Proceedings of 2020 IEEE
International Conference on Mechatronics and Automation
October 13 - 16, Beijing, China

Visual Pencil: Design of Portable Human-Computer Interaction Based on 2D Visual Tracking

Ru Tong[1,2], Tianzhu Wang[1,2], and Junzhi Yu[1,3]
[1]*State Key Lab Management and Control for Complex Systems, Institute of Automation, CAS, Beijing 100190, China*
[2]*School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China*
[3]*Dept. Mech. Eng. Sci., BIC-ESAT, College of Engineering, Peking University, Beijing 100871, China*
{*tongru2019, wangtianzhu2015, junzhi.yu*}*@ia.ac.cn*

***Abstract*—Convenient and efficient human-computer interaction has always been the goal pursued by researchers. In this paper, we present a vision-based handwriting interaction device for non-touch screens called Visual Pencil. Unlike the conventional interactive method that captures the user's gestures using a motionless visual sensor, Visual Pencil directly mounts the visual sensor in the device. More specifically, the visual data are processed by the feature detection algorithm, and the movement of the pen tip in a 2D plane is calculated, so as to control the cursor and achieve the purpose of interaction. We further define performance indicators for evaluating the design and select more reasonable parameters through specific experiments to achieve the best interaction performance. The experiments verify that the proposed design is able to fulfill the desired operations. The significance of this design lies in that it integrates the functions of the mouse and smart pen, thus removing the requirement of more auxiliary equipment. This work sheds light on the interaction with non-touch screen computers.**

***Index Terms*—Human-computer interaction, visual tracking, visual pencil, smart pen, portable interaction.**

I. Introduction

In recent years, smart pens for non-touch screens have been continuously developed. As a new type of human-computer interaction, the smart pen is freer than traditional mouse, keyboard and input methods for touch screens, more in line with human usage habits, and makes human-computer interaction more liberal. A feasible solution of the smart pen for non-touch screen is to determine the movement of the pen tip and record what the user wrote by optical measurement technology [1] such as the smart pen invented by Anotto Corporation and the Phree developed by OTM Technologies. The drawback of the scheme is that the smart pen is limited to use on specially-made coded paper [2]. Another solution follows the principle of electromagnetic induction, such as Wacom Bamboo Spark smart pen. When using such a smart pen, the paper is placed on a digital tablet capable of generating a magnetic field, and coded paper is no longer needed, which is more flexible to use than the first type. In addition, smart pens based on sonic localization have been studied, and the representative is Equil Smart pen 2. This type of smart pen uses ultrasonic and infrared technology for positioning. However, it requires an additional ultrasonic receiver, and the ultrasonic wave will continuously emit a beep sound when in use. The smart pens above are limited in certain terms, which may be improved by some more effective methods such as machine vision.

Machine vision has been increasingly used in portable human-computer interactive devices. A multitude of vision-based interactions are based on gesture recognition. Static or dynamic gestures are recognized to manipulate or communicate with a computer [3]. For example, Bourdot *et al.* implemented one-handed manipulation of three-dimensional (3D) objects in 2010 [4], and an interface developed by Kang *et al.* in 2013 defined operations to be performed based on left-hand gestures and specified their parameters based on right-handed actions [5]. A more typical application is leap motion [6], which uses infrared LEDs and gray-scale cameras to model 3D human hands, determines the shape and position of human hands, and then controls the machine. There are some inevitable shortcomings of interacting through gesture recognition. The gestures specified in the gesture dictionary can only be recognized after training with a large amount of data, requiring a greater cognitive load [7]. Although free-form gestures are not limited by high-load learning, when capturing the user's movements, once the user's hands are rotated to an angle that cannot be visually distinguished, the recognition performance will decrease sharply.

Inspired by the visual odometry [8] in SLAM and the photoelectric principle of traditional mouse, a vision-based handwriting interaction device called Visual Pencil is designed in this paper, which draws on visual interaction and implements the function of handwriting interaction. Visual Pencil, held by user, mounts the visual sensor directly in the tip, calculates the pencil's movement by processing the captured images, and issues instructions to interact with the computer. The device works in the plane about 10 cm away from the screen. When the user moves the Visual Pencil, the visual sensor continuously captures the image of the area on the screen pointed by the pen tip. The captured images are processed through feature detection and feature matching algorithms, and then the device's movement in the 2D plane is calculated by using the matched feature points. Finally, the computer interacting with the device controls the cursor's movement and does the click operations according to the results of data processing to achieve the purpose of interaction. The main innovation lies in the integration of the functions of mouse and smart pen without the need of any auxiliary equipment. The device adopts the USB Video Class(UVC) standard protocol and can be directly connected to the computer via USB, plug and play, which has strong convenience.

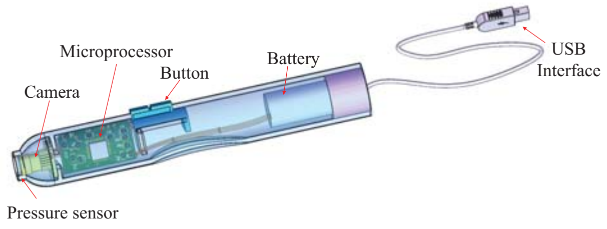
The remainder of this paper is structured as follows. The proposed design scheme of Visual Pencil is detailed in Section II. The processing method for captured image data is introduced in Section III. Tests and results are provided in Section IV. Finally, concluding remarks are offered in Section V.

II. Design Scheme of Visual Pencil

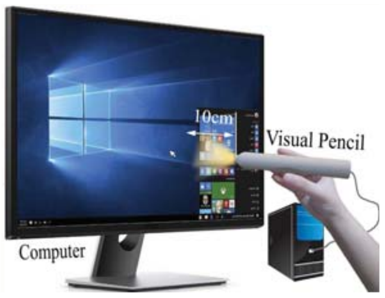
Processing image data in real time for portable vision-based interactions is a difficult problem, yet to be adequately resolved. Processing image data is a time-consuming step. For portable interactive devices, there are two main methods for processing image data. The first is to process the image data in the microprocessor mounted in the device, and the second is to transfer the image data to a computer that interacts with it for processing. There is no doubt that the former can handle data on its own with greater convenience, but it not only requires a high-performance processor, but also requires a more complex hardware platform. Considering the complexity of the feature detection algorithms involved, the device designed in this paper chooses the second method to process image data. In other words, Visual Pencil only transfers the image data to the computer that interacts with it, and then the computer runs the image processing algorithm.

Considering the curve of the human hand and the comfort when holding it, the hardware device shown in Fig. 1 is designed for Visual Pencil. The camera captures real-time images of the screen directly opposite the pen tip, and these images are used to calculate the movement of Visual Pencil in the 2D plane. In view of the demand of click and reset commands, the hardware platform of Visual Pencil also designed two buttons and a pressure-sensitive module. These two buttons control the computer to perform left and right clicks at the cursor position. The pressure-sensitive module at the pen tip is used for resetting, and can effectively protect the camera lens.

Before transmitting image data to the computer, all sensor data needs to be integrated in the device, which is significant. Data integration not only eliminates the data transmission channel for two buttons and pressure-sensitive sensors, but also requires only a USB cable to transfer the integrated data. Installing the device is exactly as easy as installing a USB camera. The specific integration process is: first, the microprocessor on the device adds the buttons and pressure-sensitive data to the end of the image sequence according to certain encoding rules to form a complete data stream, and then sends the complete data stream to the computer's USB interface according to the UVC standard protocol. After receiving the complete data stream, the computer decodes the data stream into three parts: an image, key events and reset signal. The computer processes the image through the feature detection algorithm, calculates the 2D moving distance of the Visual Pencil, and then combines the key events and reset



(a)



(b)

Fig. 1. Hardware design of Visual Pencil. (a) Model diagram of Visual Pencil. (b) Conceptual diagram of manipulating Visual Pencil.

signal to control the cursor to complete the desired actions. The complete data flow is shown in Fig. 2.

III. Processing Method for Captured Image Data

This section is developed on the assumption that the computer has obtained the complete data stream from Visual Pencil and the image data has been decoded. This section will expand on specific algorithms for processing image data captured by Visual Pencil's camera. Firstly, we analyze the characteristics of the images captured by the camera and choose a suitable feature detection method through a comparative experiment. Secondly, the feature descriptors of each frame are calculated and the feature points in adjacent frames are matched through a feature matching algorithm. Then the moving distance of the focus pointed by the Visual Pencil between the two frames is calculated according to the matches. Finally, a complete data processing and control process is given. It is worth noting that real-time performance and matching accuracy are two important factors that the processing algorithm needs to consider.

A. Feature Detection and Matching Methods

In the designed application scenario, the camera of Visual Pencil captures the content displayed on the computer screen, so the main features of the images appear in the areas where the graphics and text are located. There are two extreme cases of captured images. One is that the feature points in the image are too dense, for example, the camera captures the areas with dense text. In this case, if the method of calculating feature points and the selected feature detection threshold are not appropriate, a large number of feature points will aggregate, which not only slows down the calculation speed, but also

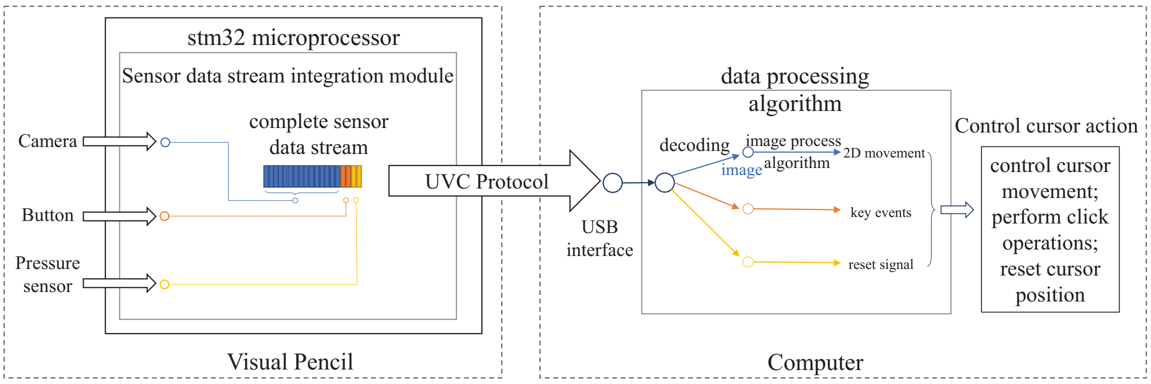


Fig. 2. Data flow diagram.

brings trouble to the feature matching. Therefore, it is very important to choose a suitable feature detection algorithm and a suitable threshold. Another extreme case is that feature points are too sparse, such as the solid-color areas that often appear at the edge of the screen and the areas that are invariant in a certain direction. In this case, it is difficult to obtain accurate feature points through feature detection. Therefore, it is very difficult to calculate the movement of Visual Pencil in the 2D plane. This situation will be discussed in the next section. This section mainly discusses how to obtain feature points and match them when the feature points are dense or moderate.

Oriented FAST and Rotated BRIEF (ORB) algorithm is a classic feature detection algorithm, developed from the brief algorithm [9]. The ORB algorithm is 100 times more efficient than the SIFT algorithm according to theoretical calculation. ORB feature detection is often used in visual SLAM [10]. In this paper, we detect the feature points of each frame through the ORB algorithm, match the feature points in adjacent frames through the FLANN algorithm, and use the Lloyd's algorithm to select good matches. Through trial and error, we discover that bad matches, the matches with inconsistent inclination, appear in almost every frame(Fig. 3(a)), which has a great impact on the subsequent calculation of 2D movement. Therefore, we plan to filter matches based on directional consistency. The geometric inclination of the matched points are calculated, and then the statistical method is used to select the interval with the densest distribution of angle of inclination (Fig. 3(b)). All matches in this interval has directional consistency, and the angular value in this interval represents the true direction of movement with a high probability. The filtered matches will be used for the subsequent calculation.

The directional consistency principle can effectively filter out the relatively good matches. However, this method does not specifically evaluate the quality of each match and only avoids bad matches through statistics, which may lead to the inability to accurately determine the statistical peak angular value when the total number of feature points is small. Regarding the selection of high-quality matches, Bian *et al.* published their related work in 2017 [11]. In the paper, they proposed the

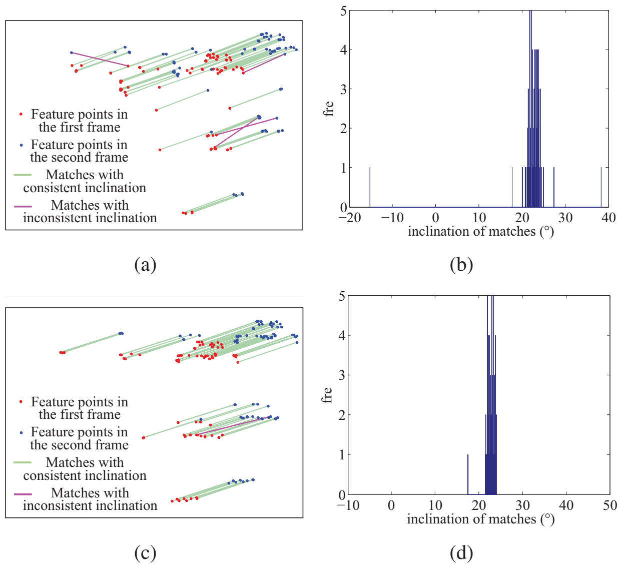


Fig. 3. Comparison of directional consistency between ORB and GMS. (a) The lines of feature points matched by the ORB feature detection algorithm between two frames. (b) Statistical chart of the inclination of the lines in (a). (c) The lines of feature points matched by the GMS feature detection algorithm between two frames. (d) Statistical chart of the inclination of the lines in (c).

GMS algorithm, which uses grid-based motion statistics to separate the good and bad matches. Bian *et al.* also released the GMS program combined with the ORB algorithm. The feature matches obtained by GMS have strong directional consistency, as shown in Fig. 3(c) and Fig. 3(d).

Table I lists the running time, the proportion of good matches, and the degree of directional consistency of ORB and GMS when the number of feature points detected by the two algorithms is close to 500. According to Table I, the running time of GMS is longer, while the matching accuracy of ORB is slightly inferior. Both high accuracy and good real-time performance are necessary for an excellent user experience. The matching accuracy through ORB feature detection and directional consistency filtering principle can basically meet the

TABLE I
COMPARISON OF ORB AND GMS ALGORITHM PERFORMANCE

Method	Running time	Proportion of good matches	Angular consistency
ORB	0.033s/30.3fps	84/500	80/84
GMS	0.262s/3.93fps	257/498	249/257

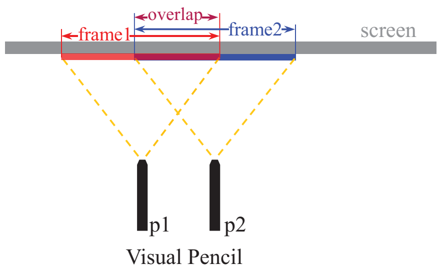


Fig. 4. Schematic diagram of the positional relationship between two adjacent frames.

needs, and considering real-time performance the subsequent experiments will use the ORB algorithm for feature detection.

B. Calculation of 2D Plane Movement

The positional relationship between Visual Pencil and the computer in two adjacent frames is shown in Fig. 4. For the time being, suppose that the pen tip keeps moving in a plane parallel to the screen and the pen body is perpendicular to the screen during the use of Visual Pencil. The matched feature points in the two adjacent frames appear in the overlapping area of the Fig. 4. When the frame rate is high, the movement between the two adjacent frames can be approximately regarded as straight. The average moving distance of the matched feature points in the two adjacent frames is used to approximate the movement of the Visual Pencil's tip in the 2D plane. By that means the computer controls the cursor's movement proportionally.

In actual use, Visual Pencil will tilt to some extent. We correct the movement according to the tilt angle of the pen. When the frame rate is 30 fps, the pixel distance of the feature points in two adjacent frames does not exceed 8 pixels through experiments. When the pen tilt angle is within 20°, the corrected value is $8 \times \cos(20°) \approx 7.52$, and the pixel distance moved is still 8 pixels after rounding. Therefore, the device has tolerance for slight tilt. Directly calculating the average moving distance of the matched feature points seems overly simple, but it works. The pose of the camera does not need to be solved in this method, which speeds up the calculation process. In addition, when the camera tilts at the opposite angle, the slight error caused by the tilt will be eliminated naturally.

The moving distance is accumulated in real time to obtain the current position of the cursor. If the reset event is triggered, the cursor position is reset to the bottom-right corner of the screen; if a key event is triggered, the cursor is dragged to the current position; when no event occurs, the cursor is only moved to the current position. The reset event has higher priority than the key event. The moving distance is continuously calculated, and the corresponding operations are performed to track the Visual Pencil.

IV. EXPERIMENTAL TESTING AND RESULTS

A. Performance Evaluation Index

1) Frame Rate: It reflects the real-time performance of the image processing algorithm. The frame rate is both the refresh rate of captured images and the frequency of cursor movement. The higher the frame rate is, the smoother the user experience is. In the experiment, the frame rate is calculated by the following formula:

$$Frame\,rate = \frac{total\,number\,of\,frames}{total\,time} \tag{1}$$

2) Invalid Frame Ratio: It serves to measure the proportion of invalid frames. When Visual Pencil moves to the regions with few feature points, valid feature matches may not exist, and the program will lose control of the cursor during these frames. Therefore, the ratio of the number of invalid frames to the total number of frames can be used to measure the tracking loss of the cursor to the Visual Pencil. Invalid frame ratio is calculated by the following formula:

$$Invalid\,frame\,ratio = \frac{number\,of\,invalid\,frames}{total\,number\,of\,frames} \tag{2}$$

3) Straight Moving Offset (SMO): It measures the accuracy of the cursor tracking the Visual Pencil. If Visual Pencil moves along an absolute straight line ideally, the cursor will track Visual Pencil along a straight line and the cursor's path will be a straight line. We record the actual track of the cursor in the experiment and calculate the average offset of the track points from the straight line, which can measure the accuracy of the moving distance calculated by the algorithm. Straight moving offset is calculated by the following formula:

$$SMO = \frac{sum(|offset\ of\,the\,track\,points|)}{number\,of\,the\,track\,points} \tag{3}$$

Fig. 5 shows the poor performance of the device in three performance evaluation indicators. When the frame rate is too low, the cursor's moving path will not be smooth enough. When the invalid frame ratio is high, the cursor will lose tracking, and the cursor's moving distance is far less than the user's expected moving distance. When the straight moving offset is large, the calculated moving distance deviates greatly from the expected movement distance, indicating that the accuracy of the algorithm is low. The device parameters will be determined by these indicators.

B. Performance Evaluation

The main parameters that affect the performance of Visual Pencil are: the threshold for feature detection, the resolution of the images, and the distance of the pen tip from the screen. The threshold of feature detection determines the number of feature points detected. The higher the threshold is, the sparser

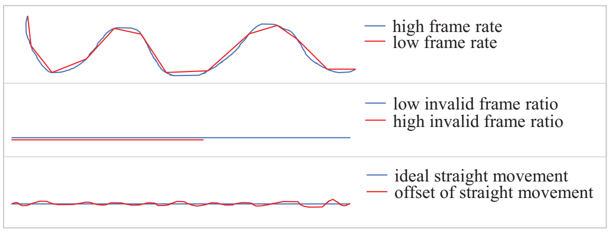


Fig. 5. Poor performance of the device reflected on three performance evaluation indicators.

the feature points are, but the faster the calculation speed is. Resolution impacts the calculation speed and accuracy of the algorithm. The distance of Visual Pencil from the screen determines the size of the camera's field of view. The larger the field of view is, the greater the number of valid feature points are. Therefore, considering the calculation accuracy and real-time performance, the above three parameters need to be considered comprehensively.

1) Threshold: We process two specific images with the same resolution as adjacent frames to measure the threshold's impact on performance. The running time t of one frame is recorded and the value of the frame rate is approximated by $1/t$. We draw the curve of the frame rate and the number of feature points as a function of the threshold, as shown in Fig. 6(a). When the threshold is around 70, the number of feature points is relatively larger and the frame rate is relatively higher.

2) Resolution of the Image: It measures the impact of image resolution on three performance evaluation indicators. We move Visual Pencil along a straight line and keep its pen tip in a plane parallel to the screen with the same moving trajectory and different resolutions to obtain video data streams. We run the program after inputting the video data, calculate the frame rate by the formula (1), and record the invalid frame ratio and the straight moving offset. The change of frame rate and straight moving offset with resolution is shown in Fig. 6(b). As the resolution decreases, the frame rate tends to increase. The invalid frame rate is 0 at the top 5 resolutions from 640×319 to 160×120. At the minimum resolution of 120×90, invalid frames occur and the invalid frame ratio is 28.16%, which is because the number of feature points in the images captured by the camera with excessively low resolution is too small, and the number of frames without effective feature matches increases. As the resolution decreases, the straight moving offset decreases first and then increases. This is because when the resolution is excessively low, the detection of feature points is not accurate enough and the offset increases. In summary, the resolution can be appropriately reduced for increasing the frame rate, but in order to ensure the accuracy, the excessively low resolution is not desirable. In the subsequent experiments, we set the resolution to 320×240 uniformly.

3) Distance of the Pen Tip from the Screen: It measures the impact of distance from the screen on three performance evaluation indicators. We change the distance of the Visual Pencil tip from the screen, adjust the camera to focus on the same position on the screen, and move the Visual Pencil along the same straight track. Based on the captured video data, three indicators are calculated, and the curves of the three indicators as a function of distance are plotted, as shown in Fig. 6(c). When the distance of the Visual Pencil tip from the screen is small, the algorithm is not stable enough, the accuracy is low, and the offset is large. According to the results of the experiment, the distance of the Visual Pencil tip from the screen is determined to be about 10 cm.

C. Experimental Results and Analysis

The cursor writes the text "ICMA" and draws a curve graphic under the control of Visual Pencil, as shown in Fig. 7 with the resolution set to 320×240, the threshold set to 100 and the pen tip to screen distance set to about 10 cm. The frame rate can reach 32 fps in the experiment.

Visual Pencil adapts well when it moves to complex areas on the screen. The accuracy of controlling the cursor moving reduces when Visual Pencil moves to simple areas because the feature points are sparse in these areas. Visual Pencil has a large field of view for it is 10 cm away from the screen. Therefore, when the Visual Pencil moves to the solid-color areas that often appears at the edge of the screen, the solid-color areas without feature points only occupy a part of the captured images. The movement of the Visual Pencil in a 2D plane can still be calculated by using these captured images. When Visual Pencil moves to the area that is invariant to a certain direction, it can no longer calculate the movement based on the captured images. This is caused by insufficient information contained in the captured images. This situation may occur when a blank drawing interface is just opened. Although some manual operations circumvent this transient situation, it is a better idea to add some auxiliary sensors to assist Visual Pencil get out of the short-term visual blind areas.

V. Conclusion

In this paper, we have proposed Visual Pencil, a novel vision-based handheld human-computer interaction design. The computer controls the cursor tracking the Visual Pencil by processing the images captured by the camera at the pen tip. The work in this paper is inspired by the visual odometer in SLAM, but it is not necessary to specifically solve the pose of the camera, which greatly reduces the running time. In addition, Visual pencil provides an effective interactive means for handwriting on non-touch screens.

In future work, sensors related to inertial navigation can be installed in the device. When the confidence of the image data is low, the data of the auxiliary sensor can be used for calculation. By that means the device's adaptability to the areas with sparse feature points may be enhanced. Furthermore, the device can track the 3D motion and accomplish 3D interactive tasks by incorporating deep vision.

Acknowledgment

This work was supported by National Natural Science Foundation of China (Grant Nos. 61725305, 61633020).

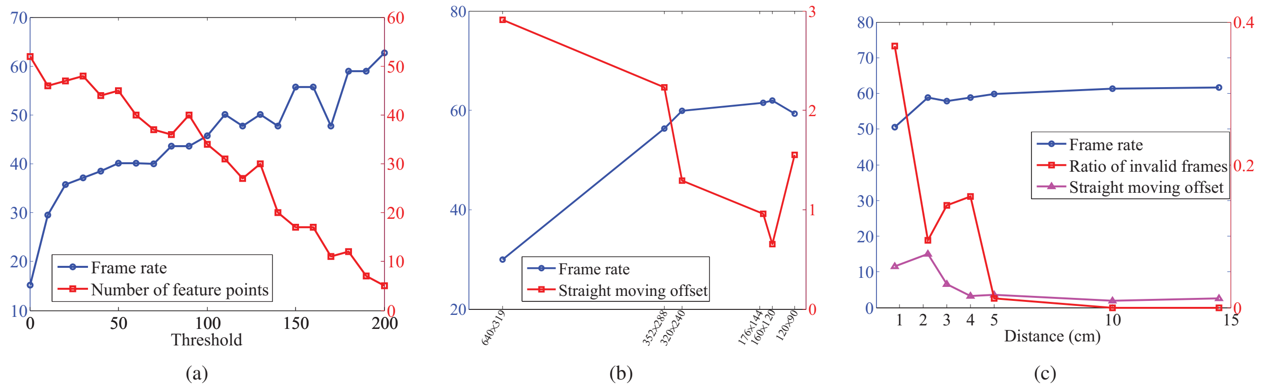


Fig. 6. The impact of three parameters on performance evaluation indicators. (a) Threshold. (b) Resolution. (c) Distance from the screen.

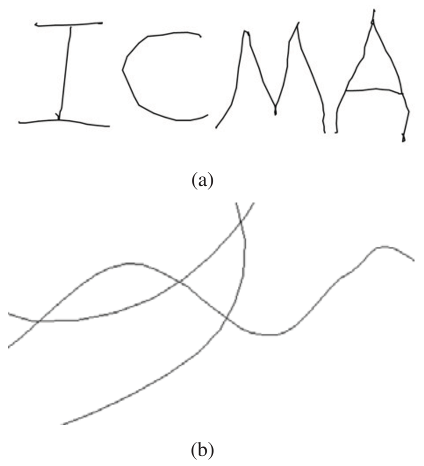


Fig. 7. The cursor's drawing and writing effects under the control of Visual Pencil. (a) The text "ICMA". (b) The curve graphics.

References

[1] X. Liu, Y. Wu, Y. Zhang, and G. Yang, "Optical measurement technology of nano-scale displacement," in *Proceedings of 2006 International Technology and Innovation Conference (ITIC 2006)*, Hangzhou, China, Nov. 2006, pp. 343–348.

[2] M. P. Peterson and T. Edso, "Coded paper for optical reading," China Patent 200 610 092 495.0, Dec. 13, 2006.

[3] T. Vuletic, A. Duffy, L. Hay, C. McTeague, G. Campbell, and M. Grealy, "Systematic literature review of hand gestures used in human computer interaction interfaces," *International Journal of Human-Computer Studies*, vol. 129, pp. 74–94, 2019.

[4] P. Bourdot, T. Convard, F. Picon, M. Ammi, D. Touraine, and J.-M. Vézien, "VR-CAD integration: Multimodal immersive interaction and advanced haptic paradigms for implicit edition of CAD models," *Computer-Aided Design*, vol. 42, no. 5, pp. 445–461, 2010.

[5] J. Kang, K. Zhong, S. Qin, H. Wang, and D. Wright, "Instant 3D design concept generation and visualization by real-time hand gesture recognition," *Computers in Industry*, vol. 64, no. 7, pp. 785–797, 2013.

[6] Leap motion. [Online]. Available at: https://www.leapmotion.com/

[7] B. S. Santos, J. Cardoso, B. Q. Ferreira, C. Ferreira, and P. Dias, "Developing 3D freehand gesture-based interaction methods for virtual walkthroughs: Using an iterative approach," in *Handbook of Research on Human-Computer Interfaces, Developments, and Applications*. IGI Global, 2016, pp. 52–72.

[8] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[9] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *Proceedings of 2011 International Conference on Computer Vision*, Barcelona, Spain, Nov. 2011, pp. 2564–2571.

[10] H.-J. Chien, C.-C. Chuang, C.-Y. Chen, and R. Klette, "When to use what feature? SIFT, SURF, ORB, or A-KAZE features for monocular visual odometry," in *Proceedings of 2016 International Conference on Image and Vision Computing New Zealand (IVCNZ)*, Palmerston North, New Zealand, Nov. 2016, pp. 1–6.

[11] J. Bian, W.-Y. Lin, Y. Matsushita, S.-K. Yeung, T.-D. Nguyen, and M.-M. Cheng, "Gms: Grid-based motion statistics for fast, ultra-robust feature correspondence," in *Proceedings of 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, Jul. 2017, pp. 4181–4190.