RESEARCH ARTICLE OPEN ACCESS

A Survey of Programming Editors for the Visually Impaired

Dr Mercy Rajaselvi V[1], Jane Gloria F[2], Mohitha V[3], Guhan Selvarajan [4]

[1] *(Department of Computer Science, Easwari Engineering College, Chennai-89*

[2] *(Department of Computer Science, Easwari Engineering College, Chennai-89*

[3] *(Department of Computer Science, Easwari Engineering College, Chennai-89*

[4] *(Department of Computer Science, Easwari Engineering College, Chennai-89*

ABSTRACT
It is commonly known fact that the differently abled are not provided with proper education and educational tools because of the lack of proper infrastructure or equipment to help them. Throughout the years, various technologies have been developed that can be used to teach differently abled students computer programming. But none of these specifically target the visually challenged student or have certain drawbacks that do not enable easy learning and usage. Audio Language Programming was introduced to help the visually impaired grasp algorithmic thinking and problem solving. But there is no concentration on code writing because the main objective of ALP is to learn programming concepts rather than syntax. Tools that aim to help the visually impaired students to code must do so without costly equipment or extreme changes to existing systems. It should be a software solution that can be easily implemented. Such solutions can also be commercialized to help easier coding for industry experts. To enable the visually impaired students to easily access computer programming tools and techniques will open new opportunities for them. One main method discussed throughout this survey will be the use of voice-enabled programming editors that will enable hands-free coding.

Keywords - Audio Programming language, Programming tool, visually challenged, Speech to text, Speech interface

Date of Submission: 07-04-2020 Date of Acceptance: 22-04-2020

I. INTRODUCTION

Computer Science is a branch of engineering which revolves around logic and reasoning, which greatly depends on visual cues. This restricts the study and practice of computer science and various other associated technologies to people who are visually impaired. Normal teaching and practice methods are ineffective to students with special needs. Flowcharts and other visual aids cannot be implemented and used by the visually impaired student. They require special hardware like braille keyboards, etc. in order for them to use computer system. This proves to be difficult in many parts of the world, especially India. And so many visually impaired students opt for descriptive subjects like arts which they can dictate to an external scribe. This proves to be a time-consuming task which limits the range of study of these students.

Foundations for computer studies start at the school level. By encouraging students to take up such technical studies, the future prospects of these students are vastly improved. Over the years, the popularity of computer studies has increased with a greater number of students opting for degrees in this field with rising opportunities for well paid jobs. Getting more students involved in this field is imperative to enhancing the lives of students, especially students with special needs.

One way to get more visually impaired students into technical subjects like coding is to make a user interface that is accessible. It could be accomplished by making the whole software voice-enabled. Each component on the screen is described in such systems enabling the user to access the components with ease. Voice-enabled systems use various technologies to achieve complete autonomy. These systems can also be further expanded to help developers who have sustained industry related injuries and have trouble typing. This survey discusses the various available software and programming tools, either implemented or under research, that enable voice programming.

II. LITERATURE SURVEY

Muhammad Shoaib et al. [1] devised and researched the use of a new type of auditory feedback called adaptive auditory feedback to enable the blind user to access the contents of the desktop. This technology has been found to be better than traditional speech-only and non-speech only auditory feedback mechanisms. This technology has been used to enable the visually impaired user to use computer systems with ease. Adaptive Auditory Feedback is found to switch between speech-only and non-speech only feedback on the basis of user state (starting state- speech only, continuing state- non-speech only). The repetitive speech-only instructions are found to become more irritating and time-consuming for the user. A survey of AAF was conducted using 15 participants who were given the task of filling a form. After the task was completed, the user response to the system was recorded. According to the response, it was identified that listening to similar speech-only instructions for more than three times becomes irritating/annoying to the visually impaired. The results and the detailed interviews with participants showed that adaptive auditory feedback is more appropriate than speech-only and non-speech feedback and more enjoyable. It also was found to convey more meaningful and critical information compared to non-speech auditory feedback.

Abhishek Chand et al [2] discussed a voice coding environment using a technology called Silvius. Silvius is a hybrid of commercially available speech recognition tools like Dragon Naturally Speaking and Aenea. A server is used to recognise sentences got from piping the microphone output. The sentence will then be parsed and run through a grammar which will produce virtual keystrokes. Silvius has created a parser tree with meta-python objects. An N-gram language model is generated by walking the parser tree. An Abstract Syntax Tree (AST) is produced using the text. The AST is walked and commands are executed. Silvius has been deemed advantageous because of its low computing power and the fact that it can work on cloud.

Anurag Singh et al [3], discussed the implementation of a voice to code editor in Java language. They outlined the need for five modules for complete operation of the system. The five modules are Language Specifying module, Text Operation module, Commands Specification module, Differentiate Text and Symbol module and Compile and Run module. Each module is designed in such a way as to efficiently help the programmer code using only spoken commands. The project is still ongoing as of the publishing of this paper. The future works the the authors have detailed are the implementation of the system to work with multiple programming languages like C, C++, HTML, etc.,

Lucas Rosenblatt et al [4] researches the use of voice assisted programming for people with upper body impairments like those with limited or no use of their upper extremities. The research had been conducted in three parts. First, a Wizard of Oz study was conducted with ten people without upper body impairments. These people were asked to complete programming tasks using only voice commands. Second, with the results of the WoZ study, VocalIDE was created as a vocal programming editor. Third was the study of the prototype when people with disabilities used VocalIDE. VocalIDE is a voice to code editor using JavaScript. It is a web application that allows user to write and edit code using a set of vocal commands. The system had two components: the automatic speech recogniser and the rule-based syntax parser. The feasibility of the system was analysed by researching the ease of use of VocalIDE by people having upper body impairments. This process had been implanted in four parts: the Box and Block Test, the baseline current computer interaction test, the system evaluation using their voice, and an unstructured interview where the participants were asked about their experience in using both traditional user interface and VocalIDE. . The participants had completed 300 tasks (150 baseline, 150 VocalIDE). The duration to complete each task in the baseline condition had been 21.24 seconds, while the average task completion time in the VocalIDE condition had been 13.81 seconds. The most frequent challenge faced had been the misinterpretation of the spoken commands. The overall result of VocalIDE has estimated that VocalIDE helps in coding and could potentially improve coding experience.

Kaveendra Lunuwilage et al, [5], discussed the development of research project of DICENS. It is targeted for visually impaired users to use voice-based commands in order to learn programming language concepts. DICENS aims to develop a set of technologies and products to help visually impaired users learn programming faster and with more accurate results. The program workflow is loaded onto the system before the user starts coding so that the logical flow of the program is maintained. The four components in the DICENS research project are the Speech Recognition Engine, Semantic Analysis Engine, Code Play Supervision Engine and User Interaction and Braille Engine. Voice commands are recognized by the speech recognition engine which are then processed to identify the keywords and the program structure is developed. The editing and navigation are done by using the last two components to make the system more user friendly. The system also provides alternative solutions which makes the user’s program more logically sound. This research project achieved an accuracy of 85% but failed in areas with high noise.

Rinor S. Maloku et al., [6] developed HyperCode, an interface that enables for voice coding in Java. HyperCode consists of python scripts which will map the continuously spoken text to custom commands in Natlink. Dragon Naturally Speaking was used as the standard speech recognition software. A survey was conducted to test the times taken by different modes of input like: keyboard, mouse and voice. The conclusion of the survey was that using a combination of keyboard, mouse and voice inputs produced the best results (46 seconds compared to 65 seconds and 84 seconds using keyboard/mouse and voice input respectively). HyperCode is used to achieve this combination input.

Amber Wagner et al. [7], suggested the need for a voice-driven tool for motorically challenged children to help them learn to code. They developed Myna, a voice driven Java application. This tool runs parallel to Scratch (graphical programming interface developed by MIT). The tool can obtain the voice commands from the user and simultaneously does processing and interpretation of the commands against a pre-defined grammar. Scratch helps in simulating the actions of the mouse and keyboard required while writing and compiling code. The implementation of this project has been completed but further improvements like adding all static commands to the grammar and adding the provision to undo or remove commands are still pending. The future scope of this project has been said to be fully understood only after obtaining solid data of its implementation and usefulness in the learning environment.

D.D. Langan, et al., [8], created a programming IDE manually disabled by making the entire system voice enabled. Since programming languages (with well-defined grammar) provide only a limited set of input at any given time, these limited inputs can be selected vocally as they are listed out by the IDE. This research led to the development of a voice-activated syntax directed editor called VASDE which was used to create Java programs. Eclipse and JSAPI were combined to create the main infrastructure of VASDE. In addition to VASDE a voice activated GUI editor called VAGUE was implemented to support buttons, labels, text-fields, combo-boxes, etc. VAGUE generated the described GUI using Java code with Eclipse as the platform. Even with both VASDE and VAGUE implemented together, it could not provide the entire solution for manually disabled programmers as only a portion of it was voice activated. The overall system resulted in time lags and certain user spoken commands being ignored, the research also resulted in low accuracy during the recognition of certain commands.

Dat Tran et al.[9], designed an audio programming tool enables the blind or visually impaired use to code in the C# programming language. All the components are voice-enabled meaning the entire project can be made and deployed using only the user’s spoken commands. The user is able to start the project by choosing a template from a list of available templates. Features like auto-completer helps the user write long class or method names. Compiling and debugging in this has been done by the compiler by auto generating code for producing voice. Errors will be read out one at a time and the user is guided to the line of code containing error in the program. This procedural error correction is repeated till all the errors are corrected. The user can also use mouse and keyboard shortcuts for checking the output.

Andrew Begel et al, [10] give an account about the cognitive effects of a speech enabled programming editor which they designed and implemented called SPEED (SPEech EDitor). It enables coding in Java on the Eclipse IDE. The author has tested the system using both a commercial speech to text converter (DNS) and a machine-based speech recogniser. The overall conclusion of this experiment was that developers found it slower to code using voice that actually manually typing the code. But SPEED was relatively simple for the developers to learn and they preferred to described the code template rather than speak literal syntax. Users using DNS found both speed and accuracy to be dismal whereas users using human voice recogniser reported higher speeds but only after sufficient training. Recognition time ranges between 0.5 to 0.75 seconds. Users testing the human speech recognizer could speak at a normal pace and were able to pause in the middle of uttering a command. These users reported 12% to 21% error while using spoken Java Spoken Commands. Even after SPEED reported such successes, developers in the industry were hesitant to use the system citing noise pollution, cognitive interferences and wasting time by not using their hands.

Alain Désilets et al.[11] have proposed a solution for people suffering from Repetitive Strain Injury (RSI). Multiple studies have taken voice-enabled coding as the solution for Repetitive Strain Injury (RSI). One such study resulted in VoiceCode. This has allowed developers to code using naturally spoken syntax which gets converted to proper code in real time. Since programming language syntax is awkward to utter, VoiceCode is aiming to create an environment in which the programmer can use naturally spoken commands to write code, navigate and also modify the code. It the Continuously spoken voice commands are interpreted as mostly independent context sensitive commands i.e based on the previous word spoken, the command is interpreted. Once the user corrects an error in the program, VoiceCode adapts itself to not make the same mistake. Code templates and flexible wording are used so that the same code can be produced by multiple naturally spoken commands. Developers found the system useful but it could not be effectively used by beginners or visually impaired students to learn or execute code.

Michael Nichols et al., [12], created a special purpose programming language for navigating VoiceXML pages called Aural Language for VoiceXML Interpretation and Navigation abbreviated as ALVIN. VoiceXML was created by the W3C consortium for the purpose of accessing internet content using audio. ALVIN allows individual statements to be differentiated based on the content rather than punctuation present in the command. The design is based on a consistent command syntax also known as sentence structure.

Alan Desilets, [13] researched the various techniques and issues available in existing voice coding and proposed a new solution termed as VoiceGrip. VoiceGrip was proposed to aid the programmer to code by voice thus enabling a hands-free experience. The user will give input in the form of spoken pseudocode syntax which will be easier to speak out loud than normal coding syntax. This pseudocode syntax will first be mapped to normal words by the speech recognition engine's dictation grammar. VoiceGrip commands are recognised as Voice Macros which invoke the Programming editor macro. These editor macros then in turn invoke one of two types of modules: Compilation command module and Translation command module. Pseudocode to Computer code translation of VoiceGrip uses simple deterministic parsing algorithm with three parts. They are translating the input pseudocode command to a programming construct, translating the construct to either a native symbol (if the construct maps with a known variable in the symbols database) or creating a new mapping for an unknown construct for future use. The error rate in VoiceGrip was between 2.7% to 6.6%. The errors mainly arose due to homophonic words and pronunciation of the pseudocode

Ann C. Smith et al, [14], designed the JavaSpeak tool specifically for the visually impaired students. It is basically a code editor with auditory feedback which gives helpful hints about the structure and flow of the program. It gives the user required information about the syntax and semantics of the programming language, in this case Java.

Arno'ld et al, [15], discussed VocalProgramming. It aims at helping veteran developers who have developed RSI to continue coding using only voice commands. The system employs a VoiceGenerator which takes as input Context Free Grammar (CFG) for a specific programming language like BASIC, C or C++ and voice literals from the programming like the names of classes, functions and so on. This is also associated with a set of pronounceable phrases that the programmer can use to edit the code. The output of such a system is a programming environment that can be controlled using only voice-based commands. The system aims to bundle its software with commercially available and popular speech to text converters which ensure that they will have a higher accuracy with time. The issue with this system is the implementation of intended nesting and iteration. It can be solved by normalizing and denormalizing the grammar to make sense of the user's intent.

Yasuhisa Niimi et al., [16], developed a voice-input programming system 'SPOKEN-BASIC-I'. They divided the system into four major components: acoustic, lexical matching syntactic and semantic processors. Spoken words that have no predefined category in this system form their own category. The result of this system was two executable programs that can be created using SPOKEN-BASIC-I in which a total of 282 utterances were processed using the best-first strategy. A total of 85.5% of the utterances were correctly recognised while 6%were incorrectly recognised and 8.5% were rejected. Processing time of this system has been found to be between ¼ to1/5 times of real time processing.

Table1: Comparison on various methods used in programming editors for visually impaired

S.NO	TECHNIQUES	RESULT	PROS AND CONS
1	Adaptive auditory feedback [1]	AAF was found to be more comfortable and satisfactory to use compared to speech only auditory feedback. AAF results in less irritation and joyful experience to visually impaired users using computer system.	Pros: AAF was easier and less irritating to use. Cons: Once AAF enters into non speech auditory feedback it cannot switch back to speech only feedback and user has to manually change modes. It is not optimized to each person's cognitive ability.

2	Dragon Naturally Speaking (DNS), Aenea., Kaldi, SPARK. [2]	Silvius can run in cloud or locally and requires low computing resources.	Pros: The grammar can be customized according to user's need. Silvius can work across all platforms. Cons: Silvius which is a hybrid of DNS and Aenea has poor recognition rate compared to it predecessor.
3	ElectonJS, pyQt4.[3]	The protype implementation of the project is still in progress. The completed system will be will be JAVA voice to code editor that will also help in error correction.	Pros: The editor after it's implementation will work leniently without any ambiguity and support for error is will be provided. Cons: Implementation will be hard as computer syntax is not naturally spoken .
4	Automatic speech recogniser and rule-based syntax parser. [4]	Users were made to work with traditional IDE and VocalIDE. It was found that the average task completion time for traditional IDE is 21.24 seconds and VocalIDE is 13.81 seconds.	Pros: The VocalIDE is comparatively faster than the traditional development environments. Cons: The system design and evaluation were primarily limited to navigation and selection and the system is limited by insufficient speech recognition accuracy.
5	Speech Recognition Engine, Semantic Analysis Engine, Code Play Supervision Engine and Braille Engine. [5]	DICENS system showed 85% accuracy for Speech Recognition Engine.	Pros: The system is easier to use and also helps the programmer by providing alternate solutions that makes the program more logical. Cons: The system failed in high noise areas.
6	Dragon Naturally Speaking (DNS) and Natlink. [6]	A time measurement was taken for common programming tasks using keyboard, voice input and a combination of both. It was found that combination of both voice input and keyboard was faster which is achieved using HyperCode.	Pros: It uses a combination of both keyboard/ mouse and voice recognition for better results. Cons: It does not provide feedback and error correction for users with disabilities.
7	Scratch, Cloud Garden, Java Robot, View. [7]	Myna successfully added static commands to the grammar, commands for editing. Features like scroll bar for navigation, multi-lingual support and grammar customization were	Pros: Myna provides a way for the user to navigate across the screen and also has multi-lingual support. Cons:

		implemented later.	Myna is not 100% voice controlled and can be used only with Scratch IPE.
8	Java Speech API (JSAPI), Voice Activated JAVA and a view named Voice Input. [8]	Users reported that it was easy to learn and master. Addition of red-green light made the system better to use.	Pros: The users stated that using this system was better than typing and it was easy to learn. Cons: The word recognition rate was slow. The system also had time lags.
9	C# and text-to-speech Software Development Toolkit (SDK). [9]	The system was tested with blind and vision impaired users. Both users were able to perform the same task. Blind users preferred applications with keyboard while vision impaired preferred applications with mouse.	Pros: The system provided automatic code completion and code templates for the user. Cons: The system is only implement for C# and can be implement only with the help of Visual Studio .NET for coding.
10	Eclipse IDE plugin named SPEED. [10]	The developed editor SPEED supports in code editing, authoring and navigation. SPEED's recognition delay was between 0.5-0.75 seconds.	Pros: SPEED was easy to learn and provided navigation to desired line where code has to be edited. Cons: Software was found to be three times slow.
11	Continuously spoken words combining both commands and normal English, Context-sensitive commands and automatic generation of abbreviation. [11]	VoiceCode was able to interpret continuously spoken commands and make effective use of templates. It also helped programmers with error correction.	Pros: VoiceCode helps programmer by making use of templates. It helps is error correction and adapts itself not to make the error again. Cons: Though VoiceCode was useful for programmers to program using voice it did not provide sufficient aide to the visually impaired.
12	VoiceXML, Prolog-based Common Gateway Interface (CGI). [12]	Prototype implementation of ALVIN language is still in progress.	Pros: ALVIN language uses mixed-interactive approach to avoid ambiguity. Cons: It is only being implemented for Voice based XML pages.
13	Pseudocode to programming construct mapping using deterministic parsing algorithm. [13]	Only 1111 of the 16734 pseudo-symbols were not matched to the correct native symbols. In 665 of those cases it was found that the native symbol matched by the algorithm was homophonic to	Pros: VoiceGrip covers most of the code editing problems in programming thus helps the users to edit code easily. Cons:

		the correct one.	2.7% to 6.6% error in recognition due to homophones and mis pronunciation in 343 unique users.
14	JAVA Compiler Compiler (JAVA CC) and IBM's ViaVoice, Java Speech Markup Language (JSML).[14]	JavaSpeak has three subsytems namely navigation subsystem, Syntatic Reader Subsytem and Aural Cue Subsytem which being developed.	Pros: JavaSpeak uses JSML to implement different voice tones and emphasis while reading the text. Cons: The focus of the system is to teach JAVA rather than efficient coding.
15	Syntax-directed editor generators and Dragon Naturally Speaking (DNS), Microsoft Visual Studio. [15]	VocalProgrammer will be built with log capabilities that will allow users to capture voice commands as interpreted by voice engine and the voice engines text output.	Pros: The system is developed in a way that all the commands can be accessed using menus. Cons: There are issues in integrating the existing technologies into VocalProgrammer. Since computer language is not spoken, this causes problems while using the system.
16	Acoustic processor, lexical matching processor, semantic processor and syntactic processor. [16]	Two executable programs were written using 'SPOKEN-BASIC-I'. With best-first strategy 85.5% were correctly recognized, 6.0% were incorrectly recognized, 8.5% of the words were rejected out of 282 words.	Pros: SPOKEN-BASIC uses two parsing strategies depth first and best first where the time required to recognize an utterance is 1/4th to 1/5th of the real time. Cons: The system is for BASIC language which is not used anymore.

III. CONCLUSIONS

From the above literature survey, it is noted that a fully autonomous system that enables the visually impaired user to code has not been completely implemented. The issue lies in either the speech recognition or in the error correction module. Speech recognition proves to an area in which improvements can be made. Accuracy and navigation modules are yet to be made user friendly.

Since the advent of speech recognition, steps have been taken to make entire systems voice-enabled. But no system has been fully developed to accurately help the visually impaired user to use the computer system with ease. Coding for the visually impaired using spoken commands is still heavily under research.

REFERENCES

[1]. Shoaib, M., Hussain, I. and Mirza, H.T., 2019. Automatic switching between speech and non-speech: adaptive auditory feedback in desktop assistance for the visually impaired. *Universal Access in the Information Society*, Springer, pp.1-11.

[2]. Abhishek Chand, Subhojeet Chakraborty, Abhinava Anand, Swapnil Dhande, Mansa Mane, & Parag Kulkarni. (2018). CODE BY VOICE USING SILVIUS. *International Journal Of Advance Research And Innovative Ideas In Education*, 4(3), (pp. 2144-2147).

[3]. Singh, A., Tambatkar, G., Hanwante, S., Agrawal, N., Hajare, R. and Khante, K., 2018. Voice to Code Editor Using Speech Recognition.

[4]. Rosenblatt, L., Carrington, P., Hara, K. and Bigham, J.P., 2018. Vocal Programming for People with Upper-Body Motor Impairments. In *Proceedings of the Internet of Accessible Things* (pp. 1-10).

[5]. Lunuwilage, K., Abeysekara, S., Witharama, L., Mendis, S. and Thelijjagoda, S., 2017, December. Web based programming tool with speech recognition for visually impaired users. In *2017 11th International Conference on Software, Knowledge, Information*

Management and Applications (SKIMA) (pp. 1-6). IEEE.

[6]. Maloku, R.S. and Pllana, B.X., 2016. HyperCode: Voice aided programming. *IFAC-PapersOnLine*, *49*(29), pp.263-268.

[7]. Wagner, A., Rudraraju, R., Datla, S., Banerjee, A., Sudame, M. and Gray, J., 2012, May. Programming by voice: a hands-free approach for motorically challenged children. In *CHI'12 Extended Abstracts on Human Factors in Computing Systems* (pp. 2087-2092). ACM.

[8]. LANGAN, D., HAIN, T., HUBBLE, T. and FRØSETH, J.,2009. A voice-activated programming IDE for manually disabled programmers.

[9]. Tran, D., Haines, P., Ma, W. and Sharma, D., 2007, September. Text-to-speech technology-based programming tool. In *Proceedings of the 7th WSEAS International Conference on Signal, Speech and Image Processing* (pp. 173-176). World Scientific and Engineering Academy and Society (WSEAS).

[10]. Begel, A. and Graham, S.L., 2006, September. An assessment of a speech-based programming environment. In *Visual Languages and Human-Centric Computing (VL/HCC'06)* (pp. 116-120). IEEE.

[11]. Désilets, A., Fox, D.C. and Norton, S., 2006, April. Voicecode: An innovative speech interface for programming-by-voice. In *CHI'06 Extended Abstracts on Human Factors in Computing Systems* (pp. 239-242).

[12]. Nichols, M., Gupta, G. and Wang, Q., 2005. Voice-commanded Scripting Language for Programming Navigation Strategies on-the-fly. In *Proceedings of the HCI International 2005*.

[13]. Desilets, A., 2001. VoiceGrip: a tool for programming-by-voice. *International Journal of Speech Technology*, *4*(2), pp.103-116.Springer

[14]. Smith, A.C., Francioni, J.M. and Matzek, S.D., 2000, November. A Java programming tool for students with visual disabilities. In *ACM SIGACCESS Conference on Assistive Technologies: Proceedings of the fourth international ACM conference on Assistive technologies* (Vol. 13, No. 15, pp. 142-148).

[15]. Arno`ld, S.C., Mark, L. and Goldthwaite, J., 2000, November. Programming by voice, VocalProgramming. In *Proceedings of the fourth international ACM conference on Assistive technologies* (pp. 149-155). ACM.

[16]. Niimi, Y. and Kobayashi, Y., 1978, April. A voice-input programming system using BASIC-like language. In *ICASSP'78. IEEE International Conference on Acoustics, Speech, and Signal Processing* (Vol. 3, pp. 425-428). IEEE.