

**DESIGN AND DEVELOPMENT OF SYSTEM
MODULE TO MEASURE THE VARIATION IN
MICROGRAPHIA AND SPEECH FOR
PARKINSON'S DISEASE**

A PROJECT REPORT

Submitted by

**SOUVIK CHAKRABORTY [RA1611017010051]
ADVEYA SINGH [RA1611017010074]**

Under the guidance of

Dr. P.A. SRIDHAR, Ph.D

(Assistant Professor (Research), Department of Electronics & Instrumentation
Engineering)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

ELECTRONICS & INSTRUMENTATION ENGINEERING

of

FACULTY OF ENGINEERING AND TECHNOLOGY



SRM
INSTITUTE OF SCIENCE & TECHNOLOGY
Deemed to be University u/s 3 of UGC Act, 1956

S.R.M. Nagar, Kattankulathur, Kancheepuram District

MAY 2020

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that this project report titled “ **DESIGN AND DEVELOPMENT OF SYSTEM MODULE TO MEASURE THE VARIATION IN MICROGRAPHIA AND SPEECH FOR PARKINSON’S DISEASE** ” is the bonafide work of “ **SOUVIK CHAKRABORTY [RA1611017010051], ADVEYA SINGH [RA1611017010074]** ”, who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Dr. P.A. SRIDHAR, Ph.D
GUIDE
Assistant Professor (Research)
Dept. of Electronics & Instrumentation Engineering

Signature of the Internal Examiner

SIGNATURE

Dr. A. VIMALA JULIET
HEAD OF THE DEPARTMENT
Dept. of Electronics & Instrumentation Engineering

Signature of the External Examiner

ABSTRACT

Parkinson's Disease (PD), one of the most common neurodegeneration disorders, have symptoms visible in most motor activities like movement (tremor, rigidity, micrographia) and speech (imprecise articulation and rhythm, stuttering). The speed and pressure data from a person handwriting stroke can have many applications in medical and psychological analysis. Moreover, a person's speech data can be used to track progression of PD based on its symptoms on vocal system using Unified Parkinson's Disease Rating Scale (UPDRS). These data, gathered from the patient, can be subjected to various mathematical analysis to track progression of PD. We aim to make this data collection seamless by designing a novel user-friendly proprietary software to record and analyse the handwriting/speech data in real time for identifying the progression/severity of PD.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my guide, Dr. P.A. Sridhar. His valuable guidance, consistent encouragement, personal caring, timely help and providing me with an excellent atmosphere for doing research. All through the work, in spite of his busy schedule, he has extended cheerful and cordial support to me for completing this research work.

Author

TABLE OF CONTENTS

ABSTRACT	iii
ACKNOWLEDGEMENTS	iv
LIST OF TABLES	vii
LIST OF FIGURES	ix
ABBREVIATIONS	x
LIST OF SYMBOLS	xi
1 INTRODUCTION	1
1.1 Parkinson's Disease	1
1.2 Archimedean Spiral	3
2 LITERATURE SURVEY	4
2.1 "Distinguishing different stages of Parkinson's Disease using Composite Index Of Speed And Pen Pressure of a spiral sketch"	4
2.2 "Efficiency of Voice Features based on Consonant for Detection of Parkinson's Disease"	5
2.3 "Speech Features for Telemonitoring of Parkinson's Disease Symptoms"	5
3 IMPLEMENTATION	7
3.1 Micrographia as a discriminative tool for Parkinson's Disease	7
3.1.1 Significance of extraction of micrographia data	7
3.1.2 Working Model	8
3.1.3 Based on the WIN32 API	9
3.1.4 Based on the WINTAB API	10
3.1.5 Algorithm description for Implementation of extraction and processing	11

3.2	Speech Data: A discriminative tool for Parkinson's Disease	14
3.2.1	Significance of extraction of speech data	14
3.2.2	Analysing acoustic parameters	15
3.2.3	Based on the Parselmouth API	22
3.2.4	Algorithm description for Implementation of extraction and processing	23
4	RESULTS	27
4.1	Micrographia data management module	27
4.1.1	Extracting data from the pen-digitiser setup	27
4.1.2	Analysing data extracted from Micrographia recording module	28
4.2	Speech data management module	30
4.2.1	Extracting two-way data from a dual mic setup	30
4.2.2	Analysing data extracted from speech recording module . .	30
4.3	Development of User Interface for management and integration of all Feature extraction and analysis tasks	32
4.4	Study of Data, extracted and machine-analysed	35
4.5	Novelty of Project	37
5	TESTING	38
5.1	Test to deduce pen pressure	38
5.2	Tests invloved in generating processed data from old micrographia data	39
5.3	Tests involved in analysing the internal implementation of the PRAAT software	40
5.4	Tests involved in debugging audio I/P device selection	43
6	CONCLUSION	45
A	CODE FOR MICROGRAPHIA DATA EXTRACTION	46
B	CODE FOR SPEECH DATA EXTRACTION	47
C	CODE FOR UI	48

LIST OF TABLES

1.1	Speech features commonly associated with PD	2
4.1	Weight in <i>g</i> v/s Raw pressure Unsigned Integer (UINT) values PD	27

LIST OF FIGURES

1.1	Archimedean Spiral	3
3.1	A typical handwriting digitiser device (developer-docs.wacom.com, 2010)	8
3.2	Flow-Chart for pre-processing of Micrographia data from digitiser	11
3.3	Flow-Chart for Feature Extraction from Micrographia data . .	13
3.4	Portrayal of Jitter and Shimmer interference steps in speech transmission (Teixeira et al., 2013)	17
3.5	Portrayal of the 1st local maximum of the autocorrelation result (Teixeira et al., 2013)	20
3.6	Steps in generation of MFCC visualized (Pratheeksha Nair, 2018)	22
3.7	Flow-Chart for recording speech data from an audio data stream	24
3.8	Flow-Chart for Feature Extraction from Speech Data	25
4.1	Pen Ink Visualisation through UI	28
4.2	Report generated for a People with PD (PWP)[75,Female]. . . .	29
4.3	Report visualised for a PWP[75,Female].	29
4.4	Visualisation of the "always-on" audio script logic	30
4.5	Excerpt of the "pitch" category of data extracted	31
4.6	Spectrogram plot of the "pitch" category of data depicting F0 frequencies	32
4.7	Home Screen of application module	33
4.8	Ready to extract Micrographia Data	34
4.9	Extraction of Speech Data underway	35
4.10	As is seen, general PWP data staggers a lot behind Control Group (CG).	36
4.11	CISP Index comparing PWP and CG.	36
5.1	The UI interface demo showing coordinate and pressure data of patient's pen-position	38

5.2	Command Line Interface (CLI) application to generate data from old recordings	39
5.3	CLI application asking to generate report	39
5.4	CLI application listing recorded wav files to query with	40
5.5	Depiction of a sample querying process using the program	41
5.6	Query results in full-text presented after successful query	42
5.7	Incomplete list of options when there is an audio device connected via Bluetooth	43
5.8	Complete list of options after using the modified module	44

ABBREVIATIONS

PD	Parkinson's Disease
CG	Control Group
UPDRS	Unified Parkinson's Disease Rating Scale
PWP	People with PD
F0	Fundamental frequency
SD F0	Standard deviation of F0
HNR	Harmonics:Noise Ratio
MFCC	Mel frequency cepstral coefficients
CISP	Composite Index Of Speed And Pen-Pressure
UI	User-Interface
DAQ	Data acquisition
CSV	Comma-Separated Values
UINT	Unsigned Integer
CLI	Command Line Interface

LIST OF SYMBOLS

Hz Hertz

CHAPTER 1

INTRODUCTION

1.1 Parkinson's Disease

Parkinson's disease (PD) belongs to a family of conditions called "motor system disorders", which cause "unintended or uncontrollable movements" of the body (National Institute of Neurological Disorders and Stroke, 2020) . The exact reason for PD is obscure, yet a few cases are inherited while others are thought to happen from a blend of hereditary qualities and ecological variables that trigger the disease. In PD, synapses become harmed or die in the piece of the cerebrum that produces dopamine-a chemical expected to deliver smooth, deliberate development.

As stated by the National Institute of Neurological Disorders and Stroke (2020), four primary features of PD include:

- **"tremor"** :a trademark shaking with a rhythmic to and fro movement
- **"rigidity"** : stiff muscles or reduced motion, "where muscles remain continually tense and contracted"
- **"bradykinesia"** : easing back of unconstrained and involuntary motion that can make it hard to perform straightforward assignments or quickly perform routine tasks
- **"postural instability"** : impeded balancing and changes in gait that can significantly increase fall damage .

Effect on handwriting

PD is related with development issue indications, for example, tremor, rigidity, bradykinesia, and postural flimsiness. The indication of bradykinesia and rigor is frequently in the beginning periods of the illness. These noticeably affect the "micrographia", which is a term for cramped handwriting and drawing capacities of PWP (Zham et al., 2017).

"Micrographia" has been utilized from beginning time conclusion of PD. While penmanship of an individual is impacted by various factors, for example, language capability and instruction, drawing of a shape, for example, "a guided spiral" has been seen as "non-intrusive" (Zham et al., 2017).

Effect on speech

"Motor speech disorder is seen developing in 90% of PWP which is termed as Hypokinetic Dysarthria" (Viswanathan et al., 2018). This influences various degrees of discourse which encompasses "phonation, verbalization, prosody, and coherence". "Hypokinetic Dysarthria" influences "laryngeal capacity", "articulatory and respiratory muscles". More often than not, PWP won't know of the problem occurring with the correspondence. The average PD patient's voice is related frequently with "low volume, monotone, high pitch, tremulous, rasp, dryness" (Viswanathan et al., 2018), and some of the time failure to maintain loudness.

With further study on literature we found 9 such features commonly associated with PD patients. The said features are denoted in table 1.1.

Feature Sets
Fundamental frequency (F0)
Standard deviation of F0 (SD F0)
Mean F0
Formant frequencies (F1, F2, F3)
Jitter-Shimmer
Glottal features and glottal based spectral features (formants)
Features based on harmonics (H1, H2, H3)
Harmonics:Noise Ratio (HNR)
Mel frequency cepstral coefficients (MFCC)

Table 1.1: Speech features commonly associated with PD

We are extracting said features for analysis which will be discussed throughout the paper.

1.2 Archimedean Spiral

The "Archimedean spiral" (otherwise called the "arithmetic spiral") is named after the Greek mathematician Archimedes. The spiral is defined as "the locus of points comparing to the areas after some time", of a point moving endlessly from a fixed point with a steady speed along a line that rotates with constant angular velocity. Identically, in polar directions (r, θ) it very well may be portrayed by the equation

$$r = a + b\theta \quad (1.1)$$

Plotting from 1 to i:

$$t = i/20 * \pi \quad (1.2)$$

$$x = (1 + 18 * t) * \cos(t) \quad (1.3)$$

$$y = (1 + 18 * t) * \sin(t) \quad (1.4)$$

where (x,y) are the dots using turtle in a scripting language like python, we get the spiral as shown in Figure 1.1 used for our study.

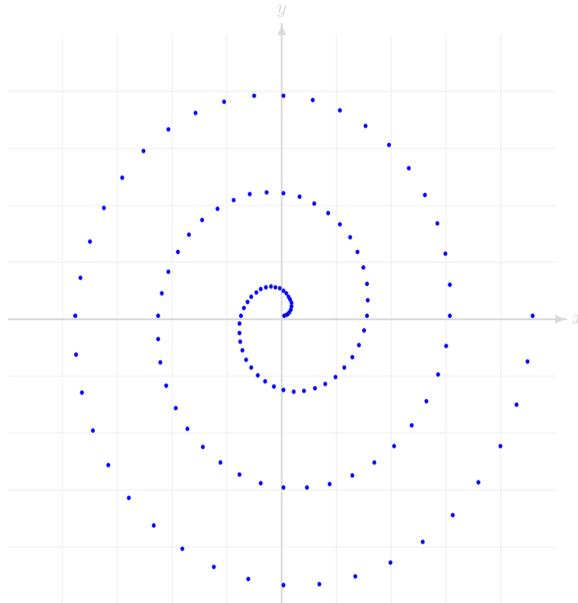


Figure 1.1: Archimedean Spiral

CHAPTER 2

LITERATURE SURVEY

2.1 "Distinguishing different stages of Parkinson's Disease using Composite Index Of Speed And Pen Pressure of a spiral sketch"

"The speed and pen-pressure while drawing a spiral are lower among Parkinson's ailment (PD) patients with higher seriousness of the illness" (Zham et al., 2017). In any case, the relationship between these highlights and the acuteness of PD "has been accounted for to be 0.4" (Zham et al., 2017). There is a requirement for recognizing parameters with a more grounded connection for thinking about this for the sickness' precise determination. This examination "proposed the utilization of the Composite Index Of Speed And Pen-Pressure (CISP) for portraying as a component for breaking down the PD acuteness". According to Zham et al. (2017), "a sum of 28 CG and 27 PD patients (absolute 55 members) were enlisted and evaluated for UPDRS".

"The exploratory convention was endorsed by RMIT University Human Research Ethics Committee and as per Declaration of Helsinki (updated 2004). All members were educated about the examination, and they gave their oral and composed assent before the beginning of the investigation" (Zham et al., 2017). One inadequacy in the utilization of spiral illustration is the critical variety between the diverse investigations. One alternative is the place the members are allowed to draw the spiral, which has the impediment that there is critical between member changeability. The other format is by giving a consistent spiral to the members to follow, which anyway isn't attainable for some old members.

2.2 "Efficiency of Voice Features based on Consonant for Detection of Parkinson's Disease"

"The target of the examination was to choose the viability of features isolated from proceeded with voiced consonant /m/ in the finish of PD" (Viswanathan et al., 2018). The diagnostics real nature of /m/ was similarly differentiated and that of bolstered /a/, the one which is routinely used in PD talk inspects. The examination included "40 subjects out of which 18 were PD and 22 were controls" (Viswanathan et al., 2018). The features removed were used in "SVM classifier model to isolate PD and strong subjects". According to Viswanathan et al. (2018), "the /m/ yielded portrayal accuracy of 93% and Matthews Correlation Coefficient (MCC) of 0.85 while the course of action precision for phonation/a/was 70% and MCC of 0.39" . The spearman relationship coefficient assessment in like manner revealed that "the features from /m/ phonation were astoundingly connected with the Unified Parkinson's Disease Rating Scale (UPDRS-III) motor score". The "tangibility of features" identifying with "nasal consonant" in this investigation and development checking of the condition, was proposed by the findings,

"It was seen from the results that the phonetic features expelled from the proceeded with consonant /m/ isolated PWP". Furthermore, "solid controls yielded an accuracy pace of 93% in the gathering model. This request accuracy is generally high stood out from that of /a/ which was 70%. For the phonation /m/, MCC was 0.85 and for phonation /an/ it was 0.39" (Viswanathan et al., 2018). The affectability and expressness was high for the request model subject to /m/ phonation, in comparision with the phonation /a/.

2.3 "Speech Features for Telemonitoring of Parkinson's Disease Symptoms"

This particular work by Ramezani et al. (2017), stresses on monitoring development of Parkinson's disease based on "the vocal system symptoms using the Unified Parkinsons Disease Rating Scale (UPDRS)". They used a standard "voice signal feature set" com-

prising "6373 static features" as "low-level descriptor (LLD) contours" functionals and selected "the most insightful ones based on correlations (mRMRC) criteria" based on what they described "maximum relevance and minimum redundancy."

The analysis of their extracted features showed that "LLDs giving information on spectrum flatness, spectral energy distribution, and voice heaviness are the most significant ones for estimating UPDRS-III", and the size, peak, low, and standard deviation of LLDs are linked to informative statistical functions as the condition causes the muscles to be weaker by affecting the contraction-relaxation abilities. They concluded that: "GMR exceeds SVR for less selected features" and "increasing number of selected features to 1000 and applying SVR enhances the efficiency of estimating UPDRS-III based on the Spearman correlation criterion" ,at the same time, "further increase in selected features decreases the accuracy." Records ,as in the data sets were used from previous works. These according to Ramezani et al. (2017), included a "total of 50 patients with Parkinson's disease (25 female, 25 male), where 35 patients are included in the training set, and the remaining 15 patients are included in the test set." Every subject completed: forty-two vocal activities, which included twenty-four isolated phrases, ten unique sentences, a single reading text, a single "monologue", and "the rapid repetition of syllables /pa-ta-ka/, /pa-ka-ta/, and /pe-ta-ka/." Patient's "neurological condition" were "evaluated by an expert neurologist according to the UPDRS metric".

CHAPTER 3

IMPLEMENTATION

3.1 Micrographia as a discriminative tool for Parkinson's Disease

3.1.1 Significance of extraction of micrographia data

The "relationship of handwriting and drawing of the spiral" in determination of PD has been established early in studies by Ramezani et al. (2017). In any case, one shortcoming in the usage of "handwriting or plotting" is that it necessitates a professional for unravelling recorded sketches, more so during the "acute stages of the illness". The accessibility of digitiser devices which seem sensible for capturing pen data, revealed a huge opportunity for machine-based examination of forming and laying out. These contraptions are moreover fitting for securing the dynamic features of portraying, which are sensible for consistent and strong examination. These "features" are extracted normally which "allow quick on-line evaluation of patients and made for applications, for instance, bio-metrics and trademark markers for PD". "The Kinematics of spiral drawing shows physiological parameters, for instance, the sufficiency of tremor and level of bradykinesia and dyskinesia" (Zham et al., 2017). It has been appeared to adequately isolated among "distal and proximal tremors and between control social affair and PD". The pen-pressure of patients is another component connected with depicting and differentiating between PD and CG groups.

While, this method is sensible for perceiving sound subjects or people afflicted by the "early stages of the condition", it has "not been exhibited to be fitting for perceiving different periods of the illness". Our purpose is to set up a strong "PC-based spiral drawing procedure" for evaluating the curve the progression takes. We look to study about relations of drawing a spiral to perceive CG and PD subjects with varying degrees

of reality and offer another component with more grounded relationship of the disorder with said progression. Earlier examinations have "set up that speed and pen-pressure during laying out decrease with the progress of the infection" and yet did not think about the blend of the two features. This examination made use of the "scalar consequence of these two features" for obtaining the CISP of drawing and attempted this against the reality of the disease.

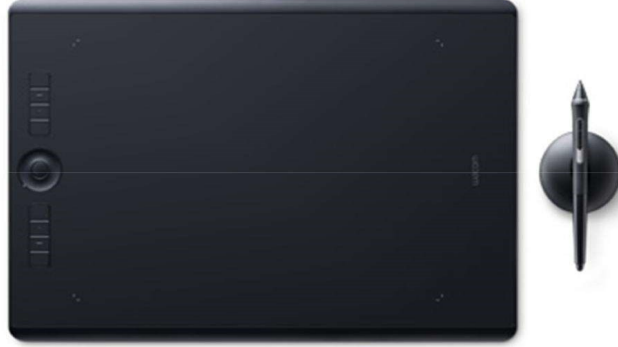


Figure 3.1: A typical handwriting digitiser device (developer-docs.wacom.com, 2010)

3.1.2 Working Model

- The total occurrences of "Pen up-down", "Time-elapsed", "Weighted Average Speed", "Weighted Average Pressure", "Number of Pressure Points" and "CISP" is recorded from the digitiser device (Fig. 3.1) by developing a proprietary digitiser data recording software application.
- **Pen up-down:** As the digitizer records coordinate data in real-time, the application resolves differences in pen pressure with an innovative algorithm to single out the number of times the PWP or CG user makes and breaks contact with the recording surface with the tip of the digitiser pen.
- **Number of Pressure Points:** Every time a PWP or CG user exerts a pressure which differentiates itself within the stream of continuous pressure data, said points are incremented.
- **CISP:** Composite Index Of Speed And Pen-Pressure is given by \bar{I}_{spr} , where \bar{S}_w is "Weighted Average Speed", $\bar{p}r_w$ is "Weighted Average Pressure", " x_n and y_n corresponding to the nth sample of data".

$$d_i = \sum_{n=0}^{m_i} \sqrt{(x_{(n+1)} - x_n)^2 + (y_{(n+1)} - y_n)^2} \quad (3.1)$$

$$T_i = \frac{m_i}{SamplingRate} \quad (3.2)$$

$$s_i = \frac{d_i}{t_i} \quad (3.3)$$

$$\bar{S}_w = \frac{\sum_{i=1}^N d_i s_i}{\sum_{i=1}^N d_i} \quad (3.4)$$

$$\bar{p}r_w = \frac{\sum_{i=1}^N d_i \bar{p}r_i}{\sum_{i=1}^N d_i} \quad (3.5)$$

$$\bar{I}_{spr} = \bar{S}_w * \bar{p}r_w \quad (3.6)$$

- The recorded data is exported as CSV files to application User-Interface (UI).
- A highly interactive UI interacts with researcher to record, organize and analyse recorded data.

3.1.3 Based on the WIN32 API

"The Win32 API (also called the Windows API) is the original platform for native C/C++ Windows applications that require direct access to Windows and hardware" (docs.microsoft.com, 2015). Thus, it provides a superior development experience without any dependency like: a "managed runtime environment like .NET" or "WinRT for UWP apps for Windows 10". As is reported by docs.microsoft.com (2015), "this makes the Win32 API the platform of choice for applications that need the highest level of performance and direct access to system hardware".

The applet written to access raw digiter (Fig: 3.1) data leverages the Win32 API. This applet forms an integral part of the larger application(UI) as it handles the "record" part of the Data acquisition (DAQ). The applet also leverages "The Microsoft Windows graphics device interface (GDI)" which "enables applications to use graphics and formatted text on both the video display and the printer. Windows-based applications do not access the graphics hardware directly. Instead, GDI interacts with device drivers on behalf of applications" (docs.microsoft.com, 2005). This specifically enables this study to create a ink-visualization of recorded data. Further light will be shed on this topic in the Results section of the paper.

3.1.4 Based on the WINTAB API

Wintab is an API that is included in Wacom tablet drivers. According to developer-docs.wacom.com (2010), "Wintab is a de facto API for pointing devices on Windows. The Wintab specification is an open industry standard interface that provides access to pointing devices such as a pen tablet, for example".

The "Wintab module is a wrapper around the Wintab Developer Kit and can be utilized to add tablet backing to our python application". However, before we can prise any information from the digitiser we need to make a "case of the Context class, set the ideal parameters and open the unique situation". When the route is open we will either get "messages from the tablet driver" or we can "survey the flow tablet state". In any case, we get the tablet information by means of Packet protests that contain all the information that was produced by the tablet (developer-docs.wacom.com, 2010). These packets contain all the required data : current coordinates and packet data. These two variables are then put through our algorithm to extract the micrographia data as explained in section 3.1.2.

Tablet settings produce and report tablet action through occasion bundles. Applications can control how they get occasions, which occasions they get, and what data they contain. Applications may get occasions either by surveying, or through Windows messages.

- Polling: Any application that has opened a setting can call the WTPacketsGet capacity to get the following condition of the tablet for that unique situation.
- Window Messages: Applications that solicit messages will get the WT_PACKET message , which shows that something occurred in the unique situation and gives a reference to more data.

Applications can control which occasions they get by utilizing occasion covers. For instance, some applications may possibly need to know when a catch is squeezed, while others may need to get an occasion each time the cursor moves. Tablet setting occasion veils actualize this kind of control. Applications can control the substance of the occasion bundles they get. A few tablets can return information that numerous applications won't need, similar to fasten weight and three dimensional position and orientation information. The setting object gives a method for indicating which information things

the application needs. This permits the driver to improve the effectiveness of bundle conveyance to applications that solitary need a couple of things for each packet. Packets are put away in setting explicit bundle lines and recovered by express capacity calls. The interface gives approaches to look at and get parcels, to question the size and substance of the line, and to re-size the queue. The interface gives capacities to tablet the board. An application can turn into a tablet chief by opening a tablet director handle. This handle permits the chief access to special capacities. These management capacities permit the application to orchestrate, cover, and adjust tablet settings. Managers may likewise perform different capacities, for example, changing default esteems utilized by applications, changing ergonomic, inclination, and setup settings, controlling tablet conduct with non-tablet mindful applications, modifying client discoursed, and recording and playing back tablet packets. Opening a chief handle requires a window handle. The window turns into a chief window and gets window messages about interface and context movement.

3.1.5 Algorithm description for Implementation of extraction and processing

Pre-Processing of Micrographia data

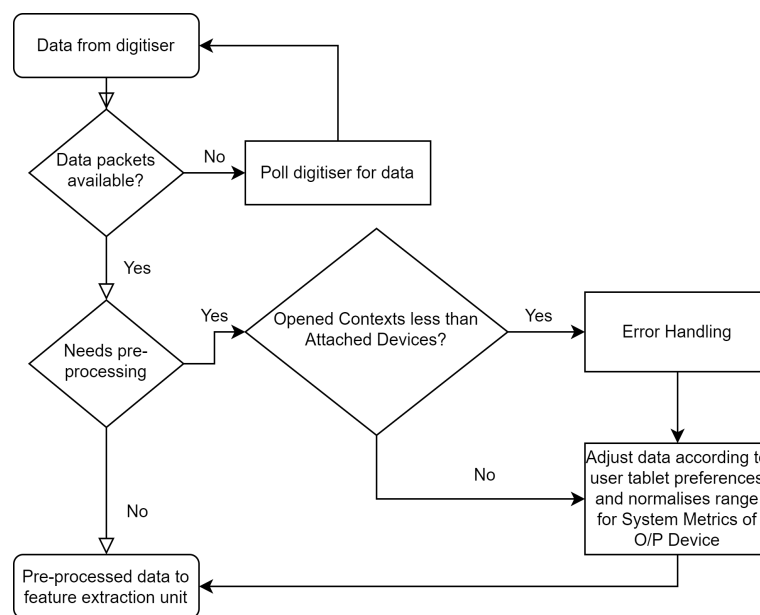


Figure 3.2: Flow-Chart for pre-processing of Micrographia data from digitiser

The incoming data from the digitiser device is received in intervals, in a group of regular bytes known as “data packets”. These packets are not always available and their availability depends on a host of parameters. The most significant of these parameters being data availability itself. When the PWP or CG subject is made to draw upon the “Archimedean guided spiral” (micrographia data acquisition), the digitiser converts transducer data to binary data and uses the driver protocols to send the data to the module in said packets.

However, this data is generalized and needs to be “pre-processed”. Pre-processing the data marks an essential phase in the process of data mining. Especially necessary for "data mining and machine learning projects" is the expression "garbage in, garbage out". Methods of data collection are also "poorly regulated", resulting in "out-of-range" values (for example, pen-pressure: -100), unlikely combinations of data (for example, coordinate data: null, pressure data: present), missing values, and so on. Analysing data which has not been screened carefully for these problems may yield misleading results. Therefore, the interpretation and accuracy of the data is mainly before an experiment is carried out.

The data obtained from the digitiser device is thus fed to a novel algorithm depicted through a concise flowchart in Fig. 3.2. The extraction module pings the driver software for data from the digitiser through a process known as “polling”. Whenever data packets are available for use the module picks them up checks for irregularities (a few of which have been discussed in prior passages). If it checks positive for irregularities then the data packets are sent to the pre-processing module.

Prior to this a check is done by the extractor module to check for whether the data is true legible data generated due to user request or random transducer outputs. This is done by confirming that the number of “Opened Contexts”, or applications demanding data, is more than “Attached Devices”, or data output from the digitiser device. Then the pre-processing taking place removes data irregularities and adjusts data according to user preference, normalizing range for “System Metrics” of output device. After all of the said processes are over, this pre-processed data is sent to the feature extraction unit.

Feature Extraction from Micrographia data

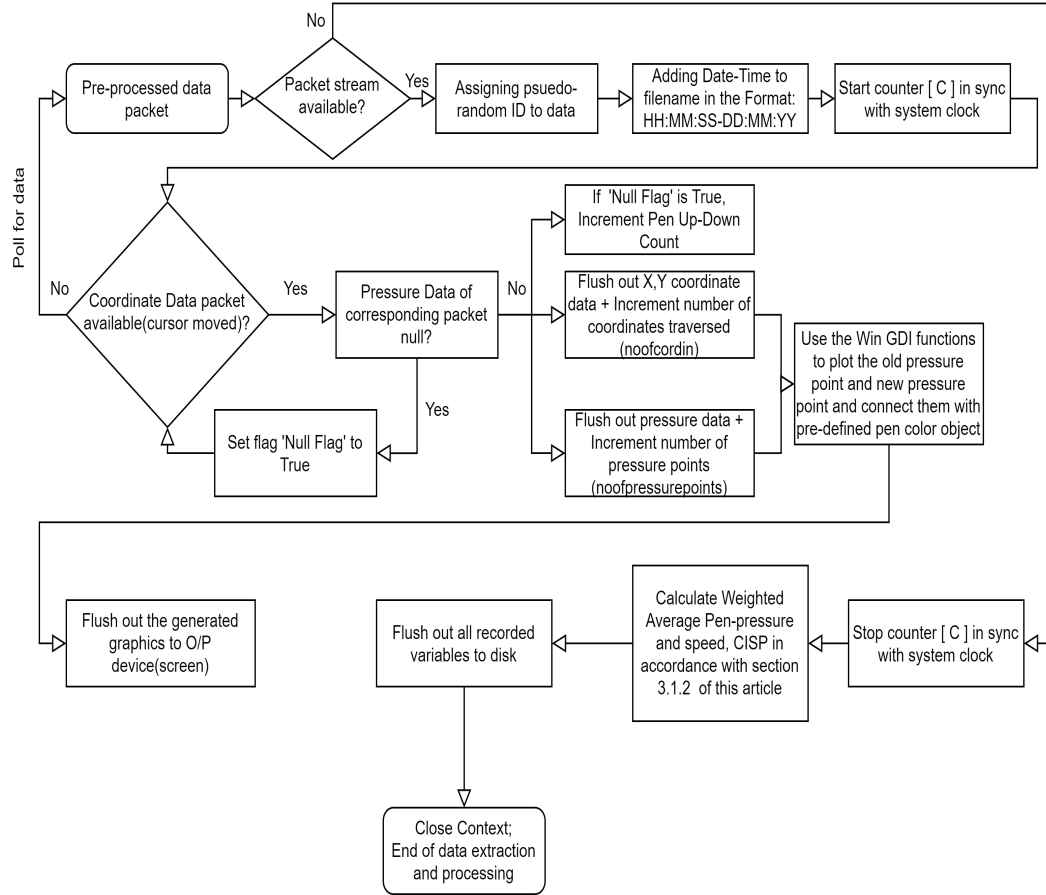


Figure 3.3: Flow-Chart for Feature Extraction from Micrographia data

The pre-processed data stream, as a result of discontinuous raw data, is also discontinuous. Whenever a data stream is available, the feature extractor unit pulls the data packet and starts operations. A pseudo-random ID is assigned to every data packet before adding it to a file chunk. The filename conventions are followed by adding “Date-Time” to every file in the Format: “HH:MM:SS-DD:MM:YY”. Immediately after the first packet is received the module starts a counter in synchronisation to the system clock. The data inside the packet is peeked to find if the coordinate data (cursor movement) is available. If the corresponding pressure data is peeked to be null it signifies that the subject is just hovering the pen and not actually writing, in which case the software flag “Null Flag” is raised. The module continues to poll for data until a positive pressure data is available.

In case of a proper data combination, the control passes to the next level where multiple control statements occur. If the “Null Flag” is found raised, the count variable

“Pen Up-Down Count” is incremented. The surface X, Y coordinate data are flushed out to disk and the “Number of coordinate traversed” variable is incremented. The corresponding pressure data is also flushed to disk and the counter variable “Number of Pressure-points” is incremented.

The next level focuses on generating the “pen ink visualization” graphics. As discussed in section 3.1.3, the Win GDI tool stacks are used to plot the old and new pressure points and connect them with a pre-defined “pen-colour object”. This is a 32-bit graphics plot which is then flushed out to the output device (computer screen) in real-time. This entire process continues till the data stream stops definitely.

Once the packet streams are no more available, the counter clock in synchronisation with the system clock stops. This signifies the end of the micrographia data input, i.e. the user input stops. This moves the control to the feature calculation part where features like “Weighted Pen-pressure and speed” and “CISP” are generated in accordance with section 3.1.2 of this article. The recorded is flushed out to disk once the data generation is complete. The feature extractor module then closes all contexts and passes the control back to the user interface. This entire process is depicted as a flowchart in Fig. 3.3.

3.2 Speech Data: A discriminative tool for Parkinson’s Disease

3.2.1 Significance of extraction of speech data

The works of Viswanathan et al. (2020) say "Using automated speech analysis to diagnose and monitor PD has several advantages as has been seen from the works of, primary among them, the ability to make unbiased and objective measurements." Significantly, machine-analysis of speech data likewise permits the progression of PD to be observed remotely diminishing the number and cost of actual hospital visits. PWP have diminished "Neuromotor Control", which influences the voice component and along these lines the voice components help assist the "Acoustic Analysis of the voice" of

PD patients, which have remedial applications additionally. "Voice Analysis" can be utilized when assisting inflicted patients in therapy. Machine examination of discourse could expand standard clinical evaluations in the early discovery of PD, yet there are critical entomb and intra-singular varieties, which makes this difficult.

In a study by (Viswanathan et al., 2018), it states that "the voice of PD patients is associated with low volume, monotonicity, high pitch, tremulousness, breathiness, hoarseness and, sometimes, the inability to spontaneously maintain loudness." The parameters obtained by audio analysis get the advantage of impartially representing the speech. It is conceivable to understand ordinary and obsessed speech or even to discern or suggest the disease in the participation of controlling repositories representing speech content by utilizing clever instruments joining the different criteria. Such tools require all psychological disposition and research to be examined, and lower the degree of perceptual test subjectivity. In this way, highlights need to be discovered to discern similarities between speech and neurotic voices that can be tailored for a given populace.

3.2.2 Analysing acoustic parameters

The Fundamental Frequency

The voice can be described by many a few parameters. The pitch of the voice is one of the primary qualities, however in the field of acoustic advancements the right name of this parameter is the F0. The F0 is straightforwardly identified with what we call the sound. Also, the pitch, for instance, is related with expressive attributes of the voice. The F0 is the vibration recurrence of the tendons when articulating voiced sounds. There are various challenges one encounters while assessing F0. For instance, it tends to be frequently mistaken for harmonics — this can prompt alleged pitch multiplying/pitch splitting (NeuroData Lab, 2018).

Air rises from both the lungs during every exhale, and passes via the voice box. They will continue to vibrate at various specific frequencies on the off chance that the pulmonary membranes are closed inside the voice box during exhalation. The base pitch being sung is the most stable and slowest sound. The stronger sounds that exist

concurrently are called overtones, or harmonics.

According to NeuroData Lab (2018), the techniques for estimation of F0 can be separated into these groups: in light of "temporal dynamics of the signal", or "time-area; in view of the frequency structure or frequency domain," and "hybrid strategies".

- "Time-domain": Prior to applying time-domain methods, the signal is isolated to filter out the lower frequencies. The edges are set: least and greatest frequencies, for example, the range: 75-500 Hz. F0 estimation is simply coordinated for harmonic discourse. Auto-relationship is the essential calculation, the approach is really fundamental: you compute the segment in the sign, learn the auto-correlational limit and thereafter portray it's first peak, that will reflect the most striking frequency segment in the signal.
- "Frequency-domain": Expanding about frequency domain, the most prominent angle seems to be the harmonic nature and structure of the signal. To explain it differently, "spectral peaks" at the recurrence that is separable into F0. This occasional example can be transformed into an undeniable peak with the assistance of Cepstrum examination. NeuroData Lab (2018) defines cepstrum as "a Fourier transform (FFT) of the logarithm of estimated power spectrum" . The peak of the cepstrum corresponds to the most periodic segment of the signal.
- "Hybrid F0 estimation methods": "YAAPT — Yet Another Algorithm of Pitch Tracking", declared as hybrid as it utilizes both "temporal" and "frequency" information.

The algorithm used by this project is the "PRAAT algorithm". "The algorithm performs an acoustic periodicity detection on the basis of an accurate autocorrelation method", as depicted in Boersma (1993). The above strategy is increasingly exact, common safe, and strong, than techniques dependent on cepstrum or brushes, or the first auto-correlation strategies.

Jitter and Shimmer

According to Teixeira et al. (2013), Jitter and shimmer measurements have proved helpful in defining the characteristics of the vocals. Jitter is known as the "frequency variation parameter from cycle to cycle, and shimmer relates to the sound wave amplitude variation". These factors can indeed be continuously tested under "a steady voice which produces a vowel" (Teixeira et al., 2013). This jitter becomes primarily induced by the

absence of chord tension regulation; there is also a greater jitter rate in the voices of patients with pathologies. Most studies find a typical meaning variation of between 0.5 and 1.0 per cent for constant phoning in young adults. The shimmer increases with "decreased glottal resistance and mass lesions on vocal cords", which is consistent with the "frequency of noise output during respiration". It is called pathological speech "for values less than 3 per cent for adults and about 0.4 and 1 per cent for children" (Teixeira et al., 2013).

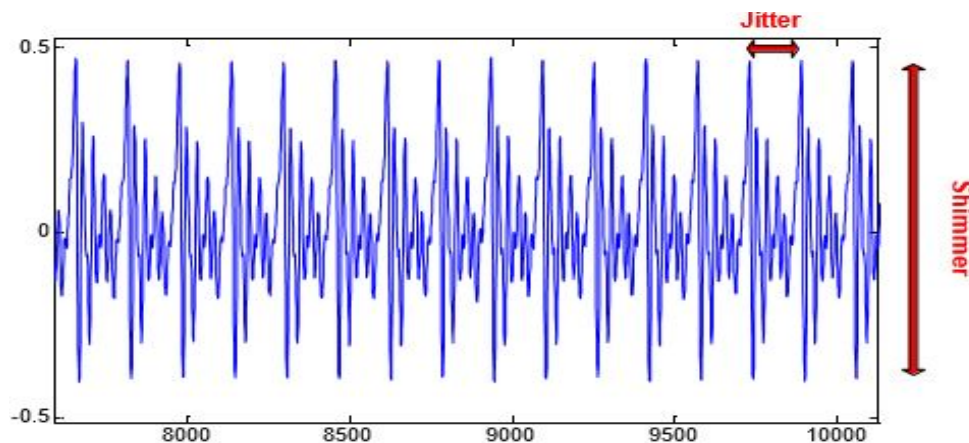


Figure 3.4: Portrayal of Jitter and Shimmer interference steps in speech transmission (Teixeira et al., 2013)

The determination of jitter is done as follows:

- In order to evaluate this parameter, which reflects the variety of progressive time frames, the calculation began to update a capacity that distinguishes the planning of the F0 period.
- The control yield function contains the pinnacle rates related to that of the beginning of a "glottal pulse signal", which implies that such a strength can generate a function of the same scale along with the pinnacles themselves.
- The above function tests the regular patterns of the Signal, and afterwards employs a weighted average duration of around 10 ms (a range like "one glottal period").
- At this stage, after the moving average maximum index, the peak is searched as the "Acoustic Signal Maximum" under a "previous window of 15 samples", and the same number of samples.

Upon deciding the "starting time of the glottal pulses", the jitter is calculated by the formulas shown below for its different measurement shapes:

- Jitter of the type absolute and also local reflects the absolute "average difference between two consecutive periods", and thereby called "*jitta*".

$$jitta = \frac{1}{N-1} \sum_{i=1}^{N-1} |T_i - T_{i-1}| \quad (3.7)$$

- Jitter (Type: local) is the average absolute difference, separated by the average duration, between two consecutive periods. It is known as *jitt* and has a "threshold limit" of 1.04 percent for identifying pathologies (Teixeira et al., 2013). "Where T_i is the length of each period in seconds, and N the number of periods."

$$jitt = \frac{jitta}{\frac{1}{N} \sum_{i=1}^N T_i} \times 100 \quad (3.8)$$

- Jitter (Type: rap) reflects the "average disturbance", i.e. the actual "absolute difference of one period and the actual period separated by the average period with its two neighbours." The threshold value 0.68 percent is used to detect disorders (Teixeira et al., 2013).

$$rap = \frac{\frac{1}{N-1} \sum_{i=1}^{N-1} \left| T_i - \left(\frac{1}{3} \sum_{n=i-1}^{i+1} T_n \right) \right|}{\frac{1}{N} \sum_{i=1}^N T_i} \times 100 \quad (3.9)$$

The determination of Shimmer is done as follows:

- The techniques used during the determination of Jitter were followed to determine the parameters for shimmer. The algorithm began by determining at that point the onset duration of "glottal pulses of the signal" and the respective pulse intensity. At this point the algorithm was implemented to calculate the estimates for each shimmer parameter correspondingly with respect to the jitter.

The following types of shimmer are extracted from the PWP or CG speech data:

- Shimmer of the type "local" represents the mean absolute deviation, separated by average amplitude, in between "two amplitudes of two consecutive periods". It is called a "shim" and it was 3.81 percent as the mark for pathology detection (Teixeira et al., 2013).

$$Shim = \frac{\frac{1}{N-1} \sum_{i=1}^{N-1} |A_i - A_{i+1}|}{\frac{1}{N} \sum_{i=1}^N A_i} \times 100 \quad (3.10)$$

- Shimmer of the type "dB" and "local" gives "the average absolute deviation between two consecutive cycles of the base 10 logarithm, and is called *ShdB*. The cap on pathology detection is 0.350 dB" (Teixeira et al., 2013).

$$ShdB = \frac{1}{N-1} \sum_{i=1}^{N-1} \left| 20 * \log \left(\frac{A_{i+1}}{A_i} \right) \right| \quad (3.11)$$

- Shimmer (Type:apq3) reflects the amplitude disturbance quotient within three periods, that is, the mean absolute deviation "between the amplitude of a period and its two neighbors' mean amplitudes, divided by the mean amplitude".

$$apq3 = \frac{\frac{1}{(N-1)} \sum_{i=1}^{(N-1)} \left| A_i - \left(\frac{1}{3} \sum_{k=1}^M A_i \right) \right|}{\frac{1}{N} \sum_{i=1}^N A_i} \times 100 \quad (3.12)$$

Harmonics to Noise Ratio (HNR) and Formants

The HNR is an evaluation of the proportion between sporadic segments and non-occasional segment including a voiced discourse component. The main element arises from the vocal ropes vibration, and the second is extracted from the glottal commotion, represented in dB. The distinction between the two sections shows the intensity of the argument, that is, the more impressive the transformation of the air extracted from the lungs into the vocal lines' vibrational force.

The HNR will take on more importance in such situations. In this way, a speech sound is represented by a high HNR similar to sonorous and symphonic speech. A low HNR means dysphonia and an asthenic voice. That is, an average of under 7 dB is regarded as neurotic in HNR.

Determination of HNR is done as follows:

- The "Harmonics to Noise Ratio" (HNR) architecture was based on the fundamentals of mathematics. This started by the detection of the voice signal's autocorrelation feature.
- The very first local is, however, the peak located at Fig. 3.5 and this value only compares to the corresponding index peak 1. The equation's " $ACV(T)$ is the value" that correlates to the signal time at the index spot. Then the estimated values for F0, for that level, define a position between the two indices.
- Considering the generally accepted values for F0 (for females: 200 to 300 Hz; Man: somewhere between the range of 80 and 200 Hz and for child voice it differs somewhere within the range of 400 and 500 Hz (Teixeira et al., 2013)) for the primary index ($fs/f0max$) and the corresponding index ($fs/f0min$). In the wake of determining the indices, finding their separate amplitude, the nearest biggest is found within the first and second number. The estimation of HNR is discovered at this point applying the corresponding formula,

$$HNR = 10 * \log_{10} \frac{AC_{\nu}(T)}{AC_{\nu}(0) - AC_{\nu}(T)} \quad (3.13)$$

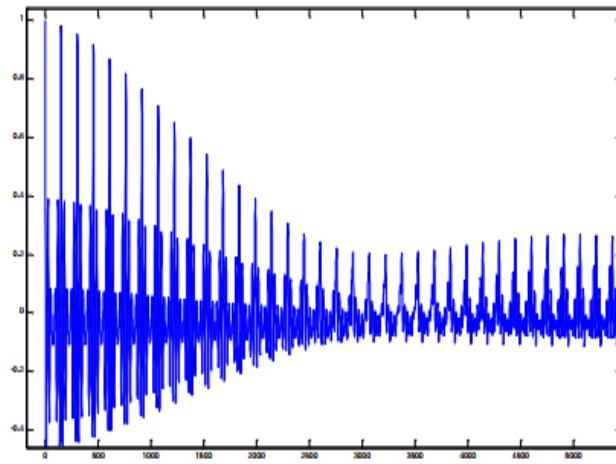


Figure 3.5: Portrayal of the 1st local maximum of the autocorrelation result (Teixeira et al., 2013)

Most of the PD diagnostic studies used continuous /a/ phonation as the voice function and a particular handful of experimentations have used "/e/, /i/, /o/ and /u / phonation" (Viswanathan et al., 2018). Some of the significant drawbacks of using "sustained vowel phonation /a/ is that it can not provide much information about the articulator". It has been shown for some research that the "voiced consonants are susceptible to higher PD distortion", and consonant articulation difficulties shown by the subjects.

Consonant development, especially including specially articulated nasal consonants, require "regulation of the pacing of breathing, larynx, articulators and contact force between the lips, tongue and palate" (Viswanathan et al., 2018).

Also according to Viswanathan et al. (2018), "phonetic features derived from the sustained consonant /m/ segregated PWP and healthy controls with an accuracy rate of 93%". This precision is "considerably higher than that of /a/ which was 70%". The relationship between the "UPDRS-III score" and the "speech characteristics" was evaluated using the Spearman correlation factor. Only the "reduced features" and the "UPDRS-III" score checked the relation. It can be seen from their results "that the /m/ features were highly associated with the UPDRS-III value ($p < 0.001$)".

The harmonic and formant series also have a unique role in differentiating the voice recordings of PWP and that of the CG.

- The first vocal fold vibration has to be transient for the harmonic sequence to

happen. In case the first vibration (or the fundamental pitch) is intermittent at predictable frequencies than the higher vibrations would be.

- The second harmonic will vibrate, consistently, "twice as much as the fundamental frequency".
- The third Harmonic can continuously vibrate as quickly as the fundamental pitch several times, and so forth.
- The harmonics are consistently to be a multiple integer of F0.

Vowel formants (F1, F2, F3 and so forth.) are conspicuous frequencies in human voice spectra which are said to separate various vowels from one another. In any case, formant frequencies of indeed the very same vowel change in way to express various speakers in an extremely huge degree. The formant with the lowest recurrence is called F1, similarly, the successive one F2, then F3 the one after, and so on. The first two formants, F1 and F2, are usually necessary to identify vowels (Pratheeksha Nair, 2018).

- "/i/ has a low F1 (300 Hz) and a high F2 (2500 Hz)"
- "/a/ has a high F1 (800 Hz) and a close F2 (1,000 Hz)"
- "/u/ has a low F1 (300 Hz) and a low F2 (900 Hz)"

Mel-frequency cepstrum (MFCC) and Mel Scale

In MFCC, Cepstrum is Spectral Band Change Rate information. Every periodic variable (for example, echoes) appears as "sharp peaks in the corresponding frequency spectrum (i.e., Fourier spectrum) in the standard time signals analysis". This is accomplished by applying a "Fourier transform on the time signal"). Taking the log of this Fourier spectrum magnitude and then again taking this log spectrum via a cosine transition, we find a plateau if the original time signal has a periodic component. Because a shift is made to the frequency continuum itself, the resultant continuum is not in the region of frequency or in the time domain, as shown in Fig. 3.6. Hence, Bogert et al. (1963) went ahead and called it "quefrequency domain". And the range of time signal log range was called cepstrum.

Mel scale is a scale in which the perceived frequency of a tone is compared to the real measured frequency. This measures the frequency to better match what the human

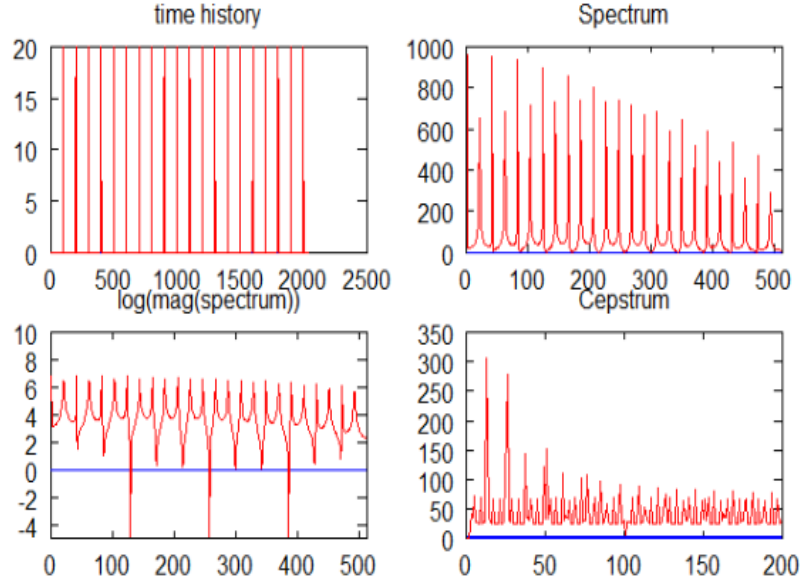


Figure 3.6: Steps in generation of MFCC visualized (Pratheeksha Nair, 2018)

ear can hear,

$$\text{Mel}(f) = 2595 \log \left(1 + \frac{f}{700} \right) \quad (3.14)$$

The various acoustic parameters which have been extracted from the voice recording of PD patients for showing how it varies from the control group have already been diverged in Table 1.1.

3.2.3 Based on the Parselmouth API

Jadoul et al. (2018) state that "Parselmouth is a Python Library for the Praat Software". Although there have been other attempts "to migrate functionality from Praat to Python", "Parselmouth is special in its goal of providing the internal Praat code with a complete and Pythonic interface". Whereas other projects either "wrap Praat's scripting language or reimplement parts of Praat's functionality in Python", Parselmouth explicitly uses Praat's native code (meaning the "algorithms and their performance are exactly the same as in Praat") and "offers effective accessibility to the software's code, and also offers an implementation that appears no different from any Python library".

As such, the performance of Parselmouth is especially swift. On the one side, when it comes to executing the functionality of Praat, the authors use the exact same code, so

it is assumed that Python scripts which access computationally costly Praat algorithms would take the same amount of time. Also, while evaluating scripts that have a "high degree of contact with the Python code and the Praat features", the author's "tests and comparisons" seem to suggest that the mixture of Python and Parselmouth runs "as quickly or even faster than the Praat interpreter's identical script does".

Leveraging the Parselmouth API Praat functions were called to implement algorithms mentioned in sections 3.2.1 and 3.2.2. Using praat calls from the parselmouth API such as this, frequent calls are made to Praat to query the recorded sound for data:

```
parselmouth.praat.call(sound, "To Pitch", 0, 75, ..., 0.14, 600)
```

The excerpt above takes a minimalist approach to visualise data querying from sound samples. Using such methods, essential data required for calculating the nine speech parameters as mentioned in Table 1.1, is mined from the recorded speech data from a PWP or CG.

3.2.4 Algorithm description for Implementation of extraction and processing

Speech Data Recording

The raw audio data for speech is captured from a two-way microphone setup. The "PortAudio" bindings for python are used to translate this binary data to a byte array. PortAudio is an open source, open access, free, cross-platform, audio I / O library. It offers a very easy sound recording and/or playback API using an easy call-back feature or a read / write interface that blocks. The user interface asks for storage preferences from user and commits changes to the working directory

Once storage preferences are sorted, the bindings are used to list audio input devices compatible with the OS platform according to their "device index". These indices allow the user to sort through the available devices to use to record audio. Once the user selects a suitable index the audio stream is opened using defined FORMAT (Audio Resolution), RATE (sampling rate at which audio is to be recorded), and CHUNK SIZE

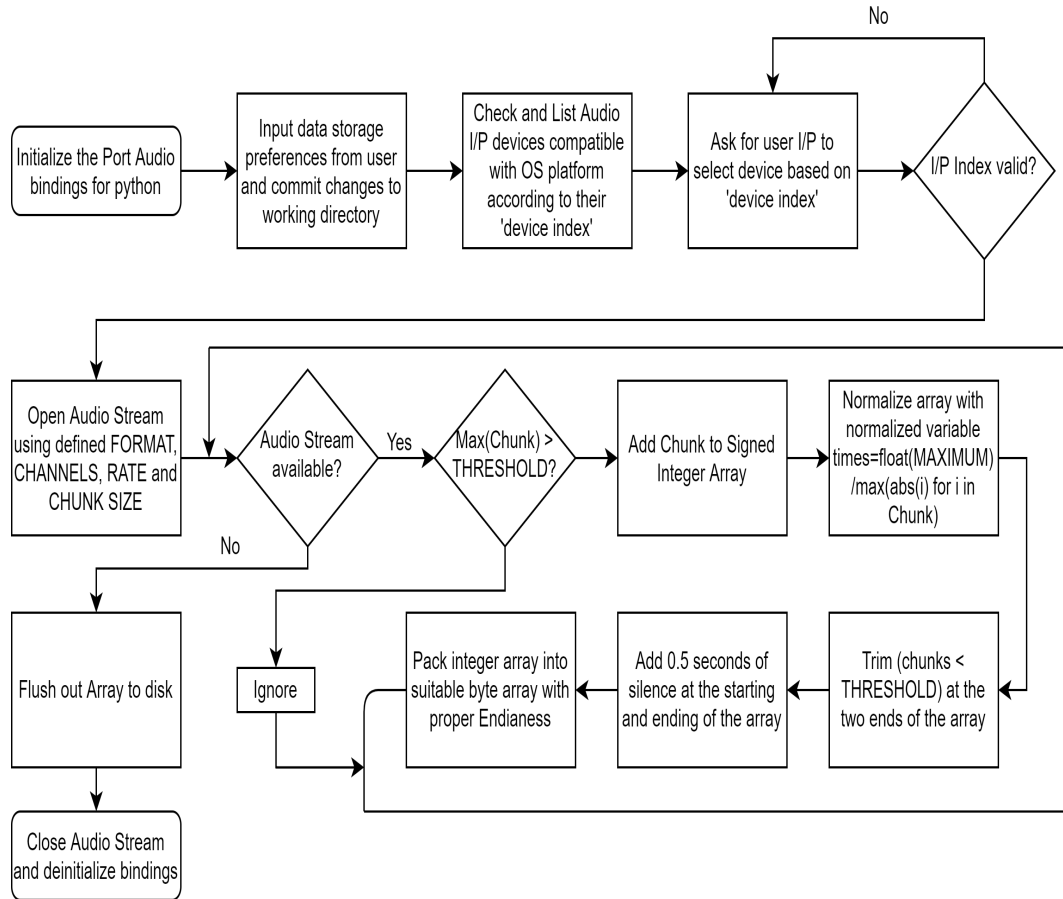


Figure 3.7: Flow-Chart for recording speech data from an audio data stream

(size in bytes of audio packets). The audio stream is read in a loop in fixed **CHUNK SIZE**. If the maximum intensity of amplitudes in the chunk is less than the defined threshold for amplitude (**THRESHOLD**) the chunk is ignored, else the chunk is added to a signed Integer array.

In every loop the signed Int array is normalized by a factor of ‘times’, where ‘times’ = (Maximum size of int in bytes) divided by the maximum of amplitudes among the chunk. This is also depicted in the flowchart shown in Fig. 3.7. A trim function removes amplitudes less than threshold in the beginning and end of the array, thereby removing leading and trailing noise in the recording. Also to avoid abrupt changes causing peaks during differentiation, a leading and trailing 0.5 seconds of silence is added to the array. This loop continues until there is a definite stop in the Audio stream, denoting a stop in recording.

Once the recording stops, the integer array is packed into a suitable byte array with proper Endianness. The order of bytes within a binary representation of a number (or

often bits) is referred to as endianness. This is to maintain compatibility with most sets of computer architecture. The byte array is flushed out to disk in .wav format. The Audio stream is closed and the bindings reinitialized.

Feature Extraction from Speech Data

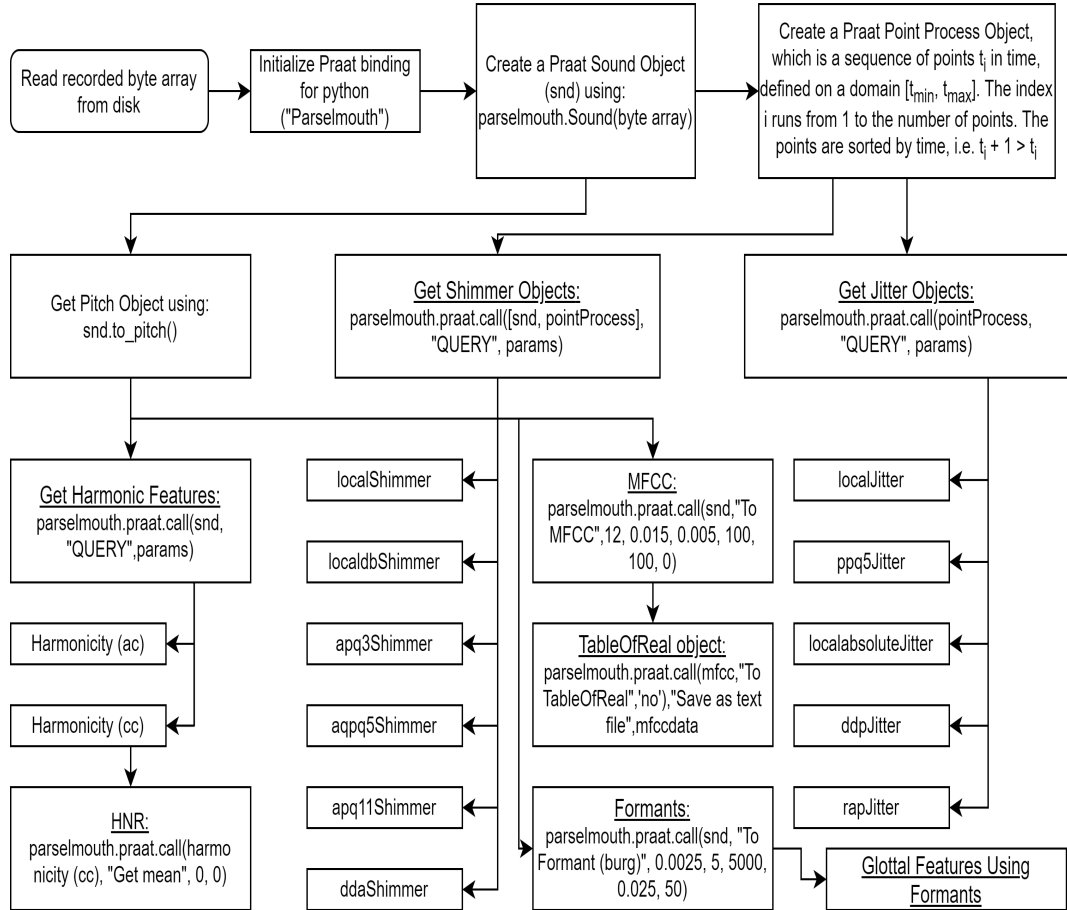


Figure 3.8: Flow-Chart for Feature Extraction from Speech Data

The recorded .wav file is converted from binary by unpacking it to a byte array using the ‘PortAudio’ bindings. The earlier discussed ‘Parselmouth’ is used to initialize the ‘PRAAT’ backend via bindings. A praat sound object is created which will act as the placeholder for the recorded audio from which the features are to be extracted. Further from this object a “Point Process” object is created. As is stated in the manuals of Boersma and Weenink (1991), “a PointProcess object represents a point process, which is a sequence of points t_i in time, defined on a domain $[t_{min}, t_{max}]$. The index i runs from 1 to the number of points. The points are sorted by time, i.e. $t_i + 1 > t_i$ ”.

This point process object is further used to get Jitter objects by using praat calls

from the bindings. The praat calls are in the format:

```
parselmouth.praat.call(pointProcess, QUERY, params)
```

Here, the “QUERY” changes according to the praat manual. For example, to get the “five-point Period Perturbation Quotient” jitter, commonly known as ppq5 Jitter, we replace “QUERY” with “Get jitter (ppq5)”. Similarly, the ‘params’ or parameters are defined by the praat manual. For ppq5 Jitter the parameters are “Time-Range (s)” = 0-0, “Period floor (s)” = 0.0001, “Period ceiling (s)” = 0.02, “Maximum period factor” = 1.3. The praat manual by Boersma and Weenink (1991) defines “Maximum period factor” as “the largest possible difference between consecutive intervals that will be used in the computation of jitter. If the ratio of the durations of two consecutive intervals is greater than this, this pair of intervals will be ignored in the computation of jitter (each of the intervals could still take part in the computation of jitter in a comparison with its neighbour on the other side).” The rest of the parameters are self-explanatory.

The point process object along with the sound object is used to get Shimmer objects in a similar fashion. A pitch object is created from the sound object, which is in-turn used for the creation of Harmonic Feature objects. The two features: Harmonicity, auto-correlation and cross-correlation types, are extracted from these objects using similar praat calls. The harmoinicity (cc) in turn helps to extract the “Harmonics to Noise Ratio”. The same pitch object is also used to create the Formant objects, which in-turn is necessary for the “Glottal Features Using Formants” objects. This complex relationship is depicted in a flowchart form in Fig. 3.8.

Lastly, an “mel frequency cepstral coefficients” object is created from the Sound object. This object helps create the MFCC datasets for the recorded audio. The parselmouth.praat call formats are according to the praat definition.

CHAPTER 4

RESULTS

4.1 Micrographia data management module

4.1.1 Extracting data from the pen-digitiser setup

A "ScribbleDemo.cpp" applet is developed to fetch data from the hardware setup. This module fetches various data as listed in section 3.1.2 according to some pre-defined parameters and stores them locally for processing. This fetching takes place every $1.8ms$. The official WIN32 API along with the WINTAB API used for this project is open source. The technical details and code links are presented in the appendix. To normalise pen pressure value an experiment (to be discussed in section 5.1) was conducted which provided us with the data presented in Table 4.1. This allowed us to form constraints for data in later stages of the data. These constraints would be necessary to minimize data fallacies and redundancies.

Weight(g)	Raw Pressure Values
50	6430-6450
100	12900-13000
150	18420-18470
200	21350-21380
400	26520-26560
450	28680-28710
500	32767

Table 4.1: **Weight in g v/s Raw pressure UINT values PD**

To provide a visual representation of the drawing a real time handwriting capture module was developed. Using coordinate and pressure values obtained from the subject writing on the tablet, a plot is dynamically made. This plot is nothing but the visual representation of the writing on the tablet surface, as shown in Fig 4.1 . The plot uses raw real-time pressure and coordinate data from the pen. These data are then exported

into a Comma-Separated Values (CSV) File format for further calculations and analysis. “coordinate-data*date-time*.csv” and “pressure-data*date-time*.csv” are the two files generated in the root directory of the application. An additional “Report Generated *datetime*” is also generated as a text file in the root directory of the application.

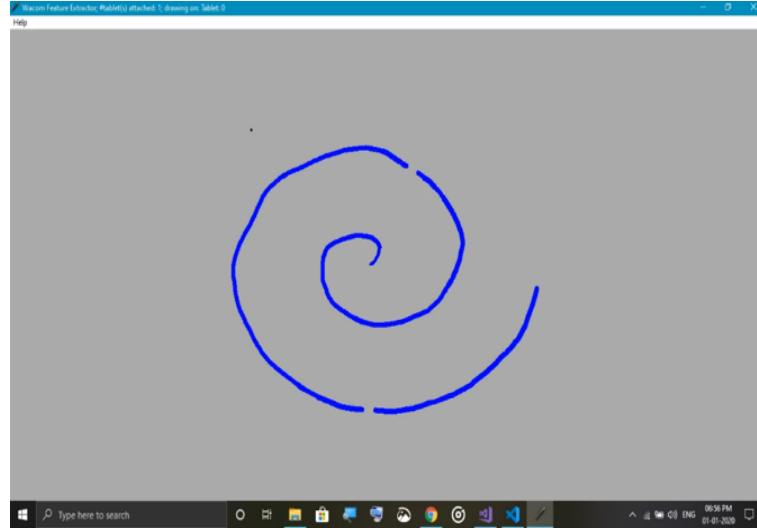


Figure 4.1: Pen Ink Visualisation through UI

4.1.2 Analysing data extracted from Micrographia recording module

A data file “Report Generated*date-time*” is also generated as a text file in the root directory of the application. This contains the analysed values of features (section 3.1.2) automatically calculated from the extracted data. The data contents of the text-file shown in Fig 4.2 , is integrated into the main UI of the module, about which is discussed in section 4.3.

Additionally to visualise the recorded data along with coordinate and pressure terms,

- A UI graph manager with inbuilt tools (saving, zoom etc.) is incorporated to pop-up after data recording.
- This is enabled using a front-end event loop python module using matplotlib back-end.
- The interface provides mouse over tips (data points).

- The two side by side plots are coordinate and data plots.
- The Coordinate plot is scattered from coordinate data while the pressure plot is 2d plotted.

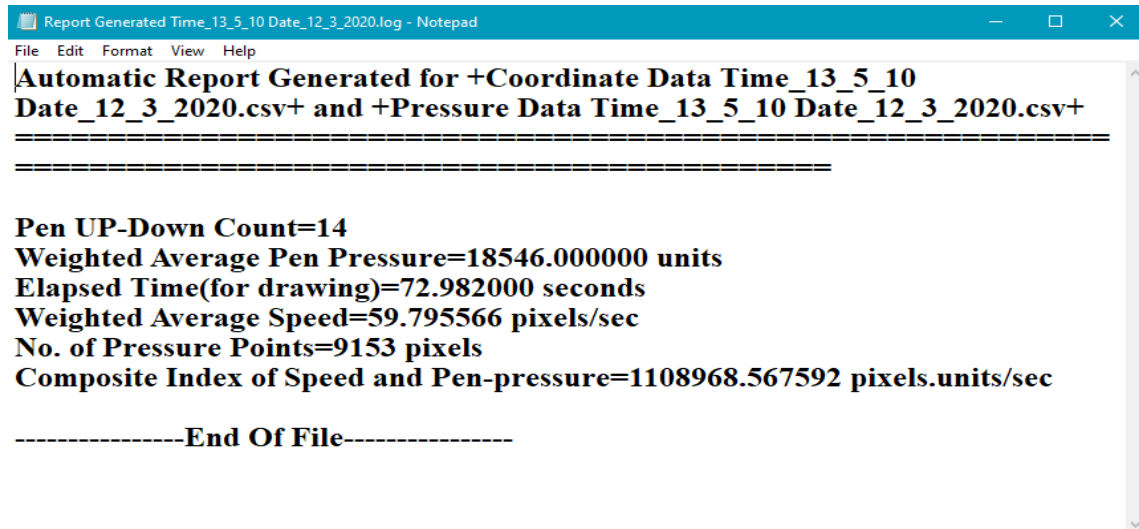


Figure 4.2: Report generated for a PWP[75,Female].

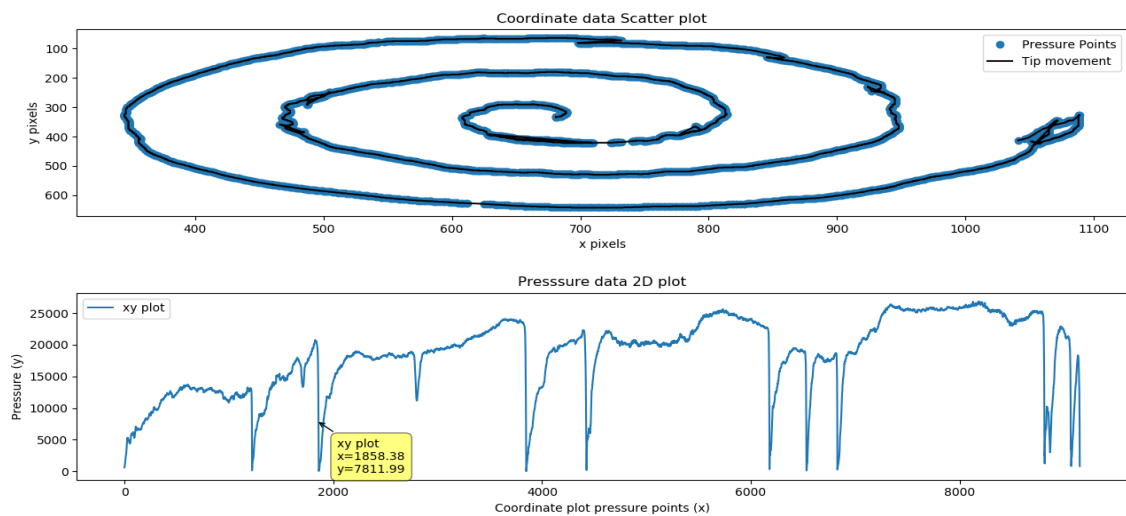


Figure 4.3: Report visualised for a PWP[75,Female].

4.2 Speech data management module

4.2.1 Extracting two-way data from a dual mic setup

The hardware chosen to record audio was an omni-directional desktop single channel condenser microphone. The recording sensitivity was kept at +14dB. The audio was to be recorded at a sampling-rate equal to 48000 Hz with a 16 bit/s resolution, with a chunk size of 1024 bytes. The recording was to be done using the “PyAudio” module and .wav file manipulation would be achieved using the “wave” module in Python 3.7. The script to extract features was decided to be written in Python 3.7 using a Python port of software PRAAT.

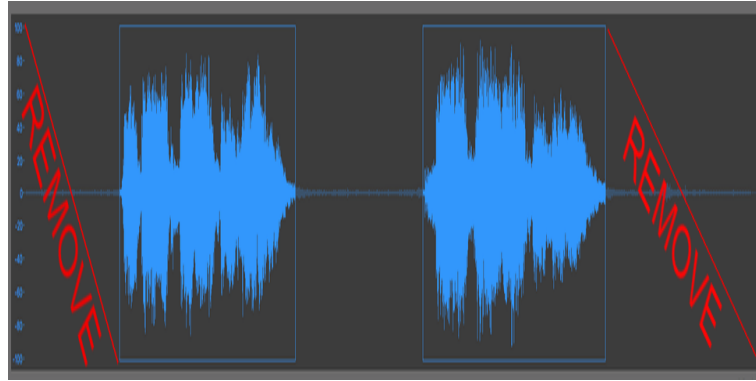


Figure 4.4: Visualisation of the "always-on" audio script logic

The script uses a ‘variable threshold’ (Fig. 4.4) procedure to remove trailing and leading silent parts of the recording to remove unnecessary data getting into the extracted features. The same procedure was used to sense vocal silence to automatically stop recording. A normalization $[(floating - point) (MAXIMUM) / maximum (|i| \text{ for } i \text{ in SOUND})]$ was applied to entire recording, where MAXIMUM is a variable obtained by hit and trial to get values with least the noise component.

4.2.2 Analysing data extracted from speech recording module

The recorded data was analysed to extract data features specified in Table 1.1. This analysis was performed, the working of which has been discussed in Section 3.2, using

a python script designed as discussed in Section 3.2.3. The recorded data being gigantic (Fig. 4.5) needed to be visualised for quick reference.

```

Pitch: Object type: Pitch
Object name: <no name>
Date: Wed Feb 19 03:24:45 2020

Time domain:
  Start time: 0 seconds
  End time: 4.5650208333333335 seconds
  Total duration: 4.5650208333333335 seconds
Time sampling:
  Number of frames: 453 (131 voiced)
  Time step: 0.01 seconds
  First frame centred at: 0.022510416666666664 seconds
  Ceiling at: 600 Hz

Estimated quantiles:
  10% = 119.000898 Hz = 107.726918 Mel = 3.01166952 semitones above 100 Hz = 3.50059264 ERB
  16% = 121.070776 Hz = 109.425982 Mel = 3.31020797 semitones above 100 Hz = 3.55254586 ERB
  50% = 127.14364 Hz = 114.380829 Mel = 4.15751153 semitones above 100 Hz = 3.70351035 ERB
  84% = 138.821251 Hz = 123.78494 Mel = 5.67874124 semitones above 100 Hz = 3.98785419 ERB
  90% = 146.042615 Hz = 129.520934 Mel = 6.55667283 semitones above 100 Hz = 4.15991811 ERB

Estimated spreading:
  84%-median = 11.72 Hz = 9.44 Mel = 1.527 semitones = 0.2854 ERB
  median-16% = 6.096 Hz = 4.974 Mel = 0.8506 semitones = 0.1515 ERB
  90%-10% = 27.15 Hz = 21.88 Mel = 3.559 semitones = 0.6619 ERB

Minimum 111.98791 Hz = 101.930958 Mel = 1.96011589 semitones above 100 Hz = 3.32264071 ERB
Maximum 481.453737 Hz = 345.843412 Mel = 27.2087661 semitones above 100 Hz = 10.0472008 ERB
Range 369.5 Hz = 243.912453 Mel = 25.25 semitones = 6.725 ERB
Average: 142.304216 Hz = 124.681317 Mel = 5.25821425 semitones above 100 Hz = 3.99065889 ERB
Standard deviation: 64.23 Hz = 43.14 Mel = 4.54 semitones = 1.197 ERB

Mean absolute slope: 271.5 Hz/s = 187.1 Mel/s = 22.07 semitones/s = 5.277 ERB/s
Mean absolute slope without octave jumps: 11.9 semitones/s

```

Figure 4.5: Excerpt of the "pitch" category of data extracted

This was achieved using a spectrogram plot which was created using ‘matplotlib’ with ‘seaborn’ styling. The lighter orange parts are the voiced samples /m/ against the backdrop of the darker unvoiced samples in Fig 4.6. A detailed data output on the pitch analysis containing related features based on F0 was programmed to be logged locally automatically. In figure 4.6, we can see how the voice peaked 5 times for 5 /m/ peaks in the plot titled "Speech Signal Representation". The hover ‘tool-tip’ feature allows queries at any instant of time from the plot.

The plot titled "Fundamental Frequency" shows the spectrogram of the speech data . The lighter orange parts are the voiced samples, made up of the individual distinct frequencies with the F0 marked out with blue-dots. The hover ‘tool-tip’ feature allows queries at any instant of time from the plot. The PRAAT algorithm which was used to extract pitch data and the fundamental frequency from the audio was being called continually in the back-end of the application with user input.

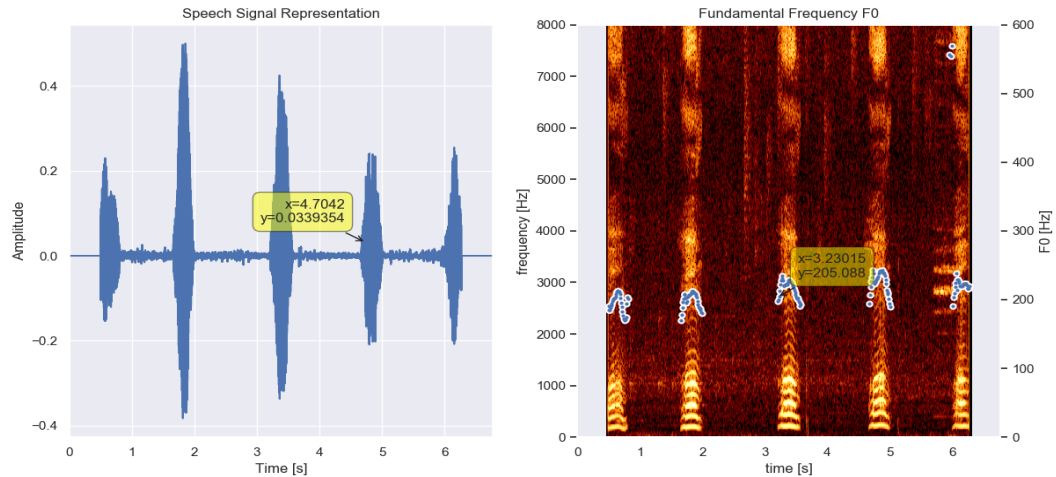


Figure 4.6: Spectrogram plot of the "pitch" category of data depicting F0 frequencies

4.3 Development of User Interface for management and integration of all Feature extraction and analysis tasks

In Information Technology, "UI Design" applies to the websites or software UI programme. It is related to optimizing the feel of items with the overall goal of encouraging user-friendliness and enhancing customer service.

- Visual Design improves an application's decorative incentive by deliberately executing components, for example, text styles, hues, and pictures in addition to other things. When expertly done, visual structure makes a page exquisite without settling on its capacity or substance.
- The Interactive Design takes a gander at how clients interface with innovation. The design at that point utilizes the comprehension of such connections to make an interface with practices that are all around considered. Amazing intelligent structure not just foresees how an individual communicates with a framework yet in addition precedes and fixes issues in great time. It might likewise design new courses through which a framework collaborates and reacts to clients.
- Data Architecture is intended to assist clients with finding the information they have to finish different errands. It, in this manner, includes naming, organizing, and arranging the web content in a way that makes it effectively available and economical.

The technical details of the front-end are provided in the appendix.

The main home screen shown in Fig 4.7. There are demarcated areas for micrographia and speech, for data collection as well as querying. The current figure shows an empty instance with "no opened projects".

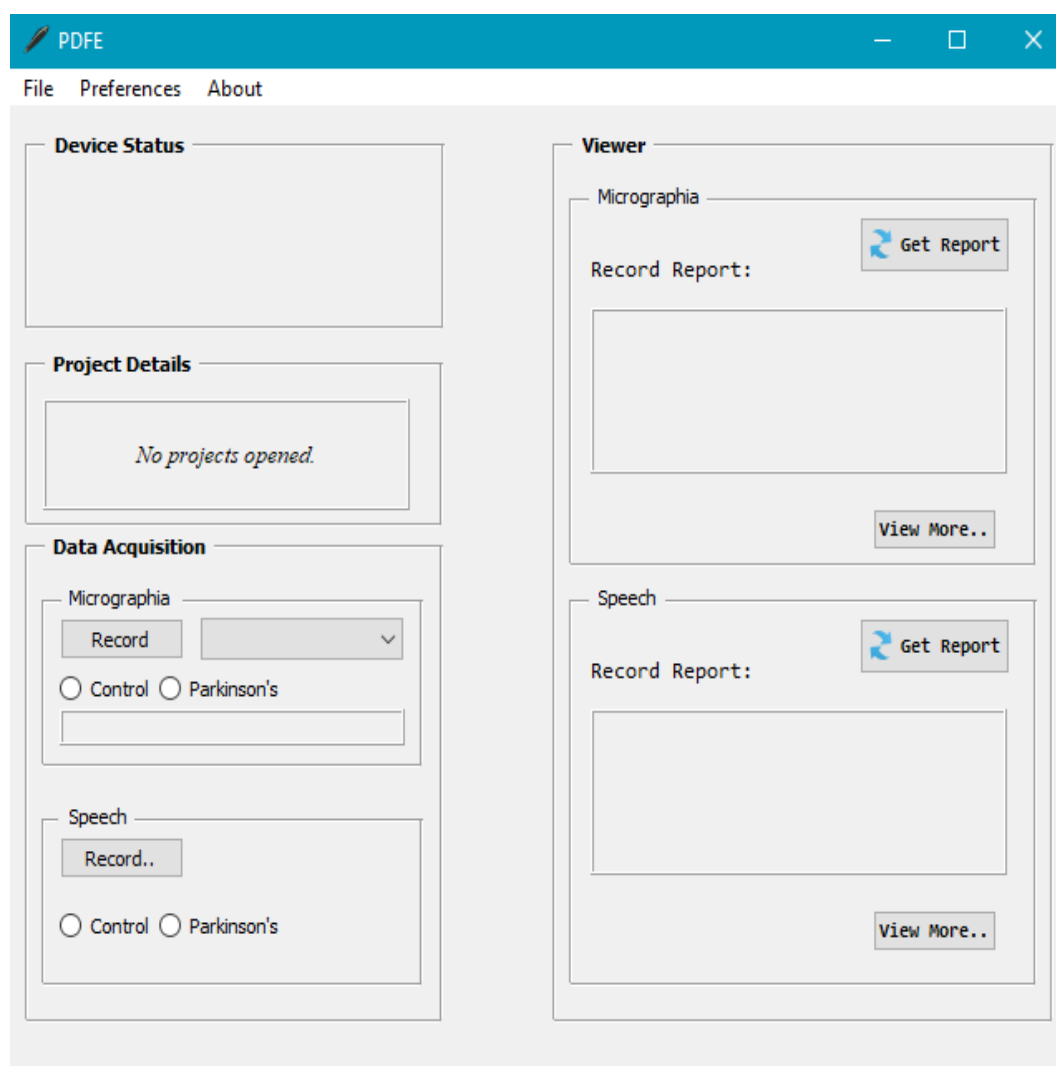


Figure 4.7: Home Screen of application module

The screen shown in Fig. 4.8 demonstrates a Micrographia data extraction process. A selection drop-down allows for selection of any of the four test types used to extract handwriting data.

- ARCH Guided Spiral: Data Recording using Fig. 1.1.
- Repeat Letters (10 times)
- Copy Sentence (if possible)
- Switching letters (switch between two letters- 10 times)

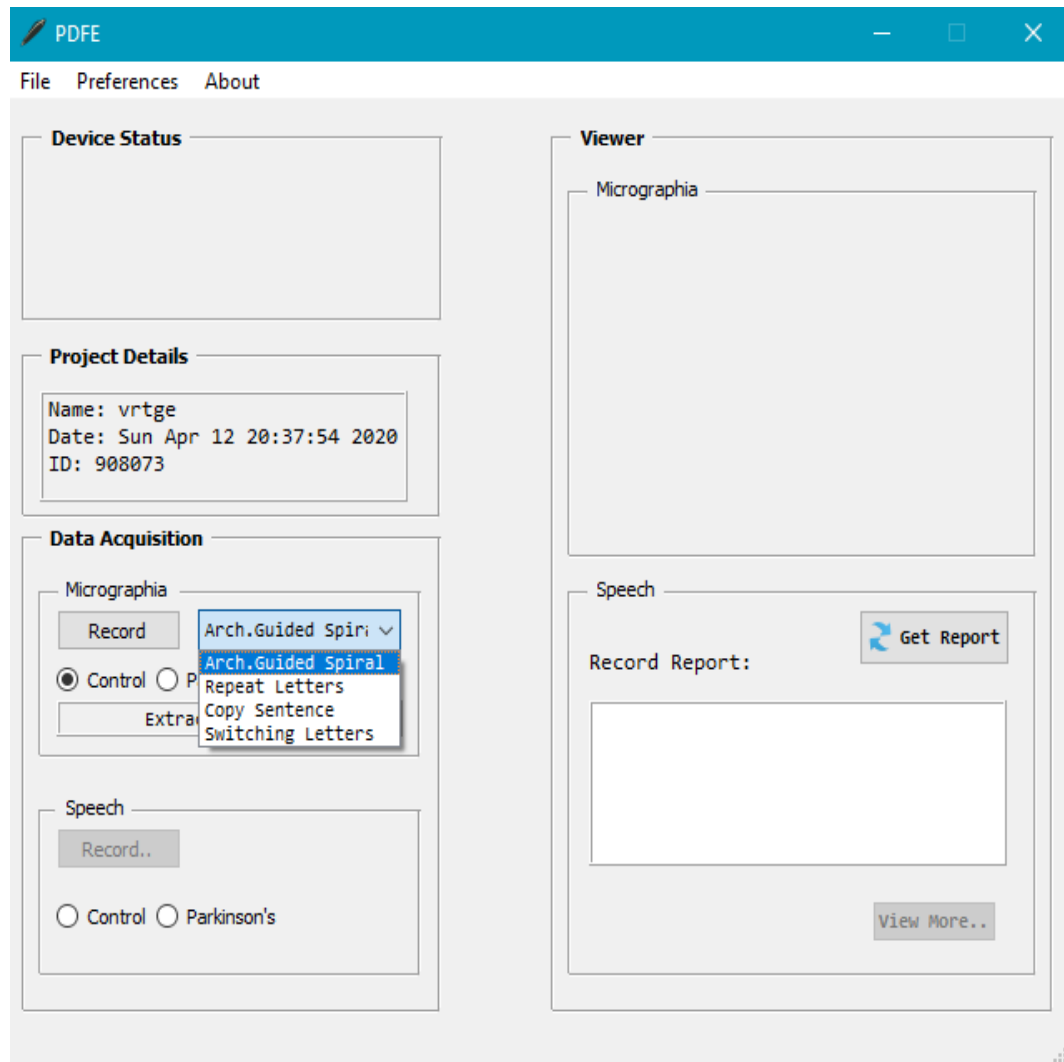


Figure 4.8: Ready to extract Micrographia Data

The screen shown in Fig. 4.9 demonstrates a speech data extraction process. The extremely informative UI changes rapidly to change in data. The "Audio Threshold Values" outputs the momentary amplitude data from speech, allowing to keep to a close watch on various factors like loudness, jitters, shimmers and so on. The threshold can be changed on the go with the "Update Threshold button" (Fig. 4.4) to adapt to the "always-on" record logic.

The internal working of the module is abstracted gracefully with seldom occurring errors(caused by query or program), to show meaningful informative dialog boxes. The recording auto-stops once error-free recording gets over to pop up with data visualisations (Fig. 4.6).

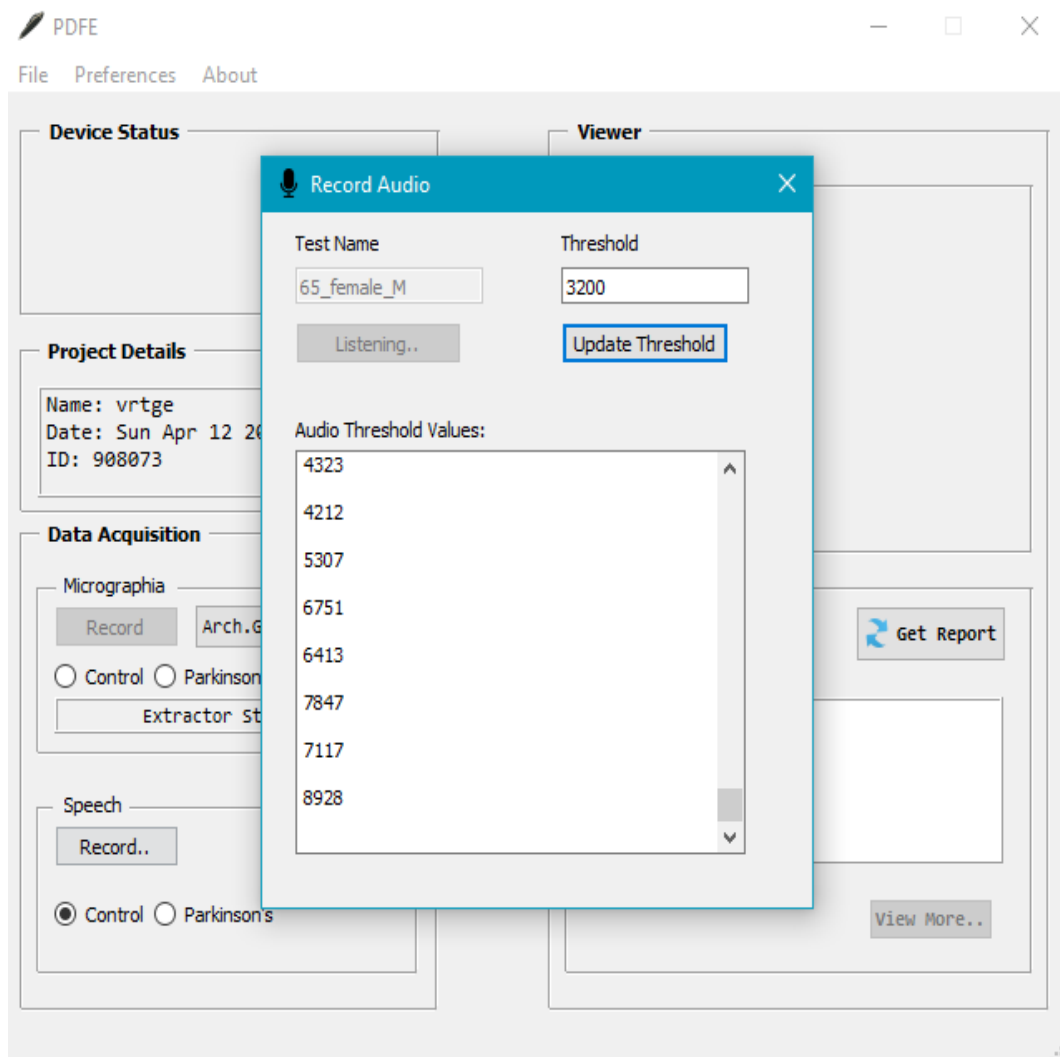


Figure 4.9: Extraction of Speech Data underway

4.4 Study of Data, extracted and machine-analysed

For the sake of this study, data from **5** PD and corresponding CG subjects were extracted and analysed by our application module. The subjects were instructed to draw spirals (Fig. 1.1) using both affected and unaffected hands. Multiple samples were recorded measuring to over 10000 KB of data. The pen-digitiser system used was an "A3 size tablet" ("Wacom Intuos Pro Large") that was commercially available. The A3 paper was put on the tablet, and the spiral was sketched using Ink pen ("Wacom Intuos ink pen"). This pen senses the touch position between the tip and the page, x and y , and the strain, Pr . The sheet's left-corner was deemed the $(0, 0)$ point.

The same group of subjects (both PWP and CG) were also made to record voice samples (/m/, /a/, /u/). The setup used was a commercially available condensor microphone and headset. Recording was done in a relatively quiet environment, with adjusted threshold.

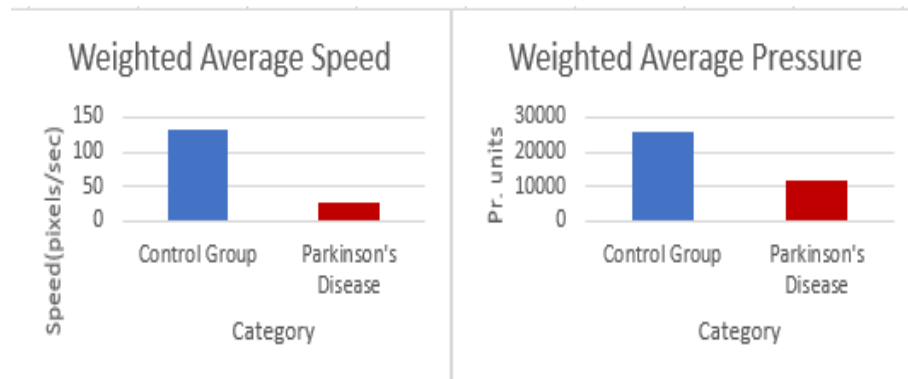


Figure 4.10: As is seen, general PWP data staggers a lot behind CG.

Although this study focuses more on extraction and management, a preliminary analysis was conducted to test real world viability of the developed module. Based on this preliminary analysis, the literature was re-inforced through solid data agreeing with theories. Fig. 4.10 shows the "Weighted Average Speed" and "Weighted Average Pressure" data collected from both PWP and CG parties. The result is that PWP data staggers a lot behind CG. A similar analysis done with CISP bore the same results, as is seen in Fig. 4.11

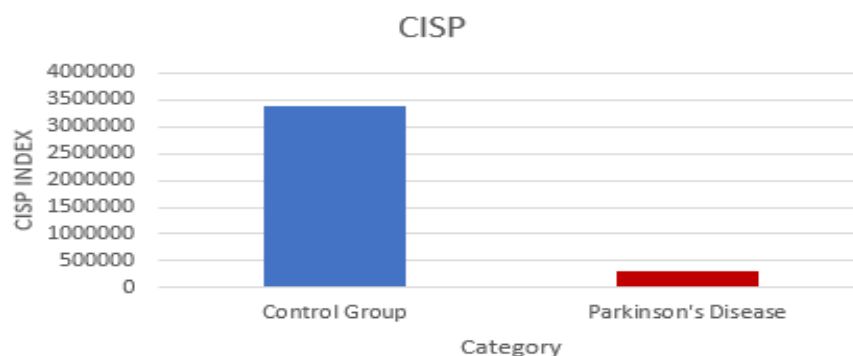


Figure 4.11: CISP Index comparing PWP and CG.

4.5 Novelty of Project

- During this development, we identified challenges related to compatibility and interoperability in collecting and processing data from pen-tablet digitiser setups. We have developed a Win32 module which is compatible with a Wacom pen tablet Operating System.
- The project can significantly help in creating discussed data-sets without hassle.
- It is an alternative to manually obtaining such data through unspecialized third party application not built for the discussed PD marker features data extraction.
- The simplistic on-point UI makes it easy for the user to operate and extract data using the application. Structured error messages for debugging is also supported.
- Easy integration with any Wacom device makes it ready to use with multiple Wacom devices (developer-docs.wacom.com, 2010).
- The end-product is a novel solution to both micrographia and speech data collection under a same roof, with added support for future development.

CHAPTER 5

TESTING

5.1 Test to deduce pen pressure

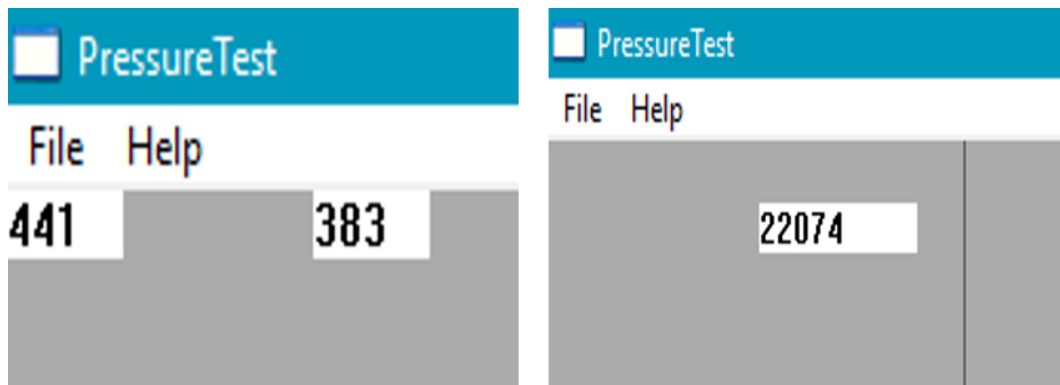


Figure 5.1: The UI interface demo showing coordinate and pressure data of patient's pen-position

The normal pressure exerted by the pen on the surface of the tablet was recorded. Through hit and trial experimentation the raw pressure values recognized by the device was found to be in the range of “0-32767” unsigned integer, as seen in Fig. 5.1. These values were calibrated against physical weights and the actual pressure values were found. The calibration table shown on the next slide was performed using a setup consisting of weights attached to the pen with it being held upright by a support. Before the calibration it was made sure that the pressure registered by the device was zero. After this test, the data populating table 4.1 was obtained.

To provide a visual representation of the drawing a real time handwriting capture module had to be developed. Using pressure values obtained by this test enabled the conversion of pressure data from a subject writing on the tablet, into digital formats for a plot to be dynamically made. This test was actually the foundation of the entire project.

5.2 Tests invloved in generating processed data from old micrographia data

```
C:\WINDOWS\py.exe
/*View recorded micrographia data*/
Pressure/Coordinate DATA

-----

1) Filename: Coordinate Data Time_0_13_8 Date_5_3_2020.csv :--
Copy this:- C:\Users\Souvik\Documents\classes\PROJECT\Wacom\Wacom_FE\Release_depc\MG_Data\CG\Adveya5_01032020_GS\Coordinate Data Time_0_13_8 Date_5_3_2020.csv

-----

2) Filename: Coordinate Data Time_17_48_34 Date_1_3_2020.csv :--
Copy this:- C:\Users\Souvik\Documents\classes\PROJECT\Wacom\Wacom_FE\Release_depc\MG_Data\CG\Skundu_01032020_CS\Coordinate Data Time_17_48_34 Date_1_3_2020.csv

-----

3) Filename: Coordinate Data Time_17_55_48 Date_1_3_2020.csv :--
```

Figure 5.2: CLI application to generate data from old recordings

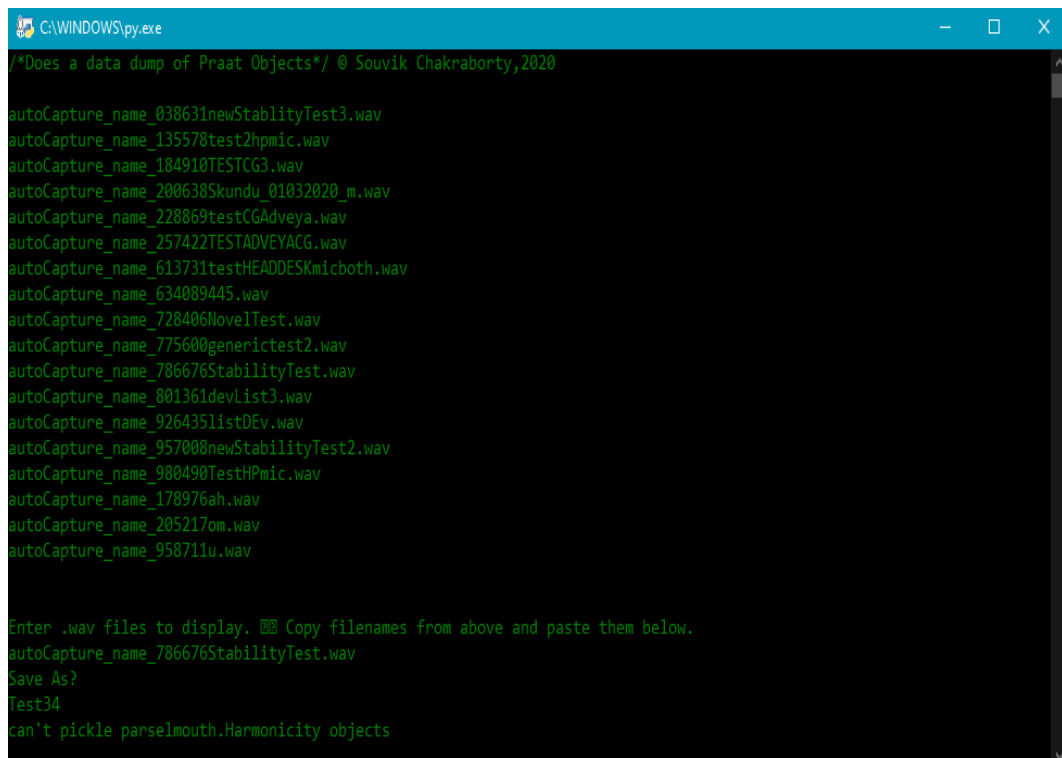
As the project extended over a long time, there arose incompatibility issues within the data generated from different versions of the application module itself. Thus a CLI solution was developed to upgrade older data formats storing micrographia data to newer formats. As seen in Fig 5.2, the idea was to create a data manager to "recognize" and list all old data formats. Then the program "tries" to open generated data, "failing" which it asks the user for consent to upgrade and generate new data, as shown in Fig. 5.3 .

```
Enter .csv files to display. Copy filenames from above and paste them below.
C:\Users\Souvik\Documents\classes\PROJECT\Wacom\Wacom_FE\Release_depc\MG_Data\PD\Patient 3\Coordinate Data Time_11_34_22 Date_29_3_2019.csv
Launching 2 .csv data files and 1 .log file, along with generated data plots in :
00:20 secs. Press 'esc' to exit and show only data plots. Press 'Enter' to open now.
No generated report found...Generate report? Y/N.
```

Figure 5.3: CLI application asking to generate report

5.3 Tests involved in analysing the internal implementation of the PRAAT software

Prior to using "parselmouth" directly in the main module, it was necessary to test its call feature. Hence a CLI based program was created to test the various praat queries (Boersma and Weenink, 1991) using the praat manual. The program automatically lists valid .wav recordings for the user to choose from, as shown in Fig. 5.4.



```
C:\WINDOWS\py.exe
/*Does a data dump of Praat Objects*/ @ Souvik Chakraborty,2020

autoCapture_name_038631newStabilityTest3.wav
autoCapture_name_135578test2hpmic.wav
autoCapture_name_184910TESTCG3.wav
autoCapture_name_200638Skundu_01032020_m.wav
autoCapture_name_228869testCGAdveya.wav
autoCapture_name_257422TESTADVEYACG.wav
autoCapture_name_613731testHEADDESKmicboth.wav
autoCapture_name_634089445.wav
autoCapture_name_728406NovelTest.wav
autoCapture_name_775600genericTest2.wav
autoCapture_name_786676StabilityTest.wav
autoCapture_name_801361devList3.wav
autoCapture_name_926435listDEv.wav
autoCapture_name_957008newStabilityTest2.wav
autoCapture_name_980490TestHPmic.wav
autoCapture_name_178976ah.wav
autoCapture_name_205217om.wav
autoCapture_name_958711u.wav

Enter .wav files to display. 📄 Copy filenames from above and paste them below.
autoCapture_name_786676StabilityTest.wav
Save As?
Test34
can't pickle parselmouth.Harmonicity objects
```

Figure 5.4: CLI application listing recorded wav files to query with

The "query commands" send you descriptions of "objects in praat". Most query commands begin with the word Get, or occasionally, the word Count. These commands can be contained in two places: in the Database menu, which normally occurs when you select an object from the chart, and in the Database menus of the author. If you click a question button, the response is written into the Data frame. In a file, these query commands can be used not only to write the details to the 'Information Page' but also to transfer it into a vector. We used this particular feature to take a deeper look at how praat is applied internally. The program asks for the number of objects that need to be passed in the query. For example, for getting a pitch type object, a sound type object

```

C:\WINDOWS\py.exe
No. of data variables required(parselmouth.data)? E.g: call([sound, pointProcess]..., 2 in this case
1
Enter these one by one in order and press return.

sound

Enter EXACT call command. E.g: "Get jitter (local)" (without quotes)
To Pitch

Enter no. of parameters. E.g: parselmouth.praat.call(snd, "To PointProcess (periodic, cc)", 75, 600)...2 in this case
3
Enter 3 param/s in order.

0
75
600
Data dumped @ /dataDump

['-----\n', 'Dump :\n', ' Object ty
pe: Pitch\n', 'Object name: untitled\n', 'Date: Wed May 6 17:01:17 2020\n', '\n', 'Time domain:\n', ' Start time: 0 s
econds\n', ' End time: 2.8462358276643993 seconds\n', ' Total duration: 2.8462358276643993 seconds\n', 'Time samplin
g:\n', ' Number of frames: 281 (3 voiced)\n', ' Time step: 0.01 seconds\n', ' First frame centred at: 0.0231179138
6219963 seconds\n', 'Ceiling at: 600 Hz\n', '\n', 'Estimated quantiles:\n', ' 10% = 473.728603 Hz = 341.708651 Mel = 2
6.9287294 semitones above 100 Hz = 9.94360868 ERB\n', ' 16% = 474.918414 Hz = 342.347508 Mel = 26.9721563 semitones ab
ove 100 Hz = 9.95963338 ERB\n', ' 50% = 481.660676 Hz = 345.953746 Mel = 27.2162057 semitones above 100 Hz = 10.049961
2 ERB\n', ' 84% = 492.114555 Hz = 351.498882 Mel = 27.5879302 semitones above 100 Hz = 10.1884298 ERB\n', ' 90% = 49
3.959357 Hz = 352.471658 Mel = 27.6527081 semitones above 100 Hz = 10.2126686 ERB\n', '\n', 'Estimated spreading:\n', '

```

Figure 5.5: Depiction of a sample querying process using the program

needs to be passed. This can be seen in Fig. 5.5. In the next step the program asks for the query command that needs to be passed. One can find the desired query command by consulting the praat manual. Then it asks for the parameter list to be sent with the query. The parameter list varies for different queries. For example, the “Get jitter (local)” query requires the ‘point process’ object. This command usually becomes available in the Query submenu in praat once the object PointProcess is selected. Normally, this command will insert the local jitter, the mean absolute variance between successive periods, divided by the mean period (a period is the time between two consecutive points), into the ‘Info window’. This command has five parameters namely: “Time start (s)”, “Time stop (s)”, “Period floor (s)”, “Period ceiling (s)” and “Maximum Period factor”. Passing these five parameters to the program is the last step in user input, after which the program queries the praat backend and dumps the queried object data into the command window. In case the user needs a closer look at the queried data, the program presents a cleaned data output to a text file as shown in Fig. 5.6. The output is properly processed to show legible data in a readable format.

This entire process actually acts as a query builder. With informative prompts at every stage, all one needs to do to take a look at the internal working of the back-end

```
C:\WINDOWS\py.exe
dataDump_for_Test34.log - Notepad
File Edit Format View Help

Dump :
Object type: Pitch
Object name: untitled
Date: Wed May 6 17:01:17 2020

Time domain:
Start time: 0 seconds
End time: 2.8462358276643993 seconds
Total duration: 2.8462358276643993 seconds

Time sampling:
Number of frames: 281 (3 voiced)
Time step: 0.01 seconds
First frame centred at: 0.02311791383219963 seconds
Ceiling at: 600 Hz

-----
pe: Pitch\n',
seconds\n', '
g:\n', ' Num
3219963 second
h.9287294 semi
ove 100 Hz = 9.5596339 kHz\n', ' 50% = 481.000070 Hz = 343.957740 Mel = 27.2102037 semitones above 100 Hz = 10.049961
2 ERB\n', ' 84% = 492.114555 Hz = 351.498882 Mel = 27.5879302 semitones above 100 Hz = 10.1884298 ERB\n', ' 90% = 49
3.959357 Hz = 352.471658 Mel = 27.6527081 semitones above 100 Hz = 10.2126686 ERB\n', '\n', 'Estimated spreading:\n', '
-----
```

Figure 5.6: Query results in full-text presented after successful query

is refer to the praat manual for queries and parameter list. This enabled us to formulate the best call algorithm in terms of both efficiency and legibility. We often noticed that in praat there were multiple ways (calls) to get a data object. Although upon further inspection made possible by this experiment, we found that there were minute and occasionally significant differences in the various ways. Through rapid query building we made adequate calls to choose the best way to get the features we want to be present in the main extraction module.

5.4 Tests involved in debugging audio I/P device selection

One of the incompatibility issues which arose with the working of the software was the issue of audio device selection for recording the speech. In the options to sort through devices, after selecting Preferences » Audio device selection, the software didn't show all the available devices as seen in Fig. 5.7.

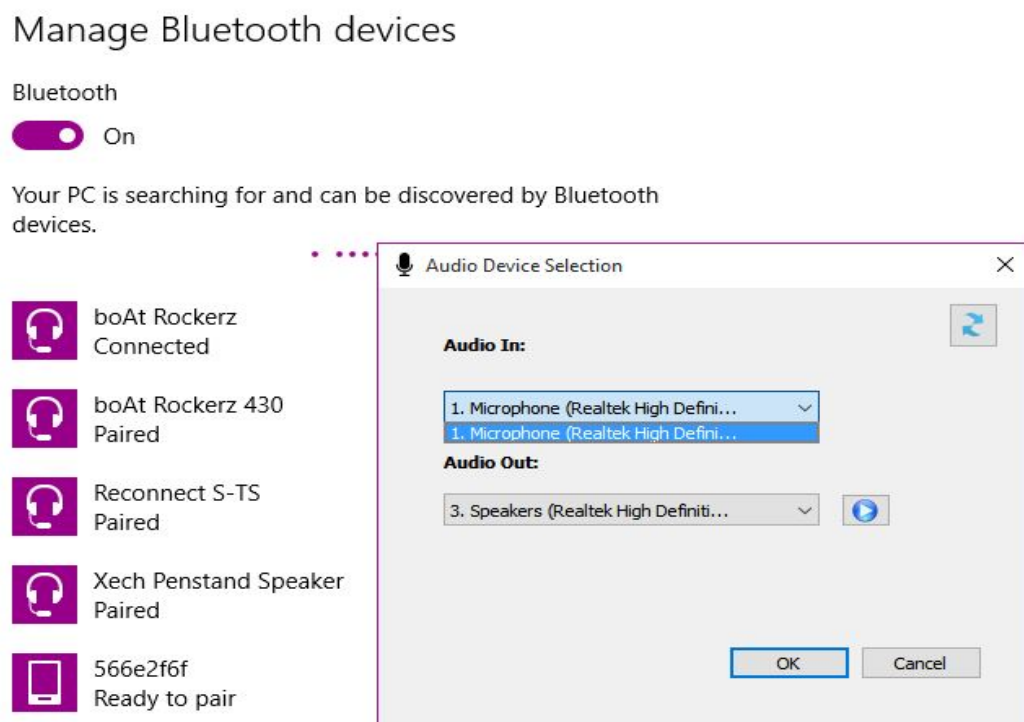


Figure 5.7: Incomplete list of options when there is an audio device connected via Bluetooth

Now recording with the in-built microphone of the PC, which the software is being used on, would fail to pick up fine details of the specific speech data which are critical to the objective of the software. Thus it was of the uttermost importance to detect and enlist the devices which had been connected to the user's PC, so that the user can select the preferred I/P device for the purpose of recording speech data either from the CG or the PWP subjects.

In this case, the software failed to enlist all the devices connected or available to the user for recording the speech data. This happened because of a technical compatibility

issue with the OS platform. For this purpose, changes were made in the program and a modified module was tested.

After the modified module had been inserted, the list of audio devices for selection by the user was updated and henceforth complete. So after this modification, the complete list was displayed correctly as seen in Fig. 5.8.

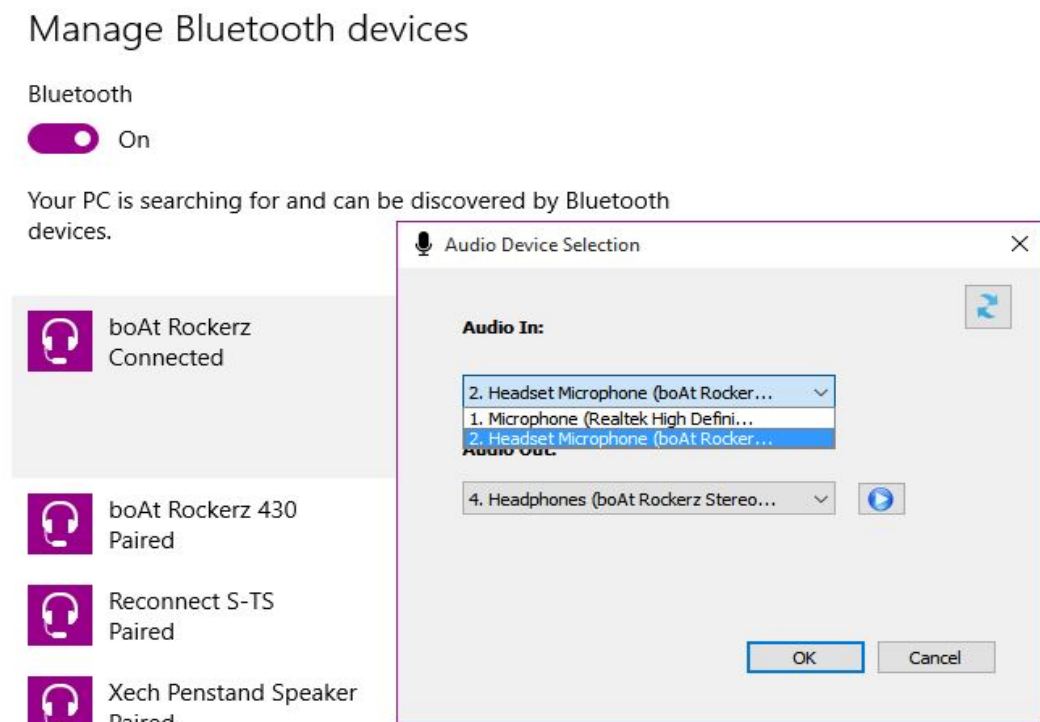


Figure 5.8: Complete list of options after using the modified module

CHAPTER 6

CONCLUSION

The extracted and "machine-analysed" data prepared by application module strongly agrees with data characteristics of said features throughout the literature study. While features like CISP, Weighted Average Speed and Weighted Average Pen-pressure extracted from the handwriting data of the "archimedean spiral sketch" are in "negative correlation with the severity of PD", others like MFCC and F0, to name a few extracted from speech data, agree with characteristics of PD obtained through previous works, both referenced and unreferenced, all giving adequate support to the fact that developed application module to extract speech and micrographia data is generating legible processed data.

APPENDIX A

CODE FOR MICROGRAPHIA DATA EXTRACTION

To view code for scribbledemo.cpp: https://github.com/souvikchakraborty98/WACOM/blob/UI/Wacom_FE/ScribbleDemo.CPP

To view code for showRecTestData.py: https://github.com/souvikchakraborty98/WACOM/blob/UI/Wacom_FE/Release/showRecTestData.py

APPENDIX B

CODE FOR SPEECH DATA EXTRACTION

To view code for Speech Data extraction and management: https://github.com/souvikchakraborty98/WACOM/tree/UI/Wacom_FE/Release/Sound
Main Speech back-end: https://github.com/souvikchakraborty98/WACOM/blob/UI/Wacom_FE/Release/Sound/speechmodule_MAIN.py

APPENDIX C

CODE FOR UI

Main front-end: https://github.com/souvikchakraborty98/WACOM/blob/UI/Wacom_FE/Release/Wafex_Ui.py
Qt Designer for designing <https://www.qt.io/>

REFERENCES

1. Boersma, P. (1993). “Accurate short-term analysis of the fundamental frequency and the harmonics-to-noise ratio of a sampled sound.” *Proceedings of the Institute of Phonetic Sciences*, 17, 97–110.
2. Boersma, P. and Weenink, D. (1991). “Praat: doing phonetics by computer. <http://www.fon.hum.uva.nl/praat/>. Accessed: 2020-04-11.
3. Bogert, B., Healy, M., and Tukey, J. (1963). “The quefrency analysis of time series for echoes: Cepstrum, pseudo-autocovariance, cross-cepstrum, and saphe cracking.” *Time Series Analysis*, 9, 209–243.
4. developer-docs.wacom.com (2010). “Windows Wintab Documentation. <https://developer-docs.wacom.com/display/DevDocs/Windows+Wintab+Documentation>. Accessed: 2020-04-11.
5. docs.microsoft.com (2005). “Windows GDI. <https://docs.microsoft.com/en-us/windows/win32/gdi/windows-gdi>. Accessed: 2020-04-11.
6. docs.microsoft.com (2015). “Get started with desktop Windows apps that use the Win32 API. <https://docs.microsoft.com/en-us/windows/win32/desktop-programming>. Accessed: 2020-04-11.
7. Jadoul, Y., Thompson, B., and de Boer, B. (2018). “Introducing Parselmouth: A Python interface to Praat.” *Journal of Phonetics*, 71, 1–15.
8. National Institute of Neurological Disorders and Stroke (2020). “Parkinson’s Disease Information Page. <https://www.ninds.nih.gov/Disorders/All-Disorders/Parkinsons-Disease-Information-Page>. Accessed: 2020-04-11.
9. NeuroData Lab (2018). “Pitch-tracking, or how to estimate the fundamental frequency in speech, on the examples of Praat, YAAPT, and YIN algorithms. <https://medium.com/@neurodatalab>. Accessed: 2020-04-11.
10. Pratheeksha Nair (2018). “The dummy’s guide to MFCC. <https://medium.com/prathena/the-dummys-guide-to-mfcc-aceab2450fd>. Accessed: 2020-04-11.
11. Ramezani, H., Khaki, H., Erzin, E., and Akan, O. B. (2017). “Speech features for telemonitoring of parkinsons disease symptoms.” *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*.
12. Teixeira, J. P., Oliveira, C., and Lopes, C. (2013). “Vocal acoustic analysis – jitter, shimmer and hnr parameters.” *Procedia Technology*, 9, 1112–1122.

13. Viswanathan, R., Arjunan, S. P., Bingham, A., Jelfs, B., Kempster, P., Raghav, S., , and Kumar, D. K. (2020). “Complexity measures of voice recordings as a discriminative tool for parkinson’s disease.” *Biosensors*, 10, 2.
14. Viswanathan, R., Khojasteh, P., Aliahmad, B., Arjunan, S., Ragnav, S., Kempster, P., Wong, K., Nagao, J., and Kumar, D. K. (2018). “Efficiency of voice features based on consonant for detection of parkinsons disease.” *2018 IEEE Life Sciences Conference (LSC)*, 1–3.
15. Zham, P., Kumar, D. K., Dabnichki, P., Arjunan, S. P., and Raghav, S. (2017). “Distinguishing different stages of parkinson’s disease using composite index of speed and pen-pressure of sketching a spiral.” *Frontiers in Neurology*, 8, 1–4.

PD

ORIGINALITY REPORT

4%

SIMILARITY INDEX

1%

INTERNET SOURCES

1%

PUBLICATIONS

2%

STUDENT PAPERS

PRIMARY SOURCES

1

Submitted to University of Limerick

Student Paper

<1%

2

João Paulo Teixeira, Carla Oliveira, Carla Lopes. "Vocal Acoustic Analysis – Jitter, Shimmer and HNR Parameters", Procedia Technology, 2013

Publication

<1%

3

fr.scribd.com

Internet Source

<1%

4

Yannick Jadoul, Bill Thompson, Bart de Boer. "Introducing Parselmouth: A Python interface to Praat", Journal of Phonetics, 2018

Publication

<1%

5

Submitted to Franklin Regional Senior High School

Student Paper

<1%

6

Poonam Zham, Dinesh K. Kumar, Peter Dabnichki, Sridhar Poosapadi Arjunan, Sanjay Raghav. "Distinguishing Different Stages of Parkinson's Disease Using Composite Index of

<1%