RANCANG BANGUN APLIKASI KLASIFIKASI GESTUR TANGAN MENGGUNAKAN METODE K-NEAREST NEIGHBORS

TUGAS AKHIR

DANIEL ADIWIRANATA SANTOSO

NIM : 311510008

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS MA CHUNG
MALANG
2022

LEMBAR PENGESAHAN

RANCANG BANGUN APLIKASI KLASIFIKASI GESTUR TANGAN MENGGUNAKAN METODE K-NEAREST NEIGHBORS

Oleh:

DANIEL ADIWIRANATA SANTOSO

311510008

dari

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS MA CHUNG

Telah dinyatakan lulus dalam melaksanakan Tugas Akhir sebagai syarat kelulusan dan berhak mendapatkan gelar Sarjana Komputer (S.Kom)

Dosen Pembimbing 1

Dr. Eng. Romy Budhi Widodo

NIP. 20070035

Dosen Pembimbing 2

Windra Swastika, Ph. D

NIP. 20070039

Dekan Fakultas Sains dan Teknologi

Kestrilia Rega Prilianti, S.Si., M.Si.

NIP. 20120035

PERNYATAAN KEASLIAN
TUGAS AKHIR

Dengan ini saya menyatakan bahwa isi sebagian maupun keseluruhan Tugas Akhir saya dengan judul "RANCANG BANGUN APLIKASI KLASIFIKASI GESTUR TANGAN MENGGUNAKAN METODE K-NEAREST NEIGHBORS" adalah benar benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diijinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka. Apabila ternyata tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Malang, Maret 2022

Daniel Adiwiranata Santoso
311510008

Rancang Bangun Aplikasi Klasifikasi Gestur Tangan Menggunakan Metode K-Nearest Neighbors

Daniel Adiwiranata Santoso
31150008

Abstrak

Pengoperasian sebuah komputer memerlukan berbagai macam alat sebagai masukan, contoh yang paling umum adalah mouse (tetikus) dan keyboard (papan tombol). Namun dalam kegunaannya dalam keseharian, ada orang dengan kondisi tertentu yang kesulitan dalam mengoperasikan komputer yang dikarenakan keterbatasannya, contohnya adalah penyandang tunarungu dan tunawicara. Penelitian ini bertujuan untuk mengklasifikasikan gestur bahasa isyarat SIBI menggunakan alat Flex Point Glove Kit.

Penelitian ini bertujuan untuk mengklasifikasi angka 1 sampai 10, serta menggunakan metode K-Nearest Neighbors, serta akan dilakukan pengujian secara real-time. Total data sampel yang digunakan pada penelitian ini sebanyak 150 buah. Pengujian pertama dilakukan dengan satu variasi data sampel yang akan digunakan sebagai data latih dan diujikan kepada 10 subyek yang berbeda, sedangkan pengujian kedua dilakukan dengan lima variasi data sampel dan diujikan kepada 5 subyek yang berbeda. Tujuan ada 2 jenis variasi yang berbeda adalah untuk membandingkan nilai k dan akurasi yang nanti akan didapatkan. Pembuatan program klasifikasi ini dilakukan menggunakan bahasa pemrograman Python dan pustaka-pustaka yang sudah tersedia contohnya Sklearn, Numpy, PySerial, Tkinter, dan lain-lain.

Dari penelitian yang sudah dilakukan, tujuan yang ingin dicapai dalam penelitian ini telah berhasil dilakukan yaitu membuat program aplikasi yang dapat mengklasifikasi gestur tangan dengan menggunakan metode KNN serta mengujinya secara real time. Hasil akurasi model menggunakan 1 variasi sampel dengan nilai k=5 menghasilkan akurasi 100%, dan menggunakan 5 variasi sampel dengan nilai k=3 menghasilkan akurasi 100%. Pengujian real-time diuji mengunakan model 1 variasi sampel mendapatkan akurasi rata-rata 80%, dan pengujian dengan 5 variasi sampel mendapatkan akurasi rata-rata 92.67%.

Kata kunci: Machine Learning, K-Nearest Neighbors (KNN), Flex Sensor, Flex Point Glove Kit, Klasifikasi gestur bahasa isyarat SIBI

Design of Hand Gesture Classification Using K-Nearest Neighbors Method

Daniel Adiwiranata Santoso
311510008

Abstract

operating computer requires many types of devices, for example the most common device is mouse and keyboard. Sometimes in daily needs, there are people with certain condition that makes them cannot operate computer normally because of their disability, for example they who suffer from deaf and mute. this research is focusing to classify SIBI (Sistem Isyarat bahasa Indonesia) sign language gesture using Flex Point Glove Kit device.

This research wants to classify SIBI sign language number 1 to 10 using K-Nearest Neighbors methods, also will do the real-times tests to few subjects. Data samples used in this research is using 150 samples. There are 2 different results in this research, the first one is only using 1 variation of samples, the second one is using 5 variations of samples. The main purpose why this research has 2 different variation is to compare the value of k, and accuracy of the model used in this research. Program that used to make this application is Python programming language, and library that already available for example Sklearn, Numpy, Tkinter, and many more.

from the research that has been done, the objectives to be achieved in this research have been successfully carried out that is made a program that can classify sign language using KNN methods and test it real-time. Accuracy when using model with 1 variation have a value of k by 5 and value of accuracy by 100%, while using model with 5 variations have a value of k by 3 and accuracy value of 100%. the real-time tests using the first model get an average accuracy by 80%, and the average accuracy with the second model is 92.67%.

Keywords: Machine Learning, K-Nearest Neighbors (KNN), Flex Sensor, Flex Point Glove Kit, SIBI sign language classification

KATA PENGANTAR

Puji syukur kepada Tuhan Yang Maha Esa karena atas rahmat dan penyertaan-Nya sehingga tugas akhir dapat berjalan dengan baik. Laporan ini berisi hasil dari tugas akhir yang telah dilaksanakan di Universitas Ma Chung, dan dikerjakan di Pusat Studi Human-Machine Interaction Research Center Gedung RND Lantai lebih satu tahun.

Dalam penyusunan tugas akhir dan laporan ini tidak terlepas dari dukungan moral dan materil yang diberikan dalam pembuatan laporan ini, penulis ingin mengucapkan terima kasih kepada:

1. Orang tua dan keluarga yang telah mendukung penulis dalam menyelesaikan tugas akhir terutama Mpek Ngah, Wak Ngah, Mama, dan Mami Lan,
2. Ibu Dr. Kestrilia Rega Prilianti, M.Si selaku Dekan dari Fakultas Sains dan Teknologi Universitas Ma Chung,
3. Bapak Hendry Setiawan, ST, M.Kom selaku Kepala Program Studi Informatika Universitas Ma Chung serta dosen penguji satu,
4. Bapak Dr. Eng. Romy Budhi Widodo selaku dosen pembimbing satu tugas akhir yang dengan sabar menghadapi saya,
5. Bapak Windra Swastika, Ph.D selaku pembimbing dua tugas akhir,
6. Teman-teman seperjuangan Marcell, Ko Andra, Vanno, Devina, Mas Ernanda, Kak Sheila, Irene, Rendy, Sovi, dan rekan-rekan peneliti di Pusat Studi HMI.

Laporan tugas akhir ini disusun berdasarkan materi yang diberikan yaitu “Rancang Bangun Aplikasi Klasifikasi Gestur Tangan Menggunakan Metode K-Nearest Neighbors”. Tugas akhir ini adalah mata kuliah yang wajib diselesaikan mahasiswa program studi Teknik Informatika Universitas Ma Chung Malang. Diharapkan dengan penulisan laporan tugas akhir ini, dapat bermanfaat bagi pembaca,

menjadi inspirasi untuk pengembangan penelitian ini, serta dapat dilakukan penelitian lanjutan yang bersifat kontinuitas.

Malang, 10 Februari 2022

Daniel Adiwiranata Santoso

DAFTAR ISI

KATA PENGANTAR ..... **i**
ABSTRAK ..... **iii**
DAFTAR ISI ..... **v**
DAFTAR TABEL ..... **viii**
DAFTAR GAMBAR ..... **ix**
Bab I **1**
Pendahuluan **1**
1.1. Latar Belakang **1**
1.2. Identifikasi Masalah **2**
1.3. Batasan Masalah **2**
1.4. Rumusan Masalah **2**
1.5. Tujuan **2**
1.6. Manfaat **3**
1.7. Luaran **3**
Bab II **4**
Tinjauan Pustaka **4**
2.1. Input Device **4**
2.1.1. Mouse 4
2.1.2. Keyboard 4
2.2. Bahasa Isyarat **5**
2.3. Flexpoint Glove Kit **6**
2.3.1. Bend Sensor (Sensor Tekuk) 8
2.3.2. Accelerometer Sensor 10
2.3.3. Magnetometer Sensor 12
2.4. K-Nearest Neighbors **12**
2.5. Confusion Matrix **13**
2.6. Regular Expression **15**
2.7. Python **15**

2.7.1. Numpy 16
2.7.2. Matplotlib 16
2.7.3. Scikit Learn 17
2.7.4. Tkinter 18
2.8. Penelitian Terdahulu 18
Bab III 20
Analisis dan Perancangan Sistem 20
3.1. Tahapan Penelitian 20
3.2. Analisis Kebutuhan 21
3.3. Desain Sistem 22
3.3.1. Gerakan Gestur Tangan 24
3.3.2. Pembacaan Data Oleh Sensor 25
3.3.3. USB Glove Kit 26
3.3.4. Pembelajaran Data Menggunakan K-Nearest Neighbors 27
3.4. Implementasi Sistem 28
3.5. Rancangan Pengujian 29
Bab IV 31
Hasil dan Pembahasan 31
4.1. Hasil Pengumpulan Data 31
4.1.1. Penggunaan Flexpoint Glove Kit 31
4.1.2. Proses Pengambilan data 32
4.1.3. Proses Pengolahan Data 34
4.1.4. Proses *Split* Data Latih dan Data Uji Serta Transformasi Data 35
4.2. Hasil Rancangan Aplikasi 36
4.2.1. Antarmuka Tampilan Aplikasi 36
4.3. Hasil Pengujian Sistem 37
4.3.1. Hasil Pengujian Data uji 37
4.3.1.1. Hasil Pengujian Data uji Dengan 1 Variasi Data Latih 38
4.3.1.2. Hasil Pengujian Data uji Dengan 5 Variasi Data Latih 42
4.4. Diskusi Hasil 45
Bab IV 47

Penutup 47
5.1. Kesimpulan 47
5.2. Saran 48
Daftar Pustaka 49
Lampiran 51

DAFTAR TABEL

Tabel 2.1 Confusion Matrix 14
Tabel 3.1 Spesifikasi Komputer 23
Tabel 4.1 Contoh file hasil perekaman data angka 1 33
Tabel 4.2 Isi file targets.csv 34
Tabel 4.3 Isi file groups.csv 35
Tabel 4.4 Tabel isi data es2_train.csv 36
Tabel 4.5 Hasil pengujian KNN dengan parameter k (1 variasi) 38
Tabel 4.6 Rangkuman hasil pengujian seluruh subyek (1 variasi) 40
Tabel 4.7 Hasil pengujian tiap gestur (1 variasi) 40
Tabel 4.8 Confusion matrix pengujial real time (1 variasi) 41
Tabel 4.9 Hasil Pengujian KNN dengan parameter k (5 variasi) 42
Tabel 4.10 Rangkuman hasil pengujian seluruh subyek (5 variasi) 43
Tabel 4.11 Hasil pengujian tiap gestur (5 variasi) 43
Tabel 4.12 Confusion matrix pengujial real time (5 variasi) 44

DAFTAR GAMBAR

Gambar 2.1 Huruf / abjad BISINDO 5
Gambar 2.2 Huruf dan angka SIBI 6
Gambar 2.3 USB glove kit 7
Gambar 2.4 Visualisasi bend sensor 8
Gambar 2.5 Bend sensor 9
Gambar 2.6 Sendi telapak tangan 10
Gambar 2.7 Orientasi akselerometer 10
Gambar 2.8 Orientasi akselerometer KMX62 12
Gambar 2.9 Contoh visualisasi Matplotlib 17
Gambar 2.10 Contoh tampilan Tkinter 18
Gambar 2.11 Pengambilan data EMG 19
Gambar 3.1 Tahapan perancangan sistem 20
Gambar 3.2 Ilustrasi proses klasifikasi 22
Gambar 3.3 Visualisasi pembacaan data flexpoint glove 24
Gambar 3.4 Hasil pembacaan sensor 25
Gambar 3.5 Hasil pembacaan sensor dalam file berformat.csv 25
Gambar 3.6 Gambar dan skema m*icrocontroller* 27
Gambar 3.7 Diagram alur KNN 28
Gambar 3.8 *Mockup* penggunaan perangkat 29
Gambar 4.1 Alat yang digunakan pada penelitian ini 32
Gambar 4.2 Tampilan proses pengambilan data 33
Gambar 4.3 Grafik persebaran data gestur angka 1 sampel pertama 34
Gambar 4.4 Antarmuka tampilan program dan *command line* pembacaan sensor 37
Gambar 4.5 Hasil latih akurasi model pada KNN dengan nilai k yang berbeda 38

Bab I
Pendahuluan

1.1. Latar Belakang

Untuk mengoperasikan sebuah komputer, dibutuhkan berbagai macam alat sebagai masukan untuk komputer, contohnya adalah tetikus atau mouse, kemudian papan tombol atau *keyboard.* Papan tombol adalah suatu alat yang digunakan untuk mengetik atau memasukkan sebuah fungsi untuk mengoperasikan komputer. Pada umumnya saat ini *keyboard* ada dua bentuk, yaitu yang pertama *keyboard* fisik yang langsung disentuh/ditekan, dan yang kedua adalah *keyboard on-screen* atau *keyboard* yang berada di dalam layar yang dioperasikan dengan cara menggunakan *mouse.*

Ada suatu kondisi dimana manusia tidak dapat menggunakan beberapa alat untuk mengoperasikan komputer, misalnya tidak dapat mengoperasikan *keyboard* atau *mouse*. Pada penelitian sebelumnya telah dibuat sebuah alat untuk membantu manusia dengan kondisi yang telah disebutkan, contohnya sebuah *pointing device* pengganti *mouse* yang menggunakan sensor akselerometer sebagai penggerak *pointer* nya (Giovanno, 2020), ada juga penelitian yang mengembangkan alat untuk mendeteksi klik kanan atau klik kiri dengan mengkontraksikan otot yang dideteksi oleh sensor EMG (*Electromyograph)* sehingga dapat digunakan sebagai pengganti fungsi klik pada *mouse* (Trixie, 2020). Terdapat juga penelitian sebelumnya yang menggunakan sensor tekuk (*bend sensor*) untuk mendeteksi bahasa isyarat berupa angka “1”, “2”, dan “3” dengan metode jaringan saraf tiruan. Pada penelitian tersebut diperoleh akurasi diatas 90% dan Tujuan terbesar dari penelitian tersebut adalah studi lanjut dengan menambahkan klasifikasi gestur untuk angka, huruf, dan kata-kata (Widodo, dkk, 2020).

Oleh karena itu pada penelitian ini akan dilakukan pengembangan berupa klasifikasi gestur untuk angka “1” hingga “10”. Gestur yang digunakan dalam penelitian ini merupakan bahasa isyarat yang ada di Indonesia, yaitu Sistem Isyarat Bahasa Indonesia (SIBI). Penelitian ini juga akan menggunakan metode yang berbeda

dari penelitian tersebut yaitu K-Nearest Neighbors. Selain itu penelitian ini juga akan melakukan pengujian secara *real-time.*

1.2. Identifikasi Masalah

Berdasarkan latar belakang yang telah dipaparkan, maka identifikasi masalahnya adalah belum adanya program/aplikasi yang dapat mengklasifikasikan semua angka SIBI dan melakukan pengujian secara *real-time*.

1.3. Batasan Masalah

Batasan masalah dalam tugas akhir ini adalah sebagai berikut:

a. gestur dilakukan dengan menggerakkan telapak dan jari tangan kanan,
b. sensor yang digunakan adalah *Bend Sensor* atau sensor tekukan dari Flex Point yang dipasangkan pada sarung tangan,
c. data yang digunakan sebagai masukan adalah hasil baca dari *bend sensor* sebanyak 10 masukan.

1.4. Rumusan Masalah

Berdasarkan identifikasi masalah di atas, maka dapat dibuat rumusan masalah yaitu bagaimana membuat sebuah aplikasi yang dapat mengidentifikasi gestur gerakan tangan dan membuat keluaran berupa angka 1 sampai 10, serta melakukan pengujian terhadap subyek yang berbeda-beda secara *real-time* terhadap setiap gestur yang sudah dipelajari.

1.5. Tujuan

Tujuan yang ingin dicapai dalam tugas akhir ini adalah membuat sebuah aplikasi yang dapat mengidentifikasi gerakan gestur tangan dengan menggunakan metode K-Nearest Neighbors berbasis *desktop* agar gestur tangan dapat diklasifikasikan kedalam Bahasa Isyarat SIBI.

1.6. Manfaat

Manfaat yang didapatkan dari tugas akhir ini adalah sebagai berikut:

a. Bagi Universitas Ma Chung, khususnya program studi teknik informatika dapat menghasilkan sebuah aplikasi yang dapat digunakan untuk media pembelajaran dan mempersiapkan lulusan yang berkompeten dan siap kerja.
b. Bagi peneliti, mampu menerapkan ilmu yang selama ini telah didapatkan dari Universitas Ma Chung, serta meningkatkan kemampuan individu dalam membuat sebuah program.
c. Bagi pembaca, dapat menambah informasi dan menjadi inspirasi untuk melakukan pengembangan terhadap aplikasi yang telah dibuat.

1.7. Luaran

Luaran yang dihasilkan diharapkan dapat menghasilkan sebuah program/aplikasi yang dapat mengidentifikasi gestur gerakan angka dalam SIBI, mengujinya langsung secara *real-time*, serta dapat menjadi inspirasi terhadap penelitian selanjutnya yang memiliki bahasan yang sama.

Bab II
Tinjauan Pustaka

2.1 Input Device

Secara umum *Input device* adalah perangkat masukan pada personal komputer yang berfungsi untuk memasukan perintah dari pengguna komputer baik berupa perintah teks, gambar, maupun suara yang nantinya akan di oleh kembali oleh perangkat processing untuk di tampilkan oleh perangkat penampil, atau output device (Riyadi, 2009). Contoh umum perangkat masukan adalah *mouse* (tetikus), dan *keyboard* (papan tombol), contoh yang lain adalah *joystick, webcam, scanner, graphic tablet*, serta alat-alat lain yang memberikan data masukan kepada komputer. Dari semua contoh tadi, yang paling sering dan penting untuk digunakan adalah *mouse* dan *keyboard*.

2.1.1 Mouse

Mouse atau tetikus dalam Bahasa Indonesia merupakan suatu perangkat penunjuk yang biasa digunakan pada komputer atau laptop. Terdapat dua macam tetikus, yaitu tetikus mekanik dan tetikus optikal. Tetikus mekanik memiliki sebuah bola di dalamnya yang menyentuh permukaan dan berputar bersama dengan geraknya tetikus. Sedangkan tetikus optikal menggunakan tembakan laser dan chip khusus untuk menghasilkan data yang sesuai untuk pergerakan mouse dan pointer pada komputer.

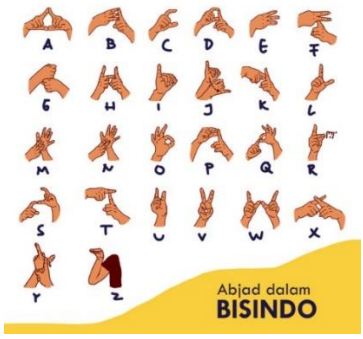
2.1.2 Keyboard

Keyboard atau papan tombol adalah perangkat masukan pada komputer yang memiliki banyak tombol dan berfungsi mengirimkan perintah pada CPU berupa huruf, angka, simbol, dan fungsi-fungsi tertentu pada komputer. *Keyboard* dapat dibedakan dari bentuk fisiknya, dan jenis tombolnya. Berdasarkan bentuk fisiknya, ada beberapa jenis yaitu *keyboard* fisik yang umum digunakan, *on-screen keyboard*, *virtual keyboard*, dan lain sebagainya. Berdasarkan jenis tombolnya ada *keyboard* QWERTY, *keyboard* DVORAK, *keyboard* KLOCKENBERG, dan lain sebagainya.

2.2 Bahasa Isyarat

Bahasa isyarat merupakan bentuk komunikasi non-verbal yang dilakukan dengan cara menggerakkan bibir, dan bahasa tubuh, termasuk ekspresi wajah, pandangan mata, gerakan tangan, serta gerakan tubuh (Peduli Kasih ABK, 2018). Bahasa isyarat yang diperkenalkan sebagai bahasa formal biasanya digunakan oleh teman tuli. Hal ini disebabkan oleh orang dengan gangguan pendengaran biasanya akan memiliki gangguan pada kemampuan bicaranya pula sehingga komunikasi efektif yang dapat dilakukan adalah menggunakan gerak tubuh. Gerakan-gerakan ini kemudian disepakati maknanya sehingga secara alami lahirlah bahasa isyarat yang digunakan secara luas. Untuk penggunan bahasa isyarat ini, walaupun menggunakan gerakan tubuh, tetap menggunakan gerak bibir dan ekspresi juga agar lawan bicara lebih memahami konteks pesan yang ingin disampaikan. Bahasa isyarat ini muncul secara alami dan disesuaikan dengan budayanya masing-masing hingga saat ini belum ada bahasa isyarat terstandar internasional. Oleh karena itu, setiap negara memiliki bahasa isyaratnya masing-masing, termasuk Indonesia.

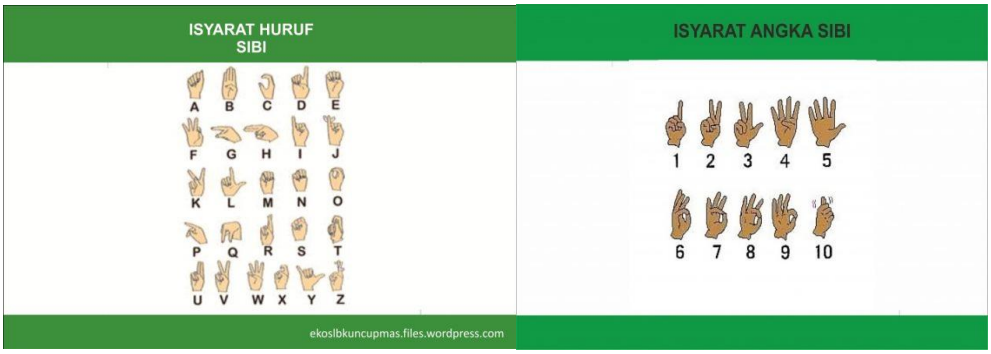
Di Indonesia terdapat dua bahasa isyarat yang digunakan, yaitu Bahasa Isyarat Indonesia (BISINDO), dan Sistem Isyarat Bahasa Indonesia (SIBI). BISINDO merupakan bahasa isyarat yang muncul secara alami dalam budaya di Indonesia, dan praktis untuk digunakan dalam kehidupan sehari-hari, sehingga BISINDO memiliki beberapa variasi di tiap daerah, beberapa isyarat dalam BISINDO dilakukan dengan dua tangan. BISINDO lebih sering digunakan oleh penderita tuna rungu dalam komunikasinya sehari-hari dengan orang lain,



Gambar 2.1 huruf/abjad BISINDO

(sumber: https://www.ypedulikasihabk.org/storage/2018/11/infografis-bhs-isyarat-4.jpg)

SIBI merupakan sistem bahasa isyarat yang diambil/diserap dari American Sign Language (ASL) atau Bahasa Isyarat Amerika, yang memiliki awalan dan akhiran. SIBI merupakan bahasa isyarat yang diresmikan oleh pemerintah dan digunakan dalam proses pembelajaran di Sekolah Luar Biasa (SLB). SIBI memiliki struktur yang sama dengan tata bahasa lisan Indonesia (Adi, 2018).

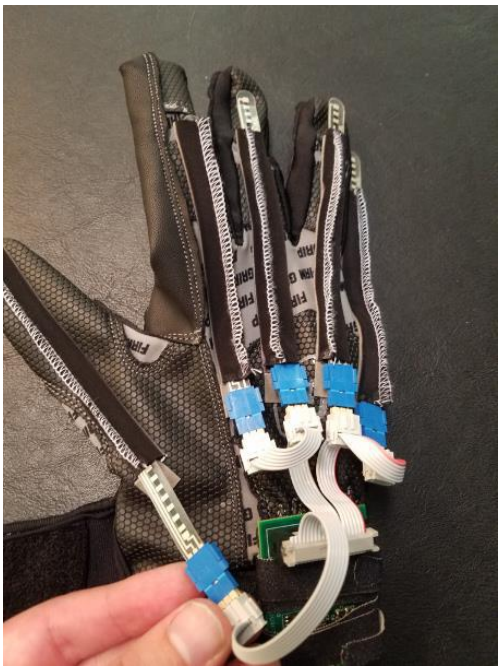


Gambar 2.2 huruf dan angka SIBI

(sumber: https://meenta.net/belajar-bahasa-isyarat-dasar/)

2.3 Flexpoint Glove Kit

Flexpoint Glove adalah sebuah produk milik Flexpoint Sensor System Incorporated. Flexpoint sensor Systems (FLXT) adalah pemasok terkemuka teknologi sensor panca indra untuk banyak industri, termasuk otomotif, medis, kontrol industri, dan produk konsumen. Salah satu produknya adalah USB Glove Kit. Flexpoint Glove Kit sebenarnya tidak dipaketkan bersama dengan sarung tangan, hal ini dikarenakan semuanya terserah pembeli atau pengguna mau membuat atau menentukan bagaimana sensor ini akan dibentuk.

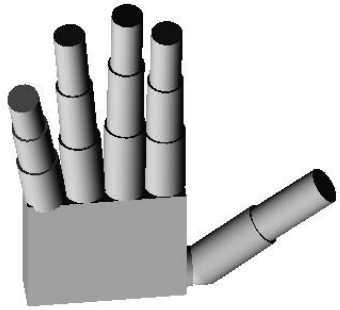


Gambar 2.3 USB Glove Kit

(sumber: https://www.flexpoint.com/usbglovekit)

Flexpoint Glove Kit ini terdiri dari lima buah sensor tekukan atau bend sensor yang berisi tiga buah sensor panjang yang akan dipasangkan pada jari telunjuk, jari tengah, dan jari manis, serta dua buah sensor yang lebih pendek yang akan dipasangkan pada ibu jari, dan jari kelingking. Fungsi dari bend sensor ini adalah mendeteksi apakah jari-jari pada tangan tegak lurus atau tertekuk, sebuah mikrokontroler yang sudah dipasang Kionix KMX62 yang merupakan sebuah sensor yang berisi sensor akselerometer, dan sensor magnetometer.

Selain Glove Kit, juga diberikan *software* (perangkat lunak) untuk menjalankan programnya, *software* tersebut adalah program C# yang membaca data serial port dari virtual COM port dan ditampilkan menggunakan OpenGL untuk menunjukkan pergerakan sensor secara 3D virtual dalam layar. Selain menunjukkan virtualisasi pergerakan sensornya, kita juga dapat melihat data-data numerik yang terbaca pada sensor dengan membuka sebuah emulator serial. Bentuk data-data yang dikeluarkan adalah dua buah tekukan pada masing-masing jari, tiga buah sumbu dari akselerometer dan magnetometer.



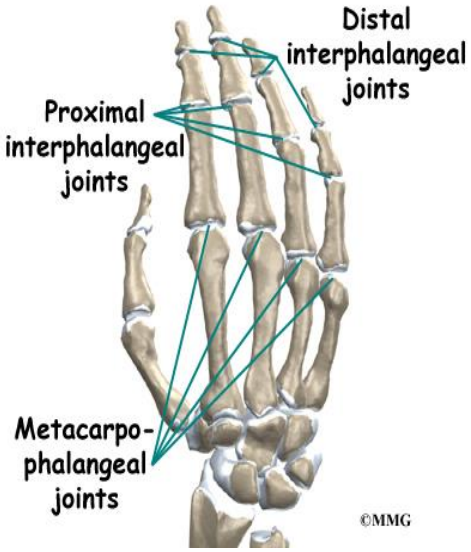
Gambar 2.4 Visualisasi Bend Sensor

2.3.1 Bend Sensor (Sensor Tekuk)

Sensor tekukan adalah sensor yang mengukur jumlah defleksi atau tekukan. Biasanya, sensor menempel atau diletakkan pada permukaan sebuah barang. Terdapat total lima buah sensor pada USB Glove Kit, tiga buah sensor tekuk yang pertama adalah sensor yang lebih panjang yang akan dipasangkan pada jari telunjuk, jari tengah, dan jari manis, serta dua buah sensor yang lebih pendek yang dipasangkan pada jari kelingking, dan ibu jari.

Gambar 2.5 Bend Sensor
(Sumber: https://www.flexpoint.com/usbglovekit)

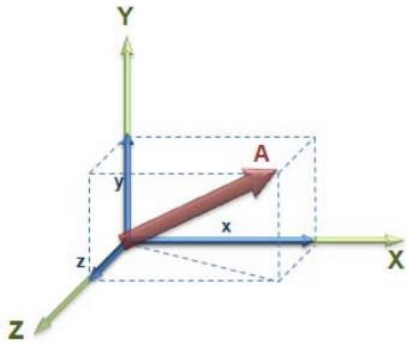
Bend sensor ini dipasangkan pada tangan dengan menggunakan sarung tangan yang dibuat khusus untuk meletakkan sensor-sensor ini seperti yang ditunjukkan pada gambar 2.3. Sedangkan bentuk masing-masing bend sensor mirip dengan penggaris yang lentur seperti yang diperlihatkan pada gambar 2.5. Bend Sensor milik Flexpoint ini dapat membaca dua buah tekukan pada tiap-tiap jari pada tangan seperti pada ditunjukkan pada gambar 2.6, yaitu *metacarpo-phalangeal* atau tekukan terdekat dengan telapak tangan, serta *proximal-interphalangeal* atau tekukan terjauh dari telapak tangan.



Gambar 2.6 Sendi Telapak Tangan
(sumber: https://eorthopod.com/sites/default/files/images/finger_PIPinjury_anatomy01.jpg)

2.3.2 Accelerometer Sensor (Sensor Akselerometer)

Akselerometer adalah sebuah sensor yang digunakan untuk mengukur percepatan sebuah objek. Percepatan yang dapat diukur oleh sensor akselerometer adalah percepatan dinamis dan percepatan statis. Percepatan dinamis adalah pengukuran percepatan pada objek bergerak, sedangkan pengukuran percepatan statis adalah pengukuran percepatan objek terhadap gravitasi bumi (Immersa Lab, 2018).



Gambar 2.7 Orientasi akselerometer

(sumber:
http://repository.usu.ac.id/bitstream/handle/123456789/47742/Chapter;jsessionid=B8D58ADE153967B7BD16C271C5000AB4?sequence=3)

Prinsip kerja akselerometer adalah prinsip percepatan (*acceleration*). Contohnya adalah sebuah per dengan beban dilepaskan, beban bergerak dengan suatu percepatan sampai kondisi tertentu lalu berhenti. Bila ada sesuatu yang menggoncangkan atau menggerakannya maka beban akan berayun kembali. Pengukuran kapasintansi inilah yang umumnya menjadi hasil pengukuran *chip* agar sensor bisa mendeteksi tiga dimensi, maka dibutuhkan tiga pasang plat yang dipasang tegak lurus antar masing-masing. Sensor akselerometer umumnya memiliki tiga sumbu yang disimbolkan dengan X, Y, dan Z. ketiga sumbu ini dapat dilihat pada gambar 2.7, sumbu A merupakan sumbupergerakan yang dideteksi oleh akselerometer. Untuk menghitung pergerakkannya dapat menggunakan persamaan sebagai berikut,

$$A^2 = x^2 + y^2 + z^2 \quad (2\text{-}1)$$

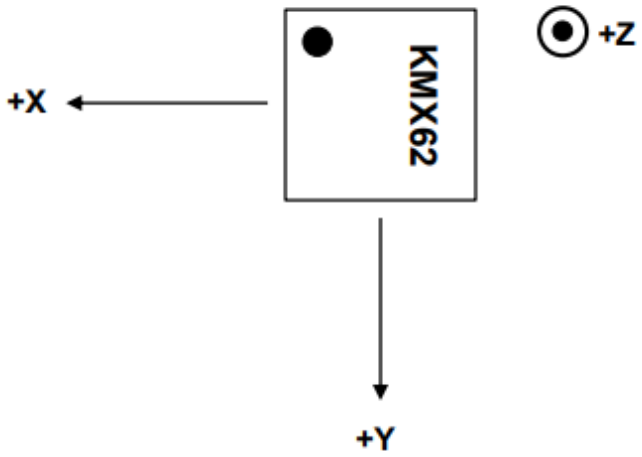
Keterangan:

x = vektor X / nilai sumbu x

y = vektor Y / nilai sumbu y

z = vektor Z / nilai sumbu z

Sensor akselerometer yang digunakan oleh USB Flexpoint adalah Kionix KMX62. Sensor Kionix KMX62 adalah sebuah sensor yang terdiri dari gabungan sensor akselerometer tri-axial (tiga arah), dan sensor magnetometer tri-axial. Orientasi akselerometer pada Kionix KMX62 dapat dilihat pada gambar 2.8, ketika perangkat digerakkan kearah +X, +Y, atau +Z nilai keluaran dari akselerometer akan bertambah, bila perangkat digerakkan kearah –X, -Y, dan –Z nilai keluarannya akan berkurang.



Gambar 2.8 Orientasi Akselerometer KMX62

(sumber: https://www.mouser.com/pdfdocs/Kionix-KMX62-1031-Datasheet.pdf)

2.3.3 Magnetometer Sensor (Sensor Magnetometer)

Magnetometer adalah instrumen atau sensor yang digunakan untuk mengukur kekuatan dan juga arah medan magnet. Instrumen ini pertama kali diperkenalkan oleh Carl Friedrich Gauss pada tahun 1833 untuk pengukuran medan magnet bumi. Satuan internasional medan magnet adalah Tesla. Untuk pengukuran geomagnet digunakan satuan nanotesla (nT). Satuan lain yang digunakan adalah Gauss, dimana 1 Gauss = 100.000 nT, atau 1 Gauss = 100.000 gamma.

Magnetometer dibagi menjadi dua tipe. Tipe pertama adalah magnetometer skalar, yaitu magnetometer yang hanya mengukur total kekuatan medan magnet. Tipe kedua adalah magnetometer vektor, yaitu magnetometer yang mengukur besar dan arah medan magnet dalam 3 koordinat, yaitu komponen XYZ atau HDZ (Buletin Komrad).

2.4 K-Nearest Neighbors

K-Nearest Neighbors atau biasa disebut K-NN adalah algoritma yang berfungsi untuk melakukan klasifikasi suatu data berdasarkan data pembelajaran (train data sets), yang diambil dari k tetangga terdekatnya (nearest neighbors) dengan k merupakan banyaknya tetangga terdekat. K-NN termasuk dalam supervised learning, dimana hasil

query instance yang baru diklasifikasikan berdasarkan mayoritas kedekatan jarak k-tetangga terdekat dari kategori yang ada dalam K-NN. Supervised learning adalah sebuah pendekatan yang sudah terdapat data yang dilatih dan terdapat variabel yang ditargetkan sehingga tujuan dari pendekatan ini adalah mengelompokkan suatu data ke data yang sudah ada. Untuk menggunakan algoritma K-NN perlu ditentukan banyaknya k tetangga terdekat yang digunakan untuk melakukan klasifikasi data baru. Banyaknya k merupakan angka ganjil, misalnya k = 3, 5, 7, dan seterusnya. Secara sederhana, K-NN bekerja bedasarkan jarak minimun dari data baru ke data latih utuk menentukan K-tetangga terdekat. Langkah-langkah dari algoritma K-NN, yaitu

1. Menentukan parameter K sebagai banyaknya jumlah tetangga terdekat dengan objek baru.
2. Menghitung jarak antar objek/data baru terhadap semua objek/data yang telah di *training*.
3. Mengurutkan hasil perhitungan tersebut.
4. Tentukan tetangga terdekat berdasarkan jarak minimum ke K.
5. Tentukan kategori dari tetangga terdekat dengan objek/data.
6. Gunakan kategori mayoritas sebagai klasifikasi objek/data baru.

Cara menghitung jarak tetangga biasanya dihitung berdasarkan jarak Euclidean atau *Euclidean Distance*. Rumus dari *Euclidean Distance* adalah sebagai berikut.

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \quad (2\text{-}2)$$

keterangan:

d(x,y) : jarak skalar dari dua buah vektor data x dan y yang berupa matrix berukuran d-dimensi

x : data uji

y : data latih

n : jumlah data latih

2.5 Confusion Matrix

Untuk menghitung perhitungan akurasi yang digunakan pada KNN, akan menggunakan metode *confusion matrix*. *Confusion matrix* adalah suatu metode yang

biasanya digunakan untuk untuk melakukan perhitungan akurasi pada konsep *data mining* atau sistem pendukung keputusan. Pada pengukuran kinerja menggunakan *confusion matrix* terdapat empat istilah representasi hasil proses klasifikasi, yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN). TP adalah data positif yang terdeteksi benar, TN adalah data negatif yang terdeteksi benar, FP adalah data negatif yang terdeteksi sebagai data positif, dan FN adalah data positif namun terdeteksi sebagai data negatif. Ada 3 buah perhitungan yang bisa didaptkan dari *confusion matrix*, yaitu *precision, recall, dan accuracy.*

Recall dapat didefinisikan sebagai rasio dari jumlah total yang benar terklasifikasikan dalam *class* positif dibagi dengan jumlah seluruh *class* positif, baik yang terprediksi dengan benar maupun tidak benar. Jika nilai *recall* tinggi maka menunjukkan bahwa suatu *class* diidentifikasi dengan benar. nilai *precision* adalah dengan cara membagi jumlah total yang benar terklasifikasikan dalam *class* positif dengan jumlah total prediksi pada *class* positif. Semakin tinggi hasil dari *precision* menunjukkan data yang terklasifikasikan dalam *class* positif adalah benar terklasifikasikan dalam *class* positif.

Tabel 2.1 *Confusion Matrix*

		Nilai True	
		True	False
Prediction	True	TP	FP
	False	FN	TN

$$precision = \frac{TP}{TP+FP} \quad (2\text{-}3)$$

$$recall = \frac{TP}{TP+FN} \quad (2\text{-}4)$$

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (2\text{-}5)$$

Keterangan:

TP = *true positive*

TN = *true negative*

FP = *false positive*

FN = *false negative*

2.6 Regular Expression

Regular Expression atau biasa disingkat Regex pada mulanya ditemukan pada tahun 1951 oleh seorang matematikawan bernama Stephen Cole Kleene. Regex merupakan sebuah teks (String) yang mendefiniskan sebuah pola pencarian sehingga dapat membantu kita untuk melakukan pencocokan, pencarian, dan manipulasi teks (Muhardian, 2021). Pada penelitian ini, Regex digunakan untuk melakukan pencarian dan pengambilan data dari pembacaan sensor tekuk, pembacaan sensor ini memiliki tipe data bit, dan terdapat beberapa simbol/karakter yang tidak diperlukan, maka dari itu regex digunakan untuk memilah dan hanya mengambil angka hasil pembacaan sensor saja yang kemudian nantinya dipindahkan kedalam sebuah *file* .csv.

2.7 Python

Python merupakan bahasa pemrograman tingkat tinggi yang diciptakan oleh Guido van Rossum di Centrum Wiskunde & Informatica (CWI) di Belanda pada awal tahun 1990-an. Bahasa Python terinspirasi dari bahasa pemrograman ABC. Hingga saat ini Guido masih menjadi penulis utama untuk Python, meskipun bersifat *open source* sehingga ribuan orang juga berkontribusi dalam mengembangkannya (Pythonindo, n.d).

Python banyak digunakan untuk membuat berbagai macam program, seperti program CLI, program GUI (desktop), aplikasi mobile, website, IoT (Internet of Things), game, program untuk hacking, dan lain sebagainya. Python dikenal dengan bahasanya yang sangat mudah dipelajari karena struktur sintaksnya yang rapi, dan mudah dipahami, selain mudah dibaca dan dipahami Python juga lebih efisien dibandingkan bahasa pemrograman lain seperti C, C++, Java, dan lain sebagainya. Untuk dapat melakukan sesuatu dengan lima baris kode pada bahasa lain, bisa jadi pada

Python hanya diperlukan satu baris kode sehingga pembuatan program dalam Python lebih ringkas dan cepat dibandingkan bahasa pemrograman lain. Python merupakan bahasa multifungsi yang dapat memproses teks, membuat *website*, membuat program jaringan, robotika, *data mining*, sampai dengan kecerdasan buatan. Selain itu Python juga banyak akan dukungan *library* (pustaka) standar, tersedia banyak sekali modul-modul dan ekstensi program yang sudah siap dipakai untuk membuat program sesuai kebutuhan. Python juga bahasa pemrograman yang memiliki sangat banyak pengguna dan komunitas yang menggunakannya.

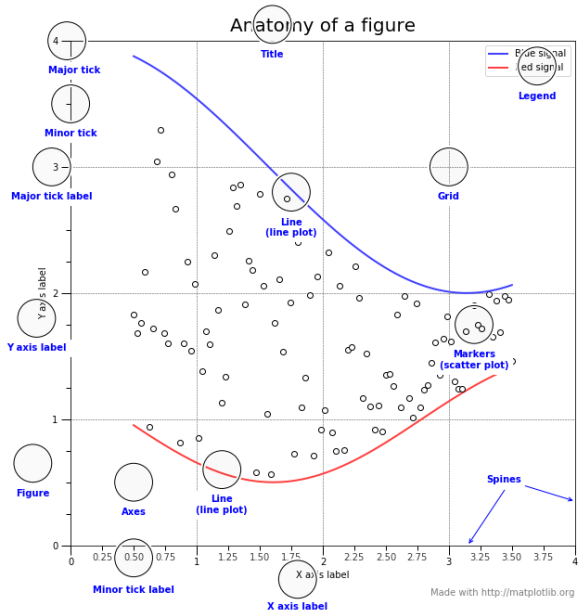
2.7.1. Numpy

Putstaka Numpy atau bisa dikenal dengan *Numerical Python* adalah pustaka pada Python yang focus pada *scientific computing.* Gunanya adalah untuk mengolah data numerik. Numpy memiliki kemampuan untuk membentuk objek N-*dimensional array* yang mirip dengan *list* pada Python. Keunggulan dari Numpy daripada *list* pada Python adalah konsumsi *memory* pada komputer yang lebih kecil, serta *run-time* yang lebih cepat. Selain itu Numpy juga memiliki keunggulan pada operasi data yang memiliki lebih dari satu dimensi seperti vektor dan Matriks.

2.7.2. Matplotlib

Matplotlib adalah pustaka Python yang berfokus pada visualisasi data seperti membuat *plot* grafik (Rohman, 2019). Matplotlib pertama kali diciptakan oleh John D. Hunter dan sekarang telah dikelola oleh tim *developer* yang besar. Awalnya Matplotlib dirancang untuk menghasilkan *plot* grafik yang sesuai pada publikasi jurnal atau artikel ilmiah. Matplotlib dapat digunakan dalam skrip Python, Python dan IPython *Shell*, server aplikasi web, dan beberapa *toolkit graphical user interface* (GUI) lainnya.

Pada umumnya visualisasi dari Matplotlib adalah sebuah gambar grafik yang terdapat satu atau lebih sumbu. Setiap sumbu memiliki sumbu horizontal (x) dan sumbu vertikal (y), data yang ditampilkan contohnya adalah lingkaran-lingkaran berwarna, garis, atau poligon. Contoh visualisasi Matplotlib dapat dilihat pada gambar 2.12.



Gambar 2.9 Contoh Visualisasi Matplotlib

(sumber: https://miro.medium.com/max/636/0*Zs-fsrrBCsyzY1Ma.png)

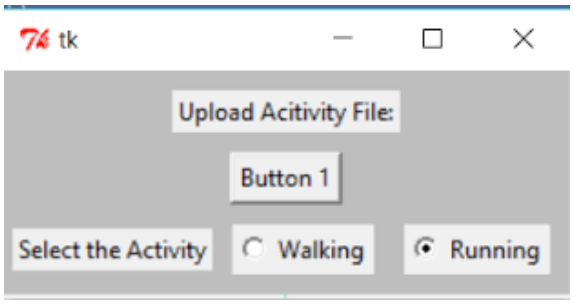
2.7.3. Scikit Learn

Scikit-learn adalah salah satu pustaka Python yang mendukung *supervised* dan *unsupervised learning.* Scikit-learn berawal dari sebuah proyek pada tahun 2007 di Google Summer of Code project oleh David Cournapeau. Pada tahun itu juga kemudian dilanjutkan oleh Matthieu Bruncher yang dikerjakan sebagai tesisnya. Pada tahun 2010 proyek ini diambil dan diketuai oleh Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, dan Vincent Michel dari INRIA dan telah dipublikasikan yaitu pada tanggal 1 Februari 2010. Sejak saat itu beberapa versi telah dirilis setiap tiga bulan sekali.

Beberapa contoh *machine learning* yang sangat umum digunakan dan didukung oleh scikit-learn adalah *Decision Tree*, SVM (*Support Vector Machine*), ANN (*Artificial Neural Network*), *Nearest Neighbors*, dan masih banyak lainnya. Karena banyaknya *machine learning* yang didukung oleh pustaka ini, banyak orang yang memillih menggunakan pustaka ini daripada harus melakukan penulisan *machine learning* secara manual. Scikit-learn biasanya di-*install* bersama dengan pustaka lain yaitu Numpy, Scipy, Matplotlib, IPython, Sympy, Pandas, dan lain sebagainya.

2.7.4. Tkinter

Untuk membuat GUI (*Graphical User Interface)* atau tampilan dari aplikasi ini, akan menggunakan salah satu pustaka dalam Python yaitu Tkinter. Tkinter adalah pustaka untuk membuat GUI standar untuk menampilkan aplikasi dengan komponen-komponen yang ada pada modul Tkinter, contoh komponennya adalah *Button* (tombol), *Textbox* (kotak teks), *Label, Frame, Window,* dan lain sebagainya. Contoh tampilan dasar dari Tkinter dapat dilihat pada gambar 2.14.



Gambar 2.10 Contoh tampilan Tkinter

(sumber: https://encrypted-tbn0.gstatic.com/images?q=tbn%3AANd9GcS8YVkOWo4fDhmxbao7Wc4jgABvJJ0xcVrAJbSZAG9T--XZF4vw)

2.8 Penelitian Terdahulu

Penelitian terdahulu mengenai pengenalan huruf dan angka bahasa isyarat menggunakan sensor tekukan, akselerometer, magnetometer dalam Glove Kit telah dilakukan dengan menggunakan metode jaringan saraf tiruan (Widodo, dkk, 2020). Pada penelitian ini yang akan dikenali adalah angka “1”, “2”, dan “3”, metode yang menggunakan adalah metode jaringan saraf tiruan. Penelitian ini memiliki tujuan utama yaitu melanjutkan penelitian terdahulu yang sudah dilakukan. Total data yang digunakan dalam penelitian ini adalah sejumlah 7250. Sebanyak 5524 buah data digunakan sebagai data latih, dan 1726 data sebagai data uji. Penelitian tersebut mendapatkan akurasi yang sangat baik yaitu diatas 95%. Namun pada penelitian tersebut hanya menggunakan gestur angka 1-3, pada angka 1-9 tidak terdapat gerakan ketika melakukan perekaman, namun angka 10 memiliki gerakan dalam gesturnya.

Sebagai dasar penggunaan metode KNN, akan diambil dari penelitian yang dilakukan oleh Devina Trixie pada tahun 2020. Penelitian ini menggunakan sensor Electromyograph (EMG) untuk mendeteksi kontraksi otot yang akan diklasifikasikan ke dalam 3 kelas yaitu yang pertama kelas *no click* pada saat tidak ada kontraksi otot, kelas kedua adalah *left click* yaitu pada saat terjadi kontraksi otot sekali, dan terakhir adalah kelas *right click* yaitu pada saat terjadi kontraksi otot dua kali secara beruntun dan cepat. Terdapat 2 metode yang dibandingkan dalam penelitian ini yaitu KNN. Pengambilan data pada penelitian ini dapat dilihat pada gambar 2.11. Pada metode KNN dilakukan percobaan menggunakan k = 3, 5, 7, 9, dan 11, dan didapatkan akurasi tertinggi pada saat k = 3 yaitu sebesar 81,81%.

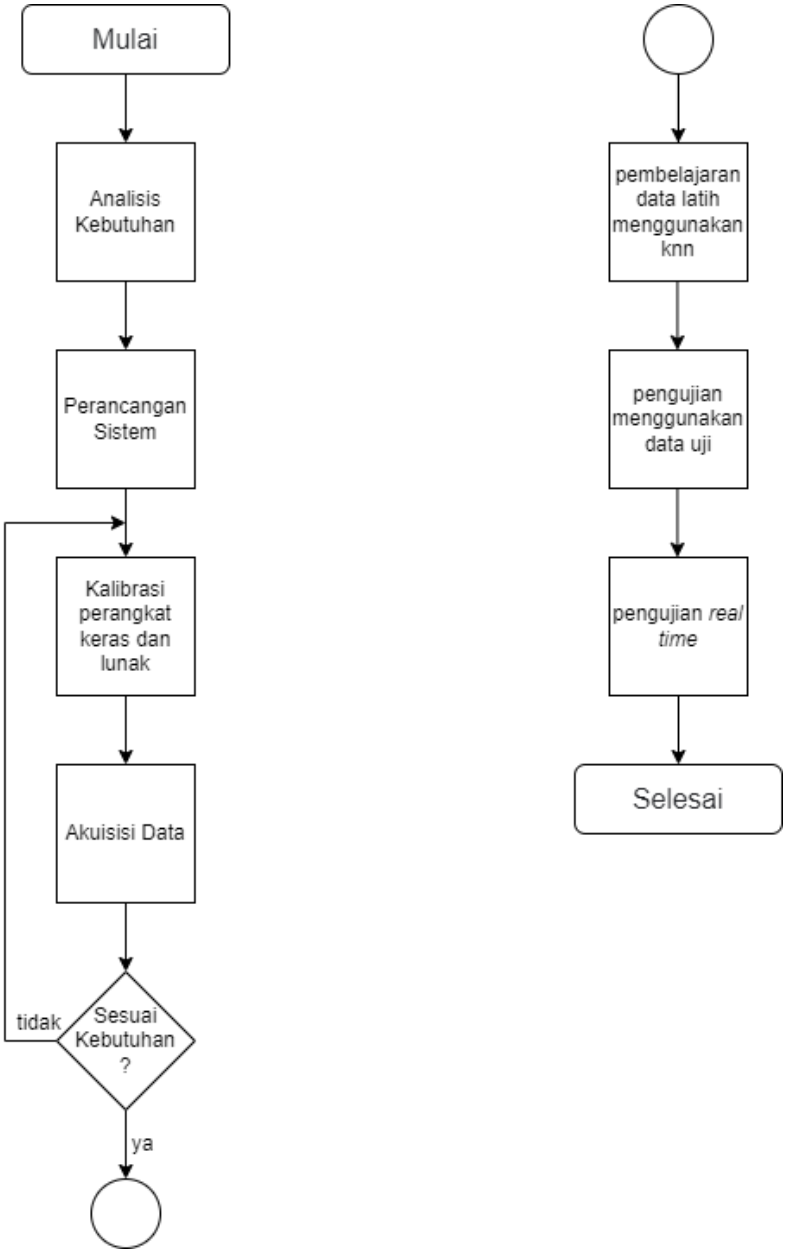
Gambar 2.11. Pengambilan data EMG

Bab III

Analisis dan Perancangan Sistem

3.1 Tahapan Penelitian

Tahapan penelitian dalam perancangan kecerdasan buatan untuk mendeteksi huruf atau angka serta penelusuran kamus tercantum pada gambar 3.1 berikut.



Gambar 3.1 Tahapan perancangan sistem

` Tahapan awal penelitian dimulai dari mencari permasalahan yang ada. Dilanjutkan dengan analisis kebutuhan untuk menentukan alat, bahan, dan hal-hal lain yang diperlukan dan berkaitan dengan proses identifikasi ini. Pada tahapan berikutnya dilakukan perancangan sistem guna membuat sistem yang memadai seefektif dan efisien mungkin. Setelah sistem berhasil dirancang, dilakukan kalibrasi untuk menentukan titik awal posisi tangan berada. Setelah proses kalibrasi dilakukan proses akuisisi data untuk mendapatkan data hasil pembacaan sensor yang akan digunakan sebagai masukan untuk mendeteksi angka yang akan diperagakan. Kemudian akan dilakukan proses pembelajaran data latih menggunakan KNN. untuk mendapatkan hasil keluaran angka. Setelah itu akan dilakukan pengujian terhadap data latih dan data uji untuk memperoleh akurasi dari hasil pembelajaran. Terakhir akan dilakukan pengujian terhadap beberapa jumlah subyek untuk mendapatkan hasil uji akhir.

3.2 Analisis Kebutuhan

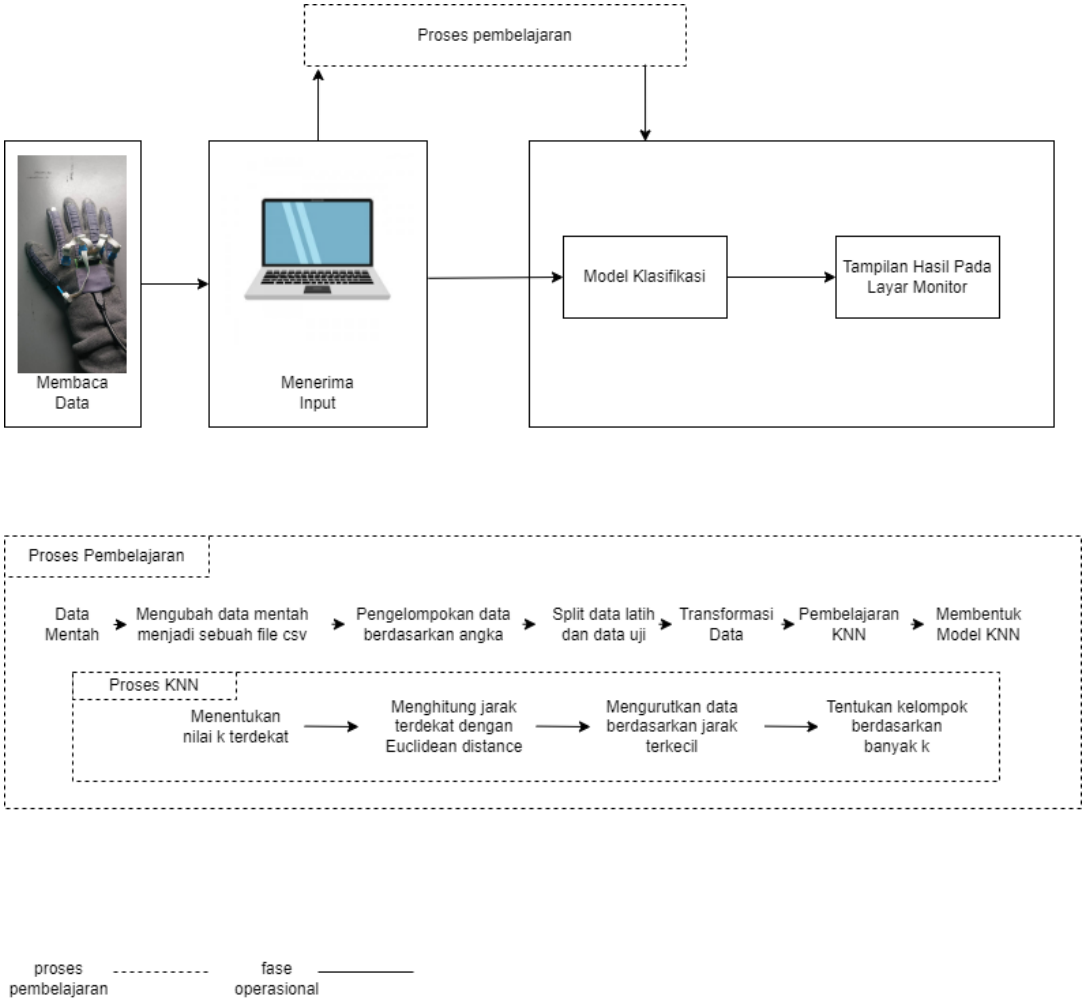
Manusia memerlukan beberapa bagian tubuh untuk mengoperasikan perangkat komputer (tetikus, papan tombol, dan lainnya). Bagian tubuh yang paling dominan dalam mengoperasikan perangkat komputer adalah tangan dan jari. Namun ada beberapa kondisi dimana manusia tidak dapat menggunakan perangkat komputer. Salah satu contohnya adalah penyandang disabilitas yang tidak dapat mengoperasikan perangkat komputer dengan semestinya dikarenakan keterbatasan fisik yang mereka miliki. Karena hal tersebut dibutuhkan adanya alat yang dapat membantu orang-orang tersebut untuk mengoperasikan perangkat komputer.

Pada penelitian sebelumnya telah dibuat alat berupa sensor tekukan yang dipasangkan pada sarung tangan untuk mendeteksi gestur yang dilakukan serta metode jaringan saraf tiruan dan memiliki akurasi diatas 90% (Widodo, dkk, 2020). Luaran dari alat ini adalah mengklasifikasikan angka 1, 2, atau 3 dari gerakan tangan yang dilakukan oleh pengguna. Pada penelitian ini akan dikembangkan alat yang dapat mengklasifikasikan angka 1 sampai 10, menggunakan gerakan gestur tangan manusia dibantu dengan sarung tangan yang dipasang sensor tekukan.

Perangkat yang digunakan mampu mendeteksi tekukan pada jari-jari tangan dengan sensor tekukan, sensor ini membaca tekukan tiap sendi pada jari dan menerjemahkannya menjadi data numerik. Selain itu ada sensor akselerometer yang mampu membaca pergerakan dan arah telapak tangan. Data-data tersebut nanti akan dijadikan masukan pada metode KNN yang kemudian akan diidentifikasi angka yang sudah diperagakan oleh bahasa isyarat tersebut. Setelah selesai proses pembelajaran, keluaran akan ditampilkan dalam layar.

3.3 Desain Sistem

Desain kinerja sistem dapat dijabarkan pada bagian alur yang dimuat pada ilustrasi yang terdapat pada gambar 3.2.



Gambar 3.2 Ilustrasi proses klasifikasi

Spesifikasi komputer yang akan digunakan dalam penelitian ini dapat dilihat pada tabel 3.1.

Tabel 3.1 Spesifikasi Komputer

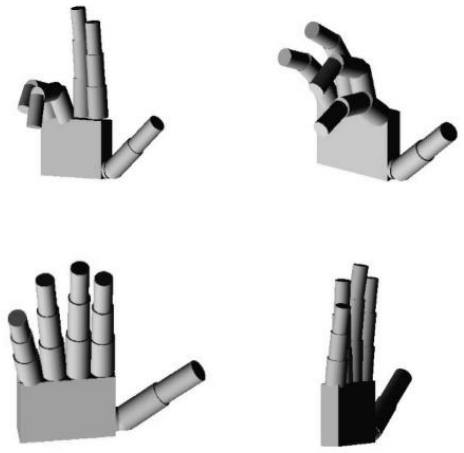
Komponen	Keterangan
Sistem operasi	Windows 10 Home 64-bit (10.0, Build 18362)
Prosesor	Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz (8CPUs), ~2.8GHz
Memori	8192 MB
Resolusi layar	1920*1080
GPU	NVIDIA GeForce GTX 960M

Seperti yang ada pada diagram alur pada gambar 3.2, alur sistem mulai berawal dari pembacaan data yang diambil dari pergerakan tangan manusia untuk memperagakan bahasa isyarat menggunakan *flex point glove kit*. Data mentah yang telah dibaca kemudian akan diubah menjadi sebuah data dengan format .csv untuk dapat mempermudah pengelompokan data yang nantinya perlu dilakukan. Pengelompokan data dilakukan dengan menjadikan setiap angka yang akan diklasifikasi memiliki nama file yang sama dan berurutan. Disini juga dilakukan pengelompokan yang nantinya akan memisahkan seluruh data menjadi data latih dan data uji. Dari 150 total data yang ada, 120 data akan digunakan sebagai data latih, dan 30 sisanya akan digunakan sebagai data uji. Langkah berikutnya adalah melakukan transformasi data yang sebelumnya sendiri-sendiri menjadi sebuah file yang baru, hal ini dilakukan karena data tadi berupa *multivariate time series*. Transformasi ini dilakukan dengan menggabungkan keseluruhan data menjadi satu, hasil transformasi ini akan menghasilkan 2 file yang berbeda yaitu 1 file berisi seluruh data latih, dan 1 file berisi seluruh data uji. Langkah selanjutnya adalah hasil dari transformasi tadi akan dilakukan pembelajaran menggunakan metode KNN dan akan menghasilkan model yang nantinya akan digunakan sebagai pengujian secara *real-time.*

Pembelajaran pertama yaitu menggunakan metode KNN. Proses awal yang dilakukan adalah menentukan nilai k. Setelah didapat nilai k tersebut, proses selanjutnya adalah menghitung jarak menggunakan *Euclidean*. Hasil dari perhitungan tersebut akan diurutkan berdasarkan nilai terkecil, selanjtunya adalah menentukan kelompok data berdasarkan banyaknya k.

3.3.1 Gerakan Gestur Tangan

Seperti yang sudah dijelaskan pada bab 2, pergerakan tangan manusia untuk memperagakan bahasa isyarat memerlukan gerakan pada jari-jari, telapak tangan, lengan bawah, serta lengan atas. Banyak bahasa isyarat yang memerlukan dua tangan untuk memperagakan bahasa isyarat, namun penelitian ini khusus hanya mengambil angka dari bahasa isyarat SIBI yang hanya menggunakan gestur satu tangan yaitu tangan kanan. Pembacaan sensor atas pergerakan tangan inilah yang nantinya akan menjadi masukan bagi pembelajaran KNN. Setiap pergerakan tangan memiliki 16 buah data yang nantinya akan diproses dan hanya diambil 10 data sensornya saja. Untuk visualisasi pergerakan jari dan telapak tangan manusia dapat diperlihatkan menggunakan aplikasi bawaan dari flexpoint glove kit dan dapat dilihat pada gambar 3.3 berikut.



Gambar 3.3 Visualisasi pembacaan data flexpoint glove

Berikut adalah penjelasan data apa saja yang dibaca oleh sensor-sensor yang ada pada perangkat tersebut,

Metacarpo-phalangeal : tekukan jari yang terdapat pada telapak tangan

Proximal interphalangeal : tekukan jari setelah *metacarpo-phalangeal*

0 : tekukan *metacarpo-phalangeal* jari kelingking

1 : tekukan *proximal interphalangeal* jari kelingking

2 : tekukan *metacarpo-phalangeal* jari manis

3 : tekukan *proximal interphalangeal* jari manis

4 : tekukan *metacarpo-phalangeal* jari tengah

5 : tekukan *proximal interphalangeal* jari tengah

6 : tekukan *metacarpo-phalangeal* jari telunjuk

7 : tekukan *proximal interphalangeal* jari telunjuk

8 : tekukan *metacarpo-phalangeal* ibu jari

9 : tekukan *proximal interphalangeal* ibu jari

10 : sudut X pembacaan sensor akselerometer

11 : sudut Y pembacaan sensor akselerometer

12 : sudut Z pembacaan sensor akselerometer

13 : sudut X pembacaan sensor magnetometer

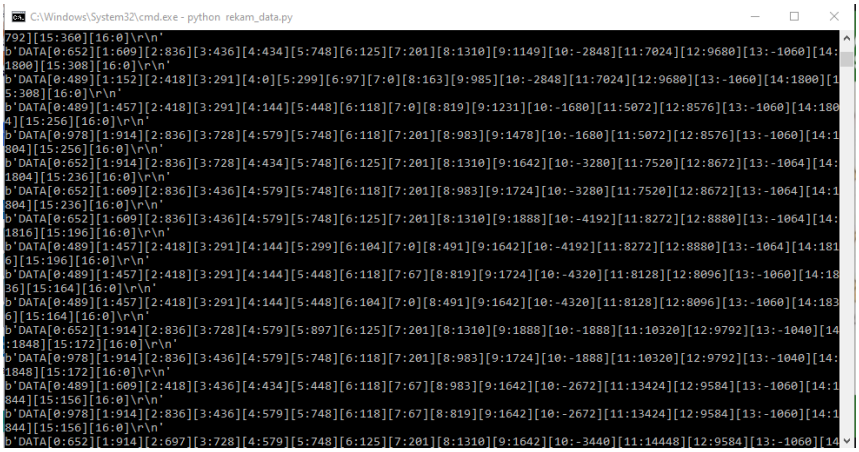
14 : sudut Y pembacaan sensor magnetometer

15 : sudut Z pembacaan sensor magnetometer

3.3.2 Pembacaan Data Oleh Sensor

Seperti yang dipaparkan pada bab 2. Perangkat dapat membaca 16 buah data yang nantinya akan diproses oleh *machine learning*. Namun hanya 10 data dari sensor tekuk yang akan digunakan sebagai data utama penelitian ini. Pengambilan 10 data dari sensor ini dikarenakan untuk gestur angka 1 sampai 10 dilakukan hanya menggerakkan pergelangan tangan, namun tidak menggerakkan lengan tangan, sehingga sensor magnetometer dan akselerometer tidak akan memiliki dampak pada penelitian ini. Contoh data yang akan diproses dapat dilihat pada gambar 3.4. Tidak semua data akan

digunakan sebagai masukan dalam *machine learning*, hanya data dari sensor tekukan yang nanti akan diambil sebagai masukan yaitu data indeks ke-0 hingga data indeks ke-9. Data akan kemudian dimasukkan kedalam file dengan format .csv yang kemudian akan dijadikan sebagai masukan untuk metode KNN. Data dalam .csv dapat dilihat dalam gambar 3.5.



Gambar 3.4 Hasil pembacaan sensor

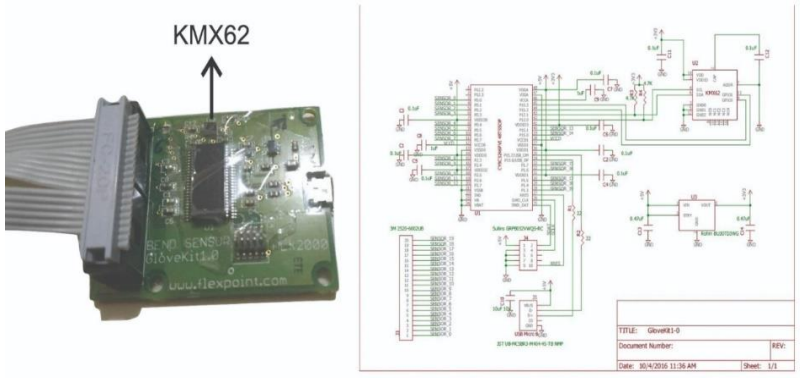
489	306	278	291	144	448	118	0	819	328	2208	896	14128	-840	1524	916
652	459	418	436	434	748	118	134	819	328	2208	896	14128	-840	1524	916
163	0	278	291	144	448	118	0	819	246	2224	896	14144	-828	1524	912
489	0	278	291	0	299	104	0	491	246	2224	896	14144	-828	1524	912
652	459	418	728	434	897	125	134	1310	492	2176	880	14224	-840	1520	904
652	306	418	436	434	748	118	134	819	328	2176	880	14224	-840	1520	904
652	306	418	436	434	897	125	134	1310	492	2224	912	14160	-828	1532	912
652	459	418	436	434	748	118	134	819	328	2224	912	14160	-828	1532	912
163	0	0	291	0	448	104	0	819	246	2192	912	14160	-824	1524	912
163	0	0	0	0	299	97	0	163	82	2192	912	14192	-824	1520	904
163	0	0	0	0	299	104	0	491	82	2192	912	14192	-824	1520	904
489	152	278	291	144	299	104	67	491	246	2192	880	14176	-832	1540	900
489	152	418	291	144	448	118	67	819	328	2192	880	14176	-832	1540	900
652	609	697	436	434	748	125	201	819	328	2224	880	14192	-848	1536	904
652	457	418	728	434	897	125	201	983	492	2224	880	14192	-848	1536	904
652	457	418	436	434	448	118	201	819	492	2224	928	14192	-836	1528	916
652	457	697	436	434	748	125	201	1310	492	2224	928	14192	-836	1528	916
489	152	278	291	144	299	104	67	491	246	2208	928	14192	-836	1512	904
489	152	278	291	144	448	118	67	819	246	2208	928	14192	-836	1512	904
489	152	278	291	0	299	104	0	491	246	2240	880	14192	-840	1524	920
652	609	418	728	434	748	125	201	983	492	2240	880	14192	-840	1524	920
652	457	418	436	434	748	118	201	819	328	2240	928	14224	-828	1512	908
489	152	418	291	144	448	118	67	819	328	2240	928	14224	-828	1512	908
978	457	418	436	434	748	125	201	819	328	2176	896	14224	-836	1512	912
489	457	418	436	144	448	118	67	819	328	2176	896	14224	-836	1512	912
163	152	0	0	0	299	97	0	163	82	2256	912	14224	-824	1516	908
652	457	418	728	434	748	125	201	1310	492	2256	912	14224	-824	1516	908
978	457	418	436	434	748	125	201	819	328	2224	928	14208	-836	1524	912
652	457	418	436	434	748	125	201	983	492	2224	928	14208	-836	1524	912
652	457	418	436	434	748	118	201	819	328	2240	944	14160	-836	1516	912
489	457	278	436	144	448	118	67	983	328	2240	944	14160	-836	1516	912
489	152	278	291	144	448	104	67	491	246	2208	896	14208	-840	1532	908
489	152	278	291	144	448	118	67	819	246	2208	896	14208	-840	1536	908
489	152	278	291	144	299	104	0	491	246	2240	928	14176	-840	1536	908

Gambar 3.5 Hasil pembacaan sensor dalam file berformat .csv

3.3.3 USB Glove Kit

USB Glove Kit terdiri dari tiga buah komponen utama yang terdiri atas 1) *microcontroller* (komputer mini) yaitu PSoC CY8C3246PVI-147. Tujuan penggunaan

microcontroller ini adalah karena memiliki arus DAC (alat pengubah sinyal digital menjadi analog) yang sudah terintegrasi. Hal ini memungkinkan sensor tekukan akan dihubungkan dengan ADC secara langsung tanpa memerlukan komponen eksternal seperti resistor pembagi tegangan atau op-AMPS. *Microcontroller* dan skemanya dapat dilihat pada gambar 3.6. 2) Lima buah sensor tekukan yang berguna untuk membaca pergerakan jari-jari tangan seperti yang sudah dipaparkan pada bab 2. 3) sebuah alat gabungan sensor akselerometer dan sensor magnetometer Kionix KMX62 yang berguna untuk membaca pergerakan tangan, hal ini juga sudah dipaparkan pada bab dua.



Gambar 3.6 Gambar dan skema *microcontroller*

3.3.4 Pembelajaran Data Menggunakan K-Nearest Neighbor

Pada penelitian ini untuk mengklasifikasikan hasil pergerakan gestur tangan menjadi huruf atau angka akan menggunakan metode K-Nearest Neighbor (KNN). Metode pembelajaran KNN yang digunakan pada penelitian ini dimulai dari menginisialisasi nilai k sebagai penentu data masukan akan diklasifikasikan pada kelas yang mana. seperti yang sudah dipaparkan pada bab dua, nilai k harus bernilai angka ganjil. umumnya inisialisasi nilai k dimulai dengan nilai 3, 5, 7, dan seterusnya. Setelah itu akan dilakukan perhitungan untuk mencari jarak Euclidean (Euclidean Distance), langkah berikutnya akan dilakukan penentuan kelompok data berdasarkan jarak terdekat. Contohnya, jika k = 3, maka akan memilih 3 jarak terdekat. dari ketiga jarak tersebut akan didapatkan kesimpulan data baru tersebut memasuki kelompok yang

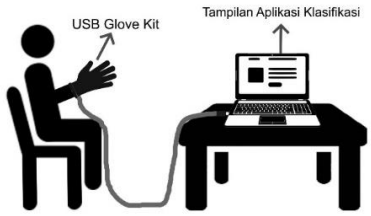
mana. setelah mendapatkan hasil dari klasifikasi, akan dilakukan proses perhitungan akurasi menggunakan confusion matrix. proses ini akan terus diulang hingga mendapatkan akurasi tertinggi dengan cara mengganti-ganti nilai k. Diagram alur pembelajaran K-NN pada penelitian ini dilihat pada gambar 3.7. Setelah menguji dengan nilai k yang berbeda-beda, nilai k dengan akurasi tertinggi akan dibuat sabagai model yang nantinya akan digunakan sebagai model pengujian secara *real-time.*

Gambar 3.7 Diagram alur KNN

3.4 Implementasi Sistem

Pada tahap implementasi sistem merupakan tahap lanjutan setelah diagram alur dari tiga metode telah selesai dibuat. Implementasi yang dilakukan adalah membuat sistem pembelajaran dengan data latih berjumlah 10 gestur (10 angka dari 1-10). Masing-masing data yang dilatih memiliki jumlah data sekitar 95-100 data dalam satu kali rekam yang direkam selama satu detik. Oleh sebab itu peneliti akan membuat setiap perekaman memiliki jumlah yang sama yaitu 100 data setiap perekaman untuk memudahkan proses pembelajaran. Setiap data akan dilakukan 15 kali rekam untuk mendapatkan sampel yang cukup demi mencapai akurasi yang tinggi.

Seperti yang dijelaskan sebelumnya, pengambilan data akan dilakukan dengan menggerakkan perangkat sesuai dengan gestur yang diperagakan dan sesuai dengan angka dalam kamus SIBI. Data tersebut akan dibaca dan kemudian akan dijadikan nilai masukan dalam pembelajaran metode yang telah dipaparkan sebelumnya. Hasil pembelajaran dan pengujian akan ditampilkan dalam aplikasi yang dibuat menggunakan TKinter. *Mockup* penggunaan perangkat terpapar pada gambar 3.8.



Gambar 3.8 *Mockup* penggunaan perangkat

3.5 Rancangan Pengujian

Pengujian penelitian ini dilakukan dengan menguji gestur angka SIBI dari "1" hingga "10". Masing-masing akan memiliki 15 data yang berbeda-beda, penelitian ini dilakukan hanya dengan menggunakan sampel yang dilakukan oleh peneliti sendiri karena adanya PPKM. Total data yang digunakan adalah 150 data (10 kelas yang akan diklasifikasi dan 15 data pada masing-masing kelas), 120 data akan digunakan sebagai data latih, dan 30 sisanya akan digunakan sebagai data uji. Data tadi akan digunakan sebagai pembelajaran pada metode KNN. Pengujian dilakukan dengan dua cara, yaitu

pengujian pertama dengan menggunakan satu variasi data sampel yang akan diujikan secara *real-time* kepada 10 subyek yang berbeda, dan pengujian kedua yaitu dengan menggunakan lima variasi data sampel yang akan diujikan secara *real-time* kepada 5 subyek yang berbeda.

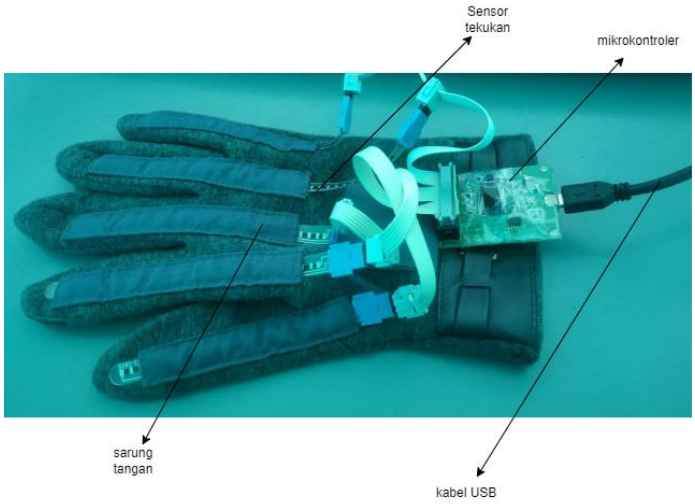
BAB IV
Hasil dan Pembahasan

4.1 Hasil Pengumpulan Data

Subab ini membahas proses pengambilan dan pengolahan data yang digunakan yaitu pengambilan data meliputi berapa data yang diambil, dan apa saja data yang digunakan. Semua langkah akan dijelaskan secara detil sehingga data dapat digunakan sebagai data latih untuk pengujian secara *real time* kepada responden.

4.1.1 Penggunaan Flexpoint Glove Kit

Tahap awal penelitian seperti yang dijelaskan pada bab sebelumnya adalah pengambilan data. Data yang diambil dari sensor tekuk Flex Point dihubungkan pada mikrokontroler yang sudah disediakan, kemudian dipasangkan pada sarung tangan. Mikrokontroler ini nantinya akan dihubungkan ke komputer menggunakan kabel USB. Setelah sensor telah terpasang dengan benar, pengambilan data akan dilakukan menggunakan program aplikasi yang dibuat menggunakan bahasa pemrograman Python. Pengambilan data dilakukan dengan melakukan gestur bahasa isyarat SIBI untuk angka 1-10 masing-masing sebanyak 15 kali perekaman dengan 12 data digunakan sebagai data latih dan 3 sisanya digunakan sebagai data uji. Sensor yang dibaca oleh program berupa 16 total sensor yaitu 10 data dari sensor tekukan, 3 data dari sensor akselerometer, dan 3 data dari sensor magnetometer. Namun hanya 10 data dari sensor tekukan yang akan diambil sebagai data yang digunakan. Tujuan hanya menggunakan 10 data dari sensor tekuk adalah karena gestur angka yang digunakan tidak memerlukan gerakan lengan, melainkan hanya menggunakan gerakan telapak tangan dan jari-jari saja. Pengambilan sampel hanya dilakukan oleh peneliti sebanyak 15 kali, karena penelitian dilakukan saat ada PPKM (Pemberlakuan Pembatasan Kegiatan Masyarakat). Gambar alat yang digunakan dapat dilihat pada gambar 4.1.

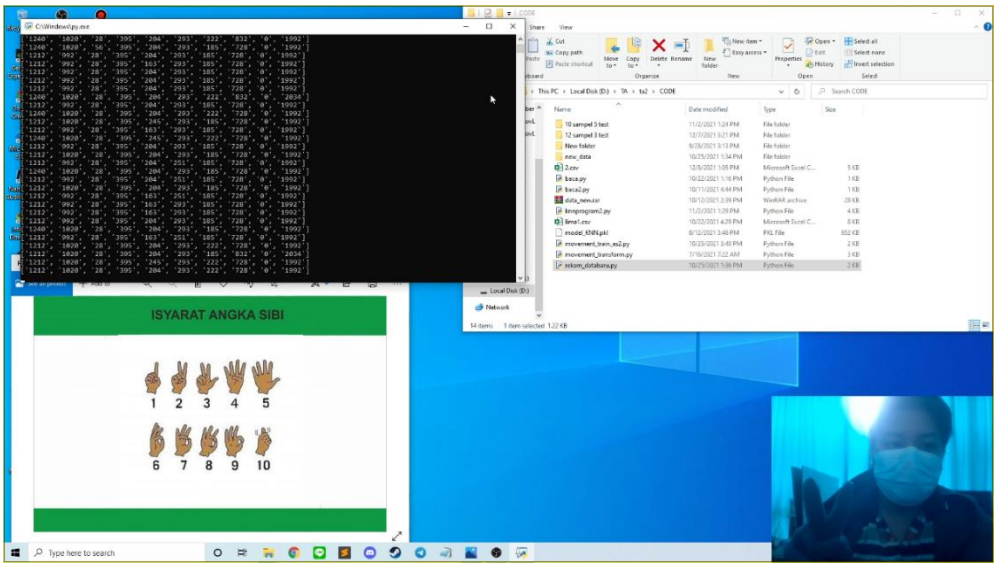


Gambar 4.1 Alat yang digunakan pada peneltian ini

4.1.2 Proses Pengambilan Data

Seperti yang sudah dipaparkan sebelumnya, Flex Point hanya menyediakan program bawaan berupa aplikasi tampilan yang hanya menampilkan ilustrasi gerakan jari yang diperagakan, namun tidak dapat mencatat/mengambil data dari sensor tersebut. Maka dari itu langkah awal adalah peneliti akan melakukan pembuatan kode yang dibuat menggunakan bahasa pemrograman Python untuk dapat membaca dan mengambil data hasil pembacaan sensor Flex Point.

Pembuatan kode ini menggunakan pustaka utama yaitu PySerial, sebuah pustaka yang dapat membaca data dari serial port. Hal ini dijelaskan dalam buku manual Flex Point yaitu data dari Flex Point dapat diperlihatkan di aplikasi terminal serial. Parameter yang diperlukan juga telah lengkap dituliskan di manual, seperti Baudrate, data bits, stop bit, dan lain-lain, namun untuk port yang terpasang pada komputer harus dilihat secara manual melalui *device manager*. Proses tampilan pengambilan data dapat dilihat pada gambar 4.2.



Gambar 4.2 Tampilan proses pengambilan data

Pengambilan data dari sensor akan dicatat dalam sebuah file berformat .csv. Data yang diambil dari sensor Flex Point adalah 10 data bacaan sensor tekuk, yaitu masing-masing jari memiliki 2 data bacaan (*metacarpo-phalangeal*, dan *proximal phalangeal*). Dalam 1 detik perekaman, setidaknya didapatkan 95 hingga 105 baris data Dalam sekali rekam gestur, kemudian untuk memudahkan penelitian dilakukan sekali perekaman akan dibuat sama semua yaitu merekam 100 baris data pembacan sensor, yang berarti total 1000 data dalam sekali perekaman. Pengambilan data setiap angka dilakukan satu per satu setiap gestur sebanyak 15 kali. Contoh 1 file perekaman data dapat dilihat pada tabel 4.1 berikut. Berikut penjelasan tentang simbol S1, S2, dan seterusnya yang ditampilkan dalam tabel berikut.

S1 dan S2 : sensor *metacarpo phalangeal*, dan *proximal interphalangeal* jari kelingking.

S3 dan S4 : sensor *metacarpo phalangeal*, dan *proximal interphalangeal* jari manis.

S5 dan S6 : sensor *metacarpo phalangeal*, dan *proximal interphalangeal* jari tengah.

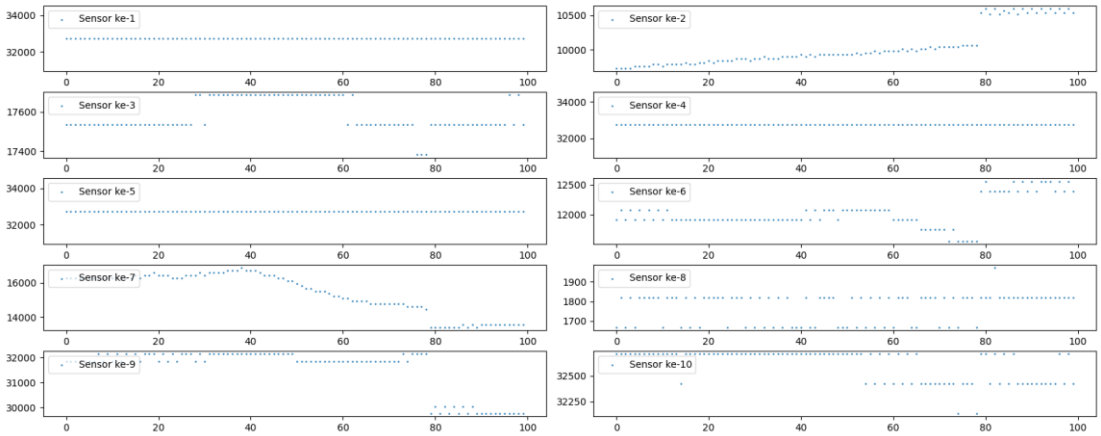
S7 dan S8 : sensor *metacarpo phalangeal*, dan *proximal interphalangeal* jari telunjuk.

S9 dan S10 : sensor *metacarpo phalangeal*, dan *proximal interphalangeal* ibu jari.

Tabel 4.1 Contoh file hasil perekaman data angka 1

No.	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10
1	32738	9732	17534	32739	32738	11912	16281	1668	31841	32710
2	32738	9732	17534	32739	32738	12074	16281	1820	31841	32710
3	32738	9732	17534	32739	32738	11912	16281	1668	31841	32710
…	…	…	…	…	…	…	…	…	…	…
99	32738	10597	17686	32739	32738	12558	13563	1820	29757	32710
100	32738	10541	17534	32739	32738	12396	13563	1820	29757	32425

Contoh grafik persebaran data dari gestur angka 1 sampel pertama dapat dilihat pada gambar 4.3 berikut, garis vertikal melambangkan nilai hasil pembacaan sensor, garis horizontal melambangkan baris keberapa data tersebut.



Gambar 4.3 Grafik persebaran data gestur angka 1 sampel pertama

4.1.3 Proses Pengolahan Data

Hasil data yang telah didapatkan pada langkah sebelumnya akan dimasukkan kedalam sebuah folder dengan nama dataset_number dan diberi nama file yang urut agar Python dapat membaca secara urut juga. Dari 150 data yang ada. 80% data akan digunakan sebagai data latih (120 data), dan sisanya 20% digunakan sebagai data uji. Untuk mengetahui data tersebut mewakili gestur apa, dilakukan dengan membuat

sebuah file bernama targets.csv. Pembuatan file ini dilakukan secara manual dengan melihat dan meneliti data tersebut memiliki gestur yang mana. isi file ini memasangkan seluruh data yang ada dalam folder dataset_number dengan gestur asli angka yang diperagakan. Hasil pembuatan file tersebut dapat dilihat dalam tabel 4.2. File berisi "Data" dari folder dataset_number yang telah diurutkan, "Targets" adalah angka gestur hasil sebenarnya dari "Data" tersebut

Tabel 4.2 Isi file targets.csv

No.	Data	Targets
1	1	1
2	2	1
…	…	…
149	149	10
150	150	10

4.1.4 Proses *Split* Data Latih dan Data Uji Serta Transformasi Data

Proses pemisahan/pembagian data antara data latih dan data uji dilakukan dengan membuat sebuah file yang mengindikasikan file mana yang berupa data latih, dan file mana yang berupa data uji. File tersebut juga dibuat dalam format .csv dengan nama groups.csv. Berbeda dengan yang sebelumnya, dalam 1 folder yang berisi seluruh data tadi, akan memasangkan seluruh data yang ada dalam folder dataset_number dengan angka 1 yang mengindikasikan data latih, dan sisanya angka 2 yang mengindikasikan data uji. Dari 150 data yang ada, 120 merupakan 1 (data latih), dan sisanya 30 merupakan 2 (data uji). Hasil pembuatan file ini dapat dilihat dalam tabel 4.3.

Tabel 4.3 Isi file groups.csv

No.	Data	Groups
1	1	1
2	2	1
…	…	…

149	149	2
150	150	2

Setelah data sudah selesai dikelompokkan, langkah berikutnya yang dilakukan adalah melakukan transformasi data. Karena dalam satu kali perekaman atau bisa dibilang dalam 1 kali perekaman terdapat 100 baris data (*multivariate time series)*, seluruh data dalam 1 kali perekaman akan ditransformasi menjadi satu baris, dan seluruh data yang ada dijadikan dalam sebuah file csv, akan dibuat dua buah file yang berbeda, 1 file berisi seluruh data dari data latih (120 baris data) dengan nama es2_train.csv, dan 1 file lagi berisi seluruh file dari data uji (30 baris data) dengan nama es2_test.csv. Contoh data es2_train dapat dilihat pada tabel 4.4.

Tabel 4.4 Tabel isi data es2_train.csv

	A	B	C	…	ALJ	ALK	ALL	ALM
1	32738	10541	17534	…	1668	31841	32710	1
2	32738	10541	22198	…	2578	32728	32140	1
3	32738	10653	20018	…	2578	28676	28029	1
…	…	…	…	…	…	…	…	…
119	5889	4944	29	…	32767	10523	21638	10
120	32735	13494	10199	…	32739	4733	3686	10

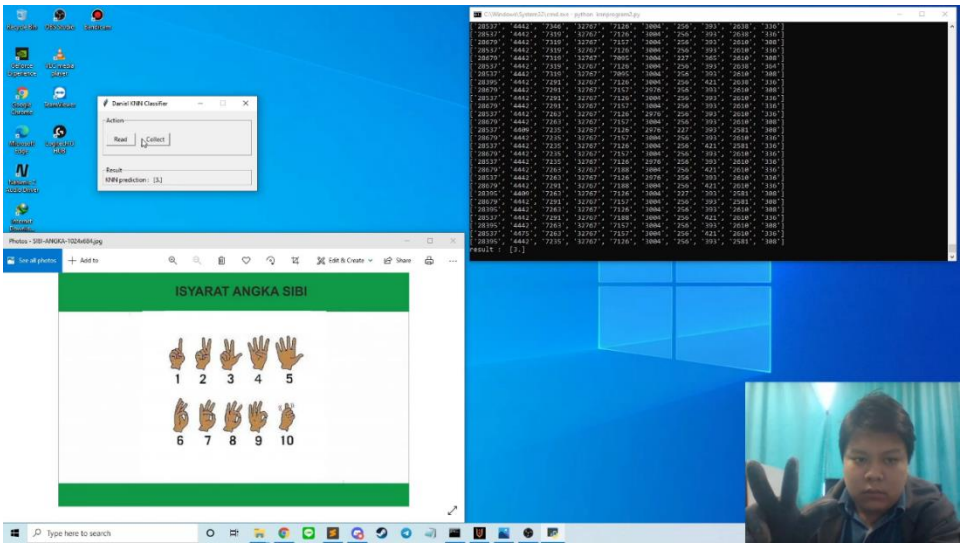
4.2 Hasil Rancangan Aplikasi

Subbab ini akan menjelaskan lebih detil tentang hasil rancangan program/aplikasi yang telah dibuat. Terdapat 3 tampilan utama yang akan diperlihatkan pada program ini yaitu tampilan utama, serta tampilan CLI sensor yang dibaca secara real-time.

4.2.1 Antarmuka Tampilan Aplikasi

Antarmuka tampilan aplikasi adalah tampilan utama pada program ini. Pembuatan tampilan ini menggunakan pustaka Tkinter dari Python. Tampilan ini

dibuat sangat sederhana, hanya memiliki 2 tombol utama yang nantinya akan digunakan pada saat pengujian *real-time,* serta sebuah label yang menunjukkan hasil klasifikasi. Tombol-tombol tersebut adalah tombol "READ" yang berfungsi untuk membaca sensor secara *real-time,* kemudian tombol "CLASSIFY" untuk mengklasifikasi gestur yang telah di peragakan. Cara kerja tombol "CLASSIFY" adalah dengan mengambil 100 data terakhir yang telah dibaca secara *real-time*, kemudian dilakukan proses transformasi seperti yang dijelaskan pada subbab 4.1.4, baru kemudian dilakukan prediksi terhadap *file* yang menyimpan data latih yang telah dibuat yaitu KNN. Dalam pembuatan program ini sebelumnya diharapkan dapat menampilkan hasil pembacaan sensor dalam tampilan/UI yang telah dibuat, namun setelah diterapkan, terjadi *delay* pada pembacaan sensornya, alhasil hal tersebut mengurangi kinerja program klasifikasi dan pembacaan sensor hanya diperlihatkan dalam bentuk *command line*. Tampilan program dapat dilihat pada gambar 4.4.



Gambar 4.4 Antarmuka tampilan program dan *command line* pembacaan sensor

4.3 Hasil Pengujian Sistem

Subbab ini akan membahas secara detil mengenai hasil pengujian dari program yang telah dibuat menggunakan metode KNN. Pengujian ini akan memperlihatkan akurasi program klasifikasi terhadap data tes yang digunakan, serta akurasi program saat dilakukan pengujian secara *real-time* kepada beberapa subyek.

4.3.1 Hasil Pengujian Data Uji

Pengujian real-time dilakukan dua kali, yaitu dengan menggunakan data latih yang hanya memiliki 1 variasi sampel, dan pengujian dengan memiliki 5 variasi sampel. Pengujian ini dilakukan untuk mengetahui perbandingan mana yang memiliki akurasi terbaik. Pada pengujian dengan 1 variasi sampel, subyek pengujian berjumlah 10 orang, sedangkan pada 5 variasi sampel pengujian dilakukan kepada subyek berjumlah 5 orang.

4.3.1.1 Hasil Pengujian Data Uji dengan 1 Variasi Sampel Latih

Sebelum melakukan pengujian terhadap data uji yang sudah disiapkan, dalam pembuatan *file* yang menyimpan data latih juga telah dilakukan proses normalisasi data untuk menghindari nilai data yang terlalu besar atau terlalu kecil. Normalisasi data yang digunakan adalah standard scaler. Pengujian menggunakan metode KNN dilakukan dengan cara mengubah parameter k dan memperhatikan akurasi yang dihasilkan. Parameter k ini merupakan angka ganjil yang biasanya dimulai dari angka 3 dan seterusnya, pada pengujian ini dilakukan pengujian pada saat k = 3 hingga k = 11.

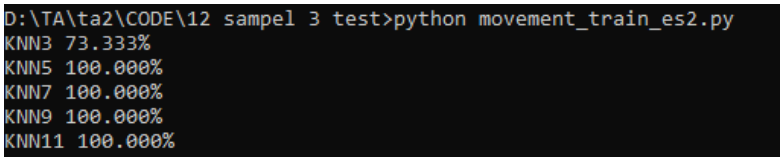
A. Pemilihan *File* Data Train dan Nilai k

Pada saat k = 3, program hanya memiliki akurasi sebesar 73.33%, sedangkan pada saat k = 5 hingga k = 11, program memiliki akurasi sebesar 100%. Diputuskan dalam pengujian secara *real-time* akan menggunakan *file* dengan parameter k=5. Hasil pengujian tiap parameter dapat dilihat pada tabel 4.5.

Tabel 4.5 Hasil pengujian KNN dengan parameter k (1 variasi)

No.	k	Akurasi
1	3	73.333%
2	5	100%
3	7	100%

4	9	100%
5	11	100%



Gambar 4.5 Hasil latih akurasi pada KNN dengan nilai k yang berbeda

B. Hasil Pengujian *Real-Time*

Pengujian secara *real-time* metode KNN ini dilakukan dengan cara menguji secara langsung kepada beberapa subyek saat sensor terus menerus membaca sensor. Pengujian dilakukan sebanyak 3 kali untuk setiap gestur angka yang diperagakan. Setiap sekali gestur diperagakan dan diklasifikasi hasilnya, akan dilakukan *reset* ulang sensor dengan cara melakukan gestur menggenggam dan membuka telapak tangan beberapa kali sebelum melakukan gestur berikutnya. Hal ini dilakukan untuk menghindari *delay* pada pembacaan sensor dan menghindari salah identifikasi pada gestur yang diperagakan. Pengujian ini dilakukan kepada 10 subyek yang berbeda-beda, variasi yang berbeda pada pengujian *real-time* ini adalah banyaknya perbedaan ukuran panjang dari pergelangan tangan hingga ujung jari tengah subyek, pemasangan tempat sensor yang kurang tepat, serta kurang banyak dan kurangnya variasi data latih/sampel yang digunakan sebagai pembuatan *file* yang menyimpan data latih membuat akurasi pengujian *real-time* program ini tidak 100% berhasil memprediksi gestur yang diperagakan. Pengujian ini memiliki akurasi tertinggi pada angka 96.7%, dan akurasi terendah pada angka 53.3%. Akurasi keseluruhan pengujian didapatkan pada angka 80.67%. Confusion matrix pada rangkuman seluruh pengujian yang telah dilakukan dapat dilihat pada tabel 4.6.

Tabel 4.6 Confusion matrix pengujial real time (1 variasi)

Predicted \ Actual	1	2	3	4	5	6	7	8	9	10
1	24	3	1	0	0	0	1	0	0	0
2	1	23	4	0	0	0	1	0	0	0
3	0	1	24	0	0	0	0	0	0	0
4	0	0	0	27	3	1	0	0	0	0
5	0	1	0	3	21	0	5	3	0	0
6	2	2	0	0	6	24	0	0	0	0
7	0	0	1	0	0	2	18	0	0	0
8	1	0	0	0	0	3	4	20	0	0
9	0	0	0	0	0	0	1	7	30	0
10	2	0	0	0	0	0	0	0	0	30

Perhitungan TP, FN, FP, dan TN pada multiclass berbeda dengan confusion matrix pada umumnya, untuk mencari nilai-nilai tersebut harus mengacu pada salah satu class. Contohnya berdasarkan class gestur 1, nilai TP, FN, FP, dan TN adalah sebagai berikut:

TP = 24

FN = (1 + 0 + 0 + 0 + 2 + 0 + 1 + 0 + 2) = 6

FP = (3 + 1 + 0 + 0 + 0 + 1 + 0 + 0 + 0) = 5

TN = (23 + 4 + 1 + 1 + 24 + 27 + 3 + 1 + 1 + 3 + 21 + 5 + 3 + 2 + 6 + 24 + 1 + 2 + 18 + 3 + 4 + 20 + 1 + 7 + 30 + 30) = 265

Sehingga bisa didapatkan nilai dari *accuracy* (berdasarkan confusion matrix), *recall*, *precision*, dan f_1-*score* terhadap class gestur 1 (1 variasi). Nilai-nilai tersebut dapat dilihat pada perhitungan dibawah.

$$Accuracy = \frac{24+265}{24+265+6+5} = 0.963$$

$$Precision = \frac{24}{24+5} = 0.828$$

$$Recall = \frac{24}{24+6} = 0.8$$

$$f1 - score = \frac{2x24}{2x24+5+6} = 0.813$$

Hasil rangkuman seluruh pengujian *real-time* yang dilakukan pada semua subyek dapat dilihat pada tabel 4.7.

Tabel 4.7 Rangkuman hasil pengujian seluruh subyek (1 variasi)

Subyek	Klasifikasi		Akurasi
	Benar	Salah	
1	26	4	86.67%
2	16	14	53.33%
3	25	5	83.33%
4	23	7	76.67%
5	26	4	86.67%
6	28	2	93.33%
7	22	8	73.33%
8	20	10	66.67%
9	25	5	83.33%
10	29	1	96.67%
Rata-rata	24	6	80.00%

Berdasarkan data tabel diatas, dapat dilihat bahwa hasil akurasi berbeda-beda berdasarkan panjang dari pergelangan tangan hingga ujung jari tengah. Perbedaan akurasi ini disebabkan oleh beberapa faktor yang ditemui selama pengujian, yaitu jumlah sampel yang kurang banyak, data latih yang digunakan hanya mengambil dari 1 subyek, beda panjang pergelangan tangan hingga jari tengah, beda letak tekukan *metacarpo phalangeal*, dan *proximal phalangeal*, lokasi sensor yang tidak dapat dirubah, serta pemasangan sensor yang sangat menempel dan mengakibatkan sensor tertekuk. Rangkuman hasil pengujian yang menunjukkan akurasi tiap gestur dapat dilihat pada tabel 4.7. Hasil pengujian ini menggunakan data dari data pengujian *real-time* sebelumnya dengan cara mengelompokkan setiap gestur menjadi satu, dan menghitung akurasi masing-masing gestur.

Tabel 4.8 Hasil pengujian tiap gestur (1 variasi)

Gestur	Jumlah Pengujian	Benar	Salah	Akurasi
1	30	24	6	80%
2	30	23	7	76.67%
3	30	24	6	80%
4	30	27	3	90%
5	30	21	9	70%
6	30	24	6	80%
7	30	18	12	60%
8	30	20	10	66.67%
9	30	30	0	100%
10	30	30	0	100%

4.3.1.2 Hasil Pengujian Data Uji dengan 5 Variasi Sampel Latih

Pada pengujian ini, normalisasi yang dilakukan menggunakan MinMaxScaler, berbeda dengan pengujian sebelumnya yang menggunakan StandardScaler, hal ini dilakukan karena MinMaxScaler dinilai lebih cocok digunakan dengan tipe data yang dimiliki. Seperti pada pengujian sebelumnya, hal yang pertama dilakukan adalah mendapatkan file dengan nilai k yang memiliki akurasi tertinggi.

A. Pemilihan *File* Data Train dan Nilai k

Pada pembuatan *file* dengan 5 variasi sampel ini, membandingkan akurasi terhadap nilai k pada angka 3 sampai 17. Hasil menunjukkan pada saat k bernilai 3 sampai k bernilai 13, akurasi yang telah didapatkan sebesar 100%, sedangkan pada saat k bernilai 15 dan 17 akurasi turun menjadi 90%. Maka dari itu sebagai *file* yang

digunakan untuk pengujian real time akan menggunakan *file* dengan nilai k=3. Tabel Pengujian tiap *file* dapat dilihat pada tabel 4.9.

Tabel 4.9 Hasil Pengujian KNN dengan parameter k (5 variasi)

No	k	Akurasi
1	3	100%
2	5	100%
3	7	100%
4	9	100%
5	11	100%
6	13	100%
7	15	90%
8	17	90%

B. Hasil Pengujian *Real Time*

Seperti pada pengujian sebelumnya, pengujian *real-time* ini dilakukan sebanyak 3 kali pada setiap gestur, namun hanya 5 subyek yang akan diuji dalam pengujian ini. Dari hasil yang didapatkan, pengujian dengan 5 variasi ini memiliki akurasi yang jauh lebih baik daripada pengujian sebelumnya. Tabel hasil Confusion matrix terhadap pengujian dengan 5 variasi dapat dilihat pada tabel 4.10.

Tabel 4.10 Confusion matrix pengujian real time (5 variasi)

Predicted \ Actual	1	2	3	4	5	6	7	8	9	10
1	15	1	0	0	0	0	0	0	0	0
2	0	14	2	0	0	0	0	0	0	0
3	0	0	13	0	0	0	0	0	0	0
4	0	0	0	14	4	1	0	1	0	0
5	0	0	0	0	11	0	0	0	0	0
6	0	0	0	0	0	14	0	0	0	0
7	0	0	0	0	0	0	14	0	0	0

8	0	0	0	1	0	0	1	14	0	0
9	0	0	0	0	0	0	0	0	15	0
10	0	0	0	0	0	0	0	0	0	15

Nilai dari TP, FP, FN, dan TN berdasarkan gestur angka 1 dengan pengujian kedua yang menggunakan 5 variasi sampel adalah sebagai berikut:

TP = 15

FN = 0

FP = 1

TN = (14 + 2 + 13 + 14 + 4 + 1 + 1 + 11 + 14 + 14 + 1 + 1 + 14 + 15 + 15) = 134

Nilai dari *accuracy* (berdasarkan confusion matrix), *recall*, *precision*, dan f_1-*score* terhadap class gestur 1 (5 variasi). Nilai-nilai tersebut dapat dilihat pada perhitungan dibawah.

$$Accuracy = \frac{15+134}{15+1+0+134} = 0.993$$

$$Precision = \frac{15}{15+1} = 0.9375$$

$$Recall = \frac{15}{15+0} = 1$$

$$f1 - score = \frac{2x15}{2x15+1+0} = 0.967$$

Hasil rangkuman seluruh pengujian *real-time* yang dilakukan pada semua subyek dapat dilihat pada tabel 4.11.

Tabel 4.11 Rangkuman hasil pengujian seluruh subyek (5 variasi)

Subyek	Klasifikasi		Akurasi
	Benar	Salah	
1	27	3	90%
2	28	2	93.33%
3	28	2	93.33%
4	27	3	90%

5	29	1	96.67%
Rata-rata	27.8	2.2	92.67%

Dari hasil pengujian tersebut, didapatkan rangkuman hasil pengujian terhadap tiap gestur yang diuji. Rangkuman tersebut dapat dilihat pada tabel 4.12.

Tabel 4.12 Hasil pengujian tiap gestur (5 variasi)

Gestur	Jumlah Pengujian	Benar	Salah	Akurasi
1	15	15	0	100%
2	15	14	1	93.33%
3	15	13	2	86.67%
4	15	14	1	93.33%
5	15	11	5	73.33%
6	15	14	1	93.33%
7	15	14	1	93.33%
8	15	14	1	93.33%
9	15	15	0	100%
10	15	15	0	100%

4.4 Diskusi Hasil

Berdasarkan hasil pengujian secara *real-time* yang telah dilakukan, klasifikasi gestur menggunakan metode dan *file* yang menyimpan data latih KNN dengan 1 varasi dan parameter k bernilai 5 menghasilkan akurasi yang cukup baik yaitu dengan rata-rata sebesar 80%, nilai terendah sebesar 53.3%, dan nilai tertinggi sebesar 96.7%, sedangkan pengujian *file* yang menyimpan data latih KNN dengan 5 variasi dan parameter k bernilai 3 menghasilkan akurasi yang lebih baik yaitu dengan rata-rata sebesar 92.67% dengan nilai terendah sebesar 90%, dan nilai tertinggi sebesar 96.67%.

Dari hasil yang telah ditunjukkan, masih terdapat cukup banyak kesalahan prediksi pada beberapa gestur, hal ini disebabkan oleh perbedaan panjang dari

pergelangan tangan hingga ujung jari tengah, selain itu yang menyebabkan terjadinya kesalahan prediksi adalah letak tekukan *metacarpo phalangeal* dan *proximal phalangeal* pada subyek yang berbeda-beda, ada juga faktor kemampuan subyek melakukan tekukan yang tidak sesuai dengan gestur pada SIBI, meskipun akurasi yang telah didapatkan pada pengujian *real-time* sudah cukup baik.

Bab V
Penutup

5.1. Kesimpulan

Berdasarkan hasil pengujian secara *real-time* yang telah dilakukan, tujuan yang perlu dicapai dalam penelitian ini telah berhasil dilakukan yaitu dengan membuat sebuah program aplikasi yang dapat mengklasifikasi gestur tangan dengan menggunakan metode KNN dan mengujinya secara *real-time*. Dalam pengujian ini telah didapatkan akurasi rata-rata sebesar 80% dengan akurasi pengujian terhadap 10 subyek memiliki akurasi terbesar bernilai 96.7% dan terendah bernilai 53.3% pada saat k bernilai 5 dan hanya menggunakan 1 variasi sampel. Sedangkan pengujian dengan 5 variasi sampel didapatkan akurasi yang jauh lebih baik yaitu rata-rata 92.67% dengan akurasi tertinggi sebesar 96.67% dan akurasi terendah sebesar 90%.

Berdasarkan data dari seluruh subyek yang diuji, ditemukan bahwa perbedaan akurasi pada tiap subyek disebabkan oleh :

1. Perbedaan panjang dari telapak tangan hingga ujung jari tengah,

   Dengan panjang pergelangan tangan hingga ujung jari tengah yang berbeda-beda tentu saja akan menghasilkan bentuk gestur yang berbeda pula, hal ini mengurangi akurasi yang dihasilkan oleh program,

2. Jumlah sampel yang sedikit,

   Dari seluruh gestur yang diuji yaitu angka 1-10, peneliti hanya menggunakan total 150 data dengan 120 data sebagai data latih dan 30 data sebagai data uji.

3. Lokasi sensor yang tidak dapat diubah,

   Setiap orang tidak memiliki tekukan *metacarpo phalangeal,* dan *proximal phalangeal* jari yang sama, hal ini tentu dapat mengurangi maksimalnya akurasi karena sensor yang sudah dipasangkan pada sarung tangan tidak dapat dipindah/digeser.

4. Kemampuan subyek melakukan gestur sesuai petunjuk.

Dari beberapa subyek yang diuji, ada beberapa subyek yang tidak mampu menekuk jari secara maksimal ketika melakukan salah satu gestur yang akan diuji, hal ini juga menimbulkan kurang maksimalnya akurasi yang didapatkan.

5.2. Saran

Berdasarkan penelitian serta pengujian yang telah dilakukan dengan program yang telah dibuat, berikut merupakan beberapa saran yang dapat dibuat dan dikembangkan untuk memperbaiki sistem ataupun melanjutkan penelitian berikutnya :

1. Pengembangan atau perbaikan yang dapat dilakukan pada penelitian ini dapat dilakukan dengan contoh menambahkan jumlah sampel, serta memperbanyak variasi yang akan digunakan sebagai model pengujian *real-time*, sehingga akan meningkatkan akurasi menjadi lebih maksimal.
2. Melakukan perbandingan klasifikasi dengan menggunakan metode *machine learning* yang lain misalnya Random Forest Classification, Decision Tree, ataupun metode yang lainnya.
3. Melanjutkan penelitian dengan tambahan dapat mengklasifikasikan tidak hanya angka SIBI 1-10, namun juga huruf A-Z, atau hingga klasifikasi kata-kata dalam kamus SIBI.

DAFTAR PUSTAKA

Brownlee, Jason. 2018, *Indoor Movement Time Series Classification with Machine Learning Algorithms,* diakses pada 20 Mei 2021, <https://machinelearningmastery.com/indoor-movement-time-series-classification-with-machine-learning-algorithms/>.

Brownlee, Jason, 2020**,** *How to Calculate Precision, Recall, and F-Measure for Imbalanced Classification,* diakses pada 20 Juni 2021, <https://machinelearningmastery.com/precision-recall-and-f-measure-for-imbalanced-classification/>.

Flex Point Sensor System Inc, 2016, *Bend Sensor USB Glove Kit User Guide,* Utah, Tersedia dari www.flexpoint.com.

Immersa Lab. 2018, *Pengertian Accelerometer dan Cara Kerjanya,* diakses pada 17 Februari 2021. <https://www.immersa-lab.com/pengertian-accelerometer-dan-cara-kerjanya.htm>.

Kamus Sistem Isyarat Bahasa Indonesia & Bahasa Murni, 2016.

Muhardian, Ahmad, 2020, *Apa itu Regex? dan Apa Manfaatnya dalam Pemrograman?*, diakses pada 24 November 2021, < https://www.petanikode.com/regex/>

Scikit-learn: *Machine Learning in Python*, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.

Singh. Aishwarya, 2018, *A Multivariate Time Series Guide to Forecasting and Modeling (with Python codes),* diakses pada 20 Juni 2021, <https://www.analyticsvidhya.com/blog/2018/09/multivariate-time-series-guide-forecasting-modeling-python-codes/>.

Trixie, Devina, 2021, *Rancang Bangun Klasifikasi Pendeteksian Jenis Klik Pada Pointing Device Menggunakan Electromyograph,* Malang: Universitas Ma Chung Malang.

Van Rossum, G., 2020, *The Python Library Reference, release 3.8.2*, Python Software Foundation.

Widodo, Romy Budhi, Windra Swastika, dan Agustinus Bohaswara Haryasena, 2020, *Studi Sensor dan Akuisisi Data Hand Gesture dengan Sarung Tangan,* Malang: Universitas Widyagama Malang.

Yayasan Peduli Kasih ABK, 2018, *Mengenal Bahasa Isyarat*, diakses pada 20 Februari 2021, <https://www.ypedulikasihabk.org/2018/11/09/mengenal-bahasa-isyarat/>.

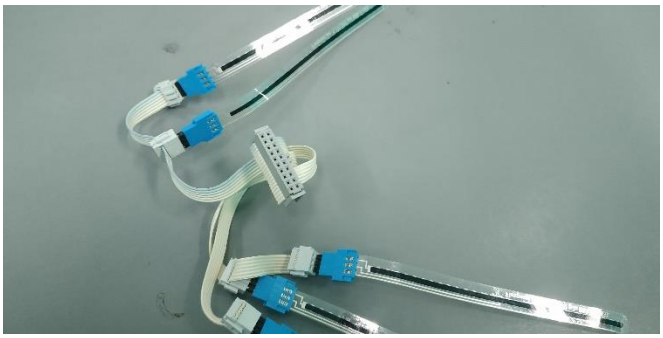
LAMPIRAN

BUKU MANUAL PENGGUNAAN PROGRAM KLASIFIKASI MENGGGUNAKAN FLEXPOINT GLOVE KIT

FLEXPOINT GLOVE KIT INFORMATION

1. Alat/Device yang Digunakan

a. Sensor Tekuk



b. Sarung Tangan

c. Mikrokontroller KMX62

2. Pustaka Python yang Diperlukan

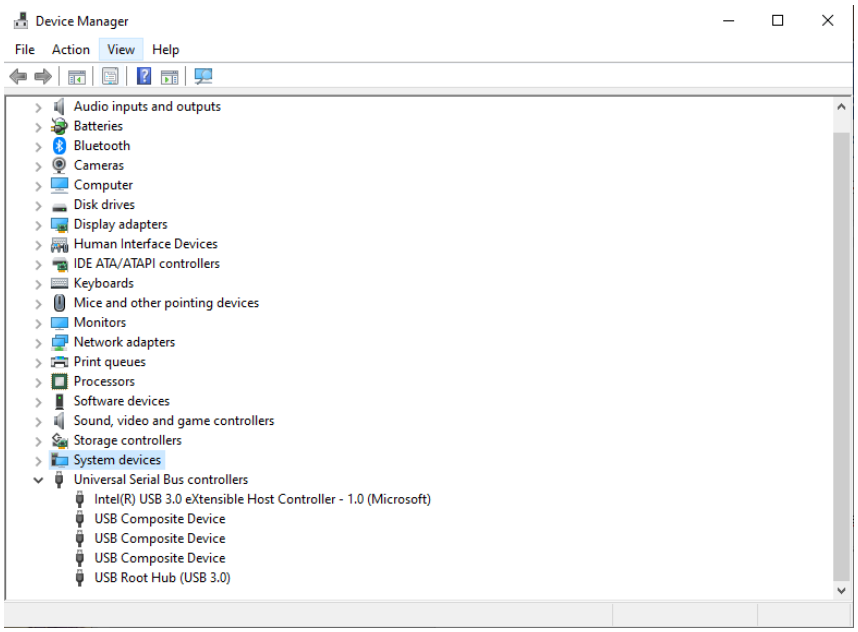
a. Tkinter
b. PIL
c. Numpy
d. Matplotlib
e. Scikit Learn
f. Pyserial
g. Csv
h. Keyboard

3. Melihat Port Serial Sensor pada PC Windows

a. Pastikan alat sudah terpasang pada salah satu port USB di PC/Laptop
b. Tekan Windows + X pada PC/Laptop anda hingga muncul seperti ini

c. Klik pada Device Manager

d. Port Serial akan terlihat pada Universal Serial Bus controller



4. Pengambilan Data

a. Pengambilan data baru dilakukan dengan membuka program python dengan nama rekam_databaru.py,

b. Pastikan Glovekit sudah terpasang pada salah satu port USB pada PC/Laptop,

c. Buka file rekam_databaru.py menggunakan text editor seperti Sublime, Notepad+, dan lain-lain,

d. Cek pada baris berikut, dan cocokkan Port Serial yang terpasang pada PC/Laptop (COM3),

```
41   ser = serial.Serial('COM3', 9600, timeout=1)     #Parameter : Port Serial #Baudrate
```

e. Jalankan program rekam_databaru.py melalui command line/IDLE python,

f. Beri nama file, kemudian tekan Enter,

g. Lakukan Gestur, kemudian tekan Enter,

h. File .csv akan berada 1 folder yang sama dengan program ini.

5. *Split* Data latih dan Data Uji

a. Data latih dan data uji yang akan digunakan akan diletakkan pada 1 folder yang sama yaitu Sign > dataset_number,
b. Dalam folder dataset_number, akan dibuat sebuah file csv dengan nama targets.csv untuk memberikan kelas pada masing-masing data. Contoh : data ke-101 merupakan kelas gestur angka 1, dan seterusnya,

101	1
102	1
103	1
104	1
105	1
106	2
107	2
108	2
109	2
110	2
111	3
112	3
113	3
114	3
115	3
116	4
117	4
118	4
119	4
120	4
121	5
122	5
123	5
124	5
125	5

c. *Split* data latih dan data uji dimulai dengan membuat folder groups_number,
d. Isi dari folder tersebut adalah sebuah file .csv yang akan membagi mana saja yang merupakan data latih (diindikasikan dengan angka 1), dan mana saja yang merupakan data uji (diindikasikan dengan angka 2),

101	1
102	1
103	2
104	2
105	2
106	1
107	1
108	2
109	2
110	2
111	1
112	1
113	2
114	2
115	2
116	1
117	1
118	2
119	2
120	2
121	1
122	1
123	2
124	2
125	2
126	1
127	1
128	2
129	2
130	2

e. Proses pemisahan akan menggunakan program bernama movement.py, karena data merupakan *multivariate time series,* seluruh data mentah yang tadinya berupa 100 baris dan 10 kolom, akan digabungkan dan dijadikan 1 baris dengan 1000 kolom. 120 data akan dijadikan data latih dengan nama es2_train.csv yang berisi 120 baris kelas, dan 1000 kolom data,

f. Data uji yang berjumlah 30 akan dijadikan satu menjadi file es2_test.csv yang berisi 30 baris kelas, dan 1000 kolom data.

6. Pembuatan Model KNN

a. Pembuatan model KNN menggunakan program bernama movement_train_es2.py,

b. Pastikan langkah-langkah sebelumnya sudah dilakukan (pengambilan data, split dan transformasi data),

c. Pengujian akurasi terbaik dengan mengubah nilai parameter k dapat dilihat pada gambar dibawah.

```
models.append(KNeighborsClassifier(n_neighbors=3))
names.append('KNN3')
models.append(KNeighborsClassifier(n_neighbors=5))
names.append('KNN5')
models.append(KNeighborsClassifier(n_neighbors=7))
names.append('KNN7')
models.append(KNeighborsClassifier(n_neighbors=9))
names.append('KNN9')
models.append(KNeighborsClassifier(n_neighbors=11))
names.append('KNN11')
```

7. Pengujian Real-Time

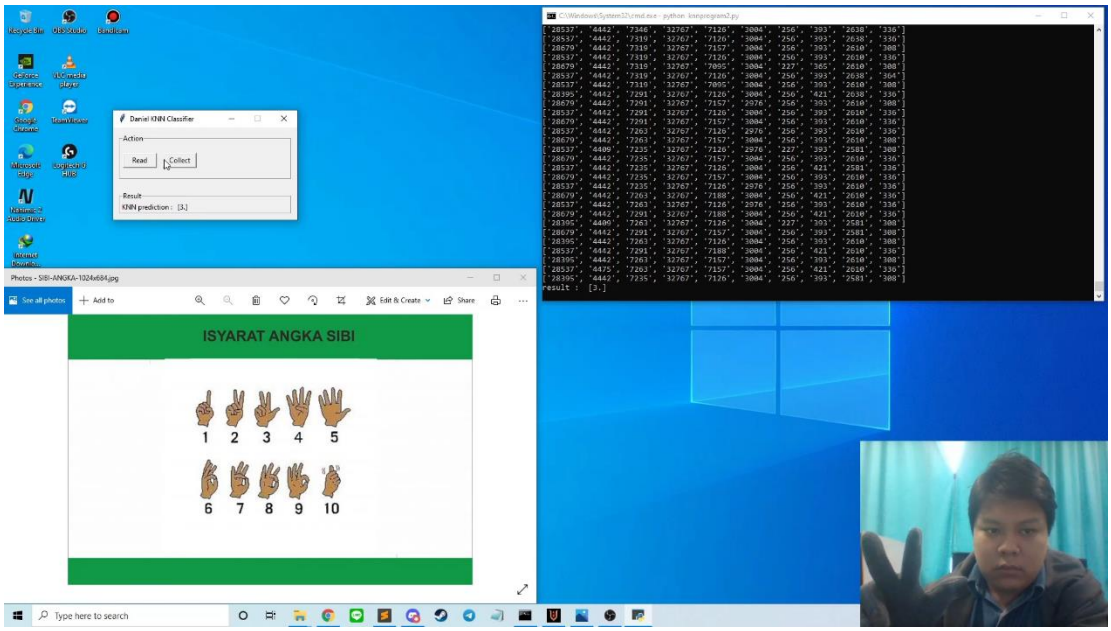
a. Setelah semua langkah-langkah telah dilakukan, dapat dilakukan pengujian *real-time*, pastikan alat sudah terpasang pada port USB PC/Komputer,

b. Jalankan program knnprogram.py,

c. 2 tombol utama pada tampilan Tkinter berupa Read, dan Classify,

d. Tombol Read berfungsi untuk mulai membaca sensor secara *real-time* terus-menerus,

e. Tombol Classify berfungsi untuk mengklasifikasikan gestur apa yang sedang diperagakan.


Daniel KNN Classifier
Action
Read
Collect
Result
KNN prediction : [3.]
Photos - SIBI-ANGKA-1024x684.jpg
ISYARAT ANGKA SIBI
1 2 3 4 5
6 7 8 9 10
result : [3.]

KODE UTAMA UNTUK KLASIFIKASI MENGGUNAKAN FLEXPOINT GLOVE KIT

1. **Pengambilan/Perekaman Data**

   a. **Import pustaka yang diperlukan**

```
import serial
import time
import csv
import re
```

   b. ***Setting* Serial Port yang digunakan**

```
ser = serial.Serial('COM3', 9600, timeout=1)
```

   c. **Pengambilan data**

```
while True:
      print('Enter Gesture Name : ')
      gesture_name = input()
      input('Press Enter to Start Recording..')
      record_data(0)
      ser.flushInput()
      ser.flushOutput()
      ser.flush()
```

   d. **Kode Lengkap**

```
import serial
import time
import csv
import re

def bacadata():
      data = ser.readline()
      return data

def record_data(index):
      if (index > 100) :
            print('Record End')
            return
      catched = bacadata()
      catched = re.findall(rb'(?<=:)(.*?)(?=])', catched)
      catched = [x.decode() for x in catched][:-7]
```

```
      if (index > 0) :
            with open(gesture_name + '.csv', 'a',
encoding='iso-8859-1', newline='\n') as csvfile:
                  writer            = csv.writer(csvfile)
                  writer.writerow(catched)
      index+=1
      print(catched)


      return record_data(index)

ser = serial.Serial('COM3', 9600, timeout=1)    #Parameter :
Port Serial #Baudrate

while True:
      print('Enter Gesture Name : ')
      gesture_name = input()
      input('Press Enter to Start Recording..')
      record_data(0)
      ser.flushInput()
      ser.flushOutput()
      ser.flush()
```

2. Split dan Transformasi *Data Train*, serta *Data Test*

a. Kode Lengkap

```
# load user movement dataset into memory
from pandas import read_csv
from os import listdir
import os
from numpy import vstack
from numpy import array
from numpy import savetxt
from matplotlib import pyplot

# return list of traces, and arrays for targets, groups and
paths
```

```
def load_dataset(prefix=''):
	grps_dir, data_dir = prefix+'groups_number/',
prefix+'dataset_number/'
	# load mapping files
	targets = read_csv(data_dir + 'targets.csv', header=0)
	groups = read_csv(grps_dir + 'groups.csv', header=0)
	# load traces
	sequences = list()
	target_mapping = None
	lstdir = listdir(data_dir)
	for name in sorted(lstdir):
		filename = data_dir + name
		if name == 'targets.csv':
			continue
		df = read_csv(filename, header=None)
		values = df.values
		sequences.append(values)
	return sequences, targets.values[:,1],
groups.values[:,1]

def create_dataset(sequences, targets):
	# create the transformed dataset
	transformed = list()
	n_vars = 10
	n_steps = 100
	# process each trace in turn
	for i in range(len(sequences)):
		seq = sequences[i]
		vector = list()
		# last n observations
		for row in range(1, n_steps+1):
			for col in range(n_vars):
				vector.append(seq[-row, col])
		# add output
		vector.append(targets[i])
		# store
```

```
            transformed.append(vector)
        # prepare array
        transformed = array(transformed)
        transformed = transformed.astype('float32')
        return transformed

# load dataset
sequences, targets, groups = load_dataset('Sign/')
# separate traces
seq1 = [sequences[i] for i in range(len(groups)) if
groups[i]==1]
seq2 = [sequences[i] for i in range(len(groups)) if
groups[i]==2]
# separate target
targets1 = [targets[i] for i in range(len(groups)) if
groups[i]==1]
targets2 = [targets[i] for i in range(len(groups)) if
groups[i]==2]
# create ES1 dataset
es1 = create_dataset(seq1, targets1)
print('ES1: %s' % str(es1.shape))
savetxt('es1.csv', es1, delimiter=',')
# create ES2 dataset
es2_train = create_dataset(seq1, targets1)
es2_test = create_dataset(seq2, targets2)
print('ES2 Train: %s' % str(es2_train.shape))
print('ES2 Test: %s' % str(es2_test.shape))
savetxt('es2_train.csv', es2_train, delimiter=',')
savetxt('es2_test.csv', es2_test, delimiter=',')
```

3. Pembuatan Model Klasifikasi KNN

a. Import pustaka yang diperlukan

```
from numpy import mean
from numpy import std
from pandas import read_csv
from matplotlib import pyplot
```

```
from sklearn.model_selection import cross_val_score
from sklearn.metrics import accuracy_score
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
import joblib
```

b. Pengujian terhadap model dengan parameter k yang berbeda-beda

```
models.append(KNeighborsClassifier(n_neighbors=3))
names.append('KNN3')
models.append(KNeighborsClassifier(n_neighbors=5))
names.append('KNN5')
models.append(KNeighborsClassifier(n_neighbors=7))
names.append('KNN7')
models.append(KNeighborsClassifier(n_neighbors=9))
names.append('KNN9')
models.append(KNeighborsClassifier(n_neighbors=11))
names.append('KNN11')
```

c. Pembuatan model KNN

```
for i in range(len(models)):
      scaler = StandardScaler()
      model = Pipeline(steps=[('s',scaler), ('m',models[i])])
      model.fit(trainX, trainy)

      joblib.dump(model, 'model_' + names[i] + '.pkl')

      yhat = model.predict(testX)
      score = accuracy_score(testy, yhat) * 100
      all_scores.append(score)

      print('%s %.3f%%' % (names[i], score))
```

d. Kode lengkap

```
# spot check for ES1
from numpy import mean
from numpy import std
from pandas import read_csv
from matplotlib import pyplot
```

```
from sklearn.model_selection import cross_val_score
from sklearn.metrics import accuracy_score
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
import joblib
# load dataset
train = read_csv('es2_train.csv', header=None)
test = read_csv('es2_test.csv', header=None)
# split into inputs and outputs
trainX, trainy = train.values[:, :-1], train.values[:, -1]
testX, testy = test.values[:, :-1], test.values[:, -1]
# create a list of models to evaluate
models, names = list(), list()

models.append(KNeighborsClassifier(n_neighbors=3))
names.append('KNN3')
models.append(KNeighborsClassifier(n_neighbors=5))
names.append('KNN5')
models.append(KNeighborsClassifier(n_neighbors=7))
names.append('KNN7')
models.append(KNeighborsClassifier(n_neighbors=9))
names.append('KNN9')
models.append(KNeighborsClassifier(n_neighbors=11))
names.append('KNN11')


# evaluate models
all_scores = list()

for i in range(len(models)):
      scaler = StandardScaler()
      model = Pipeline(steps=[('s',scaler), ('m',models[i])])
      model.fit(trainX, trainy)

      joblib.dump(model, 'model_' + names[i] + '.pkl')
```

```
        yhat = model.predict(testX)
        score = accuracy_score(testy, yhat) * 100
        all_scores.append(score)

        print('%s %.3f%%' % (names[i], score))
# plot
pyplot.boxplot(all_scores, labels=names)
pyplot.show()
```

4. Pengujian secara *real-time*

a. Kode Lengkap

```
import serial
import csv
import re
import keyboard
from tkinter import *
from tkinter import ttk
from tkinter import filedialog
import joblib
import numpy as np
from numpy import array

def transform_data(sequences):
        # create the transformed dataset
        transformed = list()
        n_vars = 10
        n_steps = 100
        # process each trace in turn
        for i in range(len(sequences)):
                seq = sequences[i]
                vector = list()
                # last n observations
                for row in range(1, n_steps+1):
                        for col in range(n_vars):
                                vector.append(seq[-row, col])
```

```
            # store
            transformed.append(vector)
      # prepare array
      transformed = array(transformed)
      transformed = transformed.astype('float32')
      return transformed

def baca_data_alat():
      data = ser.readline()
      return data

def open_pdf():
      return filedialog.askopenfilename()



def new_tampil():
      global collect_click
      global knnpred

      data = baca_data_alat()
      data = re.findall(rb'(?<=:)(.*?)(?=])', data)
      data = [x.decode() for x in data][:-7]
      print(data)

      if (len(data_bucket) < 100):
            data_bucket.append(data)
      else:
            data_bucket.pop(0)
            data_bucket.append(data)
      handle = win.after(1,new_tampil)
      if collect_click:
            win.after_cancel(handle)
            win.after_cancel(handle) #stop loop
            trans_data =
transform_data([np.array(data_bucket)]) #transform data
            model = joblib.load('model_KNN.pkl') #load model
```

```
            prediction = model.predict(trans_data) #prediksi
transformed data
            data_bucket.clear()
            collect_click = False

            knnpred = prediction
            print('result : ',prediction)
            label_result1.configure(text=knnpred)


def collectData():
      global collect_click
      if not collect_click:
            collect_click = True
      ser.flush()

ser = serial.Serial(port='COM3', baudrate=9600, timeout=1)

win = Tk()

knnpred = StringVar()
knnpred = ''

data_bucket = []

wrapper_action = LabelFrame(win, text="Action")
wrapper_action.pack(fill="both", expand="yes", padx=10,
pady=10)

#TOMBOL READ
button_read = Button(wrapper_action, text="Read",
command=new_tampil, anchor="n", width=7)
button_read.pack(side=LEFT, fill="x", padx=5, pady=5)

#TOMBOL CLASSIFY
collect_click = False #inisialisasi collect_click
```

```
button_collect = Button(wrapper_action, text="Classify",
command=collectData, anchor="n", width=7)
button_collect.pack(side=LEFT, fill="x", padx=5, pady=5)

wrapper_result = LabelFrame(win, text="Result")
wrapper_result.pack(side=LEFT, fill="both", expand="yes",
padx=10, pady=10)

#TAMPILAN HASIL KLASIFIKASI
label_knn = Label(wrapper_result, text='KNN prediction : ')
label_result1 = Label(wrapper_result, text=knnpred)
label_knn.pack(side=LEFT)
label_result1.pack(side=LEFT)

win.geometry("320x160")
win.resizable(False, False)
win.title("Daniel KNN Classifier")
win.mainloop()
```

TABEL HASIL PENGUJIAN
REAL-TIME 1 VARIASI SAMPEL

Tabel Pengujian *Real-Time* Subyek 1

No.	Gestur	Uji ke-x	Hasil	Salah	Benar
1	1	1	1		V
2	1	2	10	V	
3	1	3	1		V
4	2	1	2		V
5	2	2	2		V
6	2	3	2		V
7	3	1	3		V
8	3	2	3		V
9	3	3	3		V
10	4	1	4		V
11	4	2	4		V
12	4	3	4	V	
13	5	1	4	V	
14	5	2	4		V
15	5	3	5		V
16	6	1	6		V
17	6	2	6		V
18	6	3	6		V
19	7	1	7		V
20	7	2	7		V
21	7	3	7		V
22	8	1	9	V	
23	8	2	8		V
24	8	3	8		V
25	9	1	9		V
26	9	2	9		V
27	9	3	9		V
28	10	1	10		V
29	10	2	10		V
30	10	3	10		V
Panjang tangan 19.4 cm	Benar = 26	Salah = 4	Akurasi = 86.7%		

Tabel Pengujian *Real-Time* Subyek 2

No.	Gestur	Uji ke-x	Hasil	Salah	Benar
1	1	1	8	V	
2	1	2	6	V	
3	1	3	6	V	
4	2	1	6	V	
5	2	2	5	V	
6	2	3	6	V	
7	3	1	3		V
8	3	2	3		V
9	3	3	3		V
10	4	1	5	V	
11	4	2	5	V	
12	4	3	5	V	
13	5	1	5		V
14	5	2	5		V
15	5	3	5		V
16	6	1	6		V
17	6	2	6		V
18	6	3	6		V
19	7	1	5	V	
20	7	2	5	V	
21	7	3	5	V	
22	8	1	5	V	
23	8	2	8		V
24	8	3	5	V	
25	9	1	9		V
26	9	2	9		V
27	9	3	9		V
28	10	1	10		V
29	10	2	10		V
30	10	3	10		V
Panjang tangan = 15.3 cm	Benar = 16	Salah = 16	Akurasi = 53.3%		

Tabel Pengujian *Real-Time* Subyek 3

No.	Gestur	Uji ke-x	Hasil	Salah	Benar
1	1	1	1		V
2	1	2	1		V
3	1	3	1		V
4	2	1	2		V
5	2	2	3	V	
6	2	3	2		V
7	3	1	3		V
8	3	2	3		V
9	3	3	3		V
10	4	1	4		V
11	4	2	4		V
12	4	3	4		V
13	5	1	5		V
14	5	2	5		V
15	5	3	5		V
16	6	1	7	V	
17	6	2	6		V
18	6	3	7	V	
19	7	1	5	V	
20	7	2	7		V
21	7	3	1	V	
22	8	1	8		V
23	8	2	8		V
24	8	3	8		V
25	9	1	9		V
26	9	2	9		V
27	9	3	9		V
28	10	1	10		V
29	10	2	10		V
30	10	3	10		V
Panjang tangan = 17 cm	Benar = 25	Salah = 5	Akurasi = 83.3%		

Tabel Pengujian *Real-Time* Subyek 4

No.	Gestur	Uji ke-x	Hasil	Salah	Benar
1	1	1	1		V
2	1	2	10	V	
3	1	3	2	V	
4	2	1	2		V
5	2	2	2		V
6	2	3	2		V
7	3	1	3		V
8	3	2	2	V	
9	3	3	3		V
10	4	1	4		V
11	4	2	4		V
12	4	3	4		V
13	5	1	5		V
14	5	2	5		V
15	5	3	5		V
16	6	1	6		V
17	6	2	6		V
18	6	3	6		V
19	7	1	8	V	
20	7	2	5	V	
21	7	3	7		V
22	8	1	9	V	
23	8	2	9	V	
24	8	3	8		V
25	9	1	9		V
26	9	2	9		V
27	9	3	9		V
28	10	1	10		V
29	10	2	10		V
30	10	3	10		V
Panjang tangan = 17.2 cm	Benar = 23	Salah = 7	Akurasi = 76.7%		

Tabel Pengujian *Real-Time* Subyek 5

No.	Gestur	Uji ke-x	Hasil	Salah	Benar
1	1	1	1		V
2	1	2	1		V
3	1	3	1		V
4	2	1	2		V
5	2	2	2		V
6	2	3	2		V
7	3	1	3		V
8	3	2	3		V
9	3	3	3		V
10	4	1	4		V
11	4	2	4		V
12	4	3	4		V
13	5	1	5		V
14	5	2	5		V
15	5	3	4	V	
16	6	1	6		V
17	6	2	6		V
18	6	3	6		V
19	7	1	8	V	
20	7	2	8	V	
21	7	3	7		V
22	8	1	5	V	
23	8	2	8		V
24	8	3	8		V
25	9	1	9		V
26	9	2	9		V
27	9	3	9		V
28	10	1	10		V
29	10	2	10		V
30	10	3	10		V
Panjang tangan = 19 cm	Benar = 26	Salah = 4	Akurasi = 86.7%		

Tabel Pengujian *Real-Time* Subyek 6

No.	Gestur	Uji ke-x	Hasil	Salah	Benar
1	1	1	1		V
2	1	2	1		V
3	1	3	1		V
4	2	1	2		V
5	2	2	2		V
6	2	3	2		V
7	3	1	3		V
8	3	2	7	V	
9	3	3	3		V
10	4	1	4		V
11	4	2	4		V
12	4	3	4		V
13	5	1	5		V
14	5	2	5		V
15	5	3	5		V
16	6	1	4	V	
17	6	2	6		V
18	6	3	6		V
19	7	1	7		V
20	7	2	7		V
21	7	3	7		V
22	8	1	8		V
23	8	2	8		V
24	8	3	8		V
25	9	1	9		V
26	9	2	9		V
27	9	3	9		V
28	10	1	10		V
29	10	2	10		V
30	10	3	10		V
Panjang tangan = 18 cm	Benar = 28	Salah = 2	Akurasi = 93.3%		

Tabel Pengujian *Real-Time* Subyek 7

No.	Gestur	Uji ke-x	Hasil	Salah	Benar
1	1	1	1		V
2	1	2	1		V
3	1	3	1		V
4	2	1	2		V
5	2	2	2		V
6	2	3	2		V
7	3	1	2	V	
8	3	2	2	V	
9	3	3	1	V	
10	4	1	4		V
11	4	2	4		V
12	4	3	4		V
13	5	1	6	V	
14	5	2	6	V	
15	5	3	6	V	
16	6	1	6		V
17	6	2	6		V
18	6	3	6		V
19	7	1	7		V
20	7	2	9	V	
21	7	3	7		V
22	8	1	8		V
23	8	2	9	V	
24	8	3	8		V
25	9	1	9		V
26	9	2	9		V
27	9	3	9		V
28	10	1	10		V
29	10	2	10		V
30	10	3	10		V
Panjang tangan = 16.5 cm	Benar = 22	Salah = 8	Akurasi = 73.3%		

Tabel Pengujian *Real-Time* Subyek 8

No.	Gestur	Uji ke-x	Hasil	Salah	Benar
1	1	1	1		V
2	1	2	1		V
3	1	3	1		V
4	2	1	2		V
5	2	2	2		V
6	2	3	2		V
7	3	1	2	V	
8	3	2	3		V
9	3	3	3		V
10	4	1	4		V
11	4	2	4		V
12	4	3	4		V
13	5	1	6	V	
14	5	2	6	V	
15	5	3	6	V	
16	6	1	8	V	
17	6	2	6		V
18	6	3	8	V	
19	7	1	7		V
20	7	2	2	V	
21	7	3	7		V
22	8	1	9	V	
23	8	2	9	V	
24	8	3	9	V	
25	9	1	9		V
26	9	2	9		V
27	9	3	9		V
28	10	1	10		V
29	10	2	10		V
30	10	3	10		V
Panjang tangan = 19 cm	Benar = 20	Salah = 10	Akurasi = 66.7%		

Tabel Pengujian *Real-Time* Subyek 9

No.	Gestur	Uji ke-x	Hasil	Salah	Benar
1	1	1	1		V
2	1	2	1		V
3	1	3	1		V
4	2	1	1	V	
5	2	2	1	V	
6	2	3	1	V	
7	3	1	3		V
8	3	2	3		V
9	3	3	3		V
10	4	1	4		V
11	4	2	4		V
12	4	3	4		V
13	5	1	5		V
14	5	2	5		V
15	5	3	5		V
16	6	1	8	V	
17	6	2	6		V
18	6	3	6		V
19	7	1	7		V
20	7	2	7		V
21	7	3	8	V	
22	8	1	8		V
23	8	2	8		V
24	8	3	8		V
25	9	1	9		V
26	9	2	9		V
27	9	3	9		V
28	10	1	10		V
29	10	2	10		V
30	10	3	10		V
Panjang tangan = 18.5 cm	Benar = 25	Salah = 25	Akurasi = 83.3%		

Tabel Pengujian *Real-Time* Subyek 10

No.	Gestur	Uji ke-x	Hasil	Salah	Benar
1	1	1	1		V
2	1	2	1		V
3	1	3	1		V
4	2	1	2		V
5	2	2	2		V
6	2	3	2		V
7	3	1	3		V
8	3	2	3		V
9	3	3	3		V
10	4	1	4		V
11	4	2	4		V
12	4	3	5	V	
13	5	1	5		V
14	5	2	5		V
15	5	3	5		V
16	6	1	6		V
17	6	2	6		V
18	6	3	6		V
19	7	1	7		V
20	7	2	7		V
21	7	3	7		V
22	8	1	8		V
23	8	2	8		V
24	8	3	8		V
25	9	1	9		V
26	9	2	9		V
27	9	3	9		V
28	10	1	10		V
29	10	2	10		V
30	10	3	10		V
Panjang tangan = 16.8 cm	Benar = 29	Salah = 1	Akurasi = 96.7%		

TABEL HASIL PENGUJIAN
REAL-TIME 5 VARIASI SAMPEL

Tabel Pengujian *Real Time* Subyek 1

No.	Gestur	Uji ke-x	Hasil	Salah	Benar
1	1	1	1		V
2	1	2	1		V
3	1	3	1		V
4	2	1	2		V
5	2	2	2		V
6	2	3	2		V
7	3	1	2	V	
8	3	2	2	V	
9	3	3	3		V
10	4	1	4		V
11	4	2	4		V
12	4	3	4		V
13	5	1	5		V
14	5	2	5		V
15	5	3	5		V
16	6	1	6		V
17	6	2	6		V
18	6	3	6		V
19	7	1	8	V	
20	7	2	7		V
21	7	3	7		V
22	8	1	8		V
23	8	2	8		V
24	8	3	8		V
25	9	1	9		V
26	9	2	9		V
27	9	3	9		V
28	10	1	10		V
29	10	2	10		V
30	10	3	10		V
Panjang tangan = 19.4 cm	Benar = 27	Salah = 3	Akurasi = 90%		

Tabel Pengujian *Real Time* Subyek 2

No.	Gestur	Uji ke-x	Hasil	Salah	Benar
1	1	1	1		V
2	1	2	1		V
3	1	3	1		V
4	2	1	2		V
5	2	2	2		V
6	2	3	2		V
7	3	1	2		V
8	3	2	2		V
9	3	3	3		V
10	4	1	8	V	
11	4	2	4		V
12	4	3	4		V
13	5	1	5		V
14	5	2	5		V
15	5	3	5		V
16	6	1	4	V	
17	6	2	6		V
18	6	3	6		V
19	7	1	7		V
20	7	2	7		V
21	7	3	7		V
22	8	1	8		V
23	8	2	8		V
24	8	3	8		V
25	9	1	9		V
26	9	2	9		V
27	9	3	9		V
28	10	1	10		V
29	10	2	10		V
30	10	3	10		V
Panjang tangan = 18.5 cm	Benar = 28	Salah = 2	Akurasi = 93.33%		

Tabel Pengujian *Real Time* Subyek 3

No.	Gestur	Uji ke-x	Hasil	Salah	Benar
1	1	1	1		V
2	1	2	1		V
3	1	3	1		V
4	2	1	2		V
5	2	2	2		V
6	2	3	2		V
7	3	1	3		V
8	3	2	3		V
9	3	3	3		V
10	4	1	4		V
11	4	2	4		V
12	4	3	4		V
13	5	1	4	V	
14	5	2	4	V	
15	5	3	5		V
16	6	1	6		V
17	6	2	6		V
18	6	3	6		V
19	7	1	7		V
20	7	2	7		V
21	7	3	7		V
22	8	1	8		V
23	8	2	8		V
24	8	3	8		V
25	9	1	9		V
26	9	2	9		V
27	9	3	9		V
28	10	1	10		V
29	10	2	10		V
30	10	3	10		V
Panjang tangan = 18.2 cm	Benar = 28	Salah = 2	Akurasi = 93.33%		

Tabel Pengujian *Real Time* Subyek 4

No.	Gestur	Uji ke-x	Hasil	Salah	Benar
1	1	1	1		V
2	1	2	1		V
3	1	3	1		V
4	2	1	2		V
5	2	2	2		V
6	2	3	2		V
7	3	1	3		V
8	3	2	3		V
9	3	3	3		V
10	4	1	4		v
11	4	2	4		V
12	4	3	4		V
13	5	1	4	V	
14	5	2	4	V	
15	5	3	5		V
16	6	1	6		V
17	6	2	6		V
18	6	3	6		V
19	7	1	7		V
20	7	2	7		V
21	7	3	7		V
22	8	1	4	V	
23	8	2	8		V
24	8	3	8		V
25	9	1	9		V
26	9	2	9		V
27	9	3	9		V
28	10	1	10		V
29	10	2	10		V
30	10	3	10		V
Panjang tangan = 17 cm	Benar = 27	Salah = 3	Akurasi = 90%		

Tabel Pengujian *Real Time* Subyek 5

No.	Gestur	Uji ke-x	Hasil	Salah	Benar
1	1	1	1		V
2	1	2	1		V
3	1	3	1		V
4	2	1	1	V	
5	2	2	2		V
6	2	3	2		V
7	3	1	3		V
8	3	2	3		V
9	3	3	3		V
10	4	1	4		V
11	4	2	4		V
12	4	3	4		V
13	5	1	5		V
14	5	2	5		V
15	5	3	5		V
16	6	1	6		V
17	6	2	6		V
18	6	3	6		V
19	7	1	7		V
20	7	2	7		V
21	7	3	7		V
22	8	1	8		V
23	8	2	8		V
24	8	3	8		V
25	9	1	9		V
26	9	2	9		V
27	9	3	9		V
28	10	1	10		V
29	10	2	10		V
30	10	3	10		V
Panjang tangan = 18 cm	Benar = 29	Salah = 1	Akurasi = 96.67%		