Department of Electronic and Computer Engineering

Electronic Design Project 3A (EDPA301)

Assessment Number 1

Final Proposal (Group)

Project Title: Software for Recording Voice & Video and Time Editing

By

Group Number: P		
Student Surname	**Initials**	**Student Number**
Myeni	**SM**	**22121995**
Mbanjwa	**L**	**22118836**
Ndlovu	**SI**	**22148294**
Naidoo	**D**	**21901944**

PLAGIARISM DECLARATION

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

1. PROJECT OVERVIEW

1.1 Introduction

This project aims to develop a software application facilitating the recording and synchronization of voice and sign language video while enabling users to manually transcribe content and indicate corresponding timestamps within recordings. It addresses a significant gap in existing tools by providing a user-friendly solution for professionals in language transcription services, educational platforms, accessibility services, and content creation industries. By offering synchronized transcription capabilities, the software streamlines processes, enhances accessibility, and improves inclusivity in multimedia content creation and dissemination. With applications ranging from language transcription services to educational platforms and media production, this software contributes to the advancement of accessibility, education, and content creation across various industrial sectors.

1.2 Objectives

The software for voice & video and time editing has the following objectives:

- Develop a user-friendly recording interface for voice and sign language.
- Implement a text input area where users can type transcriptions while watching the recordings.
- Create algorithms for automatic synchronization of typed text with timestamps.
- Enable playback with synchronized transcriptions and editing features.
- Ensure cross-platform compatibility and accessibility.

1.3 Specifications

The software of voice and sign language translation incorporate the following features:

- Record button for initiating voice and video recording.
- Stop and play buttons for controlling playback.
- File management system allowing users to create, save, and delete recordings.
- Additionally, the application should display the duration of recorded content during playback.
- Advanced playback controls, including pause, forward, rewind, and stop, are essential.
- All data to be securely stored in cloud storage, ensuring accessibility from any location.
- Implement a text input area where users can manually type transcriptions while watching recorded content.

2. PRELIMINARY DATA

Commented [MT1]: no need to start on new page

Authors at [1] say that they present a new, linguistically annotated video database for automatic sign language recognition. This new corpus, named RWTH-BOSTON-400, comprises 843 sentences with multiple speakers and distinct subsets for training, development, and testing purposes. They emphasize the necessity of large corpora for evaluating and benchmarking automatic sign language recognition systems. Previous research primarily focused on isolated sign language recognition methods using video sequences captured under controlled lab conditions, often employing specialized hardware like data gloves. These databases typically featured only one speaker, making them speaker-dependent, and had limited vocabulary. To facilitate easy access to the database statistics and content, the authors developed a new database access interface. This interface allows users to quickly browse and retrieve specific subsets of the video database. The authors also provide preliminary baseline results using the new corpora. Notably, unlike other research in the field, the authors make all databases presented in their paper publicly available.

Authors at [1] propose SISLA, an automatic system to bridge the communication gap between hearing impaired individuals and others. SISLA translates speech into Indian Sign Language (ISL) using a 3-phase process: speech recognition, language translation, and avatar representation. The system's multilingual capability enhances efficiency, with trained models achieving minimum accuracies of 91%, 89%, and 89% for English, Punjabi, and Hindi, respectively. Sign language experts evaluated sign error rates, and future enhancements include non-manual SL features and sentence-level translation. Usability testing confirms SISLA's suitability for education and communication among the hearing impaired.

The authors at [1] describe their endeavour in the realm of speech recognition and transcription, motivated by the imperative to enhance communication inclusivity, especially for individuals with hearing impairments. They adopt an agile methodology to develop a transformative system that seamlessly converts spoken language into text, bridging the gap between audible discourse and digital understanding. Python serves as the primary programming language, integrating various libraries such as PyAudio, Librosa, NumPy, DeepSpeech, NLTK, and LanguageTool. Rigorous testing across accents, languages, and real-world auditory environments demonstrates the system's adaptability, while the synergy of hardware and software ensures swift and accurate transcriptions, promising increased communication inclusivity. As their project concludes, the authors assert that their creation transcends a mere technological artifact, resonating with innovation's harmonious anthem. By facilitating communication through spoken word-to-text conversion, their system serves as a bridge that enhances

interaction and comprehension, showcasing technology's transformative potential in fostering communication inclusivity and bridging the gap between audible and digital realms.

Authors [1] highlight a significant disparity between automatic sign language recognition (ASLR) and automatic speech recognition (ASR), mainly stemming from computer vision challenges. While ASR has made substantial progress over the past three decades, ASLR faces unresolved issues. To address this, the authors propose leveraging insights from ASR research to develop their approach. Their system can recognize continuous sign language sentences regardless of the speaker, using features extracted from standard video cameras, eliminating the need for specialized data acquisition devices. They emphasize the application of feature and model combination techniques from ASR, along with the utilization of pronunciation and language models in sign language recognition. These techniques are not limited to ASLR but extend to various video analysis problems where temporal context is crucial, such as action or gesture recognition. The authors report achieving a Word Error Rate (WER) of 17% on a benchmark database comprising 201 sentences and 3 signers. Key terms include Sign Language Recognition, Video signal processing, Pronunciation Model, and Language Model.

Authors [1] explore the potential of using the Kinect depth-mapping camera for sign language recognition and verification in educational games designed for deaf children. They compare a prototype Kinect-based system to their existing CopyCat system, which relies on colored gloves and embedded accelerometers to track hand movements. The adoption of a Kinect-based approach, if successful, could enhance interactivity, user comfort, system robustness, sustainability, cost-effectiveness, and ease of deployment. The researchers collected a total of 1000 American Sign Language (ASL) phrases across both systems. On adult data, the Kinect system achieved sentence verification rates of 51.5% and 76.12% when users were seated and standing, respectively. These rates are comparable to the 74.82% verification rate achieved by the current seated CopyCat system. While the Kinect system requires further tuning for seated use, the results suggest its viability for sign verification.

3. DETAILED PROJECT SPECIFICATIONS

3.1 Operation of the Project

The software for recording voice & video and time editing is a system capable of translating both sign language and voice inputs into textual representations. The system will consist of two main components which are a Sign Language video module and a Voice recording module. Each module will operate independently, allowing users to choose between sign language or voice input for translation. The operation of the system is shown in Figure 3.1.

3.2 Sign Language Translation

The sign language operation includes the following steps:

Capture sign language gestures using a stereo camera due to its ability to copy how human eyes work so that it can provide accurate, real-time dept perception. The proposed camera must not be less than 1080p for the quality of the video to be good. Captured gestures are processed for recognition. Extract relevant features from the gestures, including hand shape and movement. Utilize machine learning or pattern recognition algorithms to interpret the gestures and translate them into text. Upload the video in cloud storage.

3.3 Voice Translation

The translation of voice involves a series of distinct steps to accurately convert spoken language into text, the steps are as follows:

- Capture spoken language input through a microphone.
- Utilize speech recognition algorithms to convert the spoken language into text.
- Store the voice recording to the same cloud storage which was used to store the video for sign language.

3.4 Transcription and Time Stamping

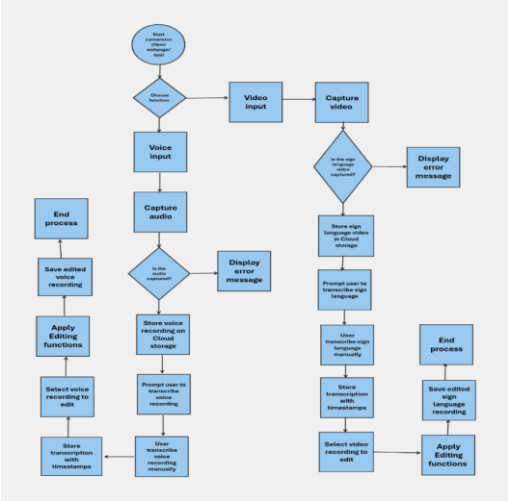
- Transcribe Voice: Use manual transcription to convert voice recordings into text.
- Time Stamp: Associate each transcribed segment with a timestamp indicating when it was spoken.
- Store Transcriptions: Store the transcribed text and timestamps alongside the voice recordings in the cloud.

3.4.1 Manual Annotation and Sign Language Video Annotation

Allow users to manually type what is being said in the voice recording and manually describe the sign language gestures or translate them into text. Link user input to the corresponding timestamp for accurate alignment.

3.5 System flow chart

The flow chart presented in Figure 3.5 illustrates the order of procedures within the system.



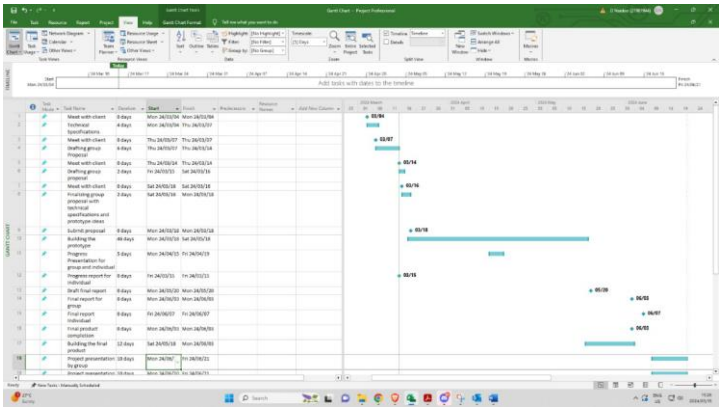
- The user will start by opening the web page/app.
- They will then come across a function that will require them to choose whether they a voice input or sign language video input.
- After choosing the input audio or video will be captured.
- If the audio or video was not correctly captured, an error message will be displayed.
- If the audio was captured, the voice recording will be stored in cloud storage, same thing applies to the video part.
- The system will prompt the user to transcribe voice recording/ video recording manually.
- The transcription will be stored with timestamps.
- If the user wants to edit the voice recording/ sign language video, then they will press the function editing then apply editing functions.
- When the user is satisfied with the editing then they may save the edited video then end the process.

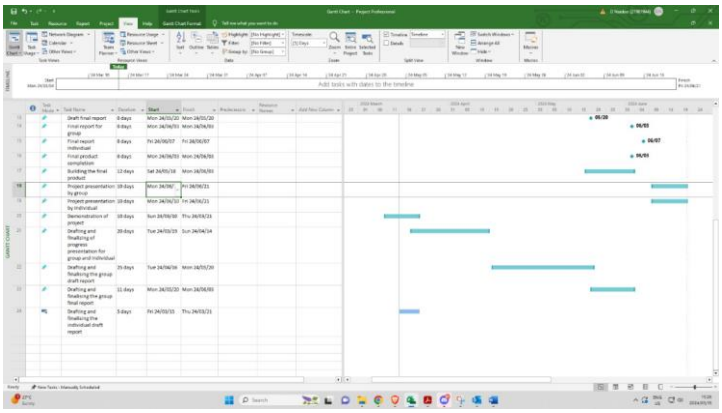
4. TIME MANAGEMENT

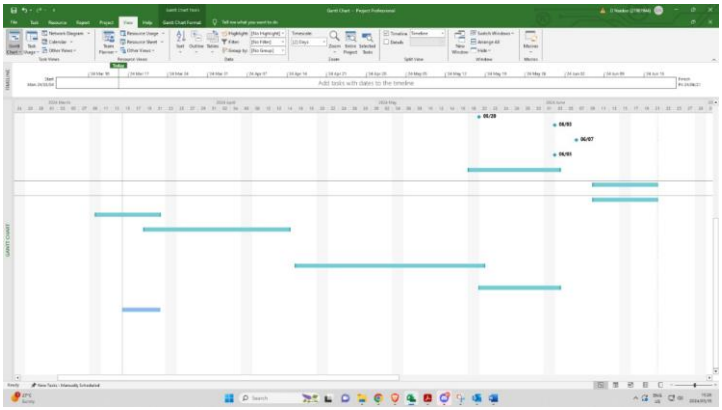
Commented [MT2]: Gantt chart

4.1. Organization

In order to ensure the successful completion of the project within the designated three-month timeframe, effective time management strategies are crucial. This involves meticulously dividing tasks among team members and adhering to a structured timeline. Key sub-tasks such as conducting research and meeting with the client, procuring necessary equipment, constructing the prototype, conducting thorough testing, implementing further development if required, and finally, deploying the project, must be meticulously planned and executed. The Gantt chart presented in Figure X.X provides a visual representation of the project timeline, allowing the team to monitor progress and make necessary adjustments to ensure timely completion. By prioritizing tasks and diligently tracking progress, the team can maximize efficiency and deliver a fully functioning project within the allocated timeframe.







5. MATERIALS AND RESOURCES

5.1 The components

The components or materials needed for this project include a microphone, camera, and speaker. The description, quantity, and amount required for each component are shown in Table 5.1.

Component	Description	Quantity	Amount
Microphone	Must come with the sound card, Frequency range must be 20 to 20 000 Hz, equivalent noise level: 14dB-A, signal/noise ratio must be 80dB be a 16Bit resolution, connects via USB	2	R500 x 2 = R1000
Camera	Stereo pair cameras, 1080p, HD USB 2.0 2MP DS-U02 Web camera, 1920 x 1080 resolution.	2	R500 x 2 = R1000
Speakers	Stereo speakers	1	R500 x 1 = R500
Total		5	R2500

5.2 The Software

The software that might be needed in conducting this project includes:

- Python programming: This software will use Python for core functionality development, ensuring flexibility and compatibility with various libraries.
- OpenCV: OpenCV will capture and process sign language video, enabling gesture recognition and analysis within the application.
- PyCharm: PyCharm will provide an efficient coding environment, aiding in the development and debugging of the software's Python codebase.
- JavaScript: JavaScript will enhance the software's frontend, enabling interactive features for improved user experience and accessibility.
- Matlab: Matlab's numerical computing capabilities will support signal processing tasks, ensuring accurate analysis of voice and sign language data.
- TensorFlow: TensorFlow will empower the software to develop deep learning models for speech recognition and gesture interpretation.

6. POSSIBLE PROBLEMS AND SOLUTIONS

6.1 Voice and Video Quality Issues

- Problem: Poor quality of voice recordings or video footage may lead to inaccurate transcriptions or difficulty in understanding sign language.
- Solution: Implement audio and video enhancement techniques, such as noise reduction, image stabilization, and resolution enhancement, to improve the quality of recordings before transcription.

6.2 Transcription Errors

- Problem: Manual transcription of voice recordings and sign language videos may result in errors or inconsistencies.
- Solution: Integrate automatic speech recognition (ASR) and sign language recognition (SLR) technologies to assist users in transcription, reducing manual effort and improving accuracy. Additionally, provide tools for users to review and correct transcriptions manually.

6.3 Cloud Storage Limitations

- Problem: Limited storage capacity or bandwidth constraints in cloud storage services may hinder the storage and retrieval of large voice and video files.
- Solution: Implement efficient data compression algorithms and utilize scalable cloud storage solutions with ample storage capacity and high-speed data transfer capabilities. Also, consider implementing data archiving and retrieval strategies to manage storage costs effectively.

6.4 Security and Privacy Concerns

- Problem: Storing sensitive voice and video data in cloud storage raises security and privacy concerns, especially regarding unauthorized access or data breaches.
- Solution: Employ robust encryption mechanisms to secure data both in transit and at rest. Implement access controls, authentication mechanisms, and audit trails to ensure only authorized users can access and modify data. Comply with relevant data protection regulations, such as GDPR or HIPAA, to safeguard user privacy.

6.5 Integration Challenges

- Problem: Integrating multiple components, such as voice/video capture, cloud storage, transcription, and editing functionalities, may pose technical challenges and compatibility issues.

- Solution: Use standardized communication protocols and APIs for seamless integration between different system modules and third-party services. Conduct thorough testing and validation to identify and resolve compatibility issues early in the development process. Additionally, consider modular design principles to facilitate easier maintenance and future enhancements.

6.6 User Interface Complexity

- Problem: Complex user interfaces for manual transcription and editing tasks may confuse users and hinder usability.
- Solution: Design intuitive and user-friendly interfaces with clear instructions and visual cues to guide users through the transcription and editing processes. Provide interactive features, such as real-time previews and feedback, to enhance user engagement and productivity. Conduct usability testing with representative users to gather feedback and iteratively improve the interface design.

7. B. ENGTECH LEVEL KNOWLEDGE

7.1 BEng Tech Knowledge aspects

In the project's building, knowledge gained throughout the years of study across the different modules will be applied. The modules that will be applied are:

7.2 TCLT101 (Technical literacy)

This module taught the concepts of technical report writing, project presentations and communication with groups of people. The main concept, which is highlighted in the teaching of the course if referencing in report writing. Referencing is imperative when report writing as it refers to the source of information. This ensures that you are not plagiarizing as the website or article will be referenced adequately, in the IEEE format.

7.3 CPUT101 & CPTP201 (Computer and IT & Computer Programming IIA)

The combination of these two modules taught us how to use a computer and program efficiently. It taught the concepts of writing a program with the correct format and how to write and combine functions to achieve a certain result. This is the main module which is going to help in the building of the software.

7.4 FNTW201

7.5 DSPB301

7.6 FIST201

Commented [MT3]: FIST

Commented [MT4]: FIST

8: REFERENCE MOTIVATION

9. PROJECT ORIGINALITY

10. PROJECT EVALUATION

This project will be a milestone-based evaluation. The software will be built in stages to ensure efficiency in the use of time and to minimize room for errors as one stage cannot begin if the previous stage is not complete. The evaluation will have six stages and it goes as follows:

- Research – the researching stage began when meeting the supervisor. After gaining a general understanding about the project, extensive research commenced. Various websites had a plethora of information but had to be narrowed down to our task at hand. Two programming languages can be used for the project's development, MATLAB, or Python. Python programming was preferred in this case as the software is free and open source.
- Development – development begins after the draft proposal and bill of materials is submitted. Whilst waiting for the materials, development of the software can begin. Teaching the software how to recognize different frequencies for different words and different hand movements for certain words.
- Integration of languages – the native languages will be a part of the machine learning process. Teaching the software different tones and pitches for the various amounts of languages.
- UI (User Interface) development – the development of a user-friendly user interface is imperative. It needs to be efficiently designed so any user of any intellectual level can use the software.
- Beta testing – this part consists of testing the prototype software for functionality and bugs. A third person’s point of view is important for debugging software as small issues unseen by the developer can be brought to their attention.
- Deployment –Deployment of the fully functioning software into the real world is the final step to the project. This is where constant maintenance or updates need to take place as when the software is constantly being used, high volumes of traffic can cause unforeseen malfunctions in the system.

11. CONCLUSION

12. REFERENCES