Kitani *et al. Hum. Cent. Comput. Inf. Sci.* (2017) 7:18
DOI 10.1186/s13673-017-0099-0

Human-centric Computing and Information Sciences

RESEARCH Open Access

Toward the reduction of incorrect drawn ink retrieval

Atsushi Kitani[1*†], Taketo Kimura[2†] and Takako Nakatani[3†]

*Correspondence: atsushi.kitani@gmail.com
[†]Atsushi Kitani, Taketo Kimura and Takako Nakatani are Equal contributors.
[1] Graduate School of Business Sciences, University of Tsukuba, Tokyo, Japan
Full list of author information is available at the end of the article

Abstract

As tablet devices become popular, various handwriting applications are used. Some of applications incorporate a specific function, which is generally called palm rejection. Palm rejection enables application users to put the palm of a writing hand onto a touch display. It classifies intended touches and unintended touches so that it prevents accidental inking, which has been known to occur under the writing hand. Though some of palm rejections can remove accidental inking afterward, this function occasionally does not execute correctly as it may remove rather correct ink strokes as well. We call this interaction Incorrect Drawn Ink Retrieval (IDIR). In this paper, we propose a software algorithm that is a combination of two palm rejection logics that reduces IDIR with precision and without latency. That algorithm does not depend on specific hardware, such as an active stylus pen. Our data provides 98.98% correctness and the algorithm takes less than 10 ms for the distinction. We confirm that our experimental application reduced the occurrences of IDIR throughout an experiment.

Keywords: User experience, Usability, Machine learning, Handwriting, Drawn Ink Retrieval, Incorrect Drawn Ink Retrieval

Background

As tablet devices are widespread, various handwriting applications have been and continue to be developed. When a user tries to write something on a tablet with a general handwriting application, a multi-touch interaction compels the user to float the hand above the display to avoid accidental inking. Since this unnatural way of writing produces difficulties [1] for digital handwriting applications, the function called "palm rejection" becomes crucial. Palm rejection distinguishes intended touches from unintended touches, and prevents accidental inking.

All intended and unintended touches are classified into the following four categories, True Positive, True Negative, False Positive and False Negative.

When palm rejection correctly distinguishes an intended touch, it results in the True Positive and the touch draws a correct ink stroke. Palm rejection correctly rejects unintended touches and the result is the True Negative. Then, there is no accidental inking under the palm. On the other hand, when an unintended touch is recognized as an intended touch, it is the False Positive and accidental inking occurs. When an intended touch cannot be recognized correctly, the touch is the False Negative and it is incorrectly rejected.

Making a distinction between intended touches and unintended touches seams to be quite straightforward, though it is rather a complicated problem. Because most touches tend to move rapidly and are not stable. Therefore, it is necessary for palm rejection to analyze all touch data within a short moment and make an immediate distinction.

There are general applications [2–4] which have already been embedded in palm rejection. While researching palm rejection of those handwriting applications, a curios interaction was detected. When a user tries to write something on a touch display with those handwriting applications and an intended touch draws an ink stroke correctly, what occasionally occurs afterward is that the stroke is removed in a very short moment against the user's will. From this interaction, we infer that some of palm rejection algorithms iteratively classify intended touches and unintended touches, and switch the distinction afterward. It is reasonable when the interaction happens for the touch that should have been classified as the True Negative but classified as the False Positive and thus draws an accidental ink stroke under the palm; though, in some cases the True Positive touch is switched to the False Negative touch afterward. It is rather perplexing for users when the interaction incorrectly switches the classification and removes correct ink strokes. We call the correct interaction "Drawn Ink Retrieval (DIR)" and call the incorrect interaction "Incorrect DIR (IDIR)" (Fig. 1).

The purpose of this paper is to propose a palm rejection algorithm which reduces the occurrences of IDIR. To realize the algorithm, we took an approach that makes use of multi-touch interaction and then, two different logics are combined. One is a machine learning technique, while the other is an occlusion area protection.

Incorrect Drawn Ink Retrieval will bring a negative outcome for the application's usability because it does not occur in a natural handwriting situation. Therefore, this approach will be an effective option as the palm rejection algorithm.

This paper is structured as follows, in "Background", the problem with palm rejection named IDIR are revealed. In "Related work", we briefly categorize two types of approaches in dealing with palm rejection. In "Our approach", the combination of two logics is introduced. In "Experiment", a process of developing a handwriting application and an experiment are explained. We discuss the results and any remaining problems in "Discussion", and offer "Conclusion".

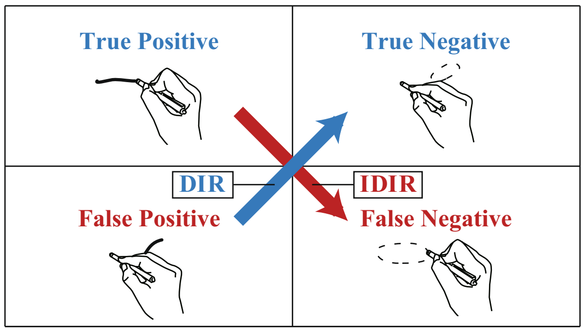

Fig. 1 The image of DIR and IDIR. This figure shows the image of DIR and IDIR occurrence

Related work

In this section, the existing approaches are surveyed and are categorized into the following two types. One is an active stylus pen interaction, while the other is a multi-touch interaction.

There are several approaches in classifying intended touches and unintended touches. Annett et al. [5] have researched such approaches and made a comparison. They categorized the various approaches in four types: user adaptions, firmware approaches, operating system approaches and application-specific approaches. Schwarz et al. [6] categorized existing approaches into hardware solution and software solution.

Generally there are two approaches to solve accidental inking. One is positively utilizing various functions that are embedded in hardware: for example, using an active stylus pen, which is distinguished as a specific touch from other general finger touches. This is the current main approach of palm rejection. The other approach is focusing on multi-touch interaction itself and solving the problem with software algorithms, which does not depend on specific hardware or devices.

The approach of multi-touch interaction is less precise than the approach of utilizing active stylus pens. Even though the above statement is the case, researching and developing the approach of multi-touch interaction is meaningful, because every standard capacitive touch device does not always embed an active stylus pen.

Active stylus pen interaction

Various prototypes were researched as novel interaction devices [7–9]. Such researches can be the base technology of future products.

Several devices have already had an active stylus pen interaction embedded. Sumsung Galaxy's S Pen [10], WACOM digitizer [11] and Windows Surface's Pro Pen [12] have a similar function to palm rejection, and further, they can even recognize pen pressure. In addition, some of them manipulate the touch device without physical touches on the display. Using those active stylus pens enables the application to simplify the touch classification. Though they are a reliable solution for accidental inking, those approaches depend on specific hardware. For instance, S Pen depends on Sumsung Galaxy, and Pro pen depends on Windows Surface.

Various active stylus pens, which utilize Bluetooth technology, freeing their dependence on specific touch devices, are available for standard capacitive touch devices such as the iPad. BambooPaper by WACOM [3], GoodNotes by Time Base Technology Limited [4] and Penultimate by EVERNOTE [2] have an option of connecting those stylus pens via Bluetooth. When utilizing the active stylus pen, those applications display more accurate palm rejection results.

Fifty three [13] provides both an original active stylus pen called Pencil, and an application called Paper. This application supplies palm rejection but it only works with the original active stylus pen.

Multi-touch interaction

In terms of a precision, the multi-touch interaction approaches are inferior to the active stylus pen interaction approaches. On the other hand, in terms of versatility, the multi-touch interaction approaches are more adaptable for several Operating Systems and multi-touch devices than the active stylus pen interaction approaches.

Several researches for the multi-touch interactions [14–16] attempt to recognize finger touches. Current capacitive touch devices can correctly receive all touches, but still have a difficulty in classifying which touches are intended and which touches are not.

One well known approach is that there is a specific region in which applications ignore all touches. Users can put their hand onto the region without ink strokes. All touches outside of the region are recognized as intended touches, and so, draw ink strokes. Note-Anytime, which is a handwriting application by MetaMoJi, takes this approach [17]. An advantage of this simple approach is that while a user is putting his or her hand on the region, accidental inking does not occur. So, DIR and IDIR also do not occur. A disadvantage is that users need to move the region manually according to the hand position and where they want to write. In terms of usability, this uncomfortable way of writing will bring negative user experiences.

Vogel et al. [18] corrected the pen and hand position data, which is named occlusion silhouettes, by means of captured images form a head mounted camera. Then, they present a scalable circle and pivoting rectangle geometric model, which detects a position of a hand and forearm from pen nib coordinates. If the pen nib coordinates are clearly pinpointed, the model can be made use of palm rejection.

Yoon et al. [19] made use of the model of Vogel et al. to reject unintended touches while an active stylus pen is recognized. Whereas for the handwriting with a general stylus pan, the pen nib does not always touch a display. Thus other logics will be needed to apply this model to reject unintended touches for handwriting applications.

Schwarz et al. [6] proposed a novel solution using spatiotemporal touch features. It votes for all touches iteratively each 50–500ms on whether they are intended touches or unintended touches through the utilization of the decision tree, which is one of the machine learning algorithms. It is said that their solution is a current baseline for palm rejection. On the other hand, Annett et al. [5] pointed out the problem of classification speed. In the paper, DIR and IDIR are not evaluated, though it is mentioned that False Positive touches would be switched to True Negative touches through the iterative classification.

BambooPaper [3], GoodNotes [4] and Penultimate [2] also have the palm rejection function, which utilizes multi-touch interactions. In order to activate their function, a registration of the users' dominant hand information is required. Additionally, GoodNotes and Penultimate require a frequent hand posture. Those applications are considered to make use of machine learning techniques to classify intended touches and unintended touches. Therefore applications need to adjust the learning data to users' writing posture. When the writing posture fits the registered posture, palm rejection works mostly correct. However, if the writing posture becomes too estranged from the registered posture, applications tend to make incorrect rejections, and thus, IDIR also tends to occur.

Our approach

According to the past researches [19] and the existing application [17], utilizing an occlusion area will be a reliable approach to reduce occurrences of IDIR. It is important to point out that in making the dynamical occlusion area without any pen nib information, this then requires other information which detects can detect hand positions.

Making use of the machine learning technique becomes a standard way to classify intended touches and unintended touches. In general, the technique is used to reject all unintended touches which are considered as unnecessary. Most unintended touches are generated by a writing hand, and thus, those touches indicate where the writing hand itself is. This means that the information of unintended touches enables the production of the dynamical occlusion area.

Our approach is to build a touch distinction model by means of Support Vector Machine (SVM), which is one of the machine learning algorithms and is suitable for solving two-class tasks. The SVM model classifies intended touches and unintended touches. The True Positive intended touches are recognized as the pen nib and draw strokes. The True Negative unintended touches do not draw strokes. Furthermore, the unintended touches are taken advantage of in that they produce the dynamical occlusion area.

Touch distinction by the SVM model

To recognize the pen nib from all touches, the following classifier is introduced.

$$y = \sum_{i=1}^{N} w_i^\top \boldsymbol{x}_i + b, \tag{1}$$

where N is a number of explanatory variable. The number of touch coordinates and touch records, which are described in "Developing the SVM model", will be utilized as the explanatory variable. The value of X coordinate and Y coordinate is $\boldsymbol{x}_i$. A bias is b, and w_i is determined by SVM, in which L2-regularized L2-loss SVC [20] solves the following primal problem:

$$\min_{w} \quad \frac{1}{2}\boldsymbol{w}^\top \boldsymbol{w} + C\sum_{i=1}^{l}\left(\max\left(0, 1 - y_i \boldsymbol{w}^\top \boldsymbol{x}_i\right)\right)^2. \tag{2}$$

Support Vector Machine is a supervised learning method, and requires a tagged dataset. In this case, the tags will be intended touches or unintended touches. To build the dataset, both w and $\boldsymbol{x}$ are matrix in the classifier (1), and therefore w and $\boldsymbol{x}$ will be vectorized to apply L2-regularized L2-loss SVC (2).

After building the SVM model by means of the dataset, the SVM model classifies intended touches and unintended touches. When y is plus in the classifier (1), the coordinate is classified as the intended touch, whereas y is minus, it is classified as the unintended touch.

The difficulty is when there is only a palm on a display and there should not be intended touches, thus, the classification algorithm occasionally detects intended touches incorrectly and accidental inking is produced. The iterative classification can retrieve it, however latency and IDIR will occur as the side effects.

Touch protection by the dynamic occlusion area

In order to reduce the occurrences of IDIR without latency, a simple and robust rejection logic is needed. The past research utilized the pen nib information and the hand posture geometric model to detect the hand position [18]. In the case of handwriting, the pen nib does not always touch a display. However, we create the dynamical occlusion area by making use of the information of a hand position, which is detected by the SVM model classification. Inside of the occlusion area, all touches will be rejected. This occlusion area supplementarily avoids accidental inking by the False Positive touches.

Experiment

In order to evaluate this approach, an experiment was implemented. For the experiment, we took the following steps.

1. Collecting the dataset for SVM
2. Developing the SVM model
2. Implementation of the dynamical occlusion area
4. Developing a handwriting application
5. Evaluation

Collecting the dataset for SVM

To collect the dataset, the technique by Schwarz et al. [6] was applied. The dataset was separately collected from 10 right-handed participants and 2 left-handed participants. Participants held a standard stylus pen, which has a simple rubber nib, and which is not active. They put their palm onto a touch display. Touches inside the circle represent intended pen touches, whereas touches outside of the circle are interpreted as unintended touches.

For the purpose of collecting realistic handwriting data, we let the circle dynamically follow the pen nib. Participants were told to put the pen nib onto the circle and make strokes on the touch display evenly. To do so, the dataset becomes closer to real handwriting data.

When any touch events occur on the display, one record will be produced, the record includes all existing X and Y touch coordinates. About 250,000 records were set as the training dataset, with 5000 records being set as the validation dataset. From the dataset, a total of 20 models were produced. The record number of the model increases by 1 up until 10 and, after that the record number increases 20 each up until 200. Before having the experiment, we applied each model to the validation dataset and examined which model is the most effective for the most precise classification. A model size with 20 records provides the highest correct percentage of 98.98% for the classification. Thus, the model size with 20 records was adopted (Fig. 2).

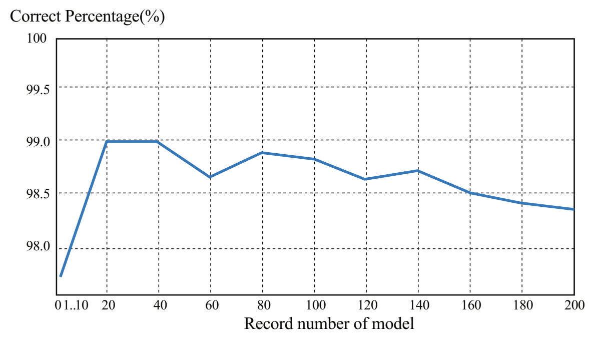

Fig. 2 The correct percentage for each record number of model. This figure indicates the correct percentage and the record number of models

Developing the SVM model

To make the distinction, actual touch dataset also needs the same 20 records. The 20 records are composed of 19 contextual records and 1 most recent record.

We let r stand for the number of the records, and let t stand for the number of touch coordinates. In this experiment, maximum number of touch coordinates is set 10. Thus, in the classifier "Touch distinction by the SVM model" (1), N will be $t \times r = 10 \times 20 = 200$.

Whenever touch events occur, all touch coordinates are stacked into the record. If there are fewer touches than the maximum number, 0 is stacked to fill the record. In the case that there are no records when the touch events start, it takes approximately 5 ms to stack all recorded 20 records in order to make a distinction. When 20 records already have been stacked, it takes approximately 1 ms for the calculation. If there are no touch events, the records are not stacked. When no touch points are detected, the stacked records expire.

Implementation of the occlusion area protection

After the SVM model has classified intended touches and unintended touches, the dynamical occlusion area is applied to avoid accidental inking and reduce the occurrences of IDIR. the dynamical occlusion area is an invisible circle. The round shaped occlusion area is adopted to cope with changing the writing hand angle. The X coordinate for the center of the circle is the mean value of the X coordinates of all unintended touches. The Y coordinate for the center of the circle is calculated in the same way.

The circle has a radius of 230 px. The radius is derived from the dataset that was collected previously. A mean length between the intended touches and the unintended touches was 390 px. We heuristically adjust the size of the radius of the circle. The suitable radius was slightly longer then half of the mean length.

When the SVM model classifies intended touches and unintended touches, and if a False Positive touch is inside of the dynamical occlusion area, the occlusion area avoids accidental inking (Fig. 3).

While a user is putting his or her hand on the display when writing something with a stylus pen, it takes approximately 1 ms to generate the dynamical occlusion area. Totally,

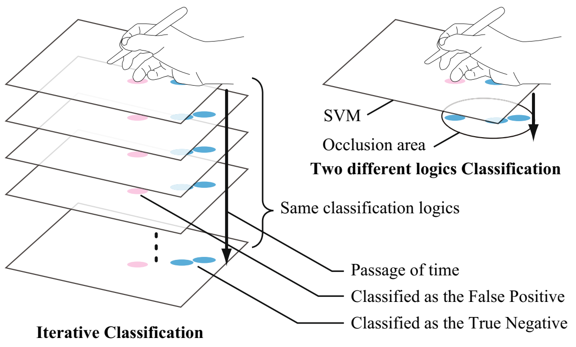
SVM
Occlusion area
Two different logics Classification
Same classification logics
Passage of time
Classified as the False Positive
Classified as the True Negative
Iterative Classification

Fig. 3 The concept of the iterative classification and our approach. It compares the concept of iterative classification and our approach

it takes 2 ms for the combined classification with the inclusion of the drawing records, and within 10 ms without the records.

In the application of this approach, the dynamical occlusion area cannot be applied without unintended touches. Tush, the first touch offers one especially difficult situation. When a first touch and a second touch occur one by one, and the first touch is recognized as an intended touch before the second touch occurs. Then the first touch will immediately draw an ink stroke. However, there will be three possible cases.

Case one is that the first touch was truly intended and the second touch was unintended. Case two is that the first touch was unintended and the second touch was intended. And case three is that both the first touch and the second touch were unintended. In both case two and three, the unintended drawn ink stroke should be removed. Therefore, to minimize the occurrences of IDIR, we embed a function that invokes DIR only for the first touch.

Developing a handwriting application

In order to realize this approach, an experimental handwriting application is developed. The application is developed as a web application with JavaScript and HTML5 Canvas. Though we have various types of multi-touch devices, a hardware specific application will have difficulties of extensibility with all of these various devices. JavaScript and HTML5 Canvas, on the other hand, can execute within most of the modern browsers and multi-touch devices.

Current capacitive touch devices, like the iPad, can utilize a touch radius. Some general applications may take advantage of the touch radius information to improve the precision of palm rejection. However, it depends on an Operating System whether the touch radius can be used or not. this approach does not make use of touch radius information. Therefore, there is a benefit of extensibility with other browsers, Operating Systems and multi-touch devices.

Evaluation

The handwriting application, in which is embedded the combined two logics, was compared with two other applications. One was GoodNotes, with the other being Penultimate. Both of these applications have palm rejection, which is supposed to be based on machine learning techniques. Furthermore, both of them have an option of connecting with active stylus pens, however, the experiment was held using a non-active stylus pen.

The iPad Air 2 was used for the experiment with the device being set on a horizontal orientation.

8 subjects participated in the experiment. Seven subjects were right-handed and one was left-handed. They were university students and were familiar with using tablet devices. All of the subjects who took part were different from the subjects who participated in collecting the dataset.

3 types of characters were displayed in each application. One was the capital alphabet letters A to G; one was the numbers 1 to 6; and the last one was Japanese Kanji, which consisted of four characters (Fig. 4).

The subjects were instructed to trace all characters according to the lists below.

- For GoodNotes and Penultimate, set a suitable angle of a dominant hand from each application's setting
- For our application, choose right or left hand.
- Write every character in the correct order and correct number of strokes
- Do not rewrite, even if a stroke is not correctly drawn
- Do not rewrite, even if DIR occurs
- Writing in script is prohibited
- Writing style should be the same as when one writes something using a pen and a notebook
- Use a non-active stylus pen, which we provide

For each application, one practice period was provided. The experiment was set in random order.

Discussion

In this section, we provide the results of the experiment and discuss about them. The writing processes were recorded on video. We classified all strokes as the below three interactions.

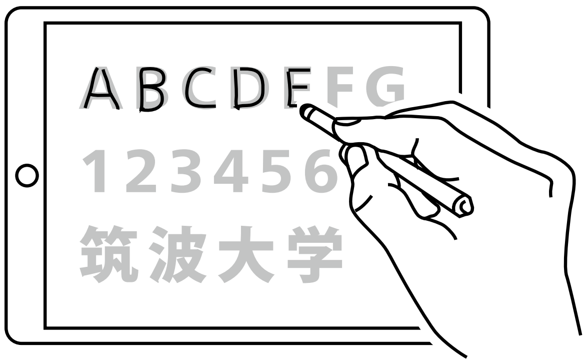

Fig. 4 The image of the experiment. This is an image of the experiment

- Correct ink stroke (True Positive)
- Accidental inking (False Positive)
- Changing the classification from True Positive to False Negative (IDIR)

The classification of True Negative are invisible and could not be evaluated. The occurrences of DIR, which is changing the classification from False Positive to True Negative, were not evaluated because the interaction mostly happens under the palm and can not be confirmed.

Observation

Table 1 shows the total number of interactions for each application. The difference in the number of All Strokes is because some subjects wrote characters following the wrong procedure. For instance, a letter A was written with two strokes instead of the correct three strokes. The number of False Positives was close to all three applications.

GoodNotes shows a low frequency of False Negative strokes. GoodNotes and Penultimate recorded 5 IDIR, while our application recorded 1.

Table 2 shows a detail of the IDIR numbers for each subject. The result of our application shows that: subject G recorded 1 IDIR, while the other seven subjects did not record any IDIRs. Compared with our application, the other applications recorded numerously more IDIRs.

Summary of techniques

In this experiment, subjects wrote three types of characters. Some of characters, typically Japanese Kanji, are composed of several strokes which makes them intricately more complex writing than the characters which are composed of a single stroke. The reason of adopting those characters is because IDIR does not occur often, and writing those intricate characters will induce more occurrences of IDIR. In addition, writing such intricate characters will be a closer simulation of a real handwriting situation.

Threats to validity

All subjects are university students and all of them are familiar with using touch devices, thus, if subjects who have had no experiences in the use of touch devices, the results will change. Also, the size of hands will influence for the result of the experiment.

The experiment of Schwarz et al. [6] defines the True Positive strokes as the Stroke Recognition, and the False Positive strokes as the Error Strokes. The results were 97.9% for the True Positive strokes, with the False Positive rate being 0.016. our approach resulted in 95.2% for the True Positive strokes and 0.082 for the False Positive rate. Their experiment was drawing below six symbols, character L, S, vertical line, horizontal line,

Table 1 Total number of interactions and classifications

Application	All strokes	True Positive	False Positive	IDIR
Our application	437	416	36	1
GoodNotes	435	429	38	5
Penultimate	436	397	35	5

Table 2 The number of IDIR for each subjects

Application	Subjects (R/L)	Strokes	IDIR
Our application	A (R)	55	0
	B (R)	56	0
	C (L)	56	0
	D (R)	55	0
	E (R)	53	0
	F (R)	54	0
	G (R)	54	1
	H (R)	54	0
GoodNotes	A (R)	54	1
	B (R)	56	2
	C (L)	56	0
	D (R)	55	1
	E (R)	53	0
	F (R)	54	0
	G (R)	53	0
	H (R)	54	1
Penultimate	A (R)	55	0
	B (R)	56	2
	C (L)	56	0
	D (R)	55	1
	E (R)	53	1
	F (R)	54	1
	G (R)	54	0
	H (R)	53	0

dot and circle. When we consider that all those symbols are composed of a single stroke and are simpler than the characters in our experiment, our results with regard to the precision is convincing.

Though the whole number of IDIR was much smaller than the total stroke number, we did not reach statistical significance.

The iPad Air 2 was used as the device for collecting the dataset and for the experiment. We set the iPad on a horizontal orientation. To use a vertical orientation or other devices, another dataset will be needed. The device specification will influence the data correction and the classification speed.

Future work

Through this research, we illustrate that unintended touches, which used to be regarded as useless information, can take advantage of generating the dynamical occlusion area. Though, the size of the area should adjust according to the user's hand size. In addition, the shape and position of the area should be considered. Those improvements will be included in our future works.

Conclusion

Past researches and existing applications focus on reducing the occurrence of accidental inking. We focus our attention on IDIR, which occurs because of a result of a re-classification from the True Positive to the False Negative.

Concerning the number of correct ink strokes and accidental inking, our application performed on par with the other applications when compared.

We can confirm that our application certainly reduced the occurrences of IDIR throughout the experiment. In reducing the occurrences of IDIR, this approach will be a possible option.

To achieve higher quality, there is still plenty of room for improving the precision in our palm rejection algorithm, and more experimental results will be needed.

Abbreviations
DIR: Drawn Ink Retrieval; IDIR: Incorrect Drawn Ink Retrieval; SVM: Support Vector Machine.

Author's contributions
AK developed the base handwriting application and carried out the experiment and drafted the manuscript. TK developed the palm rejection algorithm and helped to draft the manuscript. TN advised on the way of the experiment and helped to draft the manuscript. All authors read and approved the final manuscript.

Author details
[1] Graduate School of Business Sciences, University of Tsukuba, Tokyo, Japan. [2] Tokyo, Japan. [3] The open university of Japan, Chiba, Japan.

Acknowledgements
The authors special thank to subjects for joining the palm rejection experiment and data collection of SVM.

Competing interests
The authors declare that they have no competing interests.

Publisher's Note

Received: 7 September 2015 Accepted: 18 May 2017
Published online: 15 June 2017

References
1. Camilleri MJ, Malige A, Fujimoto J, Rempel DM (2013) Touch displays: the effects of palm rejection technology on productivity, comfort, biomechanics and positioning. Ergonomics 56(12):1850–1862. doi:10.1080/00140139.2013.847211
2. Evernote: Penultimate. https://evernote.com/intl/jp/penultimate/
3. BambooPaper: Wacom. http://www.wacom.com/ja-jp/jp/everyday/bamboo-paper
4. GoodNotes: Time Base Technology Limited. http://www.goodnotesapp.com
5. Annett M, Gupta A, Bischof WF (2014) Exploring and understanding unintended touch during direct pen interaction. ACM Trans Comput-Hum Interact 21(5):28. doi:10.1145/2674915
6. Schwarz J, Xiao R, Mankoff J, Hudson SE, Harrison C (2014) Probabilistic palm rejection using spatiotemporal touch features and iterative classification. In: Proceedings of the 32nd Annual ACM conference on human factors in computing systems. ACM, New York, pp 2009–2012. doi: 10.1145/2556288.2557056
7. Hinckley K, Yatani K, Pahud M, Coddington N, Rodenhouse J, Wilson A, Benko H, Buxton B (2010) Pen + touch=new tools. In: Proceedings of the 23nd Annual ACM Symposium on user interface software and technology. ACM, New York, pp 27–36. doi: 10.1145/1866029.1866036
8. Song H, Benko H, Guimbretiere F, Izadi S, Cao X, Hinckley K (2011) Grips and gestures on a multi-touch pen. In: Proceedings of the SIGCHI conference on human factors in computing systems. ACM, New York, pp 1323–1332. doi: 10.1145/1978942.1979138
9. Suzuki Y, Misue K, Tanaka J (2007) Stylus enhancement to enrich interaction with computers. In: Proceedings of the 12th international conference on human-computer interaction: interaction platforms and techniques. Springer, Heidelberg, pp 133–142. http://dl.acm.org/citation.cfm?id=1757268.1757284
10. SAMSUNG: S Pen. http://www.samsung.com/us/
11. Digitizer: Wacom. http://www.wacom.com/en-us
12. Microsoft: Pro Pen. http://www.microsoft.com/surface/en-us
13. Paper: FiftyThree. https://www.fiftythree.com/paper
14. Wagner J, Huot S, Mackay W (2012) Bitouch and bipad: designing bimanual interaction for hand-held tablets. In: Proceedings of the SIGCHI conference on human factors in computing systems. ACM, New York, pp 2317–2326. doi: 10.1145/2207676.2208391
15. Ewerling P, Kulik A, Froehlich B (2012) Finger and hand detection for multi-touch interfaces based on maximally stable extremal regions. In: Proceedings of the 2012 ACM international conference on interactive tabletops and surfaces. ACM, New York, pp 173–182. doi: 10.1145/2396636.2396663

16. Marquardt N, Kiemer J, Ledo D, Boring S, Greenberg S (2011) Designing user-, hand-, and handpart-aware tabletop interactions with the touchid toolkit. In: Proceedings of the ACM international conference on interactive tabletops and surfaces. ACM, New York, pp 21–30. doi: 10.1145/2076354.2076358
17. NoteAnytime: MetaMoJi. http://product.metamoji.com/ja/anytime/
18. Vogel D, Cudmore M, Casiez G, Balakrishnan R, Keliher L (2009) Hand occlusion with tablet-sized direct pen input. In: Proceedings of the SIGCHI conference on human factors in computing systems. ACM, New York, pp 557–566. doi: 10.1145/1518701.1518787
19. Yoon D, Chen N, Guimbretière F (2013) Texttearing: opening white space for digital ink annotation. In: Proceedings of the 26th Annual ACM Symposium on user interface software and technology. ACM, New York, pp 107–112. doi: 10.1145/2501988.2502036
20. Fan R-E, Chang K-W, Hsieh C-J, Wang X-R, Lin C-J (2008) Liblinear: a library for large linear classification. J Mach Learn Res 9:1871–1874