Air-Writing Recognition—Part I: Modeling and Recognition of Characters, Words, and Connecting Motions

Mingyu Chen, Ghassan AlRegib, *Senior Member, IEEE*, and Biing-Hwang Juang, *Fellow, IEEE*

***Abstract*—Air-writing refers to writing of linguistic characters or words in a free space by hand or finger movements. Air-writing differs from conventional handwriting; the latter contains the pen-up-pen-down motion, while the former lacks such a delimited sequence of writing events. We address air-writing recognition problems in a pair of companion papers. In Part I, recognition of characters or words is accomplished based on six-degree-of-freedom hand motion data. We address air-writing on two levels: motion characters and motion words. Isolated air-writing characters can be recognized similar to motion gestures although with increased sophistication and variability. For motion word recognition in which letters are connected and superimposed in the same virtual box in space, we build statistical models for words by concatenating clustered ligature models and individual letter models. A hidden Markov model is used for air-writing modeling and recognition. We show that motion data along dimensions beyond a 2-D trajectory can be beneficially discriminative for air-writing recognition. We investigate the relative effectiveness of various feature dimensions of optical and inertial tracking signals and report the attainable recognition performance correspondingly. The proposed system achieves a word error rate of 0.8% for word-based recognition and 1.9% for letter-based recognition. We also subjectively and objectively evaluate the effectiveness of air-writing and compare it with text input using a virtual keyboard. The words per minute of air-writing and virtual keyboard are 5.43 and 8.42, respectively.**

***Index Terms*—Air-writing, handwriting recognition, usability study, 6-DOF motion.**

I. Introduction

MOTION gestures provide a complimentary modality for general human–computer interaction. Motion gestures are meant to be simple so that a user can easily memorize and perform them. However, motion gestures themselves are not expressive enough to input text for motion-based control. We define "air-writing" as writing letters or words with hand or finger movements in a free space. Air-writing is especially useful for user interfaces that do not allow the user to type on a keyboard or write on a trackpad/touchscreen, or for text input for smart system control, among many applications.

Isolated letters written in the air involve a sequence of hand or finger movements. Although any snapshot of such movements can be considered a realization of a motion gesture, air-writing is more complicated than gesture recognition because of the interdependence among the involved "gestures." In conventional handwriting, a sequential discrete stroke structure is made. A stroke is an isolated writing trajectory between the pen-up/pen-down events. In contrast, air-writing is rendered on a virtual plane without visual or haptic feedback and lacks the delimited sequence of writing events. Air-writing is also more complex for automatic recognition than cursive style writing on paper due to the lack of a concrete anchoring or reference position; the person who performs air-writing can only use an imaginary coordinate to guide the writing motion. The variability of motion data that represents a letter is thus considerably broader in air-writing than in paper writing.

From a user's perspective, air-writing can be realized in several ways. The first and the most essential is writing of individual isolated letters in an imaginary box in the space, one at a time. The second is the writing of multiple letters across the space from left to right in a style much like writing on a paper. Finally, one can also write several letters, stacked contiguously one over another in the same imaginary box. We call these isolated, connected, and overlapped air-writing, respectively.

The problem of air-writing recognition can be approached progressively. Isolated air-writing carries the assumption that the hand motion to render a letter has already been roughly localized in time and in space. Localization of motion rendering may be accomplished by use of a tracker, which can be easily turned ON or OFF, to signify the beginning and ending of a writing activity. The localization is only approximate and not fluctuation-free because most users cannot precisely synchronize the tracker control (ON–OFF) and the true writing trajectory. This is similar to the notorious problem of end-pointing in spoken utterance recognition even with a push-to-talk control.

Between the approximate endpoints, the motion trajectory forms a letter that resembles a unistroke writing. Study of isolated air-writing is essential to provide the technological foundation for subsequent challenges. Beyond isolated letters, recognition of "word" poses two additional challenges: the contiguous writing of letters without segmentation, and the incorporation of sequential constraints between letters. The distinction between connected and overlapped air-writing mainly arises from system usability; the latter requires less limb movement. From the viewpoint of technology development, techniques for overlapped air-writing can be applied to connected-letter air-writing, and we shall address overlapped air-writing with emphasis.

Manuscript received November 25, 2014; revised June 30, 2015 and August 31, 2015; accepted September 27, 2015. This paper was recommended by Associate Editor R. Plamondon.

The authors are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: mingyu623@gmail.com; alregib@gatech.edu; juang@ece.gatech.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/THMS.2015.2492598

The most challenging and complicated problem is when air-writing is rendered in a tracker-less motion control system, where the hand or the finger is constantly tracked. The motion trajectory contains everything, regardless of the intent of the movement of the user. In such a scenario, detection of meaningful activities related to air-writing is another fundamental challenge. Detection of air-writing activities in a continuous stream of motion data without delimitation of the intent of movements is a fundamental problem different from the recognition problem. We address these challenges in a pair of accompanying papers.

Part I deals with the recognition of isolated characters and words rendered in a single connected stroke fashion. Writing in the air is expected to have increased uncertainty compared to writing on a hard surface. It further presents the results of overlapped air-writing recognition with a usability study. The writing activity to be recognized in this scenario is somewhat localized in time and space, requiring no particular treatment in terms of detection. Part II [1] deals with the ultimate problem of detecting and recognizing writing rendered by finger motion in the air, without any demarcation of the starting or ending points of the strokes, letters, and the letter sequences. Detection of information-carrying writing activities, which are embedded in the overall general motion data with superfluous finger movements unrelated to letters presents a unique and fundamental challenge that needs to be treated separately from the problem of recognition.

Data of air-writing are captured with a motion tracking system. We study isolated and overlapped letter recognition with the assumption that the writing activity is rendered with hand movement, which spans a larger space and is tracked at a farther distance. For embedded air-writing detection and recognition, a smaller writing space, e.g., in front of a desktop monitor, is targeted and we use the LEAP [2] motion tracking device. We learned properties of the motion data from different devices, particularly in the context of data variability for air-writing recognition.

Part I focuses on the air-writing recognition of letters and words. We study air-writing at two levels: motion characters and motion words. Motion characters are recognized in a manner similar to what we have proposed for motion gesture recognition [3], but the data normalization process is adjusted to address the offset issue when motion characters are concatenated together. We also propose ligature modeling for motion word recognition on top of motion letter/character recognition, where a ligature is a connecting motion from a character to the next.

There are two main contributions of this paper. First, we adapt the six-degree-of-freedom (DOF) motion gesture recognition [3] work for character recognition and incorporate context information to achieve air-writing recognition beyond the letter or character level. This extension includes the introduction of a modeling hierarchy, consisting of letter-based motion trajectory models and ligature models, to cope with the unique contiguous writing style of air-writing. Second, we conduct a usability study to demonstrate that air-writing is a preferred text input method on a motion-based user interface. We evaluate the text input performance of the proposed air-writing system and the use of a virtual keyboard with both subjective and objective metrics.

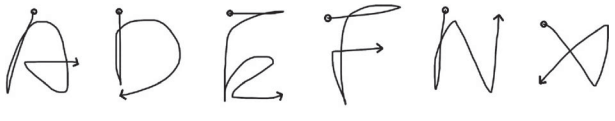
This paper is organized as follows. In the next section, we discuss the related prior work. Section III describes the motion tracking system and data acquisition procedures for air-writing. In Section IV, we explain the feature extraction and normalization procedure and present the techniques for modeling motion characters and motion words. The experiment setup and results are given in Section V. We present the usability study in Section VI. The discussion and conclusion are in Section VII.

II. Related Work

Traditional handwriting styles include cursive or print letters. These writing styles vary with writers and are often mixed in actual handwriting. To make it easier for a machine to recognize and quicker for a user to write, letters are simplified into single-stroke styles. The Graffiti alphabet [4] best exemplifies the unistroke handwriting. Because the unistroke alphabet differs from conventional writing, a novice user needs to learn and practice the writing system to attain entry speed.

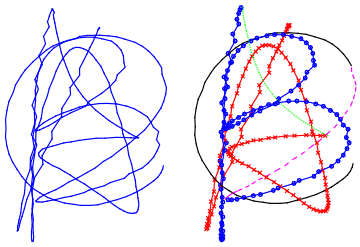
There are other text input modalities in addition to typing and writing. One alternative approach is a mixture of typing and writing. In Quickwriting [5], [6], a user swipes strokes on zones and subzones to input the associated characters. TwoStick [7] applies a similar concept to the two joysticks on a gamepad. Swype [8] allows a user to enter words on a soft keyboard by sliding from the first letter of a word to its last letter and uses a language model to guess the intended word. Similar to typing on a virtual keyboard, swiping strokes also requires the user's attention to the visual feedback while inputing text and is not eyes-free.

A review of automated handwriting recognition can be found in [9]. Hidden Markov models (HMMs) are widely used for online handwriting recognition [10], [11]. In [12], ligature models are proposed to address online recognition of cursive handwriting, in which successive letters are connected without explicit pen-up moves. Motion-based handwriting can also be considered in parallel to motion gestures or sign language. Motion gesture recognition has been studied with different types of motion tracking devices [3], [13]. Sign language is more sophisticated than motion gestures. Many sign language recognition systems use HMMs with various sensing technologies, such as data gloves and vision-based techniques [14], [15]. In [16] and [17], air-writing recognition was achieved with inertial sensors attached to a glove. Jin *et al.* [18] proposed a vision-based approach for finger-writing character recognition. Schick *et al.* [19] also proposed a vision-based hand tracking system that recognizes handwriting in mid-air. In [20], finger writing in the air is tracked with a depth sensor. Different motion sensing and tracking technologies impose various behavioral load on the user. As an example, wearing data gloves is often considered by many users as an undesirable burden and may change the wearing user's motion behavior. In our earlier work [21], we opt for a hybrid tracking system (and the accompanying device) that is simple to control and convenient to use. Due to different physical meanings and dimensions of the motion data, it is less



(a) A (b) D (c) E (d) F (e) N (f) X

Fig. 1. Illustrations of the unistroke writing of isolated letters. (a) A. (b) D. (c) E. (d) F. (e) N. (f) X.



(a) ABC (b) segmented ABC

Fig. 2. Two-dimensional projected trajectory of a motion word. (a) ABC. (b) Segmented ABC.

meaningful to apply the methods mentioned above to our dataset for direct comparison.

III. Air-Writing With Six-Degree-of-Freedom Motion Tracking

A. Unique Writing Style

Air-writing is fundamentally different from conventional handwriting on paper or a surface, which provides no haptic feedback. Similar to motion gestures, air-writing is tracked with a continuous stream of sensor data, and the writing is intuitively rendered in the air in unistroke without any pen-up and pen-down information. The user envisions a writing box in the space and writes in this imaginary space without haptic feedback. Air-writing also does not require visual feedback. Air-writing consists of two levels: motion characters and motion words. Motion characters are isolated alphanumeric letters written in one continuous stroke. In this study, we do not consider modified unistroke styles of alphabet, such as Graffiti [4]. Instead, we simply connect the pen-up and pen-down strokes to form unistroke letters altogether. In Fig. 1, we illustrate the unistroke writing trajectory of several uppercase letters.

One level up, a motion word is formed by connecting motion characters with ligature motions in-between. When there is no haptic or visual feedback, the ordinary left-to-right writing style is difficult to maintain without overlap or shape distortion. In our preliminary experiment, we discovered that users tend to shrink and overlap the last few letters of a word when the envisioned "writing space" is impacted by limited arm range. Therefore, we ask the user to write every character of a word in a layer-by-layer manner, overlapping all letters of the word in the same envisioned virtual box, a writing style we term "overlapped air-writing," which supersedes the usual connected writing style and appears to be more suitable for air-writing.

Fig. 2(a) shows the projected trajectory of an air-writing of the letters ABC. It is difficult for a human to recognize the overlapped handwriting. To better illustrate the contrast and challenge, we manually segment the motion word and show the results in Fig. 2(b). The dash lines are ligature motions that connect characters A, B, and C. When the user writes freely in the air without visual feedback of the writing trajectory, it is difficult to air-write neatly, and recognition is therefore challenging.

Figs. 1 and 2 demonstrate several key differences between air-writing (beyond single characters) and conventional hand-writing. The first major difference is the lack of pen-up/pen-down moves and haptic feedback. The pen-up/pen-down information is useful for word segmentation of cursive handwriting and stroke delimitation for hand-print writing. The second is the box-writing style with overlapping characters. Moreover, we track air-writing with 6-DOF motion data (translation and rotation), which is also different from the conventional 2-D spatial trajectory of pen-based writing. In our case, features derived for traditional handwriting recognition cannot be applied. Among related works of automatic air-writing recognition, Amma *et al.* [17] had a similar box-writing style that is tracked with only inertial sensors. To our best knowledge, we are the first to evaluate air-writing recognition with 6-DOF motions.

B. Six-Degree-of-freedom Motion Tracking and Data Acquisition

We use a hybrid framework for 6-DOF motion tracking: the Worldviz PPT-X4 [22] for optical tracking of the position of the infrared tracker and the Wii Remote Plus (Wiimote) for the inertial measurements of the acceleration and angular speed. The orientation is derived from a fusion of the acceleration and angular speed data. The system tracks a specially designed handheld device and provides both explicit (position and orientation) and implicit (acceleration and angular speed) 6-DOF data sampled at 60 Hz. For details of our 6-DOF motion tracking system, see [3].

The handheld device is equipped with marker activation buttons and the user utilizes the buttons on the device to engage in a push-to-write mode of writing. The user holds a button to start writing and releases it when a character or a word is finished. With push-to-write, while the starting point of the very first character does exist, albeit with some ambiguity due to imprecise synchronization on the user's part, the next anchoring point of the motion trajectory happens at the end of the word, after all characters having been written and again with similar ambiguity as in the beginning. A motion character or word itself is thus rendered in a single continuous stroke.

A controller-free system, such as the KINECT or the LEAP [2], has no buttons and requires different forms of delimiters for push-to-write. It is possible to use a specific posture or gesture to signal the endpoints of a writing act, e.g., a pinch gesture or a waving palm. When depth information is available, the user can segment writing by reaching out beyond a certain depth threshold [19]. With Kinect, a more sophisticated input zone was proposed for gesture delimitation in [23]. In our case, the push-to-write scheme with a physical button solves the delimitation problem. Automatic detection (spotting) of air-writing is more preferable for controller-free systems and will be covered in Part II [1].

In air-writing, both allographs and different stroke orders can result in different spatiotemporal patterns for the same letter, which require separate models for classification. We have to ensure collection of sufficient data for modeling air-writing while keeping the recording process manageable in time. Therefore, we address the air-writing recognition problem that consists of only uppercase letter A to Z with the predefined stroke order for each character (see Fig. 1) to avoid allographs or different stroke orders. Our recording procedures balances "generality of air-writing recognition" and "feasibility of air-writing recording."

We recruited 22 participants (all right-handed, 17 males and five females) to record air-writing data. Each subject was advised to write in a consistent manner, but we did not constrain his or her gripping posture of the controller, the writing scale, or speed. The only rules were the box-writing style, the stroke order for each letter, and the push-to-write scheme. We allowed each subject to practice and started recording when he or she felt comfortable with air-writing.

Each isolated motion character (A to Z) was recorded ten times by every subject. For motion words, we select 40 words from common television channels, e.g., ABC, CNN, FOX, and common digital/Internet services, such as TV, MUSIC, and GOOGLE. The shortest word has two characters, and the longest one is DISCOVERY. Our vocabulary covers all 26 characters, and the average number of characters per word is four. Every subject recorded each word five times. The implementation and recording details are identical to the 6-DOF motion gesture database [21]. The whole recording process (26 characters and 10 words) takes about 1 h for each user. In addition to these 40 words, we create another 1k-word vocabulary, which includes the most frequent 1000 two/three-letter words and three-letter prefixes from the Google Web 1T 5-gram dataset [24]. The 1000 words were recorded without repetition only by subject M1. There is no intersection of two vocabularies. Our air-writing dataset has 5720 motion characters and 5400 motion words in total and is available for download.[1]

During the recording, the subjects often started to feel arm fatigue after 10–15 min of writing. In general, longer words are harder to write in one stroke without making mistakes. The consumed time and efforts makes air-writing more appealing to input shorter text or commands.

TABLE I
DURATIONS (IN NUMBER OF SAMPLES) OF MOTION CHARACTERS BY 22 SUBJECTS

	avg	std		avg	std		avg	std
A	159.5	37.4	J	60.6	12.0	S	92.7	17.8
B	156.7	37.1	K	136.8	26.4	T	88.6	16.4
C	77.3	19.8	L	64.8	15.0	U	73.5	14.1
D	118.7	24.9	M	146.4	29.4	V	67.2	11.5
E	190.8	48.6	N	115.7	21.1	W	110.4	19.3
F	132.6	27.4	O	85.1	17.4	X	91.3	16.1
G	149.7	35.1	P	107.6	20.3	Y	105.7	20.5
H	137.5	29.9	Q	119.4	26.4	Z	94.1	18.6
I	42.8	10.3	R	134.9	24.5			

The sample rate is 60 Hz.

IV. AIR-WRITING PROCESSING AND MODELING

A. Feature Processing

From the 6-DOF motion data, we derive five features (observations): position P and velocity V from optical tracking, orientation O, acceleration A, and angular speed W from inertial tracking. Let $P^o = [p_x, p_y, p_z]^\top$ denote the positions, and $V^o = [\Delta p_x, \Delta p_y, \Delta p_z]^\top$ the rate of change in position. The orientation is represented in quaternion, $O^o = [q_w, q_x, q_y, q_z]^\top$.

The implicit 6-DOF motion data form two features. Let $A^o = [a_x, a_y, a_z]^\top$ denote the device-wise accelerations and $W^o = [w_y, w_p, w_r]^\top$ denote the angular speeds in yaw, pitch, and roll, respectively. The superscript o indicates the original data from the tracking system before processing. The notations above represent the time sequences of a recorded air-writing in the corresponding coordinates, e.g., $P^o = [p_x(i), p_y(i), p_z(i)]^\top, i = 1, 2, ...N$, where N denotes the number of samples captured in a writing pattern.

The writing style and speed vary among users. In Table I, the statistics of the writing duration (in number of samples) of motion characters from 22 subjects show a difference in the writing speed. As explained in [3], proper feature normalization is the key to make the 6-DOF motion recognizer scale and speed invariant. The original normalization process in [3] requires to offset P^o and O^o by the starting position and orientation of a motion gesture. Due to the increased writing complexity and the writing modality for motion characters and words, the normalization procedure for the motion data needs substantial modification from what was used for gesture recognition [3]. The resulting character models need to be centered in the same "virtual box" to be concatenated to form word models. Thus, we modify the normalization process for P^o and O^o to address the offset issue as follows.

First, P^o and O^o are offset to account for variations in the position and orientation of the "virtual box"

$$\begin{aligned} \tilde{P}(i) &= P^o(i) - \vec{p}_{cen} \\ \tilde{O}(i) &= O^o(i) * \vec{q}_{cen}^{-1} \end{aligned} \tag{1}$$

where $\vec{p}_{cen}$ is the center point of the bounding volume of P^o, $\vec{q}_{cen}$ is the normalized quaternion of $[\bar{q}_w, \bar{q}_x, \bar{q}_y, \bar{q}_z]$, i.e., the arithmetic mean of O^o, and $*$ denotes quaternion multiplication. The arithmetic mean of quaternions serves as a reasonable approximation for the "center" of the range of orientation. Normalization of $\tilde{P}$ is straightforward with uniform linear scaling, i.e., $P = s_p \tilde{P}$, where s_p scales the longest edge of the bounding volume of $\tilde{P}$ to unit length. Here, we use uppercase letters without superscript to denote the normalized features.

To normalize $\tilde{O}$, we first convert the quaternion into an axis-angle representation expressed as

$$\left[\cos\frac{\alpha_i}{2}, \vec{r}_i \sin\frac{\alpha_i}{2}\right] = \tilde{O}(i), \quad i = 1, 2, \ldots, N \tag{2}$$

[1]The air-writing demo video, data, vocabulary, viewer, loader, and exporter are available at http://www.ece.gatech.edu/6DMG

where α_i is the angle rotated about the axis $\vec{r}_i$ by the right-hand rule. We normalize the rotation angle and keep the rotation axis intact. In our tracking system, the orientation is updated with an incremental rotation at every sampling instant to ensure its continuity in tracking the evolving orientation. (2) defines the true rotation direction and the angle in the range from 0 to 2π. We scale the rotation angle and compute the normalized orientation as

$$O(i) = \left[\cos\frac{s_\alpha \alpha_i}{2}, \vec{r}_i \sin\frac{s_\alpha \alpha_i}{2}\right] \tag{3}$$

$$s_\alpha = \frac{\alpha_{\max}}{\max(\alpha_i)}, \quad i = 1, 2, \ldots, N. \tag{4}$$

In (4), $\alpha_{\max}$ indicates the maximum rotation angle after normalization and is set to 0.2π empirically.

Features V, W, and A do not suffer from the offset issue; we use the same normalization process as in motion gesture recognition reported in [3].

B. Air-Writing Modeling

With the processed features in the previous section, a writing motion can be represented as a spatiotemporal pattern. Each underlying state in an HMM describes a subset of this pattern with a particular kinematic meaning. A left-to-right HMM is suitable for order-constrained time-evolving signals. Skip transitions are not considered because a user is unlikely to skip a segment of the continuous motion in air-writing based on our empirical observations.

HMM models for motion characters can be readily concatenated to form a motion word with additional connecting ligature motions. Such a modeling methodology is common, known as subword modeling, in large vocabulary continuous speech recognition to circumvent insufficient training data issues [25]. Gesture recognition typically involves a limited vocabulary set. It is relatively easy to collect sufficient data of each gesture and straightforward to model each gesture directly from its own recordings. However, the vocabulary of air-writing can easily be thousands of words, and it is difficult to collect enough data for every word in the vocabulary. The data sufficiency problem prevents a designer from directly using whole "word" models ("whole word" here is referring to the entire motion trajectory that covers each and every character in the word from the beginning to the end).

HMMs of motion characters are trained directly from the isolated A-to-Z recording, and we create one model for each character. As shown in Table I, the duration of letter varies substantially, and we empirically assign the number of states for each character roughly proportional to its duration. For example, the most complicated letter `E` has 18 states, and short letters `I` and `J` have only eight states. These motion character models are the building blocks for motion words.

The word-level HMM model is built upon these individual character models. A motion word is formed by connecting motion characters with ligature motions. Here, we define the ligature as the motion from the ending point of the preceding character to the starting point of the following character. In Fig. 2, two ligatures connecting `AB` and `BC` are illustrated in dash line. These endpoints (both the starting and the ending points) of ligatures contain substantial variability and are not precisely defined; they are dealt with only in a statistical sense. Ligatures are context dependent. With 26 uppercase letters, there are 676 (26×26) combinations for connecting two letters, which lead to 676 ligature models in theory. It is nonetheless difficult to obtain enough data to cover all combinations to train the ligature models. We alleviate this problem by clustering similar ligature motions into groups and effectively reducing the total number of ligature models. Based on P of the first and the last states of the character HMMs, we heuristically cluster the start and the end points of uppercase letters into three and seven groups, respectively, in Table II. The clustering reduces the number of ligature models to 21 (7×3). We label every ligature given the preceding and the following characters and form the model for a motion word. Take the ligature between `A` and `B` for example. The preceding `A` has an ending point `E5`, and the following character `B` has a starting point `S1`. The clustered ligature becomes `lig_E5S1`. The HMM for `ABC` can then be constructed as `A·lig_E5S1·B·lig_E1S3·C`, where "·" denotes concatenation. Note that the occurrence of each ligature in our dataset is not equally distributed.

TABLE II
CLUSTERS FOR START AND END POINTS OF CHARACTERS

	Start point		End point
S1	BDEFHKLMNPRTUVWXYZ	E1	BDSX
S2	AIJOQ	E2	ITY
S3	CGS	E3	CEGHKLMQRZ
		E4	JP
		E5	AF
		E6	O
		E7	NUVW

Although the heuristic clustering in Table II with context dependence led to reasonable results, we further considered clustering ligatures with a data-driven decision tree based on likelihood and found that to be more effective. By asking questions about the previous or next connecting letter of each ligature, the decision tree attempts to find those contexts which make the largest difference in the calculated likelihood to distinguish clusters. Each question splits the current pool of training data into two sets and increases the likelihood with the use of two sets rather than one. We branch the decision tree by selecting the question that maximizes the increase of likelihood and repeat the process until the likelihood increase achieved by any question at any node is less than a prescribed threshold. In Fig. 3, an illustrative decision tree is shown, which branches the ligatures into eight clusters (leaf nodes) by asking seven questions.

In Table II, we manually clusters characters according to the position of the starting and the ending points. Based on the hard clustering, we generate several general questions, such as: *"Does the previous letter belong to* `E1`*?"* and *"Does the next letter belong to* `S2`*?"* To take into account both the endpoint position and the stroke direction, we further divide the hard clustering to create more detailed questions, e.g., *"is the previous letter* `B` *or* `D`*(ends at bottom left with a right-to-left*

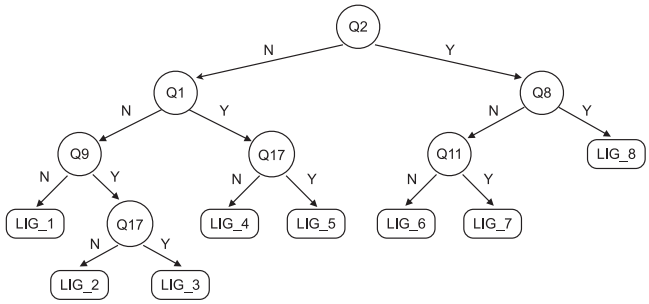


Fig. 3. Illustrative decision tree that results in eight clustered ligatures.

TABLE III
QUESTION SET FOR THE DATA-DRIVEN DECISION TREE

Is the previous letter?		Is the next letter?	
Q1	BDSX	Q17	BDEFHKLMNPRTUVWXYZ
Q2	CEGHKLMQRZ	Q18	AIJOQ
Q3	NUVW	Q19	A
Q4	AF	Q20	BDHKLMNPRU
Q5	BD	Q21	CGS
Q6	C	Q22	EFTZ
Q7	ELZ	Q23	IJ
Q8	GHM	Q24	OQ
Q9	ITY	Q25	VWXY
Q10	JS		
Q11	KQR		
Q12	NU		
Q13	O		
Q14	P		
Q15	VW		
Q16	X		

stroke)" and *"is the next letter* I *or* J*(starts at top center with a top-to-bottom stroke)?"* We list the complete question set for the data-driven decision tree in Table III.

Because the decision tree is data-driven, the questions and the resulting clusters vary upon the training data. During the branch process, not all questions in Table III are asked, and some questions may be asked multiple times. The real decision trees in our experiment have 30–40 branch nodes (asked questions) and 30–40 leaf nodes (clustered ligatures). With the help of the decision tree, we are able to synthesize unseen ligatures in the training data and generate models for all possible ligatures. The HMM for ABC now becomes A·lig_AB·B·lig_BC·C.

In a motion word, we do not have or need the character-level segmentation. With the composite word HMM, character and ligature segmentation (alignment) can be simultaneously accomplished during recognition. Because the motion trajectories of characters and ligatures usually blend together, the ground truth of segmentation may be ambiguous. To have a better understanding of the ligature motions, we manually segmented all the motion words in the 40-word vocabulary recorded by subject M1. The manual segmentation is used for initial estimates of ligature models, which is proven to work better than ones that are initialized with zero means and global variances. The procedures to reestimate ligature models will be presented in Section V-B.

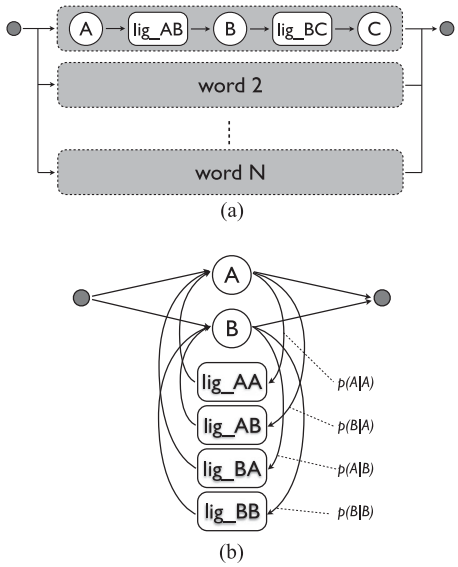


Fig. 4. Decoding word networks. (a) Word-based. (b) Letter-based (simplified).

V. MOTION CHARACTER AND MOTION WORD RECOGNITION EVALUATION

We first evaluate motion character recognition with the five basic features (P,V,O,A,W) and different combinations of them, including PV, AWO, and $PVAWO$. The combinations of features actually correspond to different motion tracking devices. PV is the feature set derived purely from optical tracking, and AWO can be considered the full feature set from inertial measurements. $PVAWO$ uses the available data from a hybrid 6-DOF motion tracking system.

A motion gesture can be defined in a 3-D space, but handwriting is actually defined on a 2-D surface regardless of the true writing motions. Therefore, we also investigate the feature $\hat{P}$ and $\hat{V}$, where $\hat{P}$ and $\hat{V}$ are the x and y components of P and V. We can consider $\hat{P}$ and $\hat{V}$ as representing the writing motion projected on a vertical plane.

Based on the experience in [2], we empirically determine the number of states for each character based on its average duration (see Table I) and choose single Gaussian mixture per state for all motion characters. Ligatures are simple and short in duration; therefore, they are modeled with only three states and a single Gaussian mixture per state. We use the Hidden Markov Model Toolkit (HTK)[2] for HMM modeling, training, and testing. Both motion character recognition and motion word recognition are studied in the user-independent case with leave-one-out cross validation. We choose one subject as the testing set and train the models with the remaining 21 subjects. This procedure is repeated for every subject to get the average recognition rate.

We perform motion word recognition with two approaches: word-based and letter-based recognition. In word-based recognition, we synthesize the HMM for every word in the vocabulary as shown in Fig. 4(a), and the recognition is done at the unit of one word. Given a vocabulary of N words, word-based

[2]The HTK is available at http://htk.eng.cam.ac.uk

TABLE IV
CER OF MOTION CHARACTER RECOGNITION

features	CER (%)	
	average	std
P	3.72	(3.60)
V	6.12	(2.88)
O	3.81	(5.05)
A	7.97	(7.38)
W	7.92	(3.34)
PV	1.61	(2.16)
AWO	1.84	(2.37)
$PVOAW$	1.05	(1.23)
$\hat{P}$	3.88	(3.55)
$\hat{V}$	6.15	(2.69)
$\hat{P}\hat{V}$	1.61	(2.06)
$\hat{P}\hat{V}OAW$	1.05	(1.33)

recognition is formulated as a one-out-of-N problem and becomes more robust to individual letter errors within a word. However, word-based recognition requires the user to finish the whole word before recognition and cannot handle words that are out of vocabulary (OOV).

Letter-based recognition decodes on a letter basis. Fig. 4(b) illustrates a simplified example of a letter-based decoding network that is built with letters A and B and the corresponding ligature models. The letter-based decoding network allows arbitrary decoded letter sequences and can handle OOV words. Another advantage is that letter-based word recognition allows progressive decoding while the user is writing, unlike the word-based recognition that requires the user to complete a word. The freedom of arbitrary sequences comes at a price of sacrificing the contextual information from the vocabulary. Under such circumstances, ligatures become the only contextual information we can use; therefore, we need more precise ligature models. To further improve the recognition performance, we can utilize contextual information from the vocabulary by introducing a language model. The decoded letter sequence will depend on both the writing motion and the conditional probability of preceding letters. In this study, we use a bigram language model and embed the conditional probabilities in the transition arcs from character nodes to ligature nodes as shown in Fig. 4(b).

A. Motion Character Recognition

The HMMs of motion characters are trained and tested with isolated characters, and we show the character error rate (CER) of leave-one-out cross validation with different features in Table IV. First, we compare the discriminative power of the basic features. The explicit 6-D features (P, V, and O) outperform the implicit 6-D features (A and W). The projected 2-D features ($\hat{P}$ and $\hat{V}$) have slightly higher error rates than the 3-D ones. Combining features of different kinematic meanings makes each HMM state more discriminative and improves the recognition rate. The improvement of combining features becomes less prominent at certain level.

Although a character is only defined by its 2-D spatial trajectory, features of different kinematic meanings prove to be informative to distinguish motion characters written in the air. We can achieve robust motion character recognition even with AWO. The CER of pure inertial tracking is slightly higher than the CER of pure optical tracking, and $PVAWO$ achieves the lowest CER. The performance of motion character recognition supports our previous study on motion gesture recognition [3].

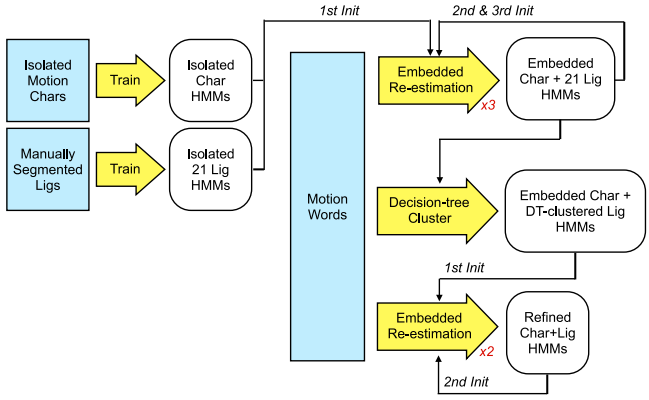


Fig. 5. Flow of the embedded reestimation for character and ligature models.

B. Reestimated Character and Ligature Models

From the experiment mentioned above, we obtain the HMMs of isolated motion characters and use them to initialize the character HMMs for motion word recognition. To construct the word model, we also need HMMs for ligatures. In Section IV-B, we propose two approaches to model ligatures: hard clustering and decision tree. First, we extract the ligatures from the manually segmented motion words in the 40-word vocabulary written by subject `M1`. We cluster these ligatures as in Table II and train the isolated ligature HMMs.

Given the initial HMMs of isolated characters and ligatures, we synthesize the HMM for each word in our vocabulary and perform embedded Baum–Welch reestimation [25], [26] on all recordings in the training set. We take the reestimated results as the initial values for the next iteration of reestimation. Iterative reestimation allows the trained models to converge to the training data. However, repeated reestimation may lead to overtraining if the models become too closely matched to the training data and fail to generalize well on unseen test data. In practice, around two to five iterations of embedded reestimation are usually sufficient for training. We use the improvement of the overall log likelihood per frame of the training data as a termination condition for repeated reestimation and stop after the third iteration of embedded reestimation.

To this point, we have the refined HMMs of characters and 21 hard clustered ligatures, which are used to generate the decision tree. Because the decision tree is data driven, the resulting ligature clusters may vary in different training sets. After clustering with the decision tree, two more iterations of reestimation are done to obtain the final HMMs of characters and decision-tree-clustered ligatures. The reestimated character and ligature HMMs are the building blocks to the decoding word network for motion word recognition. The flow of the embedded reestimation process is summarized in Fig. 5.

TABLE V
RESULTS OF MOTION WORD RECOGNITION ON THE 40-WORD VOCABULARY WITH 22 SUBJECTS

features	word-based		letter-based (backoff)			
	WER (%)		WER (%)		CER (%)	
	average	std	average	std	average	std
PV	0.045	(0.144)	10.59	(6.63)	3.48	(2.67)
$\hat{P}\hat{V}$	0.023	(0.104)	9.20	(5.36)	2.86	(1.82)
AWO	0.0	(0.0)	14.93	(11.11)	5.70	(4.86)
$PVOAW$	0.0	(0.0)	11.57	(8.23)	4.15	(3.48)
$\hat{P}\hat{V}OAW$	0.0	(0.0)	10.61	(7.31)	3.65	(2.82)

TABLE VI
RESULTS OF MOTION WORD RECOGNITION ON THE $1k$-WORD VOCABULARY WITH SUBJECT "M1"

features	word-based	letter-based (backoff)	
	WER (%)	WER (%)	CER (%)
PV	0.80	1.90	0.66
$\hat{P}\hat{V}$	0.80	2.80	0.97
AWO	1.60	7.00	2.59
$PVOAW$	0.90	4.10	1.42
$\hat{P}\hat{V}OAW$	0.90	5.00	1.73

TABLE VII
AVERAGE WER OF DIFFERENT DESIGNS OF LETTER-BASED MOTION WORD RECOGNITION ON THE 40-WORD VOCABULARY WITH 22 SUBJECTS

features	decision-tree	decision-tree + bigram	decision-tree + bigram + 2-best	decision-tree + bigram (w/o backoff)
PV	17.27	10.59	4.73	2.73
AWO	23.09	14.93	8.16	2.75
$PVOAW$	15.16	11.57	5.77	2.18

C. Word-Based Motion Word Recognition

For word-based word recognition, we use the refined HMMs of character and 21 hard clustered ligatures to build the decoding word network as shown in Fig. 4(a). The word-based word recognition is formulated as a one-out-of-N problem, where N is the vocabulary size. In the word-based decoding network, each path is a word model synthesized from corresponding character and ligature HMMs, and the letter sequences are tightly restricted to the vocabulary.

We can view the ligature model as a filler that absorbs the transition motions between characters as used in [16], [17] and [19]. In our preliminary experiment, we tried a single filler model for all ligatures and obtained reasonable recognition performance. This is because the word-based recognition aggregates all motion clues of a word to make a final decision out of the vocabulary. Refining ligature models with clusters improves the precision of character/ligature segmentation within a word but may not affect the overall word-based recognition much. Note that the single filler model fails in letter-based decoding.

The first experiment is the leave-one-(subject)-out cross validation on 22 subjects with the 40-word vocabulary. We only focus on the combined features. The average word error rates (WER) are listed in Table V, which indicates the recognition performance of the user-independent case. Also, there is no CER in the word-based word recognition. The combined feature sets all perform well, e.g., the WER is 0.045% ($= 2/4400$) for PV.

In the second experiment, we test the scalability of the word-based decoding network. The motion characters and ligatures are trained with the 40-word vocabulary recorded by all subjects except `M1`, and we use the 1k-word vocabulary recorded by `M1` for testing. The synthesized word-based decoding network actually contains both vocabularies, i.e., 1040 words, and the results are shown in Table VI. The WER of AWO is the worst, and the WER of PV is slightly lower than $PVAWO$.

If we remove the factor of user variations and only look at the results of subject `M1` as the testing set in the first experiment, the WER is zero for all feature sets. It is not surprising that a larger vocabulary incurs more ambiguity of similar words and make the recognition more challenging. However, the word-based word recognition still achieves fairly low WER, which makes it appealing for applications that require text input of a limited vocabulary.

D. Letter-Based Motion Word Recognition

We choose the refined HMMs of character and decision-tree-clustered ligatures to build the letter-based decoding word network as shown in Fig. 4(b). In our preliminary experiments of 22 subjects and the 40-word vocabulary, ligature models clustered by a decision tree achieves about 2% absolute WER reduction over hard clustered ones. With the decision-tree-clustered ligatures as the only contextual constraint, the average WER of leave-one-out cross validation of 22 subjects has a similar trend as the results of motion character recognition as shown in the second left column of Table VII.

To further improve the recognition performance, we utilize the statistics of letter sequences of the vocabulary. We estimate the bigram language model for the 40-word and 1k-word vocabulary separately. Good-Turing discounting and backoff are applied to handle unseen bigrams [27]. Each ligature model depends on its previous and next characters; therefore, we can easily embed the conditional probabilities of the bigram language model into the transition arcs from characters to ligatures as in Fig. 4(b).

We show the average WER and CER of leave-one-out cross validation with the bigram language model in Table V. The CER includes insertion, substitution, and deletion errors. The pure inertial AWO has the highest error rates. The pure optical PV outperforms the complete 6-D $PVOAW$ with the help of the language model. In addition, the 2-D projected $\hat{P}\hat{V}$ and $\hat{P}\hat{V}OAW$ achieve lower error rates than their 3-D versions. The bigram language model improves the recognition performance, e.g., the absolute WER reduction is 5.61% for PV, 7.43% for AWO, and 3.71% for $PVAWO$ (see Table VII).

Technically, there is no scalability issues for letter-based word recognition. The only difference is the vocabulary to estimate the language model. We show the results for subject `M1` and

the 1k-word vocabulary in Table VI. For a fair comparison, we also list the WER of subject M1 in the first experiment (40-word vocabulary and leave-one-out cross validation) as follows: 2% ($= 4/200$) for PV and $\hat{P}\hat{V}$, 4.5% for AWO, and 3% for $PVOAW$ and $\hat{P}\hat{V}OAW$. In Table VI, the WER is slightly higher for the 1k-word case.

After examining the recognition results, the common character errors appear to be due to ambiguity of letters, e.g., O and C, D and P, W and N. Another common character error results from similarity of sub-letters, e.g., E is wrongly decoded as FZ. If a letter sequence is wrongly decoded, the correct one usually has a likelihood close to the best one. In n-best recognition, a correct recognition means one of the top n hypotheses matches the ground-truth label. In general, the recognition errors can be roughly halved for two-best recognition. With the 40-word vocabulary and 22 subjects, the WER of $PVOWA$ is reduced from 11.57% to 5.77% for two-best recognition and 2.89% for five-best recognition. We show the WER of two-best recognition in the second column from the right of Table VII. At the point of view of system design, it can be helpful to provide a list of n-best recognition results for the user to choose the right one.

It is possible to further reduce the search space of the decoding network by applying a more restrictive language model. Instead of backoff, zero probability is assigned to unseen bigrams, i.e., disable the transition arcs of unseen ligature models. In the first right column of Table VII, the average WER for the 40-word vocabulary is reduced to 2.73% for PV, 2.75% for AWO, and 2.18% for $PVOAW$. On the other hand, the absolute WER reduction is less than 0.5% for the 1k-word vocabulary. The restrictive language model is more effective when the vocabulary spans a smaller set of bigrams (ligatures). We can consider the letter-based word recognition with the restrictive language model somewhere between the word-based word recognition (the strictest) and the letter-based word recognition with the complete bigram language model (the freest). The restrictive letter-based word network still allows progressive decoding but cannot handle OOV bigrams. If an application only accepts valid English words, the restrictive letter-based word recognition can be applicable for flexible text input.

VI. Usability Study

The purpose of the usability study is to evaluate air-writing as a motion-based text input method. With the same motion tracking system, point-and-click on a virtual keyboard displayed on screen is another possible method for text input; therefore, we consider the virtual keyboard as the comparison group.

A. Methods

The user sits in a living-room like environment with a 65-in full HD display and a handheld tracking device modified from the Wii Remote. The remote control functions as a mouse, with translation in the vertical plane mapped to the cursor movement on the display with a scale of one meter to 2000 pixels. Button A works as the mouse left click, and Button B is used for push-to-write for air-writing. We create a logger program that displays the word to input, a status box, and a start button. The subject needs to click start to start logging the time and traverse distance of writing/typing. The logging stops automatically once the correct word is recognized or typed.

For air-writing, we use the word-based word recognition with the aforementioned 1040-word vocabulary. The time and traverse distance while holding Button B (writing) is recorded separately. The status box displays the recognized word or input status, e.g., writing or recognizing. If an error occurs, the subject re-writes until the word is correctly recognized, and we accumulate the writing time of each trial. For the virtual keyboard, we use the built-in on-screen keyboard of Windows 7, which is resized to the lower half of the screen. The window of our logger is on the upper half of the screen, and the status box now shows the input letters.

We recruited 20 participants from Georgia Institute of Technology for user study (all right-handed, 13 males and seven females). Eight of them participated in the motion handwriting recording, and the rest are novice users. We let every subject get used to the system first and start the experiment once he or she feels comfortable with both input methods. Each subject is asked to "copy" 50 words, which consist of the 40-word vocabulary and ten words randomly drawn from the 1k-word vocabulary. These 50 words are shuffled and stay in the same order for both air-writing and virtual keyboard. We also randomize the order of the sessions of two input methods for all subjects.

For objective metrics, we measure the completion time and motion footprint, i.e., total traverse distance, of text input with air-writing and virtual keyboard. After the experiment, we also ask the subject to rate the intuitiveness, arm fatigue level, and his or her preference of these two methods.

TABLE VIII
Usability Result of air-Writing and Virtual Keyboard (Objective Metrics)

	Air-writing			Virtual keyboard		
word length	time (sec)	distance (cm)	attempt # per word	time (sec)	distance (cm)	extra key # per word
2	5.4	161	1.38	2.6	49	0.04
3	7.2	249	1.19	4.3	86	0.09
4	8.3	312	1.07	5.7	120	0.14
5	10.1	396	1.06	7.4	152	0.29
6+	14.0	566	1.04	9.2	174	0.29

B. Results

We show the average writing/typing time and total traverse distance for words of different length in Table VIII. Because air-writing is recognized on a word basis, we report the average number of attempts to correctly input a word. Longer words tend to have higher recognition accuracy and hence need fewer attempts. The average writing time of a two-letter word is 3.9 ($= 5.4/1.38$) s. For virtual keyboard, we report the average number of extra keystrokes, e.g., a typo and a backspace count as two extra keystrokes.

TABLE IX
USABILITY RESULT OF AIR-WRITING AND VIRTUAL KEYBOARD (SUBJECTIVE RATING FROM 1 TO 5)

Question	air-handwriting	virtual keyboard
1. Intuitiveness [5: most intuitive]	4.10	4.75
2. Arm fatigue level [5: no fatigue]	3.05	3.10
3. Vote for inputing a short word (2-3 letters)	16	4
4. Vote for inputing a long word (4+ letters)	11	9
5. Satisfaction of recognition performance [5: most satisfied]	4.25	-

Words-per-minute (WPM) is a common performance metric for text input efficiency. WPM is computed based on correctly input word units, where one word unit is five letters (keystrokes). The WPM of air-writing and virtual keyboard are 5.43 and 8.42, respectively. Compared to conventional text input methods [28], the WPM of pen-based handwriting without recognition is in the range of 15–25, and the WPM range of QWERT typing (hunt-and-peck style) is 20–40. Although handwriting is not the fastest, it is the most primitive method for text input. Motion-based text input methods are roughly three to five times slower than the conventional ones because relatively large and unconstrained control motions are involved. Our study indicates the speed for these alternative text input methods on a motion-based user interface.

The objective metrics show that air-writing is roughly 1.5 times slower and three times longer in motion footprint than the virtual keyboard. However, we get quite interesting results from the subjective evaluation as shown in Table IX. Air-writing is a variation of conventional writing, and virtual keyboard follows the same metaphor of typing on a touchscreen. Both methods are intuitive to users and have neutral scores for the arm fatigue level. Motions in the air involve more muscles than keyboard or touch-based interaction and thus cause more fatigue. Even though the motion footprint of air-writing is three times larger, it does not directly reflect arm fatigue ratings. The arm fatigue level relates to the writing or typing style. For example, air-writing could cause less fatigue for a user who rests the elbow and writes with the upper arm and wrist than a user who holds the whole arm in the air. The layout of the virtual keyboard is fixed for all subjects. To cover all keys, it requires a larger range of movement, e.g., the distance between key `Z` and `Backspace` is about 60 cm (1200 pixels). Six subjects mention that the keyboard layout is too big. Reducing the size of the keyboard layout can reduce the motion footprint. However, smaller keys can be prone to "typing" errors and require more precise pointing motions. The majority of users choose air-writing for short text input (two to three letters), and about half of users prefer air-writing for long text input (4+ letters).

Based on our study, air-writing may not be fast enough for general-purpose text input, but it is suitable for infrequent and short text input on a motion-based user interface, where conventional writing or typing is not available. Although virtual keyboard is faster than air-writing, a virtual keyboard requires a display and precise pointing. Typing on a virtual keyboard requires two foci of attention (FOA), i.e., the user needs to pay attention to the keyboard and then the input result. On the contrary, air-writing is a single-FOA task. The user does not necessarily need the visual feedback of writing and achieves "eyes-free" text input. Air-writing recognition does not require precise pointing and is applicable to a broader range of motion tracking systems.

There are other usability issues of air-writing from user feedback. The box-writing style appears to be easy to learn, but it needs some practice to write with the specified stroke order. In our current system, writing with different stroke orders can cause errors in recognition, especially for shorter words. Five users suggest to write without constraints on the stroke order, and four users would like to write without holding a button.

VII. CONCLUSION

In this study, we attempt to recognize air-writing with a 6-DOF motion tracking system. The writing motion is tracked with the position and orientation in the global frame, and the acceleration and angular speed in the device-wise coordinates. The air-writing recording process is very time consuming. To make the recording process feasible, we place constraints on stroke orders and uppercase letters with limited vocabulary to refine the scope of air-writing data acquisition without losing too much generality. From these motion data, we derive five basic features for observations of HMMs and form the combination of pure optical, pure inertial, and complete 6-DOF features. Although the handwriting is defined purely by the planar shape, we show that motion information beyond the spatial trajectory is informative for air-writing recognition.

Air-writing is unistroke without pen-up/pen-down information. The writing style and motor control are different from ordinary pen-based writing due to lack of haptic and vision feedback. We separate air-writing in two levels: motion characters and motion words. Motion characters are handled similar to motion gestures, and each character is modeled with a HMM. A motion word can be modeled by concatenating character and ligature models. We present two approaches to model ligatures: hard clustering and decision tree. The former is proven to be sufficient for word-based word recognition. The latter provides better capability of ligature modeling, which improves the performance of letter-based word recognition. The word-based word recognition achieves relatively low WER but is not able to recognize OOV words. The word-based recognizer is suitable for applications that have a limited vocabulary and stringent requirement on the accuracy. On the other hand, letter-based word recognition has around 10% WER but can handle arbitrary letter sequences and progressive decoding. To substantially improve the letter-based recognition accuracy, the system can provide suggestions with n-best decoding and lets the user choose the right one.

A user study investigates input speed, motion footprint, physical strain, and subjective evaluation of two motion-based text input methods: air-writing and virtual keyboard. The results suggest that air-writing is suitable for short and infrequent text input on a motion-based user interface.

REFERENCES

[1] M. Chen, G. AlRegib, and B.-H. Juang, "Air-writing recognition—Part II: Detection and recognition of writing activity in continuous stream of motion data," *IEEE Trans. Human-Mach. Syst*, to be published.

[2] (2015). Leap motion. [Online]. Available: http://www.leapmotion.com

[3] M. Chen, G. AlRegib, and B.-H. Juang, "Feature processing and modeling for 6d motion gesture recognition," *IEEE Trans. Multimedia*, vol. 15, no. 3, pp. 561–571, Apr. 2013.

[4] C. H. Blickenstorfer, "Graffiti: Wow!" *Pen Computing Mag.*, vol. 1, pp. 30–31, Jan. 1995.

[5] K. Perlin, "Quikwriting: Continuous stylus-based text entry," in *Proc. 11th Ann. ACM Symp. User Interface Softw. Technol.*, 1998, pp. 215–216.

[6] P. Isokoski and R. Raisamo, "Quikwriting as a multi-device text entry method," in *Proc. 3rd Nordic Conf. Human-Computer Interaction*, 2004, pp. 105–108.

[7] T. Költringer, P. Isokoski, and T. Grechenig, "Twostick: Writing with a game controller," in *Proc. Graph. Interface*, 2007, pp. 103–110.

[8] (2013). Swype—Type fast, swype faster. [Online]. Available: http://www.swype.com/

[9] R. Plamondon and S. Srihari, "Online and off-line handwriting recognition: a comprehensive survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 1, pp. 63 –84, Jan. 2000.

[10] J. Makhoul, T. Starner, R. Schwartz, and G. Chou, "On-line cursive handwriting recognition using hidden markov models and statistical grammars," in *Proc. Workshop Human Lang. Technol.*, 1994, pp. 432–436.

[11] J. Hu, M. Brown, and W. Turin, "HMM based online handwriting recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 10, pp. 1039–1045, Oct. 1996.

[12] B.-K. Sin and J. H. Kim, "Ligature modeling for online cursive script recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 6, pp. 623–633, Jun. 1997.

[13] J. Mäntyjärvi, J. Kela, P. Korpipää, and S. Kallio, "Enabling fast and effortless customisation in accelerometer based gesture interaction," in *Proc. 3rd Int. Conf. Mobile Ubiquitous Multimedia*, 2004, pp. 25–31.

[14] Z. Zafrulla, H. Brashear, T. Starner, H. Hamilton, and P. Presti, "American sign language recognition with the kinect," in *Proc. 13th Int. Conf. Multimodal Interfaces*, 2011, pp. 279–286.

[15] N. Pugeault and R. Bowden, "Spelling it out: Real-time ASL fingerspelling recognition," in *Proc. IEEE Int. Conf. Comput. Vision Workshops*, 2011, pp. 1114–1119.

[16] C. Amma, D. Gehrig, and T. Schultz, "Airwriting recognition using wearable motion sensors," in *Proc. 1st Augmented Human Intl. Conf.*, 2010, pp. 10:1–10:8.

[17] C. Amma, M. Georgi, and T. Schultz, "Airwriting: Hands-free mobile text input by spotting and continuous recognition of 3D-space handwriting with inertial sensors," in *Proc. 16th Int. Symp. Wearable Comput.*, 2012, pp. 52–59.

[18] L. Jin, D. Yang, L.-X. Zhen, and J.-C. Huang, "A novel vision based finger-writing character recognition system," in *Proc. 18th Int. Conf. Pattern Recog.*, 2006, vol. 1, pp. 1104–1107.

[19] A. Schick, D. Morlock, C. Amma, T. Schultz, and R. Stiefelhagen, "Vision-based handwriting recognition for unrestricted text input in mid-air," in *Proc. 14th ACM Int. Conf. Multimodal Interaction*, 2012, pp. 217–220.

[20] X. Zhang, Z. Ye, L. Jin, Z. Feng, and S. Xu, "A new writing experience: Finger writing in the air using a kinect sensor," *MultiMedia, IEEE*, vol. 20, no. 4, pp. 85–93, Oct.–Dec. 2013.

[21] M. Chen, G. AlRegib, and B.-H. Juang, "6DMG: A new 6D motion gesture database," presented at the 2nd Ann. ACM Conf. Multimedia Syst., Chapel Hill, NC, USA, 2012.

[22] (2015). Worldviz ppt-x. [Online]. Available: http://www.worldviz.com

[23] P. O. Kristensson, T. Nicholson, and A. Quigley, "Continuous recognition of one-handed and two-handed gestures using 3d full-body motion tracking sensors," in *Proc. ACM Int. Conf. Intell. User Interfaces*, 2012, pp. 89–92.

[24] T. Brants and A. Franz, *Web 1T 5-gram Version 1*. Philadelphia, PA, USA: Linguistic Data Consortium, 2006.

[25] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*. Upper Saddle River, NJ, USA: Prentice-Hall, 1993.

[26] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains," *Ann. Math. Statist.*, vol. 41, no. 1, pp. 164–171, 1970.

[27] S. Katz, "Estimation of probabilities from sparse data for the language model component of a speech recognizer," *IEEE Trans. Acoust., Speech Signal Process.*, vol. 35, no. 3, pp. 400–401, Mar. 1987.

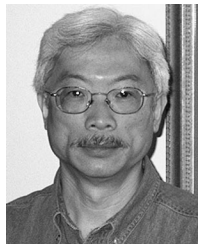
[28] C.-M. Karat, C. Halverson, D. Horn, and J. Karat, "Patterns of entry and correction in large vocabulary continuous speech recognition systems," in *Proc. SIGCHI Conference Human Factors Comput. Syst.*, 1999, pp. 568–575.

Mingyu Chen received the B.S. degree in electrical engineering from the National Taiwan University, Taipei, Taiwan, in 2005, and the Ph.D degree in electrical and computer engineering from the Georgia Institute of Technology, Atlanta, GA, USA, in 2013.

His research interests include motion tracking, motion recognition, and motion-based human–computer interaction.

Ghassan AlRegib (SM'10) received the Ph.D. degree from Georgia Institute of Technology, in 2003, he is currently a Professor with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA. His research interests include image and video processing and communications, immersive communications, collaborative systems, quality of images and videos, and 3-D video processing.

Prof. AlRegib is active within IEEE and the IEEE Signal Processing Society.



Biing-Hwang Juang (F'91) received the Ph.D. degree from the University of California, Santa Barbara, CA, USA.

He is currently the Motorola Foundation Chair Professor and a Georgia Research Alliance Eminent Scholar with the Georgia Institute of Technology, Atlanta, GA, USA.

Dr. Juang became a Fellow of Bell Labs in 1999, a Member of the U.S. National Academy of Engineering in 2004, and an Academician of the Academia Sinica in 2006.