PROJECT SUMMARY – SPECTRUM ANALYZER ANALYSIS TOOL – TEAM 4

Seema Kumaran, Alexandro Pinion, Jeffrey Amakihe, George Vendetta, Chris Williams

November 14, 2023

OVERVIEW

1. Background

- *"Military Flight Test Ranges have a variety of Radio Frequency (RF) threats used for testing Aircraft Electronic Warfare (EW) Systems during flight. Flight Test Ranges have a tracking station that is used to record RF transmissions of the threats for testing and calibration purposes. The transmissions are recorded via video from a Spectrum Analyzer and are provided to the system engineers for the System Under Test (SUT) on video. Currently, analysis of the data requires engineers to actively watch a selection of video data to verify frequency and amplitude of a specific Threat System." [1]*

2. Purpose

- *"Reduce the manual time and toil needed from an analyst or engineer to review spectrum analyzer videos containing RF signal frequency and amplitude data for electronic warfare threats. The intent of the automated analysis is to identify out of spec frequency and amplitude data as well as timeframe of the identified event." [2]*

3. Project Findings

- *Given a video where a screen is in a fixed position and a known pattern of displaying live graphs, it is possible to train open-source tooling to extract the relative values of the position of the screen*
- *Analog values can also be processed through OCR[5] from the screen to calculate the actual values of the graph in the frame*
- *Processing time increases with the complexity of data to be read from the screen. In the case of a spectrum analyzer the complexity of applying the on-screen values to calculate the actual frequency and amplitude based on applying the textual data to the graph tends to double the processing time*

4. Algorithms Explored

- *Tesseract-OCR[5]: Google python library and windows executable used to recognize onscreen values for center frequency, span frequency, and reference level*
- *Identifying areas where text may be OCR[5]'d*
- *Carving text locations out using OPENCV[4] library and OCR[5]'ing those locations with Tesseract-OCR[5]*
- *Using OPENCV [4] to adjust frame to greyscale to simplify processing*
- *Using OPENCV[4] to compare template image to frame and excluding the noise outside of that area*
- *Using a gaussian de-blur technique with OPENCV[4] to sharpen the image*
- *Using a dictionary of common misconfigured labels of reference data on screen to reduce loss of OCR[5] data*
- *Using OCR[5] values to calculate actual values at relative positions based on the configured divisions on screen in both X and Y axes*
- *Determining frequency level(khz, mhz, ghz) based on data retrieved in OCR[5]*

5. Issues

- *UI Integration a little clunky and could be made more visually appealing*
- *Processing speed is a function of video time and amount of data to scrape*
- *Refactoring would reduce cyclometric and cognitive complexity*

6. Future Recommendations

- *Improved performance may be yielded by a couple different strategies:*
  1. Break the video up by signal boundaries and multithread across different cores
  2. Re-platform into a compiled language that executes more closely to the CPU
- *Explore tooling to see if there are differences in accuracy or performance*

- *Add a template image configuration screen to allow the user to define the graph area visually*
- *Add a configuration import and export screen to enable the user to save and reuse configurations for different spectrum analyzers*
- *De-couple the configuration code into a separate entity*

7. Lessons Learned

- *Frequent meetings, clear roles and responsibilities, and communication are the keys to a successful project*
- *Be ready to pivot away from toolsets that are unwieldy or don't yield intended performance or results*

8. References

[1] E. Dayton, "Request for College of Computing and Software Engineering Capstone Project Proposal Section I -Company and Mentor Information Section II -Project for Which Students Will Participate."

[2] C. Williams, "Spectrum Analyzer Software Requirements Document (Draft -SWE 7903 Spectrum Analyzer Capstone -Team 4)," Sep. 2023.

[3] Source Code Repository: https://github.com/chwillchwill/SWE7903-SpectrumAnalyzerTool-Team4.git

[4] "OpenCV[4][4]: OpenCV[4][4] modules," *docs.OpenCV[4][4].org*. https://docs.OpenCV[4][4].org/4.x/index.html

[5] "tesseract-OCR[5]," *GitHub*. https://github.com/tesseract-OCR[5]