IDE Code Compiler for the physically challenged (Deaf, Blind & Mute)

Sayali Wangane[1], Sanika Purao[2], Vaishnavi Tiwari[3], Prof. Sujata Kullur[4]

[1,2,3]Student, Dept. of Information Technology, Usha Mittal Institute of Technology, Mumbai, Maharashtra, India
[4]Professor, Dept. of Information Technology, Usha Mittal Institute of Technology, Mumbai, Maharashtra, India

---***---

Abstract - *Compilers are critical, widely-used complex software which is often based on text format, making it difficult for the disabled.*

The main objective of this paper is to develop a compiler based on Speech Recognition and Sign Language Recognition. The compiler gives the user a choice to choose the type of input, whether it is textual or conversational (audio) or gesture (image).

Key Words: Programming, Sign Language Recognition, Speech Recognition, Compiler, Hand Gesture Recognition, Machine Learning, IDE.

1. INTRODUCTION

Most people with disabilities avoid programming jobs because of the lack of tools that accommodate their disabilities. Furthermore, it is seen that a person working on a computer cannot work or type for a longer duration as it may cause the issue of back or wrist pain. This can be avoided easily by switching from typing the code to speaking the code as IDE supports speech recognition. IDE Code Compiler for the physically challenged (Deaf, Blind & Mute) is a compiler to compile codes.

This project aims to propose a model that will enable a person to program without typing, either by using speech or sign language. The Speech Recognition module has the ability to program by evaluating a keyword, idiom, or a sentence in spoken expression through speech recognition and convert those words into a text-based format.

Similarly, the Sign Language module focuses on the ability to program by using sign language through the hand gesture recognition and converting those recognized words into a text-based format. These textual words converted from speech/ sign (gestures) are then fed to the compiler, which in turn compiles and displays the output along with the time of execution.

2. MOTIVATION AND OBJECTIVE OF STUDY

According to WHO, over 5% of the world's population or 430 million people (432 million adults and 34 million children) are suffering from hearing impairment. [4]

In 2015, an estimated 253 million people had visual impairment worldwide. Where 36 million were blind and a further 217 million had moderate to severe visual impairment (MSVI).

These statistics formed the major motivation for our project. At the current time, there is no such IDE available that provides the user with input in both voice and image format.

3. PROBLEM STATEMENT

Most people think programming for the physically challenged is an impossible task. To serve the people with disabilities and reduce the incidents of repetitive stress and injuries among people who do programming by sitting at a desk we need a system that enables a person to program without typing. A design that generates environments that enable people to program either by their voice or sign language in order to bridge the gap between Human Impairment and technology.

4. EXISTING SOLUTION VS PROPOSED SOLUTION

Table -1: Existing solution Vs Proposed Solution

	Existing Solution	**Proposed Solution**
	Compilers available are text oriented and typewritten.	The compiler will not only be text-oriented, but it will also be voice-based and sign-language based.
Voice Compiler	One of the voice-based compilers available is Serenade. Limited functions available for free use.	Voice based compiler, where the user can code using their voice. It will be free to use.
Sign language-based Compiler	No image compiler available which takes sign language as an image input.	Image based compiler, which takes sign language as an image input. It will be free to use.

5. SOLUTION

We are developing an IDE, where the user can code using their voice or by using sign language. This model will reduce the number of errors occurring while typing. Also, this will help the Deaf, Blind & Mute communities by using their speech or sign (hand gestures) to code.

6. REVIEW OF LITERATURE

Nitin Washani, Sandeep Sharma" Speech Recognition System: A Review", [1] In this paper, for better understanding and representation, they have classified the system into Front End Analysis and Back End Analysis of speech recognition systems in each part which also provided higher accuracy.

Nuzhat Atiqua Nafis, Md. Safaet Hossain, "Speech to Text Conversion in Real-Time", In this paper [2], they have used a method to design a text-to-speech version module by the use of MATLAB. The method proposed is simple to implement and involves less use of memory spaces.

M. S. Nair, A. P. Namitha, and S. M. Idicula," Conversion of Malayalam text to Indian sign language using synthetic animation", In this paper [3] they have applied a method of using a synthetic animation approach where they have converted Malayalam language to Indian sign Language. In this method HamNoSys is being used for the intermediate representation of sign language.

Vishal Trivedi," Life Cycle of Source Program - Compiler Design" In this research paper [5] the exact procedure behind the compilation task and step by step evaluation of source code is explained. They have designed a compiler using the analysis phase and Synthesis phase, additionally they have also discussed topics such as High-level languages, Low-level language, Pre-processors, Translators, Compilers, Phases of Compiler, Interpreters, Error Handling.

Jatin Chhabra, Hiteshi Chopra, Abhimanyu Vats," Research paper on Compiler Design" In this paper [6] they have discussed the storage management for collections of objects, including temporary variables. Also, about how the Error Handling occurred with failures due to many causes such as errors in the compiler or its environment, design errors in the program being compiled, transcription errors, incorrect data, etc.

7. SYSTEM ARCHITECTURE DIAGRAM

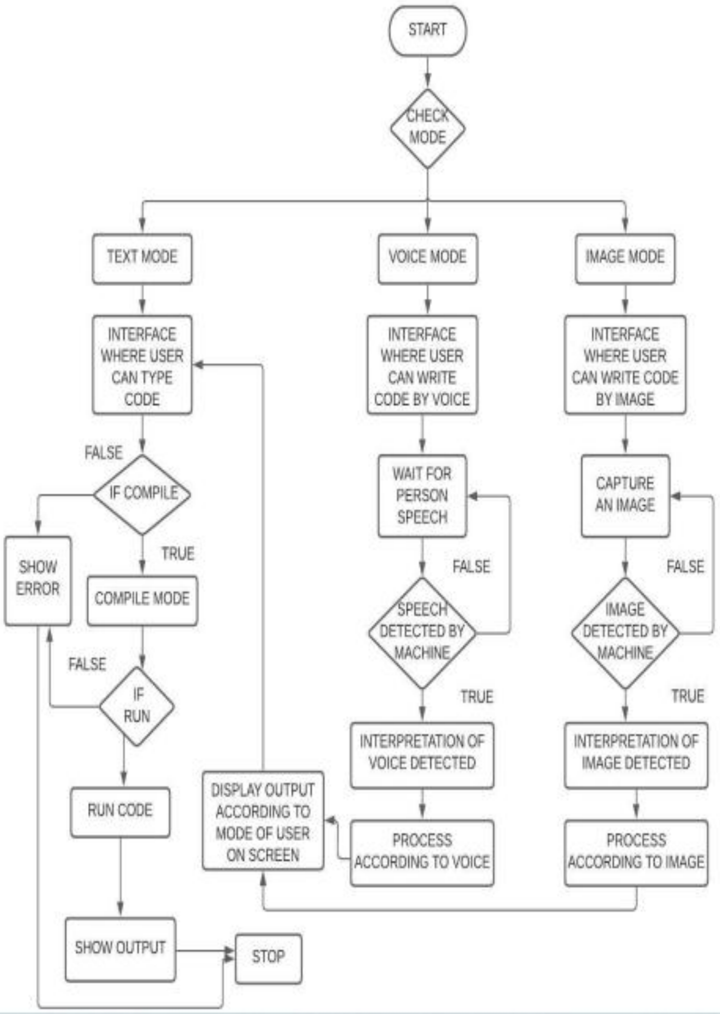
START
CHECK MODE
TEXT MODE
VOICE MODE
IMAGE MODE
INTERFACE WHERE USER CAN TYPE CODE
INTERFACE WHERE USER CAN WRITE CODE BY VOICE
INTERFACE WHERE USER CAN WRITE CODE BY IMAGE
FALSE
IF COMPILE
TRUE
SHOW ERROR
COMPILE MODE
WAIT FOR PERSON SPEECH
CAPTURE AN IMAGE
FALSE
FALSE
SPEECH DETECTED BY MACHINE
IMAGE DETECTED BY MACHINE
TRUE
TRUE
FALSE
IF RUN
INTERPRETATION OF VOICE DETECTED
INTERPRETATION OF IMAGE DETECTED
RUN CODE
DISPLAY OUTPUT ACCORDING TO MODE OF USER ON SCREEN
PROCESS ACCORDING TO VOICE
PROCESS ACCORDING TO IMAGE
SHOW OUTPUT
STOP

Fig -1: System Architecture Diagram

8. METHODOLOGY

8.1 Compiler (IDE):

The compiler compiles and executes the source code in Python Language. The compiler is made using the Django Framework. We have used URLs for viewing the frontend. View function is a python function that takes a Web request and returns the Web response. To execute the code from the text area on clicking the submit button, we have used the POST method. To return the output we have used stdout which is a built-in file object used to display output directly to the screen console. The workflow of execution of code is that we will create and save a file then execute the file and save the output in that file and then finally return the output to our main page by reading that file data.

8.2 Speech to Text Mode:

1. Recognition of user's speech. (Audio input)
2. Usage of speech recognition that converts speech to text. (translation).
3. Python Library used for the voice-based compiler: webkitSpeechRecognition API. The Web Speech API(webkitSpeechRecognition) written in JavaScript is used to incorporate voice into web apps.
4. Forming the entire program in an IDE.
5. Obtaining text error if any.
6. Obtaining text output.

8.2.1 Working of Web Speech API (webkitSpeechRecognition):

1. Create speech recognition object: In this step we create a new speech recognition object:

   recognition = new webkitSpeechRecognition();

2. Register event handlers: The speech recognition object has many properties, methods and event handlers. When recognition. Continuous is set to true, the recognition engine will treat every part of your speech as an interim result. When it is set to true, then results should be returned.
3. Start recognition: By calling therecognition.start(), it activates the speech recognizer. Once it begins capturing the audio, it calls the onstart event handler, and then for each new set of results it again calls the onresult event handler.

8.3 Sign to text conversion:

Detects American Sign Language and converts gestures from a live webcam into text. This module of our project is a sign language alphabet recognizer using Python, OpenCV and a convolutional neural network model for classification. The goal of this module is to build a neural network which can identify the alphabet of the Sign Language and translate it into text.

8.4 Dataset:

The primary source of data for this module we referred to was the compiled dataset of American Sign Language (ASL) from Kaggle and then made our own dataset. The dataset we made includes the images of all letters from A-Z. We have divided the dataset for training and testing the CNN model. We have used approximately 1000 images of each alphabet for training and testing purposes.

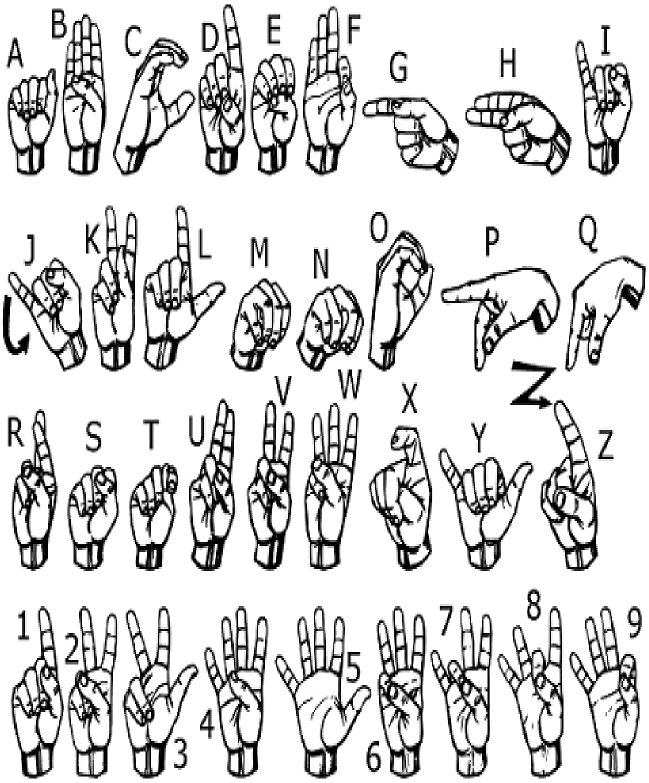

Fig-2: Gesture Dataset

8.5 Convolutional Neural Network:

The pixels of the image are extracted from each frame, and the Convolutional Neural Network is applied for training and classification. The model is further then evaluated and the system would be able to predict the alphabets. The Convolutional layers with ReLU as an activation function which decides whether the neuron input to the network is important or not. The system then predicts the input gesture of the user and displays the result.

8.6 Gesture Classification:

Algorithm Layer:

1. Apply a gaussian blur filter and threshold to the frame taken with opencv to get the processed image after feature extraction.
2. This processed image is passed to the CNN model for prediction and if a letter is detected for more than the letter is printed and taken into consideration for forming the word.

8.7 Sign to Text Data Flow Diagram:

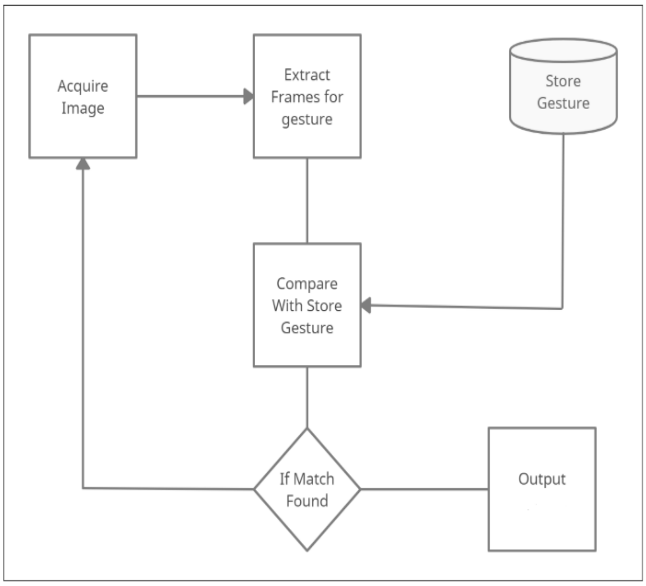

Fig-3: Sign to Text Data Flow Diagram

8.8 Model Deployment:

We have deployed our model using Tkinter through which the characters are recognised and printed on our Django code compiler using PyAutoGUI which is a cross-platform GUI automation Python module.

9. CONCLUSION AND FUTURE SCOPE

1. This IDE code compiler is still in the development phase and there will be the addition of other libraries and functionalities in this IDE.
2. We tend to increase the scope by including more programming languages such as C, Java, etc in text, Voice based and Sign Language based modes.
3. We will focus on creating a user-friendly interface.
4. In the future, we can expand our project by adding some resources for users to learn programming languages and practice using our code compiler.

ACKNOWLEDGEMENT

We have a great pleasure to express our gratitude to all those who have contributed and motivated us during our project work. We would like to thank Prof. Sujata Kullur for her support and guidance, our HOD Dr. Sanjay Shitole, our principal Dr. Shikha Nema, Dr. Anita Morey and to all the teachers whose help made us understand our topic well. Thank you all.

REFERENCES

[1]" Speech Recognition System: A Review" Nitin Washani, Sandeep Sharma. In-ternational Journal of Computer Applications (0975-8887) Volume 115-

No. 18, April 2015

[2] "Speech to Text Conversion in Real-Time" Nuzhat Atiqua Nafis, Md. Safaet Hossain. International Journal of Innovation and Scientific Research, ISSN 2351-8014 Vol.17 No. 2Aug. 2015, pp. 271-277, August 2015.

[3]" Conversion of Malayalam text to Indian sign language using synthetic animation" M. S. Nair, A. P. Nimitha and S. M. Idicula. 2016 International Conference on Next Generation Intelligent Systems (ICNGIS), Kottayam,2016, pp. 1-4, doi: 10.1109/ICNGIS.2016.7854002, 2016.

[4] World Health Organization"Deafness and hearing loss".WHO Newsroom Fact sheets details of Deafness and hearing loss, 2020.

[5]" Life Cycle of Source Program - Compiler Design" Vishal Trivedi, International Journal of Innovative Research of Technology, Volume 5, ISSN:2320-2882, November 2017.

[6]" Research paper on Compiler Design"Jatin Chhabra, Hiteshi Chopra, Abhimanyu Vats 2014 International Journal of Innovative Research of Technology ,Volume 1 Issue 5, ISSN : 2349-6002, 2014.

BIOGRAPHIES

Sayali Wangane
Dept: Information Technology
Email: sayaliwangane@gmail.com

Sanika Purao
Dept: Information Technology
Email: sanikasuhas16@gmail.com

Vaishnavi Tiwari
Dept: Information Technology
Email:
tiwarivaishnavi652000@gmail.com