Pers Ubiquit Comput (2014) 18:191–203
DOI 10.1007/s00779-013-0637-3

ORIGINAL ARTICLE

Airwriting: a wearable handwriting recognition system

Christoph Amma · Marcus Georgi · Tanja Schultz

Received: 1 October 2012 / Accepted: 21 November 2012 / Published online: 24 February 2013

Abstract We present a wearable input system which enables interaction through 3D handwriting recognition. Users can write text in the air as if they were using an imaginary blackboard. The handwriting gestures are captured wirelessly by motion sensors applying accelerometers and gyroscopes which are attached to the back of the hand. We propose a two-stage approach for spotting and recognition of handwriting gestures. The spotting stage uses a support vector machine to identify those data segments which contain handwriting. The recognition stage uses hidden Markov models (HMMs) to generate a text representation from the motion sensor data. Individual characters are modeled by HMMs and concatenated to word models. Our system can continuously recognize arbitrary sentences, based on a freely definable vocabulary. A statistical language model is used to enhance recognition performance and to restrict the search space. We show that continuous gesture recognition with inertial sensors is feasible for gesture vocabularies that are several orders of magnitude larger than traditional vocabularies for known systems. In a first experiment, we evaluate the spotting algorithm on a realistic data set including everyday activities. In a second experiment, we report the results from a nine-user experiment on handwritten sentence recognition. Finally, we evaluate the end-to-end system on a small but realistic data set.

C. Amma (✉) · M. Georgi · T. Schultz
Cognitive Systems Lab, Institute for Anthropomatics,
Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany
e-mail: christoph.amma@kit.edu

M. Georgi
e-mail: marcus.georgi@student.kit.edu

T. Schultz
e-mail: tanja.schultz@kit.edu

Keywords Handwriting recognition · Wearable computing · Gesture recognition · Inertial sensors · Hidden Markov models

1 Introduction

Gestures facilitate new forms of user interfaces, which will be particularly suited for mobile and wearable computer systems. Rather than forcing a user to manually operate a device, hand gestures allow operation without the need to focus on tiny screens and keys while leaving the hands free for other tasks. Various sensing techniques (e.g. cameras, inertial sensors) are traditionally used for the purpose of gesture recognition [19]. Accelerometers are especially appealing for mobile usage because of their small size, low cost and robustness. Research so far mostly concentrated on the recognition of a limited set of predefined single gestures, which are then assigned to commands. This limits the number of possible commands to the number of recognizable gestures. However, operations like the input of text or other complex operations require more expressive power than a small set of isolated gestures can offer.

Our approach combines the mobility and intuitivity of gestures with the expressiveness of handwriting. This paper describes a wearable handwriting recognition method based on inertial sensors that allows for spotting and continuously recognizing whole sentences written in the air, which is a special type of gesticulation. The approach comprises two main challenges. Firstly, in a real-world application scenario, the system will continuously measure hand motion, but only small portions of the signal will actually contain handwriting. The majority of the data will contain all sorts of everyday motions which are not relevant to the text input interface. Consequently, the relevant

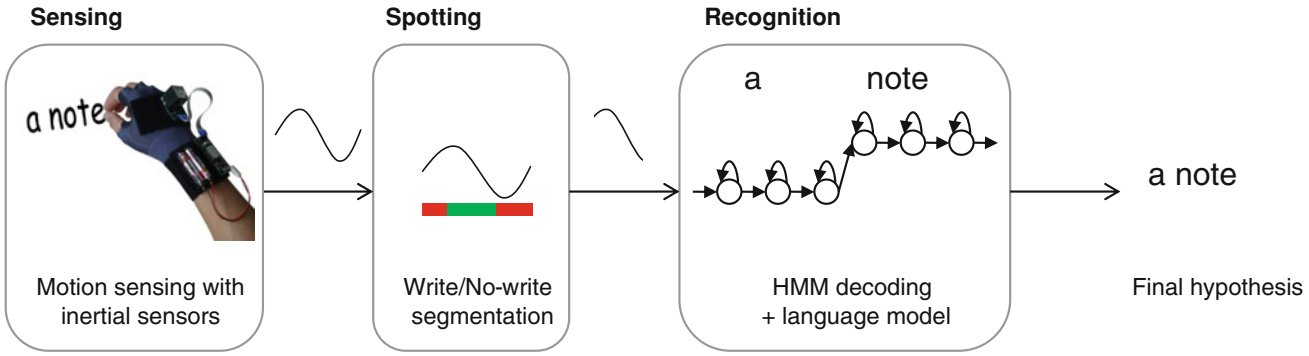


Fig. 1 Overview of the data processing pipeline with the individual stages of the system. After acquisition, the spotting stage detects parts that likely contain handwriting motion. The spotted segments are passed on to the recognition stage, where they are decoded by HMMs in combination with a language model to produce the final hypothesis

segments need to be automatically spotted from the data stream. Secondly, the text corresponding to the handwriting signal must be recognized from the sensor signals. We implemented a two-stage approach for the spotting and recognition task which is illustrated in Fig. 1. In the spotting stage, we use a support vector machine (SVM) to discriminate motion that contains handwriting from motion that does not. In the recognition stage, we use hidden Markov models (HMMs) in combination with a statistical language model to recognize the written words. We show that the performance of the spotting stage can be further improved by filtering false positives in the recognition stage.

While our current system is focused on the recognition of text from continuous gestural handwriting, it provides a proof-of-concept system for any sort of gesture recognition system that needs to continuously recognize gestures which are composed of an alphabet of primitive gestures.

The paper is organized as follows: in the remainder of this section, we provide an overview on related work. In Sect. 2, we briefly describe the hardware we have used to sense hand motion. In Sect. 3, we describe our handwriting spotting approach along with the experimental evaluation. In Sect. 4, we describe the handwriting recognition and the experiments performed to evaluate the recognizer. In Sect. 5, we evaluate the end-to-end system. Finally, we conclude the paper in Sect. 6.

1.1 Related work

The question on how to interact efficiently and intuitively with wearable and ubiquitous computer systems has led to multifaceted approaches. Besides the widely used soft keyboards in current smartphones, various methods have been proposed which use only few keys to reduce size and to allow one-handed usage [15, 16]. Recently, research investigates alternative interaction paradigms for mobile computing by allowing free-hand operation. Gestures are used to develop interfaces that do not require hand-held devices and therefore allow seamless integration into people's everyday activities. For example, miniature projectors display the screen on any surface in front of the user and cameras track hands for gestural interaction [18, 28]. Other researchers propose to drop screens altogether and show that spatial interaction is possible without visual output. The user builds a mental representation of an imagined screen [8]. We follow the same idea by assuming that handwriting can be reliably produced and recognized without any visual or haptic feedback.

The spotting of gestures requires to automatically identify relevant signal segments in the continuous data stream. This can either be done by first applying a binary classifier to detect segments which likely contain a relevant gesture and classify the gesture afterward [11, 23]. Or it can be done by continuously trying to classify the incoming data and reject any results that fall below a given probability threshold [14, 26]. HMMs have been used either as the second stage [11] or directly by making use of their implicit segmentation abilities [14].

The field of gesture recognition with accelerometers has been extensively studied in the past. Usually, a number of isolated gestures ranging from 10 to 30 are defined and classified [6, 12]. Accelerometers integrated in watches are used by Hein et al. [9] for the purpose of gesture recognition for HCI and by Amft et al. [1] to control the watch itself by gestures. Kim et al. [5, 13] propose a system for single digit and character gesture recognition based on an algorithm for reconstructing the 3D trajectory from the acceleration sensor signal. The method is able to compensate some of the problems which arise from the accumulation of sensor errors but it will only give reasonable results for short periods of time and does not scale up to continuous recognition of handwriting.

In traditional pen-based online handwriting recognition, HMMs are widely used based on features extracted from pen strokes [21]. In our case, the strokes are not available

and the specialized methods developed in traditional handwriting recognition cannot be transferred to our task. McGuire et al. [17] use HMMs for continuous mobile American sign language recognition with a 40-word vocabulary, modeling each word by one HMM. Sign language recognition entails additional challenges because both hands and also facial expressions are used and the underlying sign alphabet is more complex than the latin alphabet.

2 Hardware and sensing

We developed a hardware system, which is illustrated in Fig. 2. An Analog Devices Inertial Measurement Unit (ADIS16364) is attached to the back of the hand using a thin glove. A board with a microcontroller and a Bluetooth module, as well as the power supply are mounted on a wristlet. The sensor contains one 3D accelerometer and one 3D gyroscope, both sampled at 819.2 Hz, which is the maximum rate of the sensor. We denote the acceleration by $\boldsymbol{a} = (a_x, a_y, a_z)$ and the angular rate measured by the gyroscopes by $\boldsymbol{g} = (g_x, g_y, g_z)$. The sensor samples of the signal time-sequence are described by

$$\boldsymbol{a}(i) = (a_x(i), a_y(i), a_z(i)), \quad i = 1, 2, \ldots, N$$

and

$$\boldsymbol{g}(i) = (g_x(i), g_y(i), g_z(i)), \quad i = 1, 2, \ldots, N$$

where N is the total number of samples. We denote the complete sequence by

$$\boldsymbol{s}(i) = (\boldsymbol{a}(i), \boldsymbol{g}(i)), \quad i = 1, 2, \ldots, N.$$

While the design goals of this prototype were ease of development and integration, further miniaturization could be achieved by using off-the-shelf components. For example, all components could be integrated in an unobtrusive wristlet.

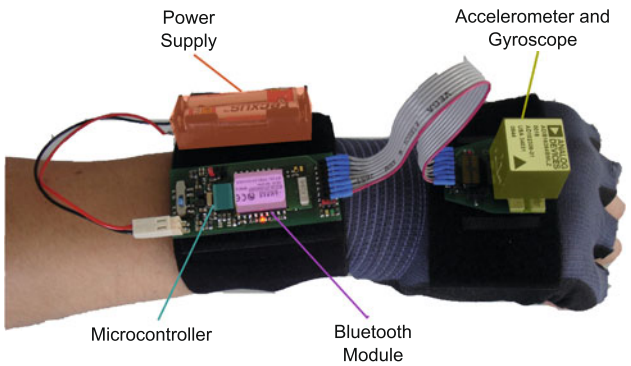


Fig. 2 Prototypical wireless data glove for signal acquisition

3 Spotting of handwriting

The spotting stage is used to perform a binary classification of the data stream into segments that likely contain handwriting and segments that do not (Fig. 3). The segments classified as potential handwriting motion are passed on to the recognition stage. Ideally, the spotting algorithm imposes a minimal processing delay and identifies all handwriting segments without producing a high amount of false positives. Figure 4 illustrates the challenges of this task. It shows the acceleration and angular rate signals of a 19-min-long recording with three handwriting segments. The objective of the spotting stage is to discriminate these segments from the background activity. We use a binary SVM classifier with an RBF kernel ($\gamma = 8$, $C = 32{,}768$) to discriminate non-writing signal segments from potential writing segments. In order to allow real-time usage and operation on continuous data streams, we use a sliding window approach. Individual overlapping windows are classified and the classification results of all overlapping windows are then combined and passed on to the recognition stage immediately. Figure 3 shows the architecture of the spotting stage. Successive windows are illustrated in the middle of the figure as horizontal bars. The color indicates the classification result, green for segments classified as handwriting, red for others. On every window w the following features are computed for classification, where N_w denotes the number of sensor samples per window and $\boldsymbol{a}_w$ and $\boldsymbol{g}_w$ denote the acceleration and angular rate samples in the window:

- Average angular velocity

$$\frac{1}{N}\sum_{i=1}^{N} \|\boldsymbol{g}_w(i)\|_2$$

- Average mean shifted acceleration

$$\frac{1}{N}\sum_{i=1}^{N} \|\boldsymbol{a}_w(i) - \bar{\boldsymbol{a}}_w\|_2$$

- Distribution of power per frequency between 0 and 8 Hz in 8 bins

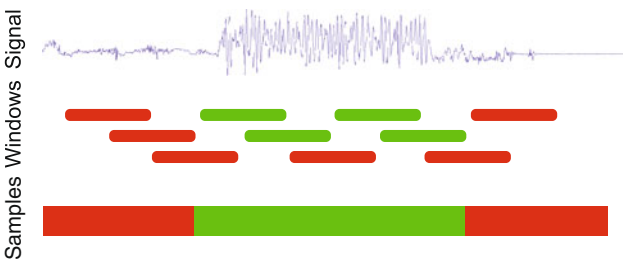


Fig. 3 Schematic illustration of the spotting stage

We chose these features due to a visual inspection of the signal statistics in Fig. 5. Handwriting parts have a high frequency and amplitude compared to the non-writing parts in both acceleration and angular rate. A frequency peak is present at around 3 Hz, depending on the writing speed. Therefore, we chose features representing angular rate and acceleration amplitude as well as the frequency distribution.

Defined by the window and shift length, every single sensor sample is part of several windows. For every win-

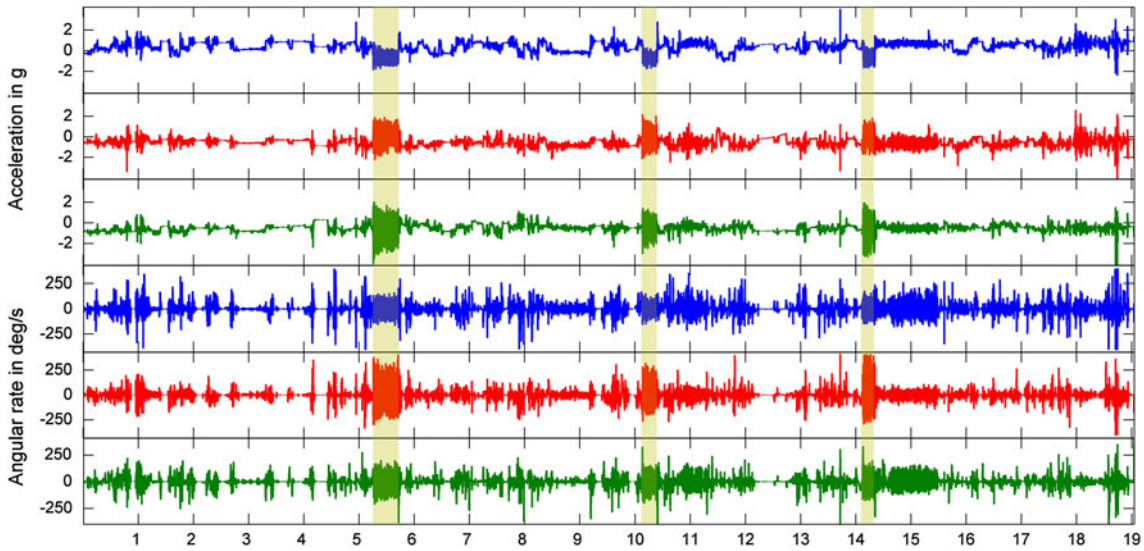
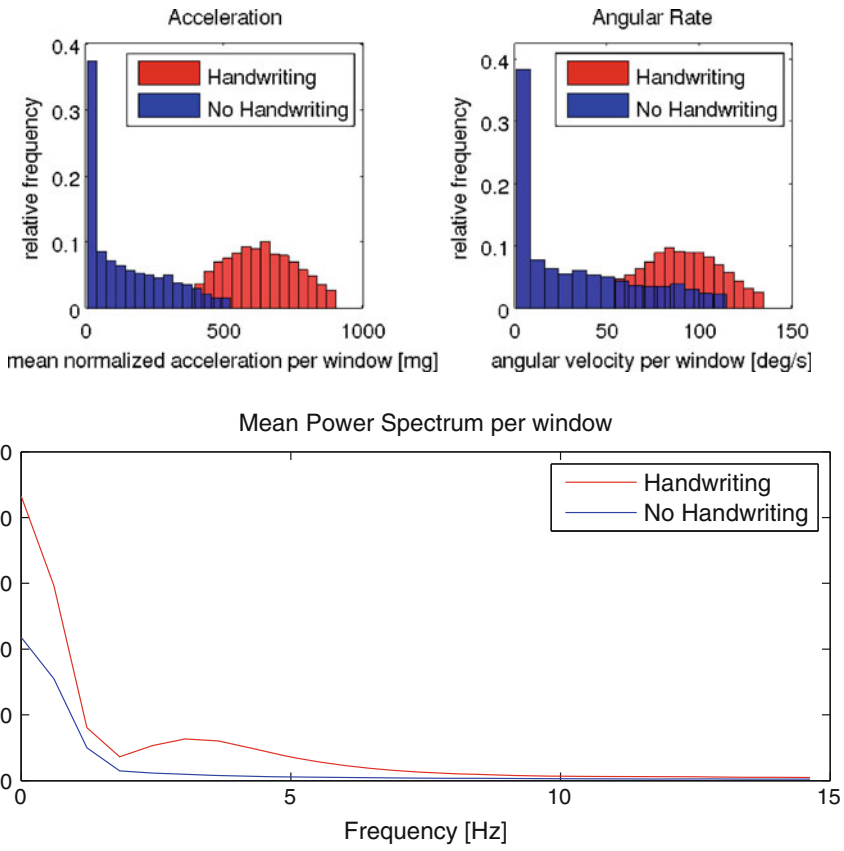


Fig. 4 Example recording acquired for the evaluation of the spotting stage (*x*, *y*, *z* axes are shown ordered from *top* to *bottom* for acceleration and angular rate respectively). The three segments show the handwriting parts; the non-highlighted signals represent background activity

Fig. 5 Comparison of features extracted from recordings with and without handwriting. Acceleration and angular velocity are distributed differently and the power per window displays different frequencies



dow w_t, the SVM Classifier $C_{\mathrm{SVM}}(w_t)$ returns 0 (red) if no handwriting is detected and 1 (green) otherwise. This is illustrated by the color of the horizontal bars in Fig. 3. A sensor sample is then classified by the classifier C_{COMB} which combines the results of the individual SVM classifiers. One sensor sample $\boldsymbol{s}(i)$ is classified as writing motion if at least one window containing $\boldsymbol{s}_t$ is classified as handwriting motion according to:

$$C_{\mathrm{COMB}}(s(i)) = \max_{k:s(i)\in w_k} C_{\mathrm{SVM}}(w_k) \quad (1)$$

A sequence of sensor samples classified as handwriting is called a handwriting segment. The lower bar in Fig. 3 shows the resulting signal segmentation (one segment) after combining the results from the individual classification results. In contrast to other classifier combination schemes (e.g. majority vote), our combination scheme has a bias toward the detection of handwriting motion, i.e. some samples not belonging to actual handwriting might be classified as handwriting. On the one hand, this approach guarantees to not loosing any important samples, since all handwriting samples that are missed in the spotting stage are never forwarded to the recognition stage and therefore lost. Furthermore, short pauses between writing periods will not lead to gaps in the detected handwriting segment. On the other hand, this approach may incorrectly forward several non-handwriting segments to the recognition stage. However, these segments are usually very short and the recognition stage therefore produces empty or very short hypotheses (0–3 characters in total). Under the assumption that hypothesis containing no or very few characters are not valid, we implemented an additional filtering step based on the length of the hypotheses produced in the recognition stage. All hypotheses up to a given length are discarded. We present the performance and show the effectiveness of this filtering step for different length settings in 3.2. Consequently, most of the false positives have no negative impact on the final recognition results. A sensor sample is passed on to the recognition stage as soon as the last window of that sample was classified. As a result, the spotting stage imposes a delay of only one window length on further processing. Choosing the window length is always a trade-off between classification accuracy and system delay.

3.1 Experiments

We collected data from 3 subjects to evaluate our proposed spotting approach. We assume that the inter-individual differences are not as important for the discrimination of handwriting and non-handwriting data as the variety of recorded activities. Therefore, we did not focus on a data corpus including many subjects. Instead, we gave priority to collecting realistic everyday data. Table 1 summarizes the recorded data used for the experiments. The handwriting training data were taken from the corpus collected for the recognition experiments (Sect. 4). The other data sets (testing and non-writing data) were recorded during typical activities at home (preparing food, eating, doing laundry, ...). The test data set consists of sessions containing single sentences of airwriting at random points in time. The test data set is imbalanced because of the realistic setting. A user will only write occasionally during everyday activities. In total, 17 sentences containing 68 words were written. Shortly, after writing the sentence, subjects made a paper note with the current time and the words they just wrote in the air. This information was used to annotate the recordings in order to generate an approximate ground truth. We then performed a Viterbi alignment of the reference sentence with the selected segment to get a more precise approximation of the actual start and stop time of the handwriting motion as ground truth.

Table 1 Data corpus used for the spotting experiment, the data sets are disjoint

Data set	Handwriting (min)	No writing (min)	Total (min)
Training	272	111	383
Testing	4	111	115

3.2 Evaluation

To evaluate the performance of the spotting algorithm, we compute recall, precision, specificity and F-score on the individual samples of the test data set using different window sizes and overlaps. As already stated in Sect. 3, our spotting approach is biased toward the recognition of handwriting, and thus, we expect a high recall and low precision. Since our data set contains significantly more non-handwriting data than handwriting data, we compute the specificity, which gives information of how many of the non-handwriting samples were correctly classified. This number is proportional to the fraction of time during which no handwriting occurred and no handwriting was spotted.

Figure 6 shows that a very high recall of up to 99 % is achievable with various combinations of window sizes and shift widths. A small shift width between windows, resulting in a larger overlap, yields the highest recall (see Eq. 1), since the combined classifier C_{COMB} incorporates all windows a sensor sample is part of. In contrast, precision and specificity can be optimized by choosing a rather small window size and a large shift between windows. Using a window size of 500 frames or 0.61 s and an overlap of half the window size, we achieve the highest precision and specificity of 35 and 92 %, respectively, and a recall of 97 %. The highest recall of 99.9 % at a window size of 700

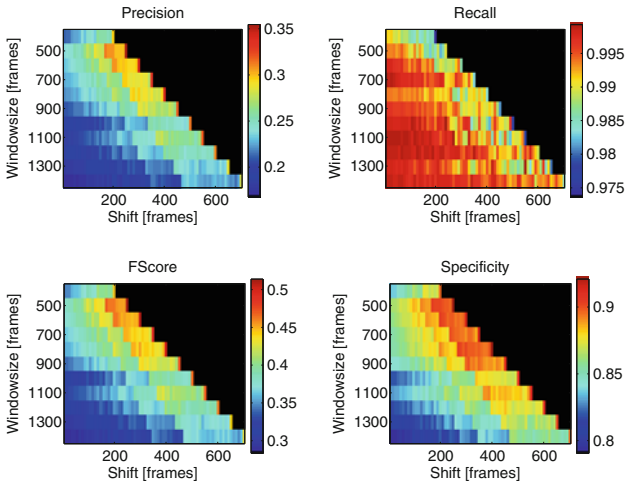


Fig. 6 Precision, recall, *F*-score and specificity in spotting stage experiments for different window sizes and frame shifts

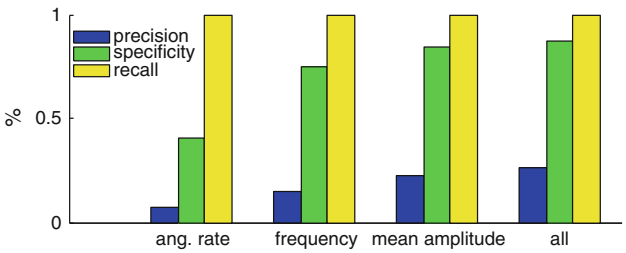


Fig. 7 Evaluation of spotting experiments for different feature sets

Table 2 Percentage of filtered false positives

N	0 (%)	1 (%)	2 (%)	3 (%)
Filtered	91.3	98.2	99.3	99.5

frames or 0.85 s and a shift of 10 frames (0.012 s) goes along with a precision of 23 % and a specificity of 85 %. As a high recall is crucial and low precision is acceptable due to the described further filtering in the recognition stage, we chose a compromise biased toward recall for further experiments. With a window size of 700 frames and a shift of 140 frames (0.17 s), we get a recall of 99 %, i.e. we miss very few handwriting samples. While a precision of 26 % seems rather low, these results are still well suited for our application. The specificity value shows that in 88 % of the time where no handwriting occurs, the spotting stage does not forward segments to the computationally costly recognition stage. We also evaluated the impact of the individual feature settings to the final result. Figure 7 shows the values for recall, precision and specificity for the individual feature settings (angular rate, frequency, amplitude and their combination). All three features contribute relevant information to the classification task.

As already stated, false positive segments are typically very short. On our data set, the mean length of false positive segments is 1.58 s with standard deviation of 1.64 s, for true positives, the mean is 15.34 s with standard deviation of 4.52 s. We compare these values with the average time it takes to write a character, which we computed on the sentence data set (see Sect. 4.1). The average time per character over all users is 0.83 s with a standard deviation of 0.17 s. That means the false positive segments are as short as the time a user needs to write two to three characters. To evaluate the effectiveness of the described filtering in the recognition stage, we passed all the false positives to the recognition stage and discarded all hypothesis containing at most *N* characters in total. Table 2 shows the number of filtered false positives depending on the value of *N*. The results show that this is a reasonable approach.

4 Handwriting recognition

In the recognition stage, a given input signal sequence is decoded into a sequence of words. Recognition results are output on word level, i.e. a set of words, called the vocabulary, defines all words that can be recognized. Text written in the air is defined by the spatial trajectory of the hand movement as this is the case for 2D pen-based handwriting. Unfortunately, the actual 3D trajectory is not available. It is theoretically possible to reconstruct the trajectory by applying a strapdown inertial navigation algorithm. Angular rate is integrated once to get the orientation. When the orientation is known, the earth acceleration can be subtracted from the accelerometer measurements and a final double integration yields the trajectory. However, sensor drift and noise lead to errors which accumulate over time due to the triple integration and increase rapidly over seconds [29]. Consequently, we rely in our approach on the raw sensor signals. The pattern matching problem can be formulated to find the most probable word sequence given the input acceleration and angular rate signals. This is one major difference to traditional 2D pen-based handwriting recognition. The other difference is the lack of the pen-up and pen-down information, which if present, would provide a natural segmentation of the writing into isolated strokes. In the case of 3D-space handwriting, there is only one continuous stroke without any segmentation information. As a result, we cannot apply the feature extraction techniques which have been developed in the field of pen-based online handwriting recognition, since they are derived from the 2D trajectory and make use of the pen-up and pen-down information [21]. Figure 8 shows the characteristics of 3D-space handwriting. In Fig. 8a, a possible trajectory is illustrated. The segments normally not written (pen-up movements) are shown in red. In Fig. 8b, the actual acceleration signals of the word “HAND” written in the air are shown.

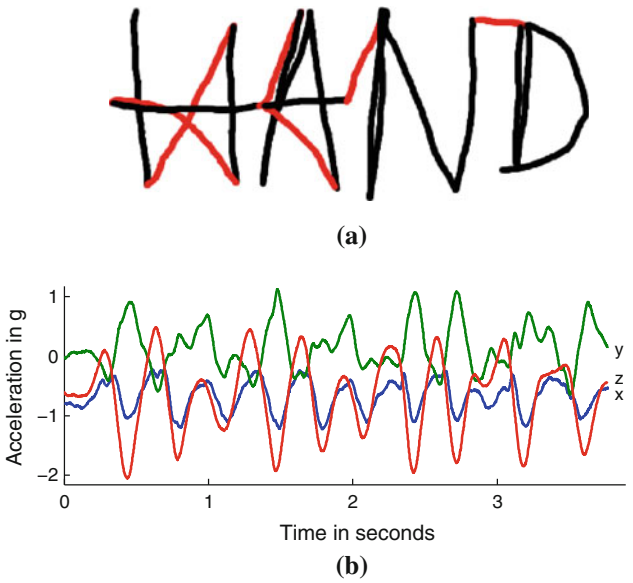


Fig. 8 Word "HAND". **a** Example possible trajectory, *red* segments would be pen-up movements in pen-based handwriting. **b** Actual acceleration signals (color figure online)

Figure 9 illustrates the pattern recognition problem for a complete sentence. For a given finite input sequence, the most likely sequence of words must be found. Words are formed by a sequence of characters which are predefined by a given alphabet. Neither the number of words, nor the boundaries of words or characters are known in advance. We constrain the problem by only allowing words from a predefined list of vocabulary words to be recognized. Furthermore, we make use of the inherent structure of language and use a statistical language model to favor word sequences which are common in the target language.

Hidden Markov models are the state-of-the-art technique to solve such problems and are widely used in speech and handwriting recognition. They have several properties that make them particularly suited for the problem presented here:

- Longer models can easily be extended from existing models by adding transitions between the end state of one model and the start state of the other model. This allows to construct models for arbitrary words based on 26 character models for all uppercase characters. Once the parameters of these 26 character models are trained, no further training of word models is necessary.
- It is not necessary to know the boundaries of the characters in the signal in advance. By performing a Viterbi decoding, these boundaries are found implicitly.
- Advanced and efficient decoding techniques exist to handle large search spaces typically encountered with large vocabularies.
- It is possible to decode the signal in almost real-time while it is received.

4.1 Data acquisition

This section describes the data sets collected for the handwriting recognition experiments. In total, 23 subjects contributed handwriting data to this study. We collected three different data sets, which are listed in Table 3. Data set D_C contains isolated characters. Nine subjects contributed 25 times the 26 characters of the alphabet each. Data set D_W contains isolated words. Five subjects contributed 99 words each. The length of the words is equally distributed between two and eleven. The words were randomly chosen from a list of frequent English words maintained by the University of Leipzig.[1] Data set D_S contains individual sentences. Nine subjects contributed 80 English sentences each. The sentences were selected such that each character of the alphabet appeared multiple times.

The recording setup was the same for all data sets. The subjects were sitting in a chair in front of a computer screen showing the next text to write. None of the subjects had used the system before. They were told to write in the air in front of them as if they would write on an imaginary blackboard. Instead of writing horizontally from left to right, the subjects were asked to write in place. All subjects were told to write with keeping the wrist fixed, using block capital letters and an approximate height of 20 cm per character. No further constraints on speed or quality of writing were set. Due to the missing visual feedback and the unusual writing, it is harder to write without errors but a writer normally realizes if he or she makes an error. The subjects were told to correct their writing errors themselves by repeating the respective character, word or sentence. The segmentation of the data was done by the participants with a key press before and after each character, word or sentence, i.e. participants were able to make pauses between recordings.

4.2 Recognition system

This section describes in detail the recognition system we used. This includes modeling, initialization, training and decoding of the HMMs as well as the vocabulary and language model.

4.2.1 Modeling

We use continuous density HMMs to model the signal sequences of each character. HMMs are state-based statistical models that are used to model signal sequences. A detailed introduction to HMMs is beyond the scope of this paper, the classic paper by Rabiner [22] gives an overview of theory and application. One important property of

[1] wortschatz.uni-leipzig.de/html/wliste.

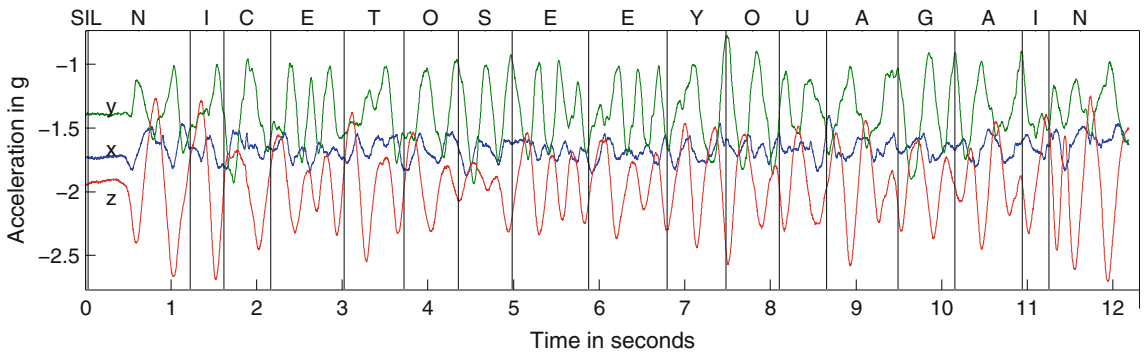


Fig. 9 Alignment of the characters of the sentence "Nice to see you again" to the respective accelerometer signals (angular rate signals are not shown). The alignment was acquired by a Viterbi decoding

Table 3 Data corpus for the recognition stage

	Type	Sent.	Words	Char.	Hours
D_C	Char.	0	0	5,850	2:25
D_W	Word	0	495	3,695	0:57
D_S	Sent.	720	3,294	16,002	3:56

HMMs is the possibility to concatenate individual models to form a new HMM. This concatenated HMM then models the complete signal sequence of the two individual models. We are therefore able to use a small set of primitive character models and construct an arbitrary number of word models by concatenation of character models. We use 26 character models for the capital letters from A to Z, one model to represent repositioning motion that occurs between single characters and one model to represent the state of no motion, e.g. pauses. All character HMMs have a left–right topology with 30 states, the repositioning model with 10 states and the "no motion" model with 1 state. Each state has a self-loop but there are no skip states, i.e. the complete model must be traversed. Transition probabilities are not trained. The observation probabilities for each state are modeled with Gaussian mixture models with six components each and diagonal covariance matrices.

4.2.2 Initialization

The aim of this step is to find meaningful initial parameter values for the GMMs. This is done by a flat start procedure, i.e. assigning each feature vector to one state of the HMM and afterward cluster the set of collected feature vectors into as many clusters as the number of components of this states GMM. Given an input signal and its corresponding HMM consisting of N states, the signal is partitioned in N parts of the same size and all feature vectors of the nth part are assigned to the nth state. This procedure is repeated for all training samples and feature vectors are accumulated for each state over all samples. Let G be the number of GMM components per state, then, the accumulated feature vectors are clustered into G clusters and each cluster defines one Gaussian by its mean and variance vectors (covariances are set to 0). The clustering is performed with the neural gas algorithm, a more robustly converging variant of k-means. The weights of the GMM components are uniformly set to one during initialization. This method gets more inaccurate, the more characters are encoded in the signal sequence. Therefore, we initialize the HMMs on the isolated character data set D_C, i.e. each input signal represents only one written character.

4.2.3 Training

After initialization, the GMM parameters are optimized with the Baum–Viterbi algorithm, a faster and more stable variant of the Baum–Welch-based EM training [7]. We perform the training procedure subsequently on all three data sets. The models are first trained on isolated character data (data set D_C), second on isolated word data (data set D_W) and finally on whole sentence data (data set D_S). The training on sentence data is done together with the evaluation in a cross-validation setup. We perform ten iterations of Baum-Viterbi training on D_C, one iteration on D_W and three iterations on D_S which sums up to 14 training iteration in total on data of growing complexity. The chosen numbers gave the best results.

4.2.4 Decoding

All possible word combinations, including repetitions, constitute the actual search space of the recognition problem, which therefore is several orders of magnitude larger than the size of the vocabulary. For example, the sentences in our data corpus contained 4.5 words in average and the largest vocabulary used contained 8,231 words. Considering the size of our vocabulary, there exist $8{,}231^4 = 4.5 \times 10^5$ possible sentences with 4 words. Since the number of words

is not known in advance, the search space is even larger. Clearly, it is not feasible to construct one HMM for each of all the possible word combinations and perform a complete Viterbi alignment for each to find the most likely word sequence. We use a Viterbi beam search on a prefixtree search graph to turn this into a computationally feasible problem.

A prefixtree search graph exploits the fact that a lot of words in the vocabulary share the same prefixes, and therefore, the decoding for the shared prefix needs to be evaluated only once for all these words. The prefixtree search graph is built by merging the character models of common word prefixes for all words of the vocabulary [10]. For example, the words "then" and "that" share the prefix "th", so the sub-HMM for these two characters needs to be evaluated only once. Figure 10 shows an example prefixtree for the three words "THEN", "THAT" and "TEA". Each of the nodes in the shown graph represents one-character HMM. Due to the number of characters in the alphabet, the number of root nodes of the search graph is 26. All leaf nodes represent word endings, and from each leaf node, there is a path to all root nodes to allow the recognition of arbitrary sequences of words.

Instead of a complete Viterbi search, a time synchronous Viterbi beam search is performed. The sequence of feature vectors is processed sequentially by propagating hypotheses through the graph according to the Viterbi criterion. Every hypothesis corresponds to one possible path. Without restricting the number and probability values of the hypotheses, this equals a complete Viterbi decoding. To speed up the process of decoding a beam search is used, in which only hypotheses are kept whose probability value falls within a fixed beam. Therefore, only the most likely paths are followed at the expense of the possibility to loose paths that temporarily have a low probability [20].

A statistical language model, which models the probabilities of word sequences, is used to further restrict the search space to word sequences that occur in real-world text data. We use the Janus Recognition Toolkit (JRTk) [25], originally developed for speech recognition, to perform our experiments.

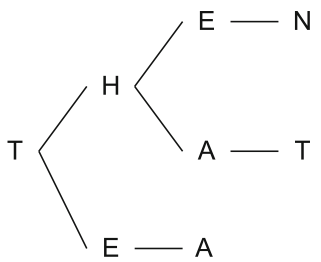


Fig. 10 Example prefixtree for three words. Every node in the *graph* represents one-character HMM

4.2.5 Vocabulary

The vocabulary defines all possibly recognizable words. We use two vocabularies of different sizes in our experiments to evaluate the impact of vocabulary size on the scalability. If the vocabulary increases, the recognition task gets harder, since the search space grows. The small vocabulary (V1k) contains 986 words and the large vocabulary (V8k) contains 8,231 words. Both vocabularies are taken from a list of frequent English words which is obtained by webcrawling and is freely available.[2] We applied additional cleaning steps on the list because it contains non-word entries. All entries consisting of only one character were deleted, except "i" and "a" (valid English one-character words). All entries consisting of two or three letters which either are not correct words (e.g. "gl") or describe abbreviations (e.g. "os") were deleted. All words longer or equal than four characters remained unchanged. Finally, all words contained in the sentences written by the subjects during the data collection were added to the vocabularies.

4.2.6 Language model

A statistical n-gram language model is used to model the dependencies between successive words in order to improve the recognition performance. Statistical language models provide information on the probability of a sequence of words and are normally generated data-driven from large text corpora. We use a trigram language model, which contains the probabilities of word sequences up to a length of three. Given a sequence of three words $w_1w_2w_3$, the language model returns the conditional probability $P(w_3|w_1w_2)$ of w_3, given the words w_1w_2 were recognized so far.

This probability is multiplied with the probability of the signal sequence given by the HMM whenever a complete word was recognized. The language model we use contains 60,000 words. It was generated for a speech recognition system by crawling English internet sites [24]. The n-gram coverage of the language model is given in Table 4. The unigram coverage for both vocabularies used in the experiments is 100 %, i.e. the language model contains all words in the vocabularies. The bi- and trigram coverage is lower, but if, for example, a trigram cannot be found, the language model falls back to bi- or uni-grams. We took the language model as is, i.e. we did not change it to optimize it toward the specific task described in this paper, like giving the word sequences occurring in our chosen sentences higher probabilities. This is reflected in the reported perplexity, which is given in Table 4. The perplexity is an

[2] wortschatz.uni-leizpig.de/html/wliste.

Table 4 Language model statistics

Coverage (1/2/3-gram)	100.00/97.98/64.13
Out-of-vocabulary rate	0.0
Perplexity	112

information theoretic measure to evaluate the predictive quality of a language model on a given set of word sequences. It can be interpreted as the average number of possibilities for the next word if the model had to choose uniformly and independently among all possibilities. The perplexity was computed using the SRI Language model toolkit [27]. We reported higher perplexities for the same language model in [3] because we formerly used a normalization method which was not appropriate for the task.

4.3 Experimental results

4.3.1 Performance metric

We measure the system performance by calculating the word error rate, which is the metric commonly used to evaluate the performance of speech and handwriting recognition systems. The hypothesis of the recognizer is aligned with the true reference sentence by computing the minimal edit distance (Levenshtein distance) on word level. This gives the number of substitution (S), insertion (I) and deletion (D) errors that occur by aligning hypothesis and reference. For a set of sentences $S = \{s_1, \ldots, s_k\}$ the word error rate is computed by

$$\mathrm{WER}_S = \frac{\sum_{i=1}^{k} S_{s_i} + I_{s_i} + D_{s_i}}{\sum_{i=1}^{k} N_{s_i}} \cdot 100,$$

where N_{s_i} is the total number of words in the reference sentence s_i. The following example illustrates WER computation for one single sentence:

```
Reference:  we had a lot of expertise
Hypothesis: he had   lot of expert ease
Error:      S      D        S      I
```

The word error rate in the above example is $\mathrm{WER} = (2+1+1)/6 \cdot 100 = 66\,\%$. One should notice that the WER can be larger than 100 % because the hypothesis can be longer than the reference.

4.3.2 Feature extraction

We use the normalized acceleration and angular rate as input features (observations) for the HMM. The normalization includes the compensation of orientation dependency caused by gravity and the compensation of variance in speed and size of the performed motion.

The gravitational acceleration is always present in the sensor signals as a constant vector approximately orthogonal to the earth surface with the absolute value of 1 g. Since we do not know the orientation of the sensor in the global earth reference frame, we cannot simply subtract the gravitational acceleration from the signals. We therefore assume that the sensor orientation does only change very little while writing. Under this assumption, the gravitational acceleration leads to a constant offset in all three acceleration sensor channels. We can then subtract the mean from the signal for each sentence to remove this offset. Clearly, this is a very rough approximation but our experiments show that it is a reasonable choice for our application, where the performed motion is primarily defined by its 3D trajectory and not by its change of orientation. In [6], a method to compute the actual orientation is used to delete gravity from the acceleration signal more accurately. The features normalized with the estimated orientation perform well on the proposed gesture recognition task but this comes at the cost of a calibration gesture which must be performed in regular intervals.

The scale and speed of the writing motion are not relevant to the actual handwriting information. Thus, the features should be independent of speed and scale. Both speed and scale of the handwriting are reflected in the amplitude and duration of the acceleration signals. To compensate the effects of different amplitudes, we normalize the variance of the signals by dividing the signal by its standard deviation. The effects of variance in duration are implicitly compensated by the self-transitions in the HMM states.

For real-time operation, the running mean and standard deviation can be computed. To compute the actual feature vector, the normalized signal is then segmented into windows of length 10 ms, and on every window, the average amplitude is computed for each channel. The resulting values are the features we use. For each window, we get a six-dimensional feature vector containing the averaged 3D acceleration $(\bar{a}_x, \bar{a}_y, \bar{a}_z)$ and the averaged 3D angular rate $(\bar{g}_x, \bar{g}_y, \bar{g}_z)$.

4.3.3 Experiment 1: person-independent

To evaluate person-independent performance a per-person cross-validation was performed. The data of one person were chosen as test set and the data of all other persons were taken as training set. As stated in Sect. 4.2, we took the initialized models from former experiments and performed three iterations of EM training. Table 5 shows the summary of cross-validation results. The average word error rate is 9 % for the small vocabulary and 11 % for the large one.

The comparison of the two vocabularies is promising. Vocabulary V8k is approximately eight times larger than

Table 5 Results of the person-independent evaluation with and without language model

	No LM		3-gram LM	
	V1k	V8k	V1k	V8k
Average WER (%)	37	49	9	11
SD WER (%)	17	19	8	9

V1k; however, the error rate on the large vocabulary is only by a factor of 1.2 higher than for the small vocabulary. Therefore, we assume our approach scales well to even larger vocabularies.

If we compare the results acquired by using the language model to the results without any language model, we see a significant boost in recognition performance. As explained in Sect. 4.2, the search space is very large, and if no language model is used, ambiguities between words cannot be resolved on the language level. Therefore, results are expected to be worse. Nevertheless, still more than 50 % of all words are correctly recognized. Considering the size of the search space, this is still a notable result and shows the discriminative quality of our HMM-based recognition system.

Figure 11 shows the breakdown of results for each subject individually. On the data of subjects D and I, the performance is very low compared to the average. From observation during the experiments, it is known that subject D did not keep the wrist fixed all the time while writing, i.e. there was more rotational motion in the wristjoint. As a result, the fraction of gravitational force in the sensor signals cannot be assumed constant during writing and the mean normalization of the signals does not remove the gravitational acceleration. This might account for the high word error rate.

For subject I, there is no obvious reason why the performance is so low. However, we showed in our past experiments that even for the case of block capital letters, people do write characters very differently. In the data set D_C, five different variants of writing an "E" were observed, mainly distinguished in the order and direction of strokes [2]. If one writer has a very different writing style compared to the majority, we would expect the system to have a much lower performance. This could be a possible reason for the performance drop for subject I. On average, the system achieves very promising accuracies. For some subjects, a word error rate of 2 % on the large vocabulary is achieved, which means that approximately 191 of 195 words were recognized correctly.

4.3.4 *Experiment 2: person-dependent*

Although our aim is to build a system that works independent of its user, we evaluated the person-dependent

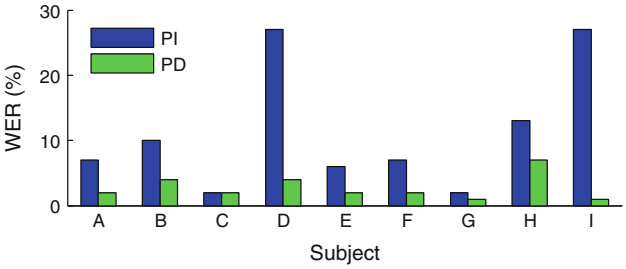


Fig. 11 Per-subject results for the person-independent (PI) and person-dependent (PD) case

performance. Firstly, this gives an upper bound to which the user-independent system can be compared to since we generally expect a higher performance if the system is specifically trained on a users own data. Secondly, the results give insight into what could be expected from runtime adaptation methods. Runtime adaptation was not used for the experiments described in this paper but might be a future option. Thirdly, the person-dependent case might be the standard usage scenario for personalized digital devices. The person-independent models from the experiments described in the former section were taken as initialization. The person-dependent performance of the system was evaluated on each subject data with a tenfold cross-validation on the 80 sentences of each user with one iteration of EM training. The large vocabulary (V8k) and the trigram language model were used for the evaluation. Figure 11 shows the results of the person-dependent evaluation for each subject together with the word error rate for the person-independent case. Subject-specific retraining of the models greatly improves recognition performance. The average word error rate drops from 11 to 3 %. Subjects D and I, for which the person-independent performance is rather low, also achieve very good performance after retraining.

4.3.5 *Qualitative feedback*

Although we did not explicitly collect user feedback on the system, we want to share our experience based on feedback we received during the collection of the data corpus. Generally, the data recording sessions were perceived to create fatigue. Almost all participants suffered from the "gorilla arm", an effect observed in interaction systems, e.g. vertical touchscreens, where users have to hold and move their arm horizontally for a longer period of time. This was expected, since our experimental setup forces the user into the exact same situation. In a real-life scenario, this situation will be different. Firstly, the user will not write for long periods of time since the system is not meant to write longer texts but for small notes, messages or commands. Secondly, it is not a requirement to hold the

arm horizontally during writing. Rather, the system works well with the arm hanging on the side of the body. It also still works well if the writing gets relatively small (below 10 cm) which makes it less fatiguing and less noticeable for others. Another possibility to reduce the motion amplitudes would be to allow the movement of the wrist, enabling a user to write solely with his hand without having to move his whole arm. However, this would obstruct the possible integration of the sensor in a wristlet because this is only possible if the relevant motion is performed by the forearm and not only the hand.

5 Combined evaluation

The main evaluation of the spotting and recognition approach was performed independently due to the lack of a large combined data corpus. A combined data corpus should include a high number of sentences of different subjects written in a realistic everyday life scenario with background activities that actually incorporate hand motion. Such a data corpus is currently not available. However, we performed a preliminary evaluation of the end-to-end system based on the data collected for the spotting experiments. The data corpus includes large parts of non-trivial background activity with occasional handwriting activity, only the number of subjects that contributed sentences and the total number of words is relatively small. We used a recognizer trained on all data sets, D_C, D_W and D_S, to evaluate the performance. This means the evaluation is not person-independent, since all of the three subjects that contributed to the spotting data also contributed data to the handwriting recognition data. The spotting data contain 68 words written in 17 sentences. To evaluate the performance, we feed all handwriting segments found by the spotting stage into the recognizer. According to the proposed filtering technique, we dismiss all hypotheses that contain 3 or less characters. A total of 19 handwriting segments remained. On these segments, we reach a WER of 17 %.

To validate that the recognizer itself works well, we manually removed the two false positive segments. On the remaining 17 segments, a word error rate of 7 % is reached. Thus, the recognizer works in the expected range on the segments output by the spotting stage, although the recognition task is slightly harder. We noticed that the motion to bring the hand in writing position and back cannot be discriminated from the actual handwriting and is likely part of the signal. Additionally, the data were recorded in daily life situations and not under as restricted conditions as for the handwriting recognition experiments.

The results of this preliminary evaluation demonstrate the functionality of the end-to-end system but do not allow for an in-depth analysis. We plan to record a large data corpus to evaluate the performance of the end-to-end system and assess the impact of the spotting stage on the overall recognition result. In addition to the proposed two-stage approach, we plan to use a garbage model, which is commonly used in speech recognition to tackle the segmentation problem.

6 Conclusion

We show that spotting and continuous recognition of text written in the air based on inertial sensors is possible. The proposed system can serve as an input device for wearable computers, allowing the input of text in an intuitive way and without the need to operate any handheld devices. The proposed spotting algorithm works with high recall and low precision but we show that additional filtering based on the results acquired from the recognition stage can filter out up to 99 % of the false positives.

We performed experiments on continuous recognition of sentences written in the air with underlying vocabularies up to more than 8,000 words. To our knowledge, a gesture vocabulary of more than 8,000 continuously recognizable gestures is significantly larger than those in any previously reported findings. For the person-independent case, an average word error rate of 11 % was achieved, and for the person-dependent case, this improves to even 3 %. We deem these error rates as low enough to allow practical usage. Although we have only a small data corpus to test the end-to-end system, the achieved results show that the recognition works as expected and only few false positive segments pass the filtering step.

We consequently apply methods developed in speech recognition to the domain of gesture recognition and show their applicability. The results can be transferred to other domains of gesture recognition tasks where specific gestures are built from a smaller set of primitives. None of the used techniques is tailored to the problem of handwriting recognition. The proposed architecture and methods allow the implementation of a system operating in realtime on continuous data [4].

References

1. Amft O, Amstutz R, Smailagic A, Siewiorek D, Tröster G (2009) Gesture-controlled user input to complete questionnaires on wrist-worn watches. In: Human–computer interaction. Novel interaction methods and techniques. Lecture Notes in Computer Science, vol 5611. Springer, Heidelberg, pp 131–140
2. Amma C, Gehrig D, Schultz T (2010) Airwriting recognition using wearable motion sensors. In: Proceedings of the 1st augmented human international conference (AH'10). doi:10.1145/1785455.1785465

3. Amma C, Georgi M, Schultz T (2012) Airwriting: hands-free mobile text input by spotting and continuous recognition of 3D-space handwriting with inertial sensors. In: IEEE 16th international symposium on wearable computers (ISWC), pp 52–59
4. Amma C, Schultz T (2012) Airwriting: demonstrating mobile text input by 3D-space handwriting. In: Proceedings of the ACM international conference on intelligent user interfaces (IUI'12)
5. Bang WC, Chang W, Kang KH, Choi ES, Potanin A, Kim DY (2003) Self-contained spatial input device for wearable computers. In: Proceedings of the IEEE international symposium on wearable computers (ISWC'03)
6. Chen M, AlRegib G, Juang B (2012) 6D motion gesture recognition using spatio-temporal features. In: IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, pp 2341–2344
7. Ephraim Y, Merhav N (2002) Hidden markov processes. IEEE Trans Inf Theory 48(6):1518–1569. doi:10.1109/TIT.2002.1003838
8. Gustafson S, Bierwirth D, Baudisch P (2010) Imaginary interfaces: spatial interaction with empty hands and without visual feedback. In: Proceedings of the 23nd annual ACM symposium on user interface software and technology (UIST'10)
9. Hein A, Hoffmeyer A, Kirste T (2009) Utilizing an accelerometric bracelet for ubiquitous gesture-based interaction. In: Universal access in human–computer interaction. Intelligent and ubiquitous interaction environments. Lecture Notes in Computer Science, vol 5615. Springer, Heidelberg, pp 519–527
10. Huang X, Acero A, Hon H (2001) Spoken language processing. Prentice Hall, NJ
11. Junker H, Amft O, Lukowicz P, Tröster G (2008) Gesture spotting with body-worn inertial sensors to detect user activities. Pattern Recognit 41(6):2010–2024
12. Kallio S, Kela J, Mantyjarvi J (2003) Online gesture recognition system for mobile interaction. In: Proceedings of the IEEE international conference on systems, man and cybernetics (ICSMC'03)
13. Kim D, Choi H, Kim J (2006) 3D space handwriting recognition with ligature model. In: Ubiquitous computing systems. Lecture Notes in Computer Science, vol 4239. Springer, Heidelberg, pp 41–56
14. Lee HK, Kim J (1999) An hmm-based threshold model approach for gesture recognition. IEEE Trans Pattern Anal Mach Intell 21(10):961–973. doi:10.1109/34.799904
15. Lyons K, Starner T, Plaisted D, Fusia J, Lyons A, Drew A, Looney EW (2004) Twiddler typing: one-handed chording text entry for mobile phones. In: Proceedings of the SIGCHI conference on human factors in computing systems (CHI'04)
16. MacKenzie IS, Soukoreff RW, Helga J (2011) 1 thumb, 4 buttons, 20 words per minute: design and evaluation of h4-writer. In: Proceedings of the 24th annual ACM symposium on user interface software and technology (UIST'11)
17. McGuire R, Hernandez-Rebollar J, Starner T, Henderson V, Brashear H, Ross D (2004) Towards a one-way american sign language translator. In: Proceedings of the sixth IEEE international conference on automatic face and gesture recognition (FGR'04)
18. Mistry P, Maes P, Chang L (2009) Wuw-wear ur world: a wearable gestural interface. In: Proceedings of the 27th international conference extended abstracts on human factors in computing systems (CHI EA '09)
19. Mitra S, Acharya T (2007) Gesture recognition: a survey. IEEE Trans Syst Man Cybern Part C Appl Rev 37:311–324
20. Odell J, Valtchev V, Woodland P, Young S (1994) A one pass decoder design for large vocabulary recognition. In: Proceedings of the workshop on human language technology. Association for Computational Linguistics, pp 405–410
21. Plamondon R, Srihari S (2000) Online and off-line handwriting recognition: a comprehensive survey. IEEE Trans Pattern Anal Mach Intell 22(1):63–84
22. Rabiner L. (1989) A tutorial on hidden markov models and selected applications in speech recognition. Proc IEEE 77(2):257–286
23. Raffa G, Lee J, Nachman L, Song J (2010) Don't slow me down: bringing energy efficiency to continuous gesture recognition. In: Proceedings of the international symposium on wearable computers (ISWC'10)
24. Schultz T (2002) GlobalPhone: a multilingual speech and text database developed at Karlsruhe University. In: Proceedings of the international conference on spoken language processing (ICSLP'02)
25. Soltau H, Metze F, Fügen C, Waibel A (2001) A one-pass decoder based on polymorphic linguistic context assignment. In: IEEE workshop on automatic speech recognition and understanding (ASRU '01)
26. Stiefmeier T, Roggen D, Tröster G, Ogris G, Lukowicz P (2008) Wearable activity tracking in car manufacturing. IEEE Pervasive Comput 7(2):42
27. Stolcke A (2002) Srilm—an extensible language modeling toolkit. In: International conference on spoken language processing
28. Tamaki E, Miyaki T, Rekimoto J (2009) Brainy hand: an ear-worn hand gesture interaction device. In: Proceedings of the 27th international conference extended abstracts on human factors in computing systems (CHI EA '09)
29. Woodman OJ (2007) An introduction to inertial navigation. Technical report. University of Cambridge