APPROVAL AND CERTIFICATION

We confirm that this final year project has been done at the University of Rwanda, College of Science and Technology, School of Engineering, Department of Electrical and Electronics Engineering for the award of bachelor's degree in Electronics and Telecommunication Engineering, under the supervision of Dr. Philibert Nsengiyumva and Mr. Gratien Muhirwa. We also confirm that to the best of our knowledge, it is our original work. Contributions from other sources have been properly acknowledged.

Students:

1. Names: **HABIYAREMYE NTWALI JANVIER**
   Enrollment Number: **218007332** Signature:…………….……….………

2. Names: **MALIZA CONSTANTINE**
   Enrollment Number: **218005312** Signature:…………….……….………

3. Names: **MBONIMPA PACOME SIMON**
   Enrollment Number: **218000124** Signature:…………….……….………

Supervisors:

I. Names: **Dr. Philibert Nsengiyumva**

   Signature: ………………………………………

II. Names: **Mr. Gratien Muhirwa**

   Signature: ………………………………………

Head of Department, EEE:

Names: **Dr. Pierre Bakunzibake**

Signature: …………………………………

ACKNOWLEDGEMENT

We would like to express our gratitude towards our supervisors and lecturers for their guidance and constant supervision as well as their mentorship and support in completing the project. We acknowledge the efforts invested by the Department of Electrical and Electronics Engineering at the College of Science and Technology, University of Rwanda, for the sake of ensuring a satisfying curriculum of studies for future engineers.

Our thanks and appreciations also go to our colleagues, classmates, students and people who have willingly helped us with their abilities to deal with difficulties in this project.

ABSTRACT

This report describes our final year project entitled “Implementation of a Real-Time Manuscript Scanner Software for Handwritten Lecturing Digitization”. The report describes a system designed for tackling the problem of additional necessity of a digital whiteboard for remote lecturing purposes, in spite of the ubiquity of other personal electronic devices (smartphones, personal computers, cameras, etc.). The project aimed at addressing that problem by providing an emulation of a digital whiteboard using an A4 white paper, a camera and a computer software (powered by real-time image processing algorithms), thus cutting off the necessity of a digital whiteboard. The objectives of this project targeted specifically a software whose performance (in terms of output video frame rate, image enhancement and obstruction-free output) emulates a digital whiteboard output for the viewer. The research methodology and the software implementation relied on MATLAB toolboxes (Image Processing Toolbox, MATLAB App Designer and MATLAB Compiler) and hardware components: a personal computer and a camera. The outcome of the project was a software that complied with the set objectives but with limitations related to input image resolution, noise and illumination non-uniformity, perspective of the camera and output video file compression. The report concludes by recommending approaches for further improvement.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS AND ACRONYMS

FETT : Feature Extraction Through Time

HSI : Hue, Saturation and Intensity

RGB : Red, Green and Blue

1. INTRODUCTION

1.1. Background of the study

In the worldwide COVID-19 pandemic, many lecturers and students faced a sudden thrust into an online-only working environment, making paper document sharing and physical whiteboard usage difficult. Despite this move to an online-only environment, students enjoy the accessibility to oral and written lectures feature (as it was provided through the usage of physical whiteboards) but are not able to use such interaction. With the availability of digital cameras, smartphones and computers, digitization is becoming a simpler task.

Digital whiteboards (both physical and software based) can be used as an option in online lectures. They provide means of capturing video frames from the board, thus helping the recording and sharing of information. The physical digital whiteboards also provide means of accessibility when it comes to the need of handwriting and/or drawings, where the user interacts as if he/she is drawing on a real board. A challenge that consumers face with physical digital whiteboards is that they impose additional expenses besides other ubiquitous devices (like smartphones, digital cameras and laptops) that are also needed for remote collaborations.

So, we had an idea: what if we could digitize images of a white paper as it is being written on and turn that into a digital whiteboard-like system? The implementation of this idea had to benefit from the availability of simple devices (a smartphone/camera and a personal computer) and provide a computer software that could provide an emulation of a digital whiteboard. The project had to leverage image processing and computer vision programming libraries, and provide a software that can be interfaced to a camera connected to a computer, and be able to scan and enhance in real time the images of a white paper as it is being written on and also removing anything from the images that may appear obstructing any part of the white paper in the image (for example, the writer's hand and its shadows).

1.2. Problem statement

Before the emergence of online-only lecturing, the normal way of teaching was to use a white board or black board so as to provide handwritten presentations and illustrations. The move to the online-only lecturing removed that accessibility that was provided through handwritten lectures. As it is shown in the background of this project, there are gaps left by the "handwritten lecturing digitization" solutions developed to tackle the challenge of remote handwritten lecturing. The developed solutions require manufacturing expensive high-resolution electronic devices (electronic whiteboards / screens, electronic pens, …) as additional devices to be interfaced with simpler personal devices (cameras, smartphones and laptops).

This research aimed at addressing that problem by providing an emulation of a digital whiteboard using an A4 white paper, a camera and a computer software, thus cutting off the necessity of an expensive digital whiteboard. This project also aimed at providing a solution whose performance (in terms of output video frame rate, image enhancement and obstruction-free output) emulates a digital whiteboard output for the viewer, as we had not yet found any known emulator that had attained such performance using only a camera and real-time image processing algorithms.

1.3. Objectives

1.3.1. General Objective

To design and implement a software that captures images of a white paper as the writer is writing and processes the images into a video that shows only the enhanced obstruction-free white paper and what is written on the paper (the pen strokes).

1.3.2. Specific Objectives

- To develop basic image capturing and processing functions using MATLAB's Image Processing Toolbox.
- To determine and implement the most efficient image processing algorithms, to increase the output video's frame rate.
- To apply the developed image processing algorithms in the implementation of a video processing standalone computer program using MATLAB App Designer and MATLAB Compiler.

1.4. Research Questions.

- Can a generalized image processing algorithm be implemented such that it processes raw white paper images (with pen inscriptions) into images of perfect white background with color-saturated pen inscriptions, despite random noises and illumination?
- By referring to the above question, can the efficiency of that algorithm be improved enough to run in real-time, in compliance with the human speed of writing and provide a real-time video of processed images, with a satisfactory frame rate for the viewers?
- Can the aspects of the algorithm be implemented into a graphical user interface software, with features comprehensible to end users, in terms of manipulation?

As we expected to get a positive answer for those question from the research, the hypothesis that had to be tested by this research was:
"It is possible to implement a comprehensible system that captures in real-time a video frames sequence of pen strokes inscription on a white paper and simultaneously, processes them into enhanced images, compensating for any pixel of the white paper obstructed from the camera by a significant opaque object, thus neutralizing the obstruction".

1.5. Scope

1.5.1. Geographical scope

During the execution of this research, the reported image processing experiments were carried out in the environment of the Nyarugenge campus of the University of Rwanda, in Kigali, Rwanda.

1.5.2. Content scope

The project was limited to deploying basic spatial domain techniques of image processing including morphological/binary image processing. The resulting software was expected to provide features limited to image processing (image enhancement, noise removal and removal of parts obstructing the white paper using segmentation), a real-time full screen presentation of processed images for online screen sharing, and also audio and video recording by combining a recorded audio with the sequence of processed images to make a recorded video file.

1.5.3. Time scope

The implementation of this project took four months, as its proposal was approved on the 28th of September, 2021. The final report was expected to be presented in the last week of January 2022. All the activities related to this project were carried along with other academic activities done by the stakeholders.

1.6. Significance of the study

1.6.1. To the organization

This study was expected to not only contribute to the repository of research projects of the University of Rwanda, but also its success was expected to provide a tool (a software) that can be used widely by the lecturers as well as students.

1.6.2. To the researchers

The study that had to be covered by this project would contribute to the knowledge and hands-on skills possessed by the researchers, as it would challenge the researchers to study and apply digital image processing and programming skills, so as to reach the projected objectives.

1.6.3. To the future researchers

This project was expected to highlight critical aspects of image processing in terms of algorithm efficiency. Future researchers would learn from our methods of dealing with the challenge of time efficiency, and also how we dealt with the limitations we met.

2. LITERATURE REVIEW

2.1. Introduction

Even if the idea expressed in this report may sound as new to anyone, we were not the first people to deal with image processing as a tool for real-time emulation of a presentation board. Besides the fame of digital whiteboards, electronic pens or any other hardware developed for lecturing, there are other few solutions reported in various technical papers that tackled the same challenge, using cameras and image processing algorithms. In this chapter we are going to discuss about the solutions we found to be the most relevant, their critics and a conceptual framework we developed from them.

2.2. Review of Related Work

In [1], L. He and Z. Zhang presented a "Real-Time Whiteboard Capture and Processing Using a Video Camera for Remote Collaboration". This project aimed at scanning and enhancing images of pen strokes written on a whiteboard (not a white paper) in real time and filtering out the effect of the obstruction that occurs as the writer (lecturer) is moving between the whiteboard and the camera.

This system had a challenge: since the whiteboard is large, its contrast and light reflection would be influenced by the non-uniform illumination in the room which greatly depended on the weather and/or the direction of the sun rays entering through the windows of the room. In response to this challenge, the project was limited to assuming that the writer (lecturer) is not stationary for more than a period of four video frames, and also that for any small segment of the whiteboard, the pixels that belong to the whiteboard background are typically the majority. The approach proposed by the project was to process each input image by dividing it into a grid of segments, doing the analysis for each segment uniquely by categorizing and updating the obstructed segments and uniquely enhancing the segments impacted by the lighting in the room.

The segments were considered to contain the obstruction if the obstruction lasted less than a duration of four video frames. If the obstruction lasted longer, the obstruction was considered as new pen inscriptions. Therefore, the system failed to remove the obstruction in case the writer (the lecturer) was stationary in front of the camera, for a longer period. This segment-by-segment processing also cost a great time for processing each input image (which was 1.3 Megapixels of resolution), and it led to a low frame rate which was about 1 second per frame, using a computer of 2.4 GHz processor.

In [2], E. Coleshill, D. D. Stacey, and A. Ferworn presented a project entitled “Obstruction removal using feature extraction through time for video conferencing processing”. This project aimed at dealing with the challenge of shadow removal when a presenter obstructs light from a projector to the screen. Using a sequence of video frames taken through time, shadows would be detected and removed, leaving the content of the presentation material on the display without distortion. This would be accomplished by implementing an algorithm that compares intensity variations between consecutive video frames and decide which pixels are obstructed by shadows so as to compensate them with pixels from previous video frames when the section was not obstructed. This approach was called “Feature Extraction Through Time (FETT)”. The Feature Extraction Through Time approach the project relied on was adopted also in our project for the Obstruction Removal process. The aspect found from this project that made it different to ours was that the shadows on the screen are most of the time having an extremely low intensity, when compared to the light reflected by the screen. This indicates that the obstruction removal process in our project has to be a bit complex, since there would be cases where the skin color may impact the intensity of obstruction.

By reviewing [1] and [2], we recognized basic aspecs that the used methodologies relied on. First, the developed algorithms had to ensure that the target object (the whiteboard for [1] or the screen for [2]) was stationary with reference to the camera’s perspective, so as to be able to compare consecutive images and handle the new obstructions that emerged in. For our case, the target object was a whitepaper, which was expected to be moved accidentaly as the writer is writing on it. To ensure that the paper is in a stational position from frame to frame, we had to rely on measuring the correlation between samples taken from each two consecutive frames. A low correlation would indicate that the paper had moved, and thus restart the operations. As it was presented in [3], correlation coefficient calculation requires too much time for computation. Therefore, we had a challenge of developing a more time efficient solution. To emulate the output of a digital whiteboard, we had to also enhance images by increasing the saturation of the pen strokes. As it can be found in [4], conversion from Red-Green-Blue (RGB) to Hue-

Saturation-Intensity (HSI) color models (and vice-versa) requires non-linear computations that take too much time. This also imposes on us a challenge of inventing a time efficient alternative.

As it was reported in [1], the time delay for processing an input video frame would take about one second. This means that the frame rate of the system is approximately equal to 1 frame per second. As we had an idea of improving the performance, we had to set a minimum frame rate that our system had to achieve. Therefore, we referred to the data reported in a forensic related research paper [5], whose objective was to determine the highest speed of handwriting that can be achieved by a person, from a selected sample of people. It was reported that in undisputed police interview records, the highest speeds observed fell in the range 120–155 characters per minute. By conversion, that range is equivalent to about 2 - 2.58 characters per second. For our case, the targeted frame rate was to be set such that it would at least exceed the highest writing speed expected. Therefore, all the parameters used in the program had to be tuned to match an output frame rate of at least 3 frames per second, for a visually convenient image resolution to be determined through experiments.

hand or anything else. The compensation will be done through two segmentation processes: thresholding and masking. By thresholding, the large object (the hand, or something else between the camera and the paper) will be represented by zero intensity pixels, and by masking, those zero intensity pixels will be replaced by their corresponding pixels from the previous video frame when the hand was not hiding them. The output image will be appended to the sequence frames for the output video, and also be used to compensate for obstruction in the next frame. If the user wishes, all this process will be done while recording an audio which will be combined with the processed frames and produce a video of pen strokes being inscribed into a white paper, without showing any other thing else. The user will be able to manipulate the system using the software's Graphical User Interface, which will allow a live full screen presentation of the output.

3.2. Description of Tools Used During the Research and the Implementation.

This project was implemented using MATLAB R2021a edition software installed on a personal computer (a laptop). Computer program written in MATLAB file format were used to simulate and to assess the performance of the proposed algorithm. The used laptop has an Intel(R) Core(TM) i7-5500U CPU @ 2.40GHz processor, a 64-bit Windows 10 operating system, Random Access Memory of 12 GB and a storage of 1TB. The used camera (webcam) had an input resolution of 8 Megapixels, and could also emulate lower resolutions up to 0.23 Megapixels. This project profited from different MATLAB tools, including MATLAB Image Processing Toolbox for digital image processing; MATLAB Timing Functions: tic(), toc() and timeit() functions to measure the time efficiency of algorithms, MATLAB App Designer and MATLAB Compiler to design and compile stand-alone applications for Microsoft Windows operating system.

3.3. Ideal Illustration of the Proposed Main Algorithm.

From MATLAB programming language, a "cell array" is a data structure with indexed data containers called cells, where each cell can contain any type of data. Let us consider the sequence of the captured input frames as a MATLAB cell called "images". In MATLAB

4. SYSTEM ANALYSIS, DESIGN, AND IMPLEMENTATION.

4.1. Introduction.

In this chapter, we are going to describe the proposed algorithms to deal with the basic parts of our implementation. Those parts include processes of Paper Movement Detection, Image Enhancement and Obstruction Removal. For each part (process), we will provide a flow chart indicating what the process should start with, how it will proceed and end. Explanations related to steps, together with sample output images are provided for illustration purposes. Note that the MATLAB code used during preparation for the 4.2, 4.3 and 4.4 sections is illustrated in Appendix A: ”MATLAB Code used in Illustrating the Main Algorithm’s Processes”.

4.2. Algorithm Design for the Paper Movement Detection Process.

To ensure that the obstruction can be removed, the system will need to first make sure that the paper has not been moved with respect to the camera's perspective. The flow chart shown in Fig.7. indicates where the Paper Movement Detection process belongs to in the main algorithm.

the upper left corner of previous image. The results shown below were got from MATLAB after calculating the coefficients.

```
correlation_from_corner1 = 0.4252
correlation_from_corner2 = 0.6665
```

Step 4: Returning a result of a test for a high correlation coefficient.

For this process, the program has to decide whether or not the two correlation coefficients satisfy a minimum correlation coefficient. Normally, a correlation coefficient will range from 0 to 1, such that a value near to 1 indicates a high correlation. By experimentation, we found that in different conditions, 0.65 is the minimum coefficient of correlation that resulted when the paper was not moved. Therefore, this process will give a true answer if each of the two correlation coefficients satisfies the threshold condition of being greater than 0.65.

4.2.2. Algorithm Improvement.

Given that the calculation of correlation coefficient takes too much time due to its high computational complexity [3], we developed a new approach to tackle the challenge of time efficiency, as a real-time constraint. The new approach proposed had to start by thresholding the equalized pixel samples from the papers' upper corners. As various thresholding models rely on contrast enhancement ([6] and [7]), the image equalization will enhance the contrast significantly, to prepare the image for the thresholding. The process had then to compare the resulting binary patterns using the binary XOR operation. The resulting data would indicate which pixels have changed, and those that didn't change, if the paper had been moved. By experimentation, different trials showed that the maximum percentage of pixels that changed when the paper was not moved was 30%. This was due to the presence of noise that varied in different cases.

2. **The “White Paper Trimmer Sensitivity” slider**: indicates how much the user wants to trim the output video with respect to the white paper position. As the user manipulates it, it returns a value between 0.1 and 0.9 which indicates the threshold percentage of white pixels the software has to detect on each of the four edges while trimming. By default, the value returned is programmed to be 0.3.

3. **The “Movement Detector Sensitivity” slider:** indicates how sensitive the system should be towards resetting in case the paper is moved with respect to the camera position. It returns values between 0.1 and 0.9 which indicates the percentage of pixels that have not changed from one video frame to another, by referring to the pixel samples trimmed from the upper corners of those frames. By default, the value returned is programmed to be 0.7.

4. **The “Enable Pen Strokes Enhancement” Checkbox and the “Present” and “Present and Record” buttons:** The checkbox enables the user to set if he/she needs that the system should include image enhancement processes (that might delay the frame rate), or not. For the two remaining buttons each enables the user to enter into a full screen presentation of his/her writings. In addition to that, the second button allows to record the voice and the video frames and later combines them into a video file.

As it is stated above, the final software can record the video along with the audio. MATLAB has the VideoWriter object that allows to process digital images as video frames, select the frame rate and combine them into a video file. MATLAB also has the audiorecorder object that can be used to record audio and write it into a computer file. MATLAB also supports any kind of webcam (camera) provided that it can be recognized by the computer’s operating system. The webcamlist object shows a list of webcams connected to the computer, the webcam() and snapshot() functions respectively, allow you to select the webcam and to take photos instantly. On the graphical user interface we built, the user can choose to present the output and also record the stream at the same time. The two buttons: "Present" and "Present and Record" allows to Only Present or Present and Record, respectively.

5. EXPERIMENTS’ RESULTS AND RELATED DISCUSSIONS.

In this section, we are going to compare the results of the research and the outcome we expected. The hypothesis of our research stated, **“It is possible to implement a comprehensible system that captures in real-time a video frames sequence of pen strokes inscription on a white paper and simultaneously, processes them into enhanced images, compensating for any pixel of the white paper obstructed from the camera by a significant opaque object, thus neutralizing the obstruction”**.

The expected result of this project was a graphical user interface software that would capture images (frames) of a white paper as the writer is writing and process the images to show only the enhanced white paper and what is written on the paper (the pen strokes). The system had to use segmentation processes to remove any part which was obstructing any part of the white paper, including the writer’s hand. The system was also expected to allow the user to record a video along with the audio, and the video will only be showing enhanced pen strokes being inscribed on a purely white paper.

While the result showed that the results matched the most parts of the hypothesis, there are some few elements that did not comply. In this section we are going to review critical aspects that characterize the limitations of the performance of our final software.

5.1. The output is not always as appealing as expected.

The imperfect elegance of the output video frames is one of the aspects a user can perceive while manipulating our software in differently illuminated locations. This is because our system relies on a long-generalized sequence of image processing algorithms that include filtering out noises, detecting edges of pen strokes, sequences of morphological processes, etc. All those were designed in a generalized way for the purpose of protecting the thin pen strokes, while also removing the thinner noises, and the huge obstruction (the hand) in different situations. Due to random fluctuations of illumination and noises that change time by time and location by location, some details of pen strokes can disappear, and also unexpected noises may appear. Also, by referring to the fact the system favors resolutions less than two megapixels (for the sake of preserving the frame rate, for Intel(R) Core(TM) i7-5500U CPU @ 2.40 GHz processor computer), the output video may not look attractive for large screens (large as digital whiteboards, for example) viewers, as we expected to extend up to 8 megapixels.

a resolution lower than 0.1 megapixels. Therefore, we searched for another alternative solution to this problem out of MATLAB’s toolboxes. The other alternative we developed was to use a Python script that relies on a Python library called “moviepy” [13]. The library provides functions that directly provide video editing solutions, with a satisfying compression. This alternative developed forces the user to install Python as well as the moviepy library in order to be able to combine the recorded audio together with the recorded video in a single video file. This also imposed another limitation, since python can not interact with MATLAB compiled applications. Therefore, the automatic combination of the two files (audio and video) together will be available for a user using a non compiled version of our program, running in an installed MATLAB R2021a edition.

6. CONCLUSION AND RECOMMENDATIONS FOR FUTURE RESEARCH.

The general objective of this project was to design and implement a software that would capture images (frames) of a white paper as the writer is writing and process the video frames to show only the enhanced white paper and what is written on the paper (the pen strokes). The research methodology and the software implementation relied on MATLAB toolboxes (Image Processing Toolbox, MATLAB App Designer and MATLAB Compiler) and hardware components: a personal computer and a camera. The outcome of the project was a software that complied with the set objectives but with limitations related to input image resolution, noise and illumination non-uniformity, perspective of the camera and output video file compression.

As it was explained in the previous chapter of this report, the overall output was not attractive as it was expected. The overall report suggests that the resolution input for which the system will perform its best is 1 megapixel. For further research, we recommend trying different alternatives that can support high resolution images while also conserving the needed frame rate. This would require using a computer with a more advanced processor to reduce the execution time. Another approach to recommend for further research is testing the system with various users of different skin color. This would give a more assurance on how the system is able to generalize and detect obstruction.

Regarding the problem of re-adjusting the processed images sequence with the change in position / perspective of the camera to the paper, we may recommend an implementation of geometric transformations, so as to re-align video frames whenever the paper is moved, and thus remove the obstruction easily, and remove the burden of resetting. As the time did not allow us to tackle the problem of video file compression, we would also recommend future researchers to consider developing other video compressing libraries different from those reported in the previous chapter of this report.

To ensure that the research schedule had to match with the researcher's academic calendar, some of the remaining challenges were not solved. Those challenges, as are reported in the previous chapter, required refinement after testing the software in different conditions of noise and illuminations, and on users with different skin colors. For a more exhaustive research, further investigations to tackle those challenges are suggested.

REFERENCES

[1] L. He and Z. Zhang, “Real-time whiteboard capture and processing using a video camera for remote collaboration,” *IEEE Transactions on Multimedia*, vol. 9, no. 1, pp. 198–206, Jan. 2007.

[2] E. Coleshill, D. D. Stacey, and A. Ferworn, “Obstruction removal using feature extraction through time for videoconferencing processing,” *Innovative Algorithms and Techniques in Automation, Industrial Electronics and Telecommunications*, pp. 131–133, 2007.

[3] A. Mahmood and S. Khan, “Correlation-coefficient-based fast template matching through partial elimination,” *IEEE Transactions on Image Processing*, vol. 21, no. 4, pp. 2099–2108, Aug. 2010.

[4] S. Zhi, Y. Cui, J. Deng, and W. Du, “An FPGA-based simple RGB-HSI space conversion algorithm for hardware image processing,” *IEEE Access*, vol. 8, pp. 173838–173853, Oct. 2020.

[5] R. A. Hardcastle and C. J. Matthews, “Speed of writing,” *Journal of the Forensic Science Society*, vol. 31, no. 1, pp. 21–29, Jan. 1991.

[6] P. Kandhway and A. K. Bhandari, “An optimal adaptive thresholding based sub-histogram equalization for brightness preserving image contrast enhancement,” *Multidimensional Systems and Signal Processing*, vol. 30, no. 4, pp. 1859–1894, Feb. 2019.

[7] S. N and V. S, “Image segmentation by using thresholding techniques for medical images,” *Computer Science & Engineering: An International Journal*, vol. 6, no. 1, pp. 1–13, Feb. 2016.

[8] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 3rd ed. Upper Saddle River, New Jersey: Pearson Prentice Hall, 2008.

[9] R. C. Gonzalez, S. L. Eddins, and R. E. Woods, “Fundamentals,” in Digital image processing using MATLAB, 2nd ed., S.L., California: Gatesmark Publishing, 2009, pp. 65–71.

[10] S. Attaway, in *MATLAB: A practical introduction to programming and problem solving*, Oxford, United Kingdom: Elsevier, B/H, Butterworth-Heinemann, 2019, pp. 475–488.

[11] R. Lee and R. Bachnak, “The Matlab Compiler Suite: M files to c/c++ executable programs,” *2002 American Society for Engineering Education Annual Conference & Exposition*, p. 7.1164.2, 2002.

[12] P. S. Mbonimpa, J. Habiyaremye Ntwali, and C. Maliza, “RealTimePenStrokesScanner,” *GitHub*, 21-Jan-2022. [Online]. Available: https://github.com/pacomesimon/RealTimePenStrokesScanner/tree/main/Real_Time_testing. [Accessed: 22-Jan-2022].

[13] M. Beacom, T. Burrows, and Zulko, “Moviepy,” *PyPI*, 05-Oct-2020. [Online]. Available: https://pypi.org/project/moviepy/. [Accessed: 21-Jan-2022].