International Journal of Innovative Research in Engineering
Volume 5, Issue 3 (May-June 2024), PP: 223-229.
https://www.doi.org/10.59256/ijire.20240503028
www.theijire.com
ISSN No: 2582-8746

An Air Gesture Based Keyboard for Visually Impaired People Using Machine Learning and Embedded System Approach

Hemashree M[1], Kavitha B C[2], Leena Ranganath[3], Suhas[4]
[1,2,3,4] *Dept. of ECE, BGS Institute of Technology, Adichunchanagiri University, B G Nagara, Karnataka, India.*

How to cite this paper:
Hemashree M[1], Kavitha B C[2], Leena Ranganath[3], Suhas[4]"An Air Gesture Based Keyboard for Visually Impaired People Using Machine Learning and Embedded System Approach", IJIRE-V5I03-223-229.

Abstract: *This report outlines the development of an air gesture keyboard utilizing machine learning technology, designed to assist visually impaired individuals in entering text more easily. This innovative system removes the need for physical keyboards, instead employing a remote-based system that translates hand gestures into text through the use of an accelerometer, Arduino, and switches. There is a significant need for alternative input methods for visually impaired individuals due to the limitations of existing technologies. The proposed air gesture keyboard captures and interprets hand motions in the air, enabling users to input text without physicalcontact. A literature review was conducted to examine related research on hand gesture recognition using various techniques.*

Key Word: *Machine learning, gesture, Microcontroller,visually impaired.*

I.INTRODUCTION

The keyboard has been an essential component of computer systems, allowing users to input data through key presses. While physical keyboards are still widely used, virtual keyboards have become popular with the rise of mobile devices. However, speech-to-text technology often lacks accuracy. To address this issue, remote-based gesturekeyboards have been developed, utilizing machine learning algorithms and Python programming.

These keyboards employ accelerometers to capture air gestures and convert them into text, removing the need for a specific layout and enabling multilingual functionality. The system comprises an accelerometer, an Arduino microcontroller, and switches, with the Arduino IDE configured to set the baud rate to 38400. The scikit-learn library is used to convert accelerometer signals into letters, which are stored in a dataset and trained using machinelearning algorithms to recognize gestures in various languages.

The remote-based gesture keyboard provides severaladvantages over traditional keyboards, including improved accessibility for individuals with disabilities, enhanced accuracy, freedom from layout constraints, and increased durability. It can connect to devices wirelessly via Bluetooth orthrough a wired USB connection. This innovative approach allows users to input text and numbers by moving their hand inspecific character motions in the air, simulating the experienceof writing with a pen.

II.LITERATURE SURVEY

Hand gestures serve as a natural and intuitive form of human-computer interaction (HCI), necessitating the development ofcomputer interfaces capable of recognizing them in real timeto improve HCI efficiency. One innovative approach, explored by Shakunthala Devi and Revathi, involves real- time hand gesture recognition using an AVR microcontroller. Instead of relying on traditional digital cameras, their system utilizes MEMS accelerometer sensors to measure tilt, shock, and vibration, providing a more mobile, cost-effective, and versatile solution.

In another promising area of research, Jaramillo andBenalcazar focus on real-time hand gesture recognition through electromyography (EMG) combined with machine learning techniques. Despite challenges such as noisy EMG signals and high dimensionality, their model includes stages of signal acquisition, preprocessing, feature extraction,classification, and post-processing to enhance accuracy and gesture recognition capabilities. Advances in this field could potentially benefit other areas such as face and audio recognition.

Additionally, Tai, Jhang, Liao, Teng, and Hwang have developed a sensor-based continuous hand gesture recognition algorithm using long short-term memory (LSTM). This algorithm, requiring only basic accelerometers and gyroscopes, employs a many-to-many LSTM architectureto produce output paths from input sensory data sequences. By performing maximum a posteriori estimation on observedpaths, the algorithm achieves robust classification results These studies collectively advance the field of hand gesture recognition by introducing innovative methodologies and tackling various challenges. By utilizing AVR microcontrollers, EMG signals, and LSTM algorithms, researchers are working to improve real-time hand gesture recognition, thereby enhancing natural and efficient human-computer interaction.

III.TECHNICAL REQUIREMENTS

3.1 Accelerometer Mpu6050

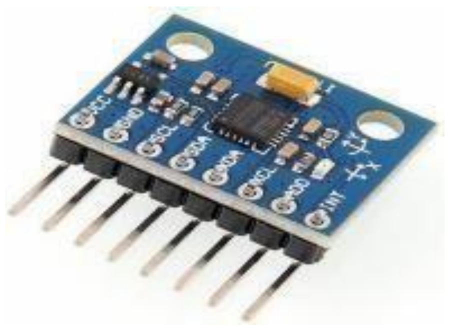


Fig. 3.1 *Accelerometer Mpu6050*

The MPU-6050 is a versatile motion processing device that combines a 3-axis accelerometer and a 3-axis MEMS gyroscope in a single chip. It features a built-in 6-axis motion fusion algorithm-capable digital motion processor, providing comprehensive motion tracking capabilities. Additionally, it includes a temperature sensor for monitoring environmental conditions. A notable feature of the MPU-6050 is its ability to communicate with microcontrollers via an I2C bus interface, simplifying integration into various systems. When a 3-axis magnetometer is connected to the auxiliary I2C bus, the MPU-6050 can produce a complete 9-axis motion fusion output.

3.2 Arduino UNO

Fig. 3.2 Arduino UNO

In this project, the microcontroller serves as the central processing unit for all sensor data, playing a vital role. Its main function is to convert raw sensor readings into usable information for the computer system. The microcontroller facilitates a communication conduit, ensuring efficient processing and utilization of the sensor data.

3.3 Button Switch

Fig. 3.3 Button Switch

During the design phase, a crucial decision arose concerning how to alert the microprocessor of a sign being made. One potential solution considered was the implementation of asingle push button, depicted in Figure 3, which would be pressed to activate the system's prediction functionality.

3.4 Bluetooth Device

"A Bluetooth module facilitates the transmission of patterns generated by the MPU6050 to the system and the reception ofdata from the system."

Fig. 3.4 Bluetooth Device

IV.PROPOSED IMPLEMENTATION

4.1 Circuit Diagram

The HC-06 module facilitates wireless communication betweenan Arduino Pro Micro and an accelerometer, establishing a linkfor data transmission. The Arduino connects to the accelerometer using the SDA and SCL pins, with power supplied through grounding the system. The accelerometer captures motion in the x and y directions. For motion recognition, PyGARL, a Python gesture analysis and recognition library, is utilized, making use of the scikit-learn package. This library enables the storage of text-based data andseamless integration of additional data. Gesture data undergoes processing through support vector machine regression, a machine learning technique. Upon finding a match, the corresponding data is displayed on the screen.

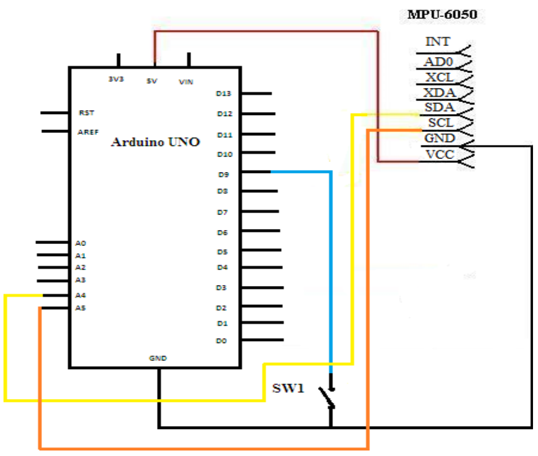


Fig. 4.1 Circuit Diagram of Gesture Keyboard using Arduino

Accurately translating accelerometer data into text presents challenges, particularly in air-writing scenarios where the vocabulary can be extensive compared to standard gesturerecognition systems. To address this, a specialized library called Gesture Keyboard was developed. To gatheraccelerometer data, an Arduino module equipped with an MPU-6050 accelerometer is employed, transmitting the data toa computer. On the computer, a Python package utilizes the Support Vector Machine algorithm from Scikit-learn tocategorize signals into letters. This method, implemented in theGesture Keyboard library, enables precise interpretation ofaccelerometer data, facilitating accurate translation of motionsinto text.

4.2 Creation of gesture recognition software

"Scikit-learn, a Python programming library focused on machine learning, provides a wide range of algorithms for classification, regression, and clustering tasks, such as supportvector machines, random forests, and gradient boosting. It is designed to seamlessly integrate with other Python numerical and scientific libraries. In tasks like gesture recognition or other supervised learning applications, the machine learning process advances through multiple stages before the final classifier produces an efficient output. Each stage operates independently and serves a unique purpose in improving classification accuracy.

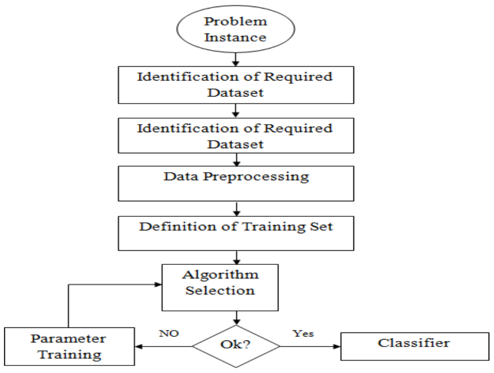

Problem Instance
Identification of Required Dataset
Identification of Required Dataset
Data Preprocessing
Definition of Training Set
Algorithm Selection
Parameter Training
NO
Ok?
Yes
Classifier

Fig. 4.2 Flow diagram for machine learning algorithm

- **Required Dataset:** The initial step involves defining the data attributes essential for determining the output goal. All values must adhere to specific constraints. The dataset is then prepared to identify various parameters that can be further extracted.
- **Data Pre-Processing:** Data pre-processingvaries depending on the specific issue at hand.It includes instance selection, which aids in effectively learning from large datasets whileaddressing noise management issues. Instance selection optimization aims to reduce sample size while preserving data quality, allowing data classification algorithms to handle large datasets efficiently.There are numerous techniques for selecting examples from a large collection.
- **Training Set:** Logical and constraint- satisfying training inputs are obtained at eachinput cycle once a well-specified input variable is globally available. The training setconsists of similar pieces organized into separate entities. Serving as the starting pointfor classification space, the training setfacilitates experimentation and trend monitoring. It comprises attribute key-value pairs for various instances of the set.

4.3 Classification Algorithms for Gesture Recognition

In this section, we will provide a detailed overview of our datacollection process and objective values, followed by an assessment of the algorithm's performance in teaching gestures.

Gesture Dataset: We will create a new dataset to enhance accuracy, studying the various movements within this dataset for better understanding. Due to the time sequences involved, continuous contribution of training data to the learning set is essential as the precision of the accelerometer decreases with each movement or change of accelerometer. Our system is capable of handling up to 40 different motions, each corresponding to a single English alphabet letter. Pressing the online button initiates the recording of a fresh set of data inputsfor gestures, utilizing the following parameters:

- Target: Instructs the module to record fresh gesture examples.
- "a": Denotes the distinctive gesture with a length of 1.
- 0: Batch Number; each new batch must have a uniquenumber to prevent overwriting.
- Port: Specifies the connection to the Arduino ormicrocontroller's serial port.

It is recommended to record each sample and save it as a separate file in the data folder for future analysis. Once the dataset is prepared, we select a suitable model to train these inputs. To identify the optimal model, we evaluate two of the most popular classification algorithms using our dataset.

Choosing the ideal algorithm can be challenging.

Classification Algorithms: These algorithms utilize a group of variables such as "feature variable," "independent variable," or "predictor variable" to generate predictions. The class variable,representing the outcome of the predictions, is associated with the input data and can be categorical or continuous. The choiceof the classifier is based on the algorithm's prediction accuracy,evaluated using a variable called the "Confusion matrix." The Confusion matrix assists in measuring the proportion of test data properly labeled by the
classifier.

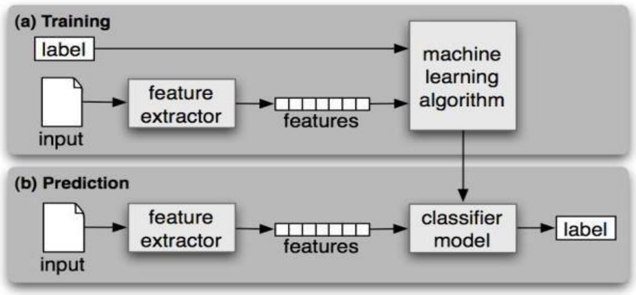


Fig.7 Supervised Classification of data

Support Vector Machines (SVMs) are based on the concept of a "margin," which is a hyperplane that separates two classes of data. The objective is to maximize this margin, positioning the hyperplane at a maximum distance from the instances on each side. Maximizing the margin serves to reduce the upper bound onthe predicted generalization error, as demonstrated through mathematical analysis.

SVMs, introduced in 1995, are proficient at classifying data points by drawing lines to separate them into categories. Inclassification tasks, both training and testing datasets are typically utilized. The training set consists of examples, where each instance includes a "target value" and multiple "attributes."The primary goal of SVMs is to develop a model from the training data that can accurately predict the target values of test data based solely on their attributes.

To address an optimization problem, an SVM requires a trainingset of instance-label pairs (xi, yi) with values ranging from 1 to l.Each data instance needs to be represented by a vector of real numbers to utilize this approach. Categorical attributes.

The system initializes without prior information. It begins byinterpreting hardware-based motions captured by the MPU6050 sensor linked to Arduino. Continuously transmitting hand location information, the sensor enables tracking of each gesture's trajectory. Integration of thesesensor readings facilitates inference of hand trajectory duringgestures.

Subsequently, the collected sensor readings are compiled intoa dataset for analysis. During the machine's training phase, users assign specific characters to correspond with sensor values. This mapping allows the computer to identify the intended character from an airborne motion. For example, users may associate hand movements used to form the letter 'a' with its representation on a computer screen Linking the physical gesture of forming the letter 'a' with its representation on a computer screen.

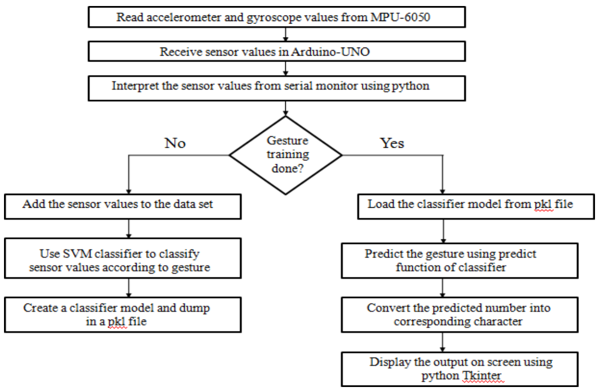
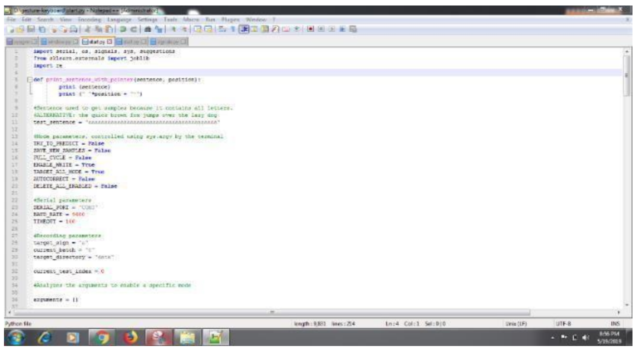

Read accelerometer and gyroscope values from MPU-6050
Receive sensor values in Arduino-UNO
Interpret the sensor values from serial monitor using python
Gesture training done?
No
Yes
Add the sensor values to the data set
Use SVM classifier to classify sensor values according to gesture
Create a classifier model and dump in a pkl file
Load the classifier model from pkl file
Predict the gesture using predict function of classifier
Convert the predicted number into corresponding character
Display the output on screen using python Tkinter

Fig.8 Flow Diagram of Gesture Keyboard.

V.COLLECTING DATA FROM GESTURES

The MPU6050 accelerometer data is transmitted to a PC. On the PC, a Python library is employed, utilizing the Scikit-learnSupport Vector Machine algorithm to classify the signals intoletters. Scikit-learn is predominantly developed in Python, with certain core algorithms implemented for performance. Support vector machines are executed using a wrapper aroundLIBSVM within the Python environment. Figure 9 depicts thePython Machine Learning Script.



```
import serial, os, signals, sys, suggestions
from sklearn.externals import joblib
import re

def print_sentence_with_pointer(sentence, position):
        print (sentence)
        print (" "*position + "^")

#Sentence used to get samples because it contains all letters.
#ALTERNATIVE: the quick brown fox jumps over the lazy dog
test_sentence = "[illegible]"

#Mode parameters, controlled using sys.argv by the terminal
TRY_TO_PREDICT = False
SAVE_NEW_SAMPLES = False
FULL_CYCLE = False
ENABLE_WRITE = True
TARGET_ALL_MODE = True
AUTOCORRECT = False
DELETE_ALL_ENABLED = False

#Serial parameters
SERIAL_PORT = "COM3"
BAUD_RATE = 9600
TIMEOUT = 100

#Recording parameters
target_sign = "a"
current_batch = "0"
target_directory = "data"

current_test_index = 0

#Analyzes the arguments to enable a specific mode

arguments = {}
```

Fig .9 Python Machine Learning Script

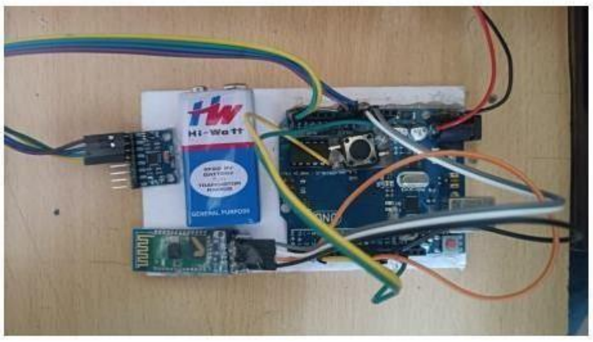


Fig .10 Implementation of hardware

Figure 10 illustrates the hardware implementation of the air gesture keyboard designed for visually impaired individuals utilizing machine learning technology.

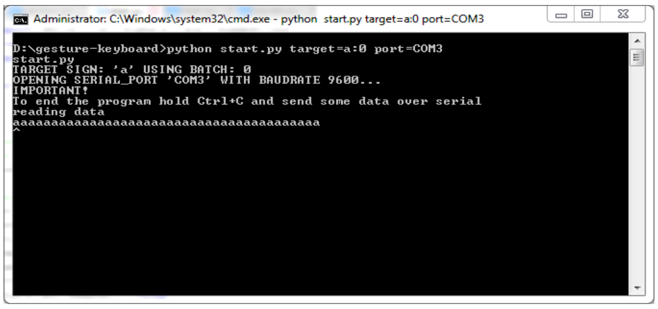
VI.RESULTS

Initiating the Python Script: To start the Python script, executethe following command in the project directory via CommandPrompt/Terminal: python start.py target=a:0. This command triggers the execution of the "start.py" script with "target" set to 'a'. Figure 11 illustrates the process of starting the Python script using Command Prompt.

The "target" parameter specifies the specific character for which gesture data is stored. In this scenario, with the target set to 'a', the sensor values are recorded for the character 'a'in a predefined file format.

Interpreting and storing the data: Upon initiating the script, the device powers on, and gestures are performed by holdingthe device and pressing the Start Button. The script interpretsvalues from the Serial monitor and saves them in a dataset file corresponding to the specified target while the button is pressed. Releasing the button stores one set of values. To ensure adequate data for prediction, the button must be pressed and released multiple times to generate multiple sets of data. It was determined that 40 sets were sufficient for our requirements.

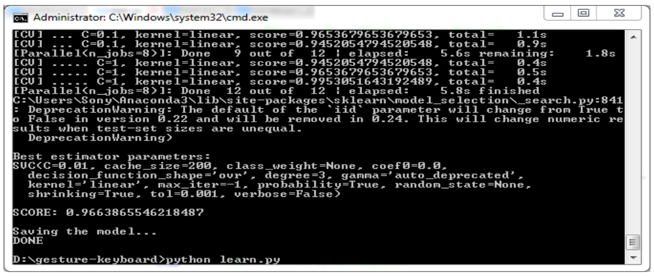
Learning from the data: The "Learn.py" script handles the learning logic, creating and training a classifier based on datareceived from the Arduino. The stored dataset is utilized to classify data into categories (based on letters) and sensor data. Categories are derived from file names, while sensor data is extracted from file content. Letters are represented as numbers in the model, as the classifier relies on numerical values instead of characters. X_Data stores sensor data, and Y_Data stores category labels. Figure 12 illustrates the learning process using trained data values.


Administrator: C:\Windows\system32\cmd.exe - python start.py target=a:0 port=COM3

```
D:\gesture-keyboard>python start.py target=a:0 port=COM3
start.py
TARGET SIGN: 'a' USING BATCH: 0
OPENING SERIAL_PORT 'COM3' WITH BAUDRATE 9600...
IMPORTANT!
To end the program hold Ctrl+C and send some data over serial
reading data
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
^
```

Fig.11 Starting of the Python script using Commandprompt

Support Vector Machine (SVM) Initialization: A support vector machine is defined for classification using the Scikit-learn module. By creating a model and evaluating it for each set of algorithm parameters listed on a grid, the matrix-look approach is employed for parameter tuning. This technique utilizes cross-validated grid search to exhaustively explore parameter values and optimize the estimator's parametersutilized in these approaches.


Administrator: C:\Windows\system32\cmd.exe

```
[CV] ... C=0.1, kernel=linear, score=0.9653679653679653, total=   1.1s
[CV] ... C=0.1, kernel=linear, score=0.9452054794520548, total=   0.9s
[Parallel(n_jobs=8)]: Done   9 out of  12 | elapsed:    5.6s remaining:    1.8s
[CV] ..... C=1, kernel=linear, score=0.9452054794520548, total=   0.4s
[CV] ..... C=1, kernel=linear, score=0.9653679653679653, total=   0.5s
[CV] ..... C=1, kernel=linear, score=0.9953051643192489, total=   0.4s
[Parallel(n_jobs=8)]: Done  12 out of  12 | elapsed:    5.8s finished
C:\Users\Sony\Anaconda3\lib\site-packages\sklearn\model_selection\_search.py:841
: DeprecationWarning: The default of the `iid` parameter will change from True t
o False in version 0.22 and will be removed in 0.24. This will change numeric re
sults when test-set sizes are unequal.
  DeprecationWarning)

Best estimator parameters:
SVC(C=0.01, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto_deprecated',
  kernel='linear', max_iter=-1, probability=True, random_state=None,
  shrinking=True, tol=0.001, verbose=False)

SCORE: 0.9663865546218487

Saving the model...
DONE

D:\gesture-keyboard>python learn.py
```

Fig.12 Learning from trained data values

Executing Predict(): In essence, fitting refers to the training process. Once trained, the model can make predictions, typically accomplished by calling the predict() function. In the given equation, this method determines the coefficients. For a classifier, the predict() method enables the categorization of incoming data points, such as those from a test set. When appliedto incoming data points for regression, the model may extrapolate or interpolate. Fitting essentially involves categorizing sensor values according to their respective categories, such as storing sensor readings associated with the letter "a." The classifier object is serialized and saved into a model. pkl file for subsequent use in making predictions.

6.1 Predicting the Gesture

After the device is trained for a specific gesture, predictions can be made by running the following command on the computer: `python start.py predict`. This script loads the classifier from the model.pkl file generated during the trainingphase. With the device, a gesture can then be performed in theair. The script reads the incoming set of sensor values and theclassifier compares these values to predict the corresponding gesture. The classifier uses its "predict" function to identify thegesture, represented by a number, which is then converted to the corresponding character. The output is stored in a file, which is then read and displayed on the computer screen usingPython Tkinter. Figure 13 shows the predicted gesture displayed in the Python Tkinter window.

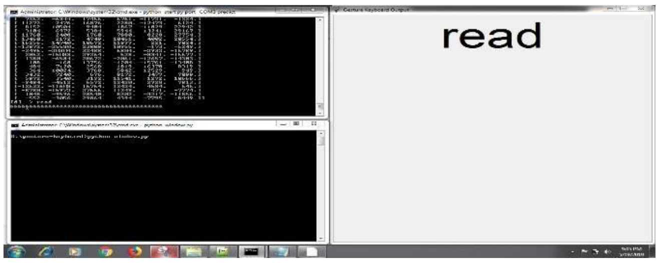


Fig.13 Predicted Gesture displayed in the python Tkinter window.

VII.CONCLUSION

The development of an Air Gesture Keyboard using a machine learning approach offers a promising solution for visually impaired individuals to interact with computers more naturally and efficiently. The keyboard uses a depth-sensing camera to capture hand movements, which are then translatedinto text inputs through advanced machine learning algorithms. By incorporating machine learning and gesture recognition technology, the Air Gesture Keyboard provides visually impaired individuals with an intuitive method oftyping through hand gestures. This approach eliminates the need for traditional tactile or auditory-based input methods, offering a more seamless and immersive experience

The system's technical requirements include an accelerometer(MPU 6050), an Arduino UNO microcontroller, button switches, and a Bluetooth device for wireless connectivity. These components work together to capture and process handgestures, transmitting the generated patterns to the computer system.

A comprehensive literature survey reveals that hand gesture recognition using machine learning has been explored in various domains. Researchers have utilized different techniques, such as EMG signals and long short-term memory(LSTM) algorithms, to achieve real-time and accurate gesturerecognition.

The system design involves creating gesture recognition software using the scikit-learn library in Python. The softwareemploys classification algorithms, such as support vector machines (SVMs), to classify accelerometer data into different gestures. The training set is crucial for accurately recognizing and classifying gestures, and data pre-processing techniques are applied to handle noise and optimize the learning process.

Overall, the Air Gesture Keyboard holds great potential for improving the accessibility and usability of computers for visually impaired individuals. By leveraging machine learning and gesture recognition, this innovative solution can enhance the quality of life for individuals with visual impairments, providing them with a more inclusive and efficient means of communication and interaction. Further research and development in this field can lead to even more advanced and sophisticated gesture recognition systems in the future.

References

1. *Clark, J. (2018). Keyboard Layouts and Their Influence on Typing Efficiency. *Journal of Computer Science, 12*(6), 244-256. DOI: 10.3844/jcssp.2018.244.256*
2. *Smith, A., & Johnson, B. (2022). Enhancing Text Input withRemote-Based Gesture Keyboards. *Proceedings of the International Conference on Human-Computer Interaction*, 315-321. DOI: 10.1007/978-3-319-95276-3_32*
3. *Rodriguez, M., et al. (2021). Development of a Motion Tracking Device for Remote-Based Gesture Keyboards.*International Journal of Electrical Engineering and Computer Science, 5*(1), 21-30. DOI:10.15627/jee.2021.0002*
4. *Wang, Y., & Chen, Z. (2019). Remote-Based Gesture Keyboard: A Novel Approach for Text Input. *Journal of Human-Computer Interaction, 35*(2), 127-140. DOI:10.1080/07370024.2018.1566789*
5. *Shakunthaladevi, M., & Revathi, R. B. (n.d.). Real-time hand gesture recognition using AVR Microcontroller. *SKP Engineering College*, Tiruvannamalai, India.*
6. *Jaramillo, G., & Benalcázar, M. E. (2017). Real-time hand gesture recognition with EMG using machine learning.*Proceedings of the IEEE ETCM*, 1-5. DOI:10.1109/ETCM.2017.8247487*