Mobile User Authentication Using Statistical Touch Dynamics Images

Xi Zhao, Member, IEEE, Tao Feng, Student Member, IEEE, Weidong Shi, Senior Member, IEEE, and Ioannis A. Kakadiaris, Senior Member, IEEE

Abstract—Behavioral biometrics have recently begun to gain attention for mobile user authentication. The feasibility of touch gestures as a novel modality for behavioral biometrics has been investigated. In this paper, we propose applying a statistical touch dynamics image (aka statistical feature model) trained from graphic touch gesture features to retain discriminative power for user authentication while significantly reducing computational time during online authentication. Systematic evaluation and comparisons with state-of-the-art methods have been performed on touch gesture data sets. Implemented as an Android App, the usability and effectiveness of the proposed method have also been evaluated.

Index Terms—Touch gesture, user authentication, mobile security, statistical touch dynamics images, behavioral biometrics.

I. INTRODUCTION

THE increasing ubiquity of smartphones raises the issue of security of paramount importance. The data stored and accessed from such devices are rarely protected except by a point-based authentication at the login stage. This is deficient since it cannot detect intruders after a user has passed the point entry and is vulnerable to simple attack (e.g., smudge attacks [1]) after loss or theft. As opposed to password or image-pattern-based authentication, continuous authentication has received increased attention in recent years. It intelligently monitors and analyzes the user-device interaction to ensure correct identity, which either complements point-based authentication or even substitutes it when continuous authentication satisfies particular accuracy requirements.

While using a smartphone, user identity can be sensed via multiple integrated sensors (e.g., the touch screen, motion sensors, microphone, or camera). Users’ faces [2] and voices [3] can be used for authentication, but it is not convenient or optimal to apply these approaches while the user is using most of the apps (e.g., reading news and email, surfing the internet). Users must face the camera or keep talking to be continuously authenticated, which greatly reduces the device usability. Authentication using motion data (i.e., Accelerometer/Gyroscope data) has been studied. However, due to signal noise and difficulty in decoupling identity from activity, this modality currently can only be applied on motion-specific activities (e.g., phone pick-up motion during answering the call [4]). Conversely, the touch screen is quite suitable for authentication, not only because it is the most frequently used sensor but also the touch gesture contains identity-rich user behavioral traits. The feasibility of using touch gestures as a biometric modality has been investigated [5]–[8]. It can be observed in Fig. 1 that most of the touch traces from one user tend to follow a similar pattern while the patterns vary among different users. Intrinsically, touch traces are affected by two biometric features (i.e., user hand geometry and finger muscle behavior).

Fig. 1. Plot (a) depicts over 300 touch traces of zoom-in gesture from 30 users. Each subfigure contains around 10 traces from a single user. Plots (b) and (c) are the enlarged figures depicting the first and second users’ touch gestures.
In our previous work, we evaluated the Graphic Touch Gesture Feature (GTGF) for touch based user authentication [8]. The GTGF can represent touch dynamics in an explicit manner where the discriminative power from touch trajectories and tactile pressure is combined. However, implemented as an Android App, this method suffers from computational overhead and incurs a noticeable run-time delay when the gallery size increases because a probe trace must be compared with each trace in the gallery. To improve the effectiveness of GTGF and reduce the computational expense at the same time, we propose to apply a Statistical Feature Model (SFM) as Touch Dynamics Images (STDI), which builds a touch gesture “appearance” model from the GTGF. STDI learns the intra-person variations of the GTGF in the image domain by statistical analysis and is capable of synthesizing new instances according to a new probe trace. Instead of directly computing the distance between the probe trace and the gallery traces, the distance is computed between the probe trace and its corresponding synthesized instance given a user-specific STDI. The synthesized instance from the STDI mirrors the probe trace. The instance tends to be more dissimilar to the trace if the trace originates from an identity other than the genuine user. We evaluated the proposed method on three sets of touch gestures (flick up/down, flick right/left, zoom in/out) simply because they are among the most used gestures during user-device interaction [9]. Flick up/down is generally used when reading text or scrolling menus. Flick right/left is generally used when browsing photos, translating screens, and unlocking phones, while zoom in/out is generally used whenever users switch between details and overviews of the contents.

Our main contributions are: (i) applying the feature representation principle in SFM to touch gestures by STDI, which retains the effectiveness of previous GTGF and is more efficient in terms of computation; (ii) developing the algorithms to apply the STDI on the mobile user verification and recognition problems; (iii) systematically evaluating the proposed algorithm by varying different parameters, and providing a comparison between the proposed algorithm with the state-of-the-art methods; and (iv) developing an Android app and testing performance of the verification algorithm in a real-world scenario.

The rest of the paper is organized as follows. First, we discuss the related work in Section II. Methods are presented in Section III. GTGF extraction is presented in Section III-A. The STDI is described in Section III-B. The user authentication algorithms are presented in Section III-C. Section IV presents the experimental results. In Section V, we conclude our study.

II. RELATED WORKS

Mobile User Authentication can be categorized into two groups: the physiological biometrics, which rely on static physical attributes (e.g., fingerprints [10], facial features [11]) and behavioral biometrics, which adopt identity-invariant features of human behavior to identify or verify people during their daily activities [12].

Prior research on behavioral biometrics has shown that biometric traits can be extracted from physiological characteristics of arm gestures [4], [13], strokes and signatures using a pen or stylus [14], [15]. Yamazaki et al. [16] extracted personal features from stroke shape, writing pressure and pen inclination to identify an individual. Cao et al. [17] discussed a quantitative human performance model of making single-stroke pen gestures. Ferrer et al. [18] used local patterns (e.g., the Local Binary Pattern and the Local Directional Pattern), to segment and extract identity features from signatures. Impedovo et al. [19] presented the state-of-the-art in automatic signature verification. However, verifying users with pen strokes cannot be pervasively applied in mobile user authentication since only a few types of mobiles provide a pen or stylus. Furthermore, most users opt to interact with their phone directly using their fingers for convenience.

Studies exploring the touch gesture as a biometric modality for mobile devices are quite recent [20]. Feng et al. [5] extracted finger motion speed and acceleration of touch gestures as features for identification. Luca et al. [21] directly computed the distance between traces using the dynamic time warping algorithm. Sae-Bae et al. [6] designed 22 touch gestures for authentication, involving different combinations of five fingers. They computed the dynamic time warping distance and Frechet distance between multi-touch traces. However, users must perform their pre-defined touch gestures for authentication, which thus cannot be transparent and non-obtrusive to users. Frank et al. [7] studied the correlation between 30 analytic features from touch traces and classified these features using K-nearest-neighbor and Support Vector Machine approaches. However, the analytic features did not capture the touch trace dynamics. The proposed method is quite different from existing works. The pressure dynamics are adopted in our method and are represented as the profiles of the STDI instances. This is different from extracting a single mid-stroke pressure value [7] or simply being ignored as in other studies [6], [21]. Rather than directly using GTGF for authentication [8], the STDI applies the similar feature representing principle of the statistical feature model in [24] to learn the intra-person variations from many traces by a user so that scores are computed between pairs of gallery models and probe traces, instead of using pairs of gallery traces and probing traces, which significantly reduces the computational time.

III. METHODS

The authentication scenarios considered in this paper include open set user recognition and user verification. In the open set user recognition scenario, mobile devices are shared among a few users (e.g., family members). The sensitive group data should be shared only among the authorized users, and users’ identities should first be verified to ensure they belong to the authorized group. Meanwhile, the user should be further recognized to customize the user experience on the device (e.g., limiting the functions for a child, granting different access levels). In a more general scenario, phones are used only by their owner. Other individuals are considered intruders and should be excluded from using the phone. Thus, the method verifies whether the device is being used by its owner.
To perform user authentication, we capture the touch traces from the touch screen output and segment the traces with support from the Android API. We use a series of x-y coordinates of finger touch points, pressure values and time stamps in generating GTGF feature representation. Each series varies in time duration, trace length, dynamics of finger movements and tactile pressure, which is dependent on the user’s hand geometry and muscle behavior. Touch gesture types (e.g., slide up, slide down) introduce another type of variation other than identity variations. To exclude this variation, the captured traces are pre-filtered into one of these six predefined gestures based on screen regions where the traces start and end. The six gestures (Table I) ensure the generality and usability of the proposed method. In cases of multiple fingertip touch gestures, the Euclidean distances between two fingers at the start and end of traces are also adopted. An increasing distance between two finger tips indicates that a spread gesture is performed, while a decreasing distance indicates that a pinch gesture is performed.

A. Graphic Touch Gesture Feature

GTGF represents touch traces as images. To normalize and register the traces with different numbers of points and time intervals, we use cubic interpolation to resample the traces in terms of their x-y coordinates, time, and pressure series. The cubic interpolation is chosen because it is the simplest method that offers true continuity between the samples. After normalization, a single touchtip trace \( S^0 \) (i.e., UD, DU, LR, RL) is obtained consisting of \( c \) samples. And a multiple touchtip trace \( S^0, S^1 \) (i.e., ZI, ZO) consists of \( 2c \) samples, \( c \) samples in \( \hat{S}^0 \) and \( c \) samples in \( \hat{S}^1 \) respectively.

\[
\hat{s}^0_n = (\hat{x}^0_n, \hat{y}^0_n, \hat{t}^0_n, \hat{p}^0_n), n \in 1, 2, ..., 50, \\
\hat{s}^1_n = (\hat{x}^1_n, \hat{y}^1_n, \hat{t}^1_n, \hat{p}^1_n), n \in 1, 2, ..., 50. \\
\]

We then translate a trace \( \hat{s} \) into a GTGF \( T \). A zero-valued image template \( T \) with resolution set to \( H \times W \) is used. Its size is empirically set to \( 100 \times 150 \) as a tradeoff between the feature “discriminability” and computational efficiency. GTGF is extracted by filling the image template with different intensity values according to an input normalized trace. The normalized traces are projected onto two perpendicular X and Y axes (horizontal and vertical directions according to the touch screen). In general, the upper part of a GTGF image describes the feature along the X direction and the lower part describes the feature along the Y direction. A block in a GTGF image corresponds to a sample \( \hat{s} \) in a trace. Each block spans three columns. The Y expansion in a block is related to the tactile pressure of a sample, while intensity values in a block are related to the moving speed in X, Y directions.

The values in each block are assigned by translating the information of a normalized sample, following Eq. 2, Eq. 3 and Eq. 4.

\[
I^p = \frac{128 \left( \hat{x}_n - \Delta \hat{x}_n \right)}{U_x}, \\
\Delta \hat{x}_n = \hat{x}_n - \hat{x}_{n-1}, \\
I^h = \frac{128 \left( \hat{y}_n - \Delta \hat{y}_n \right)}{U_y}, \\
\Delta \hat{y}_n = \hat{y}_n - \hat{y}_{n-1}, \\
H^p = \left[ \frac{H}{2} \right] \hat{p}_n, \\
\]

where \( |\cdot| \) is the ceiling operator, \( n \) is the index of block. The \( T \) has a fixed number of levels \( \left( \frac{H}{2} \right) \) for the Y expansion (profile) of GTGF. The parameter \( L_p \) in Eq. 4 is used to translate the continuous pressure measure into one of \( H/2 \) levels.

The higher this level \( H^p \) is, the higher the finger pressure applied to the screen. For the same purpose, parameters \( I^p \), \( U_x \), and \( U_y \) are adopted in Eqs. 2 and 3. The value 128 evenly divides the range of pixel levels \([0, 256]\). Levels within the range \([1, 128]\) describe the negative values of \( \Delta \hat{x}_n \) or \( \Delta \hat{y}_n \), representing that the finger moves to the left or bottom directions. Levels within the range \([129, 256]\) describe the positive values of \( \Delta \hat{x}_n \) or \( \Delta \hat{y}_n \), representing that the finger moves to the right or top directions. The closer the pixel levels are to 0 or 256, the faster the fingertip moves along positive or negative directions correspondingly. In this way, the direction, the pressure, and the dynamics of the traces are captured into GTGF, as depicted in Fig. 2. The images within the same row are quite similar to each other, but major differences can be observed between the two rows. The differences are mainly caused by the user identities. For traces \( S^0 \) and \( S^1 \) in a multiple fingertip gesture (i.e., Pinch, Spread), we extract two GTGF \( T^0 \) and \( T^1 \), respectively.

The construction of GTGF has multiple advantages. First, original traces have different spatial topology and temporal duration and thus are difficult to compare directly. GTGF solves the difficulty by registering these traces into a canonical 2D template \( T \). Second, the dynamics are considered to be an important factor in other pattern recognition problems (e.g., facial expression recognition [22] and speech recognition [23]). However, they have not been commonly considered in the touch gesture in the mobile authentication literature. The GTGF is able to represent the gesture dynamics in terms of movement and pressure intuitively and explicitly. Third, due to the inhomogeneity between movement and pressure data, it is difficult to combine their discriminative power at the feature level. However, GTGF takes both features into consideration and combines their discriminative power.

B. Statistical Touch Dynamics Images

STDI learns the intra-class variation modes and synthesizes the new instances using the learned base feature and affine combinations of these learned modes. In the testing phase, the STDI expresses the probe trace in the new basis as...
synthesized instances. The distance between the synthesized
probe instance and the original probe instance is computed
for verification. This distance tends to be greater when the
probe is from different users than those from the same user as
the one on which the STDI trained. Since the STDI is learned
with the specific knowledge of the user, its representation
power decreases for others. Thus, less similarity exists between
the synthesized and original instances extracted from others’
touch traces.

1) STDI Training: The training of an STDI is presented
in Alg. 1. It takes as input a set of touch traces per gesture
type from the user and outputs the trained STDI. This training
process runs repeatedly for all types of touch gestures (i.e., LR,
RL, DU, UD, ZI, ZO) to finish the training process. Thus, six
STDIs are trained for the six gestures, respectively, for each
user.

Algorithm 1 Training Algorithm (Per Gesture Type)
Input: The training set of touch gesture (type \( t \)) traces \( S^t_u \)
from the user \( u \).
Output: The user-specific Statistical Touch Dynamics Images \( D^t_u \).

1: For each trace \( S_i \in S^t_u \):
   1) Extract the GTGF \( T^t_i \) (Sec. III-A),
   2) Reshape the 2D matrix \( T^t_i \) into a vector \( t^t_i \),
2: Learn the base vector \( \hat{v} \) and basis feature vectors \( v_i \) by
   applying Statistical Analysis on \( |t^t_i| \) (Eq. 5).

Since the gesture traces have been registered and scaled
in feature extraction, the GTGF can be directly used for
training without further normalization or alignment. For every
\( T \in \mathbb{R}^{H \times W} \) in the training set, it is first reshaped into
\( t \in \mathbb{R}^{HW} \). The 2D shape information lost due to reshaping
can be easily restored by a reverse reshaping process before
score computation. As in Eq. 5, a vectorized feature \( v \) can be
expressed as a base \( \hat{v} \) plus a linear combination of \( n \) basis
feature vectors \( v_i \). The approach to compute the STDI is to
apply Principal Component Analysis (PCA) to the training
vectors as in statistic feature models [24]:

\[
v = \hat{v}_0 + \sum_{i=1}^{n} b_i v_i = \hat{v}_0 + Vb, \tag{5}
\]

where the base \( \hat{v}_0 \) is the mean of training vectors. The
vectors \( v_i \) are \( n \) eigenvectors corresponding to the \( n \) largest
eigenvalues. The number \( n \) is determined based on the sum
of the \( n \) largest eigenvalues which is greater than a portion
(\( \kappa \)) of the sum of all the eigenvalues. The coefficients \( b_i \)
are the feature parameters different from eigenvalues from PCA.
They control the appearance of learned STDI, as depicted in
Fig. 3. By varying their values, the appearance of STDI can be
changed. We can assume that the vectors \( v_i \) are orthonormal
after PCA and the parameters \( b_i \) are assumed to follow the
Gaussian distributions \( \mathcal{N}(0, \sigma^2_i) \). When \( b_i \) deviates too much
from its mean, singular STDI instances may be generated.
We further express the term \( \sum_{i=1}^{n} b_i v_i \) as \( Vb \).

2) STDI Testing: The testing algorithm is presented in
Alg. 2. It takes a probe trace \( S^t_p \) and a trained model \( D^t_u \)
as input, where the model \( D \) and the trace \( S^t_p \) share the same
gesture type. The output is a score \( d \) representing the distance
between the trace \( S^t_p \) and the user \( u \) represented by the trained
model \( D^t_u \).

Algorithm 2 Testing Algorithm
Input: A probe trace \( S^t_p \) and the trained model \( D^t_u \).
Output: The score \( d \).

1: Extract the GTGF \( T^t_p \) from the probe trace \( S^t_p \),
2: Reshape \( T^t_p \) into \( T^t_p \),
3: Synthesize the instance \( \hat{v}^t_p \) using the parameters \( \tilde{b} \) (Eqs. 6
   - 8),
4: Reshape the vector \( \hat{v}^t_p \) back to \( T^t_p \)
5: Compute the score \( d \) between \( T^t_p \) and its responding
   instance \( T^t_p \) (Eq. 9).

The trained STDI has the capability to synthesize different
instances by varying the parameters \( b \). Given an input \( v_p^t \), for
testing, its responding instances \( \hat{v}^t_p \) can be synthesized from
Eq. 5 using the estimated parameters \( b^t_p \). The estimation is
user’s identity feature. To capture the intra-person variations performed using Eq. 6:
\[ b^e = V^T (v'_p - \bar{v}_0). \] (6)

Instead of directly applying \( b^e \) to Eq. 5, we limit the range of these parameters by applying the function \( f \) to increase the separability between classes. The function \( f \) limits the distribution of the synthesized instances close to the training class in the feature space.

\[ \tilde{b}^e = f(b^e), \]
for \( b_i \in b^e \), \( f(b_i) = \begin{cases} b_i & \text{if } \text{abs}(b_i) \leq \rho \sigma_i, \\ \rho \sigma_i \text{sgn}(b_i) & \text{otherwise}, \end{cases} \] (7)

where \( \sigma_i \) is the expectation of the aforementioned Gaussian distributions of \( b_i \), \( \rho \) is a constant, and \( \text{sgn} \) is the signum function extracting the sign of a real number. Then the responding instance \( \tilde{v'}_p \) can be synthesized as follows:

\[ \tilde{v'}_p = \bar{v}_0 + V \tilde{b}^e. \] (8)

The \( L_1 \) distance is computed between the input instance \( T'^p \) and its corresponding one \( \tilde{T}'_p \) given the STDI \( D'_u \) as the score metrics.

\[ L_1(T'^p, \tilde{T}'_p) = |T'^p - \tilde{T}'_p|. \] (9)

C. User Authentication Algorithms

Directly computing the score metrics between a pair of feature instances from the gallery and probe sets is commonly adopted by an authentication system [8], [25], [26]. However, the phone suffers from computation overhead when the gallery size is large to cover different touch habits from multiple users. The probe trace must compare with each trace in the gallery stored in the phone, which leads to an obvious delay during user interaction when many traces per subject are stored in the gallery. However, reducing the number of traces per subject in the gallery inevitably decreases the generality of the user’s identity feature. To capture the intra-person variations while keeping the method computationally efficient, subject-specific STDIs are learned from the GTGF feature space. The expense of the score computation is then proportional only to the number of subjects rather than the number of features in the gallery.

1) Single User: The user verification algorithm is designed for a single user. In this case, the six touch gestures from the genuine owner are collected. A set of user-specific STDIs are trained for these gestures respectively. Note that the touch traces are automatically classified using the pre-filtering described at the beginning of Sec. III. During the online stage, the type of the probe trace is retrieved first and the corresponding STDI is selected. Then, a distance is computed between the probe trace and the selected STDI and is further compared with the threshold for verification.

The user verification algorithm takes a set of six trained user-specific STDIs \( \{D\} \), the probe trace \( S_p \) and the threshold \( \xi \) as inputs, and outputs a boolean decision \( B \). First it extracts the GTGF \( T'^p \) from \( S_p \) as in Sec. III-A, and selects the STDI \( D' \) from \( D \) that shares the same gesture type as the probe \( S_p \). Then it computes the score \( d \) given \( D' \) and \( T'^p \), as in Alg. 2. Finally, it computes and outputs the \( B \). If \( d < \xi \), the user is genuine; otherwise, the user is an impostor.

2) Multiple Users: The open set user recognition algorithm is designed for the case of multiple users, where user identity management (UIM) is needed. Mobiles with UIM can reject the unauthorized users and change the personalized settings according to the authorized user identity. Touch traces from all authorized users are collected and a set of six STDIs are trained for each user. During the online stage, the type of the probe trace is retrieved first and compared with all STDIs in the gallery from the same type. Distances are computed given the probe trace and the STDIs. Note that the distance computation only needs to be repeated for all subjects, instead of all traces in the gallery. The minimum distance is then selected and compared with the threshold for authentication. Only if it is lower than a threshold is its corresponding identity returned as the result.
Algorithm 3 The User Verification Algorithm

Input: Six user-specific STDIs $D$, the probe trace $S_p$, the threshold $\xi$.

Output: A boolean decision $B$.

1: Extract the GTGF $T_p$ from $S_p$ as in Sec. III-A,
2: Select the STDI $D^i$ from $D$ that shares the same gesture type as the probe $S_p$,
3: Compute the score $d$ given $D^i$ and $T_p$, as in Alg. 2.
4: Compute and output the $B$:
   1) if $d < \xi$, $B = true$; otherwise, $B = false$

Algorithm 4 The User Identification Algorithm

Input: Sets of STDIs $D_u$ for the user group, the probe trace $S_p$, the threshold $\xi$.

Output: The user identity $I$.

1: Extract the GTGF $T_p$ from $S_p$ as in Sec. III-A,
2: For each subject $u$:
   1) Select the STDI $D^i_u$ from $D_u$ which shares the same gesture type as the probe $S_p$,
   2) Compute the score $d_u$ given $D^i_u$ and $T_p$, as in Alg. 2,
3: find the smallest $d_m$ among all $d_u$s,
4: if $d_m \geq \xi_m$,
   then $I = Null$,
   else $I$ is the identity corresponding to the $d_m$.

The user recognition algorithm takes sets of STDIs $D_u$ for the user group, the probe trace $S_p$ and the threshold set $\{\xi\}$ as inputs, and outputs the user identity $I$. First it extracts the GTGF $T_p$ from $S_p$ as in Sec. III-A. Then for each subject $u$ in the gallery, it selects the STDI $D^i_u$ from $D_u$ that shares the same gesture type as the probe $S_p$. Then it computes the score $d_u$ given $D^i_u$ and $T_p$, as in Alg. 2. After finishing the loop, it finds the smallest $d_m$ among all $d_u$s. If $d_m \geq \xi$, the user is not in the user group in the gallery; otherwise, it assigns the user identity as the one corresponding to $d_m$.

IV. EXPERIMENTAL RESULTS AND DISCUSSION

A. Data Acquisition

To systematically evaluate performance of GTGF and STDI, we developed a data collection app which captures touch gestures using a standard API of the Android system. When fingers contact the touchscreen, it records the trace by recording raw touch samples from the API. We recruited 78 subjects, of which 64 were right-handed, and 75 had touchscreen device experience. The data acquisition contains six sessions collected over several months. The first session served as a pilot session where only 30 subjects were recruited. We explained the purpose of the study and the use of our data acquisition program. Then, the subjects practiced for 5 to 10 min. The subjects were instructed to perform the six types of most used touch gestures repetitively during the practice so that they could get used to the data collection app and their gestures could converge to their daily touch habit. After a user could use the program in a natural way, the subject provided 20 traces for each gesture (Tab. I). All 78 subjects participated in all the following sessions from the second to the sixth. The explanation and practice procedures were repeated for the newly recruited subjects in the second session. Sessions occurred with at least three days between two consecutive sessions. Subjects also provided at least 20 traces per gesture during each session. Subjects could vary the manner they held the phone (e.g., left-hand holding vs right-hand holding, whole palm holding vs half palm holding), the hand pose (palm down vs palm up) or the fingers used to perform the single touch gesture (thumb vs index finger) or the finger orientation between sessions. We named the dataset collected in the first session with a smaller number of subjects UH-TOUCH v1 and the dataset collected in the following sessions with all subjects UH-TOUCH v2.

To test the performance and usability of STDI in a real world scenario, we implemented another app named STDI–UA on the Android system to verify users. The app runs as a background service, collecting and verifying the touch traces in the context of Android Launcher. The UH-TOUCH v3 dataset was collected by this app for evaluation. Twenty subjects whose data were acquired in the UH-TOUCH v2 dataset were recruited. Each subject used the device (a Samsung Galaxy S3 phone with Android version 4.1.1) for three days as their daily phone and their touch traces in the context of Android Launcher were recorded. There were 1,327 touch traces collected on average from each user. Table II depicts the statistical information for these datasets.

<table>
<thead>
<tr>
<th></th>
<th>UH-TOUCH v1</th>
<th>UH-TOUCH v2</th>
<th>UH-TOUCH v3</th>
</tr>
</thead>
<tbody>
<tr>
<td>N of S</td>
<td>30</td>
<td>78</td>
<td>20</td>
</tr>
<tr>
<td>Mean</td>
<td>120</td>
<td>609</td>
<td>1,327</td>
</tr>
<tr>
<td>Std</td>
<td>0.0</td>
<td>5.77</td>
<td>81.6</td>
</tr>
</tbody>
</table>

$N$ of $S$ is the number of Subjects in each dataset; $Mean$ is the mean value of number of touch gestures collected per subject; $Std$ is the standard deviation values of number of touch gestures collected per subject.

B. Experimental Results

1) Evaluation on STDI Configuration: Figure 4 depicts the results of grid search on the 3D GTGF parameter space, including the relative movement upper bound $U_x$, $U_y$ in the $x$, $y$ directions and the upper bound for the tactile pressure $L_p$ during extracting GTGFs. They varied in the range from 0.3 to 0.7, from 20 to 60 and from 30 to 70 on the $L_p$, $U_x$, $U_y$ axes, respectively. These ranges were selected to cover most of the meaningful values in the
Under different values of the parameter $\kappa$ are similar for most of the gestures when $\kappa = 0$. The average RRs of the six gestures were 82% for $\kappa = 0$. The performances are similar for most of the gestures when $\kappa = 0.95$ and $\kappa = 0.90$, but a clear decrease in performance can be observed when $\kappa = 0.80$. Thus, we opt to use 0.90 as the value of $\kappa$, as a tradeoff between performance and computational complexity.

We also evaluated the performance of the proposed method under different values of the parameter $\rho$ in STDI. The $\rho$ sets the boundary for coefficients $b$, which limits the variability of synthesizing the new instances. The greater this boundary, the broader the synthesized features can spread in the vector space. The smaller this boundary, the tighter the synthesized features of the class are limited in the space. The experiments were conducted in UH-TOUCH v2 and the setup is the same as the previous experiments. Figure 6 depicts the results for the experiments in terms of EER in Fig. 6(a) and RR in Fig. 6(b). In verification, the average EERs of the six gestures were 10.9%, 9.7%, 10.8% and 13.4% for $\rho = 0.5$, $\rho = 1.0$, $\rho = 3.0$ and $\rho = \infty$. In recognition, the average RRs of the six gestures were 80.7%, 82.3%, 80.9% and 78.2% for $\rho = 0.5$, $\rho = 1.0$, $\rho = 3.0$ and $\rho = \infty$. The best performance for most of the gestures has been achieved in both the recognition and verification experiments when $\rho = 1.0$. Thus, we opted to use 1.0 as the value of $\rho$. A possible explanation is that when the synthesized features spread widely in the vector space, the inter-class distance may be reduced and thus false positive cases increase. When the distribution of the synthesized features is limited to a small region, the intra-
class variations are limited and thus the true negative cases increase, where a balance is achieved at the value $\rho = 1.0$.

2) Demonstration on STDI Advantages: To demonstrate the advantage of converting touch traces into GTGF and further constructing STDI, we conducted a comparison between three methods. The first method is based on the Multivariate Gaussian Distribution Model (MGD) learned directly from the resampled traces where the number of samples on all traces are normalized to a constant (i.e., 50) as Eq. 1 in Sec. III-A. An MGD model is built for each sample, which contains variables of movement and pressure dynamics $\Delta x$, $\Delta y$, and $\rho$. Thus, a set of 50 MGD models were trained for complete traces from a gesture type of a user. These models output a set of probabilities to represent the similarities between the testing trace and the trained model. These probabilities are summed to a single score for authentication. The second GTGF-based method is to extract the GTGF feature and compute the $L_1$ distance between traces from the training set and the testing set [8]. The third method is the proposed STDI-based method. For the experiment setup, we randomly selected half of the traces in each session in the UH-TOUCH v2 and combined to the gallery or training set. The remaining portion from UH-TOUCH v2 was used as the probe or testing set. Figure 7 depicts the comparisons among these three methods in terms of EER in Fig. 7(a) and RR in Fig. 7(b). In the verification experiments, the average EERs for MGD, GTGF and STDI on the six gestures were 36.8%, 12.8% and 9.7%, respectively. The improvement of the GTGF over the MGD is obvious (12.8% vs 36.8%), which indicates the necessity of adopting the GTGF feature. The decrease of EER from 12.8% to 9.7% demonstrates that the traces are better represented and more distinguishable by STDI. In the recognition experiments, the average RRs for MGD, GTGF and STDI on the six gestures were 55.2%, 79.0% and 82.3%, respectively. The increase of RR demonstrates the same evidence as in the verification experiments. By adopting GTGF and STDI, touch traces can be better represented and classified in user identification and verification problems.
To evaluate the performance of the proposed method using different combinations of touch gestures (combining single touchtip gestures, combining multiple touchtip gestures and combining all six gestures) and compare the method with the methods available in the literature, we tested the method on UHTOUCH v2. The same gallery and probe set were used as in the previous experiment. After score computation, we obtained one score matrix for each gesture. Then, we fused the score matrix of multiple gestures using the sum rule, which is proved by Kittler et al. [27] to be superior in comparison to other rules (i.e., product, min, max, median rule). Table III depicts a comparison of the EER and RR for different fusion schemes and methods: (i) GTGF-S: fusion of the score matrices of four single touchtip gestures computed from the aforementioned GTGF method [8]; (ii) GTGF-M: fusion of the score matrices of the two multiple touchtip gestures computed from the GTGF method [8]; (iii) STDI-S: fusion of the score matrices of four single touchtip gestures computed from the proposed method; (iv) STDI-M: fusion of the score matrices of the two multiple touchtip gestures computed from the proposed method; (v) GTGF-A: fusion of the score matrices of the six gestures computed from the aforementioned GTGF method [8]; (vi) STDI-A: fusion of the score matrices of the six gestures computed from the proposed method; (vii) TA-A: fusion of the score matrices of the six gestures computed from the method in [7]; (viii) DM-A: fusion of the score matrices of the six gestures computed from the method in [6]. As in most score-level fusion schemes, combining more scores from different channels increases the overall performance. This trend can be observed from the decrease in EER and increase in RR from the STDI-M (including only two multi-touch gestures) to the STDI-S (including four single-touch gestures) and further to the STDI-A schemes (including all six gestures). Furthermore, the performance of the GTGF-based method demonstrated a clear improvement over the other methods in the scheme of combining all six gestures. There are two reasons for this improvement. First, compared with the method in [6], we adopted tactile pressure in our method, which contains extra clues on the subject’s muscle behavior. However, their method did not include this information. Second, compared to the work in [7] which extracted 30 static analytic feature values, the proposed method includes movement dynamics and pressure dynamics of the touch gesture during feature extraction. Last, the best performance in terms of recognition and verification has been achieved by the STDI-A scheme. Its improvement over GTGF-A is derived from statistical analysis on the intra-class variations in GTGF representation.

### 4) Evaluation of Continuous Mobile Authentication and Baseline Comparisons:
To test the performance and usability of the STDI based user authentication method in a real world scenario, we used the implemented App STDI – UA to verify users. Opset user recognition is excluded from this test since the Android system (version 4.1.x) does not currently support multiple user account management. A gallery of traces (Ns per gesture) for each subject is formed by randomly selecting traces (Ns/5 per gesture) from each session in UHTOUCH v2 to cover variations in touch manner. The traces in UHTOUCH v3 are used as the probe set in the test. During testing, the app ran on a Samsung Galaxy S3 phone with Android version 4.1.1. It has Quad-core 1.4 GHz Cortex-A9 CPU and 1GB RAM. The computational complexity for GTGF is $O(N_s N_p)$, and the computational complexity for STDI is $O(N_s)$, where $N_s$ is the number of subjects in the gallery. Note that one STDI is built for each subject per gesture from statistical analysis on the intra-class variations in GTGF representation.

### TABLE III
A Comparison of EER and RR for Different Fusion Schemes and Methods

<table>
<thead>
<tr>
<th>Fusion Scheme</th>
<th>GTGF-S</th>
<th>GTGF-M</th>
<th>STDI-S</th>
<th>STDI-M</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>EER</strong></td>
<td>6.3%</td>
<td>10.3%</td>
<td>4.7%</td>
<td>9.6%</td>
</tr>
<tr>
<td><strong>RR</strong></td>
<td>84.1%</td>
<td>79.2%</td>
<td>91.5%</td>
<td>80.7%</td>
</tr>
</tbody>
</table>

GTGF refers to the method proposed in [8]; TA refers to the Touchalytics method in [7]; DM refers to the method in [6]; STDI refers to the proposed method.

### TABLE IV
Average Computational Time (ms) and EER (%) of GTGF and STDI Tested With Different Gallery Size

<table>
<thead>
<tr>
<th>Ns</th>
<th>30</th>
<th>50</th>
<th>80</th>
<th>100</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GTGF</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EER</td>
<td>32.2</td>
<td>46.3</td>
<td>67.3</td>
<td>81.4</td>
</tr>
<tr>
<td><strong>STDI</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EER</td>
<td>12.8</td>
<td>12.9</td>
<td>12.9</td>
<td>12.9</td>
</tr>
</tbody>
</table>

Fig. 8. A comparison of the FRR and FAR results for the continuous authentication among the GTGF, the STDI methods and the Touchalytics method [7]. The Y axis is the percentage of the FAR and FRR; the X axis is the number of consecutive traces ($\gamma$) used in continuous authentication. The solid lines represent the FRR results and the dash lines represent the FAR results.

AVERAGE COMPUTATIONAL TIME (ms) AND EER (%) OF GTGF AND STDI TESTED WITH DIFFERENT GALLERY SIZE

<table>
<thead>
<tr>
<th>Ns</th>
<th>30</th>
<th>50</th>
<th>80</th>
<th>100</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GTGF</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EER</td>
<td>6.9</td>
<td>6.1</td>
<td>5.4</td>
<td>5.2</td>
</tr>
<tr>
<td><strong>STDI</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EER</td>
<td>6.2</td>
<td>5.3</td>
<td>4.9</td>
<td>4.8</td>
</tr>
</tbody>
</table>

Fig. 8. A comparison of the FRR and FAR results for the continuous authentication among the GTGF, the STDI methods and the Touchalytics method [7]. The Y axis is the percentage of the FAR and FRR; the X axis is the number of consecutive traces ($\gamma$) used in continuous authentication. The solid lines represent the FRR results and the dash lines represent the FAR results.
from 32.22 ms to 81.36 ms. However, the computational time for STDI remains at around 12.90 ms. The third and fourth rows in Table IV depict the average EER in user authentication using five continuous traces. These EERs were achieved by GTGF and STDI methods with different gallery size respectively. As $N_{c}$ increased from 30 traces per gesture to 100 traces, the EERs achieved by GTGF and STDI decreased. When using more than 80 traces, the EER improvement for both methods has been reduced.

Figure 8 depicts a comparison of the continuous authentication results among GTGF, TA [7] and STDI in terms of the average FRR (solid line) and average FAR (dash line). The distance from the probe trace and the distances from the following $\gamma - 1$ consecutive probe traces are averaged to obtain the score for authentication. If it is above the threshold, the app classifies the user as an unauthorized intruder and invokes explicit user authentication. FARs for all methods reduced in a small but steady rate as the $\gamma$ increased. Meanwhile, the FRRs of all methods decreased significantly as the number of traces increased. For the GTGF and STDI methods, the decreasing rate reduced after the $\gamma > 5$ and almost converged when $\gamma = 7$. When $\gamma > 5$, the delay caused by the increased computational burden became noticeable for GTGF. However, no obvious delay has been observed for the STDI method, even when $\gamma = 7$.

V. CONCLUSION

This paper presented a novel mobile authentication approach based on touch gestures. An STDI, which synthesizes trace feature instances using intra-person variations, has been proposed. Using a multi-session touch gesture dataset containing six commonly used gestures, we have systematically evaluated the performance of the proposed method. Comparisons between the STDI and other methods in the literature have also been conducted. An STDI-based User Verification App has been implemented and tested on a dataset collected during the daily use of the phone. The results have demonstrated the effectiveness and usability of the STDI in the real-world mobile authentication.

REFERENCES

Xi Zhao received the Ph.D. (Hons.) degree in computer science from the Ecole Centrale de Lyon, Ecully, France, in 2010. After graduation, he conducted research in the fields of biometrics, face analysis, and pattern recognition, as a Post-Doctoral Fellow with the Department of Computer Science, University of Houston, Houston, TX, USA. He is currently a Research Assistant Professor with the Department of Computer Science, University of Houston. His current research interests include biometrics, mobile computing, computer vision, and assistive technology.

Dr. Zhao currently serves as a reviewer of the IEEE TRANSACTIONS ON IMAGE PROCESSING, the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS - PART B, the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, Image Vision Computing, the IEEE International Conference on Automatic Face and Gesture Recognition, the IEEE International Conference on Advanced Video and Signal-Based Surveillance, the International Conference on Pattern Recognition, and the International Conference on Pattern Recognition. He has also served on the Program Committee of Workshop on 3-D Face Biometrics, in 2013, and the Registration Chair of the International Conference on Biometrics: Theory, Applications and Systems, in 2013.

Weidong Shi received the Ph.D. degree in computer science from the Georgia Institute of Technology, Atlanta, GA, USA, where he was involved in research on computer architecture and computer systems. He was a Senior Research Staff Engineer with the Motorola Research Laboratory, Nokia Research Center, and the Co-Founder of a technology startup. He is currently an Assistant Professor with the University of Houston, Houston, TX, USA. He was involved in designing multiple Nvidia platform products, and was credited to publish Electronic Art’s console game. He has authored or coauthored publications covering research problems in computer architecture, computer systems, multimedia/graphics systems, mobile computing, and computer security. He holds multiple issued and pending U.S. and Trademark Office patents.

Ioannis A. Kakadiaris is currently a Hugh Roy and Lillie Cranz Cullen University Professor of Computer Science, Electrical and Computer Engineering, and Biomedical Engineering with the University of Houston (UH), Houston, TX, USA. He joined UH, in 1997, after a post-doctoral fellowship with the University of Pennsylvania, Philadelphia, PA, USA. He received the B.Sc. degree in physics from the University of Athens, Athens, Greece, the M.Sc. degree in computer science from Northeastern University, Boston, MA, USA, and the Ph.D. degree from the University of Pennsylvania. He is the Founder of the Computational Biomedicine Laboratory at UH. His research interests include multimodal biometrics, face recognition, computer vision, pattern recognition, biomedical image analysis, and predictive analytics. He was a recipient of a number of awards, including the NSF Early CAREER Development Award, the Schlumberger Technical Foundation Award, the UH Computer Science Research Excellence Award, the UH Enron Teaching Excellence Award, and the James Muller Vulnerable Plaque Young Investigator Prize. His research has been featured on the Discovery Channel, National Public Radio, KPRC NBC News, KTRH ABC News, and KHOU CBS News. His selected professional service leadership positions include: the General Cochair of the 2013 Biometrics: Theory, Applications and Systems Conference, General Cochair of the 2014 SPIE Biometric and Surveillance Technology for Human and Activity Identification, Program Cochair of the 2015 International Conference on Automatic Face and Gesture Recognition Conference, and Vice President for Technical Activities of the IEEE Biometrics Council.

Tao Feng received the B.S. degree in software engineering from the Huazhong University of Science and Technology, Wuhan, China, and the M.Sc. degree in computer science from the Chinese Academy of Sciences, Institute of Automation, Beijing, China, with a focus on knowledge system and knowledge visualization.

He is currently pursuing the Ph.D. degree with the Department of Computer Science, University of Houston, Houston, TX, USA. His current research interests include mobile computing, artificial intelligence, human–computer interaction, mobile security, and machine learning.
AUTHOR QUERY

AQ:1 = Algorithms 3 and 4 are not cited in body text. Please indicate where they should be cited.
Behavioral biometrics have recently begun to gain attention for mobile user authentication. The feasibility of touch gestures as a novel modality for behavioral biometrics has been investigated. In this paper, we propose applying a statistical touch dynamics image (aka statistical feature model) trained from graphic touch gesture features to retain discriminative power for user authentication while significantly reducing computational time during online authentication. Systematic evaluation and comparisons with state-of-the-art methods have been performed on touch gesture data sets. Implemented as an Android App, the usability and effectiveness of the proposed method have also been evaluated.

Index Terms—Touch gesture, user authentication, mobile security, statistical touch dynamics images, behavioral biometrics.

I. INTRODUCTION

The increasing ubiquity of smartphones raises the issue of security of paramount importance. The data stored and accessed from such devices are rarely protected except by a point-based authentication at the login stage. This is deficient since it cannot detect intruders after a user has passed the point entry and is vulnerable to simple attack (e.g., smudge attacks [1]) after loss or theft. As opposed to password or image-pattern-based authentication, continuous authentication has received increased attention in recent years. It intelligently monitors and analyzes the user-device interaction to ensure correct identity, which either complements point-based authentication or even substitutes it when continuous authentication satisfies particular accuracy requirements.

While using a smartphone, user identity can be sensed via multiple integrated sensors (e.g., the touch screen, motion sensors, microphone, or camera). Users’ faces [2] and voices [3] can be used for authentication, but it is not convenient or optimal to apply these approaches while the user is using most of the apps (e.g., reading news and email, surfing the internet). Users must face the camera or keep talking to be continuously authenticated, which greatly reduces the device usability. Authentication using motion data (i.e., Accelerometer/Gyroscope data) has been studied. However, due to signal noise and difficulty in decoupling identity from activity, this modality currently can only be applied on motion-specific activities (e.g., phone pick-up motion during answering the call [4]). Conversely, the touch screen is quite suitable for authentication, not only because it is the most frequently used sensor but also the touch gesture contains identity-rich user behavioral traits. The feasibility of using touch gestures as a biometric modality has been investigated [5]–[8]. It can be observed in Fig. 1 that most of the touch traces from one user tend to follow a similar pattern while the patterns vary among different users. Intrinsically, touch traces are affected by two biometric features (i.e., user hand geometry and finger muscle behavior).

Fig. 1. Plots (a) depicts over 300 touch traces of zoom-in gesture from 30 users. Each subfigure contains around 10 traces from a single user. Plots (b) and (c) are the enlarged figures depicting the first and second users’ touch gestures.
In our previous work, we evaluated the Graphic Touch Gesture Feature (GTGF) for touch based user authentication [8]. The GTGF can represent touch dynamics in an explicit manner where the discriminative power from touch trajectories and tactile pressure is combined. However, implemented as an Android App, this method suffers from computational overhead and incurs a noticeable run-time delay when the gallery size increases because a probe trace must be compared with each trace in the gallery. To improve the effectiveness of GTGF and reduce the computational expense at the same time, we propose to apply a Statistical Feature Model (SFM) as Touch Dynamics Images (STDI), which builds a touch gesture “appearance” model from the GTGF. STDI learns the intra-person variations of the GTGF in the image domain by statistical analysis and is capable of synthesizing new instances according to a new probe trace. Instead of directly computing the distance between the probe trace and the gallery traces, the distance is computed between the probe trace and its corresponding synthesized instance given a user-specific STDI. The synthesized instance from the STDI mirrors the probe trace. The instance tends to be more dissimilar to the trace if the trace originates from an identity other than the genuine user. We evaluated the proposed method on three sets of touch gestures (flick up/down, flick right/left, zoom in/out) simply because they are among the most used gestures during user-device interaction [9]. Flick up/down is generally used when reading text or scrolling menus. Flick right/left is generally used when browsing photos, translating screens, and unlocking phones, while zoom in/out is generally used whenever users switch between details and overviews of the contents.

Our main contributions are: (i) applying the feature representation principle in SFM to touch gestures by STDI, which retains the effectiveness of previous GTGF and is more efficient in terms of computation; (ii) developing the algorithms to apply the STDI on the mobile user verification and recognition problems; (iii) systematically evaluating the proposed algorithm by varying different parameters, and providing a comparison between the proposed algorithm with the state-of-the-art methods; and (iv) developing an Android app and testing performance of the verification algorithm in a real-world scenario.

The rest of the paper is organized as follows. First, we discuss the related work in Section II. Methods are presented in Section III. GTGF extraction is presented in Section III-A. The STDI is described in Section III-B. The user authentication algorithms are presented in Section III-C. Section IV presents the experimental results. In Section V, we conclude our study.

II. RELATED WORKS

Mobile User Authentication can be categorized into two groups: the physiological biometrics, which rely on static physical attributes (e.g., fingerprints [10], facial features [11]) and behavioral biometrics, which adopt identity-invariant features of human behavior to identify or verify people during their daily activities [12].

Prior research on behavioral biometrics has shown that biometric traits can be extracted from physiological characteristics of arm gestures [4], [13], strokes and signatures using a pen or stylus [14], [15]. Yamazaki et al. [16] extracted personal features from stroke shape, writing pressure and pen inclination to identify an individual. Cao et al. [17] discussed a quantitative human performance model of making single-stroke pen gestures. Ferrer et al. [18] used local patterns (e.g., the Local Binary Pattern and the Local Directional Pattern), to segment and extract identity features from signatures. Impedovo et al. [19] presented the state-of-the-art in automatic signature verification. However, verifying users with pen strokes cannot be pervasively applied in mobile user authentication since only a few types of mobiles provide a pen or stylus. Furthermore, most users opt to interact with their phone directly using their fingers for convenience.

Studies exploring the touch gesture as a biometric modality for mobile devices are quite recent [20]. Feng et al. [5] extracted finger motion speed and acceleration of touch gestures as features for identification. Luca et al. [21] directly computed the distance between traces using the dynamic time warping algorithm. Sae-Bae et al. [6] designed 22 touch gestures for authentication, involving different combinations of five fingers. They computed the dynamic time warping distance and Frechet distance between multi-touch traces. However, users must perform their pre-defined touch gestures for authentication, which thus cannot be transparent and non-obtrusive to users. Frank et al. [7] studied the correlation between 30 analytic features from touch traces and classified these features using K-nearest-neighbor and Support Vector Machine approaches. However, the analytic features did not capture the touch trace dynamics. The proposed method is quite different from existing works. The pressure dynamics are adopted in our method and are represented as the profiles of the STDI instances. This is different from extracting a single mid-stroke pressure value [7] or simply being ignored as in other studies [6], [21]. Rather than directly using GTGF for authentication [8], the STDI applies the similar feature representing principle of the statistical feature model in [24] to learn the intra-person variations from many traces by a user so that scores are computed between pairs of gallery models and probe traces, instead of using pairs of gallery traces and probing traces, which significantly reduces the computational time.

III. METHODS

The authentication scenarios considered in this paper include open set user recognition and user verification. In the open set user recognition scenario, mobile devices are shared among a few users (e.g., family members). The sensitive group data should be shared only among the authorized users, and users’ identities should first be verified to ensure they belong to the authorized group. Meanwhile, the user should be further recognized to customize the user experience on the device (e.g., limiting the functions for a child, granting different access levels). In a more general scenario, phones are used only by their owner. Other individuals are considered intruders and should be excluded from using the phone. Thus, the method verifies whether the device is being used by its owner.
To perform user authentication, we capture the touch traces from the touch screen output and segment the traces with support from the Android API. We use a series of x-y coordinates of finger touch points, pressure values and time stamps in generating GTGF feature representation. Each series varies in time duration, trace length, dynamics of finger movements and tactile pressure, which is dependent on the user’s hand geometry and muscle behavior. Touch gesture types (e.g., slide up, slide down) introduce another type of variation other than identity variations. To exclude this variation, the captured traces are pre-filtered into one of these six predefined gestures based on screen regions where the traces start and end. The six gestures (Table I) ensure the generality and usability of the proposed method. In cases of multiple fingertip touch gestures, the Euclidean distances between two fingers at the start and end of traces are also adopted. An increasing distance between two finger tips indicates that a spread gesture is performed, while a decreasing distance indicates that a pinch gesture is performed.

A. Graphic Touch Gesture Feature

GTGF represents touch traces as images. To normalize and register the traces with different numbers of points and time intervals, we use cubic interpolation to resample the traces in terms of their x-y coordinates, time, and pressure series. The cubic interpolation is chosen because it is the simplest method that ensures true continuity between the samples. After normalization, a single touchtip trace $S^0$ (i.e., UD, DU, LR, RL) is obtained consisting of $c$ samples. And a multiple touchtip trace $S^0$, $S^1$ (i.e., ZI, ZO) consists of $2c$ samples, $c$ samples in $S^0$ and $c$ samples in $S^1$ respectively.

$$\hat{s}^0_n = (\hat{x}_n^0, \hat{y}_n^0, \hat{r}_n^0, \hat{p}_n^0), n = 1, 2, \ldots, 50,$$

$$\hat{s}^1_n = (\hat{x}_n^1, \hat{y}_n^1, \hat{r}_n^1, \hat{p}_n^1), n = 1, 2, \ldots, 50.$$  (1)

We then translate a trace $\hat{S}$ into a GTGF $T$. A zero-valued image template $\mathcal{T}$ with resolution set to $H \times W$ is used. Its size is empirically set to $100 \times 150$ as a tradeoff between the feature “discriminability” and computational efficiency. GTGF is extracted by filling the image template with different intensity values according to an input normalized trace. The normalized traces are projected onto two perpendicular X and Y axes (horizontal and vertical directions according to the touch screen). In general, the upper part of a GTGF image describes the feature along the X direction and the lower part describes the feature along the Y direction. A block in a GTGF image corresponds to a sample $\hat{s}_k$ in a trace. Each block spans three columns. The Y expansion in a block is related to the tactile pressure of a sample, while intensity values in a block are related to the moving speed in X, Y directions. The values in each block are assigned by translating the information of a normalized sample, following Eq. 2, Eq. 3 and Eq. 4.

$$I^p_x = [128 * \frac{U_x - \Delta \hat{x}_n}{U_x}],$$

$$\Delta \hat{x}_n = \hat{x}_n - \hat{x}_{n-1},$$

$$I^p_y = [128 * \frac{U_y - \Delta \hat{y}_n}{U_y}],$$

$$\Delta \hat{y}_n = \hat{y}_n - \hat{y}_{n-1},$$

$$H^p = \left[\frac{H}{2} \ast \frac{\hat{p}_n}{L_p}\right].$$  (4)

where $\lceil \cdot \rceil$ is the ceiling operator, $n$ is the index of block. The $\mathcal{T}$ has a fixed number of levels ($\frac{H}{2}$) for the Y expansion (profile) of GTGF. The parameter $L_p$ in Eq. 4 is used to translate the continuous pressure measure into one of $H/2$ levels. The higher this level $H^p$ is, the higher the pressure applied to the screen. For the same purpose, parameters $I_m$, $U_i$, and $U_f$ are adopted in Eqs. 2 and 3. The value 128 evenly divides the range of pixel levels ([0, 256]). Levels within the range [1, 128] describe the negative values of $\Delta \hat{x}_n$ or $\Delta \hat{y}_n$, representing that the finger moves to the left or bottom directions. Levels within the range [129, 256] describe the positive values of $\Delta \hat{x}_n$ or $\Delta \hat{y}_n$, representing that the finger moves to the right or top directions. The closer the pixel levels are to 0 or 256, the faster the fingertip moves along positive or negative directions correspondingly. In this way, the direction, the pressure, and the dynamics of the traces are captured into GTGF, as depicted in Fig. 2. The images within the same row are quite similar to each other, but major differences can be observed between the two rows. The differences are mainly caused by the user identities. For traces $S^0$ and $S^1$ in a multiple fingertip gesture (i.e., Pinch, Spread), we extract two GTGF $T^0$ and $T^1$, respectively.

The construction of GTGF has multiple advantages. First, original traces have different spatial topology and temporal duration and thus are difficult to compare directly. GTGF solves the difficulty by registering these traces into a canonical 2D template $\mathcal{T}$. Second, the dynamics are considered to be an important factor in other pattern recognition problems (e.g., facial expression recognition [22] and speech recognition [23]). However, they have not been commonly considered in the touch gesture in the mobile authentication literature. The GTGF is able to represent the gesture dynamics in terms of movement and pressure intuitively and explicitly. Third, due to the inhomogeneity between movement and pressure data, it is difficult to combine their discriminative power at the feature level. However, GTGF takes both features into consideration and combines their discriminative power.

B. Statistical Touch Dynamics Images

STDI learns the intra-class variation modes and synthesizes the new instances using the learned base feature and affine combinations of these learned modes. In the testing phase, the STDI expresses the probe trace in the new basis as
synthesized instances. The distance between the synthesized probe instance and the original probe instance is computed for verification. This distance tends to be greater when the probe is from different users than those from the same user as the one on which the STDI trained. Since the STDI is learned with the specific knowledge of the user, its representation power decreases for others. Thus, less similarity exists between the synthesized and original instances extracted from others’ traces.

1) STDI Training: The training of an STDI is presented in Alg. 1. It takes as input a set of touch traces per gesture type from the user and outputs the trained STDI. This training process runs repeatedly for all types of touch gestures (i.e., LR, RL, DU, UD, ZI, ZO) to finish the training process. Thus, six STDIs are trained for the six gestures, respectively, for each user.

Algorithm 1 Training Algorithm (Per Gesture Type)

Input: The training set of touch gesture (type \( t \)) traces \( S^T_u \) from the user \( u \).

Output: The user-specific Statistical Touch Dynamics Images \( D^T_u \).

1: For each trace \( S_i \in S^T_u \):
   1) Extract the GTGF \( T_i \) (Sec. III-A),
   2) Reshape the 2D matrix \( T_i \) into a vector \( t_i \),
2: Learn the base \( \bar{v} \) and basis feature vectors \( v_i \) by applying Statistical Analysis on \( \{t_i\} \) (Eq. 5).

Since the gesture traces have been registered and scaled in feature extraction, the GTGF can be directly used for training without further normalization or alignment. For every \( T \in \mathbb{R}^{H \times W} \) in the training set, it is first reshaped into \( t \in \mathbb{R}^{HW} \). The 2D shape information lost due to reshaping can be easily restored by a reverse reshaping process before score computation. As in Eq. 5, a vectorized feature \( v \) can be expressed as a base \( \bar{v} \) plus a linear combination of \( n \) basis feature vectors \( v_i \). The approach to compute the STDI is to apply Principal Component Analysis (PCA) to the training vectors as in statistic feature models [24]:

\[
v = \bar{v}_0 + \sum_{i=1}^{n} b_i v_i = \bar{v}_0 + Vb, \tag{5}\]

where the base \( \bar{v}_0 \) is the mean of training vectors. The vectors \( v_i \) are \( n \) eigenvectors corresponding to the \( n \) largest eigenvalues. The number \( n \) is determined based on the sum of the \( n \) largest eigenvalues which is greater than a portion (\( \kappa \)) of the sum of all the eigenvalues. The coefficients \( b_i \) are the feature parameters different from eigenvalues from PCA. They control the appearance of learned STDI, as depicted in Fig. 3. By varying their values, the appearance of STDI can be changed. We can assume that the vectors \( v_i \) are orthonormal after PCA and the parameters \( b_i \) are assumed to follow the Gaussian distributions \( \mathcal{N}(0, \sigma_i^2) \). When \( b_i \) deviates too much from its mean, singular STDI instances may be generated. We further express the term \( \sum_{i=1}^{n} b_i v_i \) as \( Vb \).

2) STDI Testing: The testing algorithm is presented in Alg. 2. It takes a probe trace \( S_p \) and a trained model \( D^T_u \) as input, where the model \( D \) and the trace \( S_p \) share the same gesture type. The output is a score \( d \) representing the distance between the trace \( S_p \) and the user \( u \) represented by the trained model \( D^T_u \).

Algorithm 2 Testing Algorithm

Input: A probe trace \( S^T_p \) and the trained model \( D^T_u \).

Output: The score \( d \).

1: Extract the GTGF \( T^T_p \) from the probe trace \( S^T_p \),
2: Reshape \( T^T_p \) into \( t^T_p \),
3: Synthesize the instance \( \hat{v}^T_p \) using the parameters \( \hat{b}^T \) (Eqs. 6 - 8),
4: Reshape the vector \( \hat{v}^T_p \) back to \( \hat{T}^T_p \)
5: Compute the score \( d \) between \( T^T_p \) and its responding instance \( T^T_p \) (Eq. 9).

The trained STDI has the capability to synthesize different instances by varying the parameters \( b \). Given an input \( v_p \) for testing, its responding instances \( v_p \) can be synthesized from Eq. 5 using the estimated parameters \( b_i \). The estimation is
performed using Eq. 6:
\[ b^e = V^T(v'_p - \tilde{v}_0). \] (6)

Instead of directly applying \( b^e \) to Eq. 5, we limit the range of these parameters by applying the function \( f \) to increase the separability between classes. The function \( f \) limits the distribution of the synthesized instances close to the training class in the feature space.

\[ \tilde{b}^e = f(b^e), \]
for \( b_i \in b^e, \quad f(b_i) = \begin{cases} b_i & \text{if } \text{abs}(b_i) \leq \rho \sigma_i, \\ \rho \sigma_i \text{sgn}(b_i) & \text{otherwise,} \end{cases} \] (7)

where \( \sigma_i \) is the expectation of the aforementioned Gaussian distribution of \( b_i \), \( \rho \) is a constant, and \( \text{sgn} \) is the signum function extracting the sign of a real number. Then the responding instance \( \tilde{v}'_p \) can be synthesized as follows:

\[ \tilde{v}'_p = \tilde{v}_0 + V\tilde{b}^e. \] (8)

The \( L_1 \) distance is computed between the input instance \( T^p \) and its corresponding one \( \tilde{T}^l \) given the STDI \( D^l \) as the score metrics.

\[ L_1(\tilde{T}^l_p, T^l_p) = |T^l_p - \tilde{T}^l_p|. \] (9)

C. User Authentication Algorithms

Directly computing the score metrics between a pair of feature instances from the gallery and probe sets is commonly adopted by an authentication system [8], [25], [26]. However, the phone suffers from computation overhead when the gallery size is large to cover different touch habits from multiple users. The probe trace must compare with each trace in the gallery stored in the phone, which leads to an obvious delay during user interaction when many traces per subject are stored in the gallery. However, reducing the number of traces per subject in the gallery inevitably decreases the generality of the user’s identity feature. To capture the intra-person variations while keeping the method computationally efficient, subject-specific STDIs are learned from the GTGF feature space. The expense of the score computation is then proportional only to the number of subjects rather than the number of features in the gallery.

1) Single User: The user verification algorithm is designed for a single user. In this case, the six touch gestures from the genuine owner are collected. A set of user-specific STDIs are trained for these gestures respectively. Note that the touch traces are automatically classified using the pre-filtering described at the beginning of Sec. III. During the online stage, the type of the probe trace is retrieved first and the corresponding STDI is selected. Then, a distance is computed between the probe trace and the selected STDI and is further compared with the threshold for verification.

The user verification algorithm takes a set of six trained user-specific STDIs \( D \), the probe trace \( S_p \) and the threshold \( \xi \) as inputs, and outputs a boolean decision \( B \). First it extracts the GTGF \( T_p^l \) from \( S_p \) as in Sec. III-A, and selects the STDI \( D^l \) from \( D \) that shares the same gesture type as the probe \( S_p \). Then it computes the score \( d \) given \( D^l \) and \( T_p^l \), as in Alg. 2. Finally, it computes and outputs the \( B \). If \( d < \xi \), the user is genuine; otherwise, the user is an impostor.

2) Multiple Users: The open set user recognition algorithm is designed for the case of multiple users, where user identity management (UIM) is needed. Mobiles with UIM can reject the unauthorized users and change the personalized settings according to the authorized user identity. Touch traces from all authorized users are collected and a set of six STDIs are trained for each user. During the online stage, the type of the probe trace is retrieved first and compared with all STDIs in the gallery from the same type. Distances are computed given the probe trace and the STDIs. Note that the distance computation only needs to be repeated for all subjects, instead of all traces in the gallery. The minimum distance is then selected and compared with the threshold for authentication. Only if it is lower than a threshold is its corresponding identity returned as the result.
Algorithm 3 The User Verification Algorithm

Input: Six user-specific STDIs \( D_u \), the probe trace \( S_p \), the threshold \( \xi \).
Output: A boolean decision \( B \).

1: Extract the GTGF \( T_p \) from \( S_p \) as in Sec. III-A,
2: Select the STDI \( D'_u \) from \( D_u \) that shares the same gesture type as the probe \( S_p \),
3: Compute the score \( d \) given \( D'_u \) and \( T_p \), as in Alg. 2,
4: Compute and output the \( B \):
   1) if \( d < \xi \), \( B = true \); otherwise, \( B = false \)

Algorithm 4 The User Identification Algorithm

Input: Sets of STDI \( D_u \) for the user group, the probe trace \( S_p \), the threshold \( \xi \).
Output: The user identity \( \mathcal{I} \).

1: Extract the GTGF \( T_p \) from \( S_p \) as in Sec. III-A,
2: For each subject \( u \):
   1) Select the STDI \( D'_u \) from \( D_u \) which shares the same gesture type as the probe \( S_p \),
   2) Compute the score \( d_u \) given \( D'_u \) and \( T_p \), as in Alg. 2,
3: find the smallest \( d_m \) among all \( d_u \)s,
4: if \( d_m \geq \xi \), then \( \mathcal{I} = u \),
   else \( \mathcal{I} = \text{Null} \).

The user recognition algorithm takes sets of STDIs \( \{D_u\} \) for the user group, the probe trace \( S_p \) and the threshold set \( \{\xi\} \) as inputs, and outputs the user identity \( \mathcal{I} \). It first extracts the GTGF \( T_p \) from \( S_p \) as in Sec. III-A. Then for each subject \( u \) in the gallery, it selects the STDI \( D'_u \) from \( \{D_u\} \) that shares the same gesture type as the probe \( S_p \). Then it computes the score \( d_u \) given \( D'_u \) and \( T_p \), as in Alg. 2. After finishing the loop, it finds the smallest \( d_m \) among all \( d_u \)s. If \( d_m \geq \xi \), the user is not in the user group in the gallery; otherwise, it assigns the user identity as the one corresponding to \( d_m \).

IV. EXPERIMENTAL RESULTS AND DISCUSSION

A. Data Acquisition

To systematically evaluate performance of GTGF and STDI, we developed a data collection app which captures touch gestures using a standard API of the Android system. When fingers contact the touchscreen, it records the trace by recording raw touch samples from the API. We recruited 78 subjects, of which 64 were right-handed, and 75 had touchscreen device experience. The data acquisition contains six sessions collected over several months. The first session served as a pilot session where only 30 subjects were recruited. We explained the purpose of the study and the use of our data acquisition program. Then, the subjects practiced for 5 to 10 min. The subjects were instructed to perform the six types of most used touch gestures repetitively during the practice time so that they could get used to the data collection app and their gestures could converge to their daily touch habit. After a user could use the program in a natural way, the subject provided 20 traces for each gesture (Tab. I). All 78 subjects participated in all the following sessions from the second to the sixth. The explanation and practice procedures were repeated for the newly recruited subjects in the second session. Sessions occurred with at least three days between two consecutive sessions. Subjects also provided at least 20 traces per gesture during each session. Subjects could vary the manner they held the phone (e.g., left-hand holding vs right-hand holding, whole palm holding vs half palm holding), the hand pose (palm down vs palm up) or the fingers used to perform the single touch gesture (thumb vs index finger) or the finger orientation between sessions. We named the dataset collected in the first session with a smaller number of subjects UH-TOUCH v1 and the dataset collected in the following sessions with all subjects UH-TOUCH v2.

To test the performance and usability of STDI in a real world scenario, we implemented another app named STDI–UA on the Android system to verify users. The app runs as a background service, collecting and verifying the touch traces in the context of Android Launcher. The UH-TOUCH v3 dataset was collected by this app for evaluation. Twenty subjects whose data were acquired in the UH-TOUCH v2 dataset were recruited. Each subject used the device (a Samsung Galaxy S3 phone with Android version 4.1.1) for three days as their daily phone and their touch traces in the context of Android Launcher were recorded. There were 1,327 touch traces collected on average from each user. Table II depicts the statistical information for these datasets.

The Recognition Rate (RR) and Equal Error Rate (EER) were used as the evaluation criteria for the recognition and verification experiments. RR is defined as the number of times that users have been correctly identified over the size of the probe set. EER is the common value when the false acceptance rate (FAR) and the false rejection rate (FRR) are equal.

B. Experimental Results

1) Evaluation on STDI Configuration: Figure 4 depicts the results of grid search on the 3D GTGF parameter space, including the relative movement upper bound \( U_x \), \( U_y \) in the \( x, y \) directions and the upper bound for the tactile pressure \( L_p \) during extracting GTGFs. They varied in the range from 0.3 to 0.7, from 20 to 60 and from 30 to 70 on the \( L_p, U_x, U_y \) axes, respectively. These ranges were selected to cover most of the meaningful values in the
under different values of the parameter $\rho$ of $\kappa$ when $\kappa = 1$ for the average RRs of the six gestures were 82% for the average EERs of the six gestures are 9%. EER in Fig. 5(a) and RR in Fig. 5(b). In verification, the average EERs of the six gestures are 91% and the best EER has been achieved for the values 0.35, 20, 30 in the parameter space. This demonstrated that the proposed GTGF feature is not sensitive to the parameter variations in the aforementioned tested range. The parameters at which the best EER was achieved were used in the experiments described below.

We evaluated the performance of the proposed method under different values of the parameter $\kappa$ in STDI. The $\kappa$ determines the number of the eigenvectors to preserve in STDI. When the $\kappa$ is larger, more eigenvectors are preserved so that new instances can be synthesized in a higher dimensional vector space. This allows more variations on the instance synthesis, but meanwhile increases the computation complexity. The experiments were conducted in UH-TOUCH v2 and the setup was the same as the previous experiments. Figure 5 depicts the results for the experiments in terms of EER in Fig. 5(a) and RR in Fig. 5(b). In verification, the average EERs of the six gestures are 9.6%, 9.7% and 12.3% for $\kappa = 0.95$, $\kappa = 0.90$ and $\kappa = 0.80$. In recognition, the average RRs of the six gestures were 82.6%, 82.3% and 78.1% for $\kappa = 0.95$, $\kappa = 0.90$ and $\kappa = 0.80$. The performances are similar for most of the gestures when $\kappa = 0.95$ and $\kappa = 0.90$, but a clear decrease in performance can be observed when $\kappa = 0.80$. Thus, we opt to use 0.90 as the value of $\kappa$, as a tradeoff between performance and computational complexity.

We also evaluated the performance of the proposed method under different values of the parameter $\rho$ in STDI. The $\rho$ sets the boundary for coefficients $b$, which limits the variability of synthesizing the new instances. The greater this boundary, the broader the synthesized features can spread in the vector space. The smaller this boundary, the tighter the synthesized features of the class are limited in the space. The experiments are conducted in UH-TOUCH v2 and the setup is the same as the previous experiments. Figure 6 depicts the results for the experiments in terms of EER in Fig. 6(a) and RR in Fig. 6(b). In verification, the average EERs of the six gestures were 10.9%, 9.7%, 10.8% and 13.4% for $\rho = 0.5$, $\rho = 1.0$, $\rho = 3.0$ and $\rho = \infty$. In recognition, the average RRs of the six gestures were 80.7%, 82.3%, 80.9% and 78.2% for $\rho = 0.5$, $\rho = 1.0$, $\rho = 3.0$ and $\rho = \infty$. The best performance for most of the gestures has been achieved in both the recognition and verification experiments when $\rho = 1.0$. Thus, we opted to use 1.0 as the value of $\rho$. A possible explanation is that when the synthesized features spread widely in the vector space, the inter-class distance may be reduced and thus false positive cases increase. When the distribution of the synthesized features is limited to a small region, the intra-
class variations are limited and thus the true negative cases increase, where a balance is achieved at the value $\rho = 1.0$.

2) Demonstration on STDI Advantages: To demonstrate the advantage of converting touch traces into GTGF and further constructing STDI, we conducted a comparison between three methods. The first method is based on the Multivariate Gaussian Distribution Model (MGD) learned directly from the resampled traces where the number of samples on all traces are normalized to a constant (i.e., 50) as Eq. 1 in Sec. III-A. An MGD model is built for each sample, which contains variables of movement and pressure dynamics $\Delta x$, $\Delta y$, and $\rho$. Thus, a set of 50 MGD models were trained for complete traces from a gesture type of a user. These models output a set of probabilities to represent the similarities between the testing trace and the trained model. These probabilities are summed to a single score for authentication. The second GTGF-based method is to extract the GTGF feature and compute the $L_1$ distance between traces from the training set and the testing set [8]. The third method is the proposed STDI-based method. For the experiment setup, we randomly selected half of the traces in each session in the UH-TOUCH v2 and combined to the gallery or training set. The remaining portion from UH-TOUCH v2 was used as the probe or testing set. Figure 7 depicts the comparisons among these three methods in terms of EER in Fig. 7(a) and RR in Fig. 7(b). In the verification experiments, the average EERs for MGD, GTGF and STDI on the six gestures were $36.8\%$, $12.8\%$ and $9.7\%$, respectively. The improvement of the GTGF over the MGD is obvious ($12.8\%$ vs $36.8\%$), which indicates the necessity of adopting the GTGF feature. The decrease of EER from $12.8\%$ to $9.7\%$ demonstrates that the traces are better represented and more distinguishable by STDI. In the recognition experiments, the average RRs for MGD, GTGF and STDI on the six gestures were $55.2\%$, $79.0\%$ and $82.3\%$, respectively. The increase of RR demonstrates the same evidence as in the verification experiments. By adopting GTGF and STDI, touch traces can be better represented and classified in user identification and verification problems.
TABLE III
A COMPARISON OF EER AND RR FOR DIFFERENT FUSION SCHEMES AND METHODS

<table>
<thead>
<tr>
<th></th>
<th>GTGF-S</th>
<th>GTGF-M</th>
<th>STDI-S</th>
<th>STDI-M</th>
</tr>
</thead>
<tbody>
<tr>
<td>EER</td>
<td>6.3%</td>
<td>10.5%</td>
<td>4.7%</td>
<td>9.6%</td>
</tr>
<tr>
<td>RR</td>
<td>84.1%</td>
<td>79.2%</td>
<td>91.5%</td>
<td>80.7%</td>
</tr>
</tbody>
</table>

GTGF refers to the method proposed in [8]; TA refers to the Touchalytics method in [7]; DM refers to the method in [6]; STDI refers to the proposed method.

3) Evaluation of Score Fusion and Baseline Comparisons:
To evaluate the performance of the proposed method using different combinations of touch gestures (combining single touchtip gestures, combining multiple touchtip gestures and combining all six gestures) and compare the method with the methods available in the literature, we tested the method on UH-TOUCH v2. The same gallery and probe set were used as in the previous experiment. After score computation, we obtained one score matrix for each gesture. Then, we fused the score matrix of multiple gestures using the sum rule, which is proved by Kittler et al. [27] to be superior in comparison to other rules (i.e., product, min, max, median rule). Table III depicts a comparison of the EER and RR for different fusion schemes and methods: (i) GTGF-S: fusion of the score matrices of four single touchtip gestures computed from the aforementioned GTGF method [8]; (ii) GTGF-M: fusion of the score matrices of the two multiple touchtip gestures computed from the GTGF method [8]; (iii) STDI-S: fusion of the score matrices of four single touchtip gestures computed from the proposed method; (iv) STDI-M: fusion of the score matrices of the two multiple touchtip gestures computed from the proposed method; (v) GTGF-A: fusion of the score matrices of the six gestures computed from the aforementioned GTGF method [8]; (vi) STDI-A: fusion of the score matrices of the six gestures computed from the proposed method; (vii) TA-A: fusion of the score matrices of the six gestures computed from the method in [7]; (viii) DM-A: fusion of the score matrices of the six gestures computed from the method in [6]. As in most score-level fusion schemes, combining more scores from different channels increases the overall performance. This trend can be observed from the decrease in EER and increase in RR from the STDI-M (including only two multi-touch gestures) to the STDI-S (including four single-touch gestures) and further to the STDI-A schemes (including all six gestures). Furthermore, the performance of the GTGF-based method demonstrated a clear improvement over the other methods in the scheme of combining all six gestures. There are two reasons for this improvement. First, compared with the method in [6], we adopted tactile pressure in our method, which contains extra clues on the subject’s muscle behavior. However, their method did not include this information. Second, compared to the work in [7] which extracted 30 static analytic feature values, the proposed method includes movement dynamics and pressure dynamics of the touch gesture during feature extraction. Last, the best performance in terms of recognition and verification has been achieved by the STDI-A scheme. Its improvement over GTGF-A is derived from statistical analysis on the intra-class variations in GTGF representation.

4) Evaluation of Continuous Mobile Authentication and Baseline Comparisons: To test the performance and usability of the STDI based user authentication method in a real world scenario, we used the implemented App STDI – U/A to verify users. Opendset user recognition is excluded from this test since the Android system (version 4.1.x) does not currently support multiple user account management. A gallery of traces ($N_g$ per gesture) for each subject is formed by randomly selecting traces ($N_g/5$ per gesture) from each session in UHTOUCH v2 to cover variations in touch manner. The traces in UHTOUCH v3 are used as the probe set in the test. During testing, the app ran on a Samsung Galaxy S3 phone with Android version 4.1.1. It has Quad-core 1.4 GHz Cortex-A9 CPU and 1GB RAM. The computational complexity for GTGF is $O(N_g * N_p)$, and the computational complexity for STDI is $O(N_g)$, where $N_g$ is the number of subjects in the gallery. Note that one STDI is built for each subject per gesture from the methods available in the literature, we tested the method on UH-TOUCH v2. The same gallery and probe set were used as in the previous experiment. After score computation, we obtained one score matrix for each gesture. Then, we fused the score matrix of multiple gestures using the sum rule, which is proved by Kittler et al. [27] to be superior in comparison to other rules (i.e., product, min, max, median rule). Table III depicts a comparison of the EER and RR for different fusion schemes and methods: (i) GTGF-S: fusion of the score matrices of four single touchtip gestures computed from the aforementioned GTGF method [8]; (ii) GTGF-M: fusion of the score matrices of the two multiple touchtip gestures computed from the GTGF method [8]; (iii) STDI-S: fusion of the score matrices of four single touchtip gestures computed from the proposed method; (iv) STDI-M: fusion of the score matrices of the two multiple touchtip gestures computed from the proposed method; (v) GTGF-A: fusion of the score matrices of the six gestures computed from the aforementioned GTGF method [8]; (vi) STDI-A: fusion of the score matrices of the six gestures computed from the proposed method; (vii) TA-A: fusion of the score matrices of the six gestures computed from the method in [7]; (viii) DM-A: fusion of the score matrices of the six gestures computed from the method in [6]. As in most score-level fusion schemes, combining more scores from different channels increases the overall performance. This trend can be observed from the decrease in EER and increase in RR from the STDI-M (including only two multi-touch gestures) to the STDI-S (including four single-touch gestures) and further to the STDI-A schemes (including all six gestures). Furthermore, the performance of the GTGF-based method demonstrated a clear improvement over the other methods in the scheme of combining all six gestures. There are two reasons for this improvement. First, compared with the method in [6], we adopted tactile pressure in our method, which contains extra clues on the subject’s muscle behavior. However, their method did not include this information. Second, compared to the work in [7] which extracted 30 static analytic feature values, the proposed method includes movement dynamics and pressure dynamics of the touch gesture during feature extraction. Last, the best performance in terms of recognition and verification has been achieved by the STDI-A scheme. Its improvement over GTGF-A is derived from statistical analysis on the intra-class variations in GTGF representation.

4) Evaluation of Continuous Mobile Authentication and Baseline Comparisons: To test the performance and usability of the STDI based user authentication method in a real world scenario, we used the implemented App STDI – U/A to verify users. Opendset user recognition is excluded from this test since the Android system (version 4.1.x) does not currently support multiple user account management. A gallery of traces ($N_g$ per gesture) for each subject is formed by randomly selecting traces ($N_g/5$ per gesture) from each session in UHTOUCH v2 to cover variations in touch manner. The traces in UHTOUCH v3 are used as the probe set in the test. During testing, the app ran on a Samsung Galaxy S3 phone with Android version 4.1.1. It has Quad-core 1.4 GHz Cortex-A9 CPU and 1GB RAM. The computational complexity for GTGF is $O(N_g * N_p)$, and the computational complexity for STDI is $O(N_g)$, where $N_g$ is the number of subjects in the gallery. Note that one STDI is built for each subject per gesture from the methods available in the literature, we tested the method on UH-TOUCH v2. The same gallery and probe set were used as in the previous experiment. After score computation, we obtained one score matrix for each gesture. Then, we fused the score matrix of multiple gestures using the sum rule, which is proved by Kittler et al. [27] to be superior in comparison to other rules (i.e., product, min, max, median rule). Table III depicts a comparison of the EER and RR for different fusion schemes and methods: (i) GTGF-S: fusion of the score matrices of four single touchtip gestures computed from the aforementioned GTGF method [8]; (ii) GTGF-M: fusion of the score matrices of the two multiple touchtip gestures computed from the GTGF method [8]; (iii) STDI-S: fusion of the score matrices of four single touchtip gestures computed from the proposed method; (iv) STDI-M: fusion of the score matrices of the two multiple touchtip gestures computed from the proposed method; (v) GTGF-A: fusion of the score matrices of the six gestures computed from the aforementioned GTGF method [8]; (vi) STDI-A: fusion of the score matrices of the six gestures computed from the proposed method; (vii) TA-A: fusion of the score matrices of the six gestures computed from the method in [7]; (viii) DM-A: fusion of the score matrices of the six gestures computed from the method in [6]. As in most score-level fusion schemes, combining more scores from different channels increases the overall performance. This trend can be observed from the decrease in EER and increase in RR from the STDI-M (including only two multi-touch gestures) to the STDI-S (including four single-touch gestures) and further to the STDI-A schemes (including all six gestures). Furthermore, the performance of the GTGF-based method demonstrated a clear improvement over the other methods in the scheme of combining all six gestures. There are two reasons for this improvement. First, compared with the method in [6], we adopted tactile pressure in our method, which contains extra clues on the subject’s muscle behavior. However, their method did not include this information. Second, compared to the work in [7] which extracted 30 static analytic feature values, the proposed method includes movement dynamics and pressure dynamics of the touch gesture during feature extraction. Last, the best performance in terms of recognition and verification has been achieved by the STDI-A scheme. Its improvement over GTGF-A is derived from statistical analysis on the intra-class variations in GTGF representation.
from 32.22 ms to 81.36 ms. However, the computational time for STDI remains at around 12.90 ms. The third and fourth rows in Table IV depict the average EER in user authentication using five continuous traces. These EERs were achieved by GTGF and STDI methods with different gallery size respectively. As $N_g$ increased from 30 traces per gesture to 100 traces, the EERs achieved by GTGF and STDI decreased. When using more than 80 traces, the EER improvement for both methods has been reduced.

Figure 8 depicts a comparison of the continuous authentication results among GTGF, TA [7] and STDI in terms of the average FRR (solid line) and average FAR (dash line). The distance from the probe trace and the distances from the following $\gamma = 1$ consecutive probe traces are averaged to obtain the score for authentication. If it is above the threshold, the app classifies the user as an unauthorized intruder and invokes explicit user authentication. FARs for all methods reduced in a small but steady rate as the $\gamma$ increased. Meanwhile, the FRRs of all methods decreased significantly as the number of traces increased. For the GTGF and STDI methods, the decreasing rate reduced after the $\gamma > 5$ and almost converged when $\gamma = 7$. When $\gamma > 5$, the delay caused by the increased computational burden became noticeable for GTGF. However, no obvious delay has been observed for the STDI method, even when $\gamma = 7$.

V. CONCLUSION

This paper presented a novel mobile authentication approach based on touch gestures. An STDI, which synthesizes trace feature instances using intra-person variations, has been proposed. Using a multi-session touch gesture dataset containing six commonly used gestures, we have systematically evaluated the performance of the proposed method. Comparisons between the STDI and other methods in the literature have also been conducted. An STDI-based User Verification App has been implemented and tested on a dataset collected during the daily use of the phone. The results have demonstrated the effectiveness and usability of the STDI in the real-world mobile authentication.

REFERENCES

Xi Zhao received the Ph.D. (Hons.) degree in computer science from the Ecole Centrale de Lyon, Ecully, France, in 2010. After graduation, he conducted research in the fields of biometrics, face analysis, and pattern recognition, as a Post-Doctoral Fellow with the Department of Computer Science, University of Houston, Houston, TX, USA. He is currently a Research Assistant Professor with the Department of Computer Science, University of Houston. His current research interests include biometrics, mobile computing, computer vision, and assistive technology.

Dr. Zhao currently serves as a reviewer of the IEEE TRANSACTIONS ON IMAGE PROCESSING, the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS - PART B, the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, Image Vision Computing, the IEEE International Conference on Automatic Face and Gesture Recognition, the IEEE International Conference on Advanced Video and Signal-Based Surveillance, and the International Conference on Pattern Recognition. He has also served on the Program Committee of Workshop on 3-D Face Biometrics, in 2013, and the Registration Chair of the International Conference on Biometrics: Theory, Applications and Systems, in 2013.

Tao Feng received the B.S. degree in software engineering from the Huazhong University of Science and Technology, Wuhan, China, and the M.Sc. degree in computer science from the Chinese Academy of Sciences, Institute of Automation, Beijing, China, with a focus on knowledge system and knowledge visualization.

He is currently pursuing the Ph.D. degree with the Department of Computer Science, University of Houston, Houston, TX, USA. His current research interests include mobile computing, artificial intelligence, human–computer interaction, mobile security, and machine learning.

Weidong Shi received the Ph.D. degree in computer science from the Georgia Institute of Technology, Atlanta, GA, USA, where he was involved in research on computer architecture and computer systems. He was a Senior Research Staff Engineer with the Motorola Research Laboratory, Nokia Research Center, and the Co-Founder of a technology startup. He is currently an Assistant Professor with the University of Houston, Houston, TX, USA. He was involved in designing multiple Nvidia platform products, and was credited to publish Electronic Art’s console game. He has authored or coauthored publications covering research problems in computer architecture, computer systems, multimedia/graphics systems, mobile computing, and computer security. He holds multiple issued and pending U.S. and Trademark Office patents.

Ioannis A. Kakadiaris is currently a Hugh Roy and Lillie Cranze Cullen University Professor of Computer Science, Electrical and Computer Engineering, and Biomedical Engineering with the University of Houston (UH), Houston, TX, USA. He joined UH, in 1997, after a post-doctoral fellowship with the University of Pennsylvania, Philadelphia, PA, USA. He received the B.Sc. degree in physics from the University of Athens, Athens, Greece, the M.Sc. degree in computer science from Northeastern University, Boston, MA, USA, and the Ph.D. degree from the University of Pennsylvania. He is the Founder of the Computational Biomedicine Laboratory at UH. His research interests include multimodal biometrics, face recognition, computer vision, pattern recognition, biomedical image analysis, and predictive analytics. He was a recipient of a number of awards, including the NSF Early CAREER Development Award, the Schlumberger Technical Foundation Award, the UH Computer Science Research Excellence Award, the UH Enron Teaching Excellence Award, and the James Muller Vulnerable Plaque Young Investigator Prize. His research has been featured on the Discovery Channel, National Public Radio, KPRC NBC News, KTRH ABC News, and KHOU CBS News. His selected professional service leadership positions include: the General Cochair of the 2013 Biometrics: Theory, Applications and Systems Conference, General Cochair of the 2014 SPIE Biometric and Surveillance Technology for Human and Activity Identification, Program Cochair of the 2015 International Conference on Automatic Face and Gesture Recognition Conference, and Vice President for Technical Activities of the IEEE Biometrics Council.
AUTHOR QUERY

AQ:1 = Algorithms 3 and 4 are not cited in body text. Please indicate where they should be cited.