Leveraging graphics tablet and JPen library to detect essential tremor

Ivan Osiichuk[1,†], Vitaly Brevus[1,†], Dmytro Bishchak[1], Yaroslav Mashtaliar[2], Ivan Mudryk[1,†]

[1] *Ternopil Ivan Puluj National Technical University, 56 Ruska str., Ternopil 46001, Ukraine*
[2] *Ivan Franko National University of Lviv, 1 Universytetska str., Lviv 79000, Ukraine*

Abstract

This article delves into the topic of essential tremor detection through the integration of software recording techniques with graphics tablets and the JPen library.

Essential tremor, a common neurological disorder characterized by involuntary rhythmic shaking, presents challenges in accurate diagnosis and monitoring. Leveraging the precision and versatility of graphics tablets, coupled with the functionalities offered by the JPen library, this research introduces an approach for tremor detection.

Despite the promise of this approach, the article also addresses the challenges and limitations encountered during implementation with the JPen library and graphics tablet, including issues related to calibration and signal recording. By discussing these challenges, this research aims to provide insights for overcoming technical hurdles and optimizing the use of the JPen library in essential tremor detection.

Keywords

tremor detection, digitizer, graphics tablet, capturing library, spiral recording

1. Introduction

Essential tremor (ET), also known as familial tremor, is a neurological disorder characterized by involuntary shaking or trembling of part of the body, most commonly the hands. Still, it can also affect other areas such as the head, voice, arms, or legs. The tremor typically occurs with movement and can worsen with stress, fatigue, caffeine, or certain medications [1].

The primary symptom of essential tremor is trembling, which can range from mild to severe and may worsen over time. The tremor usually affects both sides of the body

CITI'2024: 2nd International Workshop on Computer Information Technologies in Industry 4.0, June 12–14, 2024, Ternopil, Ukraine

* Corresponding author.
† These authors contributed equally.
osiichuk100@gmail.com (I. Osiichuk); v_brevus@tntu.edu.ua (V. Brevus); dmytro.bischak@gmail.com (D. Bishchak); mashtaliaryaroslav@gmail.com (Y. Mashtaliar); i1mudryk@ukr.net (I. Mudryk)
0009-0003-6844-2374 (I. Osiichuk); 0000-0002-7055-9905 (V. Brevus); 0009-0005-9993-9302 (D. Bishchak); 0009-0004-6797-5483 (Y. Mashtaliar); 0000-0002-4305-1911 (I. Mudryk)

© 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR-WS.org/Vol-3742/paper8.pdf

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

symmetrically. In addition to the hands, it can involve the head (nodding or shaking), voice (quavering or shaky speech), and other body parts.

The exact cause of essential tremor is unknown, but it is believed to involve abnormal electrical brain activity that affects the pathways responsible for controlling movement. Essential tremor often runs in families, implying the possibility of a genetic component. In addition, environmental factors may also play a role in triggering or exacerbating the tremors.

Essential tremor is diagnosed based on the medical history, symptoms, and physical examination of a patient. Doctors may order tests, such as blood tests or imaging scans, to rule out other conditions causing the tremors. One way to assess tremor is to draw a spiral [2].

While there is no cure for essential tremor, several treatment options can help manage symptoms and improve quality of life. These include medications such as beta-blockers, anti-seizure drugs, or tranquilizers to help reduce tremors.

Certain lifestyle modifications may also help reduce the severity of essential tremor symptoms. Avoiding triggers such as caffeine, stress, and fatigue can be beneficial. Occupational therapy or physical therapy may also help individuals with essential tremor learn techniques to cope with daily activities affected by tremors.

Essential tremor can significantly impact a person's daily life, making tasks that require precise and steady motility, such as writing, eating, or drinking, challenging. The tremors may also affect self-esteem and confidence, leading to social withdrawal or the avoidance of certain activities.

Essential tremor is not life-threatening, and the severity of symptoms can vary widely among individuals. While the condition may worsen over time, especially in older adults, some people experience periods of remission or find that their symptoms remain stable with treatment.

Overall, essential tremor is a chronic condition that requires ongoing management and support. With proper treatment and lifestyle adjustments, many individuals with essential tremor can effectively manage their symptoms and lead fulfilling lives.

2. Using the spirals drawing for detection

The ability of individuals with ET to draw both spirals and straight lines could aid in the clinical assessment process. Digital graphic boards facilitate the collection of data regarding the location of the pen tip, pressure applied, and tilt angles [3]. Various analytical approaches have been employed, encompassing both Cartesian representations (y- versus x-position) and polar views (radius versus angle) of pen tip coordinates [4, 5, 6, 7]. Numerous metrics related to spiral frequency and time have been documented, including alterations in radius per angle, radial error, and deviation analysis from an ideal or pre-established spiral. Additionally, deviations from the starting point have been explored [8]. Studies have examined the variability in spiral width [9] and the primary tremor axis of the spiral [10, 11, 12].

While drawing straight lines with a pen or digital tool is a more commonplace daily activity, these tasks (with or without tracing) have been comparatively underexplored in previous studies and tremor severity assessments compared to the spiral task. Alternative versions of line trajectory drawing have been examined, such as drawing a line between two points.

Efforts have been made to ascertain whether visual assessments of spirals and digital analyses can complement the clinical diagnostic process. Findings indicate that features derived from digital assessments, such as peak amplitude from velocity spectral curves and residual deviation and fluctuation characteristics from radial fitted curves, exhibit correlations with visual rating scores when logarithmic relationships are applied [6]. Moreover, researchers have employed spiral scores to monitor changes in tremor severity over time, revealing an upward trend in spiral scores. However, there appears to be a dearth of published data comparing digital spiral outcome measures between essential tremor (ET) and control groups, as well as in comparison to other drawing tasks, such as lines.

Drawing tasks entail both fine and gross motor skills and may necessitate different hand postures, such as with or without the forearm resting on a surface [7]. They may also involve multiple directional features (e.g., spirals) or necessitate a more defined drawing direction (e.g., straight lines). In writing, vertical movement is controlled by finger joints, and horizontal movement by the wrist. Tremor manifestations in ET may vary across upper limb joints and in different movement directions within each joint, for example, exhibiting more flexion-extension than a supination-pronation postural tremor in the wrist. Furthermore, tremors may influence grasping forces and prehension kinematics in precision grips. Previous studies have identified a primary tremor axis in drawing a spiral and a correlation between the tremor axis and joint movements [13].

3. Recording a drawing pattern using a graphics tablet

Using a graphics tablet for essential tremor detection presents a promising avenue for both research and practical application in the medical field. This can improve the quality and precision of diagnosing and monitoring ET progression.

One of the key components of movement that characterizes a specific disorder is frequency. For example, for essential tremor, it is 4-10 Hz, and for Parkinson's disease, it is 4-6 Hz [1]. Therefore, the polling rate is an especially important characteristic of the tablet for diagnosing movement disorders.

However, technological advancements, especially the integration of graphics tablets, offer innovative solutions for detecting and tracking tremor symptoms. Using a graphics tablet has certain advantages:

- Graphics tablets provide a quantitative and objective method for measuring tremor severity. Traditional assessments rely on subjective observations by clinicians, whereas using a graphics tablet allows for precise recording and analysis of tremor characteristics such as frequency, amplitude, and direction [14].
- With the ability to capture tremor data in real-time, graphics tablets enable continuous monitoring of tremor fluctuations over time. This approach offers insights into the progression of the condition and the effectiveness of treatment interventions.
- Graphics tablets are compact, portable devices that can be easily integrated into clinical settings or used in patients' homes. This accessibility enhances the feasibility of frequent monitoring, particularly for individuals who may have difficulty traveling to medical facilities for assessments [15].

The process has its peculiarities:

- The sensitivity of the graphics tablet's sensor can influence the accuracy of tremor measurements. Calibration procedures are necessary to ensure reliable data collection and minimize discrepancies caused by variations in tablet sensitivity.
- Patients with ET may experience challenges in navigation through the graphical user interface of the tablet, especially if tremor symptoms affect their fine motor skills. User-friendly interfaces and intuitive design features are essential for optimizing usability and minimizing user frustration.
- External factors such as environmental vibrations or unintentional movements can introduce noise or artifacts into tremor data collected via graphics tablets [15]. Robust filtering algorithms and signal processing techniques are required to differentiate pathological tremor from extraneous disturbances.

And there are problems:

- There is a lack of standardized protocols and guidelines for utilizing graphics tablets in essential tremor assessment. Establishing consensus on measurement techniques, data analysis methods, and outcome metrics is crucial for ensuring consistency and comparability across studies.
- While preliminary studies have demonstrated the feasibility and validity of using graphics tablets for tremor assessment [15], further validation is needed to corroborate findings and establish the reliability of this approach. Large-scale clinical trials and comparative studies are necessary to evaluate the accuracy and clinical utility of graphics tablet-based tremor detection methods.
- The cost of graphics tablets may pose barriers to widespread adoption, particularly in resource-constrained healthcare settings or for individuals with limited financial means [15]. Efforts to develop affordable and accessible solutions, such as open-source software platforms or low-cost sensor technologies, could facilitate a broader integration of graphics tablet-based tremor detection systems.

In summary, leveraging graphics tablets for essential tremor detection offers numerous advantages in terms of objective measurement, real-time monitoring, and accessibility. However, addressing peculiarities related to sensor sensitivity, user interface design, and interference factors, as well as overcoming problems such as standardization, validation, and cost, are essential for realizing the full potential of this technology in clinical practice.

4. The library for pattern recording

4.1. The JPen library

In addition to the hardware part, software is no less important, being a necessary layer between the tablet's sensors and the image on the screen or a set of recorded data. The JPen Java library stands out as a good option for developers seeking to integrate graphics tablet functionality into their Java applications.

JPen, short for Java Pen, is an open-source Java library designed to simplify the integration of graphics tablet support into Java applications. JPen provides an abstraction layer that allows developers to interact with graphics tablets in a platform-independent manner. This means that regardless of the underlying operating system or tablet hardware, developers can leverage JPen to access pen input and pressure sensitivity data seamlessly [16].

One of the key features of JPen is its support for various types of graphics tablets, including those from popular manufacturers like Wacom and Huion. This broad compatibility ensures that developers can adopt a single, standardized approach and omit device-specific implementation details, which greatly widens the range of targeted devices and eases the development and maintenance efforts.

The library offers a straightforward API that abstracts the complexities of dealing with pen input, making it easy for developers to incorporate tablet support into their Java applications. With JPen, developers can capture pen events such as pressure, position, tilt, and rotation, enabling precise and responsive interaction with digital canvas or other graphical elements.

JPen also provides utilities for handling common tasks associated with graphics tablet integration, such as configuring pen sensitivity settings and calibrating tablet input. Additionally, the library includes support for multi-touch gestures, allowing developers to create rich, multi-modal user experiences that combine pen input with touch interactions.

In conclusion, JPen is a valuable tool for developers looking to incorporate graphics tablet support into their Java applications. With its broad compatibility, intuitive API, and active community support, JPen simplifies the process of integrating pen input and pressure sensitivity into Java-based creative software, empowering developers to unleash the full potential of graphics tablets in their applications.

4.2. Technical aspects of JPen

JPen detects connected graphics tablets using platform-specific APIs or drivers provided by tablet manufacturers. Upon detection, it initializes communication channels with the tablet, establishing a bidirectional data flow for sending commands and receiving pen input events.

It employs an event-driven architecture where developers register event listeners to handle various pen input events. These events include pen-down, pen-up, movement, pressure changes, tilt, rotation, and multi-touch gestures. When a pen input event occurs, JPen triggers the corresponding event listener, passing relevant data such as pen coordinates, pressure levels, and pen properties.

The library translates raw pen input data from the tablet into standardized Java objects or data structures. For example, it might represent pen coordinates as (x, y) coordinates on the canvas and pressure levels as floating-point values normalized within a specified range (e.g., 0.0 to 1.0) [17].

The library uses providers for device access. JPen has providers for Linux (XInput), Windows (Wintab), Mac OS X (Cocoa, OS X 10.5), and the Java system mouse (Java AWT mouse).

XInput is an extension of the X Window System protocol used in Linux operating systems. It provides a standardized interface for handling input devices such as keyboards, mice, touchpads, and graphics tablets.

Cocoa is Apple's native application framework for macOS and iOS platforms. It provides a set of APIs and libraries for developing applications that run on Apple's operating systems.

Wintab is an API on Windows for accessing graphics tablets and pen input devices on Windows systems. It offers a standardized interface for communicating with Wintab-compatible tablets and accessing features such as pressure sensitivity, tilt, and rotation. Since the library was mainly tested on Windows, more Wintab aspects are explained in the next section.

Java AWT is a platform-independent toolkit for building graphical user interfaces (GUIs) in Java. The use of this provider is universal, but it can degrade the speed of data retrieval and its completeness. Figure 1 depicts the functional diagram of JPen:

1. The unprocessed information coming from the drawing tablet to the USB allows the operating system to communicate with the tablet [16]. Without a driver, the operating system wouldn't know how to interpret the raw data from the tablet. The data flows either through the driver layer, if it is installed, or gets into a rawer form.
2. Providers are operating system-specific APIs that can be used to communicate with the tablet driver and get the raw data. Wintab is for Windows; XInput is used in Linux; and Cocoa is needed for macOS. If the system does not have a driver, a standard Java AWT is used. With AWT, not all types of data from the tablet are available and may not be as accurate. When using drivers, data is available in higher resolution.
3. JPen is a key player in this interaction and connects this raw data with custom-written software. It converts this data from the providers into Java objects, which can then be used to write programs. Since specific providers are native to their OS, the Java Native Interface (JNI) is used to retrieve data from such environments.
4. Then, the data can be used in written programs, where it is stored in the form of Java objects. Now the information can be displayed on the screen or recorded into a file.

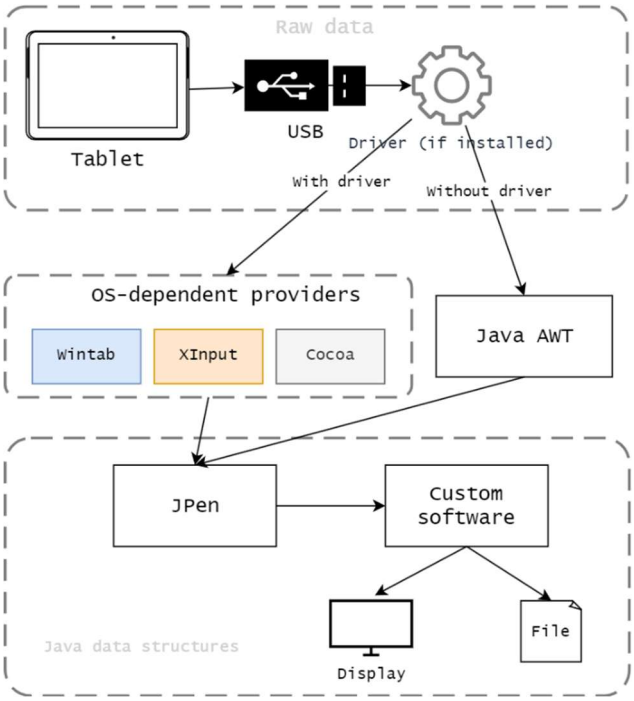

Figure 1: Functional diagram of the JPen with custom-written software

4.3. Wintab provider

After installing the tablet driver on Windows, it also installs DLLs that support Wintab. These DLLs enable applications to communicate with the driver via a private data context. This allows the application to integrate the tablet into its own data space and receive pen data packets and configuration from the driver. Due to the partitioning of Wintab support within the driver, multiple applications can have their own separate Wintab pen data streams from the driver (Figure 2).

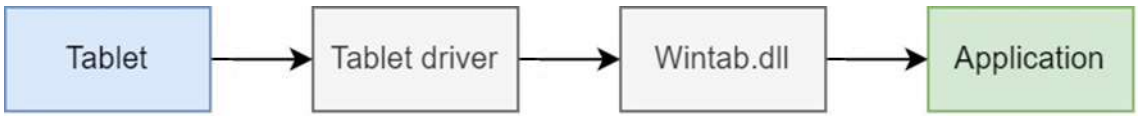

Figure 2: The general pipeline with Wintab

The core components of the Wintab programming model revolve around Wintab contexts and Wintab data acquisition. In this model, an application is responsible for creating, configuring, and opening a Wintab context. Within the application's message loop, it either reacts to Wintab data messages and retrieves data or responds to other events like mouse actions and retrieves data accordingly. Subsequently, the acquired data can be processed within the application's environment for various purposes, such as drawing or signature capture, both of which are common examples.

The Wintab API has long supported multiple independent contexts per application and per tablet, allowing apps to receive separate pen data streams from each attached tablet. Contexts are crucial for managing per-application data, defining what data is sent to the app, and how it maps to the system display. Each app has its own context, which can be customized without affecting other apps or global tablet preferences. Contexts serve as the interface for tablet usage in applications, containing information about the tablet area, event types, and delivery methods. While apps can open multiple contexts, most require only one. Contexts are managed in an overlapping container, with the topmost context processing tablet events based on overlapping order and attributes. Wintab ensures that each application receives data only when it has keyboard focus, preventing background apps from interfering with foreground tasks. This approach was implemented due to many apps not handling their contexts properly.

Wintab offers two approaches for capturing pen data: message-driven and polling. In message-driven capture, pen data is delivered to the application via Windows messages triggered by the driver when data is available. The application processes this data and awaits the next message. In polling capture, the application prompts Wintab for data when certain non-Wintab messages (e.g., mouse-down or move messages) are received. This allows continuous data caching as the pen moves within the tablet's range. Regardless of the capture method, Wintab applications should register for messages notifying them of key events like driver status changes or tablet attachment or detachment to manage their state effectively. Since each application's Wintab context relies on the tablet driver's support, it remains valid only while the driver is active. If the driver restarts (e.g., after a system reboot) or the application is relaunched, a new Wintab context is created, requiring the application to reopen it. Wintab message notifications facilitate managing such situations [18].

4.4. The overview of available data

It is important to know what data is available to us to build an analysis based on known input components. For a general overview, it's possible to take a demo offered by JPen. Figure 3 shows the graphical interface of this program.

It provides various types of data that can be obtained from a graphics tablet. Here are the key data types available:

- First, X and Y coordinates represent the horizontal and vertical positions of the pen on the tablet surface.
- It measures the pressure exerted by the pen on the tablet.
- Tilt-X and Tilt-Y indicate the angles at which the pen is tilted in the X and Y directions.
- Side Pressure is an additional pressure metric, indicating the pressure applied from the side of the pen.
- Also, the rotation of the pen around its axis is measured.

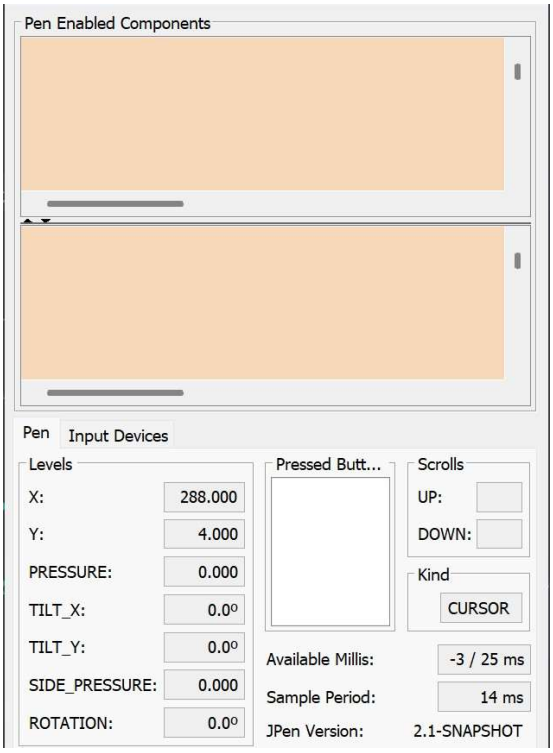

Figure 3: The GUI of the JPen demo

In addition to these characteristics, the sampling period is also shown, which is important for further work with the data. The optimal period gives the least data loss and less noise [19].

4.5. Event-listener architecture of JPen

JPen uses the event-listener architecture for interaction, an example of which is shown in Figure 4. Event/listener architecture is a mechanism for handling events generated by user interactions or system actions within applications. This architecture allows developers to define event listeners to respond to specific events, such as button clicks, mouse movements, key presses, and more [20].

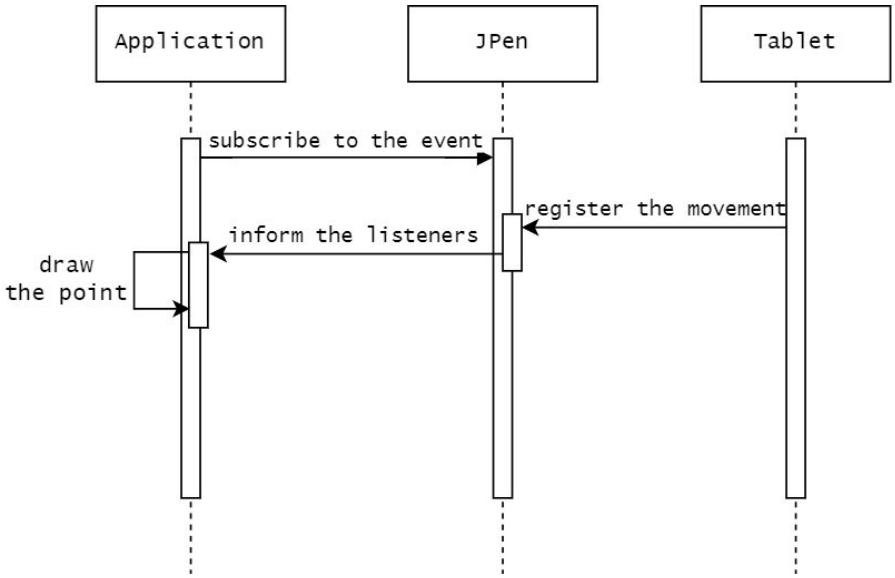

Figure 4: Sequence diagram of JPen's event/listener architecture

In general, the library's work follows this architecture without any major peculiarities:

- When an interaction occurs on the tablet, the JPen library creates an event object encapsulating the details of the interaction. This object contains information such as coordinates, pressure, tilt, etc., depending on the type of interaction and capabilities of the device.
- The JPen library defines a listener interface, which defines methods for processing different types of events. To use it in an application, you need to create a class implementing these methods and add it as a listener.
- When a user interacts with a tablet, the library detects the event and dispatches it to all registered listeners.

Custom event-handling logic is implemented within the listener methods. For instance, developers might process pen-down, pen-move, or pen-up events by updating the UI, capturing input data, triggering actions, etc.

5. Tablet input recording solution

5.1. The overview of the prototype

To use JPen, it is required to create a data-recording application. The basic implementation of the library requires the use of at least one AWT component. Building a Swing application as the base is one possibility. The general class diagram is shown in Figure 5.

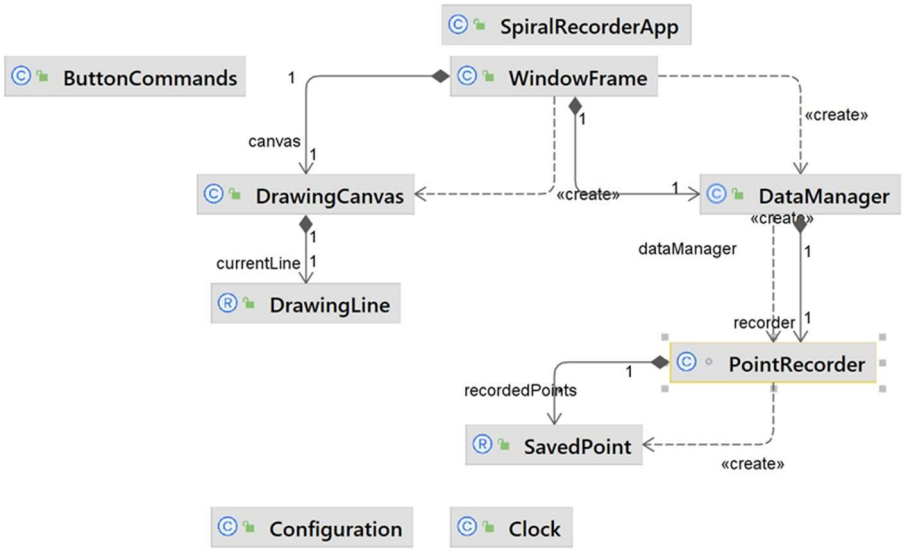

Figure 5: The class diagram of the tablet input recorder app

The program is launched by SpiralRecorderApp, which also initializes the Swing application's main window. In addition to representing the GUI's primary window, WindowFrame houses the buttons that allow a user to begin recording or saving a file. The DrawingCanvas, a canvas for showing the spiral pattern, is a component of this class. It shows the points that the graphics tablet has provided. The DrawingLine object is used to create a link between such points because they might be received at various frequencies. This ensures smooth tracing by drawing a line between the previously registered point and the current one instead of showing individual points.

The DataManager controls the recording's status and writes it to a file when it comes to working with input data from the tablet. PointRecorder is a component that works directly with JPen; it maintains a collection of received points and has event handlers.

PenManager is responsible for connecting components that can handle library events and is used in two different places in the program: DrawingCanvas and PointRecorder, which are the two main places that interact with the library. Both places use the device manager singleton from the library. These two classes save individual registered points, together with their coordinates and time, using an object called SavedPoint.

Additionally, utility classes are required at certain points throughout the application. The tablet's configuration reads from a file and provides access to the configuration; both features

are necessary to calculate relative coordinates. With a clock, you can obtain the current time in the required format. The view of the prototype window is shown in Figure 6.

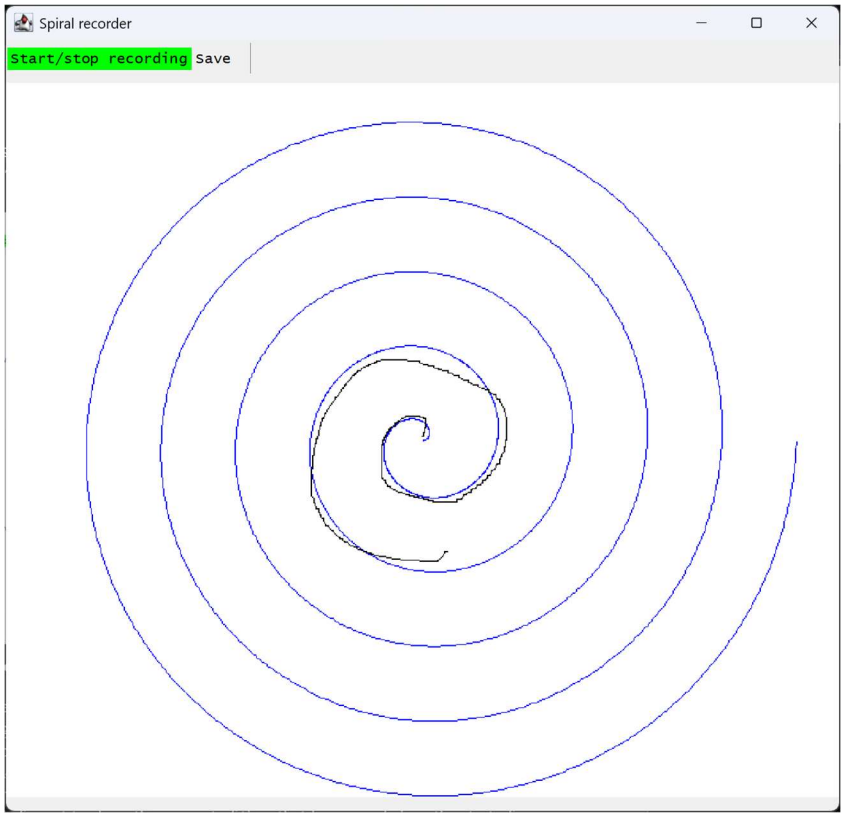

Figure 6: The GUI of the prototype

5.2. The models of the tablet used for motion capturing

Hardware, especially graphics tablets, is an important component of motion detection solutions. Wacom Bamboo Capture and Huion Kamvas Pro 16 are used in the experiments (Table 1).

Table 1
Tablets specification

Property	Wacom Bamboo Capture	Huion Camvas Pro
Resolution (pixels)	- (no screen)	1920x1080
Polling rate (Hz)	133	220
Working area (mm)	147 x 92	344.16 x 193.59

5.3. The timestamp duplication problem

One of the problems with the data is the repetition of timestamps for different coordinates. An example of such repetition for scheduled time can be seen in Table 2.

Table 2
The example of scheduled time repetition for different coordinates

registered time	scheduled time	r delta	s delta	x	y	z
0:0:0.937	0:0:0.946	0	0	80.1	-63.4	0.0
0:0:0.943	0:0:0.946	6	0	80.3	-63.2	0.0

Potential reasons for this behavior may include:

- JNI, which has a certain overhead.
- The driver used by a tablet (Wintab).
- Hardware issues, such as a peculiarity of the tablet or incorrect communication with a driver.

To understand the overall picture, it is useful to analyze groups of coordinates with the same timestamp. Figure 7 shows the chart of coordinates over registered time for Bamboo (left) and Huion (right).

The right part shows a significant number of "steps" representing different coordinates for the same timestamp. On the other hand, the left chart shows a smooth line, with each coordinate having a unique time. This pattern is repeated along the entire time axis, but the intensity varies.

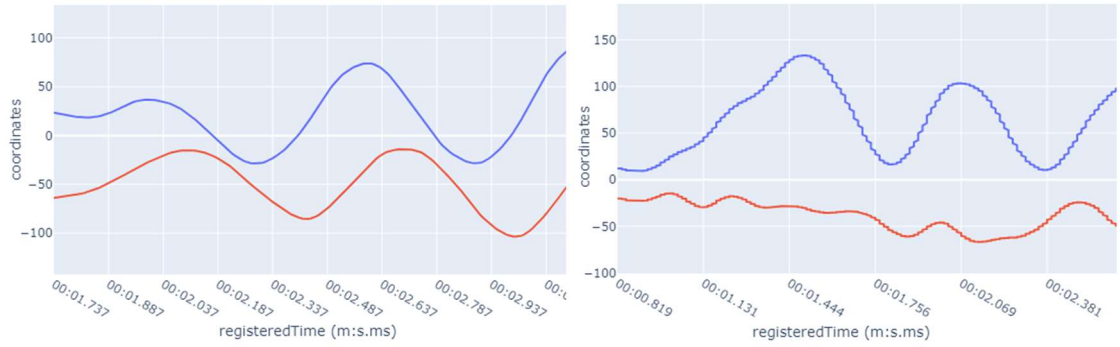

Figure 7: Coordinates by time for Bamboo (left) and Huion (right)

For greater clarity, the distribution of the number of points for each unique registered time value can be analyzed. Since there may be additional hand movements at the beginning and end of the recording, the first and last seconds of the recording were skipped for each dataset to remove this factor. Table 3 shows the statistics for both tablets. As can be seen from Table 4, all registered points from Bamboo have a unique timestamp. In turn, 1 unique time point for Huion is rare, as there is significant duplication of time values for coordinates.

To disclaim the responsibility of JNI and the library, it is possible to record a report using the Wintab diagnostic tool [21]. When used, a file is created that similarly contains data on the position of the pen at certain points in time. Thus, the data will be obtained closest to the source using the software. Statistics for this tool are presented in the columns labeled as wtcapt and calculated for both tablets. In total, 15 experiments were conducted for the Java-

prototype data and 20 for wtcapt. The tables show aggregated statistics for these data, with the median taken for each characteristic.

Table 3
The statistics for the number of points for each unique registered timestamp

Statistics	Bamboo (Java)	Bamboo(wtcapt)	Huion (Java)	Huion (wtcapt)
groups	2213	296	895	308
mean	1	1	1.8	3.4
std	0	0	1	0.4
min	1	1	1	3
25%	1	1	1	3
50%	1	1	1	3
75%	1	1	3	4
max	1	1	4	4

Table 4
The distribution of the points number in each time group

Number of points in the group	Number of such groups			
	Bamboo (Java)	Bamboo(wtcapt)	Huion (Java)	Huion (wtcapt)
1	2213	296	494.5	-
2	-	-	131.5	-
3	-	-	232.5	174
4	-	-	60	134
7	-	-	-	19

The frequency is calculated based on the first and last timestamps and the total number of points:

$$period = \frac{lt - ft}{p - 1}, \qquad frequency = \frac{1}{period} \tag{1}$$

Where lt – the last timestamp (seconds), ft – first timestamp (seconds), p – the number of points.

The results are presented in Table 5. The maximum period is obtained as the maximum value from the *registeredDelta* column, which shows the period for each pair of points.

Table 5
Calculated data frequency

Property	Bamboo (Java)	Bamboo(wtcapt)	Huion (Java)	Huion (wtcapt)
period (s)	0.0075	0.0075	0.00459	0.00453
frequency (Hz)	133	133	217	220
frequency w/o duplicates (Hz)	133	133	118	63
maximum period (s)	0.008	0.008	0.032	0.032

The Huion Kamvas tablet emits data more frequently, but in both cases, there are many coordinates with the same timestamp. In turn, Wacom Bamboo has a unique time for each coordinate with the lower rate.

The recording frequency with all the points coincides with the manufacturer's claim. However, after removing duplicates from the Huion data, this frequency is much lower. When deleting duplicates, the last registered value of the coordinate group was retained.

As a solution to this problem, it is possible to discard the earlier points, leaving only the last one for this time, since the movement is linear. However, all the points may be useful for further analysis, in which case it is possible to add a marker for such points, realizing that these are sequential movements, and the next point has a slightly later time stamp.

Still, the period between two consecutive records is not uniform throughout the whole captured dataset in the case of both tablets, which may complicate carrying out any further analysis of the dataset. To convert the dataset into one with a homogeneous period, one of the existing tabulated function interpolation methods could be adopted, such as B-spline. For this, the dataset could be represented as two functions dependent on time, for X and Y coordinates, respectively. For each function, an interpolation is built, and a new dataset is created by combining the evaluated values over an evenly spaced time-series for the whole capture period. However, such techniques often require the domain of an interpolated function to contain unique values. Hence, a dataset with duplicated time points, like those obtained from Huion, must be carefully preprocessed and cleaned before applying such methods.

5.4. Java Native Interface (JNI)

An additional delay may be imposed by JNI. It is a technology that enables Java code to interact with native applications and libraries written in other languages, such as C, C++, and assembly. JNI allows Java applications to leverage existing native code libraries or access system-level resources that are not directly accessible from Java.

JNI starts with Java code running within the JVM. When Java code needs to interact with native code, it uses special syntax and constructs provided by the JNI framework. The native code, typically written in languages like C or C++, contains the implementations of the functions that Java code wants to call or interact with. This code must be compiled into platform-specific binary files, such as DLLs (Dynamic Link Libraries) on Windows or shared objects (.so) on Unix-like systems.

To invoke native code from Java, JNI provides a set of native methods and functions that serve as a bridge between Java and native code. Java code calls these native methods, which are declared using the native keyword. The JVM then locates and invokes the corresponding native functions defined in the native code.

Java and native code have different data representations, and JNI provides functions for converting data between Java types (such as int, String, Array, etc.) and their corresponding native types (such as jint, jstring, jarray, etc.).

Since Wintab is written in C++ and is native to Windows, the Java app can only communicate with it using such interfaces. And this can be a weak point in the system.

Conclusion

In conclusion, this set of tools is well-suited for motion analysis. The JPen library makes it possible to record coordinates from various devices and allows you to use tablets with or without a screen. Since the library is written in Java, the solution is cross-platform and works on different operating systems.

The article compares Huion and Bamboo graphics tablets using a prototype for motion recording. Among the problems considered is the repetition of time for different coordinates. The analysis of the resulting trajectory with this problem is presented, and both tablets are compared in terms of their behavior concerning this problem. Bamboo shows no duplication of points by registered time but does not have a display, which can make it somewhat difficult to perform the drawing test. Huion, on the other hand, has a higher polling rate and display but has a significant number of different coordinates at the same time.

This analysis also made it possible to obtain the nominal frequency of each graphics tablet, which is important for understanding the range of movement disorders that can be assessed using such hardware. Both tablets can be used for the task of diagnosing tremor because the frequency (133 Hz for Bamboo), even when duplicated (60 Hz for Huion), is higher than that defined for tremor as up to 10 Hz.

The solution is suitable for motion recording in movement disorders and allows for efficient retrieval of this data for further analysis. The advantage of custom-written software for such a task is the ability to customize the required data to be recorded. Also, it allows for the preprocessing of this data at the capturing stage.

References

[1] Haubenberger, Dietrich, and Mark Hallett. "Essential tremor." New England Journal of Medicine 378.19 (2018): 1802-1810.

[2] Performance tests, 2024. URL: https://www.mayoclinic.org/diseases-conditions/essential-tremor/diagnosis-treatment/drc-20350539.

[3] Zajki-Zechmeister, Tibor, et al. "Quantification of tremor severity with a mobile tremor pen." Heliyon 6.8 (2020).

[4] Szumilas, Mateusz, et al. "A multimodal approach to the quantification of kinetic tremor in Parkinson's disease." Sensors 20.1 (2019): 184.

[5] Yu, N.-Y.; Van Gemmert, A.W.A.; Chang, S.-H. Characterization of graphomotor functions in individuals with Parkinson's disease and essential tremor. Behav. Res. Methods 2017, 49, 913–922.

[6] Lin, Po-Chieh, et al. "A digital assessment system for evaluating kinetic tremor in essential tremor and Parkinson's disease." BMC neurology 18 (2018): 1-8.

[7] Louis, Elan D. "Utility of the hand-drawn spiral as a tool in clinical-epidemiological research on essential tremor: data from four essential tremor cohorts." Neuroepidemiology 44.1 (2015): 45-50.

[8] Ishii, Nobuyuki, et al. "Spiral drawing: Quantitative analysis and artificial-intelligence-based diagnosis using a smartphone." Journal of the neurological sciences 411 (2020): 116723.

[9] Michalec, M.; Hernandez, N.; Clark, L.N.; Louis, E.D. The spiral axis as a clinical tool to distinguish essential tremor from dystonia cases. Park. Relat. Disord. 2014, 20, 541–544.
[10] Roth, Navit, Orit Braun-Benyamin, and Sara Rosenblum. "Drawing Direction Effect on a Task's Performance Characteristics among People with Essential Tremor." Sensors 21.17 (2021): 5814.
[11] M. Petryk, M. Bachynskyi, V. Brevus, I. Mudryk, and D. Mykhalyk, "Analysis technology of neurological movements considering cognitive feedback influences of cerebral cortex signals" in Proceedings of the 2nd International Workshop on Information Technologies: Theoretical and Applied Problems (ITTAP 2022), Eds., in CEUR Workshop Proceedings, vol. 3309. Ternopil, Ukraine: CEUR, Nov. 2022, pp. 45–54. Accessed: Sep. 10, 2023. [Online]. Available: https://ceur-ws.org/Vol-3309/#paper4
[12] Petryk M.R. Methods and models for identification of parameters of complex multicomponent nanoporous and nanoscale structures and processes: monograph / M.R. Petryk, I.V. Boyko, M.I. Lebovka, V.M. Brevus– Ternopil: Ivan Pulyuy TNTU Publishing House, 2023 – 156 p.
[13] Buijink, A. W., van der Stouwe, A. M., Broersma, M., Sharifi, S., Tijssen, M. A., & Speelman, J. D. (2015). Motor network disruption in essential tremor: a functional and effective connectivity study. Brain, 138(10), 2934-2947.
[14] Kim, Christine Y., et al. "Repeated spiral drawings in essential tremor: a possible limb-based measure of motor learning." The Cerebellum 18 (2019): 178-187.
[15] Mcgurrin, Patrick, et al. "Quantifying tremor in essential tremor using inertial sensors—Validation of an algorithm." IEEE journal of translational engineering in health and medicine 9 (2020): 1-10.
[16] JPen, 2024. URL: https://github.com/nicarran/jpen
[17] San Luciano, Marta, et al. "Digitized spiral drawing: A possible biomarker for early Parkinson's disease." PloS one 11.10 (2016): e0162799.
[18] Wintab basics, 2024. URL: https://developer-docs.wacom.com/docs/icbt/windows/wintab/wintab-basics/
[19] Gil-Martín, Manuel, Juan Manuel Montero, and Rubén San-Segundo. "Parkinson's disease detection from drawing movements using convolutional neural networks." Electronics 8.8 (2019): 907.
[20] Korunović, Ana, and Siniša Vlajić. "An Example of Integration of Java GUI Desktop Technologies Using the Abstract Factory Pattern for Education Purposes." ETF Journal of Electrical Engineering 29.1 (2023): 3-11.
[21] Wintab diagnostic, 2024. URL: https://developer-support.wacom.com/hc/en-us/articles/9354461019927-Wintab-diagnostic