Food Safety in the Digital Era

Sponsor: Decagon Devices
Mentor: Fred Lionetti
Kyle Owens, Nick Hayward, Connor Jackson, Adam Aboutalep

Motivation
The users of laboratory devices have traditionally stored copies of their data in bound notebooks written by hand. While this worked well in the past, the volume of data collected has become cumbersome. Our project’s main goal is to alleviate the burden of data collection for laboratories while increasing the amount of access to this data.

Goals
● Review and evaluate existing OCR solution
● Design and implement OCR process to extract data
● Build native applications to capture and relay images to server
● Investigate image processing in the Linux environment
● Investigate image processing in the iOS and Android ecosystems
● Design, implement and test end-to-end solution for capturing user data in realistic laboratory settings

Architecture
A client server architecture is used for communications between handheld devices and the OCR processing server. Android and iOS native applications communicate with a RESTful API deployed on a Ubuntu Linux server.

The API is served using an Nginx web server for caching, reverse proxying, and load balancing. The Nginx server directs requests to a small cluster of uWSGI worker processes.

These worker processes run using the Django framework making calls in a custom library that makes extensive use of OpenCV and Tesseract. When the reading extraction is done, the data comes to rest in a PostgreSQL database.

Handheld applications use background processes to start and finish communication with the server. This keeps the main thread’s memory under control for a better user experience.

If the handheld applications can’t reach the server via cellular networks or Wi-Fi networks, the handheld will save the image locally to be uploaded later when the server is reachable.

Optical Character Recognition
Image contours are determined and search using a best approximate fit to the shape of the screen.

The image is cropped down to the target text area and verified using a fast template match.

Denoise algorithms are applied including a Non-Local Mean Denoising and Bilateral Filter. The bilateral filter was used to preserve and sharpen edges.

To remove and normalize lighting, CLAHE was used with a large region of comparison to minimize effects of inconsistent lighting.

An Otsu Threshold was applied to binarize the image. Erosion and Dilation operations remove artifacts resulting in a cleaner image.

With a binarized image of the target text, Tesseract is run with the image. With the resulting reading, OCR is complete.

Glossary
API - Application Programming Interface
Bilateral Filter - Nonlinear edge preserving noise reducing filter
CLAHE - Contrast Limited Adaptive Histogram Equalization
Dilation - Increase size of bright area while diminishing dark area
Erosion - Increase size of dark area while diminishing light area
OCR - Optical Character Recognition
OpenCV - Open Source Computer Vision
Otsu Method - Algorithm to determine middle point of histogram distribution
REST - Representational State Transfer, architecture design paradigm
GPGPU - General purpose graphics processing unit

Mobile Application
The mobile application was developed in Xcode using Swift and Android Studio using Java. Features of the mobile applications include logging in with credentials, taking a picture, uploading it, receiving the reading from the server, and saving pictures in the event of no internet connectivity.

Impact
Users will be able collect and analyze data much more efficiently. Although this project solves one problem, it is clear that this solution could be applied to a broad range of data collection. This project improved upon an older non-technical solution, taking advantage of the proliferation of mobile devices. New solutions will continue this trend as technology continues to develop. By storing critical quality assurance readings in a highly accessible manner, safer products with a higher confidence in shelf life can be shipped directly to customers.

Road Ahead
OCR on the phone
OCR processing on the phone will allow complete functionality with no internet access and will provide a faster reading as a result.

Server Performance
Profile to find computationally intense bottlenecks. Move to an asynchronous worker model. Migrate image processing and OCR to GPGPU.

Improved OCR Performance
Build larger test database of real world images. Work to mitigate extreme lighting.

Training Tesseract for Seven Segment Display
Develop image library to train Tesseract to better recognize seven segment display characters.

Road Ahead

Special thanks to Sakire Arslan Ay and Fred Lionetti for everything they did for us.

Obiwan