First publ. in: Proceedings of the 2nd International Conference on Tangible and Embedded Interaction 2008, Bonn, Germany, February 18-20, 2008, pp. 11-14

Gesture Recognition with a Wii Controller

Thomas Schlömer,[1] **Benjamin Poppinga,**[1] **Niels Henze,**[2] **Susanne Boll**[1]

[1]University of Oldenburg
firstname.lastname@uni-oldenburg.de

[2]OFFIS Institute for Information Technology
firstname.lastname@offis.de

ABSTRACT
In many applications today user interaction is moving away from mouse and pens and is becoming pervasive and much more physical and tangible. New emerging interaction technologies allow developing and experimenting with new interaction methods on the long way to providing intuitive human computer interaction. In this paper, we aim at recognizing gestures to interact with an application and present the design and evaluation of our sensor-based gesture recognition. As input device we employ the Wii-controller (Wiimote) which recently gained much attention world wide. We use the Wiimote's acceleration sensor independent of the gaming console for gesture recognition. The system allows the training of arbitrary gestures by users which can then be recalled for interacting with systems like photo browsing on a home TV. The developed library exploits Wii-sensor data and employs a hidden Markov model for training and recognizing user-chosen gestures. Our evaluation shows that we can already recognize gestures with a small number of training samples. In addition to the gesture recognition we also present our experiences with the Wii-controller and the implementation of the gesture recognition. The system forms the basis for our ongoing work on multimodal intuitive media browsing and are available to other researchers in the field.

Author Keywords
tangible user interfaces, gesture recognition, Wiimote

ACM Classification Keywords
[H.5.2 User Interfaces]: Haptic I/O

INTRODUCTION

In recent years, we find more and more affordable hardware that allows the development of multimodal user interfaces. Recently one of these interfaces is the so called Wiimote [1], the device that serves as the wireless input for the Nintendo Wii gaming console. The Wiimote can detect motion and rotation in three dimensions through the use of accelerometer technology. Separating the controller from the gaming console, the accelerometer data can be used as input for gesture recognition. In our work, we address the recognition of gestures for new multimodal user interfaces. We are interested in recognizing arbitrary gestures of users that are performed by one hand. We choose the Wiimote as our input device for its ease of use, the hardware price and the design.

Figure 1. The Wii Controller (Wiimote).

Accelerometer-based gesture recognition has been discussed in many publications, most prominently in those by Hofmann et al. in [4] and most recently in those by Mäntyjärvi et al. in [6] and [7]. Like the commercial work by AiLive Inc. (cf. [2]) we aim for a system allowing the training and recognition of arbitrary gestures using an accelerometer-based controller. In doing so we have to deal with spatially as well as temporally variable patterns and thus need a theoretical backbone fulfilling these demands. We transfer the methods proposed in [6, 7] who are using special hardware for 2D gesture recognition to the consumer hardware of the Wiimote and recognize 3D hand gestures. With the controller the user can make her own, closed gestures and our gesture-recognition aims at a Wii-optimized recognition. Our components as well as the filtering process is specifically targeted to the Wiimote. With this paper we also share our experiments and the resulting implementation with other researchers.

CONCEPT

In gesture recognition using an acceleration sensor, gestures are represented by characteristic patterns of incoming signal data, i.e. vectors representing the current acceleration of the controller in all three dimensions. Hence, we need a system pipeline preparing and analyzing this vector data in order to train as well as recognize patterns for distinct gestures. For this purpose we revert to the classic recognition pipeline shown in Figure 2. It consists of the three main components *quantizer*, *model* and *classifier*.

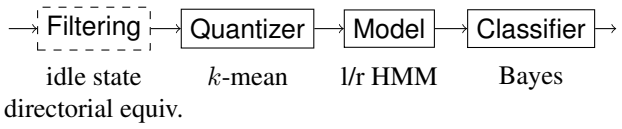


Figure 2. Components of our recognition system. We use a total of two filters before following a traditional pipeline like [7]. The quantizer applies a common k-mean algorithm to the incoming vector data, for the model a left-to-right hidden Markov model is used and the classifier is chosen to be a bayesian.

As an accelerometer constantly produces vector data we first need a *quantizer* clustering the gesture data. Here, a common k-mean algorithm (cf. e.g. [5]) is applied. The *model* has been chosen to be a discrete hidden Markov model since it offers a long history in the service of gesture recognition and promises to deliver reliable results for patterns with spatial and temporal variation (cf. e.g. [4]). The remaining component is a classic Bayes-classifier. In addition to these main components we establish two filters for pre-processing the vector data, an "idle state" and a "directorial equivalence" filter. Both serve the purpose to reduce and simplify the incoming acceleration data.

As we want optimize the HMM for the task of an accelerometer based gesture recognition we select the reference gestures shown in Figure 3 during the following tests and evaluations. With regard to the components of the classic gesture recognition approach in Figure 2 we identify three components for analysis and improvement: vector quantization, the concrete hidden Markov model and filters.

Vector quantization

Like other acceleration-sensors the one integrated into the Wiimote delivers too much vector data to be put into a single HMM. In order to cluster and abstract this data the common k-mean algorithm is applied with k being the number of clusters or codes in the so-called codebook. Since k must be determined empirically we decided to conduct tests to find a codebook size delivering satisfying results and as we are evaluating true 3D gestures we cannot rely on previous results by Mäntyjärvi et al. who empirically identified $k = 8$ for gestures in a two-dimensional plane. However, we adopt their idea of arranging the 8 cluster centres on a circle by extending it to the 3D case. Instead of distributing the centres uniformly on a two-dimensional circle we put them on a three-dimensional sphere, intersecting two circles orthogonal to each other (cf. Figure 4). Consequently this leads to $k = 8 + 6 = 14$ centres. For comparison, we also enhanced the spherical distribution to include another four centers on the XZ-plane and thus gain $k = 18$ cluster centres. The radius of each circle/sphere dynamically adapts itself to the incoming signal data.

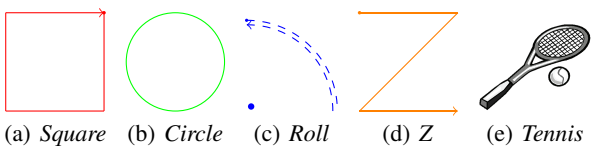


Figure 3. Reference Gestures. The gesture in (b) does not show a starting point because the gesture might start anywhere on the circle. Gesture (c) describes a $90°$-roll around the z-axis (forth and back) and gesture (e) symbolizes the serve of a regular tennis match: raising the controller and then rapidly lowering it in a bow-curved manner.

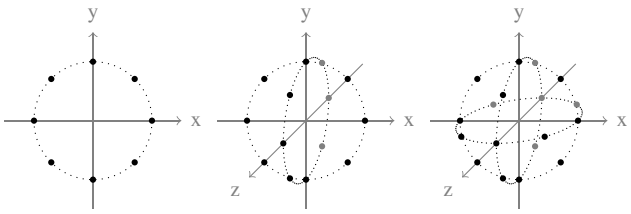


Figure 4. Distribution of the cluster centres during quantization for $k \in \{8, 14, 18\}$. We extrapolate from the two-dimensional case for $k = 8$ with a simple circular distribution to a three-dimensional using two orthogonal circles for $k = 14$ to another three-dimensional using three orthogonal circles for $k = 18$ and evaluate which of them results in the most reliable behavior.

We conducted a small evaluation comparing the three settings shown in Figure 4 using the reference gestures from Figure 3. We found that for $k = 8$ the recognition process cannot clearly differentiate between the five reference gestures. Since the gestures explore all three dimensions, laying out the centres on a two dimensional plane is not sufficient. With $k = 14$ the probabilities for the respective gestures improve as expected and the model can clearly distinguish between the five gestures. Using $k = 18$ results in "overtrained" HMMs, do not improve the probabilities and slow down performance. Consequently we choose $k = 14$ with the distribution shown in Figure 4(b).

Hidden Markov Model

In our system a HMM is initialized for every gesture and then optimized by the Baum-Welch algorithm (cf. [3]). However, there are two competing HMM instances we might revert to: a left-to-right vs. an ergodic. While [4] claims that both approaches deliver comparable results, [9] states that a left-to-right model is clearly to be preferred when the incoming signals change over time. We implemented both models and ran a test to determine which model better suits our needs. Table 1 shows the results for both possible instances and a varying number of states. Our results confirm the statement by [4] that no instance is significantly better than the other as well as the statement by [8] that the influence of the

	Square	*Circle*	*Roll*	*Z*	*Tennis*
5 states					
left-to-right	$9.87 \cdot 10^{-24}$	$2.34 \cdot 10^{-18}$	$2.56 \cdot 10^{-20}$	$6.79 \cdot 10^{-65}$	$4.1 \cdot 10^{-61}$
ergodic	$8.11 \cdot 10^{-20}$	$5.86 \cdot 10^{-23}$	$1.96 \cdot 10^{-21}$	$8.86 \cdot 10^{-72}$	$7.42 \cdot 10^{-67}$
8 states					
left-to-right	$2.28 \cdot 10^{-22}$	$7.86 \cdot 10^{-16}$	$7.42 \cdot 10^{-21}$	$6.24 \cdot 10^{-62}$	$3.17 \cdot 10^{-56}$
ergodic	$9.45 \cdot 10^{-26}$	$6.93 \cdot 10^{-23}$	$2.38 \cdot 10^{-21}$	$1.11 \cdot 10^{-71}$	$9.56 \cdot 10^{-67}$
10 states					
left-to-right	$1.49 \cdot 10^{-21}$	$1.59 \cdot 10^{-14}$	$4.6 \cdot 10^{-21}$	$2.63 \cdot 10^{-60}$	$5.3 \cdot 10^{-54}$
ergodic	$1.02 \cdot 10^{-25}$	$7.55 \cdot 10^{-23}$	$2.64 \cdot 10^{-21}$	$1.25 \cdot 10^{-71}$	$1.09 \cdot 10^{-66}$

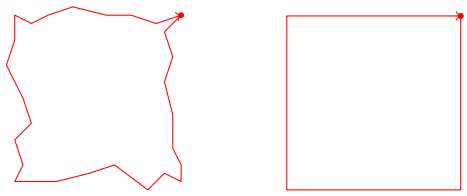
Table 1. Model probabilities for left-to-right and ergodic HMM with varying number of states. Our evaluation confirms the statement by [4] that neither the number of states nor the concrete HMM instance influence the results all too much.

number of states is rather weak. In the end we chose our model to be a left-to-right HMM with 8 states for convenience.

Filtering

Before the actual recognition process our system applies two filters to the vector data establishing a minimum representation of a gesture before being forwarded to the HMM for training or recognition. The first filter is a simple threshold-filter eliminating all vectors which do not contribute to the characteristic of a gesture in a significant way, i.e. all $\vec{a}$ for which $|\vec{a}| < \Delta$. We call this filter the "idle state filter" and determined Δ to a value of $\Delta = 1.2g$, g being the acceleration of gravity. The second filter is called "directorial equivalence filter" and eliminates all vectors which are roughly equivalent to their predecessor and thus contribute to the characteristic of a gesture only weakly. Vectors are omitted if none of their components $c \in \{x, y, z\}$ is all too different to the corresponding component of their predecessor, i.e. if $|\vec{a}_c^{(n)} - \vec{a}_c^{(n-1)}| \leq \epsilon$ for all c. ϵ was chosen to be 0.2 in the case of the Wiimote.

As Figure 5 shows, this filter would ideally lead to just four characteristic acceleration vectors in the case of the gesture "square". In addition, Figure 6 demonstrates the reduction of the number of vectors for every reference gesture after applying both filters.



(a) Before filtering (b) After filtering

Figure 5. Effect of the directorial equivalence filter. Applying it would ideally lead to just four acceleration vectors for the gesture *Square*.

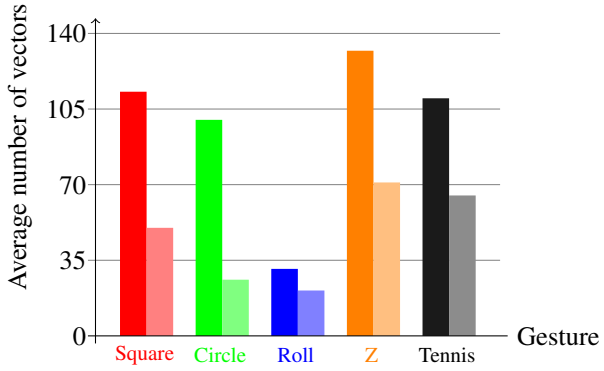


Figure 6. Reduction of vector data during filtering. The first bar for each gesture represents the average number of vectors after applying the first filter ("idle state"), the second bar the average number of vectors after applying the second, the "directorial equivalence" filter. As one can see the number of vectors are heavily reduced by this process which leads to more reliable as well as faster recognition results.

IMPLEMENTATION

In our prototype we use the Nintendo Wiimote Wireless Controller with an integrated three axis acceleration sensor (Analog Devices ADXL330). Since the Wiimote is designed for human interaction with the Wii-Console it provides the ability for basic in-game gesture recognition. Connected via the Bluetooth Human Interface Device (HID) protocol it is possible to readout its self-description data. The meaning of this communicated data has been reverse engineered by the open-source community.[1] Based on these findings it is possible to establish a basic communication with the Wiimote.

We implemented the gesture recognition in Java using the standardization of Java APIs for Bluetooth Wireless Technology (JABWT) defined by the JSR-82 specification. Using Java ensures platform independency, for developing and testing purposes we use the GNU/Linux platform with the Avetana Bluetooth implementation.[2]

The recognition process is realized as a reusable and extensible gesture recognition library based on an event-driven design pattern. The library provides an interface for basic functions, e.g. acceleration readout with the `WiiListener` interface, as well as recognition functions using a `GestureListener` interface. Through its modularity it is easy to adapt our prototype to other acceleration-based controllers. We intend to make the library available to other researchers in the field.

EVALUATION

In order to determine the performance of our system we conducted an evaluation. We collected quantitative data to determine the percentage of correctly recognized gestures for gestures trained by users themselves. In order to make the results comparable among the individual participants the five gestures described in Figure 3 were used by all participants. The group consists of one woman and five men aged between 19 and 32 years. All participants had some minor experience with the Wiimote and none used the Wiimote regularly. None of the participants was physically disabled.

Preparing the evaluation we set up our environment and the Bluetooth connection to the Wiimote. The participants got a brief explanation of the purpose of the system and how to interact with the Wiimote. Afterwards we introduced the five gestures using drawings of the five gestures (see Figure 3) and demonstrated the execution of the first gesture Square. Each participant was asked to perform each gesture fifteen times resulting in 75 gestures per participant. The participants had to push and hold the A-button on the Wiimote while performing gestures. After each completing of the respective fifteen gestures the user had to press the Wiimote's HOME-button and the drawing of the next gesture was shown. Each session lasted for fifteen minutes on average and the participants received no feedback from the system. During the evaluation we stored the complete raw data transmitted by the Wiimote.

[1] E.g., www.wiili.org

[2] www.avetana-gmbh.de/avetana-gmbh/produkte/jsr82.eng.xml

Figure 7. Participant during the evaluation of the gesture recognition.

To analyze the determined results we trained the gesture recognition system with the collected data. The system was trained using the leave-one-out method to make sure that the models were evaluated on sequences that were not used for training. That means for each participant fifteen training sets each containing the five gestures were computed. These training sets were used to recognize the remaining five gestures. The average rate of correctly recognized gestures was 90 percent. The averaged recognition rate for each of the five gestures is shown in Figure 8. The averaged recognition rate for the six participants is shown in Figure 9.

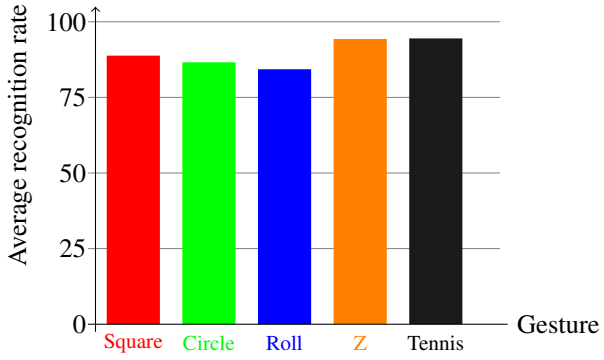


Figure 8. Average recognition rate of the five gestures. The results for the five gestures were Square = 88.8%, Circle = 86.6%, Roll = 84.3%, Z = 94.3%, and Tennis = 94.5%.

CONCLUSION

Developing new intelligent user interfaces involves experimentation and testing of new devices for interaction tasks. In our research, we are working in the field of multimodal user interfaces including visual, acoustic and haptic I/O. Based on the Wiimote we developed a gesture recognition that employs state of the art recognition methodology such as HMM, filters and classifiers, and aim to optimize hand gesture recognition for the Wiimote. As the gestures can be user-chosen the system is not limited to predefined gestures but allows each user to train and use individual gestures for a personalized user interaction with gestures. To be able to measure recognition results we trained and evaluated the system based on a set of reference gestures taken to be relevant for different task such as gaming, drawing or browsing. The recognition results vary between 85 to 95 percent, which is promising but leaves room for further optimization of the model and filters. We make the implementation of the gesture recognition library publicly available[3] and as the Wiimote is a low-cost device we invite other researchers to extend and share their experiences.

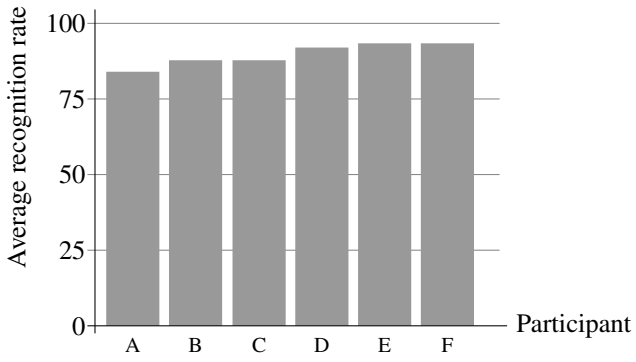


Figure 9. Average recognition rate of the four users. The results for the six participants were 84.0%, 87.8%, 87.8%, 92.0%, 93.4%, and 93.4%.

REFERENCES

1. Nintendo. http://wii.nintendo.com
2. LiveMove, AiLive Inc. http://www.ailive.net/liveMove.html.
3. Baum, L.E. and Petrie, T. Statistical inference for probabilistic functions of finite state Markov chains. *Annals of Mathematical Statistics*, (1966), 1554-1563.
4. Hofmann, F., Heyer, P. and Hommel, G. Velocity Profile Based Recognition of Dynamic Gestures with Discrete Hidden Markov Models. *Proc. of the International Gesture Workshop on Gesture and Sign Language in Human-Computer Interaction*, Springer London (2004), 81–95.
5. MacQueen, J. B. Some Methods for classification and Analysis of Multivariate Observations. *Proc. of 5-th Berkeley Symposium on Mathematical Statistics and Probability*, University of California Press 1967, 281-297.
6. Mäntyjärvi, J., Kela, J., Korpipää, P. and Kallio S. Enabling fast and effortless customisation in accelerometer based gesture interaction. *Proc. of the MUM '04*, ACM Press (2004), 25–31.
7. Mäntyjärvi, J., Kela, J., Korpipää, P., Kallio S., Savino, G., Jozzo L. and Marca, D. Accelerometer-based gesture control for a design environment. *Personal Ubiquitous Computing*, Springer London (2006), 285–299.
8. Mäntylä, V. M. Discrete hidden Markov models with application to isolated user-dependent hand gesture recognition. VTT Publications (2001).
9. Rabiner, L.R. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proc. of the IEEE*, IEEE (1989), 257–286.

[3] http://wiigee.sourceforge.net