Multifactor User Authentication with In-Air-Handwriting and Hand Geometry

Duo Lu, Dijiang Huang[*], Yuli Deng, Adel Alshamrani
Arizona State University, Tempe, Arizona
{duolu, dijiang.huang, ydeng19, aalsham4}@asu.edu

Abstract

On wearable and Virtual Reality (VR) platforms, user authentication is a basic function, but usually a keyboard or touchscreen cannot be provided to type a password. Hand gesture and especially in-air-handwriting can be potentially used for user authentication because a gesture input interface is readily available on these platforms. However, determining whether a login request is from the legitimate user based on a piece of hand movement is challenging in both signal processing and matching, which leads to limited performance in existing systems. In this paper, we propose a multifactor user authentication framework using both the motion signal of a piece of in-air-handwriting and the geometry of hand skeleton captured by a depth camera. To demonstrate this framework, we invented a signal matching algorithm, implemented a prototype, and conducted experiments on a dataset of 100 users collected by us. Our system achieves 0.6% Equal Error Rate (EER) without spoofing attack and 3.4% EER with spoofing only data, which is a significant improvement compared to existing systems using the Dynamic Time Warping (DTW) algorithm. In addition, we presented an in-depth analysis of the utilized features to explain the reason for the performance boost.

1. Introduction

Virtual Reality (VR) headsets, wearable computers, and other mobile computing platforms with gesture input interface are gaining popularity. Meanwhile, it is necessary to verify the identity of a user to unlock these devices or login onto a remote “virtual site” to access private data or personalized services. In such an environment, presenting a keyboard or touchscreen is usually impractical, while gesture [10, 5] and hand geometry [13] based authentication would be efficient and favorable. For example, VR game consoles (*e.g.*, Sony PSVR) and wearable computing platforms (*e.g.*, Microsoft Hololens) are operated through a native gesture user interface. Hence, hand gestures and especially in-air-handwriting can be considered as a user authentication method for these platforms. However, there are three major technical challenges in the inherent characteristics of hand gestures. First, tracking the position of the hand and fingers with enough resolution and accuracy is difficult, especially when they move quickly. Second, there are minor variations in the writing speed and shape even for the same content by the same person, which must be tolerated by the matching algorithm. Third, it is unclear what features in the gesture can be used to distinguish legitimate users from others. Therefore, various algorithms and machine learning models are employed in existing systems, whose performance can rarely be comparable to traditional biometrics such as the fingerprint on a dataset with reasonable size.

In this paper, we propose a multifactor authentication framework which can authenticate a user by observing the shape of the user’s hand and asking the user to write a passcode in the air with the index finger. The content of the passcode can be a meaningful phrase, a piece of scribbling like a signature, or a doodle as long as it can be easily reproduced by the same user but difficult to mimic by attackers. This piece of in-air-handwriting, as well as the hand geometry, is captured by a depth camera and compared with a template created at registration. We fuse three modalities including a secret (*i.e.,* the passcode), a behavior biometric (*i.e.,* the handwriting convention), and a physiological trait (*i.e.,* the hand geometry) in one unified framework, which provides better security than password based authentication systems. Unlike traditional biometrics such as the fingerprint, iris and face, the gesture passcode we use is revocable. Our contributions in this paper are as follows:

1) We provide an analysis on the motion signals as well as hand geometry features, which helps us understand and construct the model to verify the identity of a user.

2) Based on the analysis, we invent a matching algorithm fusing the information from both hand motion and hand geometry, which is able to tolerate limited capability of the gesture input device and the variation of the writing. Additionally, our algorithm can work with only two repetitions of the in-air-handwriting for the template construction at registration, instead of requiring large training data com-

[*]This research is supported by NSF (award 1528099).

pared to other machine learning based systems.

3) We build a prototype system and provide a thorough evaluation with three datasets we collected from 100 users. Our system achieves 0.6% Equal Error Rate (EER) without spoofing and 3.4% EER with spoofing only data, which outperforms existing systems based on DTW.

The remainder of the paper is organized as follows. Section 2 describes related works on gesture authentication systems. Section 3 presents the architecture of our framework, and section 4 explains the matching and score fusion algorithms. Section 5 shows the empirical results, and finally, section 6 draws conclusions and discusses the future work.

2. Related Work

Early gesture based authentication methods use the accelerometer to track simple motions like shaking [11], walking [9], or other predefined gesture patterns [7], which proves the feasibility of gesture based authentication. Fine hand gestures like in-air-handwriting have better potential because more information in the complex motions can be utilized to distinguish different users, and the handwriting movements are usually captured by a smartphone [8], smartwatch [12], data glove [10], or other handhold remote control device [3]. However, waving such devices to mimic writing is unnatural, which may lead to unstable patterns and memory burden.

In-air-handwriting can also be obtained by camera based gesture interface such as Kinect [15, 17, 16], Google Glass [14], and Leap Motion controller [4, 2, 5]. Their algorithms are based on template matching [15], comparing frequency or statistical components [11, 18], or machine learning models such as random forest [4, 5]. The common challenge is tolerating the imperfection in motion tracking and user behavior variations. Moreover, because of the lack of feature analysis, it is not clear what information can be reliably used for decision making. Based on the assumption that writing the same word generates motion signals of similar shape, the DTW algorithm is the most widely used matching algorithm due to its effectiveness, simplicity, and efficiency [3]. Among the existing work, only a few systems join the gesture and the hand geometry for authentication [5]. Furthermore, due to the very nature of gesture authentication, it is challenging to collect a large scale dataset, and hence, comparing the performance of different algorithms on the same dataset [6].

3. System Architecture

The proposed authentication framework consists of five components (shown in Figure 1): a hand motion tracking device (*e.g.,* Leap Motion controller [1]), an account database, two matching algorithms, and the decision maker which fuses the matching scores and decides whether the authentication request is a match or a non-match.

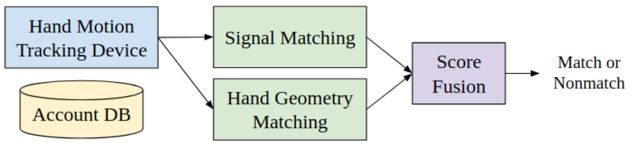

Figure 1. Authentication framework architecture.

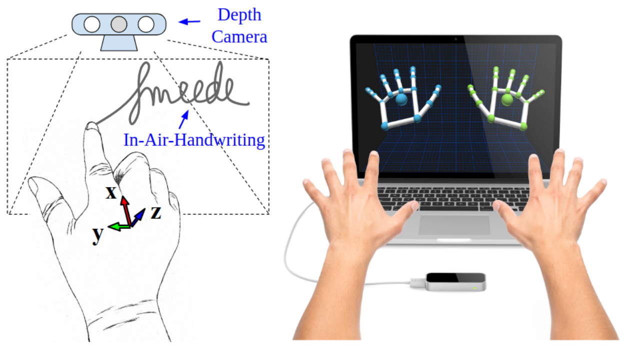

Figure 2. Hand motion tracking device, an illustration (left) and the Leap Motion controller [1] used in our prototype system (right).

3.1. Hand Motion Tracking Device

When a user writes in the air, a contactless device captures the motion signal which contains a series of physical states of the hand sampled at a certain rate, such as the position trajectory. In our prototype system we use the Leap Motion controller, which is able to provide 3D coordinates of each joint of the hand in sub-millimeter precision at around 110 Hz. Yet we discover that the accuracy of motion tracking signal is not very reliable, *e.g.,* sometimes it cannot see the index finger when the hand points in certain directions and it often misrecognizes a right hand facing downward as a left hand facing upward. As a result, only the position of the center of the hand is used. The motion signal is denoted as a matrix R, where each column represents one axis (*i.e.,* x, y or z of position), and each element is an individual sample in an axis at a specific time (denoted as R_{ij}). R is further preprocessed and aligned to a template signal T to obtain an l by d matrix S. d is the number of axes (usually more than 3 because we can also derive speed and acceleration in x, y, z from the time-stamped position series), and l is the number of samples (usually from 100 to 500). Besides the motion signal, the device also records a vector $\mathbf{h}$ consisting of the length of each bone of the user's hand as well as hand width (detailed in section 4.2). Usually R and $\mathbf{h}$ are sent to the authentication server as an authentication request. It should be noted that as long as the hand motion tracking device can provide position samples and geometry of the hand, our framework can be applied. Our matching algorithm is not restricted to the Leap Motion controller.

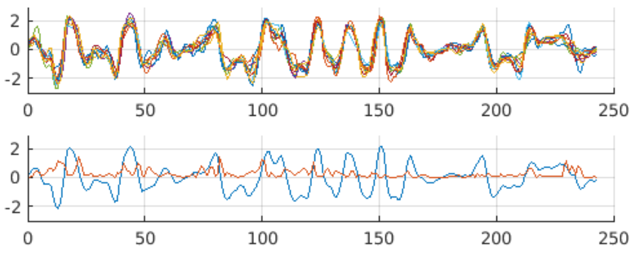

Figure 3. Example of 10 aligned in-air-handwriting signals (upper) and the generated template with variance (lower). Variance is enlarged 5 times to show significance.

3.2. Account Database

Each account in the authentication server is a four tuple of <ID, T, C, $\mathbf{t}$ >. Our framework allows one user to possess multiple accounts (*i.e.,* multiple distinct account IDs). Here T is the signal template, C is the variance with the same dimension as T, and $\mathbf{t}$ is the hand geometry template. T, C and $\mathbf{t}$ are all derived from a few preprocessed and aligned example signals $\{S^1, S^2, ..., S^k\}$ and example hand geometry vectors $\{\mathbf{h}^1, \mathbf{h}^2, ..., \mathbf{h}^k\}$ at registration as follows:

$$\mathbf{t} = mean(\mathbf{h}^1, \mathbf{h}^2, ..., \mathbf{h}^k),$$

$$T_{ij} = mean(S^1_{ij}, S^2_{ij}, ..., S^k_{ij}),$$

$$C_{ij} = var(S^1_{ij}, S^2_{ij}, ..., S^k_{ij}).$$

Templates can be updated explicitly by the user or implicitly on a successful authentication attempt S and $\mathbf{h}$ to adapt minor changes of user behaviors in the following way:

$$\mathbf{t} \leftarrow (1-\lambda)\mathbf{t} + \lambda\mathbf{h},$$

$$T_{ij} \leftarrow (1-\lambda)T_{ij} + \lambda S_{ij},$$

$$C_{ij} \leftarrow (1-\lambda)C_{ij} + \lambda(S_{ij} - T_{ij})^2.$$

Here λ is the update rate that controls the influence of the newly accepted signal, determined empirically (discussed in detail in section 5). C may be updated gradually to very large values if the behavior changes are large, which may need further regularization to keep the decision boundary in the same place. An example of aligned signals and the generated template is shown in Figure 3.

3.3. Matching Algorithms and Score Fusion

Once the authentication request is received, the server runs the following two matching algorithms in parallel:

1) Signal matching, which matches the motion signal R with the stored signal template T with auxiliary information from the variance C, detailed in section 4.1.

2) Hand geometry matching, which matches the hand geometry $\mathbf{h}$ with the stored hand geometry template $\mathbf{t}$ of the account owner, detailed in section 4.2.

After the results of the two matching algorithms are obtained, the authentication server fuses the results and makes the final decision, detailed in section 4.3.

3.4. Datasets

We collected the following three datasets for evaluation:

1) We asked 100 users to create 200 accounts, where each user created exactly two accounts. For each account, the user wrote a passcode in the air five times as the registration and five more times as authentication requests. Among the 100 users, 50 are male, the other 50 are female, and their age spans from 15 to 62. They have various educational backgrounds from middle school student to university professors, and diverse occupations including both office workers and non-office workers. Each user created his or her own passcode without any constraints on the content or the length, but none of the 200 passcodes are identical.

2) We asked seven impostors to write each passcode in the first dataset for five times as spoofing attacks. The impostors are informed with the content of the passcode.

3) Additionally, for 22 users participating in the first dataset, we asked them to write the passcodes of the corresponding 44 accounts five times every three or four days, which is denoted as a session. In total there are 10 sessions, lasting for four weeks. This dataset is used in the study of long term stability of our algorithms, detailed in section 5.

Given an account with template signal T, each testing signal S obtained from the first dataset can be associated with a class label c as "true-user" or "false-user". If S and T are from the same user writing the correct passcode, $c =$ true-user; otherwise $c =$ false-user. Especially, if S is from the spoofing attack in the second dataset, $c =$ spoof. "false-user" and "spoof" are collectively called "not-true-user". Without special notification, we use five signals to construct the template.

4. Matching Algorithm Details

4.1. Signal Matching

The signal matching algorithm defines a distance between the signal in the authentication request and the stored template. It contains four stages (shown in Algorithm 1): signal preprocessing, signal alignment, element-wise distance scaling, and threshold-then-vote (TTV). The TTV procedure represents the major difference between our algorithm and traditional DTW algorithm, and hence, our signal matching algorithm is called the TTV algorithm.

4.1.1 Signal Preprocessing

To improve signal quality and facilitate matching in later stages, the raw signal obtained from the hand motion capture device goes through the following preprocessing steps:

Step 1) Fix missing data samples due to the limited capability of the motion capture device by linear interpolation.

Step 2) Derive velocity and acceleration for each sample from the time-stamped trajectory.

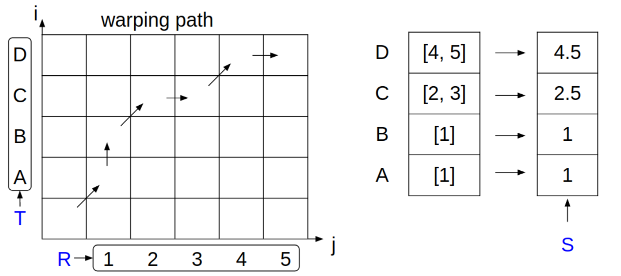

Figure 4. Example of temporal alignment: the warping path (left), and the alignment result (right).

Step 3) Remove any high frequency components above 10 Hz (low-pass filtering).

Step 4) Throw away the samples at the start and the end of the signal when the hand stays still.

Step 5) Translate the coordinate system, *i.e.,* make the average pointing direction of the hand as the x-axis.

Step 6) Down-sample the signal to 50 Hz.

Step 7) Normalize each column to zero mean and unit variance.

4.1.2 Signal Alignment

To accommodate small differences in writing speed, the raw signal R is aligned to the template signal T before matching. First, we run dynamic time warping on R and T to obtain a warping path, with a window constraint of ± 50 samples (*i.e.,* ± 1 second). Then each sample S_{ij} of the aligned signal S is calculated by taking the average of a range of samples in the original signal mapped to T_{ij} on the warping path. Unlike most of the related works which use DTW to directly calculate distance of two signals, we only employ DTW for temporal alignment. An example of signal alignment given the warping path is shown in Figure 4. Here we try to align the signal (1, 2, 3, 4, 5) to the template (A, B, C, D). Given the warping path, A is mapped to 1, B is mapped to 1, C is mapped to 2.5 (*i.e.,* average of 2 and 3), D is mapped to 4.5 (*i.e.,* average of 4 and 5).

4.1.3 Element-wise Distance Scaling

After the alignment, element-wise distance D is calculated,

$$D_{ij} = |S_{ij} - T_{ij}|$$

and two element-wise scalars P and Q are computed,

$$P_{ij} = 1/(1 + p \times T_{ij}),$$

$$Q_{ij} = 1/(1 + q \times C_{ij}).$$

Then D is scaled element-wise by P and Q. The logic is that less importance (*i.e.,* more tolerance) should be applied if the template has a larger value (*i.e.,* higher intensity) or a

Algorithm 1: signal_match()

```
input      : $R, T, C$
parameter: $p, q, th_1, th_2$
output     : dist
$R' \leftarrow$ preprocess($R$).
$S \leftarrow$ align($T$, $R'$).
$D \leftarrow$ element-wise-abs($T$ - $S$).
$P \leftarrow$ element-wise-inverse($\mathbf{1} + p \times T$)
$Q \leftarrow$ element-wise-inverse($\mathbf{1} + q \times C$)
$D \leftarrow$ element-wise-multiply($D$, $P$).
$D \leftarrow$ element-wise-multiply($D$, $Q$).
for $i$ = 1 to $l$ do
    for $j$ = 1 to $d$ do
        if $D_{ij} < th_1$ then
            $D_{ij} \leftarrow 0$
        else if $D_{ij} > th_2$ then
            $D_{ij} \leftarrow 1$
        else
            $D_{ij} \leftarrow 0.5$
        end
    end
end
dist $\leftarrow \frac{1}{l \times d} \sum_i^l \sum_j^d D_{ij}$
```

larger variance (*i.e.,* higher uncertainty). This phenomenon is discovered in [10] on signals of inertial sensor but it is also applicable to our framework. p and q are both tweakable parameters (in our prototype system $p = 1$, $q = 0.5$) determined by empirical results, in a similar way as [10].

4.1.4 Threshold-Then-Vote (TTV)

Once the scaled element-wise distance is obtained, we further trim the beginning 5% and ending 10% of the signal as well as the template. Then the final signal level distance is calculated as follows:

$$dist(S,T) = \frac{1}{l \times d} \sum_i^l \sum_j^d TTV(D_{ij} \times P_{ij} \times Q_{ij}).$$

Here l is the trimmed template length, and d is the number of sensor axes. $TTV(x)$ is a step function determining whether a single sample point is matched or not, defined as follows:

$$TTV(x) = \begin{cases} 0, & \text{if } x \le th_1 \\ 0.5, & \text{if } th_1 < x \le th_2 \\ 1, & \text{if } x > th_2 \end{cases} \quad (1)$$

This stage maps element-wise distance to “match” (*i.e.,* 0), “non-match” (*i.e.,* 1), or “unsure” (*i.e.,* 0.5), then treats each element-wise distance as a vote. The insight is to

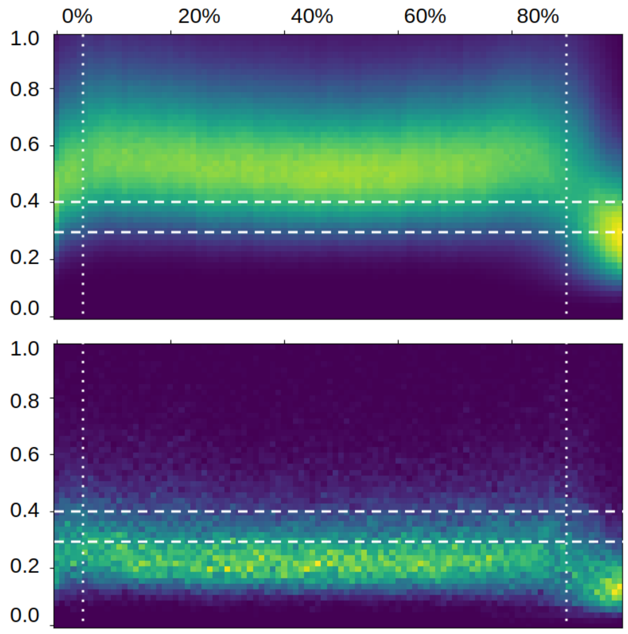

Figure 5. Distribution of the scaled element-wise distance D_{ij} respect to the rows from the false-user class (upper) and true-user class (lower). The horizontal axis is the i direction, in percentage of the length of the aligned signal. The vertical axis is the value of D_{ij}. The brightness denotes the distribution.

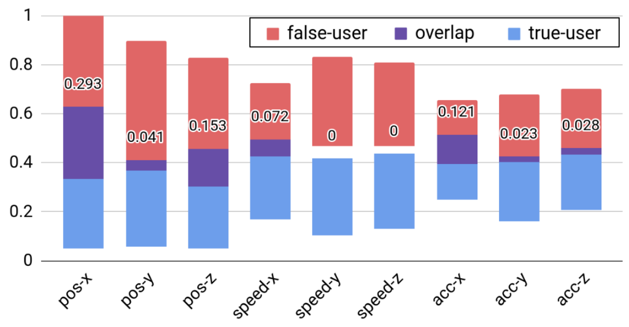

Figure 6. $\pm 2\sigma$ ranges of the scaled element-wise distance D_{ij} respect to the sensor axes from the false-user class and true-user class. The annotated numbers denote the size of the range overlap.

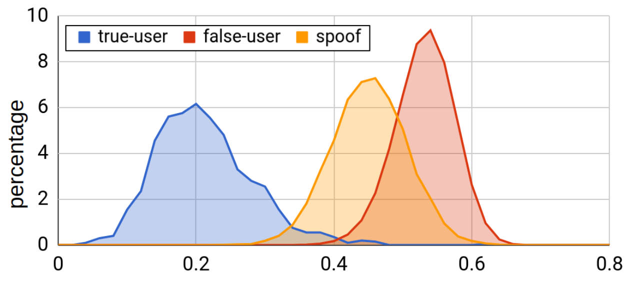

Figure 7. Distribution of signal level distances.

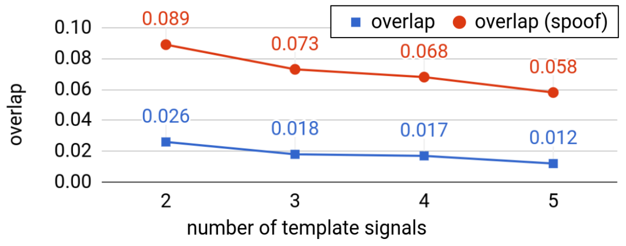

Figure 8. Influence of the number of signals for templates in terms of the overlap of signal level distance between the true-user class and the false-user class (blue-square), as well as between the true-user class and the spoof class (red-circle).

prevent large scarce element-wise distance (*i.e.,* local non-matches) from propagating to the final signal level distance. On the one hand, such local non-matches appear quite often when the signal and the template are obtained from the same user writing the same passcode, which should be tolerated during the decision making stage, but such tolerance is omitted by DTW. On the other hand, signals and templates obtained from different users or from writing different passcodes are different everywhere, and the voting result is likely to be dominated by non-matches even with the tolerance of a few local non-matches.

4.1.5 Analysis

First, we show the distribution of the scaled element-wise distance in respect to the rows averaged over all sensor axes and over all accounts, *i.e.,* an approximation of the class-conditional probability $p(\frac{1}{d}\sum_j^d D_{ij}|c)$, shown in Figure 5. From the figure, it can be observed that local differences between the signal and the template are generally evenly spread along the rows given a large amount of accounts, while the beginning 5% and the ending 10% have anomalies (shown as the dotted vertical lines, which determines the trimming parameters in the TTV algorithm). This is mainly caused by the nature of the alignment algorithm which forces the beginning and the end to be mapped together. For signals and templates generated by the same user, the element-wise distance is smaller than those generated by different users or by writing different passcodes. Still, their distributions overlap between 0.3 and 0.4 (shown as the dashed horizontal lines, which determine the th_1 and th_2 parameters in the TTV algorithm). It can also be observed that for the occasional instances of D_{ij} greater than 0.4 in the true-user class. These distances are tolerated due to their limited amount of non-match votes, but for those in the false-user class, such large distances dominate the votes.

Second, we show the distribution of the element-wise distance D_{ij} in respect to the columns averaged over all rows and over all accounts, *i.e.,* an approximation of the class-conditional probability $p(\frac{1}{l}\sum_i^l D_{ij}|c)$, shown in Figure 6. This figure tells us the quality of the signal in each

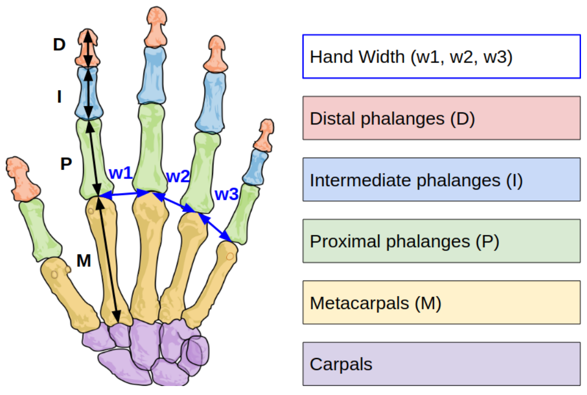

Figure 9. Skeleton Geometry of Human Hand.

sensor axis, *i.e.,* larger overlap indicates more difficulty in distinguishing the two classes based on that column. In general, the element-wise distances in the x direction vary greater than other directions even for the same user writing the same passcode. This is reasonable because the way we write in the air resembles writing in an invisible surface, and here the x axis is defined as the averaged pointing direction of the index finger. Also the element-wise distances in position columns are more widely spread, which means that human users are better at maintaining the same speed and force (which is equivalent to acceleration) than maintaining the same trajectory.

Third, we show the normalized histogram of the signal level distances $dist(S,T)$ over all accounts in Figure 7, *i.e.,* an approximation of the class-conditional probability $p(dist(S,T)|c)$. The overlap between the histograms of different classes denotes the discriminating capability of the signal matching algorithm as a whole according to the Bayes Decision Theory. Our prototype system achieves 1.2% overlap between the true-user class and the false-user class, as well as 5.8% overlap between the true-user class and the spoof class. This also shows that different people write the same content in significantly different ways.

Fourth, we show the influence of the number of signals used to build the template in Figure 8. Clearly more signals at registration help to construct a better template, but the benefit is marginal. Our framework can achieve acceptable results even with only two signals for template construction.

4.2. Hand Geometry Matching

In our framework, hand geometry is defined as the length of each bone of the hand (excluding the wrist) as well as the width of the hand collectively, shown in Figure 9. The bone length features contain 19 components, which are the lengths of metacarpals, proximal phalanges, intermediate phalanges, and distal phalanges of the five fingers on a hand (note that the thumb does not have intermediate phalanges).

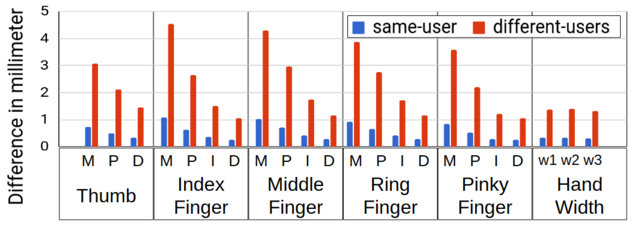

Figure 10. Average length difference of hand bones in millimeter.

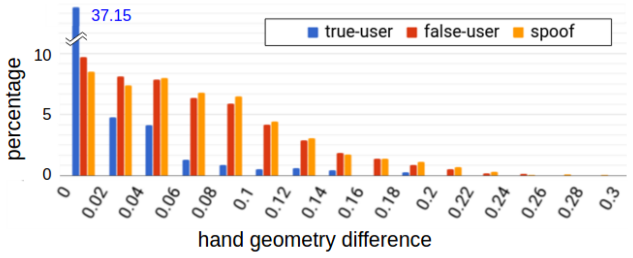

Figure 11. Distribution of hand geometry difference in ratio.

The hand width features include 3 components, which are the distances between the far ends of metacarpals of two adjacent fingers except the thumb (shown in the figure as w1, w2, and w3). Collectively, the hand geometry vector $\mathbf{h}$ is constructed by these 22 components, so is the hand geometry template $\mathbf{t}$.

If the authentication requester is the account owner, they must have the same hand for writing, and hence, the hand geometry in the request should not differ from the stored hand geometry template significantly. We measured the difference of each individual hand geometry component of the same user as well as different users, shown in Figure 10. Obviously, the average length differences of different hands are greater than the same hand. However, it is highly possible that two different hands have similar shapes and there is no way to determine whether they are from the same user merely based on the geometry. Also the hand motion tracking device needs to obtain the hand geometry measurement from the user's hand image in sub-millimeter precision, which is challenging. Thus, each measurement of the same hand will vary slightly, even the sensor itself reports 100% confidence. Another important fact is that all these 22 components have strong correlations, because the five fingers on the same hand grow together. Based on these factors, we define the hand geometry difference as follows:

$$dist(\mathbf{h},\mathbf{t}) = \frac{1}{22}\sum_i \frac{\mathbf{h}_i - \mathbf{t}_i}{\mathbf{t}_i}$$

The distribution of $dist(\mathbf{h},\mathbf{t})$ is shown in Figure 11. In general, hand geometry is considered a weak feature in the decision-making procedure.

Table 1. Empirical Results of Authentication

Algorithm	EER	EER (spoof)	FMR 10K	FMR 100K	Zero -FMR
TTV	0.8%	3.7%	3.9%	6.9%	9.1%
FUSION	**0.6%**	**3.4%**	**3.6%**	**6.2%**	**6.3%**
DTW	1.3%	4.8%	7.8%	20.0%	22.0%
FUSION†	1.3%	5.0%	8.4%	13.0%	18.0%
DTW†	2.0%	6.5%	11.0%	15.0%	17.0%

4.3. Score Fusion

The final score of an authentication request is calculated by fusing the information from the signal matching and the hand geometry matching as follows:

$$score = dist(S,T) + w_1 dist(\mathbf{h},\mathbf{t}) + w_2|l_T - l_R|/l_T$$

Essentially, the final score is a weighted sum of signal level distance, hand geometry difference, and signal length difference, which is the third term $|l_T - l_R|/l_T$, where l_T is the length of the template, and l_R is the length of the signal in the authentication request before alignment. Here the two weights w_1 and w_2 are determined empirically (in our prototype system $w_1 = 0.4$ and $w_2 = 0.05$). Since hand geometry difference and signal length difference are all weak features, their weights are smaller than 1 and they largely depend on the correlation to the signal level distance.

Finally, our algorithm uses the signal level distance to make the decision, *i.e.,* if the score is below the decision threshold, the request is accepted and hence the user passes authentication; otherwise, the request is rejected and the user fails authentication. The decision threshold is set as a trade off between accuracy and convenience. If the decision threshold is small (*i.e.,* strict), legitimate users may be rejected unexpectedly, while if the decision threshold is large (*i.e.,* loose), attackers may be accidentally accepted. In the next section, we show how to determine the decision threshold by empirical results.

5. Empirical Results

Using the fused score described above, we run experiments on the first and second datasets while varying the decision threshold, and the results are shown in Table 1. Here EER is the rate when False Non-Match Rate (FNMR) is equal to the False Match Rate (FMR) (also shown in Figure12). FMR 10K, FMR 100K, and Zero-FMR are the corresponding FNMR when FMR is 10^{-4}, 10^{-5}, and zero respectively, also shown in the Receiver Operating Characteristic (ROC) curve in Figure 13. Compared with the traditional DTW, the EER of our framework drops more than half, which is a significant performance improvement. In Table 1, we also show the performance under the condition of only two signals for template construction as the rows

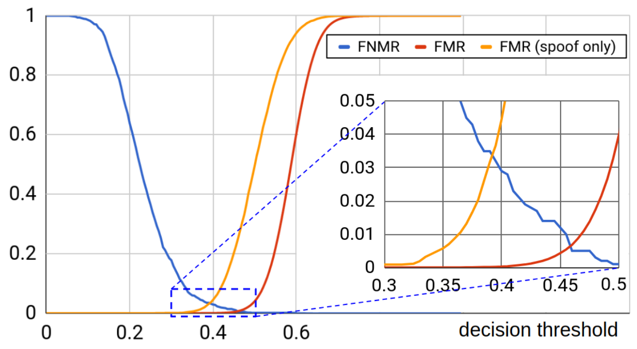

Figure 12. False Non-Match Rate (FNMR) and False Match Rate (FMR). EER is shown in the enlarged rectangle.

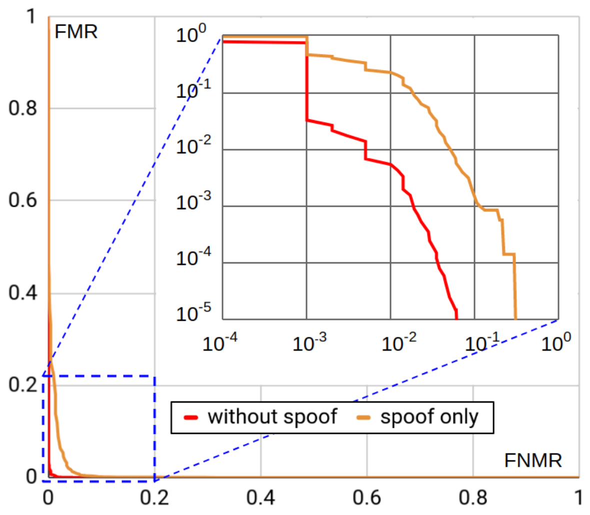

Figure 13. Receiver Operating Characteristic (ROC).

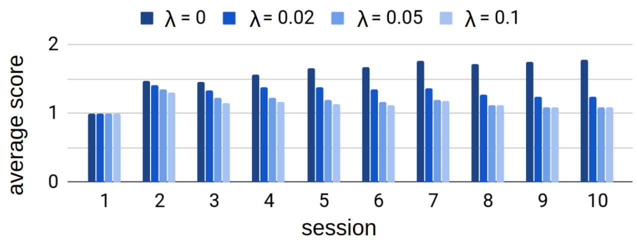

Figure 14. Long term changes of the scores of the true-user class averaged over all accounts (relative to the first session) with different template updating factors.

of FUSION† and DTW†. This indicates that our framework has acceptable performance and outperforms DTW even with extremely limited example signals for template construction, which requires less effort of the user at registration, and hence, better usability.

Besides, we also run experiments on the third dataset and show the changes of the fused score over the 10 sessions in Figure 14. Without template update, the score of the true-user class will increase to around 1.75 times relative to the first session. However, with a template update of 0.1, the score increase converges to about 1.09 times. This effect is mainly caused by the limited number of sample signals for template construction, as explained in section 4.1.5. A deeper reason may be the inherent complexity of in-air-handwriting, especially considering the writing behavior variation, which is difficult to represent merely with a template built from several example signals. The long-term stability of the authentication performance differs by account, *e.g.,* for most accounts, the score at the first session is far away from the decision threshold and the score increase with time will not generate false non-matches.

The performance improvement of our framework comes from several aspects. First, the preprocessing steps retains valuable information for authentication and makes the matching algorithm robust against poor signal quality and small variation in users' postures. Second, the threshold-then-vote procedure prevents the discriminating capability degradation caused by locally mismatched signal segments, which can be justified by the performance of the TTV algorithm in the first row of Table 1. Third, fusing hand geometry and length difference with the signal level distance further removes many false matches.

6. Conclusion and Future Work

In this paper, we proposed a multifactor user authentication framework with both in-air-handwriting and hand geometry. This framework is especially useful for wearable and VR platforms with gesture input interface. We built a prototype system, collected a dataset of 100 users, and evaluated the system, which shows good performance with 0.6% EER. Although there are limitations such as long term permanence, requirement of template protection on the server side, need of parameter tweaking effort, and constraints of sensor accuracy, we believe there is still a great potential in this hand gesture based authentication method. In the future, we will carry on an in-depth study on the influence of the passcode content and strength, and verify the performance of our framework on bigger datasets with other types of gesture tracking sensors.

References

[1] Leap motion controller. URl: https://www.leapmotion.com.

[2] I. Aslan, A. Uhl, A. Meschtscherjakov, and M. Tscheligi. Mid-air authentication gestures: an exploration of authentication based on palm and finger motions. In *Proceedings of the 16th International Conference on Multimodal Interaction*, pages 311–318. ACM, 2014.

[3] G. Bailador, C. Sanchez-Avila, J. Guerra-Casanova, and A. de Santos Sierra. Analysis of pattern recognition techniques for in-air signature biometrics. *Pattern Recognition*, 44(10):2468–2478, 2011.

[4] A. Chahar, S. Yadav, I. Nigam, R. Singh, and M. Vatsa. A leap password based verification system. In *Biometrics Theory, Applications and Systems (BTAS), 2015 IEEE 7th International Conference on*, pages 1–6. IEEE, 2015.

[5] A. Chan, T. Halevi, and N. Memon. Leap motion controller for authentication via hand geometry and gestures. In *International Conference on Human Aspects of Information Security, Privacy, and Trust*, pages 13–22. Springer, 2015.

[6] G. D. Clark and J. Lindqvist. Engineering gesture-based authentication systems. *IEEE Pervasive Computing*, 2015.

[7] E. Farella, S. OModhrain, L. Benini, and B. Riccó. Gesture signature for ambient intelligence applications: a feasibility study. In *International Conference on Pervasive Computing*, pages 288–304. Springer, 2006.

[8] F. Hong, M. Wei, S. You, Y. Feng, and Z. Guo. Waving authentication: your smartphone authenticate you on motion gesture. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, pages 263–266. ACM, 2015.

[9] J. Lester, B. Hannaford, and G. Borriello. are you with me?–using accelerometers to determine if two devices are carried by the same person. In *International Conference on Pervasive Computing*, pages 33–50. Springer, 2004.

[10] D. Lu, K. Xu, and D. Huang. A data driven in-air-handwriting biometric authentication system. In *International Joint Conf. on Biometrics (IJCB 2017)*. IEEE, 2017.

[11] R. Mayrhofer and H. Gellersen. Shake well before use: Authentication based on accelerometer data. In *International Conference on Pervasive Computing*. Springer, 2007.

[12] B. Nassi, A. Levy, Y. Elovici, and E. Shmueli. Handwritten signature verification using hand-worn devices. *arXiv preprint arXiv:1612.06305*, 2016.

[13] A. Ross, A. Jain, and S. Pankati. A prototype hand geometry-based verification system. In *Proceedings of 2nd conference on audio and video based biometric person authentication*, pages 166–171, 1999.

[14] H. Sajid and S. C. Sen-ching. Vsig: Hand-gestured signature recognition and authentication with wearable camera. In *Information Forensics and Security (WIFS), 2015 IEEE International Workshop on*, pages 1–6. IEEE, 2015.

[15] J. Tian, C. Qu, W. Xu, and S. Wang. Kinwrite: Handwriting-based authentication using kinect. In *NDSS*, 2013.

[16] J. Wu, J. Christianson, J. Konrad, and P. Ishwar. Leveraging shape and depth in user authentication from in-air hand gestures. In *2015 IEEE International Conference on Image Processing (ICIP 2015)*, pages 3195–3199. IEEE, 2015.

[17] J. Wu, J. Konrad, and P. Ishwar. Dynamic time warping for gesture-based user identification and authentication with kinect. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013.

[18] A. Zaharis, A. Martini, P. Kikiras, and G. Stamoulis. User authentication method and implementation using a three-axis accelerometer. In *International Conference on Mobile Lightweight Wireless Systems*. Springer, 2010.