Handwriting-Assistant: Reconstructing Continuous Strokes with Millimeter-level Accuracy via Attachable Inertial Sensors

YANLING BU, State Key Laboratory for Novel Software Technology, Nanjing University, China
LEI XIE*, State Key Laboratory for Novel Software Technology, Nanjing University, China
YAFENG YIN, State Key Laboratory for Novel Software Technology, Nanjing University, China
CHUYU WANG, State Key Laboratory for Novel Software Technology, Nanjing University, China
JINGYI NING, State Key Laboratory for Novel Software Technology, Nanjing University, China
JIANNONG CAO, The Hong Kong Polytechnic University, China
SANGLU LU, State Key Laboratory for Novel Software Technology, Nanjing University, China

Pen-based handwriting has become one of the major human-computer interaction methods. Traditional approaches either require writing on the specific supporting device like the touch screen, or limit the way of using the pen to pure rotation or translation. In this paper, we propose *Handwriting-Assistant*, to capture the free handwriting of ordinary pens on regular planes with mm-level accuracy. By attaching the inertial measurement unit (IMU) to the pen tail, we can infer the handwriting on the notebook, blackboard or other planes. Particularly, we build a generalized writing model to correlate the rotation and translation of IMU with the tip displacement comprehensively, thereby we can infer the tip trace accurately. Further, to display the effective handwriting during the continuous writing process, we leverage the principal component analysis (PCA) based method to detect the candidate writing plane, and then exploit the distance variation of each segment relative to the plane to distinguish on-plane strokes. Moreover, our solution can apply to other rigid bodies, enabling smart devices embedded with IMUs to act as handwriting tools. Experiment results show that our approach can capture the handwriting with high accuracy, *e.g.*, the average tracking error is 1.84mm for letters with the size of about 2cm×1cm, and the average character recognition rate of recovered single letters achieves 98.2% accuracy of the ground-truth recorded by touch screen.

CCS Concepts: • **Human-centered computing → Ubiquitous and mobile computing**.

Additional Key Words and Phrases: Handwriting Reconstruction, Inertial Sensor, Millimeter-level Tracking

ACM Reference Format:
Yanling Bu, Lei Xie, Yafeng Yin, Chuyu Wang, Jingyi Ning, Jiannong Cao, and Sanglu Lu. 2021. Handwriting-Assistant: Reconstructing Continuous Strokes with Millimeter-level Accuracy via Attachable Inertial Sensors. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 5, 4, Article 146 (December 2021), 25 pages. https://doi.org/10.1145/3494956

*Lei Xie is the corresponding author.

Authors' addresses: Yanling Bu, State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China, yanling@smail.nju.edu.cn; Lei Xie, State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China, lxie@nju.edu.cn; Yafeng Yin, State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China, yafeng@nju.edu.cn; Chuyu Wang, State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China, chuyu@nju.edu.cn; Jingyi Ning, State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China, ningjy@smail.nju.edu.cn; Jiannong Cao, The Hong Kong Polytechnic University, Hong Kong, China, jiannong.cao@polyu.edu.hk; Sanglu Lu, State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China, sanglu@nju.edu.cn.

2474-9567/2021/12-ART146 $15.00
https://doi.org/10.1145/3494956

1 INTRODUCTION

Nowadays, as one of the most common and natural ways to record information, handwriting has been the major human-computer interaction (HCI) method, especially for the pen-based handwriting. A series of smart pens have emerged to assist the handwriting-based HCI, *e.g.*, Apple Pencil [2] and Samsung S Pen [10]. Smart devices can capture the handwriting through these smart pens, so as to provide the fundamental data for handwriting input or recognition. Particularly, *handwriting input* means directly leveraging the original writing trace as itself, serving for signing names, checking stroke orders, drawing pictures, *etc. Handwriting recognition* means inferring the characters based on the input writing trace, mainly serving for entering text. Despite different forms of application scenarios, the basis of handwriting-based HCI is to capture the handwriting accurately. Note that, these commercial smart pens usually require users to write on specific supporting devices, *e.g.*, tablet [12], touch screen [2, 10] or special paper [1]. There are two main drawbacks of using the supporting devices as the writing plane. First, the size of supporting devices can greatly affect the user experience, *i.e.*, the small size provides good portability but poor writing experience, while the big size does the opposite, it is difficult to select a suitable size to balance the portability and writing experience. Second, the supporting devices are not always available or usable. In one case, as people get used to writing with ordinary pens on paper or board in most scenarios, both the analog and digital handwriting need to be recorded. For example, cartoonists create drawings on the paper, users record important items on sticky notes at hand when on the phone, and children write on the paper when learning the stroke order. Since the analog handwriting is hard to edit, easy to lost, inconvenient to search and lack of timestamp, it is in great demand to record the analog handwriting digitally just when writing [12], but it can be hardly achieved if requiring supporting devices to write on. In the other case, as supporting devices are hard to deploy everywhere, the usage of commercial smart pens is not flexible due to the dependence of these devices, *e.g.*, it is inconvenient for users wearing Microsoft Hologens to use the touch screen for text input. Therefore, to benefit the handwriting-based HCI in a more ubiquitous manner, we aim to capture the accurate writing trace of ordinary pens for smart devices without the requirement of writing on specific devices.

Recently several studies have focused on capturing the handwriting of independent pens using magnetic or inertial sensors. MagHacker [27] uses the magnetic signal to sense the handwriting, but as the magnetic signal is easily distorted indoors, the captured handwriting does not have high accuracy. Since the inertial measurement unit (IMU) is low-cost, small-size, light-weight, sensitive to motion and robust to indoor environments, the IMU can be exploited to capture the accurate handwriting. The IMU-based solutions usually assume the pen motion is in the scope of pure translation or pure rotation. Thus, the writing trace is regarded as the IMU trace by ignoring the trace difference due to rotation [30, 35], or calculated based on the IMU rotation by ignoring the translation of rotating center [14, 23]. In fact, the pen motion generally includes both rotation and translation, such assumptions limit the application scope of these solutions and degrade the accuracy of captured handwriting. Meanwhile, prior work only considers the handwriting with a single stroke, yet the handwriting consists of multiple strokes during the continuous writing process. The tip motion between adjacent strokes in the air leads to off-plane traces, which indicate the relative position of strokes and should be removed. Therefore, to capture the handwriting with multiple strokes accurately, on one hand, we should consider the *rotation* and *translation* of pen motion comprehensively, on the other hand, we need to capture the exact *handwriting on the writing plane*.

In this paper, we propose *Handwriting-Assistant*, an IMU-based approach for capturing the handwriting of ordinary pens with mm-level accuracy, as illustrated in Fig. 1. Specifically, we attach the inertial sensor to the customized pen cap, such that the inertial sensor can be easily fixed to the tail of the ordinary pen without affecting the writing experience. To derive the relative position of the pen tip to the IMU, we rotate the pen around the tip as the initialization action of *Handwriting-Assistant*, which only needs to be conducted once in advance. We observe that due to different writing habits of users, the motion of pen is various during the writing process, but the relative motion between the pen tip and the pen tail conforms to the rigid motion. By considering

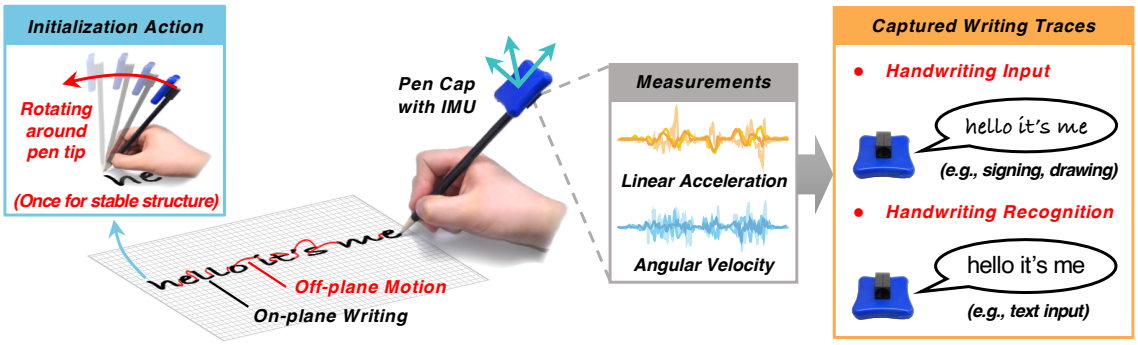


Fig. 1. Illustration of *Handwriting-Assistant* with inertial sensors

the rotation and translation of IMU comprehensively, we build a generalized model to infer the tip trace in the 3D space from the IMU motion. Moreover, during the continuous writing process, apart from the handwriting on the plane, denoted as *on-plane writing*, the pen tip is usually lift up after finishing one stroke and moved to the start position of next stroke, denoted as *off-plane motion*. The tip trace of on-plane writing is our target to capture, while that of off-plane motion indicates the relative position between on-plane strokes and needs to be removed from the captured tip traces. To distinguish the on-plane tip trace from the off-plane tip trace, we propose the principal component analysis (PCA) based method to detect the writing plane, and use the distance variation of the tip relative to the writing plane to distinguish traces. In this way, we are able to effectively project the tip trace from the 3D space onto the writing plane, and derive the effective handwriting with mm-level accuracy.

There are three challenges to realize *Handwriting-Assistant*. 1) *How to capture the handwriting of ordinary pen from the motion of IMU attached on the pen tail with mm-level accuracy?* Although the IMU trace is highly correlated to the tip trace, it is not directly consistent with the tip trace, since the IMU and the tip are at different positions of the pen. To tackle this challenge, since the relative motion of the pen tip and the pen tail conforms to the rigid body motion, we build a generalized model to infer the precise tip trace based on the rotation and translation of IMU at the pen tail in a comprehensive way. By continuously tracking the IMU motion, we can infer the tip trace in the 3D space regardless of different writing habits. 2) *How to determine the writing plane from the captured tip traces in the 3D space?* To drive the 2D handwriting from the 3D tip traces, it is essential to determine the writing plane by the captured tip traces. To tackle this challenge, we propose a PCA-based method to find the candidate writing plane. Specifically, the short time energy (STE) is adopted to split the data and extract motion segments. Note that the motion segments can be caused by either the on-plane writing or the off-plane motion. We first leverage the PCA algorithm to estimate the principal plane for each motion segment. Then, by calculating the included angle between principal planes of motion segments, we cluster the planes with small included angles, and use the common plane of most segments as the candidate writing plane. 3) *How to distinguish the tip trace caused by the on-plane writing from the off-plane motion, so as to reconstruct the effective handwriting of multiple strokes?* During the continuous writing process, the handwriting is usually constituted of multiple strokes, yet the off-plane motion between strokes leads to additional tip traces instead of the actual handwriting, we have to distinguish the motion of tip traces. However, due to the speed of the pen during the off-plane motion is not definitely larger than that during the on-plane writing, the traditional motion-intensity-based approaches are not always reliable for the motion separation. To tackle this challenge, we propose a height-variation-based scheme to distinguish motion segments. The intuition is that, during the on-plane writing process, the pen tip is always on the writing plane, whereas during the off-plane motion process, the pen tip leaves the writing plane first, then returns. Therefore, we use the distance variation of the pen tip relative to the writing plane, denoted as *height variation*, to filter out off-plane segments and capture the exact handwriting.

We make three key contributions in this paper. Firstly, we provide a novel input mode for smart devices by capturing the free handwriting of ordinary pens on regular planes with mm-level accuracy. Through attaching the pen cap with IMU to the tail of an ordinary pen, we can effectively infer the tip trace from the IMU motion based on the rigid-motion-based writing model. Moreover, the writing model can apply to other rigid body used as the pen, enabling smart devices embedded with IMU to act as handwriting tools. Secondly, we propose the solution to capturing the effective handwriting with multiple strokes during the continuous writing process. We employ the PCA-based method to detect the writing plane, and further leverage the height variation to identify whether the motion segment is on-plane or off-plane, thereby we can extract the effective handwriting without redundant off-plane traces. Thirdly, we have implemented a prototype system of *Handwriting-Assistant* and evaluated its performance in the real environment. Experiment results show that our approach can capture the handwriting with high accuracy, *e.g.*, the average tracking error is 1.84mm for letters with the size of about 2cm×1cm, and the average character recognition rate of captured single letter achieves 98.2% accuracy of the ground-truth recorded by touch screen.

2 RELATED WORK

2.1 Commercial Pen-based Handwriting Solutions

With the great demand for capturing the handwriting digitally, many manufacturers have introduced commercial devices to assist the handwriting process. Most of these solutions involve a customized pen and its supporting device, *e.g.*, tablet [12], touch screen [2, 10] or special paper [1]. For example, Wacom provides the tablet [12] to capture the handwriting of the specific pen, Apple provides the Apple Pencil [2] to work with iPad, and Anoto provides the Livescribe smartpen [1] to turn the handwritten information on the dot paper into digital ones. However, as the writing area of the supporting device is limited, it is inconvenient for users especially when performing the continuous handwriting. Moreover, the smart pen and its supporting devices are not always available or usable [28] in most scenarios, people get used to writing with ordinary pens on the paper, board or other regular planes. Compared to them, *Handwriting-Assistant* can capture the handwriting of ordinary pens on the regular writing plane digitally, without the additional equipment except the ordinary pen attached with IMU.

2.2 Novel Sensor-based Handwriting Solutions

Many studies employ sensors to develop independent digital pens, including *classification-based* or *trace-based*.

For classification-based solutions, they directly classify the handwriting into specific characters or gestures based on the measurements. Specifically, they extract features from acoustic [16, 31, 38, 42, 44] or inertial [22, 24, 26, 29, 36, 39, 41, 45] signals, and then feed features into classifiers to infer characters or gestures. However, the classification-based solutions require large amounts of training data, which would cost much human efforts but still hardly achieve good accuracy for brand-new users or environments. Indeed, compared with the extracted writing features, the trajectory of the same character or gesture is relatively stable for untrained cases. Meanwhile, as there are many off-the-shelf handwriting recognition platforms, it is easy to realize the robust handwriting recognition for any user in any environment if obtaining the precise writing traces. Thus, rather than directly classifying the handwriting, we prefer building models to capture the accurate handwriting in a general case.

For trace-based solutions, they focus on recovering the handwriting as we expect. The captured handwriting can serve for many applications, *e.g.*, checking stroke orders, drawing pictures, making notes, signing names and verifying signatures [17, 18, 25], while the most common one is character recognition. For example, MagHacker [27] leverages the magnetic signal to sense the handwriting. Since the magnetic field is easily distorted indoors, the captured handwriting of MagHacker is not very accurate, but fine for character recognition. Moreover, as the IMU is low-cost, small-size, light-weight, sensitive to motion and robust to indoor environments, it is more reliable to

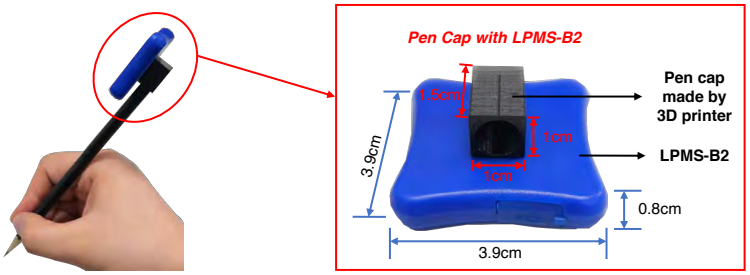


Fig. 2. Pen cap with inertial sensor (LPMS-B2 as an example)

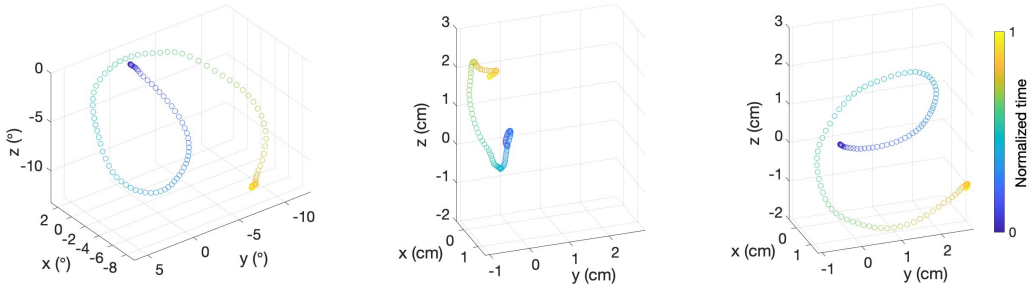
capture the accurate handwriting based on IMU. According to the assumption of pen motion during the writing process, traditional IMU-based handwriting reconstruction solutions can be mainly divided into two categories:

- *Pure-translation-based Solutions:* Assume the pen motion is pure translation [13, 30, 35, 43], hence the writing trace is directly regarded as the IMU trace by ignoring the trace difference due to the pen rotation.
- *Pure-rotation-based Solutions:* Assume the pen motion is pure rotation [14, 23], hence the writing trace is calculated based on the IMU rotation by ignoring the translation of rotating center.

Particularly, both work [30] and Gyropen [14] take the smartphone embedded with IMU as a pen. Work [30] thinks the IMU trace as the writing trace, which would be accurate only when the pen tip and the IMU are at the same position or there is no pen rotation during the writing process. However, there exists the position difference between the pen tip and the IMU, and the writing process usually incurs the attitude change of the pen. Gyropen [14] assumes there is a fixed rotation center, and leverages the rotation of smartphone to reconstruct the writing trace only on the horizontal plane. However, the writing process usually also incurs the position change of the whole pen, and the writing plane is not definitely horizontal, *e.g.*, the blackboard is vertical. Therefore, if simplifying the pen motion, the accuracy of captured handwriting will degrade significantly for the free writing. Meanwhile, these solutions only consider the handwriting with a single stroke, but during the continuous writing process, the tip motion between adjacent strokes in the air leads to off-plane traces, which indicate the relative position of strokes and should be removed for deriving the exact handwriting with multiple strokes. Overall, compared to previous IMU-based solutions, we consider the rotation and translation of the pen comprehensively, and deal with the continuous handwriting with multiple strokes. In this way, *Handwriting-Assistant* can accurately capture the exact free handwriting on any plane during the continuous writing process.

3 EMPIRICAL STUDY

We conduct the empirical study to investigate the challenges and opportunities of inferring the handwriting by attaching the inertial sensor to the ordinary pen. To power the IMU and transfer the measurements wirelessly, the size of commercial inertial sensor is usually at cm-level [6]. To attach the sensor to the ordinary pen without affecting the writing comfort, we prefer the pen tail to attach the sensor instead of the pen tip. In our studies, without loss of generality, we select the LPMS-B2, a 9-axis Bluetooth IMU of size $3.9\times3.9\times0.8\text{cm}^3$ [6], as the inertial sensor. We keep collecting the angular velocity and linear acceleration samples during the writing process. The size of inertial sensor could be smaller if customizing the sensor in a compact form [15, 20, 29, 31, 32], we would improve it in the future. For the convenience of fixing the sensor to the pen tail, we design a pen cap made by 3D printer as shown in Fig. 2. The pen cap with IMU weighs around 13 grams, without influencing the normal writing. By easily attaching the pen cap to the ordinary pen, we can use the pen to write on the paper as usual, and capture the digital handwriting from the IMU measurements simultaneously. In order to capture the handwriting with multiple strokes on the writing plane, we need to deal with two main technical issues:



(a) Rotation of IMU (b) Translation of IMU (c) Tip trace

Fig. 3. Inconsistent tip trace from IMU trace

1) *Writing Model:* What is the mathematical relationship between the tip trace and the IMU measurement when attaching the IMU to the pen tail?

2) *Motion Feature:* What kind of motion features can be used to distinguish the on-plane writing from the off-plane motion between different strokes?

3.1 IMU Motion during Writing Process

To tackle technical issue 1 "writing model", we first conduct an experimental study to evaluate the relationship between the tip trace on the writing plane and the IMU motion in the 3D space. We attach the pen cap to the tail of a pencil, and use the pencil to write a letter 'e' with the size of 3.5×3.5cm.

Observation 1: The trace of pen tip is usually not directly consistent with the trace of pen tail due to the rotation of pen during the writing process. Nevertheless, the tip trace is highly correlated to the IMU motion.

Fig. 3 plots the IMU motion in the 3D space during the writing process, including the rotation and translation. It is observed that the translation of IMU in Fig. 3(b) is different from the tip trace in Fig. 3(c), while the rotation pattern of IMU in Fig. 3(a) is correlated to the tip trace to a certain extent. This is because the size of writing trace is small, the IMU motion is mainly in the scope of the rotation, so the trace of IMU at the pen tail is much different from the tip trace. However, it does not mean we can ignore the translation of IMU. On one hand, although the rotation pattern is correlated to the tip trace, we cannot determine the trace size only based on the rotation pattern. On the other hand, not all handwriting mainly incurs the rotation of IMU. When enlarging the size of handwriting, *e.g.*, a 20×20cm square, the IMU motion will be mainly in the scope of the translation, such that the tip trace is mainly correlated to the IMU translation. Therefore, it is a challenge to infer the precise tip trace based on the IMU motion regardless of whether the IMU motion is mainly in the scope of the rotation or translation. To handle the issue, it is essential to build a writing model that correlates the rotation and translation of IMU with the tip trace comprehensively, so as to infer the precise tip trace based on the IMU motion.

3.2 On-plane Motion versus Off-plane Motion

When making notes or drawing pictures, people usually cannot finish the handwriting with only one stroke. According to whether the pen tip writes on the plane, the motion of pen can be divided into *on-plane writing* and *off-plane motion*, as shown in Fig. 4. The *on-plane writing* happens when the pen tip writes on the writing plane, while the *off-plane motion* happens when the hand lifts up the pen tip and moves the pen tip to the start position of next stroke. Consequently, the on-plane strokes are our targets to capture, while the off-plane traces indicate the relative position of two adjacent on-plane strokes. To tackle technical issue 2 "motion feature", we also conduct an experimental study by writing a word 'hello' sequentially, and investigate the difference of IMU readings between the on-plane and off-plane motion.

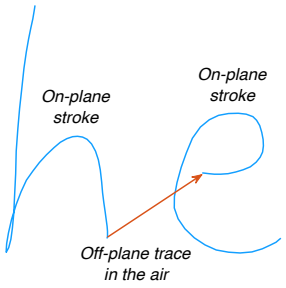


Fig. 4. On-plane v.s. off-plane

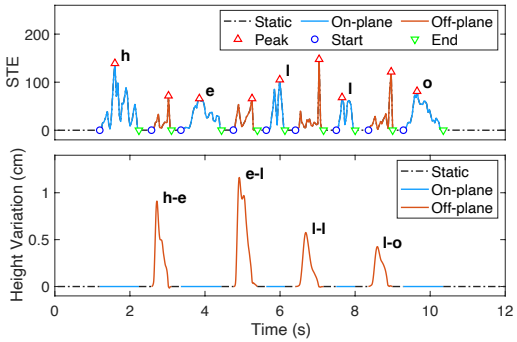


Fig. 5. Short time energy and height variation when writing 'hello'

Observation 2: The energy of IMU readings can be used to effectively segment data and extract motion segments, but cannot be used to reliably identify whether the motion segment is on-plane or off-plane. Nevertheless, the height variation can be used to reliably distinguish the motion segment.

During the motion period, either the angular velocity or the linear acceleration changes significantly compared to the static period when the user holds the pen without moving, leading to the large energy of data. Fig. 5 shows the short time energy (STE) of IMU readings when writing 'hello' (the calculation of STE will be introduced in Section 5). The STE reflects the changing rate of data: The more intense the changing rate is, the larger the STE is. In previous works [27, 45], researchers usually think the off-plane motion tends to be more intense, leading to the large STE accordingly. Nevertheless, such assumption greatly relies on the writing habit, which is not always valid. As shown in the top figure of Fig. 5, the on-plane writing and off-plane motion appear alternately, but the STE of off-plane motion has no obvious difference from that of on-plane writing. Instead of focusing on the motion intensity, we observe that the off-plane motion can incur the distance variation of the pen tip relative to the writing plane, denoted as *height variation*. We leverage OptiTrack [9] to capture the height variation of pen tip during the writing process. As shown in the bottom figure of Fig. 5, it is observed that the height variation of on-plane writing keeps zero due to the constant touch between the pen tip and the writing plane, while the height variation of off-plane motion increases first then decreases due to the motion of lifting up and pinning down. Therefore, we can leverage the motion intensity, *i.e.*, energy of data, to segment data and extract motion segments [27, 33, 34, 45]. Further, we prefer the height variation to distinguish whether the motion segment is on-plane or off-plane. In this way, we can effectively remove the traces caused by the off-plane motion and reconstruct the on-plane handwriting consisting of multiple strokes.

4 SYSTEM OVERVIEW

The main goal of our work is to capture the handwriting when a user makes notes on the paper or board with an ordinary pen. The captured handwriting can provide a lot of services, like recording the signature, or recognizing the input to smart devices. To achieve the goal, we attach an inertial sensor, *i.e.*, LPMS-B2, to the pen tail, such that we can ensure the user experience and track the handwriting of ordinary pens, without requiring specialized devices like a stylus pen or the touch screen.

The basic idea is to infer the trace of pen tip on the 2D writing plane from the motion of IMU in the 3D space. Fig. 6 illustrates the architecture of our system. With the collected readings from gyroscope and accelerometer, *i.e.*, angular velocity and linear acceleration, we design three modules to recover the handwriting. Firstly, the *Motion Extraction* module calculates the Short Time Energy (STE) of raw data from gyroscope and accelerometer to extract the motion segments. Note that, both the on-plane writing and the off-plane motion can incur the significant STE, so we only extract the motion segments in this step, without distinguishing whether the motion

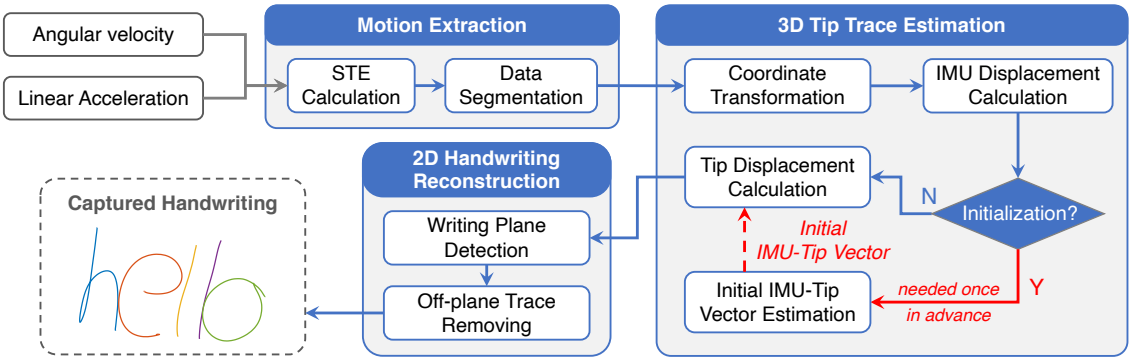


Fig. 6. System architecture

is on-plane or off-plane. Secondly, the *3D Tip Trace Estimation* module individually processes each segment to estimate the tip trace in the 3D space. It leverages the angular velocity to track the attitude change of the pen with IMU, and calculates the IMU displacement by converting the collected linear acceleration samples to the fixed coordinate system. Then, it utilizes the proposed writing model to calculate the tip displacement in the 3D space. Moreover, the tip position in the body coordinate system of IMU, *i.e.*, the initial IMU-tip vector, is needed for calculating the tip displacement, thereby, we should perform the initialization action to estimate the initial IMU-tip vector before capturing the handwriting. For the pen with a fixed IMU, the initialization only needs to be done once in advance. Thirdly, the *2D Handwriting Reconstruction* module distinguishes the on-plane writing from the off-plane motion by detecting the height variation relative to the writing plane. The off-plane motion is used to determine the relative position between adjacent on-plane writing strokes, such that we can recover the handwriting consisting of multiple strokes with high accuracy.

5 MOTION EXTRACTION

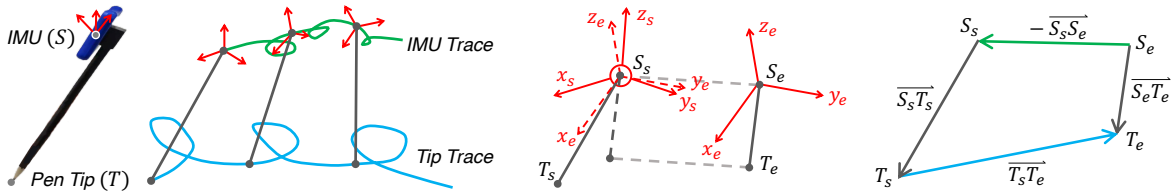
To recover the handwriting, it is necessary to extract the data caused by the on-plane writing and the off-plane motion during the handwriting process precisely. The on-plane writing leads to exact strokes, while the off-plane motion indicates the relative position between strokes. In this section, we introduce how to extract the motion segments from IMU readings based on the short time energy (STE).

5.1 STE Calculation

The intuition of using STE to segment data is that, during the writing process, the user usually keeps still for a little while after finishing one action, which we call *rest period*. Hence, we can leverage the low STE of rest period to segment the IMU readings for each action [19]. The rest period can be very short for the successful segmentation, *i.e.*, around 0.25s in Fig. 5, requiring little time cost or human effort [26]. As mentioned in the empirical study, the motion of pen may lead to the obvious rotation or translation or both of IMU at the pen tail. The IMU readings during the motion period are much more intense than the rest period. Thus, we calculate the STE of data with both angular velocity samples and linear acceleration samples.

Specifically, denote the linear acceleration as $\boldsymbol{a} = [a_x, a_y, a_z]^T$, the angular velocity as $\boldsymbol{\omega} = [\omega_x, \omega_y, \omega_z]^T$, so their amplitudes are calculated as: $|\boldsymbol{a}| = \sqrt{a_x^2 + a_y^2 + a_z^2}$, $|\boldsymbol{\omega}| = \sqrt{\omega_x^2 + \omega_y^2 + \omega_z^2}$, respectively. For the i^{th} sample, the combined amplitude can be represented as:

$$e(i) = |\boldsymbol{a}(i)| + k|\boldsymbol{\omega}(i)|, \tag{1}$$



(a) Rigid motion of pen with IMU during writing process (b) Attitude change in 3D space (c) Position change in 3D space

Fig. 7. Model the relationship between tip displacement and IMU motion

where k is a constant to balance the amplitude difference. Further, the STE is adopted to depict the energy of the combined amplitude using a sliding window. For the f^{th} frame, the signal energy is calculated as:

$$E(f) = \sum_{i=1}^{N} e_f^2(i), \tag{2}$$

where N means the sample number in a frame, $e_f(i)$ represents the combined amplitude of the i^{th} sample in the f^{th} frame. Based on the STE of IMU readings, we next separate the data and extract motion segments.

5.2 Data Segmentation

Because the motion segment has more energy than the rest segment, we divide the data by searching for the consecutive readings with energy above the threshold. Specifically, we first find the peaks of STE with certain time interval, as the red triangles plotted in Fig. 5. The certain time interval can filter out many redundant peaks within the motion segment to save the efforts of seeking split points around the peaks. For each peak, we search for the split points before and after the consecutive readings with energy below a threshold e_{th}. For example, to find the start point of a motion segment, we forward seek the split point from the peak by checking whether its neighboring energies are all below the threshold. The peaks in the same motion segment would indicate the same pair of split points. In this way, we are able to extract the motion segment effectively.

6 3D TIP TRACE ESTIMATION

6.1 Tip Displacement Calculation

During the writing process, the position and attitude of the pen are changing continuously along with time. For a pen with IMU, the attitude change of the pen is identical to the IMU, and the position change of the pen is related to the IMU. As shown in Fig. 7(a), a pen with IMU can be regarded as a line segment with the IMU and the pen tip as endpoints, denoted as S and T, respectively. For the inertial sensor, the point S indicates the origin of its body coordinate system. Due to the stable structure of the pen with IMU, the relative motion of the pen tip and the pen tail conforms to the rigid body motion, so the position of pen tip T is fixed in the body coordinate system of IMU. That is, regardless of the changing attitude or position of the pen in the fixed coordinate system, the coordinate value of pen tip T is constant in the changing body coordinate system. Based on such stability, we are able to infer the tip trace based on the IMU motion, including the attitude change and the position change.

Specifically, as shown in Fig. 7(b), suppose from time t_s to t_e, the IMU-tip segment moves from S_sT_s to S_eT_e, $|S_sT_s| = |S_eT_e|$. Denote the body coordinate system at time t_s as the global coordinate system, the body coordinate system at time t_e as the local coordinate system. Due to the attitude change, vector $\overrightarrow{S_eT_e}$ is different from $\overrightarrow{S_sT_s}$ in the global coordinate system, which can be represented as:

$$\overrightarrow{S_eT_e} = \mathbf{C}\,\overrightarrow{S_sT_s}, \tag{3}$$

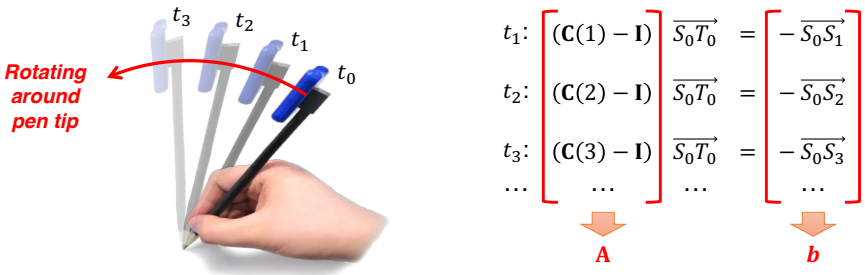


Fig. 8. Estimate initial IMU-tip vector

where $\mathbf{C}$ is a 3×3 rotation matrix, specifying the attitude of the local coordinate system relative to the global coordinate system. While for the position change in the global coordinate system, as shown in Fig. 7(c), the vector $\overrightarrow{S_eT_e}$ can be divided into:

$$\overrightarrow{S_eT_e} = -\overrightarrow{S_sS_e} + \overrightarrow{S_sT_s} + \overrightarrow{T_sT_e}, \tag{4}$$

where $\overrightarrow{S_sS_e}$ means the IMU displacement from t_s to t_e, $\overrightarrow{S_sT_s}$ means the IMU-tip vector at time t_s, $\overrightarrow{T_sT_e}$ means the tip displacement from t_s to t_e. By substituting Eq.(4) into Eq.(3), the tip displacement can be calculated as:

$$\overrightarrow{T_sT_e} = (\mathbf{C}-\mathbf{I})\,\overrightarrow{S_sT_s} + \overrightarrow{S_sS_e}, \tag{5}$$

where $\mathbf{I}$ is an identity matrix. According to Eq.(5), we are able to infer the tip displacement from t_s to t_e based on the IMU-tip vector at time t_s, the rotation and displacement of IMU. During the writing process, we can derive the tip trace by continuously tracking the tip displacement relative to the initial tip position.

Note that, although the rotation and displacement of IMU can be measured by the gyroscope and accelerometer, the IMU-tip vector at time t is not easy to be measured. Due to the tip position is fixed in the body coordinate system of IMU, the body IMU-tip vector is constant. Therefore, for the writing process, let the initial body coordinate system be the global coordinate system and t_s represent the initial time, thus the IMU-tip vector at time t_s only relies on the structure of the pen with IMU, denoted as the *initial IMU-tip vector*. As the structure is stable for the pen with IMU, the initial IMU-tip vector is constant for a random writing process, so we only need to estimate the initial IMU-tip vector once in advance as the initialization step of capturing the handwriting.

6.2 Initial IMU-Tip Vector Estimation

The initial IMU-tip vector reflects the position of pen tip in IMU's coordinate system. However in practice, it is hard to find the origin point of IMU or determine the projected positions of pen tip along each axis of IMU's coordinate system, so it is difficult to derive the accurate initial IMU-tip vector through the physical measurement. Even if the vector is measured physically, it is significant to verify its accuracy. Therefore, to derive or validate the initial IMU-tip vector, we propose an initialization operation to estimate the IMU-tip vector.

Particularly, we fix the pen tip and let the pen rotate around the tip, as depicted in Fig. 8. As the pen tip is at the same position during the rotation, the tip displacement is always equal to zero, Eq.(5) can be rewritten as:

$$(\mathbf{C}(t)-\mathbf{I})\,\overrightarrow{S_0T_0} = -\overrightarrow{S_0S_t}, \tag{6}$$

where $\overrightarrow{S_0T_0}$ is the initial IMU-tip vector to be estimated, $\mathbf{C}(t)$ and $\overrightarrow{S_0S_t}$ are the rotation matrix and IMU displacement at time t, which can be measured from gyroscope and accelerometer, respectively. In principle, one equation could be enough for solving Eq.(6), but it would be not accurate due to the measurement error of IMU, so we use multiple samples to find the optimal solution of $\overrightarrow{S_0T_0}$. Assume we obtain N samples during the rotation process,

let $\boldsymbol{x} = \overrightarrow{S_0T_0}$, $\mathbf{A} = \begin{bmatrix} \mathbf{C}(1) - \mathbf{I} \\ \mathbf{C}(2) - \mathbf{I} \\ \cdots \\ \mathbf{C}(N) - \mathbf{I} \end{bmatrix}$, $\boldsymbol{b} = \begin{bmatrix} -\overrightarrow{S_0S_1} \\ -\overrightarrow{S_0S_2} \\ \cdots \\ -\overrightarrow{S_0S_N} \end{bmatrix}$, the optimal solution of $\boldsymbol{x}$ is determined by:

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} ||\mathbf{A}\boldsymbol{x} - \boldsymbol{b}||^2. \tag{7}$$

To solve Eq.(7), we leverage the least squares method, the solution is calculated as:

$$\hat{\boldsymbol{x}} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\boldsymbol{b}. \tag{8}$$

In this way, we can determine the initial IMU-tip vector for the pen with IMU. Due to the stable structure of the pen with IMU, the initial IMU-tip vector only needs to be estimated once. Moreover, this estimation can apply to other rigid body used as the pen. We provide some examples in Section 8.8.2. With the known initial IMU-tip vector, we can infer the tip displacement based on the rotation and displacement of IMU based on Eq.(5). Note that, to derive the tip trace or the initial IMU-tip vector, it is essential to obtain the accurate rotation matrix and IMU displacement. In the following, we introduce how to calculate the rotation matrix and IMU displacement.

6.3 Coordinate Transformation

The IMU readings are measured in its body coordinate system, as the IMU rotates during the writing process, it is necessary to transform the IMU readings into a fixed coordinate system. Because the indoor magnetic field is easily disturbed, we only use the angular velocity measured by gyroscope to track the attitude change of IMU.

Specifically, during the writing process, we select the initial body coordinate system of IMU as the fixed coordinate system, denoted as the global coordinate system of the current handwriting. The attitude of IMU relative to the global coordinate system can be tracked by the integration of angular velocity [37]. In particular, if the attitude of IMU at time t is $\mathbf{C}(t)$, the attitude at time $t + \delta t$ can be updated with the angular velocity $\boldsymbol{\omega}_b = \left[\omega_{bx}, \omega_{by}, \omega_{bz}\right]^T$ in this period, as:

$$\mathbf{C}(t + \delta t) = \mathbf{C}(t)\left(\mathbf{I} + \frac{\sin\sigma}{\sigma}\mathbf{B} + \frac{1 - \cos\sigma}{\sigma^2}\mathbf{B}^2\right), \tag{9}$$

where $\mathbf{I}$ is an identity matrix, $\sigma = |\boldsymbol{\omega}_b\delta t|$, $\mathbf{B} = \begin{bmatrix} 0 & -\omega_{bz}\delta t & \omega_{by}\delta t \\ \omega_{bz}\delta t & 0 & -\omega_{bx}\delta t \\ -\omega_{by}\delta t & \omega_{bx}\delta t & 0 \end{bmatrix}$. Because we take the initial body coordinate system as the global one, we have: $\mathbf{C}(0) = \mathbf{I}$. When the new angular velocity sample comes, the rotation matrix from the current body coordinate system to the global coordinate system can be updated based on Eq.(9). After the long time tracking, the rotation error may accumulate, we can leverage the orientation calibration method proposed in [40] to reduce the accumulated error, so as to obtain the accurate rotation matrix all the time.

6.4 IMU Displacement Calculation

With the rotation matrix, we can further calculate the IMU displacement in the global coordinate system based on the linear acceleration. Let the measured linear acceleration in the body coordinate system be $\boldsymbol{a}_b$, the corresponding linear acceleration in the global coordinate system should be $\boldsymbol{a}_g = \mathbf{C}\boldsymbol{a}_b$, where $\mathbf{C}$ is the rotation matrix from the body coordinate system to the global coordinate system. The IMU displacement can be calculated by twice integration of the global linear acceleration. Similarly, the calculated displacement could be erroneous due to accumulated errors. To improve the accuracy, we leverage the zero velocity algorithm [30] to calibrate the estimated velocity, and further integrate the calibrated velocity to obtain the accurate IMU displacement.

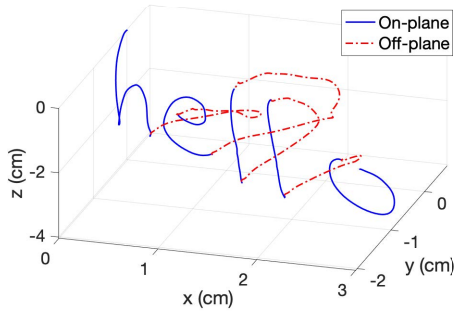


Fig. 9. Recovered tip traces in 3D space

6.5 Summary

So far, we have introduced how to infer the tip trace in the fixed coordinate system from the IMU motion. Before tracking the tip trace during the writing process, we need to estimate the initial IMU-tip vector in advance, which only needs to be conducted once for a pen with the fixed IMU. After the initialization operation, we can start capturing the handwriting. During the writing process, the IMU keeps collecting readings, for each sampling time point, we calculate the tip displacement from the initial position to the current position with following steps.

1) *Coordinate Transformation*: Take the initial body coordinate system of the handwriting as the global coordinate system, and leverage the angular velocity readings to calculate the rotation matrix from the changing body coordinate system to the global coordinate system.

2) *IMU Displacement Calculation*: Transform the collected linear acceleration samples into the global coordinate system, and calculate the IMU displacement in the global coordinate system.

3) *Tip Displacement Calculation*: With the estimated initial IMU-tip vector, measured rotation matrix and IMU displacement, we calculate the tip displacement according to Eq.(5).

By connecting the estimated tip positions at different time points chronologically, we can derive the tip trace in the global coordinate system. It is worth noting that the tip trace is in the 3D space, caused by either the on-plane writing or the off-plane motion. Fig. 9 shows an example of writing 'hello' individually. We can see that the recovered 3D tip traces include the on-plane writing strokes and the moving trace in the air between strokes. To display the exact handwriting effectively, we have to project the 3D traces onto the writing plane and filter out these off-plane traces, which will be introduced in the next section.

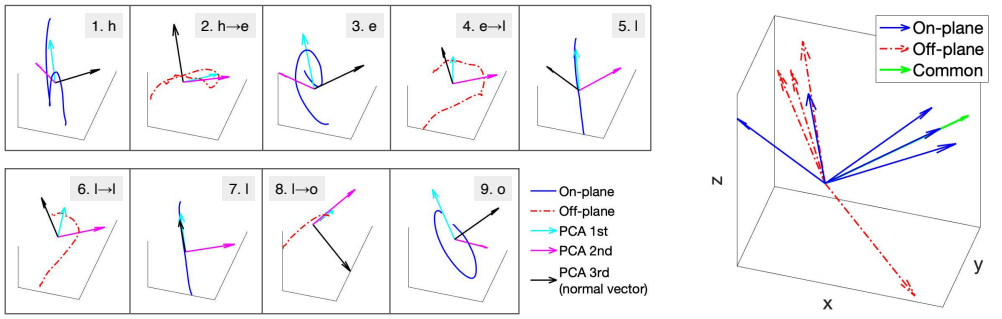
7 2D HANDWRITING RECONSTRUCTION

To distinguish the on-plane writing from the off-plane motion, we leverage the observation that on-plane strokes are all on the writing plane, while the off-plane motion incurs the variation of distance from the pen tip to the writing plane. Therefore, we first apply the principal component analysis (PCA) method to determine the writing plane, then use the distance variation of the pen tip relative to the writing plane to filter out off-plane traces and reconstruct on-plane strokes.

7.1 Writing Plane Detection

We leverage the PCA algorithm to estimate the principal plane for each motion segment, and further find the common plane of most segments as the candidate writing plane. Specifically, there are two key steps: *Included Angle Calculation*, and *Common Plane Estimation*.

7.1.1 Included Angle Calculation. We apply the PCA algorithm [7] for each motion segment, thus the first and second eigenvectors form the principal plane of the corresponding segment, denoted as *pca plane*, and the third



(a) Principal axes for each segment (b) Normal vectors of all segments

Fig. 10. Detect candidate writing plane by common normal vector of most motion segments

eigenvector is the normal vector of pca plane. By checking the included angle between normal vectors, we can determine whether the two corresponding pca planes are close to each other. Let $\boldsymbol{n}$ represent the unit normal vector, *i.e.*, $|\boldsymbol{n}| = 1$, the angle ϕ_n between $\boldsymbol{n}_i$ and $\boldsymbol{n}_j$ is calculated as:

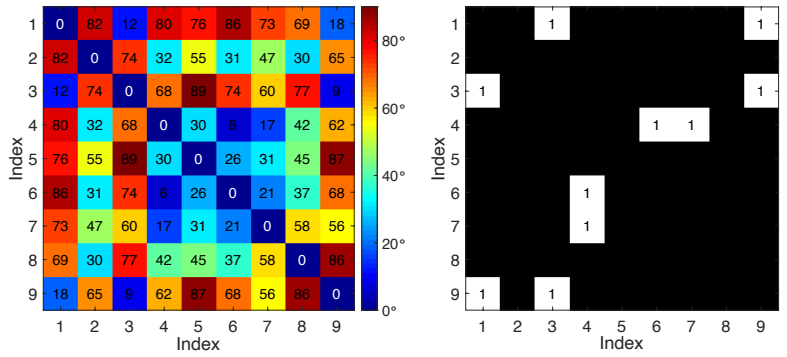
$$\phi_n = \arccos\left(\boldsymbol{n}_i \cdot \boldsymbol{n}_j\right), \tag{10}$$

where operation $(\cdot)$ means the inner product, $\phi_n \in [0, \pi)$. Considering that the normal vector may point two directions relative to the pca plane, *i.e.*, pointing upwards or pointing down for a horizontal plane, the included angle ϕ_p of two planes is calculated as:

$$\phi_p = \min\left(\phi_n, \pi - \phi_n\right). \tag{11}$$

In principle, for on-plane segments, as they are written on the same writing plane, their pca planes should be consistent with each other. For off-plane segments, as the pen tip lifts up and pins down, their pca planes tend to be perpendicular to the writing plane. Meanwhile, because the displacement from the end position of last stroke to the start position of next stroke is random, the pca planes of off-plane segments are usually inconsistent. However, in practice, it is possible that the pca plane of a simple on-plane stroke is inconsistent with the actual writing plane, or the principal plane of an off-plane segment is close to the writing plane. Taking the handwriting of 'hello' in Fig. 9, there are five on-plane segments and four off-plane segments, *i.e.*, totally nine segments. Fig. 10(a) respectively shows the principal axes of these nine segments at the same view point, numbering from 1 to 9, *i.e.*, index 1/3/5/7/9 refer to stroke 'h'/'e'/'l'/'l'/'o', and index 2/4/6/8 refer to off-plane segments. When clustering these normal vectors in Fig. 10(b), we can observe that most normal vectors of off-plane segments are disperse, whereas those of strokes {'h', 'e', 'o'} are concentrated, yet the linear stroke of 'l' is too simple to find the actual writing plane. Fig. 11(a) shows the corresponding included angles for each pair of pca planes. Similarly, we can observe that the included angles among {'h', 'e', 'o'} are small, whereas most angles between on-plane and off-plane segments are much larger, yet the angle between stroke 'l' (index 7) and the off-plane motion from 'e' to 'l' (index 4) is very small. We have to deal with the abnormal angles to find the actual writing plane.

7.1.2 Common Plane Estimation. The orientation of on-plane segments tends to be similar, whereas the orientation of off-plane segments usually distributes randomly, thus we can determine the common plane, *i.e.*, the candidate writing plane, by clustering the pca planes of similar orientation. Specifically, if the angle between two pca planes is smaller than the angle threshold ϕ_{th}, the corresponding planes are defined as *neighbors*. If multiple planes are all neighbors to each other, they form a *neighbor group*. The maximum neighbor group is selected to calculate the common plane. To determine the common plane, we reapply the PCA algorithm with the selected segments in the maximum neighbor group.



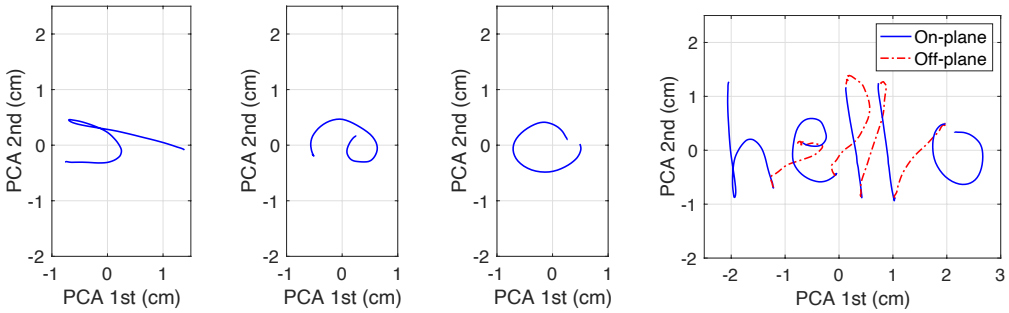
(a) Included angle between pca planes

(b) Neighbor identification

Index	Neighbors	Common?
1	3, 9	√
2	-	
3	1, 9	√
4	6, 7	
5	-	
6	4	
7	4	
8	-	
9	1, 3	√

(c) Neighbor counting

Fig. 11. Find common plane of most segments



(a) Projection by 'h' (b) Projection by 'e' (c) Projection by 'o' (d) Projection by group {'h', 'e', 'o'}

Fig. 12. Recovered tip traces on pca planes determined by different segments

Taking the same example as above, if the angle threshold is set too small, the actual neighbor will be missed due to measurement errors, otherwise, the fake neighbor will be included, thereby we set the angle threshold as 20° empirically. Fig. 11(b) shows the identification of neighbors for each segment, the counting results are listed in Fig. 11(c). Although index 4 has two neighbors, *i.e.*, index 6 and 7, index 6/7 are not neighbors to each other, so they cannot form a neighbor group. On the contrary, index 1/3/9, corresponding to strokes 'h'/'e'/'o', are neighbors to each other, thus we select them for calculating the common plane. Fig. 12(d) shows the whole recovered tip trace on the common plane. We can see that, in comparison to other pca planes with single segment in Fig. 12(a)-12(c), it is simpler for the common plane determined by the neighbor group to find the suitable axis directions for the writing plane, such that the recovered tip traces seem more comfortable and easier to understand. Note that, we only use the selected segments to estimate the candidate writing plane, it does not mean the selected segments are definitely on-plane, or the remaining are all off-plane. We have to further remove off-plane segments to reconstruct the handwriting.

7.2 Off-Plane Trace Removing

Up to now, we have obtained the projected tip traces on the common plane. The tip traces include not only on-plane strokes on the writing plane, but also off-plane traces in the air. The off-plane trace indicates the relative position of on-plane strokes, which should be removed from the recovered handwriting. It is worth noting that although the on-plane and off-plane segments appear alternately in the above example, the actual order could be chaotic, *e.g.*, on-off-off-on-on, so we cannot rely on the fixed appearing order to determine whether the segment

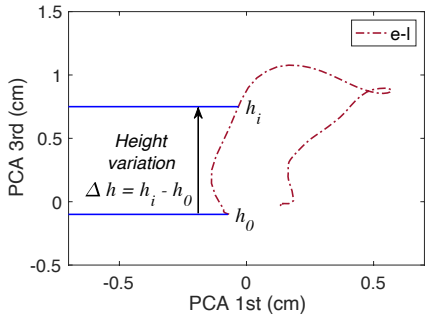


Fig. 13. Illustration of height variation

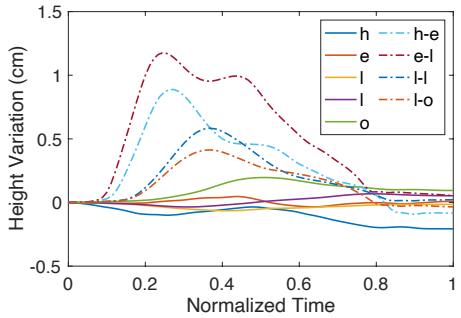


Fig. 14. Height variation of pen tip relative to common plane

is on-plane or off-plane. As mentioned in Section 3.2, the reliable difference between the on-plane writing and the off-plane motion is the distance variation relative to the writing plane, denoted as the *height variation*. Therefore, we calculate the height variation of each segment relative to the common plane to distinguish motion segments. If the range of height variation exceeds the variation threshold h_{th}, the corresponding segment is viewed as off-plane, otherwise, it is on-plane. Specifically, for the i^{th} sample of one segment, we project it to the normal vector of common plane, and record the coordinate as h_i. As illustrated in Fig. 13, the height variation of one segment is calculated as $\Delta h_i = h_i - h_0$, where h_0 is the initial height coordinate at the beginning of the segment. In principle, for the on-plane segment, its height variation should be zero. For the off-plane segment, its height variation increases when lifting up the pen tip, decreases when pinning down, and returns to zero when touching the writing plane again. However, in practice, due to measurement errors, it is hard for the on-plane segment to keep zero or for the off-plane segment to return zero. Nevertheless, the difference is small, which has little influence on distinguishing the segment. Taking the same example of writing 'hello', Fig. 14 plots the height variation of the nine segments. We can observe that the height variation of on-plane segments is very close to zero over time, whereas that of off-plane segments increases first then decreases, whose range is much larger than on-plane ones. Hence, we can use the range of height variation to distinguish on-plane segments from off-plane segments. In this way, we are able to recover the handwriting composed of multiple strokes effectively.

Further, to refine the candidate writing plane, we reapply PCA method with all estimated on-plane segments. It is worth noting that although the PCA method determines the writing plane, the directions of axes might be not suitable for displaying the recovered handwriting. Specifically, the first and second eigenvectors indicate the x-axis and y-axis of the writing plane, and the third eigenvector indicates the z-axis relative to the writing plane, *i.e.*, normal vector. To determine the positive direction of x-axis, the writing strokes are assumed to be roughly written along the positive direction of x-axis. This assumption is based on the natural writing habit, *i.e.*, a word is usually written from left to right. Similarly, we assume the positive direction of z-axis points upwards, *i.e.*, the height variation of off-plane segment should be positive. After determining the positive directions of x-axis and z-axis, the y-axis can be calculated by the cross product of the unit vectors along the position directions of z-axis and x-axis. In this way, we are able to display the recovered handwriting on the writing plane naturally.

7.3 Putting Things Together

Overall, our goal is to recover the tip trace of an ordinary pen with IMU attached to the pen tail. We take the angular velocity and linear acceleration readings from IMU as input, and output the recovered handwriting on the writing plane. As shown in Algorithm 1, we first segment the collected data and extract motion segments using STE by referring to Section 5. Then, we calculate the 3D tip trace in a fixed coordinate system by referring to Section 6. Next, we compute the principal plane for each segment, and detect the common plane of most segments as the candidate writing plane using the PCA-based method. By checking the height variation relative to the

Algorithm 1 Capture the handwriting of ordinary pen by the inertial sensor on the pen tail

Input: Angular velocity samples $\{[\omega_x(t), \omega_y(t), \omega_z(t)]^T\}$; Linear acceleration samples $\{[a_x(t), a_y(t), a_z(t)]^T\}$; Initial IMU-tip vector $\overrightarrow{S_0T_0}$; STE threshold e_{th}; Angle threshold ϕ_{th}; Height variation threshold h_{th};
Output: Captured handwriting on the writing plane $\{[x(t), y(t)]^T\}$.

1: Calculate the short time energy (STE) with the angular velocity and linear acceleration samples;
2: Segment data by STE, and extract motion segments based on e_{th};
3: Take the initial body coordinate system of the first segment as the global coordinate system;
4: **for** each motion segment **do**
5: Calculate the rotation matrix from the body coordinate system to the global on for each sampling time;
6: Calculate the IMU displacement in the global coordinate system for each sampling time;
7: Calculate the tip displacement based on $\overrightarrow{S_0T_0}$, rotation matrix and IMU displacement for each sampling time, and obtain the tip traces within the current segment;
8: **end for**
9: Estimate the principal plane of tip traces for each segment, and detect the common plane based on a_{th};
10: Calculate the height variation for each segment, and distinguish these segments based on h_{th};
11: Refine the writing plane with all estimated on-plane segments, and adjust the axis direction of writing plane to display the exact handwriting naturally;

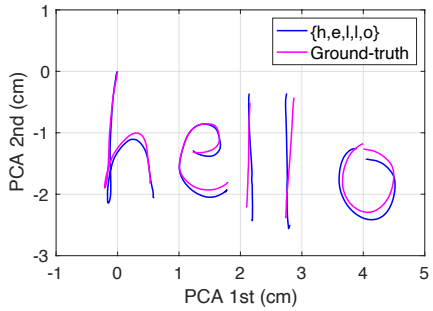


Fig. 15. Illustration of recovered handwriting

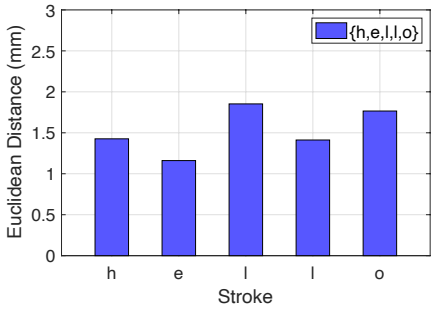


Fig. 16. Average Euclidean distance of each stroke

common plane, we can distinguish on-plane strokes from off-plane traces. Further, we estimate the writing plane with all estimated on-plane strokes, so as to refine the recovered handwriting. Fig. 15 illustrates the recovered handwriting of 'hello'. We can see that the recovered handwriting is quite similar to the ground-truth. The average Euclidean distance between the recovered handwriting and the ground-truth is 1.53mm in Fig. 16. Therefore, *Handwriting-Assistant* can realize the mm-level handwriting reconstruction.

8 PERFORMANCE EVALUATION

8.1 Experiment Settings

We have implemented a prototype system of *Handwriting-Assistant*. As shown in Fig. 17, we use Samsung Note 8 to collect the measurements of LPMS-B2 via Bluetooth[1] and capture the ground-truth of handwriting on the touch screen. The S pen and LPMS-B2 are attached on a stick to simulate the ordinary pen with IMU. 20 volunteers are invited to join in the experiments, including 12 males and 8 females, aged between 23 and 32. They are required to use the S pen with IMU to write on the smartphone. Note that, the Samsung devices are utilized to record the

[1]https://lp-research.com/support/

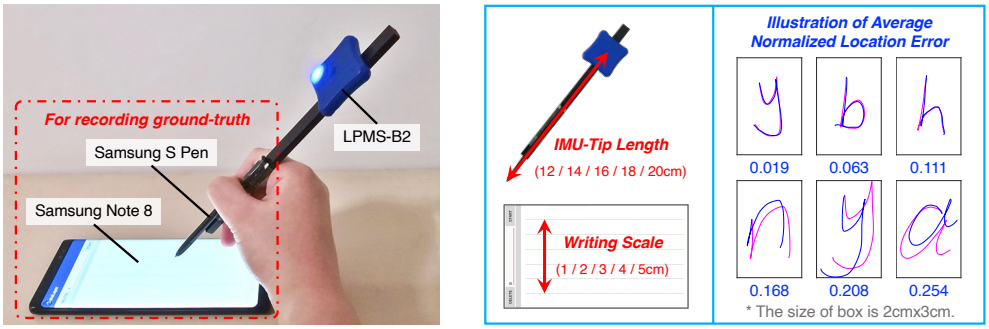


Fig. 17. Experiment setup

ground-truth of handwriting, we support the writing of ordinary pens on regular planes. The collected data is processed with MATLAB. In advance, we derive the initial IMU-tip vector according to Section 6.2.

We vary the experiment settings to evaluate the performance sufficiently. 1) *IMU-Tip Length*: The length from IMU to pen tip is varied from 12cm to 20cm, simulating pens of different lengths common in life. 2) *Writing Scale*: The height of stroke is varied from 1cm to 5cm. 3) *Writing Habit*: Different users would have different writing habits, we evaluate the performance for each volunteer, involving 20 volunteers in total. 4) *Sampling Rate*: The sampling rate is varied from 50Hz to 400Hz. 5) *Stroke Number*: The number of strokes in the handwriting is varied from 1 to 5. 6) *IMU Quality*: Apart from the expensive LPMS-B2 ($249), we test five kinds of low-cost IMUs from three vendors, *i.e.*, InvenSense, STMicroelectronics and BOSCH, whose unit prices are all below $4. For each experiment, the volunteers are asked to write each lowercase letter five times from left to right and repeat it twice, such that we collect five on-plane strokes and four off-plane traces of one letter from one volunteer. In total, we collect 26×5×2×20 on-plane strokes, 26×4×2×20 off-plane traces for each setting. By default, we attach LPMS-B2 to pen tail, the IMU-tip length is 14cm, the writing scale is with the stroke height of 2cm, all volunteers' samples are involved, and the sampling rate is 100Hz.

To evaluate the accuracy of capturing the writing trace and distinguishing the motion segment, we propose *Normalized Location Error* and *Motion Recognition Rate*, respectively. For the accuracy of capturing the handwriting, the normalized location error of the i^{th} sample is calculated by $\frac{\sqrt{(x_i-\hat{x}_i)^2+(y_i-\hat{y}_i)^2}}{L}$, where (x_i, y_i) means the ground-truth coordinate of the i^{th} sample, $(\hat{x}_i, \hat{y}_i)$ means the estimated coordinate, and L is the diagonal distance of the bounding box of the stroke to reduce the effect of writing scale on the evaluation. Note that, the orientation difference between the exact handwriting and the recovered handwriting can incur additional errors, which actually does not influence the quality of captured handwriting but influence the evaluation. To minimize the orientation difference, we align the starting position of the estimated handwriting with the exact one, then rotate the recovered handwriting to minimize the average normalized location error of the stroke. After reducing the orientation difference, we recalculate the normalized location error for the effective accuracy evaluation. Fig. 17 illustrates several examples of the average normalized location error for letters. For the accuracy of distinguishing motion segments, we evaluate the motion recognition rate, *i.e.*, $\frac{n_s}{N}$, for on-plane and off-plane segments, separately. Here, n_s is the number of successful identification, N is the number of on-plane/off-plane segments.

8.2 Overall Accuracy

Our solution can capture the handwriting with mm-level accuracy, which is much more accurate compared with traditional methods. We first compare our method with traditional methods that assume the pen's motion is pure rotation or translation in terms of capturing the handwriting. As shown in Fig. 18, the average normalized location errors of pure-rotation-based method and pure-translation-based method are 0.382 and 0.463, respectively.

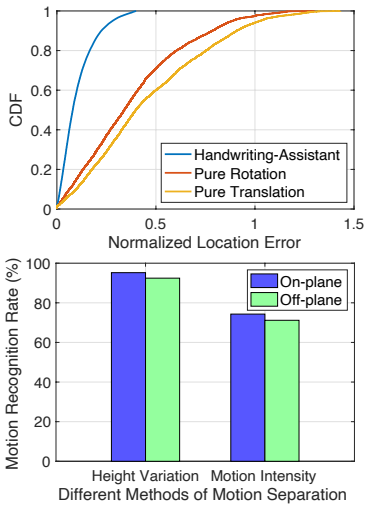


Fig. 18. Overall accuracy

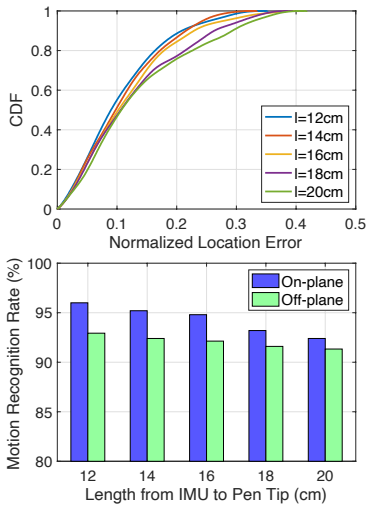


Fig. 19. Impact of IMU-tip length

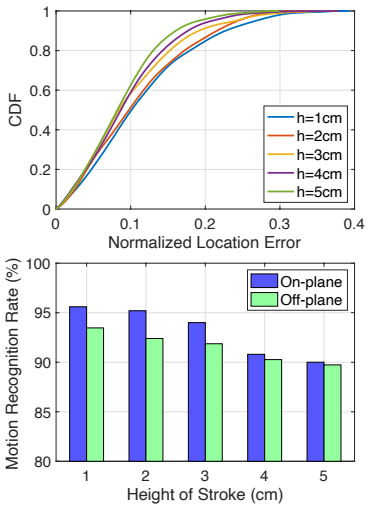


Fig. 20. Impact of writing scale

Compared to traditional methods, *Handwriting-Assistant* achieves the average normalized location error of 0.103, which is only about 27% and 22.3% of pure rotation and translation methods, respectively. The results verify the necessity of considering the motion of pen comprehensively for capturing the precise handwriting via inertial sensors. Moreover, the corresponding Euclidean distance without normalization is about 1.84mm. Therefore, *Handwriting-Assistant* can realize the mm-level handwriting reconstruction.

Our solution can distinguish on-plane segments from off-plane segments more accurately than traditional methods. We next compare our height-variation-based method with traditional motion-intensity-based methods in terms of distinguishing motion segments. As shown in Fig. 18, the motion recognition rates of *Handwriting-Assistant* are 95.2% and 92.4% for on-plane and off-plane segments, and those of motion-intensity-based method are 74.3% and 71.2%, respectively. That is, *Handwriting-Assistant* outperforms the traditional method by ×1.29. The results verify the robustness of using the height variation to distinguish on-plane segments from off-plane segments. Therefore, *Handwriting-Assistant* can reliably distinguish motion segments.

8.3 Impact of IMU-Tip Length

Our solution can achieve good accuracy with different IMU-tip lengths. To evaluate the impact of IMU-tip length, we vary the IMU-tip length from 12cm to 20cm, with the step of 2cm. As shown in Fig. 19, in terms of capturing the handwriting, when the IMU-tip length increases from 12cm to 16cm, the average normalized location error changes slightly from 0.1 to 0.117, whereas when the length keeps increasing, the average error increases to 0.135 when the length is 20cm. This is because that when the length is relatively large, the amplitude of IMU motion gets significant, so the slight shaking of the pen when stopping writing incurs the non-negligible measurement variation. Thereby, the slight motion decreases the accuracy of data segmentation, further affecting the captured handwriting. Similarly, in term of distinguishing motion segments, the motion recognition rate decreases with the increasing length, nevertheless, the average recognition rate is above 91% when the length is 20cm. Therefore, *Handwriting-Assistant* can perform robustly for pens with different lengths.

8.4 Impact of Writing Scale

Our solution can achieve good accuracy with different writing scales. To evaluate the impact of writing scale, we adjust the height of writing letter from 1cm to 5cm, with the step of 1cm. The range of writing scale covers

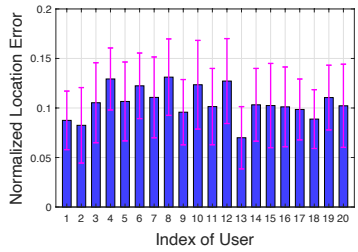


Fig. 21. Impact of writing habit

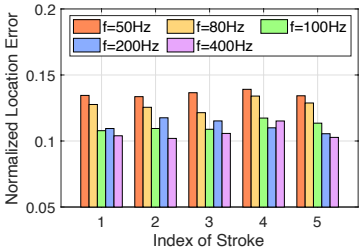


Fig. 22. Impact of sampling and stoke

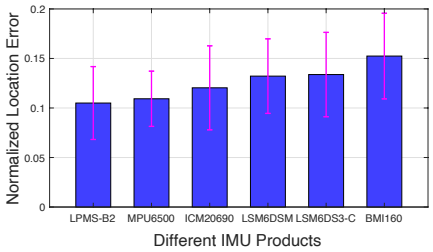
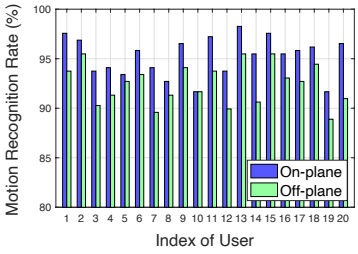
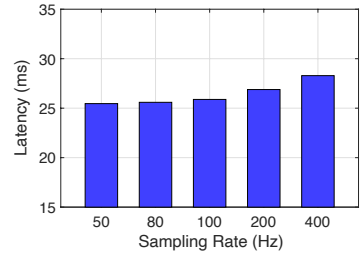
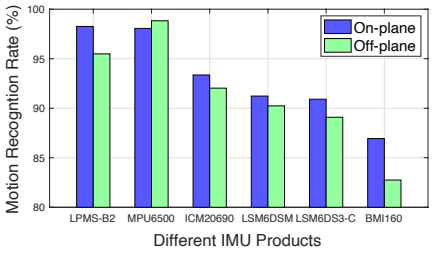


Fig. 23. Impact of IMU quality







most cases of the daily handwriting. As shown in Fig. 20, in terms of capturing the handwriting, the normalized location error gets smaller with the increasing height, and the average normalized error is about 0.088 when the height is 5cm. This is because that for the letter with a larger writing scale, its diagonal distance of the bounding box gets larger, yet the estimated position still has the high accuracy, *i.e.*, close to the ground-truth position, so the corresponding normalized error gets smaller. In terms of distinguishing motion segments, when the height increases from 1cm to 5cm, the average motion recognition rate decreases from 94.5% to 89.9%. This is because that as the writing scale increases, the absolute error of height variation increases, it is easier for the on-plane writing to incur the larger height variation. Nevertheless, the motion recognition rate is still high when the height is 5cm, *i.e.*, 90% for on-plane segments and 89.7% for off-plane segments. Therefore, *Handwriting-Assistant* can perform robustly for the handwriting with different writing scales.

8.5 Impact of Writing Habit

Our solution can achieve good accuracy with different writing habits. To evaluate the impact of writing habit, we collect the handwriting from 20 users, and evaluate the accuracy among users. As shown in Fig. 21, the average normalized location error is within [0.07, 0.13], the motion recognition rate of on-plane segments is within [91.7%, 98.3%], and that of off-plane segments is within [88.9%, 95.5%]. It is likely because of the same threshold to segment data for different users. As the shaking amplitude of the pen is different when different users hold the pen without movement, the general threshold of data segmentation may be not very precise for certain users. Moreover, different writing habits lead to different amounts of rotation and translation for the IMU motion. As the error of rotation estimated from gyroscope is not the same as the error of translation estimated from both gyroscope and accelerometer, the error of the estimated tip displacement based on Eq.(5) would a bit different. Nevertheless, the average normalized location error is 0.105, and the motion recognition rate is 93.8%. Therefore, *Handwriting-Assistant* can perform robustly among different users.

8.6 Impact of Sampling Rate and Stroke Number

Our solution can perform better with the larger sampling rate, yet the latency of processing one single stroke increases accordingly. Moreover, for the handwriting with multiple strokes, the accuracy of different strokes keeps steady. To evaluate the impact of sampling rate, we vary the sampling rate from 50Hz to 400Hz, and estimate the

accuracy of captured handwriting and latency. Meanwhile, for the handwriting with multiple strokes, we also evaluate the impact of the appearing order of stroke by calculating the accuracy for each stroke, numbering from 1 to 5. As shown in Fig. 22, with the higher sampling rate, the normalized location error gets smaller, especially when the sampling rate increases from 80Hz to 100Hz. The latency of processing a single stroke increases from 25.5ms to 28.3ms when the sampling rate increases from 50Hz to 400Hz. This is because that using the higher sampling rate can collect more measurements from IMU, as the handwriting is relatively small, using the more fine-grained measurement is probable to capture the more accurate handwriting, yet taking more time to process samples. Moreover, the accuracy for different strokes is quite similar, as the orientation error will not accumulate due to the orientation calibration, and the displacement error is calculated and calibrated within the single segment. Therefore, *Handwriting-Assistant* can capture the handwriting with multiple strokes fast and accurately.

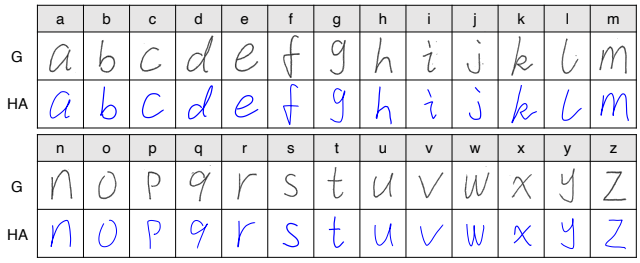
8.7 Impact of IMU Quality

Our solution can achieve good accuracy with low-cost IMUs. To evaluate the impact of IMU quality, we select five kinds of low-cost IMUs, including InvenSense MPU6500 and ICM20690 [5], STMicroelectronics LSM6DSM and LSM6DS3-C [11], and BOSCH BMI160 [3]. Their unit prices are all below $4, much lower than LPMS-B2 ($249). Note that, to avoid the difference of signal processing algorithms among different IMUs, we actually employ the IMUs embedded in smartphones, and attach the smartphones to the pen to perform the evaluation. In this way, all measurements of the five kinds of low-cost IMUs are provided by Android. As shown in Fig. 23, the accuracy of low-cost IMUs is still high, *i.e.*, the normalized location error is below 0.152, and the motion recognition rate is above 82.7%. Particularly, MPU6500 can not only achieve the similar normalized location error as LPMS-B2, but also perform slightly better than LPMS-B2 regarding the motion recognition rate. That is, *Handwriting-Assistant* does not rely on the expensive LPMS-B2, the precision of low-cost IMU is just sufficient for achieving good accuracy. It is worth noting that although MPU6500 performs best among the selected low-cost IMUs, the technical parameter of MPU6500 is not the best. For example, the resolution of MPU6500 is lower than LSM6DSM, yet MPU6500 achieves higher accuracy than LSM6DSM. The reason is likely that the higher resolution enables the IMU to sense the unexpected tiny motion. Therefore, *Handwriting-Assistant* does not necessarily require the high-precision of LPMS-B2 to achieve good accuracy, indeed, we can choose other low-cost IMU with the suitable precision to take place of LPMS-B2.

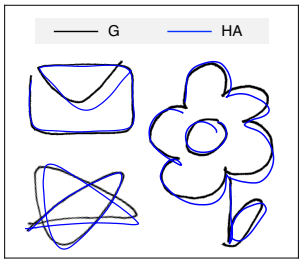
8.8 Case Study of Handwriting Input

Our solution can capture the free handwriting of ordinary pens on the regular paper, including characters of different languages and patterns. As shown in Fig. 24(a), we attach an LPMS-B2 to the tail of an ordinary pen, and let user 13 write freely on the paper. The height of handwriting is mainly around 2cm. To intuitively show the accuracy, we scan the writing contents on the paper as the ground-truth (G), and compare it with the recovered traces by *Handwriting-Assistant* (HA), the pure-translation-based method (T) [30] and the pure-rotation-based method (R) [14]. Fig. 24(b)~24(d) illustrate the free handwriting from different sources. We can observe that, for English lowercase letters in Fig. 24(b), not only letters with one stroke, *e.g.*, 'a' and 'b', but also letters with two strokes, *e.g.*, 'i', 'j', 't' and 'x', are all captured accurately by our method. When increasing the complexity of stroke, *e.g.*, the cursive handwriting of different languages in Fig. 24(d), the captured handwriting by our method (HA) is quite similar to the ground-truth (G), much more accurate than that by other traditional methods. It is because traditional methods consider the pen motion as pure translation or rotation, while we take it in a comprehensive way. During the continuous writing process, we can further leverage the method of distinguishing motion segments in Section 7 to derive the exact handwriting with multiple strokes. Moreover, *Handwriting-Assistant* can capture the simple pattern with one stroke or multiple strokes. As plotted in Fig. 24(c), the turning points of patterns are effectively captured, and the off-plane strokes of painted flower are well removed. Therefore,

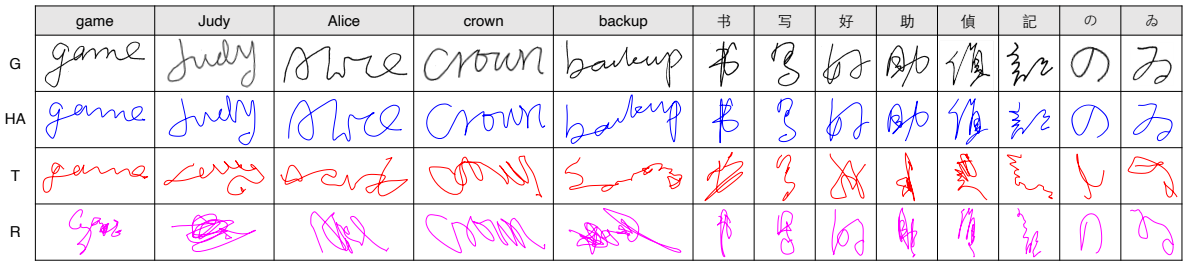
(a) Writing scenario



(b) Handwriting of English lowercase letters



(c) Handwriting of patterns



(d) Cursive handwriting of different languages (English, Chinese, Japanese)

Fig. 24. Examples of free handwriting on the paper (G: Ground-truth by scanner, HA: Recovered traces by *Handwriting-Assistant*, T: Recovered traces by pure-translation-based method, R: Recovered traces by pure-rotation-based method)

Handwriting-Assistant is capable of capturing various kinds of free handwriting, enabling applications like making notes, signing names, checking stroke orders, drawing pictures, *etc.*

8.9 Case Study of Handwriting Recognition

Besides the handwriting input, the captured writing traces can be further recognized as characters or gestures for smart devices, so as to extend the text entry methods or interaction operations. Taking the character recognition, as there exist many trace-based handwriting recognition systems or platforms, *e.g.*, MyScript [8] and Google IME [4], we can just employ these tools to recognize the accurate captured writing traces. Compared with the recognition methods that directly classify the IMU measurements into English letters [21, 26, 39, 45], the trace-based recognition methods require no training data, which is much more robust for new users or environments and easier to interface with off-the-shelf applications. Hence, we leverage the recognition interface[2] provided by Google to infer the captured writing traces. The input handwriting is English lowercase letters in the form of 2D coordinates. The recognition program is developed in Python. *Character Recognition Rate* is proposed for evaluating the recognition accuracy, defined as: $\frac{k_s}{K}$. Here, k_s is the number of correctly recognized characters, K is the total number of recognized characters. To show the efficiency of *Handwriting-Assistant* regarding the character recognition, we first compare it with traditional methods, and then vary writing modes.

8.9.1 Impact of Handwriting Reconstruction Method. Our solution can effectively improve the character recognition rate, and achieve the comparable accuracy as the ground-truth. As shown in Fig. 17, we take the recorded traces by the touch screen as the ground-truth (G), and recover the writing traces using *Handwriting-Assistant* (HA), the pure-translation-based method (T) [30] and the pure-rotation-based method (R) [14], respectively. The recovered traces are based on the collected data from 20 users with the default settings as described in Section 8.1.

[2]https://www.google.com/inputtools/request

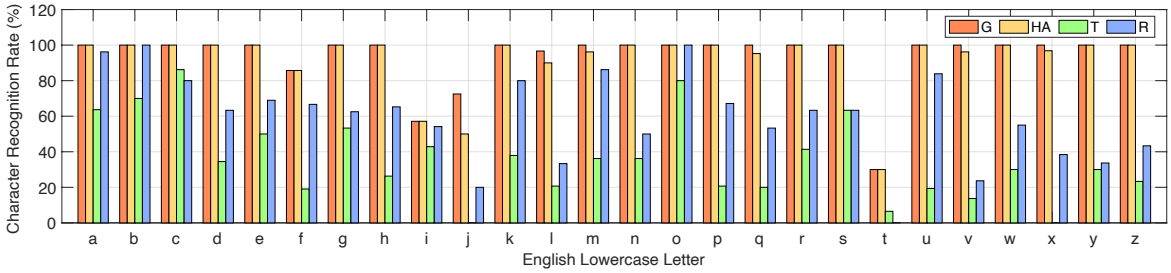


Fig. 25. Character recognition rate of different handwriting reconstruction methods

(a) Application scenarios of handwriting recognition with different writing modes (b) Character recognition rate of writing modes

Fig. 26. Evaluation of handwriting recognition with different writing modes (H: Horizontal, I: Inclined, V: Vertical)

Fig. 25 plots the character recognition rates of different methods. It is observed that Google IME can infer most handwriting from the ground-truth and our method, but it performs poorly with the handwriting from traditional handwriting reconstruction methods (T and R). The average character recognition rates are 93.9% for G, 92.2% for HA, 35.6% for T, and 59.7% for R, respectively. That is, *Handwriting-Assistant* can reach 98.2% accuracy of the ground-truth, and perform better than the pure-translation-based method by ×2.6 and the pure-rotation-based method by ×1.5. Therefore, *Handwriting-Assistant* can significantly improve the accuracy of character recognition compared to traditional handwriting reconstruction methods, making it easy to extend the text entry ways for smart devices based on off-the-shelf character recognition platforms.

8.9.2 Impact of Writing Mode. Our solution can achieve good accuracy with other rigid body used as a pen, and perform robustly for different writing planes. As many smart devices have been embedded with IMUs currently, *Handwriting-Assistant* provides the new opportunity for them to be used as handwriting tools. Fig. 26(a) illustrates several scenarios with different writing modes. Apart from the ordinary pen attached with IMU (LPMS-B2), the smartphone embedded with IMU can also be used as a pen [14, 30]. Meanwhile, the hand wearing a smartwatch can be viewed as a rigid body when the position of finger tip is stable relative to the smartwatch. Although the position of IMU embedded in the smart devices is unknown for users, it is easy for *Handwriting-Assistant* to derive the initial IMU-tip vector using the method proposed in Section 6.2. To evaluate the performance of different writing modes, we let 5 users write on different planes with different 'pen' forms. These users are selected from the prior 20 volunteers, including 2 males and 3 females, aged between 23 and 26. The writing planes are horizontal (H), inclined (I) or vertical (V), while the 'pen' forms include the ordinary pen with IMU (Pen), the Samsung Note 8 (Phone), and LG W150 (Watch). For each experiment, users are required to cursively write the lowercase English letters 5 times. The height of letters is varied from 2cm to 5cm, and the sampling rate is 100Hz. As shown in Fig. 26(b), the character recognition rate changes slightly for different writing modes, ranging from 91.9% to 92.5%. Therefore, *Handwriting-Assistant* is always able to capture the accurate handwriting on different regular planes with various 'pen' forms.

9 DISCUSSION

1) Threshold of data segmentation: *The threshold of data segmentation has the great influence on the accuracy of captured handwriting, it is better to set it according to the IMU quality, writing habit, writing plane, etc.* If setting too big, we cannot extract the complete motion segment, otherwise, we cannot effectively calibrate the velocity using the zero velocity algorithm. Theoretically, the segmentation threshold should be zero as the IMU will stop for a small while before or after writing. However in practice, due to the noise of IMU or the unconscious shaking of hands, the IMU measurement keeps changing all the time. Since the IMU noise varies with the IMU quality and the unconscious shaking varies with the writing habit of users or the material and angle of writing planes, a general threshold would perform poorly in certain cases. Nevertheless, we can further utilize online learning methods to automatically learn the appropriate threshold, making the value adaptive to the actual scenario.

2) Display of captured handwriting: *The display orientation of captured handwriting depends on the writing contents.* As the coordinate system of IMU is independent of the coordinate system of writing plane, it is difficult to determine which orientation is the best for displaying the captured handwriting only based on the IMU. Since we use the PCA-based method to determine the axes of writing plane, there are implicit assumptions about the handwriting, *e.g.*, a word is written from left to right as the assumption of MagHacker [27]. If the handwriting contains only one letter like 'h', the displayed handwriting may rotate by 90° relative to the normal orientation as shown in Fig. 12(a). Nevertheless, the display orientation does not affect the accuracy of captured handwriting. For the applications requiring the normal display orientation, we can further introduce some tricks according to application characteristics. Taking the character recognition for example, since most captured letters stand normally or lie flat, we can input both the original captured handwriting and the handwriting rotated by 90° to the recognition system, so as to improve the character recognition rate.

10 CONCLUSION

In this paper, we propose *Handwriting-Assistant*, an IMU-based approach to capture the handwriting of ordinary pens by attaching the inertial sensor to the pen tail. By rotating the pen around the tip, we can activate *Handwriting-Assistant*, and further infer the free handwriting with the pen on the paper, board or other regular planes. Particularly, we build a comprehensive rigid-motion-based model to correlate the displacement of pen tip with the rotation and translation of IMU, in this way, we can recover the tip trace in the 3D space during the continuous writing process. We also propose a PCA-based method to detect the writing plane, and a height-variation-based method to distinguish on-plane segments from off-plane segments. In this way, we can filter out the tip traces caused by the off-plane motion, and display the effective handwriting on the writing plane. Moreover, our solution can apply to other forms of rigid body used as the pen, enabling smart devices embedded with IMU to act as handwriting tools. We have implemented a prototype system and evaluated its performance in the real environment. Experiment results show that our approach can capture the handwriting with high accuracy, *e.g.*, the average tracking error is 1.84mm for letters with the size of about 2cm×1cm, and the average character recognition rate of captured single letter achieves 98.2% accuracy of the ground-truth recorded by touch screen.

ACKNOWLEDGMENTS

This work is supported in part by National Natural Science Foundation of China under Grant Nos. 61832008, 61872174, 61802169, 61902175; Jiangsu Natural Science Foundation under Grant Nos. BK20180325, BK20190293; The Key K&D Program of Jiangsu Province under Grant BE2020001-3. This work is supported in part by Hong Kong RGC Research Impact Fund (No. R5060-19) and Collaborative Research Fund (No. C6030-18G). This work is partially supported by Collaborative Innovation Center of Novel Software Technology and Industrialization; The program B for Outstanding PhD candidate of Nanjing University; Nanjing University Innovation Program for PhD candidate CXYJ21-52.

REFERENCES

[1] 2021. Anoto Livescribe Smartpen. https://www.anoto.com/solutions/livescribe/.
[2] 2021. Apple Pencil. https://www.apple.com/apple-pencil/.
[3] 2021. BOSCH. https://www.bosch-sensortec.com/products/motion-sensors/imus/.
[4] 2021. Google IME. https://www.google.com/inputtools/try/.
[5] 2021. InvenSense. https://invensense.tdk.com/products/motion-tracking/6-axis/.
[6] 2021. LP-RESEARCH. https://lp-research.com/9-axis-bluetooth-imu-lpmsb2-series/.
[7] 2021. MATLAB-pca. https://ww2.mathworks.cn/help/stats/pca.html?lang=en.
[8] 2021. MyScript. https://developer.myscript.com/.
[9] 2021. OptiTrack. https://optitrack.com/.
[10] 2021. Samsung S Pen. https://www.samsung.com/us/mobile/mobile-accessories/phones/galaxy-note10-s-pen-black-ej-pn970bbegus/.
[11] 2021. STMicroelectronics. https://www.st.com/en/mems-and-sensors/inemo-inertial-modules.html.
[12] 2021. Wacom Intuos Pro. https://www.wacom.com/en-us/products/pen-tablets/wacom-intuos-pro.
[13] Sung-Jung Cho, Jong Koo Oh, Won-Chul Bang, Wook Chang, Eunseok Choi, Yang Jing, Joonkee Cho, and Dong Yoon Kim. 2004. Magic wand: a hand-drawn gesture input device in 3-D space with inertial sensors. In *Ninth International Workshop on Frontiers in Handwriting Recognition*. IEEE, 106–111.
[14] Thomas Deselaers, Daniel Keysers, Jan Hosang, and Henry A Rowley. 2014. Gyropen: Gyroscopes for pen-input with mobile phones. *IEEE Transactions on Human-Machine Systems* 45, 2 (2014), 263–271.
[15] Zhuxin Dong, Guanglie Zhang, Chi Chiu Tsang, Guangyi Shi, Wen J Li, Philip HW Leong, and Ming Yiu Wong. 2007. μIMU-based handwriting recognition calibration by optical tracking. In *2007 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 382–387.
[16] Haishi Du, Ping Li, Hao Zhou, Wei Gong, Gan Luo, and Panlong Yang. 2018. Wordrecorder: Accurate acoustic-based handwriting recognition using deep learning. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*. IEEE, 1448–1456.
[17] Isaac Griswold-Steiner, Richard Matovu, and Abdul Serwadda. 2017. Handwriting watcher: A mechanism for smartwatch-driven handwriting authentication. In *2017 IEEE International Joint Conference on Biometrics (IJCB)*. IEEE, 216–224.
[18] Isaac Griswold-Steiner, Richard Matovu, and Abdul Serwadda. 2019. Wearables-driven freeform handwriting authentication. *IEEE Transactions on Biometrics, Behavior, and Identity Science* 1, 3 (2019), 152–164.
[19] Xiaonan Guo, Jian Liu, and Yingying Chen. 2017. FitCoach: Virtual fitness coach empowered by wearable mobile devices. In *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*. IEEE, 1–9.
[20] Prakhar Gupta, Rishabh Agarwal, Surbhi Saraswat, Hari Prabhat Gupta, and Tanima Dutta. 2017. S-Pencil: A Smart Pencil Grip Monitoring System for Kids Using Sensors. In *GLOBECOM 2017-2017 IEEE Global Communications Conference*. IEEE, 1–6.
[21] Jiahui Hou, Xiang-Yang Li, Peide Zhu, Zefan Wang, Yu Wang, Jianwei Qian, and Panlong Yang. 2019. Signspeaker: A real-time, high-precision smartwatch-based sign language translator. In *The 25th Annual International Conference on Mobile Computing and Networking*. 1–15.
[22] Hao Jiang. 2019. Motion eavesdropper: Smartwatch-based handwriting recognition using deep learning. In *2019 International Conference on Multimodal Interaction*. 145–153.
[23] Lei Jing, Zeyang Dai, and Yiming Zhou. 2017. Wearable handwriting recognition with an inertial sensor on a finger nail. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, Vol. 1. IEEE, 1330–1337.
[24] Wolf Kienzle and Ken Hinckley. 2014. LightRing: always-available 2D input on any surface. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*. 157–160.
[25] Alona Levy, Ben Nassi, Yuval Elovici, and Erez Shmueli. 2018. Handwritten signature verification using wrist-worn devices. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 3 (2018), 1–26.
[26] Xinye Lin, Yixin Chen, Xiao-Wen Chang, Xue Liu, and Xiaodong Wang. 2018. SHOW: Smart handwriting on watches. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 4 (2018), 1–23.
[27] Yihao Liu, Kai Huang, Xingzhe Song, Boyuan Yang, and Wei Gao. 2020. MagHacker: eavesdropping on stylus pen writing via magnetic sensing from commodity mobile devices. In *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services*. 148–160.
[28] Sohom Mukherjee, Sk Arif Ahmed, Debi Prosad Dogra, Samarjit Kar, and Partha Pratim Roy. 2019. Fingertip detection and tracking for recognition of air-writing in videos. *Expert Systems with Applications* 136 (2019), 217–229.
[29] Felix Ott, Mohamad Wehbi, Tim Hamann, Jens Barth, Björn Eskofier, and Christopher Mutschler. 2020. The OnHW Dataset: Online Handwriting Recognition from IMU-Enhanced Ballpoint Pens with Machine Learning. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 3 (2020), 1–20.
[30] Tse-Yu Pan, Chih-Hsuan Kuo, Hou-Tim Liu, and Min-Chun Hu. 2018. Handwriting trajectory reconstruction using low-cost imu. *IEEE Transactions on Emerging Topics in Computational Intelligence* 3, 3 (2018), 261–270.

[31] Maximilian Schrapel, Max-Ludwig Stadler, and Michael Rohs. 2018. Pentelligence: Combining pen tip motion and writing sounds for handwritten digit recognition. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–11.

[32] Yilei Shi, Haimo Zhang, Kaixing Zhao, Jiashuo Cao, Mengmeng Sun, and Suranga Nanayakkara. 2020. Ready, steady, touch! sensing physical contact with a finger-mounted IMU. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 2 (2020), 1–25.

[33] Chen Wang, Jian Liu, Xiaonan Guo, Yan Wang, and Yingying Chen. 2019. WristSpy: Snooping passcodes in mobile payment using wrist-worn wearables. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2071–2079.

[34] He Wang, Ted Tsung-Te Lai, and Romit Roy Choudhury. 2015. Mole: Motion leaks through smartwatch sensors. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*. 155–166.

[35] Jeen-Shing Wang, Yu-Liang Hsu, and Jiun-Nan Liu. 2010. An inertial-measurement-unit-based pen with a trajectory reconstruction algorithm and its applications. *IEEE Transactions on Industrial Electronics* 57, 10 (2010), 3508–3521.

[36] Raveen Wijewickrama, Anindya Maiti, and Murtuza Jadliwala. 2019. deWristified: handwriting inference using wrist-based motion sensors revisited. In *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks*. 49–59.

[37] Oliver J Woodman. 2007. *An introduction to inertial navigation.* Technical Report. University of Cambridge, Computer Laboratory.

[38] Kaishun Wu, Qiang Yang, Baojie Yuan, Yongpan Zou, Rukhsana Ruby, and Mo Li. 2020. Echowrite: An acoustic-based finger input system without training. *IEEE Transactions on Mobile Computing* (2020).

[39] Qingxin Xia, Feng Hong, Yuan Feng, and Zhongwen Guo. 2018. MotionHacker: Motion sensor based eavesdropping on handwriting via smartwatch. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 468–473.

[40] Lei Xie, Qingliang Cai, Alex X Liu, Wei Wang, Yafeng Yin, and Sanglu Lu. 2018. Synchronize Inertial Readings From Multiple Mobile Devices in Spatial Dimension. *IEEE/ACM Transactions on Networking* 26, 5 (2018), 2146–2159.

[41] Chao Xu, Parth H Pathak, and Prasant Mohapatra. 2015. Finger-writing with smartwatch: A case for finger and hand gesture recognition using smartwatch. In *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*. 9–14.

[42] Huanpu Yin, Anfu Zhou, Guangyuan Su, Bo Chen, Liang Liu, and Huadong Ma. 2020. Learning to Recognize Handwriting Input with Acoustic Features. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 2 (2020), 1–26.

[43] Yafeng Yin, Lei Xie, Tao Gu, Yijia Lu, and Sanglu Lu. 2019. AirContour: Building Contour-based Model for In-Air Writing Gesture Recognition. *ACM Transactions on Sensor Networks (TOSN)* 15, 4 (2019), 1–25.

[44] Tuo Yu, Haiming Jin, and Klara Nahrstedt. 2016. Writinghacker: audio based eavesdropping of handwriting via mobile devices. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. 463–473.

[45] Jian Zhang, Hongliang Bi, Yanjiao Chen, Mingyu Wang, Liming Han, and Ligan Cai. 2019. SmartHandwriting: Handwritten Chinese Character Recognition With Smartwatch. *IEEE Internet of Things Journal* 7, 2 (2019), 960–970.