*Intelligent Automation & Soft Computing*
DOI: 10.32604/iasc.2023.029893
***Article***

Tech Science Press

# User Interface-Based Repeated Sequence Detection Method for Authentication

**Shin Jin Kang[1] and Soo Kyun Kim[2,*]**

[1]School of Games, Hongik University, Sejong, 30016, Korea
[2]Department of Computer Engineering, Jeju National University, Jeju, 63243, Korea
*Corresponding Author: Soo Kyun Kim. Email: kimsk@jejunu.ac.kr
Received: 14 March 2022; Accepted: 21 April 2022

**Abstract:** In this paper, we propose an authentication method that use mouse and keystroke dynamics to enhance online privacy and security. The proposed method identifies personalized repeated user interface (UI) sequences by analyzing mouse and keyboard data. To this end, an Apriori algorithm based on the keystroke-level model (KLM) of the human–computer interface domain was used. The proposed system can detect repeated UI sequences based on KLM for authentication in the software. The effectiveness of the proposed method is verified through access testing using commercial applications that require intensive UI interactions. The results show using our cognitive mouse-and-keystroke dynamics system can complement authentication at the application level.

**Keywords:** Authentication; keystroke-level model (KLM); keystroke; mouse dynamics

## 1 Introduction

With the rapid expansion of the cyber world, security- and privacy-enhancing issues have become a focus in many fields, such as in networking, games, and web services. Security and authentication methods have been explored for a long time, and many efficient approaches have been developed. Password-based user authentication methods are the most commonly used online user authentication method. However, if a password is exposed to an imposter, it can easily be disabled. Biometric and behavioral technologies have been used to address this issue, because they provide a more reliable means of authentication when combined with traditional authentication methods. Mouse and keystroke dynamics are two emerging behavioral technologies that are used to authenticate users when using mouse and keyboard peripherals [1]. Authentication based on mouse and keyboard dynamics is performed by observing changes in the user's typing patterns. This process uses a temporal typing pattern or a diagram. Earlier mouse and keystroke authentication methods were typically performed during user login on a fixed string or diagram, such as a password or pattern drawing. Recent mouse and keystroke authentication methods focus on handling both free input data and fixed data, even after the login stage [2]. However, technical difficulties arise when capturing mouse motion because measurement errors occur at low operating system levels. This is because software typically only measures the pointer motion rather than physical mouse motion. There are numerous mouse types available, each with different tracking resolutions. In addition, many different preprocessing methods can be applied by the operating system to obtain onscreen pointer motions from the raw mouse motion [3].

To solve this problem, we propose a mouse and keystroke dynamics model that adopts a higher cognitive-level model. The latter model is intended to be less sensitive to hardware and input/output preprocessing than motor-level behavior. The system uses a user interface (UI) sequence pattern matching system at the application level. This system can be used in any application after the login stage. Our system requires simple basic interaction data, specially, the mouse trajectory and keyboard input—for continuous data monitor. With these data, we use the Apriori algorithm for repeated sequence detection based on the keystroke-level model (KLM) of the human–computer interface (HCI) domain [4]. Our system consists of all KLM components, that are automatically measured. The system verifies the user's identity by considering sequence similarity and using application log data. We verified the proposed system using Adobe Photoshop and Autodesk 3DS MAX software, which require several repeated UI sequence behaviors.

Fig. 1 depicts the authentication process of the proposed system. The system characteristics are outlined as follows.

- The proposed system uses UI sequence data for behavioral feature authentication. Because the UI usage pattern is personalized and difficult to emulate, it can be useful for authentication in applications that require intensive UI interactions with a non-intrusive validation process.
- The proposed model uses KLM as data encoding rules for authentication. Because KLM has expandable rules that can integrate keyboard and mouse data into a single data format, the previously separated keyboard and mouse dynamics methods can be easily integrated.
- The proposed model employs a cognitive-level behavioral analysis model with simple interaction data. In contrast existing methods that use motor-level analysis, the proposed method is unaffected by individual hardware settings. Therefore, it enables a robust high-level context analysis without hardware-setting dependency.

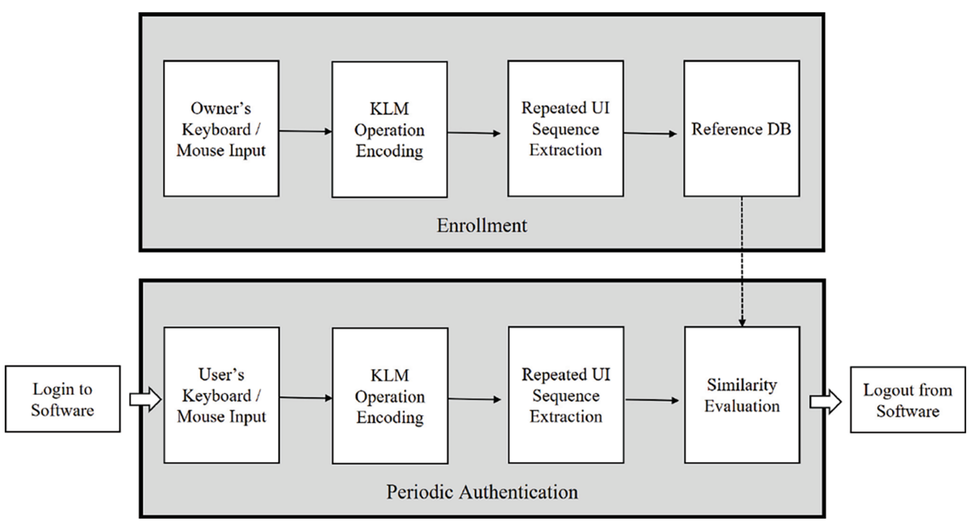



**Figure 1:** Proposed authentication process during software operation

## 2 Related Work

In terms of work related to the present study, an authentication system can be performed in three ways: through information authentication (e.g., password and, personal identification number), ownership authentication (e.g., RFID card and, security token), and biometric or behavioral feature authentication [5]. Biometric feature authentication uses automated methods to verify the identity of a person based on their physiological characteristics. Biometric technologies have become widespread, because they provide an additional level of security when used in combination with traditional information and ownership authentication methods. Specifically, biometric technologies involve the identification of personal characteristics, such as fingerprints, irises, faces, audio, and voice [6–8].

Biometric feature authentication is an effective approach because the identified characteristic are typically permanent and universally unique. However, it is limited by the requirement of obtrusive special hardware to capture biometric data. Moreover, the algorithm has high computational complexity. However, behavioral feature authentication methods, such as mouse [9] and keystroke dynamics [10], are strong approaches because they require simple non-intrusive hardware and have low computational complexity. Mouse and keystroke dynamics comprise of an authentication system based on strong behavior-recognition technology. It analyzes the manner in which a user types at a terminal by monitoring the mouse and keyboard. Accordingly, users are identified based on their habitual typing patterns [11]. Gamboa et al. [12] proposed a user verification method based on mouse dynamics from a user's interaction with a simple game. In their study, the user was tasked with identifying the matching tiles. The user's identity was verified based on the characteristics of the mouse strokes performed to reveal tiles. In their method, a mouse stroke is defined as a set of point trajectories. A set of one or more strokes was used to verify the user.

Moreover, Pusara et al. [13] proposed a web-based verification method that records users' mouse movements while browsing a website. The users were classified using a decision tree algorithm. This method resulted in a false acceptance rate (FAR) of 0.46% and false rejection rate (FRR) of 1.75%. Ahmed et al. [14] used the mouse activities of the users performing daily tasks in their selected applications. The features were sampled and aggregated into histograms to characterize each user. Revett et al. [15] proposed a system that uses a UI in which the correct sequence of elements was arranged using a mouse. This system was intended to replace text-based passwords. Their results showed an FAR of 3.5% and FRR of 4.0%. Furthermore, Bours et al. [16] authenticated users by navigating a mouse through an onscreen maze. The method was resulted in a EER of 27%. Zheng et al. [9] proposed a method to identify users with lesser than 20 mouse clicks. It is computationally less complex than earlier methods because it use a simple SVM classification method. Their method showed an FAR of 1.3% and FRR of 1.3%. Furthermore, Feher et al. [17] proposed a verification method based on the observation of each mouse action performed by the user. Mouse actions are atomized into a single click or movement. More complex actions are decomposed into unit actions such as left-click, right-click, mouse-move sequence, and drag-and-drop.

The keyboard dynamics model uses a classification model that uses the feature vectors extracted while the user types fixed or free data. Bergadano et al. [18] extracted typing durations of consecutive characters from a sample and associated them with the user. The extracted graphs were sorted by their durations; therefore, the relative ordering results were compared with those of other users. Some keyboard-based methods use feature vectors, whereas the user freely type the text. Gunetti et al. [19] extended this approach to handle free text and proposed another distance measure based on absolute times. Curtin et al. [20] used a nearest-neighbor classifier trained using the duration of common characters, transition times of common digraphs, and occurrence frequencies of keys. Ferreira et al. [21] used n-graph duration and down-up (DU) and up-down (UD) times. They proposed a method for continuous access control enforcement using keystroke dynamic biometrics, which were adapted for the host intrusion detection operation. Monaco et al. [22] used the DU and UD times along with the down-down (DD) time in their feature vectors, whereas Locklear et al. [23] used cognitive features. They proposed the continuous identification of 123 behavioral features extracted from discrete cognitive units. Several software applications have been presented to characterize user behavioral biometrics at the application level [24]. These include mobile devices [25–27], industrial Internet of Things (IIoT) systems [28], and behavior profiling systems [29]. Augmented reality [30], mixed reality [31], and synthetic image [32] technologies are also available. The proposed method follows a different behavioral feature authentication approach that focuses on cognitive aspects during the application operation. In contrast mouse and keystroke dynamics in sketching applications, which use data from mouse and keyboard strokes from a simple drawing test, we focus on UI input sequences for authentication. We define a sketch as a set of UI interactions and statistically correlate drawing primitives of different complexities. To record UI interactions as a set of behavior sequences, we adopted KLM, which is a representative behavior modeling method in the HCI domain. This represents a practical use of the mouse and keystrokes. KLM can predict the behavior of users with few errors; however, it requires a specific method to predict the time interval. KLM does not have set goals or method-selection rules, facilitating its application to various application domains.

The proposed method captures the behaviors that operate at the cognitive level. They should be less sensitive to hardware and IO preprocessing than existing mouse and keystroke dynamic models that use motor-level behaviors. To this end, we adopted KLM-based evaluation tools designed to provide support various usability evaluation methods. These methods are integrated with a repetitive sequence-detection algorithm for authentication. To the best of our knowledge, our system is the first to adopt KLM for behavioral feature authentication with UI-related behaviors.

The remainder of this paper is organized as follows. In Section 3, we explain the KLM model. Section 4 describes the repeated sequence detection algorithm, Section 5 provides the implementation details, and Section 6 presents the results. Finally, we discuss the results and provide our conclusions in Section 7.

## 3 KLM Model and Methods

KLM evaluates the time required for a user to complete a specific task without errors in a typical computer environment. If Ttask is the total time required to finish a task, then it can be calculated as follows:

$$T_{task} = T_{acquire} + T_{execute}$$

where, $T_{acquire}$ represent the time required to select the method to accomplish the task, and $T_{execute}$ represent the time required to select the approach to perform the selected method. Thus, KLM estimates $T_{execute}$. In KLM, $T_{execute}$ can be calculated as the sum of the primitive operators.

$$T_{execute} = \sum \text{time to execute primitive operators}$$

Here, the primitive operations consist of a button and key press (K), point to target with a mouse (P), hold hands on the keyboard and mouse (H), draw a line with a mouse (D), mental preparation; pause (M), and system response time (R). The first four operations are physical motor operators, followed by a mental operator, and a system response operator. If an operator is fully anticipated, then M can be ignored. In general, the time required to complete primitive operations is predicted from the experiments. In this study, we classified the users based on their repetitive task sequences. To this end, each keyboard and mouse behavior was encoded using a cognitive-level semantic rule. We adopted KLM as the semantic rule because its versatility has been verified in the HCI domain for various applications [4].

To generate reliable encoded data, we used the results of Card et al. [33]. Tab. 1 lists the KLM operation encoding styles and their descriptions. Because experts simultaneously use keyboard input operations while using a PC with a mouse (e.g., modeling and digital painting behavior), the homing operation was not considered. Therefore, in this study, we assumed that mouse and keyboard data could be sampled in parallel. We additionally assumed that M occurred during an idling interval, and the remaining interval comprised other operation times. In this study, Fitt's law was used to measure pointing (P) [34]. Each KLM operation was recorded in a "KLM operator (property)" format. To achieve this, users must assign the application area using an area assignment tool. Typically, the application area was labeled for each basic UI component.

**Table 1:** Customized KLM in the proposed system

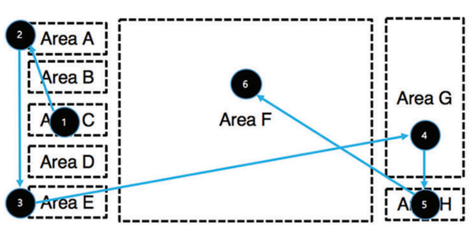
| Operation | Description | Encoding style |
|---|---|---|
| K | Keystroke, pressing a key or button on the keyboard | K(*Pressed_Key_Name*) |
| P | Pointing with the mouse to a target on the display | P(*Pressed_Application_Area*) |
| D | Dragging | D(*Dragging_Application_Area*) |
| B | Pressing or releasing the mouse button | B(*Clikced_Application_Area*) |
| H | Homing hands to the keyboard or mouse | N/A |
| M | Estimated mental operation | M(*Idling_Time*) |

## 4 Repeated Sequence

### *4.1 Repeated Sequence Detection*

In a digital painting or 3D modeling environment, the keyboard and mouse are used with many repetitive movements to correlate with UI components. To determine a high sequential association pattern, we used the Apriori algorithm [35] in the datamining field. This algorithm mines frequent sets of items and learns association rules in a transactional database. It identifies individual items that are frequently used in the database and expands to larger sets of items if the set appears sufficiently frequently in the database. The set of frequent items determined by Apriori can be used to determine the association rules.

Professional users of digital painting or 3D modeling software generally use unique sequential shortcuts to improve their content creation productivity. For example, users may frequently use certain repeating sequences (e.g., Press P (Pencil) > Change Size) in a digital painting application. This sequence can be used as the key authentication data. Therefore, using the KLM operation, a pattern table with compressed code values was generated every second. The n-gram, which appeared repeatedly, was extracted. Each time a new code was detected, a new pattern was added to the pattern table. After the pattern table was created, continuous keyboard patterns and mouse movement patterns were extracted from 3- to n-gram [4]. Fig. 2 illustrates the repeated sequences in Photoshop, which were used as reference data for authentication.




| Patterns | Percentage |
|---|---|
| B(C)-B(A)-B(E)-B(G)-B(5)-B(F) | 5.6% |
| D(F)-B(G)-B(H) | 2.4% |
| ..... | ...... |
| K(F)-K(3)-K(2)-K(M) | 1.2% |

**Figure 2:** Personalized UI sequential behaviors in photoshop and the repeated UI sequence pattern table

### *4.2 Repeated Sequence Similarity Evaluation*

The basic similarity evaluation involves comparing a criterion set of a specific user's typing characteristics to a test set of the same user's typing characteristics or an imposter's test set. The distance between these two sets (reference and test) must be less than a certain threshold. Otherwise, the user will be perceived as an imposter. The first task was to improve the precision of the sequence similarity problem. A KLM string is a sequence of KLM operations obtained by analyzing the mouse and keyboard movements. This basic concept is based on the research of Gusfield [36].

Given two KLM strings, similarity was identified by finding an exact alignment between the two KLM strings. The proposed system evaluates the similarity between such alignments using a simple scoring function. For example, if an exact match between two characters occurs, the system confers a +2 score for the pairs; for each occurrence of a mismatch, the system confers a −1 score.

K(D)-K(3)-K(5)-D(7)

K(D)-K(3)-K(3)-D(7)    Similarity Score: 3 x (2) + 1 x (-1) = 5

We used three- to six-character sequences and compared them with sequences of the same length. Our sequence detection algorithm can generate static n-gram sequences; therefore, the alignment problem does not occur between different string lengths. Tab. 2 list examples of the similarity score calculation.

**Table 2:** Example of similarity scores

| Grams | Authentication pattern | Input pattern | Similarity score |
|---|---|---|---|
| 3 grams | D(1)-K(5)-K(4) | D(1)-K(5)-K(4) | 6 |
| 4 grams | K(D)-K(3)-K(5)-D(7) | K(D)-K(3)-K(3)-D(7) | 5 |
| 5 grams | K(D)-K(3)-K(5)-D(7)-B(N) | K(H)-K(2)-K(5)-D(7)-B(N) | 4 |
| 6 grams | B(D)-P(3)-B(5)-P(7)-B(N)-K(M) | B(D)-P(3)-B(K)-P(7)-B(N)-K(M) | 6 |
| Total similarity score | | | 21 |

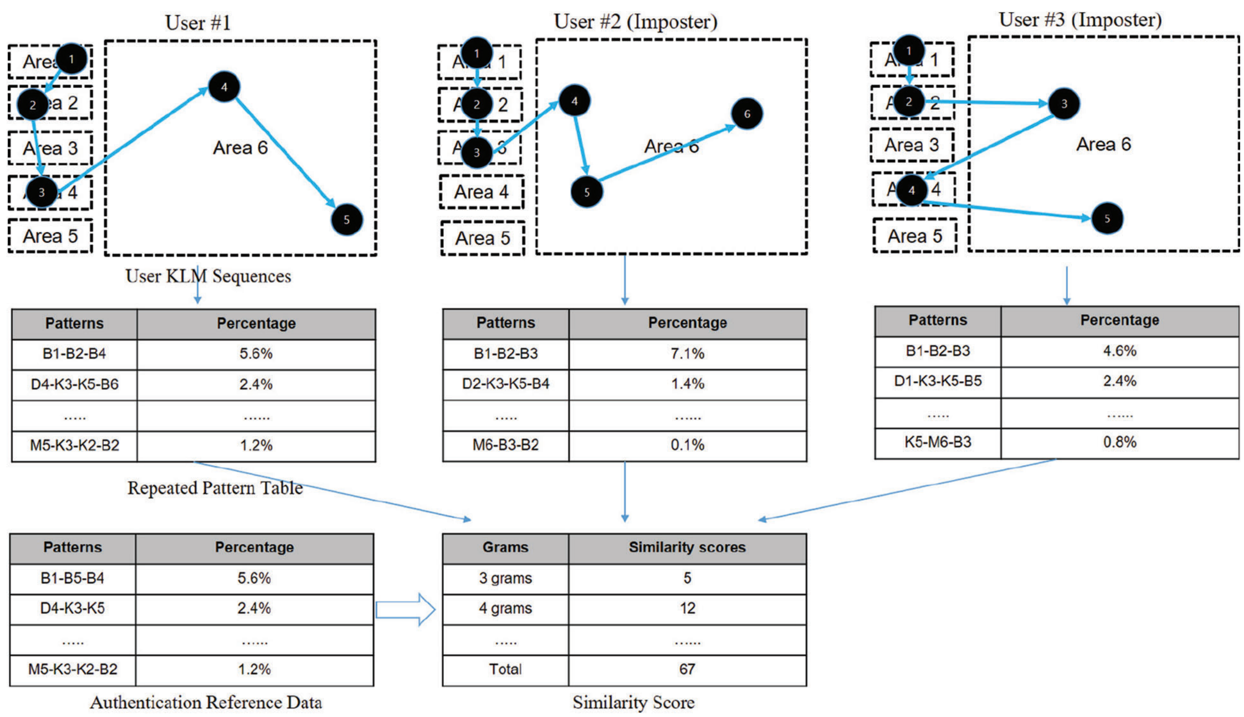
In the proposed system, users have their own authentication table consisting of 3–6 grams. This was generated during the training stage. A target image can be obtained from items ranging from simple diagrams to complex natural paintings. In this study, we used a simple cartoon-style painting and 3D modeling. When a supposed imposter uses the application after hacking the password-based authentication process, the system compares the imposters' pattern table with the owners' pattern table of similarity scores. If the score is below the threshold value, the system assumes that the currently logged-in user could be an imposter. Fig. 3 shows the process of calculating the similarity scores using a repeated pattern table.

## 5 Implementation

The objective of the proposed system is to identify users effectively based on repetitive UI behavior sequences. Therefore, we extended our previous log-sampling system [37]. It detects input UI behavior sequences in real-time using event-hocking techniques. The system was implemented in C# and expanded to four modules for this research: 1) encoding module for the KML operation, 2) sequence detection module, 3) UI area assignment tool, and 4) similarity check module.

Fig. 4 shows the architecture of the proposed system and its authentication process. Fig. 5 (left) shows the UI of the proposed modules 1 and 3. Because our system can run independently with test target software, its behavior with any application can be easily detected. The sampled results were exported to the CSV files.

Fig. 5 (right) shows an example of the application area (A1 to A7), which is marked for KLM encoding. The KLM operation encoding module generates a time-series KLM operation log based on the keyboard and mouse events. Every PC activity is converted into a KLM operation, as outlined in Tab. 1. This module is application-independent; therefore, the behavior of any application can be easily detected. The UI area assignment tool functions in user assignments. This is necessary because external experimental systems cannot access an application's internal UI identification from outside the application without using a software development kit. However, this approach is not feasible in most applications.




| Patterns | Percentage |
| --- | --- |
| B1-B2-B4 | 5.6% |
| D4-K3-K5-B6 | 2.4% |
| ..... | ...... |
| M5-K3-K2-B2 | 1.2% |

| Patterns | Percentage |
| --- | --- |
| B1-B2-B3 | 7.1% |
| D2-K3-K5-B4 | 1.4% |
| ..... | ...... |
| M6-B3-B2 | 0.1% |

| Patterns | Percentage |
| --- | --- |
| B1-B2-B3 | 4.6% |
| D1-K3-K5-B5 | 2.4% |
| ..... | ...... |
| K5-M6-B3 | 0.8% |

| Patterns | Percentage |
| --- | --- |
| B1-B5-B4 | 5.6% |
| D4-K3-K5 | 2.4% |
| ..... | ...... |
| M5-K3-K2-B2 | 1.2% |

| Grams | Similarity scores |
| --- | --- |
| 3 grams | 5 |
| 4 grams | 12 |
| ..... | ...... |
| Total | 67 |

**Figure 3:** Processing of a repeated pattern table based on KML operations

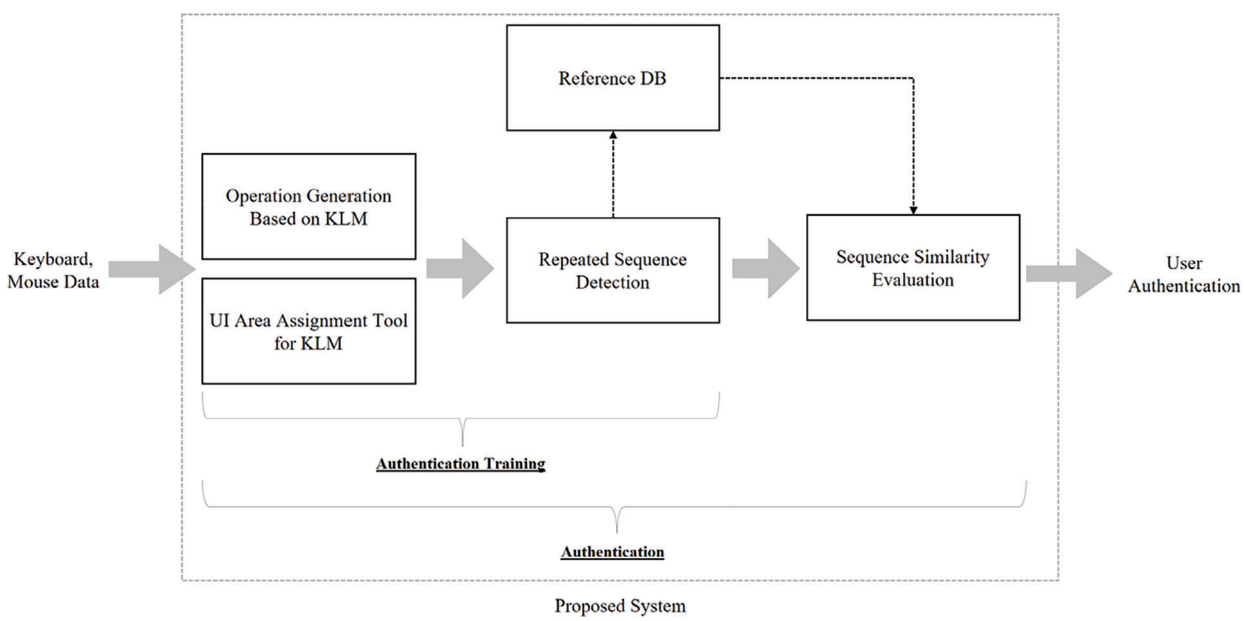



**Figure 4:** Overall system process

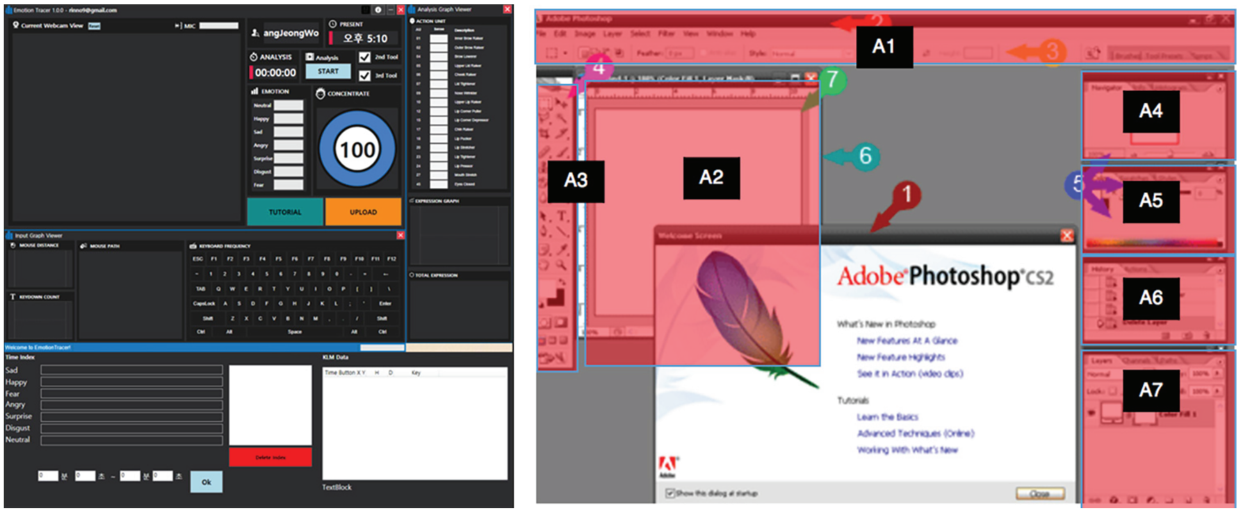



**Figure 5:** KML operation encoding module and UI area assignments with a rectangle painting interface

The repeated sequence detection module analyzes the areas labeled by the KLM task sequences and generates a table of the most repeating sequence patterns. Specifically, this table lists the most repeated KLM task sequences converted to higher-level context sequences using a replacement table. It is possible for users to save their repeated sequence pattern tables in real-time using these three modules during the authentication training stage. This stage can be repeated to achieve a more stable authentication performance. After authentication training, the system calculates the similarity using the sequence similarity evaluation module. These processing steps automatically create KLMs. Computer owners were identified based on previously collected information on keystroke dynamic profiles for specific applications. The repeated UI sequence was calculated during this training stage and tested during the application-run stage.

Encoding exceptions exist because of the physical limitations of human behavior. For example, in an actual event log, there are a small number of mouse movement events between the mouse clicks and input. To compensate for this problem, we include a post-processing step that removes these extremely short-lived events by setting a 0.1 s sampling interval. When sampling is complete, all information is converted into a series of KLM operators.

## 6 Experiments

To verify the proposed system, two applications, Autodesk 3DS MAX and Adobe Photoshop, were used. Because these applications require intensive, simultaneous UI interaction with the mouse and keyboard, UI sequences can be a key feature of behavioral feature authentication. These applications are also widely used in 3D/2D digital content creation.

Thus, the usability of the proposed method can be easily evaluated. In these experiments, the system was expected to accept the owner's sequence UI behavior and reject other's sequence UI behavior, based on the recorded repeated pattern table. For each user, the pattern to be identified was matched against each known template, yielding a score that describes the similarity between the template and pattern. The system assigns a pattern to the user with the most similar biometric template. Similarity must be above a certain level to prevent the impostor pattern from being properly identified. If this level is not reached, then the pattern is rejected. First, three painting experiments were conducted using Adobe Photoshop. A total of 30 test participants with 3 or more years of Photoshop experience were recruited. The experiments were conducted as follows. Training and test stages were divided. Each test participant drew three paintings during the training stage, using their own painting styles. Therefore, our system had 90 reference data items (30 test participants × 3 paintings). During the training stage, each participant drew the same three paintings. To focus on the different behavioral styles of the participants and guarantee the same painting experiments as much as possible, we added simple painting sequence numbers with a transparent layer.

During the test stage, whenever each tester painted three paintings, the system detected the given tester using recorded reference data. Because painting behaviors were personalized, testers struggled to perform intrusion attempts by mimicking legitimate access attempts. Therefore, we believe that the system can confer high security by simply passing through repeated sequence pattern matching.

In the experiments, each session was separated by behavior in the painting area (usually in the central area of Photoshop). This is because in painting experiences, each sequential action is completed by performing actions in the center canvas area.

Data were collected during the creation of a digital painting; the canvas size was 2000 × 2000 pixels at 300 dpi, and image resolution was 1920 × 1080 pixels. Fig. 6 shows the target painting images (left) and mouse trajectories (right) recorded using our proposed system.

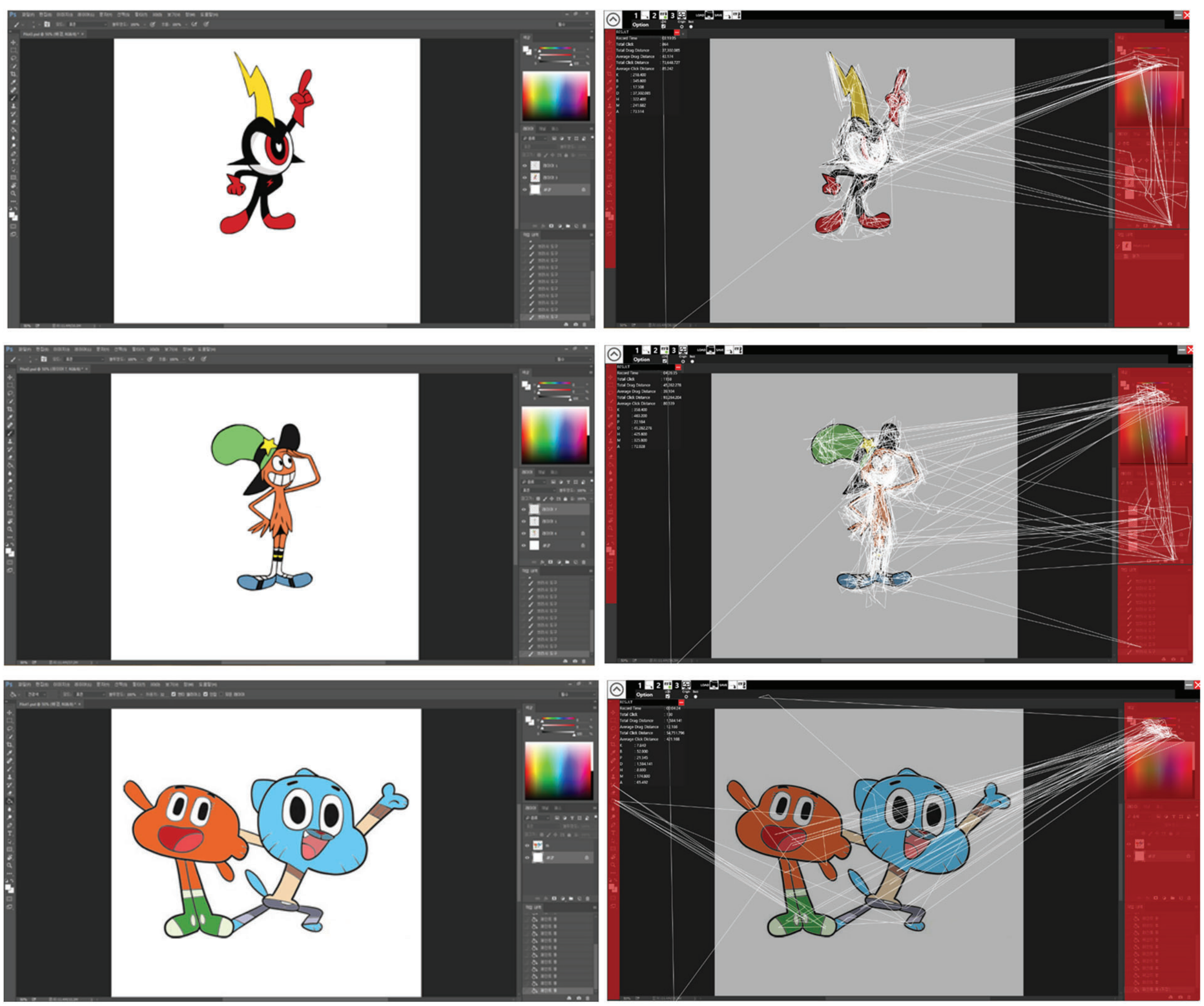


**Figure 6:** Three experiment paintings and sampled mouse trajectories results

Tab. 3 lists the repeated sequence detection results of the owner who completed authentication training with three paintings. Initially, each item was considered as a candidate one-item set. After counting the supports, candidate high-number item sets were generated. The results showed that the selected sequences were the most frequent in all experiments.

**Table 3:** Most frequently repeated owner test patterns for photoshop experiments (Player #1)

| Experiment | Count | Percent | pattern |
|---|---|---|---|
| Painting #1 | 323 | 25.9 | SelectColor() SelectColor() SelectColor() B(0) |
| | 27 | 2.1 | B(0) SelectColor() B(0) |
| | 26 | 2.0 | SelectColor() SelectColor() SelectColor() B(0) |
| | 24 | 1.9 | B(0) SelectColor() SelectColor() B(0) |
| Painting #2 | 41 | 9.1 | SelectColor() SelectColor() SelectColor() Brush() |
| | 7 | 1.5 | SelectColor() Dragging() |
| | 6 | 1.3 | SelectColor() SelectColor() Brush() K(Oem6) |
| | 4 | 0.8 | SelectColor() K(Oem6) |
| Painting #3 | 22 | 11.8 | SelectColor() SelectColor() SelectColor() Brush() |
| | 16 | 8.5 | Click(Target2) Dragging(Target2) |
| | 3 | 1.6 | SelectColor() K(Oem6) |
| | 3 | 1.6 | Click(Target2) Click(Target2) Click(Target2) |

Color selection is a key frequent behavior in digital painting because it occurs during brushstrokes. The ratio in this study could differ depending on the painting size; however, the ratio was high, ranging from 9%–25% of the overall behaviors. Color selection sequences were shown in other paintings, and the patterns were similar. This indicate that the key UI sequence behaviors are similar and independent of the target image style.

Tab. 4 presents the repeated sequence detection results for all testers from 2- to 6-grams. The results showed that different paintings required common painting behaviors. As presented in the table, the 'Color Selection,' 'color slide adjustment,' and 'select layer' operations comprised the most frequently detected behaviors. The minor differences between these two behaviors can be used for personalized pattern detection.

The behavior after color selection may differ depending on the tester. For example, some players used 'add a new layer > color selection' sequences, while others used 'color selection > change brush size' sequences. The patterns belonged to the users and not to the target image style. Some testers did not use the keyboard; therefore, their logs did not include UI-related clicks, except for color selection. In particular, the intensive usage of keyboard shortcuts involved fewer visits to the left screen area; thus, left-area use was not logged. In this case, the repeated pattern was different from the hybrid usage of both the mouse and keyboard. From this result, the characteristic UI sequence pattern can be used as authentication reference data. Because three target images were cartoon-style paintings and segmented into small areas, the 'paint bucket' function was most useful for painting. Essentially, this function requires three related behaviors: 'select brush,' 'select color,' and 'select layers.' Combinations of these three related sequences generated different authentication reference data.

**Table 4:** Summary of most frequently repeated patterns in the photoshop experiments

| Painting #1 | | |
|---|---|---|
| Count | Percentage | Pattern |
| 66 | 8.7766 | SelectColor() B(0) |
| 47 | 6.2500 | SelectColor() SelectColor() |
| 47 | 6.2500 | SelectColor() SelectColor() B(0) |
| 30 | 3.9894 | SelectColor() SelectColor() SelectColor() |
| 30 | 3.9894 | SelectColor() SelectColor() SelectColor() B(0) |
| 18 | 2.3936 | SelectColor() SelectColor() SelectColor() SelectColor() |
| 18 | 2.3936 | SelectColor() SelectColor() SelectColor() SelectColor() B(0) |
| 14 | 1.8617 | B(0) SelectColor() |
| 13 | 1.7287 | B(0) SelectColor() B(0) |
| 13 | 1.7287 | Click(Target2) Click(Target2) Dragging(Target2) |
| 13 | 1.7287 | Click(Target2) Dragging(Target2) Click(Target2) |
| Painting #2 | | |
| Count | Percentage | Pattern |
| 18 | 0.2993 | SelectColor() SelectColor() |
| 6 | 0.0998 | SelectColor() Brush() |
| 6 | 0.0998 | SelectColor() SelectColor() Brush() |
| 6 | 0.0998 | SelectColor() SelectColor() SelectColor() |
| 5 | 0.0831 | SelectColor() SelectColor() SelectColor() Brush() |
| 4 | 0.0665 | SelectColor() K(Oem6) |
| 4 | 0.0665 | Dragging() SelectColor() |
| 3 | 0.0499 | SelectColor() Dragging() |
| 3 | 0.0499 | SelectColor() Brush() K(Oem6) |
| 3 | 0.0499 | SelectColor() SelectColor() Brush() K(Oem6) |
| Painting #3 | | |
| Count | Percentage | Pattern |
| 6 | 0.1479 | SelectColor() SelectColor() |
| 5 | 0.1232 | SelectColor() Brush() |
| 5 | 0.1232 | SelectColor() SelectColor() Brush() |
| 3 | 0.0739 | SelectColor() SelectColor() SelectColor() |
| 3 | 0.0739 | SelectColor() SelectColor() SelectColor() Brush() |
| 3 | 0.0739 | SelectColor() K(Oem6) |
| 7 | 0.1725 | Click(Target2) Dragging(Target2) |
| 3 | 0.0739 | Click(Target2) Click(Target2) Click(Target2) |
| 3 | 0.0739 | Click(Target2) Click(Target2) Dragging(Target2) |
| 3 | 0.0739 | Click(Target2) Dragging(Target2) Click(Target2) |

Experiments were performed on 3D modeling software using the 3DS MAX. Thirty participants were recruited for this study. The same experimental process was performed as in the Photoshop experiment (30 test participants × 3 models). The test participant created three simple chair models consisting of 300–350 vertices. 3DS MAX has a complex interface that supports creative activities in 3D. We assigned six UI areas and tried to check repeated visiting patterns. Tab. 5 shows three representative repeated sequence results from the modeling studies. Because few differences were observed in the target modeling in the 3DS MAX experiments, determining the personalized UI sequence was easier. Five repetitive sequences were selected for each player. This result shows that there was a difference in the simple modeling method for each person. As shown in the table, the repetitive sequence of Player #1 did not include the K operation compared to Players #2 and #3. This indicates that Player #1 used the mouse interface as its main interface. Players #2 and #3 exhibit different UI sequences. Players #2 and #3 simultaneously used the keyboard interface as their input interface. They did not create a 3D model by using the box addition style modeling, but instead performed the vertex editing style modeling suitable for delicate modeling. The 3D MAX experiment revealed differences in the adaptability of the test participants. This led to differences in keyboard shortcut usage and modeling style. The proposed system can detect differences by referencing the repeated UI sequences.

**Table 5:** Reference data examples of owner-players #1 to #3 in the 3DS MAX experiment

| Player #1 | | |
|---|---|---|
| Count | Percentage | Pattern |
| 42 | 8.4 | P(Target4, 0) B(Dragging) P(Target4, 0) |
| 30 | 6.0 | B(Dragging) P(Target4, 0) B(Dragging) |
| 28 | 5.6 | B(Dragging) P(Target4, 0) B(Dragging) P(Target4, 0) |
| 25 | 5.0 | P(Target4, 0) B(Dragging) P(Target4, 0) B(Dragging) P(Target4, 0) |
| 19 | 3.8 | P(Target4, 0) P(Target4, 0) P(Target4, 0) |
| Player #2 | | |
| Count | Percentage | Pattern |
| 41 | 8.2 | P(Target4, 0) B(0) |
| 35 | 7.0 | K(LMenu) B(2) P(Target4, 0) |
| 27 | 5.4 | P(Target4, 0) K(LMenu) B(2) |
| 22 | 4.4 | K(LControlKey) P(Target4, 0) P(Target4, 0) |
| 14 | 2.8 | K(LMenu) B(2) P(Target4, 0) B(0) |
| Player #3 | | |
| Count | Percentage | Pattern |
| 42 | 8.4 | K(LMenu) B(2) P(Target4, 0) |
| 30 | 6.0 | P(Target4, 0) K(LMenu) B(2) |
| 28 | 5.6 | P(Target4, 0) B(0) |
| 25 | 5.0 | P(Target4, 0) B(Dragging) |
| 19 | 3.8 | K(LMenu) B(2) P(Target4, 0) K(LMenu) B(2) |

Painting and modeling behaviors consist of several mouse and keystroke datasets. Therefore, all the sampling data had approximately 5,000–10,000 KLM operators for each experiment. Each experiment lasted approximately 20–30 min. We used the threshold value by calculating the average of the 70% similarity scores of the test stage of the user reference data against those of the training stage data.

Fig. 7 shows the receiver operating characteristic (ROC) curve results for the three painting experiments. The area under the curve (AUC) value range was between 0.7 0.9. Compared to the Photoshop experiment, the 3DS MAX experiment showed a more robust performance because it was conducted on similar modeling targets. Compared with other sketch-based approaches [27,28] that use a simple static image at the login stage, the proposed method can be applied to any application after login.

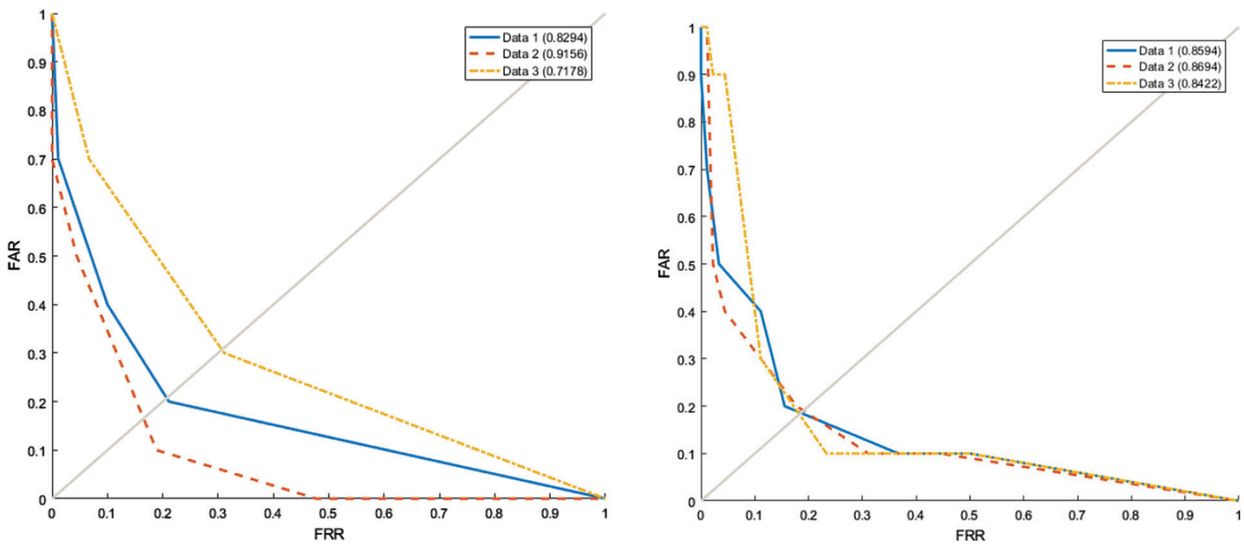



**Figure 7:** ROC curve results of the experiments (left: Photoshop, right: 3D MAX)

Because this experiment was performed on real painting and modeling environments, it was difficult to expect low FAR and FRR values. The results showed that the best performance was achieved using a repeated pattern table ranging from 3- to 6-grams. FRR (10%–20%) and FAR values were produced in these experiments. The results are outlined as follows. As presented in Tabs. 3 and 5, the proposed repeated UI sequences can be adopted as authentication references for behavioral feature authentication. Although painting and modeling are highly complex UI behaviors, Apriori-algorithm-based KLM can detect repeated common UI sequences. According to the results, application adaptability has a significant impact on the outcomes. Keyboard shortcuts influence the behavior of classifying repeated UI sequences. Compared with previous sketching- or painting-based methods that periodically require a specific simple drawing test stage for authentication, the proposed method can be applied to a system backend as a non-intrusive validation process. According to interviews with test participants, they could not recognize any type of authentication delay during their work because the proposed system required minimal system resources for sampling and classification. This non-intrusive process to assists prevent intentional intrusion through authentication during software operations.

## 7 Conclusion

In this paper, we proposed a keyboard and mouse authentication technique based on KLM. Our method detects frequently repeated UI sequences using an Apriori algorithm. Then conducts a sequence similarity comparison was performed for authentication. The practical contributions of this technique were tested and demonstrated. Experiments were performed using a commercial software, and the results showed low FAR and FRR values. Overall, the results indicated that the proposed method generally enhances online privacy. Our experiments demonstrate that the proposed system can authenticate users based on personalized repeated UI sequences. An in-house tool for evaluating application-level log data can be used in our system. The proposed method requires improvement in several improvements. If more recent classifiers, such as deep neural networks, are used from end to end, improved results can be obtained. To enhance the precision of similarity analysis, normalizing the UI sequence using various tests is necessary. For example, various types of paintings can be investigated, a learning dataset for given applications can be developed, and a customized classifier model can be built. Because the repeated UI sequence depends on the content, every threshold value (i.e., the threshold for the similarity score and number of grams for the Apriori algorithm) must be dynamically adjusted to achieve the best performance.

UI sequence data can be content- and hardware-independent. Therefore, the proposed method can be easily applied in various applications. The proposed system consists of four independent modules and can be applied to any application after the user-login stage. Moreover, it complements any type of password-based authentication method under a periodically monitored implementation.

**Funding Statement:** This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF), funded by the Ministry of Education (2021R1I1A3058103, 2019R1A2C1002525).

**Conflicts of Interest:** The authors declare that they have no conflicts of interest to report regarding the present study.

## References

[1] V. M. Patel, R. Chellappa, D. Chandra and B. Barbello, “Continuous user authentication on mobile devices: Recent progress and remaining challenges,” *IEEE Signal Processing Magazine*, vol. 33, no. 4, pp. 49–61, 2016.

[2] S. Mondal, “Continuous user authentication and identification: Combination of security & forensics,” (Ph.D. dissertation), Norwegian University of Science and Technology, Kingdom of Norway, 2016.

[3] P. Baudisch, E. Cutrell and G. Robertson, “High-density cursor: A visualization technique that helps users keep track of fast-moving mouse cursors,” in *Proc. Int. Conf. on Human-Computer Interaction*, Zurich, Switzerland, pp. 236–243, 2003.

[4] J. Chung, S. Hong, Y. Kim, S. J. Kang and C. Kim, “Layout placement optimization methods using repeated user interface sequence patterns for client applications,” *Information Visualization*, vol. 18, no. 3, pp. 357–370, 2019.

[5] I. Traore and A. A. E. Ahmed, *Continuous Authentication Using Biometrics: Data, Models, and Metrics*, IGI Global, 1st ed., Pennsylvania, USA, 2011.

[6] N. Kaur, “A study of biometric identification and verification system,” in *Proc. Int. Conf. on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, Greater Noida, India, pp. 60–64, 2021.

[7] X. R. Zhang, W. F. Zhang, W. Sun, X. M. Sun and S. K. Jha, “A robust 3-D medical watermarking based on wavelet transform for data protection,” *Computer Systems Science & Engineering*, vol. 41, no. 3, pp. 1043–1056, 2022.

[8] X. R. Zhang, X. Sun, X. M. Sun, W. Sun and S. K. Jha, “Robust reversible audio watermarking scheme for telemedicine and privacy protection,” *Computers, Materials & Continua*, vol. 71, no. 2, pp. 3035–3050, 2022.

[9] N. Zheng, A. Paloski and H. Wang, “An efficient user verification system via mouse movements,” in *Proc. ACM Conf. on Computer and Communications Security*, Chicago Illinois, USA, pp. 139–150, 2011.

[10] B. Ayotte, M. K. Banavar, D. Hou and S. Schuckers, “Group leakage overestimates performance: A case study in keystroke dynamics,” in *Proc. of IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, Nashville, TN, USA, pp. 1410–1417, 2021.

[11] I. Traore, I. Woungang, M. S. Obaidat, Y. Nakkabi and I. Lai, "Combining mouse and keystroke dynamics biometrics for risk-based authentication in Web environments," in *Proc. Fourth Int. Conf. on Digital Home*, Guangzhou, China, pp. 138–145, 2012.

[12] H. Gamboa and A. Fred, "A behavioral biometric system based on human computer interaction," in *Proc. Biometric Technology for Human Identification*, Orlando, Florida, USA, vol. 5404, pp. 381–392, 2004.

[13] M. Pusara and C. E. Brodley, "User re-authentication via mouse movements," in *Proc. ACM Workshop on Visualization and Data Mining for Computer Security*, Washington DC, USA, pp. 1–8, 2004.

[14] A. A. E. Ahmed and I. Traore, "A new biometric technology based on mouse dynamics," *IEEE Transactions on Dependable and Secure Computing*, vol. 4, no. 3, pp. 165–179, 2007.

[15] K. Revett, H. Jahankhani, S. T. de Magalhes and H. M. D. Santos, "A survey of user authentication based on mouse dynamics," in *Proc. Int. Conf. on Global e-Security*, London, UK, vol. 12, pp. 210–219, 2008.

[16] P. Bours and C. J. Fullu, "A login system using mouse dynamics," in *Proc. Int. Conf. on Intelligent Information Hiding and Multimedia Signal Processing*, Kyoto, Japan, pp. 1072–1077, 2009.

[17] C. Feher, Y. Elovici, R. Moskovitch, L. Rokach and A. Schclar, "User identity verification via mouse dynamics, information sciences," *Information Sciences: An International Journal*, vol. 201, pp. 19–36, 2012.

[18] F. Bergadano, D. Gunetti and C. Picardi, "User authentication through keystroke dynamics," *ACM Transactions on Information and System Security*, vol. 5, no. 4, pp. 367–397, 2002.

[19] D. Gunetti and C. Picardi, "Keystroke analysis of free text," *ACM Transactions Information and System Security*, vol. 8, no. 3, pp. 312–347, 2005.

[20] M. Curtin, C. C. Tappert, M. Villani, G. Ngo, J. Simone *et al.,* "Keystroke biometric recognition on long text input: A feasibility study," in *Proc. Int. Workshop on Scientific Computing and Computational Statistics*, Hong Kong, pp. C2.1–C2.5, 2006.

[21] J. Ferreira and Santos, "Keystroke dynamics for continuous access control enforcement," in *Proc. Int. Conf. on Cyber-Enabled Distributed Computing and Knowledge Discovery*, Sanya, China, pp. 216–223, 2012.

[22] J. Monaco, N. Bakelman, S. -H. Cha and C. Tappert, "Recent advances in the development of a long-text-input keystroke biometric authentication system for arbitrary text input," in *Proc. European Intelligence and Security Informatics Conf.*, Uppsala, Sweden, pp. 60–66, 2013.

[23] H. Locklear, S. Govindarajan, Z. Sitova, A. Goodkind, D. G. Brizan *et al.,* "Continuous authentication with cognition-centric text production and revision features," in *Proc. IEEE Int. Joint Conf. on Biometrics*, Clearwater, FL, USA, pp. 1–8, 2014.

[24] I. M. Alsaadi, "Study on most popular behavioral biometrics, advantages, disadvantages and recent applications: A review," *International Journal of Scientific & Technology Research*, vol. 10, pp. 15–21, 2021.

[25] R. Dave, N. Seliya, L. Pryor, M. Vanamala and E. Sowells, "Hold on and swipe: A touch-movement based continuous authentication schema based on machine learning," arXiv preprint arXiv:2201.08564, 2022.

[26] I. Stylios, S. Kokolakis, O. Thanou and S. Chatzis, "Behavioral biometrics & continuous user authentication on mobile devices: A survey," *Information Fusion*, vol. 66, pp. 76–99, 2021.

[27] S. Lee, "User behavior of mobile enterprise applications," *KSII Transactions on Internet and Information Systems*, vol. 10, no. 8, pp. 3972–3985, 2016.

[28] G. Zhao, P. Zhang, Y. Shen and X. Jiang, "Passive user authentication utilizing behavioral biometrics for IIoT systems," *IEEE Internet of Things Journal*, 2021.

[29] W. Liu, K. Zheng, B. Wu, C. Wu and X. Niu, "Flow-based anomaly detection using access behavior profiling and time-sequenced relation mining," *KSII Transactions on Internet and Information Systems*, vol. 10, no. 6, pp. 2781–2800, 2016.

[30] K. Kounlaxay, Y. Shim, S. Kang, H. Kwak and S. K. Kim, "Learning media on mathematical education based on augmented reality," *KSII Transactions on Internet and Information Systems*, vol. 15, no. 3, pp. 1015–1029, 2021.

[31] X. Duan, S. Kang, J. I. Choi and S. K. Kim, "Mixed reality system for virtual chemistry lab," *KSII Transactions on Internet and Information Systems*, vol. 14, no. 4, pp. 1673–1688, 2020.

[32] S. Kang and T. Hahn, "Eyeglass remover network based on a synthetic image dataset," *KSII Transactions on Internet and Information Systems*, vol. 15, no. 4, pp. 1486–1501, 2021.

[33] S. K. Card, T. P. Moran and A. Newell, *The Psychology of Human-Computer Interaction*, 1st ed., Boca Raton, USA: CRC Press, 1983.

[34] P. M. Fitts, "The information capacity of the human motor system in controlling the amplitude of movement," *Journal of Experimental Psychology*, vol. 47, no. 6, pp. 381–391, 1953.

[35] M. Hegland, "The apriori algorithm–A tutorial," in *Mathematics and Computation in Imaging Science and Information Processing*, World Scientific Publishing, Singapore, pp. 209–262, 2007.

[36] D. Gusfield, *Algorithms on Strings, Trees, and Sequences*, Cambridge, UK: Cambridge University Press, 1997.

[37] Y. B. Kim, S. J. Kang, S. H. Lee, J. Y. Jung, H. R. Kam *et al.,* "Efficiently detecting outlying behavior in video-game players," *PeerJ*, 2015. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4690374/.