Bulletin of Electrical Engineering and Informatics
Vol. 10, No. 3, June 2021, pp. 1546∼1557
ISSN: 2302-9285, DOI: 10.11591/eei.v10i3.2565

Examining a display-peeping prevention method that uses real-time UI part transparency linked to motion detection by video analysis

Koki Kishibata, Masaki Narita
Faculty of Software and Information Science, Iwate Prefectural University, Takizawa, Japan

Article Info

Article history:

Received Apr 23, 2020
Revised Dec 29, 2020
Accepted Apr 2, 2021

Keywords:

Motion analysis
Peep prevention
Privacy protection
Shoulder hacking
UI transparency

ABSTRACT

In recent years, the use of various information terminals such as smartphones and personal computers have become widespread, and situations where information terminals are used have become diverse. With increased opportunities to use information terminals outdoors and during travel, some users have been using peep-prevention filters, or software with an equivalent function, on their displays, in order to protect their privacy. However, such filters have problems with regards their effectiveness, ease of use, and the user being able recognize when they are vulnerable to peeping. Decrease in display visibility, unprotected angles, and the fact that it is difficult for users to notice when others are watching their screen, are some examples of such problems. Also, recently, many information terminals recently distributed have built-in cameras. In this paper, in order to solve the aforementioned problems, we propose to detect motion, video analyze , and transparentize part of the user interface (UI) in real time by using a laptop's built-in camera. This method is enabled with low-load and can be applied to various terminals. Further, in order to verify the effectiveness of the method, we implemented a prototype, and carried out an evaluation experiment on experimental subjects. Results from the experiment confirmed that real-time UI transparentization is a very effective method for protecting privacy of information terminals.

Corresponding Author:

Masaki Narita
Faculty of Software and Information Science
Iwate Prefectural University
152-52 Sugo, Takizawa, Iwate 020-0693, Japan
Email: narita_m@iwate-pu.ac.jp

1. INTRODUCTION

It is difficult for information terminal users to instantaneously protect their information from unexpected shoulder hacking [1]. Meanwhile, due to the rapid spreading of various information terminals (e.g. smartphones, tablets, laptops), as well as the informatization of society, "peeping" is becoming an increasing threat [2], [3]. As displays are indispensable for a user, some users have adopted countermeasures of using commercially available peep-prevention filters, or using software with similar functions.

However, many of these filters change the light outputted from the display, causing a decrease in usability and readability. Therefore, at present, a trade-off generally exists between the protection method's performance and maintaining usability. Another feature of these filters is that the more a "peeper" approaches an angle perpendicular to the screen, the more difficult it is to defend against peeping. There is also the issue

where, if important information is leaked because of peeping, it is difficult for the user to recognize that they have been the victim to peeping. Thus, at present, one cannot expect to prevent the occurrence of peeping in advance, and there exists the possibility of individuals and organizations incurring immense damage. In this study, we examine a prevention and detection method against peeping which enables maintenance of both protection function and usability by linking motion detection with real-time transparentizing of user interface (UI).

2. DAMAGE RESULTING FROM PEEPING, AND EXISTING PROTECTION METHODS

Peeping can be categorized into two types: i) malicious, intentional occurrence, ii) accidental occurrence. Also, damages resulting from peeping may cause serious damages to individuals and organizations [4]. Here, peeping refers to a third party other than the user recognizing readable information on a user's display. This peeping occurs, firstly, from the user's creating a situation where third parties may able to read out information on the user's display.

In the following sections, we first discuss types of information that could lead to incurrence of damages. Next, we explain situations and damages relating to information leakage caused by peeping. Finally, we discuss peep-protection filters, the conventional defense method.

2.1. Information that could lead to damage

Information that could be leaked because of an act of peeping includes all information that is visible on the display. Owners of such information can broadly be categorized to individuals or other groups. Regarding the former, personal information is a typical example. According to the personal information protection commission's guideline on the personal information protection Act [5] in USA, personal information is defined as an information that enables identification of an individual, through referencing single or multiple pieces of information, including first and last name, date of birth, contact information, and affiliation. Peeping into personal information is a violation against individual's privacy. Further, malicious use of such information could possibly lead to the illegal theft of individual property. This information generally visible on the display is in text-form. Also, with account-based internet services becoming more widespread, occasions where IDs and passwords are used have increased.

There are many that allow manipulation of individual's assets such as internet banking accounts. Furthermore, display screens of social networking services include information on conversation and comments and it is known that individuals may be identified from such information [6]-[8]. Images such as maps may also be shown on display. Various situations where maps are used by users are thought to be diverse, and it can easily be assumed that locations accessed by users are where they have an interest in. These locations might be, for instance, the user's home, the surrounding area of their home, the user's destination, or the route to the destination. Leakage of such information could be used for stalking [9], [10].

For the latter, many types of information exist. Here, we will discuss confidential information that organizations own. Corporations typically have information that should not be disclosed outside of management, and this information is important for competition with rival corporations. However, as with the case of individuals, when this information appears as text on the display, it is vulnerable to peeping. Generally, these information includes design drawings or process charts which are shown as images on the display.

2.2. Situations where peeping occurs

Regarding peeping, there always exists the peeper and the person who is subject to peeping. In order for peeping to occur, firstly, a situation of possible peeping must be created by the person subject to peeping. There then must be a situation where intentional or accidental action is made by the peeper. One of the most common situations where peeping occurs is the daily use of information terminals. For example, attention of a user working at home with a tablet terminal is turned to information on the display screen. Such information includes information of individuals and groups needed for work. If there are subjects difficult to pay attention to, such as family members of the user, there is risk that information visible on the tablet display might be peeped in. At this point, we will not consider whether or not such information obtained by one who peeped in will be leaked. However, possibility will exist where information will be passed into the hands of an unauthorized third party, causing damage to the owner of information. In addition, risk can be said to exist mainly in information owned by individuals when information terminals are used on a daily basis.

A situation commonly observed in recent years is the use of information terminals on the move [11], [12]. Situations include the use of trains and buses, as well as use of smartphones while walking, which has become a problem in recent years. Peeping on others' in the train is thought to occur on a regular basis; at the same time, there are many cases where the act of peeping is accidental, and the peeper does not have the special intention. From the perspective of information protection, this is quite a serious threat.

As mentioned above, in the discussion of everyday use of information terminals, the possibility of a third party obtaining information without authorization is a problem [13]. Regarding situations on the bus, standing spaces within the bus is almost the same as is in a train. When using seats facing the travelling direction, there is a risk of intentional peeping. Although this will not be covered in this paper, while the user is outdoors, or on the move, there may be situations where a video camera is used to indirectly peep at others' devices [14], [15].

2.3. Current protection methods against peeping

Many products or software exist for peep-prevention filters. This is currently the most common protection method against peeping. Some filters are sold by personal computer manufacturers; there are many other types of filters in the market, designed for each terminals, and categorized as accessory items of information terminals [16], [17]. Filters include ones that manipulate light quantity, change colors, and manipulate angle-limits of light. Software with the same functions are also recognized [18], [19]. The mechanism of such software is similar to that of physical filters; manipulating information such as display brightness or luminance of the display, or utilizing a single semitransparent image to realize the same functions. Because peep-prevention filters add treatment to the light emitted by the devices, many of the currently existing types reduce usability of information terminals used by the users. Also, while these existing filters are capable of protecting peeping from the side, their effectiveness is low when peeped behind the user. Also, because these filters' focus is on protecting the information on the display when peeped in, it is difficult for users to recognize the damage incurred from it. These are some of the problems that peep-protection filters have.

Regarding protection methods other than peep-protection filters, there is also software that works in accordance with a user's actions and hides the information that appears [20]. This is done by detecting a particular predefined action of the user with a sensor, and making the screen invisible. This protection method is effective for protecting information when the user's attention is directed other than the display. However it is merely a simplification of a series of steps, such as putting the user's information terminal into sleep mode. In other words, it is not an effective against unexpected peeping.

3. PROPOSED METHOD

As mentioned in the previous section, much information exists that relates to the user's properties which could be subject to peeping and there exists a wide range of situations for potential damage. Meanwhile, current protection methods have problems of usability, effectiveness, and user's ability to recognize the damage.

In the proposed method, motion detection with respect to video captured from a camera directed at the user, and permeabilization of user interphase is used. First, frame images are obtained from the camera. Next, the frame images are preprocessed for conducting image analysis. Preprocesses include treatments such as noise removal or gray-scale conversion, which need to take place before peep detection. Next, the preprocessed images are used to judge whether or not peeping is occurring. If peeping is judged to be occurring, the UI is transparentized. This is the outline of the proposed method as shown in Figure 1.

The method allows for maintaining usability, increasing effectiveness against unexpected peeping, and notifying the user when peeping is occurring. When peeping occurs, application which the method has been applied or the UI display of the display is not affected at all; therefore, the decline in usability often observed in peep-prevention filers does not occur. The only case where a decline in usability could occur is when the user interphase is transparentized upon detection of peeping. Such will not be a problem as transparency is carried out to protect information on the display necessary upon peeping. Also, the range of protection capability in the method, is equal to that capturable by the camera.

As mentioned before, peep-prevention filters have low effectiveness when peeped behind the user. However, because in recent years, many cameras built into information terminals are pointed at the user, the area surrounding the user becomes the range of protection. The space behind the user is one location where it is easy to obtain information through peeping, and thus, the method that protects against peeping in this area

can be considered to be a highly effective method. Also, regarding this method's real-time UI transparentization process, whenever the user interface becomes invisible, either momentarily or for an extended period of time, the state of invisibility can be equated to the occurrence of peeping. Thus, a shift to invisibility using transparentization also serves to notify the user of the occurrence of peeping.

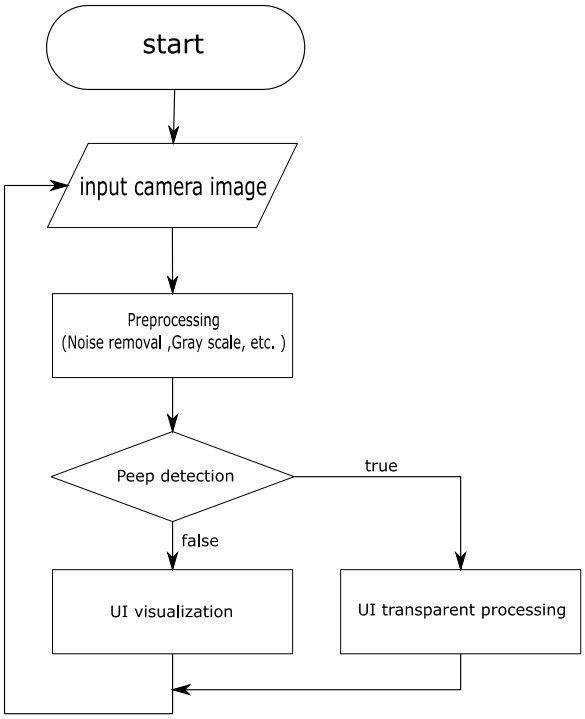

Figure 1. A summary diagram of proposed method

3.1. Peep detection via analysis of camera images

It is common for recent information terminals to have built-in cameras. The performance of built-in cameras has improved along with the improvement of the hardware performance of the information terminals. In many cases, these cameras are used when the user intentionally uses a particular function, and it is rare that they are used for software control. In the proposed method, we use the camera as a sensing function for detecting peeping. In this method, by conducting motion detection with respect to image-information obtained from the camera, and also using a learned mask, the environment as well as user characteristics is excluded to detect peeping. Also, with cameras built into terminals are generally directed at the user, the range of detection is the vicinity of the user. This means protection in regions where the peeping-side is most able to obtain information and where protection is difficult with existing peep-prevention filters. Because of this, the method is considered highly effective. The condition of a camera applicable to the method is to be able to capture the situation surrounding other than the user. The wider the angle of view or the range of the camera, the greater the range of peep detection will be.

3.1.1. Preprocessing for analysis

As a preprocess for peep detection using frame images obtained from the camera, gray-scale conversion and image noise removal are performed. Median filter is applied to noise removal. These processes are performed to improve the detection-accuracy of peep-detection, which is a process that is taken later.

3.1.2. Motion detection

Motion detection is used to detect peeping in the proposed method. Act of peeping have been defined often in the past, yet, detailed movements of people for such have not been defined. In the method, we set the definition of peeping as all moving subjects behind the user other than the environment. In the method, peep detection is conducted by using motion detection, as well as learning values (described later), in order to detect the intrusion or the existence of objects other than the background environment obtained by

$$n_{now} = (1 - w) \cdot n_{cum} + w \cdot f. \quad (1)$$

In the motion detection employed in the method, a weighted average (n_now) is calculated by the above equation, using the value (f) of the preprocessed frame image section 3.1.1, as well as the cumulative value (n_cum) from the previous frames. The calculated value (n_now) is treated as the cumulative value (n_cum) when processing the next frame. w in the equation above is an arbitrary weight used in calculation, and is determined separately for each system.

3.1.3. Learning the pixel characteristics of the user and environment

As mentioned in section 3.1.2, motion detection alone is not enough for establishing peep detection. Thus, we conducted peep-detection by setting up a method where image-information on moving bodies in the environment is learned in advance, and this information is used to create a mask image to detect peeping by

$$l_{new} = max(l_{now}, n_{cum}). \quad (2)$$

In this learning processing, the weighted average value (l_now) of the frame images from a state where peeping has not yet occurred is compared with the previous maximum weighted average value (n_cum). The larger of the two is then retained as the new maximum weighted average value (l_new). This retained learning value is used as a mask in the peep detection process described later.

In feature learning, it is possible to carry out learning that includes the user's actions, however, if a wide range of user actions are learned, this creates areas of exception, where it is difficult to detect peeping. Our learning process aims to improve accuracy of judgment by excluding the pixel-related characteristics of the user's behavior when they are using the display information, as well as of the environment. This feature-learning process serves to measure in advance feature values of the user and environment that are captured by camera. This learning processing must be carried out in advance.

3.1.4. Peep detection

Peep detection is conducted by using, in the form of a mask image, the results from motion detection section 3.1.2, and the learning values obtained from feature learning. For peek detection, a predetermined and arbitrary judgment-threshold value is needed. The judgment threshold is determined based on camera specifications, required judgment accuracy, and frame rate. Users' behavior that could not be learned during the learning process section 3.1.3, influences the judgment value, and so there is also the need to determine the degree of tolerance for non-learned values of the user.

When there are two states of peeping, "peeping is occurring" and "peeping is not occurring," one peep threshold is set. If there are three states of peeping, " peeping is occurring," "undefined," and "peeping is not occurring," two peep thresholds are set. Whether to use an "undefined" region depends on the system that is actually applied; doing so enables finer judgment, which we think will be useful for maintaining usability to greater degree, and for realizing increased accuracy in information protection.

Regarding the process of detection, this is done by subtracting the learning values (obtained from the learning process of section 3.1.3, from the motion-detection result section 3.1.2, rounding this number down to the nearest number whole, taking the sum of the array-element values to be the judgment value, and then comparing this value against the threshold values. If there are two states of peeping, a value above the threshold signifies detection, and all other values signify non-detection. While incapable of realizing complex judgments, the detection algorithm above is one that enables peep detection to be low-load.

3.2. Transparentization of the user interface

Regions that are transparentized on the display become invisible. In our method, we link this general effect with detection in order to protect information from peeping, and also notify users when they are vulnerable to peeping. Part or all of the user interface is subjected to transparentization. The scope of this is determined by system specifications as well as the range requiring protection.

To transparentize something generally means to convert it to a see-through state. In our method, transparentization is a process where the subjected information is made invisible (the display background is shown instead), and the information's existence is thus hidden from the peeper. Even in cases where the

peeper knows about the existence of such information, transparentization of the user interface – linked with a peep detection process section 3.1, – protects the user against intentional peeping. The treatment of the user interphase is determined by the result of peep detection section 3.1.4. For example, if there are two peep states, "peeping is occurring" and "peeing is not occurring," then the following two treatments will exist, respectively: "transparentization of the user interface" and "standard view of the user interface." If "undefined" is added to two states mentioned above, a treatment called "no change" will be added to treatments to select from. When judgment region is in an "undefined" state, this means that after transparentization is performed, due to peeping, a judgment that peeping is completely finished, and the consequent redisplaying action has not occurred yet. Adding this region enables information protection that is highly accurate.

While our method aims to prevent peeping, carrying out transparentization also serves the role of notifying the user when peeping occurs. Generally, the user's awareness and actions are greatly important for maintaining information security. And notifying the user of harm or vulnerability is one effective method relating to this. Peeping notification refers to when the user is notified of another person's attempt to peep on their device. Such notification allows the recipient user to enact countermeasures. Recognition of harm or vulnerability is also important for preventing further damage once the information is leaked.

4. PERFORMANCE EVALUATION

We conducted a performance evaluation in order to confirm whether our proposed method section 3, was effective for peep prevention. In this performance evaluation, we conducted an evaluation experiment by creating a prototype in order to apply our proposed method.

4.1. Prototype

We developed a prototype for the evaluation experiment. We used Python 3.6 for the development language, Tkinter and PIL as libraries for GUI display, and OpenCV and NumPy for acquiring and processing the video used. We created three types of windows: "motion detection results," "transparentization subject," and "learning settings." In the window for motion detection results as shown in Figure 2, a motion-detection result image was drawn in real time. For the "transparentization subject" window, we used the character-string spelling "string" displayed inside. Redrawing of motion detection results, peep detection, and transparentization took place every 50 milliseconds. In the "learning settings" window, two checkboxes and one text box were placed. One checkbox was created for switching between output/nonoutput of internal processing results, and another for switching between learning mode and detection mode. The text box was created for inputting judgment threshold values.

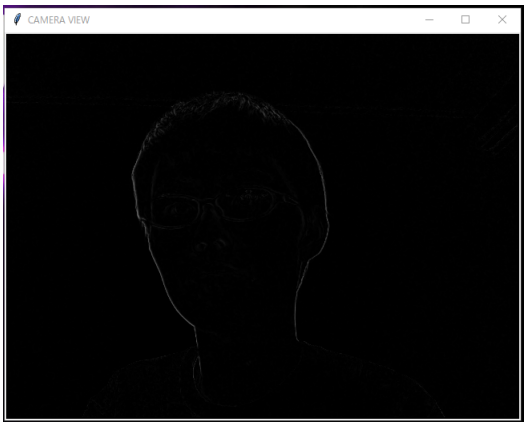

Figure 2. Motion detection for peep detection

4.1.1. The subject of transparentization

The subject of transparentization was a label with the character string spelling "string" placed in the aforementioned "transparentization subject" window. In this evaluation of our propose method, text color was not considered. Having the character string disappear, making the background visible, was here considered to be synonymous with having the character string transparentized. Regarding transparentization treatment,

we realized transparentization by changing the reference of the character-string label's text to blank character variables.

Figures 3 and 4 show the window images for before and after transparentization, respectively. Regarding the user display, a by-the-book transparentization process would involve changing the alpha value to 0. However, this method is partially dependent on GUI and API, so we implemented our prototype using variable-switching. Transparentization of the text label, in addition to protecting the text information in the label, also notifies the user that peeping has been detected.

Figure 3. Transparentization off

Figure 4. Transparentization on

4.1.2. Learning and peep detection

The learning process and peep detection process were implemented using the algorithm described in section 3. In the detection process for this prototype, the state of peeping was defined as either "peeping occurring" or "peeping not occurring," with a single threshold value being used to determine which of the two states is occurring. The initial value of the threshold was set as 10000.

4.2. Evaluation experiment

Using the created prototype section 4.1, an evaluation experiment was conducted on 10 experimental subjects, all male. Regarding operating environment, Windows 10 was used as the OS for the prototype. The camera was built into the upper section of the display, with HD resolution (1280×720), and a diagonal angle of view of 74.6 degrees. For all experimental subjects, the experiment took place in a seat in the back row of the university lecture room (measurement location 1) or in a student chair in the university laboratory (measurement location 2). During the experiment, no items were moved, and the environment was set up so that instruments and so forth used daily were also captured by the camera.

Figure 5 shows the method of our evaluation experiment. Experimental subjects were paired for this experiment, with one subject acting as the user and the other acting as the "peeper." We conducted the experiment in three broad categories. First, we asked the user to minimize their window after being notified of detection; we inspected whether the peeper could recognize the word "string" displayed on the screen. The peeper would peep while moving from the user's right to left side. If the peeper could recognize the text, we asked them how often the text was displayed. Next, we inspected the case where the user did not take any countermeasure against peeping (i.e. did not minimize their window), interviewing the peeper in a similar manner as before. Finally, we allowed the peeper to stop at a location of their choosing during the experiment, and interviewed the peeper as part of our inspection. For in all of these categories, we also carried out experiments by broadly changing their respective threshold values. Learning was carried out before the inspection, and the arbitrary features of each subject while they work was learned. Peeping-judgment values were recorded before and after learning, and the learning values were recorded after learning.

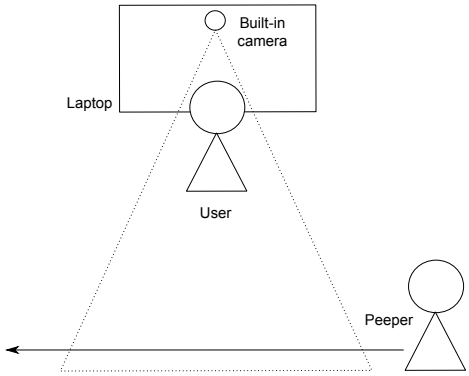

Figure 5. Evaluation experiment method

4.3. Experimental results

In this section we share the results from the evaluation experiment. Regarding the prototype's output, in the case of the detection judgment value, it was the average value of 100 judgment processes, and in the case of the learning value, it was the total value of elements in the grayscale frame-image array. The average of judgment values in the measurement before learning was 127724.2 (maximum value: 358558, minimum value: 65590). The average of judgment values after learning was 96.7 (maximum value: 256, minimum value: 1). The average of learning values was 5356014.8 (maximum value: 12584240, minimum value: 941732). The judgment values before learning differed based on measurement location: the average for Measurement location 1 was 117889.5, whereas the average for Measurement location 2 was 167063.0. These results indicate that the original feature values are different due to the different environments in each measurement location. After learning, differences in values based on measurement location were small, demonstrating that the differences in environment were absorbed through the learning process. The learning values differed greatly depending on the subject, with differences in each subject's everyday style of use showing significantly, rather than differences that were based on the environment.

4.3.1. Peep prevention taking into account peeping countermeasures by the user

By inspecting the results for when the subjects enacted a countermeasure against peeping, we evaluated comprehensive prevention/protection capability that includes countermeasures taken by the user. Figure 6 shows the accuracy of peep prevention when the user enacted a countermeasure against peeping. The chart shows the average value for all subjects, by threshold value. Cases where the peeper could not recognize the text are set as 1, and cases where the peeper could recognize the text (even if slightly) are set as 0, and the average value for all subjects is shown. At the threshold value of 10000, peep prevention was successful for all subjects. That is to say, at the threshold of 10000, peep-detection and UI transparentization showed effectiveness in preventing peeping, and notifying the user of the occurrence of peeping.

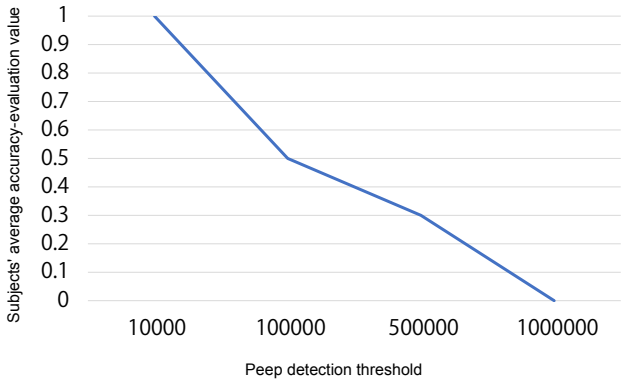

Figure 6. Accuracy of peep prevention with user countermeasure

Results also showed that detection accuracy was very high for the initial stage of intrusion, which is when the peeper intrudes into the detection space from out of range. On the other hand, failures in detection could be observed when using a threshold value of 100000. There were instances where the peeper could recognize the text before the user minimized their window. This is because the higher threshold value allowed for the system to be more accepting of movements made by the peeper as well.

4.3.2. Evaluation of peep prevention without countermeasures by the user

In section 4.3.1, we evaluated the user and system's protection capability against peeping for a scenario of peep prevention where a countermeasure by the user was taken into account. However, it is conceivable that there are situations where the user cannot be expected to enact countermeasures. In this section, we use peeping experiment results to evaluate the system's protection capability for cases where the user is not able to enact a countermeasure.

Figure 7 shows the accuracy of peep prevention in the case where the user does not enact a countermeasure against peeping. Cases where the peeper could not recognize the text are set as 1, cases where the peeper could recognize the text (even if slightly) are set as 0, and cases where the text was displayed for a short period of time were set as 0.5, and an evaluation was performed. Cases where the text was displayed for a short period of time, are situations where the characters in the text were not recognizable, but the fact that the text was displayed was recognized. Figure 7 shows, for each threshold value, the average value for the results for all experimental subjects. At a threshold of 10000, cases were observed where the text was seen momentarily, but it was difficult to decipher the text. For this threshold of 10000, the location where the text was temporarily shown was the same for all subjects. This location was behind the user on the side opposite (from the user's perspective) to the side the peeper enters from. From this location, feature learning was conducted with regard to the users, and this is the range where, in our method's motion-detection algorithm, the smallest judgment values can be outputted by calculation. Therefore, experiments with even smaller threshold values will be required in the future. For the majority of locations behind the user, our system was successful in preventing peeping, showing particular effectiveness for angles that current peep-prevention filters have difficulty protecting.

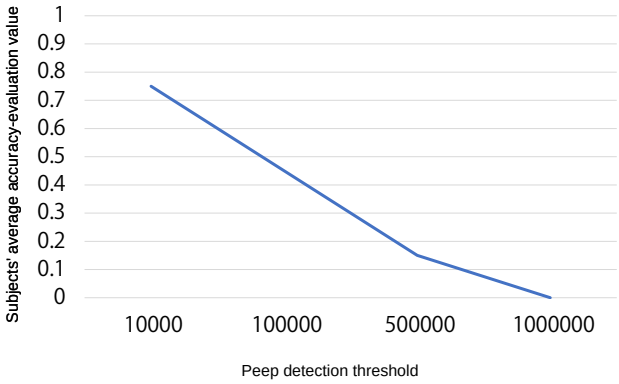

Figure 7. Accuracy of peep prevention with no user countermeasure

4.3.3. The "peeper" stopping

Our proposed method is one where motion detection is used to detect peeping. Our evaluation discussed in this section was based on the assumption that our proposed method would not be able to handle a situation where the peeper stops. In an evaluation where a person stops, movement will always have occurred before they stop, which means peep detection and prevention will have been activated once. Our evaluation's purpose is to evaluate the range of peeping that our proposed method is capable of responding to.

Figure 8 shows peep-prevention accuracy by threshold value, for our peep-prevention experiment where the peeper was allowed to stop. Cases where peeping was successfully prevented even when the peeper stopped are set as 1, cases where flickering, occurred when the peeper stopped are set as 0.5, and cases where the text was fully recognizable when the peeper stopped are set as 0. Based on this classification, the chart shows the average values for all experimental subjects. Regarding the initial value, 10000, of the previous cases where the peeper did not stop, when we had the peeper stop, the text was either displayed or was flickering.

Our proposed method did not show high effectiveness for a scenario where the peeper stops. However, considering that when compared to the higher thresholds, improvement in protection performance was observed for lower thresholds, we predict that lowering the judgment threshold would enable our method to flexibly handle scenarios where the user stops as well.

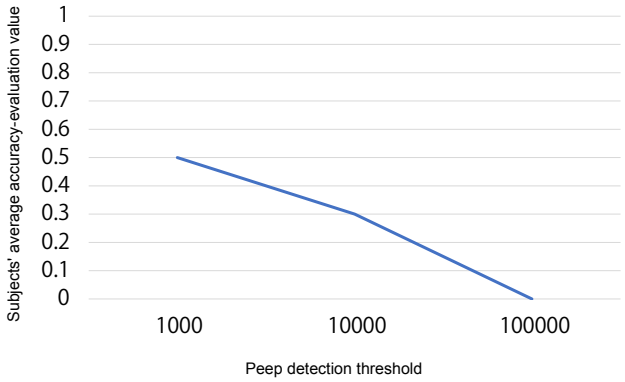

Figure 8. Accuracy of peep prevention when the "peeper" is allowed to stop

5. FUTURE ISSUES

The number of subjects in the performance evaluation experiments in section 4 was not a sufficient amount for a performance evaluation experiment. In order to show our method's effectiveness with greater reliability, experimentation with more subjects is required. However, we were able to confirm through our performance evaluation that linking UI transparentization with peep detection was an effective method for protecting information on the display of an information terminal. Also, in our discussion of evaluation-experiment results in section 4.3.1, we mentioned that our proposed method's detection accuracy was very high for the initial stage of intrusion, in scenarios where the peeper intrudes from out of range. This means that maintaining UI transparentization for a long period after initial detection would enable peep-prevention with very high accuracy.

However, our method's effectiveness is greatly affected by angle of view, quantity, and capturing method of cameras. Also, our method is not able to take into account the regions that are not covered by camera. Regions outside the camera's scope are regions where current peep-prevention filters show high effectiveness, and therefore, in order to prevent peeping with greater accuracy, currently existing methods would need to be combined with our proposed method.

Also, in order to improve peep-prevention accuracy, both the learning method and detecting method require improvement. One problem with the detection method in our proposed method, is that it detects image changes other than peeping to also be occurrences of peeping. This problem is caused by the characteristics of the motion detection algorithm used. In order to be able to respond to users' diverse styles of use, our method would need to be able to handle changes in image features that occur from causes other than peeping [21]-[23]. Further, our proposed method is a very simple detection method, because one of its aims was to achieve low-load. By improving the learning and detection methods, and thus obtaining more detailed detection results, it is possible to carry out transparentization with higher accuracy. Generally, a trade-off exists between load and accuracy, with both being important elements. However, it can be said that peep detection of higher accuracy would enable the prevention of peeping, and the realization of this would contribute to increasing the range of available protection methods for information security. Also, a method for automatic determination of the threshold would be needed in order to apply this method at a lower cost; this is one improvement we are planning for the future.

In the paragraph above, we discussed the improvement of peep-prevention accuracy. For ease of implementation and efficient operation, improvement of the transparentization method will also be required. The functions realized by our proposed method have a one-to-one relationship with software, which means that if our method is applied to multiple software systems, it needs to be implemented individually to each software system, which would generate a high load. For software systems that use the same judgment thresholds, it

would be ideal to be able to perform detection and transparentization through one process where our method is applied. This would involve the graphics API specifications of the OS [24]-[26], as well as an issue of permissions given to the process. Proposing a transparentization method that resolves these issues will be important in order to generalize transparentization as a peep prevention measure. Transparentization for peep prevention that is offered as part of the functions of the OS is one example of a possible transparentization method for accomplishing the above.

6. CONCLUSION

In this paper, we examined a peep-prevention method that uses UI transparentization linked to peep-detection using camera images. In our proposed method, peeping is detected using motion detection as well as feature learning, and a part or all of the user interface is transparentized in order to protect information, and also notify the user of the occurrence of peeping. Based on this method, we created a prototype and conducted an evaluation experiment. In a peep-prevention evaluation experiment that involved the user enacting a countermeasure, for the case of many of the experimental subjects, we were able to confirm efficacy of our method for protecting information and notifying the user of the occurrence of peeping. Improvements in detection method, as well as the method of transparentization will be required in the future.

REFERENCES

[1] M. Eiband, M. Khamis, E. V. Zezschwitz, H. Hussmann, and F. Alt, "Understanding Shoulder Surfing in the Wild: Stories from Users and Observers," in *Proc. 2017 CHI Conf. Human Factors in Computing Systems (CHI '17)*, 2017, pp. 4254–4265, doi: 10.1145/3025453.3025636.

[2] D. S. Thosar, and M. Singh, "A Review on Advanced Graphical Authentication to Resist Shoulder Surfing Attack," in *Proc. Int. Conf. Advanced Computation and Telecommunication (ICACAT)*, 2018, pp. 1-3, doi: 10.1109/ICACAT.2018.8933699.

[3] R. Kühn, M. Korzetz, and T. Schlegel, "User Strategies for Mobile Device-Based Interactions to Prevent Shoulder Surfing," in *Proc. 18th Int. Conf. Mobile and Ubiquitous Multimedia (MUM '19)*, no. 43, 2019, pp. 1–5, doi: 10.1145/3365610.3368412.

[4] P. Barker, "Visual Hacking-Why it Matters and How to Prevent it," *J. Network Security*, vol. 2019, no. 7, pp. 14-17, 2019, doi: 10.1016/S1353-4858(19)30085-6.

[5] California Consumer Privacy Act (CCPA), Accessed: Apr. 18, 2020. [online]. Available: https://oag.ca.gov/privacy/ccpa.

[6] Y. Jeong, and Y. Kim, "Privacy Concerns on Social Networking Sites: Interplay among Posting Types, Content, and Audiences," *J. Computers in Human Behavior*, vol. 69, pp. 302-310, 2017, doi: 10.1016/j.chb.2016.12.042.

[7] S. Guo, and K. Chen, "Mining Privacy Settings to Find Optimal Privacy-Utility Tradeoffs for Social Network Services," in *Proc. 2012 Int. Conf. Privacy, Security, Risk and Trust and 2012 International Conf. Social Computing*, 2012, pp. 656-665, doi: 10.1109/SocialCom-PASSAT.2012.22.

[8] S. Machida, S. Shimada and I. Ecizen, "Settings of Access Control by Detecting Privacy Leaks in SNS," in *Proc. 2013 Int. Conf. Signal-Image Technology and internet-Based Systems*, 2013, pp. 660-666, doi: 10.1109/SITIS.2013.108.

[9] M. Tschersich, and M. Niekamp, "Pros and Cons of Privacy by Default: Investigating the Impact on Users and Providers of Social Network Sites," in *Proc. 48th Hawaii Int. Conf. System Sciences*, 2015, pp. 1750-1756, doi: 10.1109/HICSS.2015.211.

[10] C. Hallam, and G. Zanella, "Online Self-Disclosure: The Privacy Paradox Explained as a Temporally Discounted Balance Between Concerns and Rewards," *J. Computers in Human Behavior*, vol. 68, pp. 217-227, 2017, doi: 10.1016/j.chb.2016.11.033.

[11] S. Trewin, C. Swart, L. Koved and K. Singh, "Perceptions of Risk in Mobile Transaction," in *Proc. 2016 IEEE Security and Privacy Workshops (SPW)*, 2016, pp. 214-223, doi: 10.1109/SPW.2016.37.

[12] F. Maggi, A. Volpatto, S. Gasparini, G. Boracchi, and S. Zanero, "Poster: Fast, Automatic iPhone Shoulder Surfing," in *Proc. 18th ACM Conf. Computer and Communications Security (CCS '11)*, 2011, pp. 805-808, doi: 10.1145/2046707.2093498.

[13] S. Tabrez, and D. J. Sai, "Pass-Matrix Authentication a Solution to Shoulder Surfing Attacks with the Assistance of Graphical Password Authentication System," in *Proc. 2017 Int. Conf. Intelligent Computing and Control Systems (ICICCS)*, 2017, pp. 776-781, doi: 10.1109/ICCONS.2017.8250568.

[14] T. Takada, "FakePointer: An Authentication Scheme for Improving Security against Peeping Attacks Using Video Cameras," in *Proc. 2nd Int. Conf. Mobile Ubiquitous Computing, Systems, Services and Technologies*, 2008, pp. 395-400, doi: 10.1109/UBICOMM.2008.76.

[15] X. Shi, and J. Gu, "An Authentication Method Resistant to Video-Recording Attacks," in *Proc. 2nd International Conf. Computer Science and Network Technology (ICCSNT)*, 2012, pp. 1967-1972, doi: 10.1109/ICCSNT.2012.6526304
[16] Belkin, SCREENFORCE InvisiGlass Ultra Privacy Screen Protector, Accessed: Apr. 18, 2020. [online]. Available: https://www.belkin.com/us/p/P-F8W924/.
[17] 3M, Privacy and Screen Protectors, Accessed: Apr. 19, 2020. [online]. Available: https://www.3m.com/3M/en_US/privacy-screen-protectors-us/.
[18] MortisApps, Screen Guard Privacy Screen, Accessed: Apr. 18, 2020. [online]. Available: https://play.google.com/store/apps/details?id=com.mortisapps.privacyfilter&hl=en.
[19] Hardy-Infinity, Privacy Filter, Accessed: Dec. 30, 2020. [online]. Available: https://apps.hardyinfinity.com/en/.
[20] Huey Soft, Privacy Screen Filter, Accessed: Apr. 18, 2020. [online]. Available: https://play.google.com/store/apps/details?id=com.hueysl.privacyscreen&hl=en.
[21] S. Ding, S. Qu, Y. Xi, A. K. Sangaiah, S. Wan, "Image Caption Generation with High-Level Image Features," *Pattern Recognition Letters*, vol. 123, pp. 89-95, 2019, doi: 10.1016/j.patrec.2019.03.021.
[22] A. Gordo, J. Almazan, J. Revaud, and D. Larlus, "End-to-End Learning of Deep Visual Representations for Image Retrieval," *Int. J. Computer Vision*, vol. 124, no. 2, pp. 237-254, 2017, doi: 10.1007/s11263-017-1016-8.
[23] J. Yan, S. Li, K. Liu, R. Zhou, W. Zhang, Z. Hao, X. Li, D. Wang, Q. Li, and X. Zeng, "An Image Features Assisted Line Selection Method in Laser-Induced Breakdown Spectroscopy," *Analytica Chimica Acta*, vol. 1111, pp. 139-146, 2020, doi: 10.1016/j.aca.2020.03.030.
[24] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," in *Proc. 2014 IEEE Conf. Computer Vision and Pattern Recognition (CVPR '14)*, 2014, pp. 580-587, doi: 10.1145/3106237.3106298.
[25] T. Su, G. Meng, Y. Chen, K. Wu, W. Yang, Y. Yao, G. Pu, Y. Liu, and Z. Su, "Guided, Stochastic Model-Based GUI Testing of Android Apps," In *Proc. 11th Joint Meeting on Foundations of Software Engineering*, 2017, pp. 245-256, doi: 10.1145/3106237.3106298.
[26] T. D. White, G. Fraser, and G. J. Brown, "Improving Random GUI Testing with Image-Based Widget Detection," in *Proc. 28th ACM SIGSOFT Int. Symposium on Software Testing and Analysis (ISSTA '19)*, 2019, pp. 307-317, doi: 10.1145/3293882.3330551.