed to interact with a computerized system but which such a system understands as input" [15], i.e. the computer understands what the user is doing and knows the consequences.

To allow the user to primarily focus on real-world tasks, the wearable computer obviously needs to be in the background which makes implicit interaction the preferred interaction technique. Because implicit HCI can most likely be reached by knowing the environment conditions and current user activities, sensors are often used to recognize such context information. Context recognition with sensors is a popular research topic where many work has already been published (see e.g. [6, 11, 16]). However, this work often demands users to wear special sensors, cables, hubs, etc. that are specifically designed and optimized for context recognition and typically rather expensive. Additionally, interaction devices are still needed for explicit interaction with the actual application. These circumstances currently neither lead to a better acceptance of wearable computing, nor are they unobtrusive to use as claimed at different points in the community [18, 14].

However, even for usability aspects, the development of unobtrusive wearable systems is a key issue. Thus, research questions such as, how implicit HCI can still be achieved even if no additional special-purpose sensors are available but only explicit interaction devices with their built-in sensors are used, arise.

This paper investigates the question above and presents an approach how to enable implicit HCI for a wearable maintenance application only with use of an explicit interaction data glove device and no additional sensors that could reduce unobtrusiveness of the system or freedom in movement of the user. In connection with this, our example will show how basic context recognition is done with only the built-in sensors of an interaction device and what results have been achieved.

1.1 Outline

The organization of the paper is as follows. Section 2 gives an overview of the range of available interaction devices and their capabilities as well as related work in the field. Section 3 discusses the general approach of using interaction device sensors also for context recognition. In connection with this, the experiment setup used in this paper is introduced as well as the interaction hardware and its built-in sensors. Section 4 shows how context recognition is performed with the sensors of the interaction device and presents the results. Section 6 concludes the paper and section 7 is pointing out some future work.

2. RELATED WORK

There is a broad range of domain specific interaction devices for wearable computers. Beside text-input oriented devices designed for wearable computing such as Twiddler2 [7] or FrogPad [5] more complex devices for wearable computer interaction were developed. Those devices usually utilize set of specific sensors to recognize the users input, e.g. by gestures.

The Winspect [3] data glove allows one-hand free interaction by performing list navigation through hand-rotations and item selections by pushing together two fingers of the data glove. To achieve this functionality a tilt-sensor and different micro-click buttons are used. GestureWrist [14] is a wrist-watch type input device. It recognizes hand gestures by analyzing finger and forearm movements that can be mapped to a set of application control commands. To recognize movements an acceleration- and capacitance-sensor is used. In [4] two low power cameras that capture black and white images and a distance-sensor have been combined to build the Fingermouse, a wearable mouse input device that is controlled by finger movements in front of the body. In [19] a multipurpose sensor board for building Bluetooth enabled interaction devices was presented.

Beside this interaction oriented hardware there are specific sensor boards that contain a different number of sensors. Examples include the Smart-Its [8] or low-power sensor boards described in [2]. Although such sensor boards can be used in different setups they are often used for context recognition only. But as shown with the FreeDigiter [13], a sensor board with different sensors can also be used to build interaction devices. However, also the FreeDigiter focuses only on interaction and does not combine interaction and context recognition with sensors available on the sensor board. Context recognition by using special-purpose sensor devices has often been demonstrated, e.g. [6, 11, 16]. However, these approaches have in common that special sensor devices have been used to track user activities. Often these sensor devices also need lengthy setup and tuning procedures to yield optimal results.

3. USING INTERACTION DEVICE SENSORS FOR CONTEXT RECOGNITION

In general, as mentioned before, explicit interaction devices are typically equipped with different sensors that are only needed during explicit user interaction. But while sensors are idle they can be used for other tasks like recognizing a user's current activity or environment conditions.

When looking at explicit interaction in wearable computing it seems to be true that interaction device sensors are of-



Figure 1: Layer-based wearing concept of the used data glove

ten in an idle status. Usually, the reason is that wearable computers are commonly used to support users *during* their work so that controlling a wearable computer becomes in the majority case a secondary task. Nevertheless, recognizing activities of a primary task by using interaction device sensors depends on the particular interaction device and application domain.

In the remainder of this paper, we will present an approach of using available interaction device sensors of a data glove without any special-purpose sensor setup or tuning to recognize context during sensor idle time's.

3.1 Experiment Setup

To demonstrate our idea of using interaction device sensors during their idle time, a scenario (similar to [12]) was chosen where different user activities should be recognized.

In the experiment we used an application in style of the Winspect [3] application. It supports workers during crane maintenance through a list oriented graphical user interface operated by a special data glove. In our case the data glove was used as explicit interaction hardware for list navigation and item selection, too. Additionally, the data glove sensors are now also used during their idle time to detect user activities in the primary task. With the built-in tilt-sensor we tried to recognize user activities that are useful in crane maintenance, e.g. *screw driving* or *hammering*, to verify the feasibility of our approach.

3.2 Used Interaction Hardware

The hardware platform used in our experiments is based on a small multipurpose sensor-board with wireless communication support [19]. The current version of our data glove features a special wearing concept and is capable of handling different sensor configurations.

The wearing concept is based on three different gloves that build the actual data glove. By changing the outer glove the device can be adapted to specific application domains whereas the inner glove can be used for hygiene aspects. Figure 1 shows the data glove used in our experiments and its layer-based wearing concept.

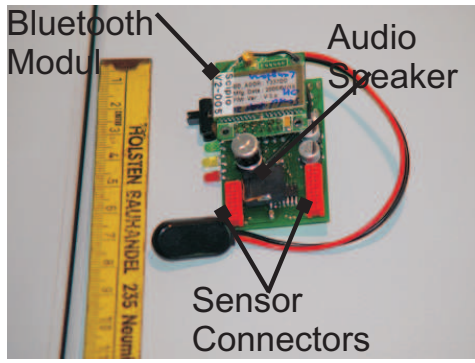


Figure 2: Sensor board hardware

The hardware platform integrated into the data glove to build the explicit interaction device covers two main aspects:

- **Controller Hardware**

The core of the data glove hardware is a sensor-board with a PIC Micro Controller Unit (MCU). It features a 7 channel A/D-Converter, a PWM generator, a number of digital I/O pins, and a Bluetooth module. Sensor data is transmitted wireless over a lightweight proprietary protocol. Figure 2 shows an image of the hardware including its dimension.

- **Built-in sensors**

For interaction the data glove is equipped with a tilt-sensor, buttons and a RFID reader. The tilt-sensor is a two axes liquid tilt-sensor with a measurement frequency of up to 400Hz. Moreover, three trigger buttons have been integrated. They are positioned into the tip of the index, middle and ring finger. Finally, the sensor board features a visual and acoustic feedback system that consists of three different colored LED's and a small piezo speaker for audio signals.

4. CONTEXT RECOGNITION DURING IDLE TIME

Obviously, recognizing context is strongly related to the capabilities of available sensors in a given situation. Although only measurements of a two axes liquid tilt-sensor were used for analyzing user activities in our experiment and no special setup or tuning of the sensors has been made, we were still able to recognize the activities *screw driving*, *hammering*, and *no activity* as we will show later on.

Lukowicz et al. already demonstrated in [12] the usefulness of recognizing activities such as drilling, hammering, sawing etc. in specific application domains. However, compared to their work, that uses accelerometers and microphones to recognize activities, here only the low-cost two axes liquid tilt-sensor with limited capabilities is used. Though, this limited tilt-sensor is still sufficient to do basic context recognition for implementing implicit interaction capabilities into our maintenance application.

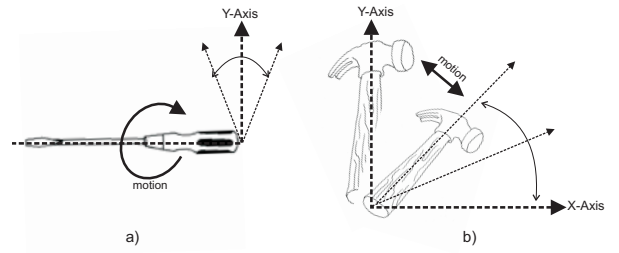


Figure 3: Principle motion difference analysis a) screw driving b) hammering

4.1 Sensor Data Analysis

In principle the activities screw driving and hammering are relatively easy to detect. The reason is that they can be easily distinguished in terms of their characteristic motion, i.e. their rotation around different axes. Analyzing the essential rotations around the x- and y-axis during screw driving and hammering activities, it can be stated that they behave complementary to each other. Under ideal conditions, one axis should always remain constant while the other does not (see figure 3).

By using real world sensor data this situation only remains valid in general, but raw data differs significantly from data acquired under ideal conditions. One reason is signal noise due to sensor inaccuracy. Particularly, in our case where we used a liquid tilt-sensor without any tuning, filtering signal noise becomes even more important. A main reason is liquid splashing between measuring heads inside the sensor during motion. This finding is especially important when analyzing extreme motions and orientations such as those in hammering where the liquid inside the sensor splashes several times uncoordinated back and forth between the two measuring heads for x- and y-axis values. Figure 4 shows a snapshot of sensor signals gathered from the tilt-sensor while performing the two different activities with inactivity phases in between.

As figure 4 shows it is in principal possible to identify both activities visually, although considerable signal noise is existent. Particularly, on the transition of activities an exact determination of the beginning and ending becomes difficult.

Due to a sampling rate of 2000 Hz the sensor board acquires data from the sensors and a specified time constant of the tilt-sensor with less then 100 ms for high accuracy, over sampling is given in our experiments. However, as we are not interested in a highly accurate absolute angle, but in the sequence and relation of the peaks of the angular signal, we used preprocessing to overcome the problem. The preprocessing was done by applying a simple filter algorithm filtering sequentially x and y sensor values when remaining equal. It turned out that by applying this filtering procedure, the amount of sensor values is reduced significantly which is positive in terms of performance for following clas-

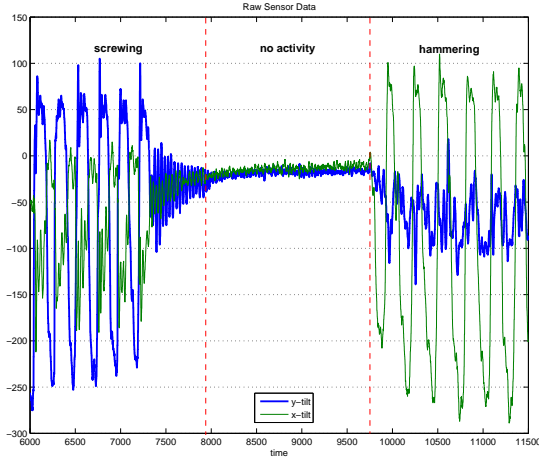


Figure 4: Raw sensor data (snapshot)

sification algorithms. However, after applying the filter a drawback is that the original time intervals have been modified making any feature extraction relying on exact time intervals difficult. But it will be shown that this is neither problematical for our feature set used nor for the results of the overall recognition rate.

4.2 Finding the Feature Set

After having discussed the raw sensor data analysis and described the appliance of filters to the raw sensor data the next step is to identify the feature set used in the following classification process to reach appropriate recognition results.

Although in our scenario limited computation resources are not considered as a key issue we nevertheless tried to avoid the selection of computationally complex features being aware of power and performance limitations of wearable devices [17].

But good recognition rates can already be achieved by selecting a set of relatively simple features which are shown in table 1.

Table 1: Features used for Classifiers

Feature	Feature Name	Window Size n measurements
F_1	Zero Crossing Rate	10
F_2	Mean Zero Crossing Rate	10
F_3	Mean	10
F_4	Standard Deviation	10
F_5	Variance	10
F_6	Mean Gradient	10

This set of features selected can be classified as a set of standard features successfully applied to different context recognition problems in wearable computing.

The feature set was applied to the raw sensor data on the

Table 2: Confusion matrix of the C4.5 algorithm for training data

a	b	c	← classified as	correctly classified
518	24	25	a = no activity	91.36%
34	1209	0	b = screw driving	97.26%
27	1	822	c = hammering	96.71%

same window sizes. As table 1 shows, we applied all features F_1 to F_6 to a window size of 10 measurements which is approximately equivalent to a time interval of 2 seconds. Since our experiments were carried out by continuously performing the different activities with breaks of doing some other movements between two activities, the features were applied without paying attention to start or end time of the activities, i.e. no segmentation has been considered.

4.3 Classification

For demonstrating the classification we selected only a single algorithm for classifying activities and did not evaluate recognition rates of different classification algorithms. Thus, we used a C4.5 classifier, as it is also used by others for context recognition, e.g. [9, 10].

To carry out our recognition experiments we used the WEKA environment [20] and its built-in implementation of the C4.5 algorithm. To record training and test sets we performed continuously screw driving and hammering activities with inactivity phases in between. The activities were performed in sequential order over a certain period of time with 3 different persons. From each person 4 different data sets were recorded. The training set for the classifier was composed from 2 different data sets of each test person. The remaining 2 data sets of each person have then been given to the classifier as test sets. We assumed for our recognition rate evaluation that the appearance of different activities are equally probable. Therefore, the selection of the overall recognition error seemed to be a good metric for evaluation purpose.

As derived from the confusion matrix given in table 2, we achieved an overall recognition rate of correctly classified instances of 95.83%. Although this particular recognition rate has been achieved by using training and test sets in cross validation mode, the result for applying other test sets will still remain quite valuable as we will show next.

Comparing the recognition rate from the 3 different test persons, as depicted in figure 5 (every two successive test data sets belong to a single test person), an overall recognition rate of 89.47% can be computed. This mean a loss in recognition accuracy when using test sets from different persons for the classification. And also the recognition rate for the activity screw driving with 87.59% on the average is lower compared to the cross validation results in table 2. However, this finding is not very surprising, because of two different factors. First, there are obviously small differences in the way different persons perform tasks like screw driving or hammering although they seem to be only performable in one way, e.g. left- and right-handed persons. Second, from our analysis we found that overall recognition rates decrease mainly because of error-prone classification of inactivity in-

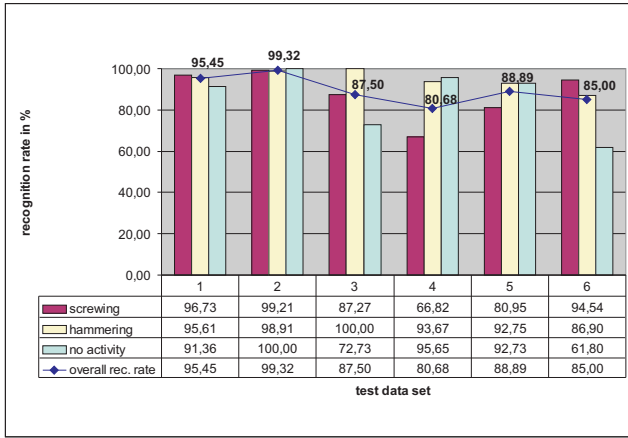


Figure 5: Overall recognition rate of test persons

tervals between activities. As the main driver for this problem we identified the position of the test person’s hand during inactivity which has strong impact on the behavior of the tilt-sensor. When looking at the very same test person, the classification errors produced are reasonable. Thus, we assume that the same person behaves almost identically during the same activities and particularly in inactivity phases which turned out as most error-prone, whereas two different test persons do not.

While investigating the influencing factors for an increasing or decreasing overall recognition rate, we also looked at the C4.5 classifiers tree model built from our provided features. There we found that the tree model predominantly includes features based on the x-axis values of our tilt-sensor whereas y-axis based features are rare. This is particularly interesting, because the findings of our general motion analysis on the difference of screw driving and hammering, which we carried out before we applied classifiers to our data sets, brought contradictory findings compared to the actual C4.5 classifier behavior (see section 4.1). The reasons can be various, from dependencies on the feature set provided over the selection of the classification algorithm used to the sensor setup used. However, we can state that for our scenario results are sufficient to implement implicit interaction. And also compared to past work results in activity recognition which demonstrated recognition rates from 85% to 95% [1] our results are comparable.

5. CHANGING BETWEEN INTERACTION AND CONTEXT RECOGNITION

After successfully implementing context recognition with available interaction device sensors during their idle time, another problem arises that is the question of how to determine when sensors are idle, i.e. how to distinguish between explicit and implicit interaction modes.

To answer such questions different approaches are possible that range from simple to complex:

1. Explicit Context-Switch

Giving the control to the user is obviously the simplest way to solve the problem. However, for an im-

plementation different approaches are thinkable that affect the usability of the entire system positively or negatively. Assume for example the introduction of a gesture specifically matched to the application domain to switch interaction mode, e.g. performing a virtual stroke from left to right with the data glove. It might be a good possibility, as the gesture is natural for a user and has a well defined semantic, e.g. “stopping”, “changing” etc. in a real world environment. But other approaches can also be useful or are easier to implement. Thus, in our scenario we took the approach to simply switch between the two interaction modes by defining a certain action trigger to be pushed for switching the modes back and forth. In our special case, the needed action trigger was already integrated in the data glove interaction hardware.

2. Implicit Context-Switch

Trying to do the context switch between the interaction modes by implicit interaction is more ambitious. Although implicit context switching seems similar to recognizing specific gestures mentioned in the previous paragraph, here gestures become more like behaviors. Recognizing such behaviors needs to evaluate sensor data over larger periods compared to gestures which are typically short. Moreover, the appearance of a behavior can be arbitrary at any point in time during application execution. Additionally, the selection of suitable behaviors for a certain application can become difficult.

In our scenario “walking” would not be an appropriate behavior for context switching. Although it implies that the user is currently not maintaining any engine, recognizing this activity can still provide useful information for implicit interaction anyhow and thus should not cause a switch from implicit to explicit mode. In any case, recognizing a certain behavior for switching context requires continuous context recognition. This is difficult if only one sensor is available and we stick to our idle sensor usage approach strictly. By using interaction devices with several built-in sensors, the set of temporarily idle sensors can be used for recognizing behaviors even if the set is changing over time.

The problem of having only one sensor available for explicit interaction and context recognition can also be managed by other approaches. However, this will require a modification of our suggested approach so that sensors are usable all the time and not only when idle. But this is currently beyond the scope of the paper.

6. CONCLUSION

In this paper we showed that basic activity recognition with non-special-purpose sensors of interaction devices is possible. We described in detail how built-in sensors of an explicit interaction device can be used to gather additional context information without additional sensors or special sensor tuning.

The approach of using wearable interaction device sensors for more than one task, i.e. during their idle time, is feasible. Considering crane maintenance, our approach is not only an optimization approach, but can reduce obtrusiveness of wearable applications as the amount of necessary sensors is not increased to implement activity recognition.

It was shown that domain relevant activities such as screw driving or hammering can be recognized even with very limited and difficult to handle sensors often found in devices originally designed as explicit interaction devices only. However, for sensors with limited capabilities, like the used tilt-sensor, different side effects can occur that make the recognition of activities more complicated compared to a situation where a specific special-purpose sensor is responsible for recognizing a certain context only. But with an overall recognition rate of 89.47% valuable results are still achievable and correspond to results in past work on activity recognition [1]. Although in our scenario semi-complex activities were sufficient, other scenarios might require extensions.

7. FUTURE WORK

In the future we will substitute the data glove with the very limited tilt-sensor by other available wearable interaction devices for testing the boundaries of our approach. Furthermore, we want to find a solution how an implicit context-switch can be implemented that distinguishes between an explicit interaction mode and an implicit mode. Finding such mechanism might involve further usability experiments to find suitable gestures or other strategies.

In connection with this, we will investigate how implicit user interaction can be used in wearable computing to build adaptive user interfaces that are self-configuring towards a current situation.

Acknowledgment

This work has been partly funded by the European Commission through IST Project wearIT@work: Empowering the Mobile Worker by wearable Computing (No. IP 004216-2004).

8. REFERENCES

- [1] L. Bao and S. S. Intille. Activity recognition from user-annotated acceleration data. In *Pervasive*, pages 1–17, 2004.
- [2] N. B. Bharatula, M. Stäger, P. Lukowicz, and G. Tröster. Empirical study of design choices in multi-sensor context recognition systems. In *2nd International Forum on Applied Wearable Computing - IFAWC*, pages 79–83, 2005.
- [3] M. Boronowsky, T. Nicolai, C. Schlieder, and A. Schmidt. Winspect: A case study for wearable computing-supported inspection tasks. In *5th International Symposium on Wearable Computers (ISWC '01)*, volume 5, pages 163–164, 8–9 October 2001.
- [4] P. de la Hamette, P. Lukowicz, G. Tröster, and T. Svoboda. Fingermouse: A wearable hand tracking system. In *4th International Conference on Ubiquitous Computing (UbiComp '02)*, pages 15–16, October 2002.
- [5] FrogPad Inc. Frogpad wearable keyboard, <http://www.frogpad.com>, 2005.
- [6] A. R. Golding and N. Lesh. Indoor navigation using a diverse set of cheap, wearable sensors. In *ISWC '99: Proceedings of the 3rd IEEE International Symposium on Wearable Computers*, page 29, Washington, DC, USA, 1999.
- [7] Handykey Corporation. Twiddler2 wearable coording keyboard, <http://www.handykey.com>, 2005.
- [8] L. E. Holmquist, H.-W. Gellersen, G. Kortuem, A. Schmidt, M. Strohbach, S. Antifakos, F. Michahelles, B. Schiele, M. Beigl, and R. Mazé. Building intelligent environments with smart-its. *IEEE Computer Graphics and Applications*, 24(1):56–64, 2004.
- [9] S. S. Intille, L. Bao, E. M. Tapia, and J. Rondoni. Acquiring in situ training data for context-aware ubiquitous computing applications. In *CHI '04: Conference on Human factors in computing systems*, pages 1–8, New York, NY, USA, 2004. ACM Press.
- [10] H. Junker, P. Lukowicz, and G. Tröster. Sampling frequency, signal resolution and the accuracy of wearable context recognition systems. In *ISWC*, pages 176–177, 2004.
- [11] K. V. Laerhoven and H.-W. Gellersen. Spine versus porcupine: A study in distributed wearable activity recognition. In *ISWC*, pages 142–149, 2004.
- [12] P. Lukowicz, J. A. Ward, H. Junker, M. Stäger, G. Tröster, A. Atrash, and T. Starner. Recognizing workshop activity using body worn microphones and accelerometers. In *Pervasive Computing: Proceedings of the 2nd International Conference*, pages 18–22. Springer-Verlag, Apr. 2004.
- [13] C. Metzger, M. Anderson, and T. Starner. Freedigiter: A contact-free device for gesture control. In *ISWC*, pages 18–21, 2004.
- [14] J. Rekimoto. Gestur wrist and gestur pad: Unobtrusive wearable interaction devices. In *5th International Symposium on Wearable Computers (ISWC '01)*, volume 5, pages 21–27, 8–9 October 2001.
- [15] A. Schmidt. Implicit human computer interaction through context. *Personal and Ubiquitous Computing*, 4(2/3), 2000.
- [16] D. P. Siewiorek, A. Smailagic, J. Furukawa, A. Krause, N. Moraveji, K. Reiger, J. Shaffer, and F. L. Wong. Sensay: A context-aware mobile phone. In *ISWC*, pages 248–249, 2003.
- [17] T. Starner. The challenges of wearable computing: Part 1. *IEEE Micro*, 21(4):44–52, 2001.
- [18] T. Starner. The challenges of wearable computing: Part 2. *IEEE Micro*, 21(4):54–67, 2001.
- [19] H. Witt, R. Leibrandt, A. Kemnade, and H. Kenn. Scipio: A miniaturized building block for wearable interaction devices. In *3rd International Forum on Applied Wearable Computing (IFAWC)*, Bremen, Germany, March 15–16 2006.
- [20] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, San Francisco, 2000.