Journal of Mobile Multimedia, Vol. 1, No.2 (2005) 092-112

GESTURE INTERACTION FOR SMALL HANDHELD DEVICES TO SUPPORT MULTIMEDIA APPLICATIONS

JANI MÄNTYJÄRVI, SANNA KALLIO, PANU KORPIPÄÄ, JUHA KELA, JOHAN PLOMP

VTT Electronics

P.O. Box 1100

FIN 90571 Oulu, Finland

Email: firstname.lastname@vtt.fi

Received February 25, 2005
Revised April 16, 2005

Accelerometer-based gesture control is proposed as a complementary interaction modality for small handheld devices to enable a variety of multimedia applications. The motivation for experimenting with gesture interaction is justified by the personal and public domain prototype applications developed. The challenges related to developing user-dependent and independent gesture control are presented. In this article, we experiment with methods for user-dependent gesture recognition with a low number of training repetitions, and for feasible user-independent gesture recognition from a moderately large set of gestures. The user-dependent gesture recognition performance of the continuous Hidden Markov Model (HMM) is better when compared to discrete HMM with three gesture repetitions in a training set. With continuous HMM, a recognition accuracy level of 95% is obtained with or without tilt normalization, while for discrete HMM a best recognition accuracy of 90% is obtained. The user-independent gesture recognition performance with continuous HMM of 89% is considerably better compared to tests with discrete HMM, when both are obtained with cross-validation from 2,520 gestures. An important result is that the effect of using tilt normalization notably increases the user-independent gesture recognition performance by 10-15% depending on the method used. The chosen methods show great potential for gesture-based interaction in multimedia applications.

Key words: Human computer interaction, input technology, mobile devices, gesture recognition, gesture control
Communicated by: D Taniar

1 Introduction

The research themes of mobile and pervasive computing are preparing the way for anytime-anywhere-anything-anytype control of devices and applications. One potential intuitive communication channel is gestures, which has not yet been fully utilized in human-computer interaction. The role of this complementary interaction modality is expected to increase as the enabling technologies develop. For example, development in the area of micro electronics mechanical systems (MEMS) has already led to the market penetration of low cost, high accuracy inertial sensors e.g. accelerometers, and novel applications in consumer electronics are already emerging.

Mobile terminals, e.g. PDAs or mobile phones, and devices in our everyday environment contain an increasing number of multimedia-related applications. Currently, interaction with various types of multimedia applications in distinct devices is not only rich but it has also introduced new challenges with I/O devices. Solutions for enabling more intuitive human-machine interaction have been developed. The present gesture input methods enabled by inertial sensors e.g. accelerometers for detecting movements can be integrated into clothing, wristwatches, jewellery or mobile terminals. The input methods can be used to control devices, for instance mobile or console games, toys, multimedia shows, media players, stereos, TV, and other home appliances, with simple predefined or user definable hand movements.

This article concerns accelerometer-based gesture interaction to support multimedia application control with personal and public devices. In related work, we focus on movement sensor-based approaches, which utilise different kinds of sensors, e.g. tilt, acceleration, pressure, conductivity, capacitance, etc. to measure movement. One example of an implementation is GestureWrist, a wristwatch-type gesture recognition device using both capacitance and acceleration sensors to detect simple hand and finger gestures [13]. Accelerometer-based gesture recognition is used, for example, for a musical performance control and conducting system [14], and a glove-based system for recognition of a subset of German sign language [3]. In a wearable interface called Ubi-finger acceleration, touch and bend sensors are used to detect a fixed set of hand gestures, and an infrared LED for pointing a device to be controlled [16]. Yet another gesture-based interaction device is XWand [17]. It utilizes both sensor-based and camera-based technologies capable of detecting the orientation of the device using a 2-axis accelerometer, a 3-axis magnetometer and a 1-axis gyroscope, and position and pointing direction using two cameras. The user can select a known target device from the environment by pointing, and control it with speech and a fixed set of simple gestures.

Gesture recognition has been also studied for implicit control of functions in cell phones, e.g. for answering and terminating a call without the user having to explicitly perform the control function [1], [12]. In implicit gesture control, the main problem is that users usually perform the gesture very differently in different cases, e.g. picking up a phone can be done in several different ways depending on the situation. Explicit control directly presupposes that the gestures trained for performing certain functions are always repeated as they were trained.

As a complementary interaction modality, a acceleration-based gesture command interface is quite new, and many research problems still need to be solved. The topic is very wide in scope, since there are a very large number of possible suitable gestures for certain tasks, as well as many tasks that could potentially be performed using gestures. These tasks can be divided into two basic categories: personal and public. Personal tasks refer to using a personal device, e.g., a mobile phone for controlling its internal applications. These applications are only used by one person; the owner of the device. The other category refers to using a public device, e.g., a remote control for controlling external devices, such as appliances at home or in public spaces.

From the gesture recognition viewpoint, the two task categories have different requirements. For tasks in the personal category, it is feasible to use personal gestures, which are often preferred by the users [8]. In this case, user-dependent gesture recognition needs to be applied. Using personal free form gestures requires practicing them online [6]. If training is too laborious, it may cause users to abandon the interaction method. Therefore, the training process should be as effortless and quick as possible [11]. For the tasks in the public category, gestures cannot be personal, since there may be many users of the control device. In this case, user-independent gesture recognition is required, where fixed gesture models have been trained beforehand by multiple users with the aim of covering the

variability between different users. Obviously, the recognition accuracy for detecting the gesture commands should be high for user satisfaction in both categories, since too many mistakes cause the users to abandon the method.

As a sensing device, SoapBox *(Sensing, Operating and Activating Peripheral Box)* is utilized in this work. It is a sensor device developed for research activities in pervasive computing, context awareness, multi-modal and remote user interfaces, and low power radio protocols [15]. It is a light, matchbox-sized device with a processor, a versatile set of sensors, and wireless and wired data communications. Because of its small size, wireless communication and battery-powered operation, it is easy to install in different places, including moving objects. The basic sensor board of the box includes a three-axis acceleration sensor and other sensors for monitoring the environment [15].

Various statistical and machine learning methods can potentially be utilized for training and recognizing gestures [1], [12]. This study applies a well-known method, HMM. HMM is widely used in speech and hand-written character recognition as well as in gesture recognition in video-based and glove-based systems.

In our previous work we have found that people usually prefer to define their own gestures, concluding that personal category gesture control should be customizable [8]. Moreover, it should be possible to configure which functions are performed with gestures [9]. This result leads to the important requirement that one should be able to train and recognize free form gesture commands. The users should be able to carry out training of personal gestures with as few repetitions as possible, since performing many repetitions can be a nuisance. The other problem with device or application control in public spaces concerns providing accurate user-independent gesture recognition with a moderately large set of gestures. Additionally, it must be noted that we are experimenting with gesture recognition for very small mobile handheld devices with limited processing, memory and power resources.

In this article we discuss gesture control as a novel interaction modality for small handheld devices that enables control of a variety of multimedia applications. We present the challenges related to developing user-dependent and independent gesture control via several prototype multimedia applications that we have designed. The motivation for experimenting with gesture interaction is justified by our prototype applications. The main contribution of this article is experimenting with methods for user-dependent gesture recognition with a low number of training repetitions, and for feasible user-independent gesture recognition from a moderately large set of gestures.

The organization of the paper is the following. The basic concepts regarding gesture interface are first defined and categorized in Section 2. Methods for gesture training and recognition and for decreasing the number of training repetitions are presented in Section 3. Prototype applications for personal and public device domains demonstrating the practical feasibility of gesture recognition are introduced in Section 4. Section 5 provides the experiments and results. Finally, discussion and suggestions for future work are given together with the conclusions.

2 Gesture Control

As discussed in the previous section, accelerometers are utilized in implementing various types of user interfaces. To clarify the differences between various approaches, the types of movement sensor-based user interfaces are categorized in Table 1.

Table 1: Categorization and properties of movement sensor-based user interfaces

Interface type	**Operating principle**	**Customization**	**Complexity**
1. Measure & control	Direct measurement of tilting, rotation, or amplitude	-	Very low
2. Discrete gesture command	Gesture recognition (user-dependent / user-independent)	Machine learning, freely customizable	High
3. Continuous gesture command	Continuous gesture recognition (user-dependent / user-independent)	Machine learning, freely customizable	Very high

Direct measurement and control systems are not considered gesture recognition systems since their operating principle maps measurement of tilt, rotation or amplitude directly to control. In this article, gestures are referred to as user hand movements collected with a set of sensors in a handheld device. Hand movements are modeled by machine learning methods in such a way that any movement performed can be trained for later online recognition. Furthermore, a gesture-based device control command is executed based on the hand movement recognized. In the case of a discrete gesture command, the start and end of a gesture is defined, e.g., with a button, while in the case of continuous gesture commands, the recognition of gestures is carried out online from a flow of hand movements. Handheld devices with gesture recognition enable the control of applications located in a handheld device or in external devices in the vicinity.

In terms of sensor-based hand movement interfaces, the paper hence focuses on discrete gesture command interfaces (the second category in table 1). User-dependent and user-independent gesture recognition are both addressed to cover the personal and public categories of usage tasks. A multitude of simple measure & control applications exist e.g. tilt and rotation-based actions. We consider them to be a part of category one in table 1.

3. Gesture Training and Recognition

In this section, we focus on methods enabling gesture interaction with small handheld devices. Common issues for these types of devices are limited processing, memory and power resources. We experiment with a well-known pattern recognition method for time series modeling HMMs and adapt the method to meet the requirements of small devices. According to our previous studies, users prefer intuitive user-definable gestures [8]. This is a challenge for the recognition and training system, since both online training and recognition are required. In particular, a low number of repetitions of a gesture are required from a user during the training in order to make usage of the system comfortable. On the other hand, good generalization performance in recognition must be achieved while maintaining good recognition accuracy. Other requirements are that a recognizer must maintain models of several gestures, and when a gesture is performed, training or recognition operations by a system must not take a long time. Our research is focused on devices with limited processing, memory and power resources. Thus, in addition to positive user experience, other motivations for seeking efficient yet light gesture

training and recognition methods are limitations to the above-mentioned device. This section presents methods used in online gesture recognition in our prototypes.

In accelerometer-based gesture interaction, sensors produce signal patterns typical for gestures. These signal patterns are used in generating models that allow the recognition of distinct gestures. We have experimented with Hidden Markov Models (HMM) for recognizing the gestures. The main motivation for choosing HMM for our purposes is that it can be applied to modeling time-series with spatial and temporal variability. In our experiments, we have tested two types of HMM: discrete HMM (dHMM) and continuous HMM (cHMM). HMM has also been utilized in other experiments for gesture and speech recognition [12], [13]. Acceleration sensor-based gesture recognition using HMM has been studied for example in [3], [6], [12].

The recognition system works in two phases: training and recognition. Common steps for these phases in both HMM types are signal sampling from three accelerometers to 3D sample signals and pre-processing. Repeating the same gesture produces variation of measured signals, because the tempo and the scale of the gesture can change. In the pre-processing step, data from gestures is first normalized to an equal length and amplitude. In the case of discrete HMM, the pre-processed 3D signal must be converted into discrete symbols before being submitted to the trainer or recognizer. This step is called vector quantization. Our procedure for gesture training and recognition with discrete HMM includes experiments with adding noise to the data. For both discrete and continuous HMM recognition systems, we also experiment with tilt normalization. The gesture recognition approach using discrete HMM is presented in Figure 1a) and using continuous HMM in Figure 1b).

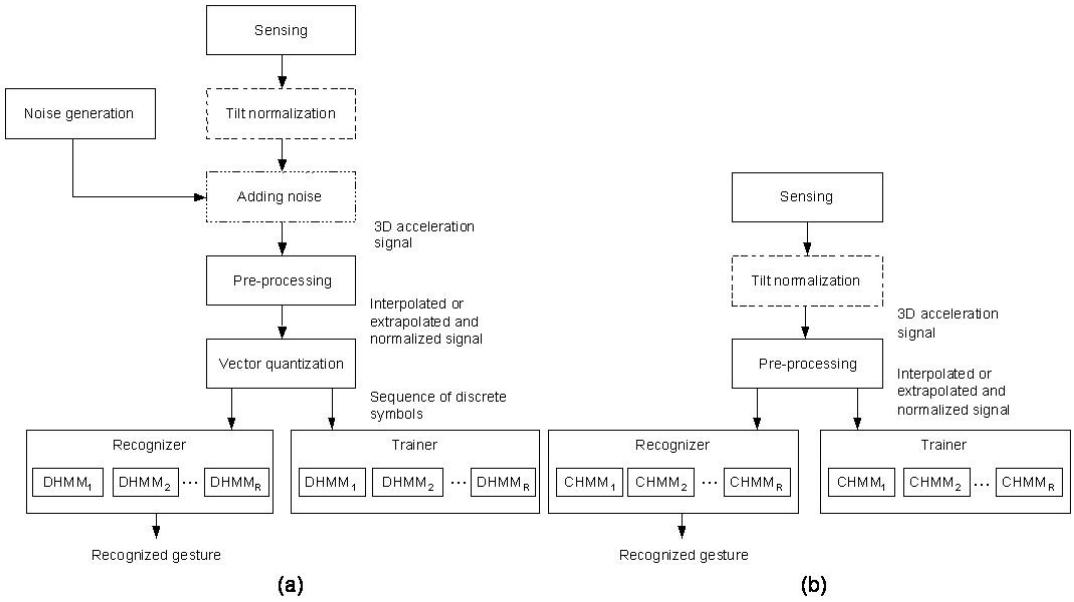


Figure 1. Block diagram of a gesture recognition / training system (a) with discrete HMM (b) with continuous HMM.

3.1 Pre-processing

The pre-processing stage consists of interpolation or extrapolation and normalization. Gesture data is first linearly interpolated or extrapolated if the data sequence is too short or too long, respectively. Then the amplitude of data is normalized to have a mean of $\mu = 0$ and variance of $\sigma^2 = 1$. The same parameters are used both in the training and in the recognition phase.

3.2 Tilt compensation

Optionally, a tilt-compensation procedure can be applied to the data. Assuming that the user does not intend to use the tilt of the device as a discriminating factor between trained gestures, the tilt can be estimated and compensated in the data in order to achieve recognition that is less prone to differences in how the device is held in the hand. The tilt is estimated by calculating the mean vector of the gesture. This vector will roughly correspond with the direction of the gravity vector. Subsequently a rotation is performed that aligns this vector with a predefined ideal position, (the z-axis). This rotation implies a rotation around a line perpendicular to both the the gravity vector and the calculated mean vector, the rotation angle being the angle between these vectors. The rotation matrix $\mathbf{M}_{\mathbf{rot}}$ is applied to all data vectors:

$$\mathbf{M}_{\mathbf{rot}} = \begin{bmatrix} \cos^2\alpha\sin\beta + \sin^2\alpha & \cos\alpha\sin\alpha(\sin\beta - 1) & -\cos\alpha\cos\beta \\ \cos\alpha\sin\alpha(\sin\beta - 1) & \sin^2\alpha\sin\beta + \cos^2\alpha & -\sin\alpha\cos\beta \\ \cos\alpha\cos\beta & \sin\alpha\cos\beta & \sin\beta \end{bmatrix} \quad (1)$$

where α is the angle between the projection of the mean vector on the *x*-*y* plane and the *x*-axis, and β is the angle between the *x*-*y* plane and the mean vector (figure 2).

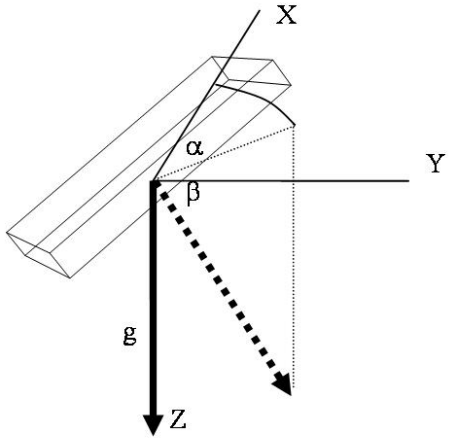


Figure 2. Illustration of coorinate planes and projection angles.

3.3 Discrete HMM

The Hidden Markov Model (HMM) is a stochastic signal modeling method. The HMM is an extension of the Markov process. The output of the Markov model is the set of states at each instant of time, where each state corresponds to a physical, observable event. The HMM includes the case where the observation is a probabilistic function of the state. Hence, the HMM is a double stochastic process with an underlying stochastic process that is not observable, but can be observed through another set of stochastic processes that produce the sequence of observation. Formally, a HMM can be expressed as

$$\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi}) \tag{2}$$

where $\mathbf{A} = \{a_{ij}\}, 1 \le i, j \le N,$ denotes the state transition probability matrix, $\mathbf{B} = \{b_j(o_k)\}, 1 \le j \le N, 1 \le k \le S,$ is the observation symbol probability matrix and $\boldsymbol{\pi} = \{\pi_i\}, 1 \le i \le N,$ is the initial state probability vector. The specification of the discrete HMM involves choosing a number of states *N*, a number of discrete symbols *S*, and a definition of the three probability densities with matrix **A**, **B**, and $\boldsymbol{\pi}$.

Usually, three basic problems must be solved for the real applications: the classification, the decoding and the training. In this study, only classification and training are relevant. The training, i.e., estimating the model parameters (**A**, **B**, $\boldsymbol{\pi}$) is solved with an iterative Baum-Welch algorithm. The classification, i.e., calculating the probability of the observation sequence $\mathbf{o} = o_1 o_2 \ldots o_T$ given the model $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ is solved using a Viterbi algorithm. The global structure of the HMM recognition system is composed of parallel connection of each trained HMM $(\lambda_1, \lambda_2, \ldots \lambda_R)$, where λ_i indicates a trained HMM model for each gesture and *R* is a number of gestures [18]. Hence, adding a new HMM or deleting the existing one is feasible. Recognition of the given unknown gesture is performed by finding the index of the discrete HMM that produces the maximum probability of the observation symbol sequences.

An ergodic topology was utilised in this article. In ergodic HMM, every state of the model could be reached from every other state of the model. The left-to-right model is a special case of an ergodic model and often preferred when modeling a time-series whose properties sequentially change over time. However, in the case of gesture recognition from acceleration signals, both ergodic and left-to-right models have been reported to provide similar results [3]. We heuristically evaluated the number of states in our ergodic HMMs to be seven.

3.3.1 Vector quantization

Using discrete HMM for training and recognition of the gesture data requires vector quantization to convert the continuous pre-processed three-dimensional signal **x** into a sequence of discrete codebook indices **o**. The codebook is a collection of prototype vectors and the each three-dimensional observation of the **x** is assigned the index of the nearest prototype vector. In our experiments, we use a k-means algorithm [2] to generate an appropriate codebook for quantization and test HMMs with different codebook sizes.

3.3.2 Decreasing user effort in training with dHMM

The user experience in accelerometer-based gesture interaction should be as positive as possible and making several training repetitions can be a nuisance for the user. Thus, an approach for trying to decrease the amount of training repetitions is well justified. It has been shown that adding noise increases detectability in decision-making under certain conditions [7]. It has also been shown that adding noise augmented data duplicates to training data decreases the user effort in the training phase and improves the gesture recognition performance in the case of one user [11]. In this article, we apply the idea of adding noise to experiment with user-dependent and independent gesture recognition for personal and public multimedia applications.

The idea of this approach is to generate new training data, the three-dimensional noise-distorted gesture signal duplicates $\mathbf{x}_i + \mathbf{n}_i$, by copying the original gesture data vector $\mathbf{x}_i$ and adding random noise vector $\mathbf{n}_i$ into the copy. We consider uniform noise distribution in our experiments. Random samples are generated between [$-a$, $+a$], mean $\mu = 0$, variance $\sigma^2 = a^2/3$. Various signal to noise ratios (SNR) are experimented with. SNR is determined as a ratio of signal variance to noise variance.

3.4 Continuous HMM

When observations are continuous signals, vector quantization might cause degradation. In that case, it is possible to use HMMs with continuous observation densities. To insure that the parameters of the probability density function (*pdf*) of the model can be re-estimated in a consistent way, some restrictions have to be placed on the form of the used *pdf*. The most general representation of the pdf is a finite mixture of the form

$$b_j(\mathbf{x}) = \sum_{m=1}^{M} c_{jm} \Re\left[\mathbf{x}, \boldsymbol{\mu}_{jm}, \mathbf{U}_{jm}\right], \quad 1 \le j \le N \tag{3}$$

where $\mathbf{x}$ is the vector being modeled, $c_{j,m}$ is the mixture coefficient for the mth mixture in state j and $\Re$ is any log-concave or elliptically symmetric density with mean vector $\boldsymbol{\mu}_{j,m}$ and covariance matrix $\mathbf{U}_{j,m}$ for the mth mixture component in state j. In our experiments, the number of mixtures is $M = 2$ and $\Re$ is a Gaussian density. The re-estimation procedure for the coefficients $c_{j,m}$, $\boldsymbol{\mu}_{j,m}$, and $\mathbf{U}_{j,m}$ is formulated in [10], [4] and [6]. In the training phase, the HMM uses Baum-Welch algorithm to estimate model parameters $\lambda = (\mathbf{A},\mathbf{B},\boldsymbol{\pi})$, which maximize the probability of the observation sequence given the model $P(\mathbf{O}|\lambda)$. In the recognition phase, probability $P(\mathbf{O}|\lambda)$ is calculated from estimated parameters $(\mathbf{A},\mathbf{B},\boldsymbol{\pi})$ with Viterbi algorithm. The computational complexity of the Viterbi algorithm is $O(S^2T)$, where S is the number of states and T is the length of the observation. The training is carried out using Baum-Welch algorithm, which is iterative process and its complexity is $O(kS^2T)$, where k is the number of iterations. The same algorithms are used in both discrete and continuous HMMs. However, in the case of the cHMM, the estimation of observation density within each state j (i.e. estimation of the matrix $\mathbf{B}$) requires estimation of the coefficients of the continuous *pdf*, which are mixture gain c_{jm}, mean vector $\boldsymbol{\mu}_{jm}$ and covariance matrix $\mathbf{U}_{jm}$, whereas in the case of the dHMM observation density within each state can be calculated directly according the occurrence of the codebook symbols. In the dHMM the size of the matrix $\mathbf{B}$ depends on size of the codebook, i.e., larger codebook means larger $\mathbf{B}$. In cHMM this does not count.

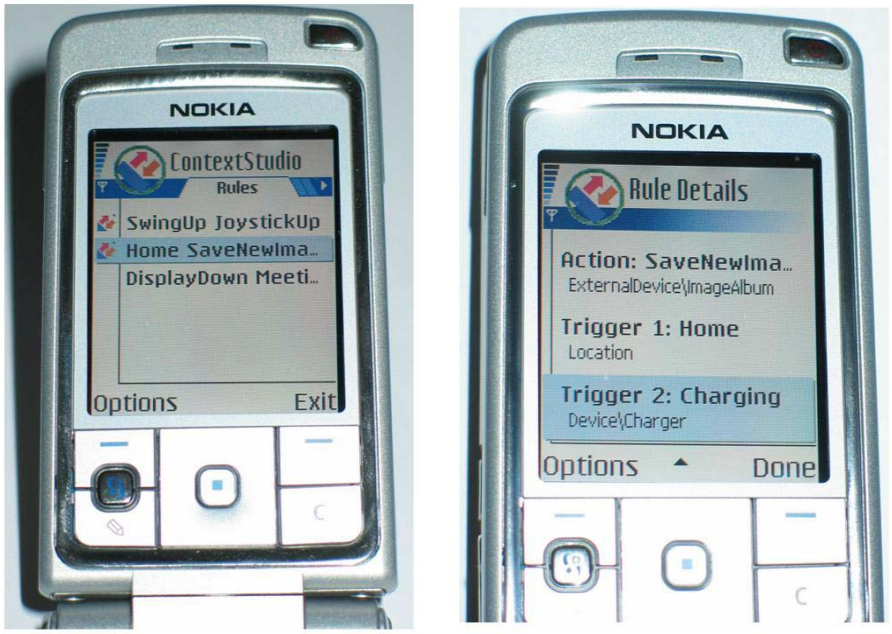
4. Prototypes for Controlling Multimedia Applications

We have implemented prototypes for both personal and public device application domains.

4.1 Personal device

The user controls a personal device with customizable gestures. In this case, user-dependent training of gesture models is preferable. In our personal device prototype, gesture training and recognition are performed inside a small handheld device, which is presented in figure 3. Acceleration sensors are embedded. The prototype includes a tool and software framework, which enable the user to freely customize which applications and functions are executed based on the gesture commands [9]. In the device, gestures can be used for interacting with various existing multimedia applications without changing the applications. For example, the user browses an image album in a smart phone with up/down gestures. Gestures similar to throwing and catching can be used to initiate sending and getting pictures to/from an external device, such as home media terminal.

a)



b)

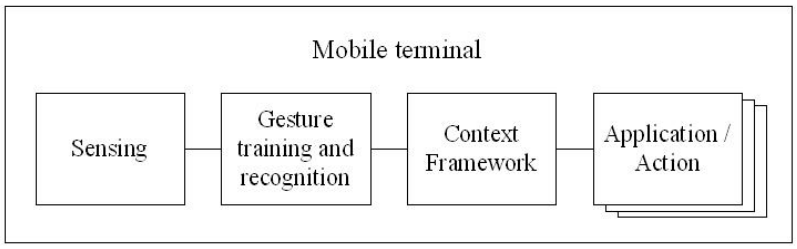


Figure 3. a) Examples of user interfaces, and b) a conceptual overview of a personal device gesture control system

4.2 Gesture visualization

Visualization of the gestures is a useful application to provide feedback on the gesture made and/or provide the user with information on what kind of gesture should be made. Ideally the gesture displayed should be a projection of the three-dimensional path onto a suitable plane unless the gesture is visualized in 3D by means of e.g. pre-recorded video or VRML. Although gestures can be performed in 3 dimensions, people often prefer gestures in two dimensions to avoid needless complexity. In that case, a suitable projection plane can be found by determining the plane that maximizes the variation of the data set. Data obtained from gestures is used to demonstrate the visualization of gestures tests in the plane of view from the user, i.e. by moving the object to the left and right and up and down. The gestures were compensated for tilt using the method described in Section 3, and subsequently the projection was performed onto this plane of view. The acceleration data were then integrated twice in order to obtain the position of the point. In figure 4, the trajectories of a square and a heart gesture are shown.

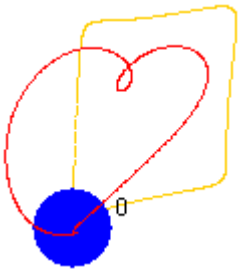


Figure 4. Visualized trajectories of a heart and a square gesture

Some variation exists among the visualized gesture trajectories drawn according to the data. This is due to changes in the orientation of the box during the gesture and offset and gain errors. The orientation of the box is only compensated for the whole gesture by means of the tilt compensation method. However, the box is tilted during the gesture significantly, typically towards the user when going up and vice versa. This tilt is difficult to compensate reliably without the addition of other sensors, like gyroscopes. Local tilt compensation methods that attempt to estimate the changing direction of the gravity vector during the gesture and compensate accordingly provide some improvement over the straightforward visualization method described above. Thus, visualization of gestures from accelerometer data can be used to obtain an indication of the gesture made.

4.2 Public Space Devices

We have developed two different public space gesture control prototypes that are currently available to the public in the Heureka and Tietomaa National Science Centers in Finland. Both prototypes utilize rugged control device hardware that is specially designed for public space usage in order to take both usability requirements and easy maintenance into account, figure 5. The prototypes are based on wireless handheld sensor box, SoapBox [15]. Acceleration sensors (ADXL202) of the box measure both dynamic acceleration (e.g. motion of the box) and static acceleration (e.g. tilt of the box). The acceleration is measured in three dimensions and sampled at a rate of 46 Hz. The measured signal values are wirelessly transmitted from the handheld box to a receiver that is connected to a Windows PC with a serial connection. The handheld gesture controller is equipped with two buttons, one for discrete gesture recognition and the other for tilting-based continuous control. The start and

end of the gestures are marked by pushing the right button on the box at the start of the gesture and releasing it at the end, which then activates either the training or the recognition algorithm. The recognition results are be mapped to different control commands and transmitted to the control target using infrared control signals or TCP/IP socket communication. The mapping between gestures and output functions is done in the training phase.

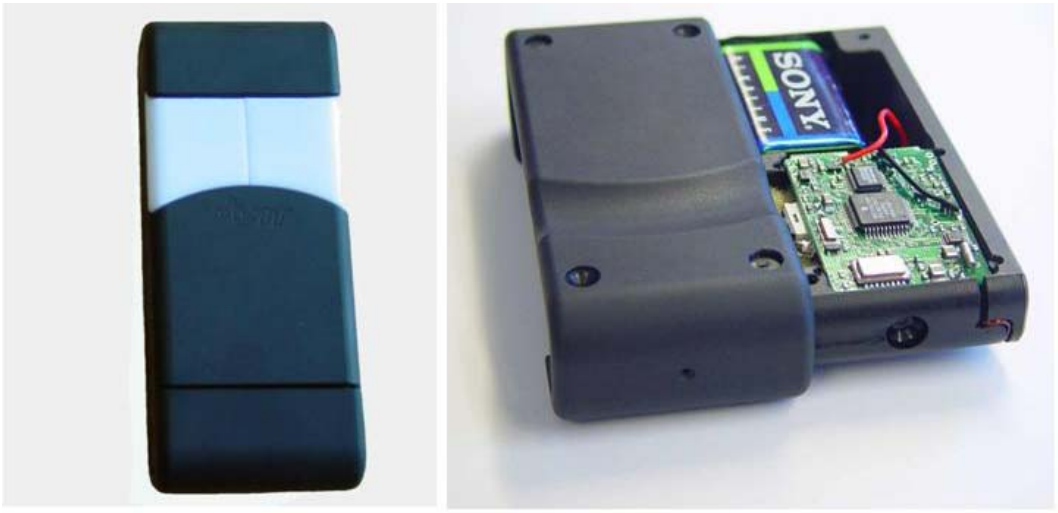


Figure 5. Gesture control hardware with specially designed casing.

Figure 6. A Heureka Science Center visitor interacting with the DVD player.

The Heureka Science Center prototype is used to control a DVD multimedia presentation and the height of a motor operated table-top where the DVD player and a display are located, figure 6. By pressing the right control button of the gesture device and performing one of the predefined six gestures, users can control the playback (play, stop, FWD and REW) and sound volume functions of the DVD presentation. The height setting of the table top can be controlled by pressing the left control button and simultaneously tilting the device upwards or downwards. The gestures used in the demonstration were selected from amongst the most popular control gestures for a VCR control according to a gesture vocabulary questionnaire [8].

Figure 7. A Tietomaa Science Center visitor controlling the lights.

The Tietomaa Science Center prototype utilizes gestures and tilting to individually control Red, Green and Blue color intensity levels of a tube shaped RGB light projector, figure 7. Users can select the controllable color by "drawing" the first letter of the desired color (e.g. S, P or V in Finnish) or by selecting the corresponding gesture symbol from the instruction poster. The gesture symbols used are triangles pointing left (red), up (green) and right (blue). When the gesture is performed, the recognition result is visualized to the user by flashing the selected color after which the user can either increase or decrease the selected light intensity by pressing the second button of the control device and tilting the device. The RGB light tube shows the sum of the three colors so, for example, if red and green are set to maximum and blue is set to minimum, the output color is yellow. In addition to supporting the gesture exploration and to reward the users, there was also one extra secret gesture which activated the automatic flashing light-mode, making the RGB light tube go smoothly through all possible color combinations.

Compared to the personal control application domain, the public space sets certain system requirements that have to be considered. User groups vary greatly in public spaces such as libraries, art exhibitions, shopping malls, airport lounges and science centers. The age of the users can range from infants to retired people, including both left- and right-handed persons, typically with no prior experience of gesture-based control. In general, the first contact with the gesture recognition system can be considered an explorative learning experience where users do not bother to read the provided

instructions but instead try to play and experiment with the system on their own. The instructions and manuals are consulted only when there are errors or unexpected system behavior. Due to these requirements, the handheld gesture control device should be designed to have a symmetric, easy to grasp form and big buttons that support both left- and right-handed usage and persons with poor vision. The recognition system should be optimized for the best possible user-independent recognition. Due to the fact that user-independent recognition is seldom 100% accurate for a set of different users, fast system response and good feedback are essential. If for some reason the gesture is not recognized, the system should provide an error indication to the user. This is particularly important in the case of explorative non-critical fun and entertainment-related control tasks, such as interacting with games, toys and multimedia shows, where continuous error messages can often become annoying.

These prototypes confirmed that the users like to try and experiment with the system without referring to written instructions. We found out that it is useful to have some form of animated multimedia instructions to better illustrate to the users how to perform the gestures and how the gestures were performed.

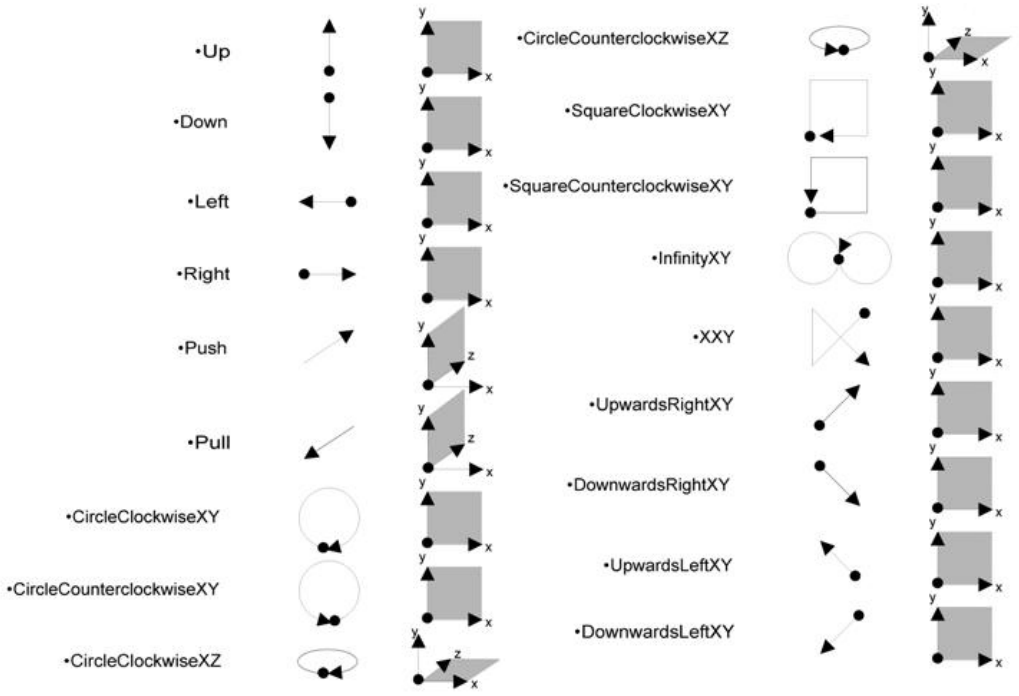


Figure 8. Gestures used in experiments. The origin of the gesture is presented as a dot and an arrow indicates the trajectory of the gesture. The x-, y-, z-coordinates indicate that the gesture is drawn in the air in the corresponding space in front of the user.

5. Experiments and Results

As discussed in the previous section, gesture-based applications can be either personal or public. There are certain characteristics affecting the requirements for recognition methods for both cases: For personal applications, for example in mobile terminals, gestures are user-dependent and interaction during training must be as effortless as possible. In the case of public space applications, the gestures

are user-independent and good generalization performance is required. In this section we present experiments and results for both cases.

5.1 Experiments

To test our gesture recognition approach, 18 types of gestures were selected for experiments, figure 8. Gestures were performed in a three-dimensional space in front of the user, figure 8. For each gesture, 20 distinct three-dimensional acceleration vectors were collected from 7 persons, and thus the total data set consisted of 2,520 gestures. An example of the signals from a gesture called "infinity" is presented in figure 9.

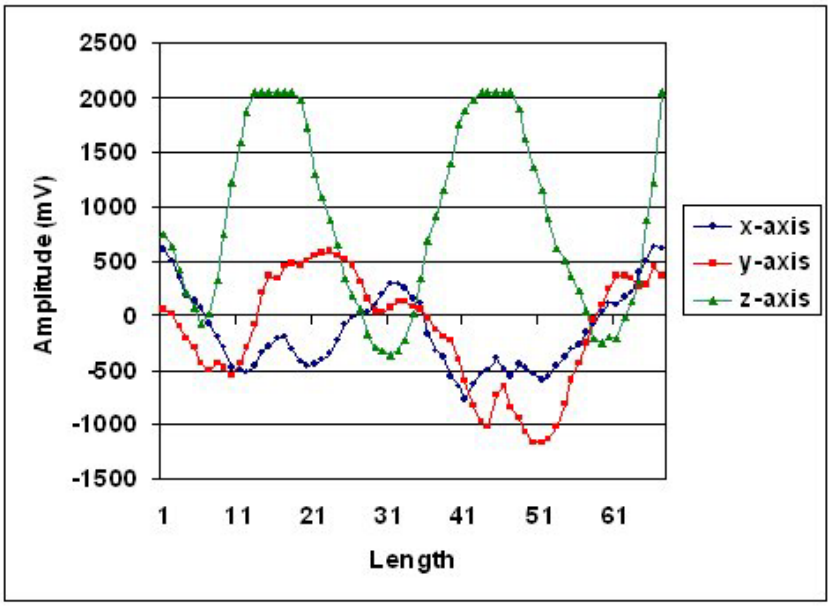


Figure 9. Three-dimensional acceleration data vector for gesture infinity.

Each three-dimensional vector was either interpolated or extrapolated to a length of 40 samples. Thereafter in the case of a discrete HMM, a vector quantizer was used to map three-dimensional vectors into a one-dimensional sequence of codebook indices. The codebook was generated from collected gesture vectors using a the k-means algorithm. After the vector quantization, the gesture was used either to train HMM or to evaluate the recognition capability of the HMM. In the case of a continuous HMM, interpolated/extrapolated three-dimensional vectors were used in tests. When testing the effect of tilt normalization, the transformation matrix in Eq. (1) was applied to interpolated or extrapolated three-dimensional vectors before applying the HMMs.

5.2 User-dependent recognition

According to our previous research, users prefer personal gestures [8]. In this case, gesture recognition is user-dependent and the gesture set consists entirely of gestures performed by one user. With user-dependent gestures the training has to be done by the user. This means that the training situation should be as quick and easy as possible for the user. Here, we do not speculate on user interface issues affecting the above-mentioned issues. Instead, we study gesture training and how to

keep the number of gesture repetitions required from the user as low as possible. In our previous work on user-dependent recognition, we have shown that in the case of eight DVD-player control gestures, excellent recognition accuracies can be obtained with cHMM and dHMM [11]. In this subsection we extend the study on user-dependent gesture recognition and also experiment with discrete and continuous HMMs together with tilt normalization and noise distorted duplicates in the case of 18 types of gestures.

First, the recognition accuracy using different amount of gesture repetitions in a training set was examined. We empirically evaluated the number of states in the HMMs to be seven. The maximum number of iterations in training was heuristically set to ten. The effect of the number of gestures in a training set on continuous HMM and on discrete HMM with different size codebooks is presented in Figure 10. The results show that performance of continuous HMM is superior compared to discrete HMM, even with three gesture repetitions in a training set.

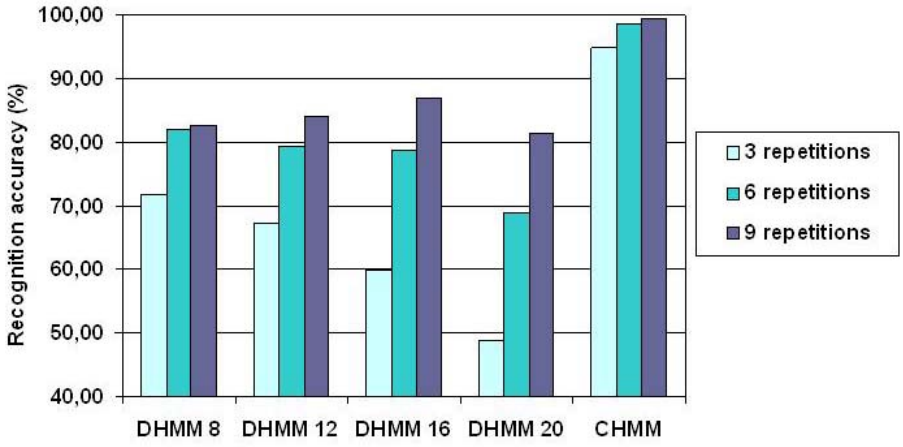


Figure 10. Recognition performance with different number of gestures in a training set for dHMM and cHMM.

Since we are aiming at a good user experience in, training the effect of adding noise distorted gesture duplicates with three gesture repetitions to a training set is examined. In particular, we examined with various noise SNR levels and codebook sizes to find a noise level leading to the best results with dHMMs. Recognition accuracies with various SNRs for uniformly distributed noise were calculated using three original and three noise distorted duplicates as a training set. Recognition rates with various SNRs for different codebook sizes are presented in figure 11. The best recognition performance of 90% is obtained with SNR 1 with a codebook size of 20. The results indicate that cHMM performs better compared to any optimised configuration of dHMM.

One challenge in gesture recognition is that people tend to hold a small device in slightly different orientations when performing gestures. We examined the effect of eliminating orientation differences with the transformation matrix Eq. (1), in user-dependent gesture recognition. Figure 12 presents the recognition accuracies obtained with original data and with rotation normalized data. In these experiments, the parameters were as follows: the number of states was seven and the number of iterations in training was 10 in both discrete and continuous HMMs. Three gesture repetitions were used in the training set for continuous HMMs while three gesture repetitions and three noise distorted

duplicates were used in the training set for discrete HMMs. The result in figure 12 is a mean of three separate person's recognition accuracy. The results show that user-dependent gesture recognition accuracy can not be improved with tilt normalization.

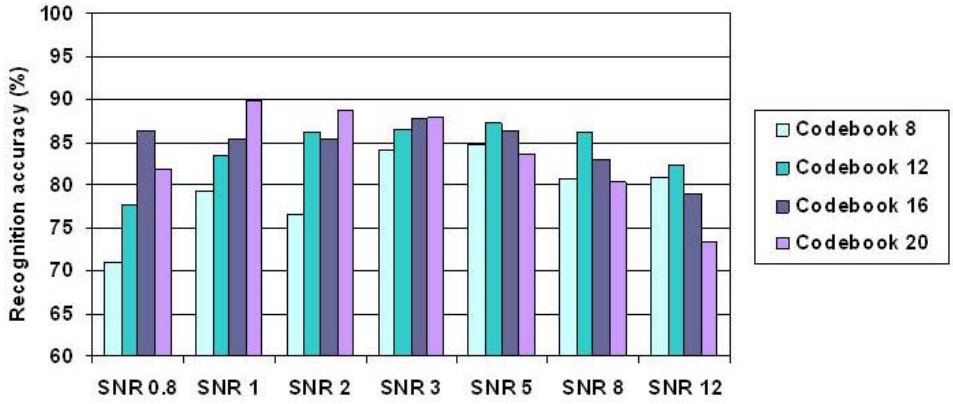


Figure 11. Recognition rates with various SNRs for different codebook sizes.

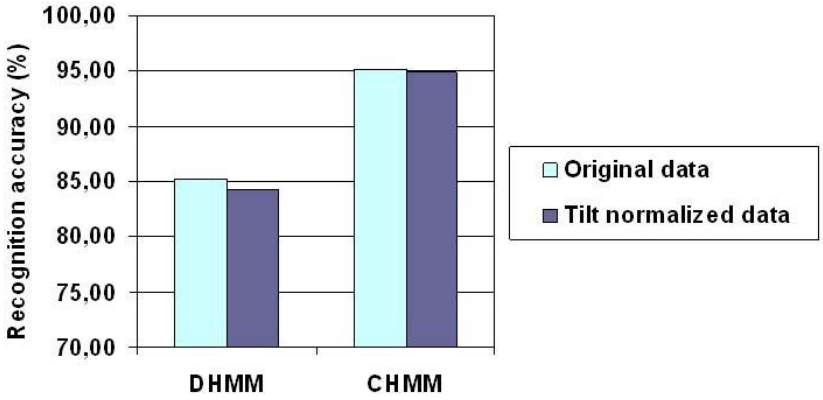


Figure 12. User-dependent recognition with original data and with rotation normalized data.

The user-dependent gesture recognition results show that performance of continuous HMM is better compared to discrete HMM even with three gesture repetitions in a training set. A recognition accuracy level of 95% is obtained for cHMM with or without tilt normalization while a best recognition accuracy of 90% was obtained for dHMM in tests with noisy data. These results show that cHMM provides more reliable user-dependent gesture recognition performance that dHMM.

5.3 User-independent recognition

In gesture-based control in public spaces, the recognition sets is particularly challenging since people tend to perform similar gestures in a different manner. The problem is illustrated in figure 13, which shows how different users can perform a similar gesture. Some people make smaller gestures than the others, and the device might be tilted during the gesture. Moreover, the speed of performing gestures may be different, leading to fewer samples in a data vector. These factors may result in the

fact that very different waveforms represent the same gesture. Furthermore, this presents some challenges in user-independent gesture recognition.

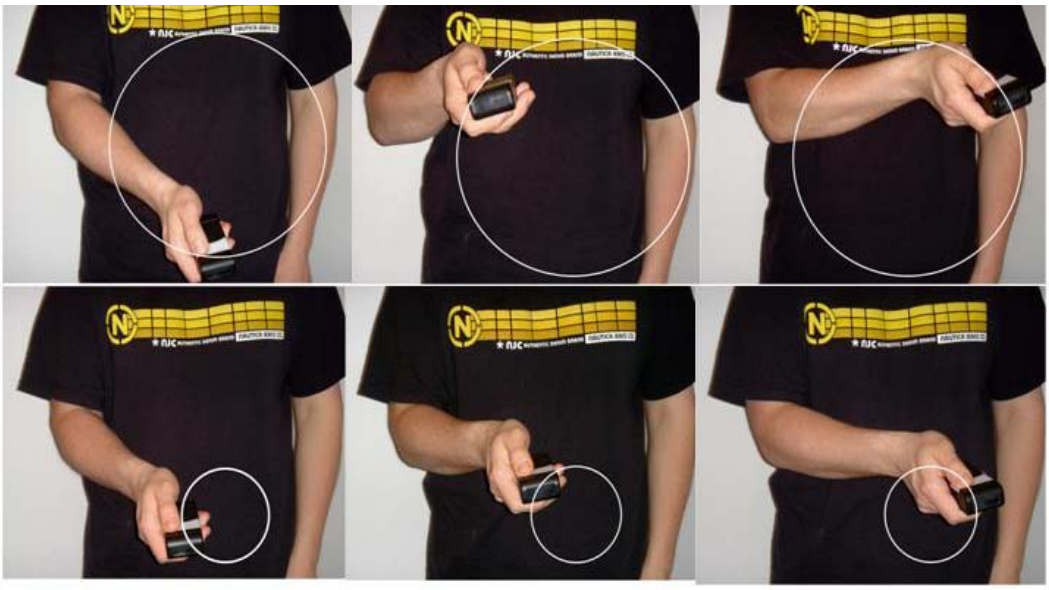


Figure 13. Typical differences in making gestures.

In order to examine user-independent recognition, we performed experiments with an entire data set consisting of 2,520 gestures from seven users. Experiments included a comparison of discrete HMM, discrete HMM with added noise distorted duplicates and continuous HMM with and without tilt normalization. Results were obtained with seven-fold cross validation, in which models were trained with data from six users and tested with data from one user. The models were trained and tested seven times with different data sets. The reason for utilizing user datasets as folds in cross validation is that the generalization capability obtained better simulates the performance in a real public space usage scenario, where it is impossible to include all users in the training set. The results are presented in figure 14.

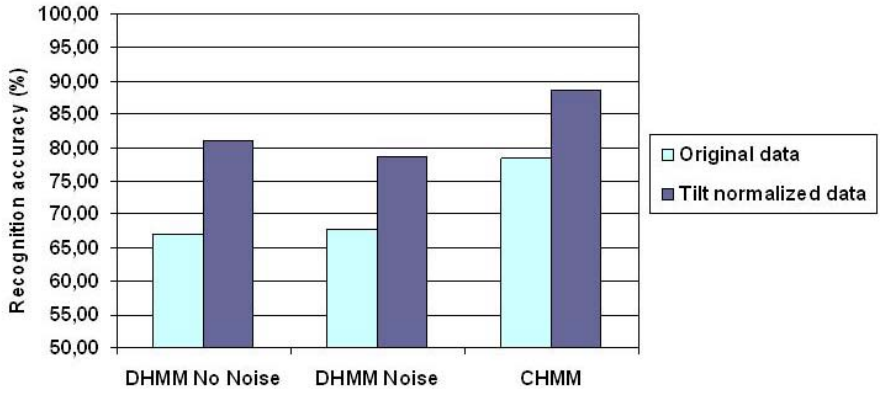


Figure 14. User-independent gesture recognition performance with discrete and continuous HMM.

The best recognition performance of 89% is obtained with cHMM and with tilt normalization. The results obtained with dHMM with or without noise distorted duplicates are on the same level, ~80%. The effect of using tilt normalization is notable: it increases user-independent gesture recognition performance by 10-15% depending on the method used.

6. Conclusions and Future Work

Accelerometer-based gesture control as a novel interaction modality to enable a variety of multimedia applications for small handheld devices was presented. The challenges regarding the development of user-dependent and independent gesture control via prototypes were described. The prototype application for a personal device was a customization application for mobile devices. It enables, for example, browsing of an image album in a smart phone or in home media terminal with gestures. In addition, gestures similar to throwing and catching can be used to initiate sending and getting pictures to/from an external device, e.g. home media terminal. Two public device prototype applications for National Science Centers were explained. Public space applications include the gesture control of DVD-based multimedia presentation on a public screen and the gesture control of lights. Furthermore, the concept of visualizing gestures was discussed and methods for performing the gestures according to three-dimensional accelerometer signals were provided.

We experimented with methods for user-dependent gesture recognition with a low amount of training repetitions, and for feasible user-independent gesture recognition from a moderately large set of gestures. The experiments with 18 different gestures from seven users, and the results with both user-dependent and independent cases were presented in detail.

The user-dependent gesture recognition performance of cHMM was better when compared to any optimised configuration of discrete HMM. With three gesture repetitions in a training set, for cHMM the recognition accuracy level of 95% was obtained with or without tilt normalization, while for dHMM the best recognition accuracy of 90% was obtained in tests with noisy data.

The user-independent gesture recognition performance of 89% with cHMM was better when compared to tests with dHMM, both of which obtained with cross-validation from 2,520 gestures. An important result is that the effect of using tilt normalization is notable in increasing the user-independent gesture recognition performance by 10-15% depending on the method used. Moreover, the user independent recognition accuracy is very likely to further increase when the user independent gesture models are trained with gestures from more than six users.

The amount of different gesture classes, 18, was high for acceleration-based discrete gesture control. Such a large amount of different gesture classes was chosen to emphasise the differences in recognition accuracy between the methods. In practice, a set of less than ten gestures is often enough for controlling a single multimedia application, such as a DVD player or an image album. Furthermore, when the amount of different gesture classes is reduced, the error is also reduced in relation to the amount of different gestures. For practical use of gesture control, the results are thus very promising.

In conclusion for user-dependent gesture control, we declare that the excellent results obtained with cHMM facilitate quick and effortless customization. In the case of user-independent gesture control, the results are also promising and a good gesture control experience can be provided for large user groups. Future work includes studying the effect of additional or alternative sensors, e.g., gyroscopes, for the recognition accuracy and visualisation feasibility. Furthermore, the feedback from

the gesture interface should be studied, e.g., using means like vibration or audio, and various multimedia applications must also be studied from usability viewpoint.

Acknowledgements

We gratefully acknowledge the support of our partners in the Nomadic Media - ITEA project and Tekes for the funding.

References

1. Flanagan J, Mäntyjärvi J, Korpiaho K, Tikanmäki J (2002). Recognizing movements of a handheld device using symbolic representation and coding of sensor signals. Proceedings of the First Intl. Conference on Mobile and Ubiquitous Multimedia, pp. 104-112.

2. Gersho A, Gray R.M (1991). Vector Quantization and Signal Compression. Kluwer.

3. Hoffman F, Heyer P, Hommel G (1997). Velocity Profile Based Recognition of Dynamic Gestures with Discrete Hidden Markov Models. Proceedings of Gesture Workshop ’97, Springer Verlag.

4. Juang, B. H., “Maximim likelihood estimation for mixture multivariate stochastic observations of Markov chains”, AT&T Tech. J., vol. 64, no. 6, pp. 1235-1249, July-Aug., 1985.

5. Juang, B. H., Levinson, S. E., Sondhi, M. M., “Maximum likelihood estimation for multivariate mixture observations of Markov chains”, IEEE Trans. Informat. Theory, vol.IT-32, no. 2, pp. 307-309, Mar., 1986.

6. Kallio S, Kela J, Mäntyjärvi J (2003). Online Gesture Recognition System for Mobile Interaction. IEEE International Conference on Systems, Man & Cybernetics, Volume 3, Washington D.C. USA pp 2070-2076.

7. Kay S. (2000) Can detectability be improved by adding noise? IEEE Signal Processing letters, Vol 7 No 1. pp. 8-10.

8. Kela, J. Korpipää, P. Mäntyjärvi, J. Kallio, S. Savino, G. Jozzo, L. Di Marca, S.“Accelerometer based gesture control for a design environment”, Personal and Ubiquitous Computing, Springer, 2005, In Press.

9. Korpipää, P., Malm, E., Salminen, I., Rantakokko, T., Kyllönen, V., Känsälä, I. (2005). Context Management for End User Development of Context-Aware applications. Proc. Mobile Data Management MDM'05. To appear.

10. Liporace, L.A., “Maximum likelihood estimation for multivariate observations of Markov sources”, IEEE Trans. Informat. Theory, vol. IT-28, no. 5, pp.729-734, 1982.

11. Mäntyjärvi , J., Kela, J., Korpipää, P., Kallio, S., Enabling fast and effortless customisation in accelerometer based gesture interaction, In International Conference on Mobile and Ubiquitous Multimedia (MUM), Maryland USA, 2004, pp. 25–31.

12. Mäntylä V-M, Mäntyjärvi J, Seppänen T, Tuulari E (2000). Hand Gesture Recognition of a Mobile Device User. Proceedings of the International IEEE Conference on Multimedia and Expo, pp. 281-284.

13. Rekimoto J (2001). GestureWrist and GesturePad: Unobtrusive Wearable Interaction Devices. Proceedings of the Fifth International Symposium on Wearable Computers, ISWC 2001.

14. Sawada H, Hashimoto S. (2000). Gesture Recognition Using an Accelerometer Sensor and Its Application to Musical Performance Control. Electronics and Communications in Japan Part 3, pp 9-17.

15. Tuulari E, Ylisaukko-oja A (2002). SoapBox: A Platform for Ubiquitous Computing Research and Applications. First International Conference, Pervasive 2002, pp. 26-28.

16. Tsukada K, Yasumura M (2002). Ubi-Finger: Gesture Input Device for Mobile Use. Proceedings of APCHI 2002, Vol. 1, pp 388-400.

17. Wilson A, Shafer S (2003). Between u and i: XWand: UI for intelligent spaces. Proceedings of the conference on Human factors in computing systems, CHI 2003, April 2003. pp 545-552.

18. Yoon H.S (2001). Hand Gesture Recognition Using Combined Features of Location, Angle and Velocity. Pattern Recognition 34, pp 1491-1501.