EN 601.456 Computer Integrated Surgery II Project Final Report

Vital Monitor and ID Detection through Machine Vision for Improving EMS Communication Efficiency

Robert Huang, Biomedical Engineering Senior, rhuang22@jhu.edu

Mentors:
Dr. Nick Dalesio, MD AirSupport Co-founder, Pediatric Anesthesiologist
Dr. Laeben Lester, MD AirSupport Co-founder, Emergency Physician
Dr. Mathias Unberath: Assistant Professor in Department of Computer Science

April 2021

Contents

1 Introduction 3

2 Setup 4

3 Technical Approach 6

4 Tests and Results 11

5 Discussion 13

6 Progress Evaluation 14

7 Conclusion 17

8 References 18

1 Introduction

1. **Background**

   Smart glasses, while themselves are not a recent invention, are only just being implemented into healthcare environments. These glasses are most often used by surgeons or emergency medicine (EMS) workers to record and live-stream video feed from, respectively, surgical rooms and on-the-field operations [1] [4]. This feed is fed through a smart phone and then sent over the internet to remote healthcare workers, who can either provide their advice and assistance for the current patient, or, if they are trainees, learn from watching these healthcare procedures [1].

2. **Motivation**

   Currently, only the raw live footage is used. Therefore, there is opportunity in taking this feed and running it through artificial intelligence in order to streamline healthcare processes to ultimately increase efficiency of patient care. In particular, there are two objectives to improve EMS communication efficiency:

   (a) **Automatically extract information from IDs:**

   During a typical emergency medical response, 25-65% of the patient's care is spent documenting [5], with 1-10 minutes of it being spent on obtaining and recording 'simple' information of both the patient and health care workers, such as names, birth dates, ages, addresses, or ID numbers [2]. Therefore, streamlining patient identification by pulling key information from standard documents like driver's licenses means that medics spend less time on paperwork and more time treating patients. EMS users should be able to bring an ID in front of the smart glasses camera for 1-3s and complete simple information documentation. Consequently, this project seeks to implement a deep learning and optical character recognition (OCR) algorithm that detects identification and extracts the relevant information.

   (b) **Provide View of Data Monitors Remotely and Extract Vitals Information:**

   During an emergency response on-the-field, it is often necessary to obtain advice, assistance, and/or clearance from a remote physician to carry out a procedure. To improve care, the remote physician should have direct access to the information available from medical devices used by the medics, especially vitals signs monitors. With visual confirmation of vitals, physicians are 2-3x more confident and give treatments 2-3x faster [4]. Furthermore, without clear view and recording of vitals, 43.4% of information can be lost, resulting in repeated procedures once the patient reaches the hospital [4]. Therefore, it is pertinent to obtain the visuals of and the values displayed on these data monitors. Unfortunately, while certain data monitors that come packaged with smart glasses will provide direct access of the data to physicians, all in all there are poor interoperability standards [3]. However, the human-readable displays of these devices are always readable when in view of the smart glasses camera. Therefore, this project aims to take the video feed of these devices and incorporate machine vision to provide physicians direct visual access to data monitors on the field, and to automatically extract and store the vitals data.

3. **Objectives**

   The goal of this project is to develop the elements of artificial intelligence that will take the video feed and perform two objectives. The main objectives in this project will be to extract and input information from identification and to provide and extract direct visuals of on-the-field data monitors to remote health care workers. Extracting information from identification will improve on-the-field outcomes by moving time spent on documentation to patient care. Providing visuals will improve on-the-field outcomes by increasing confidence and treatment speeds, and by reducing repeated procedures. Furthermore, this project's success will serve as an initial assessment into the viability of artificial intelligence in smart glasses in a health care environment.

2 Setup

1. **Package Requirements** All code was written in a Python 3.8, 64-bit environment. The packages used were:

   - numpy 1.19.4 (pip install numpy)
   - cv2 4.2.0 (pip install opencv-python)
   - scikit-image 0.18.1 (pip install scikit-image)
   - pytesseract 0.3.7 (pip install pytesseract)
   - imutils 0.5.4 (pip install imutils)
   - argparse (pip install argparse)

2. **Datasets**

   Two datasets were used to train respective detection algorithms, one containing driver licenses and one containing the vital monitors, were generated from online or MDAirsupport sources. Their respective documentations can be found linked in the Wiki. Below contains an overview.

   The license data set contains thousands of data augmented, natural, and negative (does not contain) images of IDs. The data augmented images were generated by taking flat driver license images from an article titled "What a Driver's License Looks like in Every State" [6] and transforming them, and superimposing them on top of background images found in the Stanford Background Dataset [7]. The natural images were generated by processing the videos found in the MIDV500 dataset [8], a dataset containing short 30 frame recordings of identity documents. Finally, negative images were obtained from the Stanford Background Dataset [7] and the Open Image Dataset [9]; there are outdoor images, and images containing objects that the algorithm may confuse as licenses.

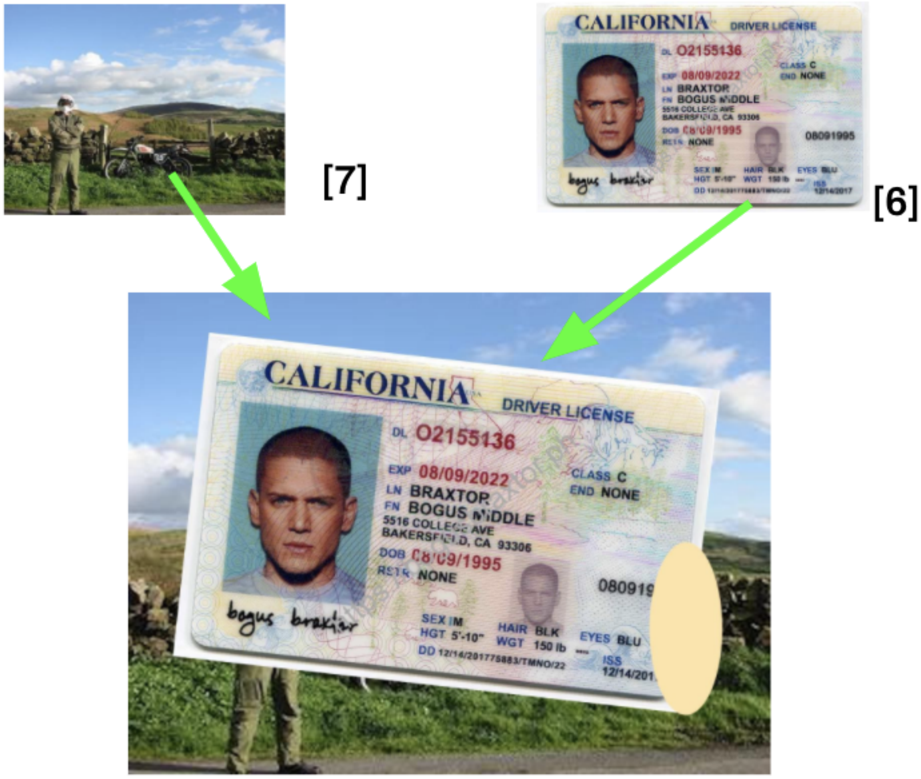


Figure 1: ID transformed and 'pasted' on background to create custom data. Note that the size of the ID and the presence of a tan 'obstruction' is representative of a potential user's situation.

The vitals monitor data set contains thousands of data augmented images of the two desired vitals monitors, the Zoll and the LIFEPAK. As in the case of the IDs, the initial images of these monitors were randomly transformed, and their color and brightnesses were altered. The same negative images found in the license dataset were also used here to reduce detection of potential confusers.

3 Technical Approach

1. **Overview**

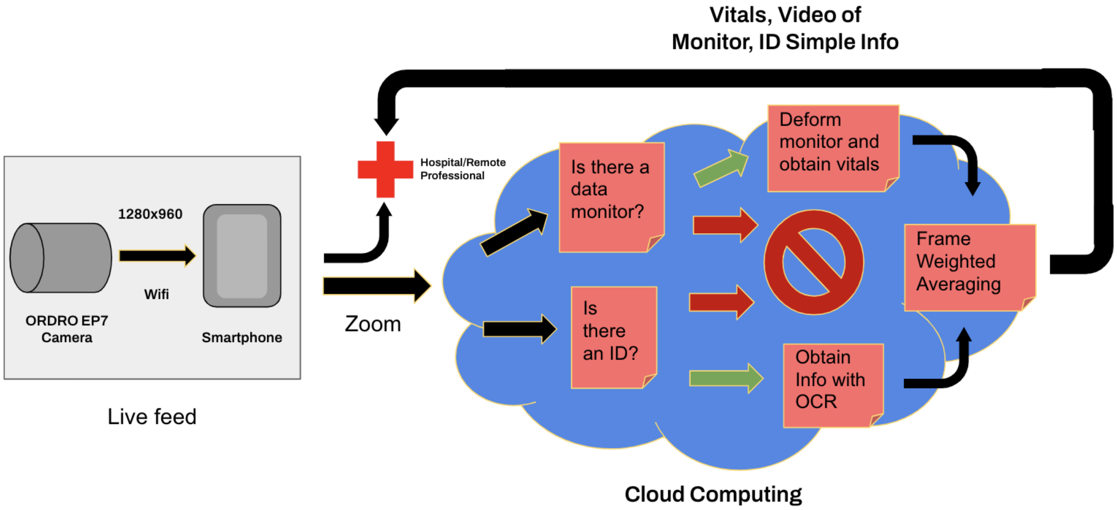


Figure 2: Ideal overall workflow for the detection and extraction of vitals monitors and IDs.

The ideal overall workflow of the smart glasses is shown above. Video feed of images of size 1280×960 will be sent through Wifi. From there, these images will be sent to the remote healthcare provider Zoom stream and then over the internet into the cloud, where the algorithm will process the video. For this initial project, video feed of the target objects was recorded from the remote health care provider's Zoom stream and then the input into the algorithm to produce an accurate estimation of the desired ID or vital information.

Prior to reaching the algorithms, the video feed will be preprocessed through resizing, normalization, and gaussian blur to ensure standardization and removal of noise. Then, for either algorithms, it will first check if its respective object is within the image. If the object is detected, the algorithms will process the images respectively.

If identification is detected, the algorithm will crop out and deform the identification, grab the text from the resulting image, and categorize the text into meaningful values (such as Name, Date of Birth, etc.).

If a data monitor is detected, the pixels of the monitor will be cropped out, and the image will be deformed such that it appears head-on. This image will then be sent back to the smartphone, and the image will be appended to the video feed and sent to the remote health care worker. At the same time, vitals information will be grabbed from the image and categorized.

After the individual processing, the results from both algorithms will be input into a frame weighting algorithm. Every frame generated, while the object of interest is within view, will, depending on their weight, contribute to a running overall estimation of the textual results. For identification, weights are based on how in-focus the frame is. For monitors, weights are based on how in-focus they are, and their time since detection (so that, if a heart beat changes, its estimation will change too).

2. **Identification Extraction Flow and Descriptions**

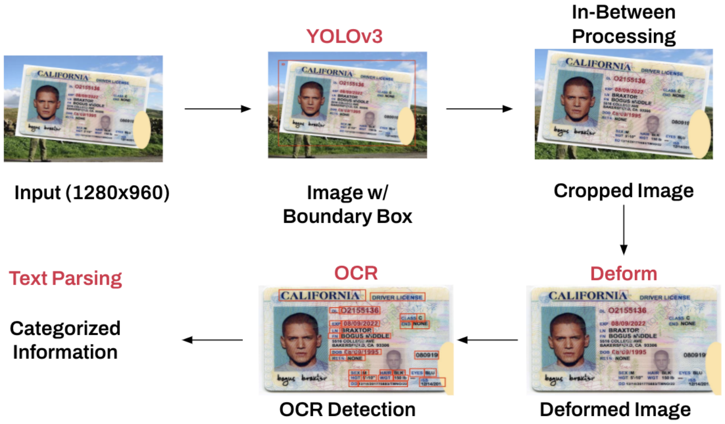


Figure 3: Illustration of stages during the extraction of IDs.

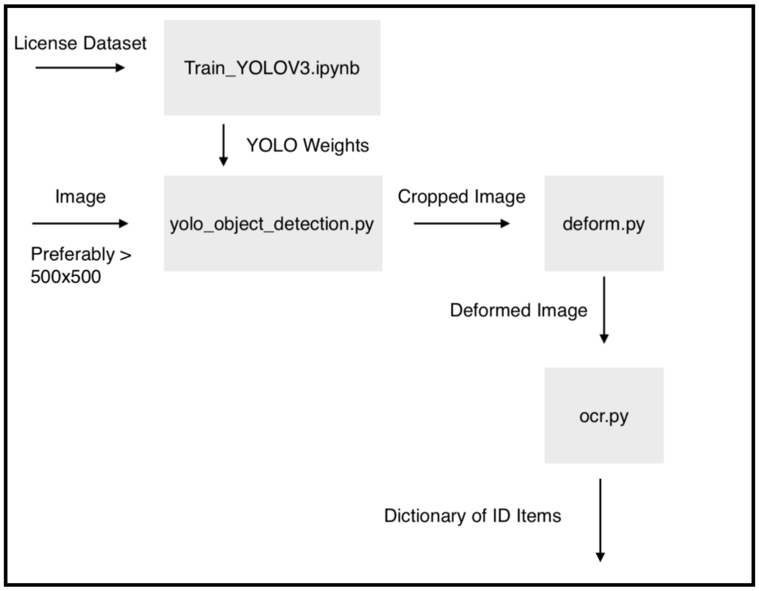


Figure 4: Flow Chart of Files and File Types for both Information Extraction Algorithms.

There are **three main steps** in order to detect and extract information from identification from camera feed.

(a) **Detect the presence of an ID**. In order to detect the presence of identification, a YOLO Deep Learning Framework, trained on the driver license dataset, is run on the image. The network draws bounding boxes around the detected driver licenses.

(b) **Detect the spatial orientation of an ID and deform**. To optimize optical character recognition, the identification's image is determined using Hough Transform Edge Detection and deformed such that it appears head-on with no rotation.

The images are preprocessed with Canny Edge Detection to assist Hough Transform in detecting edges. The contours are then detected and filtered based off certain characteristics that are indicative of the ID’s contour:

- % Area: 75-100% of area
- # of Corners: 4-5 corners
- Aspect ratio: 0.5-0.6

Once the ID’s contour is detected, its corners are obtained, and the ID is modeled as a rectangle. This rectangle is deformed to a head-on view.

(c) **Read and categorize information on the ID**. Tesseract OCR in Python is used to extract text from the license. The OCR algorithm will first preprocess the Post-YOLO deformed image using the OpenCV functions Binarize, Gaussian Blur, TopHat, BlackHat, and Threshold Local. It then finds contours, which are labeled as ‘Regions of Interest’. The python Tesseract OCR will then be run on the regions, and the text that is obtained will be returned in a list of text. The location of the text will also be recorded. Generic Text Parsing is then used to sort the information into desired categories, such as name, birth date, etc. The text parsing algorithm uses heuristics and regular expressions to categorize the received list of data from the OCR algorithm:

- Sex: “F” or “M”
- Dates:
- Expiry: Date in Future
- Birth: Date furthest in past
- Issued: Date in Between
- States:‘Nevada’,’California’,etc.
- Address: Fill up Two - Three lines, ‘St’,’Rd’
- Height: 4/5(’)?-(0/1)?#(“)?
- Weight: ###
- Eye: BRO, etc.
- Hair: BRO, BLK, etc.

Locations of ambiguous items, such as Eye and Hair, are referenced on an ‘Atlas of state IDs’ to categorize them.

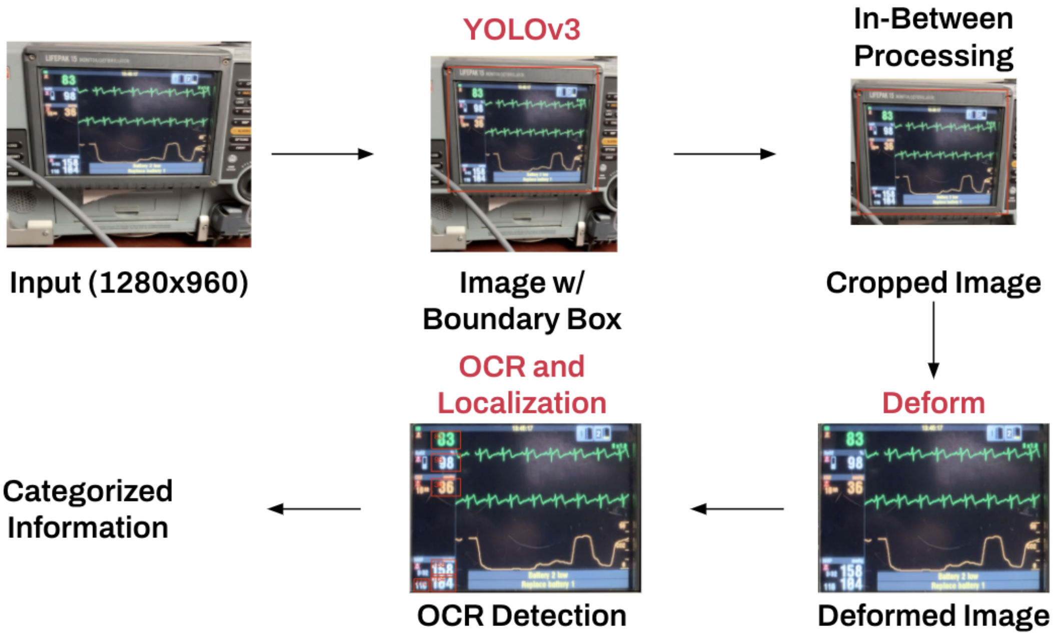


Figure 5: Illustration of stages during the extraction of vitals.

3. **Vitals Extraction Flow and Descriptions** There are **three main steps** in order to detect and extract information from vitals from the camera feed:

   (a) **Detect the presence of a monitor**. A YOLO Deep Learning Framework, trained on the vitals monitor dataset, is run on the image. The network will draw bounding boxes around the detected vitals monitors.

   (b) **Detect the spatial orientation of the monitor and deform**. To optimize optical character recognition, the identification's image will be determined using Hough Transform Edge Detection. The deformation algorithm takes the closely cropped image of the vitals monitor as an input. The images are then color separated in order to isolate the blackness of the vitals monitor that contains the vitals information. Similar in the driver license case, these new separated images are preprocessed with Canny Edge Detection and run through Hough Transform to detect the edges of the vitals monitor. The contours are then detected and filtered based off certain characteristics that are indicative of the vital monitor's contour:

      - % Area: 75-100% of area
      - # of Corners: 4-5 corners
      - Aspect ratio: 0.3-0.5

      Once the vital monitor's contour is detected, its corners are obtained, and the monitor is modeled as a rectangle. This rectangle is deformed to a head-on view.

   (c) **Read and categorize information on the vitals monitor**. Tesseract OCR is used to extract text from the vitals monitor in a similar fashion as in the ID case. Here, text parsing is not required, and solely the location of the detected numbers is used to sort the information into desired categories, such as heart rate, blood pressure, and oxygen saturation.

4. **Weighted Frame Averaging** The implementation of the weighted frame averaging algorithm was inspired by Petrova et al [10]. In order to increase the accuracy of the program, reduce the effect of glare, and exploit the fact that multiple frames are used per object, the algorithm runs an integration algorithm that takes the following steps:

   (a) The algorithm first takes the categorical string outputs of multiple frames (ex. many frames report a Birth Date) and represents them as a matrix.

   - Every character, x, is represented as a vector of probabilities, each probability indicating a certain letter.

     $$x = \left(x_1, x_2, \ldots, x_K\right) \in [0.0, 1.0]^K, \quad \sum_{k=1}^{K} x_k = 1,$$

     Character representation [10].

   - Every text, or string of characters, is therefore represented as a 2D matrix, where each character estimate is stacked on top of one another.

     $$X = \left(x_{jk}\right) \in \left[0.0, 1.0\right]^{M \times K}, \quad \forall j : \sum_{k=1}^{K} x_{jk} = 1,$$

     Word representation [10].

   (b) The algorithm then creates weights for each representation, depending on the focus/blurriness of the image that generated the output, with more weight given to more in-focus frames.

   - The weights of each frame are determined based on the focus estimation of an image, which is based on the MINIMUM of the UPPER 95% QUARTILE of the GRADIENTS of a string. The weight is calculated using the minimum of the upper 95% quartile of the gradients.

     $$G^{V}_{\mathrm{r,c}}(I_i(\bar{X})) = \left|\mathrm{I}_{r+1,\mathrm{c}} - \mathrm{I}_{\mathrm{r,c}}\right|,$$
     $$G^{H}_{\mathrm{r,c}}(I_i(\bar{X})) = \left|\mathrm{I}_{\mathrm{r,c+1}} - \mathrm{I}_{\mathrm{r,c}}\right|,$$
     $$G^{D_1}_{\mathrm{r,c}}(I_i(\bar{X})) = (1/\sqrt{2})\left|\mathrm{I}_{r+1,\mathrm{c+1}} - \mathrm{I}_{\mathrm{r,c}}\right|,$$
     $$G^{D_2}_{\mathrm{r,c}}(I_i(\bar{X})) = (1/\sqrt{2})\left|\mathrm{I}_{\mathrm{r,c+1}} - \mathrm{I}_{r+1,\mathrm{c}}\right|,$$

     Image Gradient (G) Calculation in the x,y,xy,yx direction, using image intensity (I) [10].

     $$F(I_i(\bar{X})) = \min\left\{q(G^{V}(I_i(\bar{X}))), q(G^{H}(I_i(\bar{X}))),\right.$$
     $$\left. q(G^{D_1}(I_i(\bar{X}))), q(G^{D_2}(I_i(\bar{X})))\right\},$$

     Focus equation using the minimum of the upper 95% quartile (q) of the gradients [10].

(c) The algorithm then, using the weights and the string outputs, generates a running estimation of the data. This estimation also considers if some frames have read too many or too few characters.

- We begin with an empty column vector or the first recognized result. When two recognized texts are obtained, due to variable length recognition, the texts are first aligned using the distance equation below. In essence, the algorithm aligns the text to maximize agreeability.

  $$\rho\left(x^1, x^2\right) = \frac{1}{2}\sum_{k=0}^{K}\left|x_k^1 - x_k^2\right|$$

  Alignment distance equation [10].

- Then, the algorithm combines the two texts using the weighted equation below to produce a resulting estimation of the desired text.

  $$r = \left(r_k\right) \in \left[0.0, 1.0\right]^{K+1}, \quad \forall k : r_k = \frac{x_k^1 \cdot w(x^1) + x_k^2 \cdot w(x^2)}{w(x^1) + w(x^2)}$$

  Weighted combination equation [10].

In the case of the ID, once the object is no longer detected for some time, the running values in step 3 will be reported. In the case of the vitals monitors, since vitals information will change, frames that were taken in the past will gradually lose weight, and the running estimation of the vitals will continuously be reported.

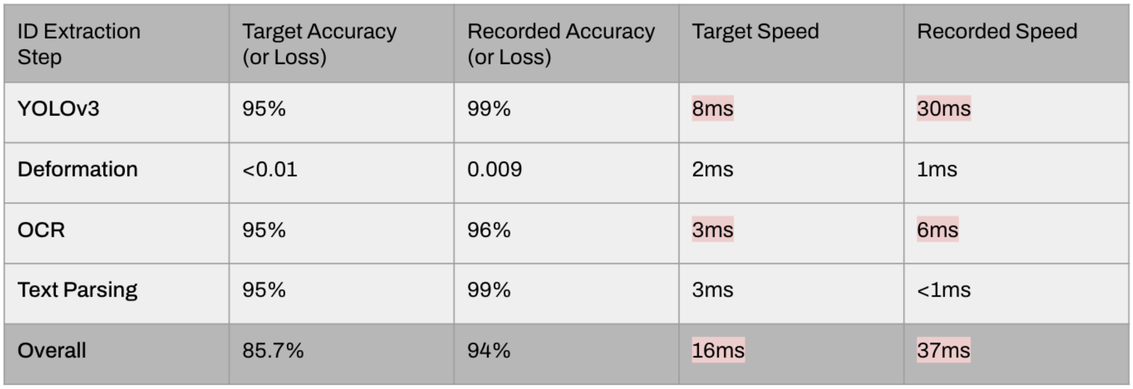
4 Tests and Results

1. **Components Tests**

   - **YOLO Detection:**
     Due to their representativeness, more data augmented images were created as described in setup, and these, as well as the negative images, were fed into the YOLO algorithm. The accuracy was reported as (the number of images correctly detected)/(the number of images).
   - **Deformation Algorithms:**
     Using the manual data augmentation demonstrated in the Driver License YOLOv3 Dataset Documentation, we created custom transforms (and recorded them as the truth), applied them to flat images of IDs and pasted them onto background images. These labeled images were then fed into the YOLO and then the deformation algorithm to calculate a transform. The Loss was then calculated as the sum of the squared difference of all entries between the true transform and predicted transform: Loss = n(true-predicted)2.
   - **OCR Algorithms:**
     Images from the smart glasses camera were first fed through the YOLO and deformation algorithms. 200 ‘Regions of interest’ (ROI) from the resulting images were then manually labeled with their truth. These same post-YOLO deform images were then fed into the OCR algorithm, and the output was compared with truth. The

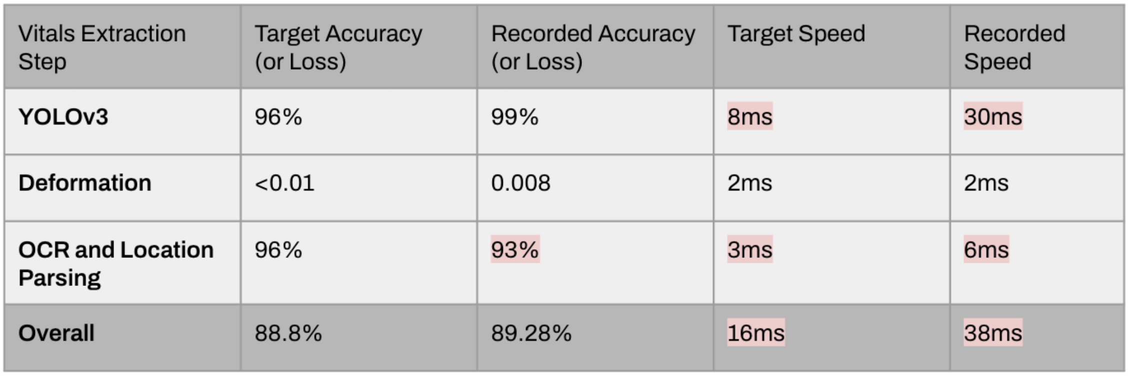
accuracy is determined as the number of ROIs correctly detected and read, over the total number of ROIs.

- **Text Parsing Algorithm:**
  Many 1000 sized list of dates, names, states, sex, series of numbers, heights, and weights were automatically generated and fed into the algorithm to categorize the items. The accuracy was determined as the number of correctly categorized items over the total number of items.



ID Extraction Step	Target Accuracy (or Loss)	Recorded Accuracy (or Loss)	Target Speed	Recorded Speed
YOLOv3	95%	99%	8ms	30ms
Deformation	<0.01	0.009	2ms	1ms
OCR	95%	96%	3ms	6ms
Text Parsing	95%	99%	3ms	<1ms
Overall	85.7%	94%	16ms	37ms

Table 1: ID Extraction Algorithm: Resulting Accuracies and Speeds of Individual Component Algorithms.



Vitals Extraction Step	Target Accuracy (or Loss)	Recorded Accuracy (or Loss)	Target Speed	Recorded Speed
YOLOv3	96%	99%	8ms	30ms
Deformation	<0.01	0.008	2ms	2ms
OCR and Location Parsing	96%	93%	3ms	6ms
Overall	88.8%	89.28%	16ms	38ms

Table 2: Vitals Extraction Algorithm: Resulting Accuracies and Speeds of Individual Component Algorithms.

2. **Frame Averaging Test** Two videos featuring the IDs and vitals were taken with the smart glasses and recorded using zoom in order to assess the performance of the overall algorithm, including frame averaging.

   (a) The first video begins in an empty environment. Then, sequentially, driver licenses made of paper are brought into frame for 3-5 seconds and then taken out of frame for 3-5 seconds. Reports of the driver licenses are output in a text file. Accuracy is determined by correct results/all results.

   - Accuracy = 95

   (b) The second video begins by observing a vitals monitor. Throughout the video, the camera moves around this monitor, during which the vitals information changes. Accuracy is determined based on correctly reporting vitals information across time.

   - Accuracy = 99

5 Discussion

1. **Accuracy**

   Among all component algorithms and the whole workflow, we observe high accuracies. This is good considering that, with respect to the tasks of documentation and vitals reporting, mistakes can be costly to correct. The only component that performed under the desired accuracy is the vitals extraction's OCR algorithm. However, this may be due to the presence of significant noise in a large chunk of the tested dataset. Removing simply the noisy parts produces accuracies of 97%, but this was not done as to be representative. Further testing with other datasets may elucidate the OCR's performance. Still, we may not need to worry about the vitals extraction's OCR algorithm; the frame averaging algorithm significantly improves the performance of the individual algorithms, as can be seen from improving drivers license extraction accuracy from 94% -¿ 95% and, even higher, improving vitals monitor extraction accuracy from 89% -¿ 99%.

2. **Algorithmic Speeds with respect to Camera**

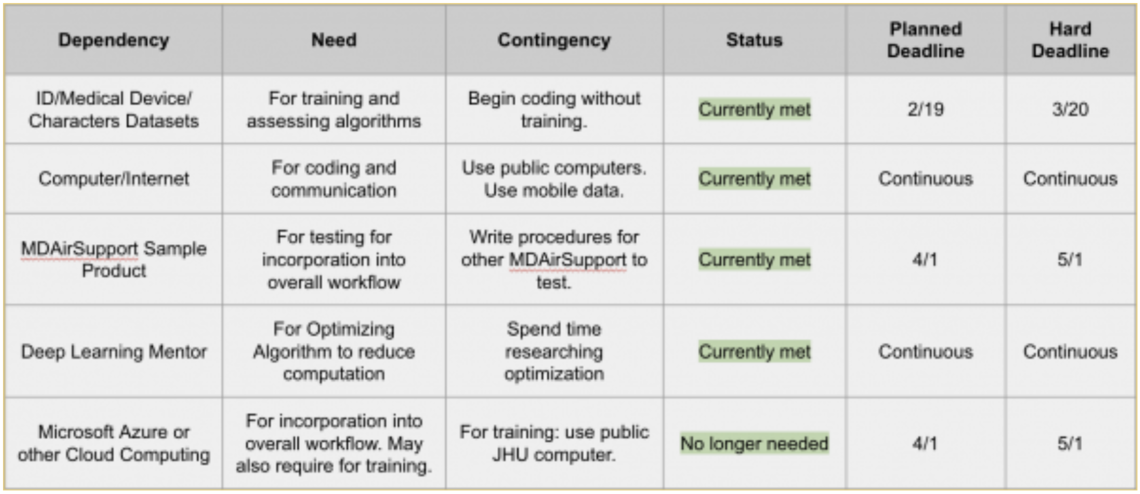
   The camera noted in the overview workflow runs on 60 frames per second (FPS). Therefore, the reported algorithmic speeds of 37ms and 38ms per frame is too slow, only allowing 27FPS. As most of the interval time comes from the YOLOv3 algorithm, which is difficult to alter and test without a lot of time and computing power, this speed may need to be conceded in order to maintain high accuracies. Still, future endeavors could look into alternative detection algorithms in order to maintain accuracies while improving speeds.

6 Progress Evaluation

In general, the project went according to schedule and without hindrance. The maximum deliverable of creating an integrated app that would take live feed from the smart glasses was removed, as the integration with live feed was deemed currently too difficult by the company mentors. This resulted in a change in the scope of the project: rather than building an integrated app to take in live feed, the final algorithm takes in a Zoom recording.

1. **Dependencies**

   Almost all dependencies were acquired on schedule, with the only exception of acquiring cloud computing resources being canceled due to the change in the scope of the project.

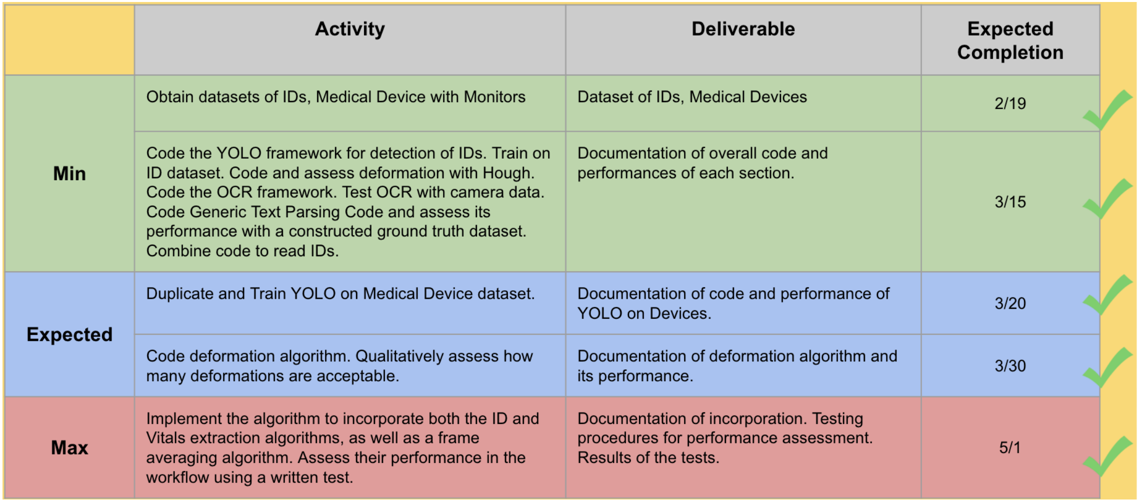


Dependency	Need	Contingency	Status	Planned Deadline	Hard Deadline
ID/Medical Device/ Characters Datasets	For training and assessing algorithms	Begin coding without training.	Currently met	2/19	3/20
Computer/Internet	For coding and communication	Use public computers. Use mobile data.	Currently met	Continuous	Continuous
MDAirSupport Sample Product	For testing for incorporation into overall workflow	Write procedures for other MDAirSupport to test.	Currently met	4/1	5/1
Deep Learning Mentor	For Optimizing Algorithm to reduce computation	Spend time researching optimization	Currently met	Continuous	Continuous
Microsoft Azure or other Cloud Computing	For incorporation into overall workflow. May also require for training.	For training: use public JHU computer.	No longer needed	4/1	5/1

Table 3: Table of Dependencies

2. **Deliverables**

All minimum activities/deliverables, all expected activities/deliverables, and a small bit of the maximum activity/deliverable were completed. Our deliverables are shown below.



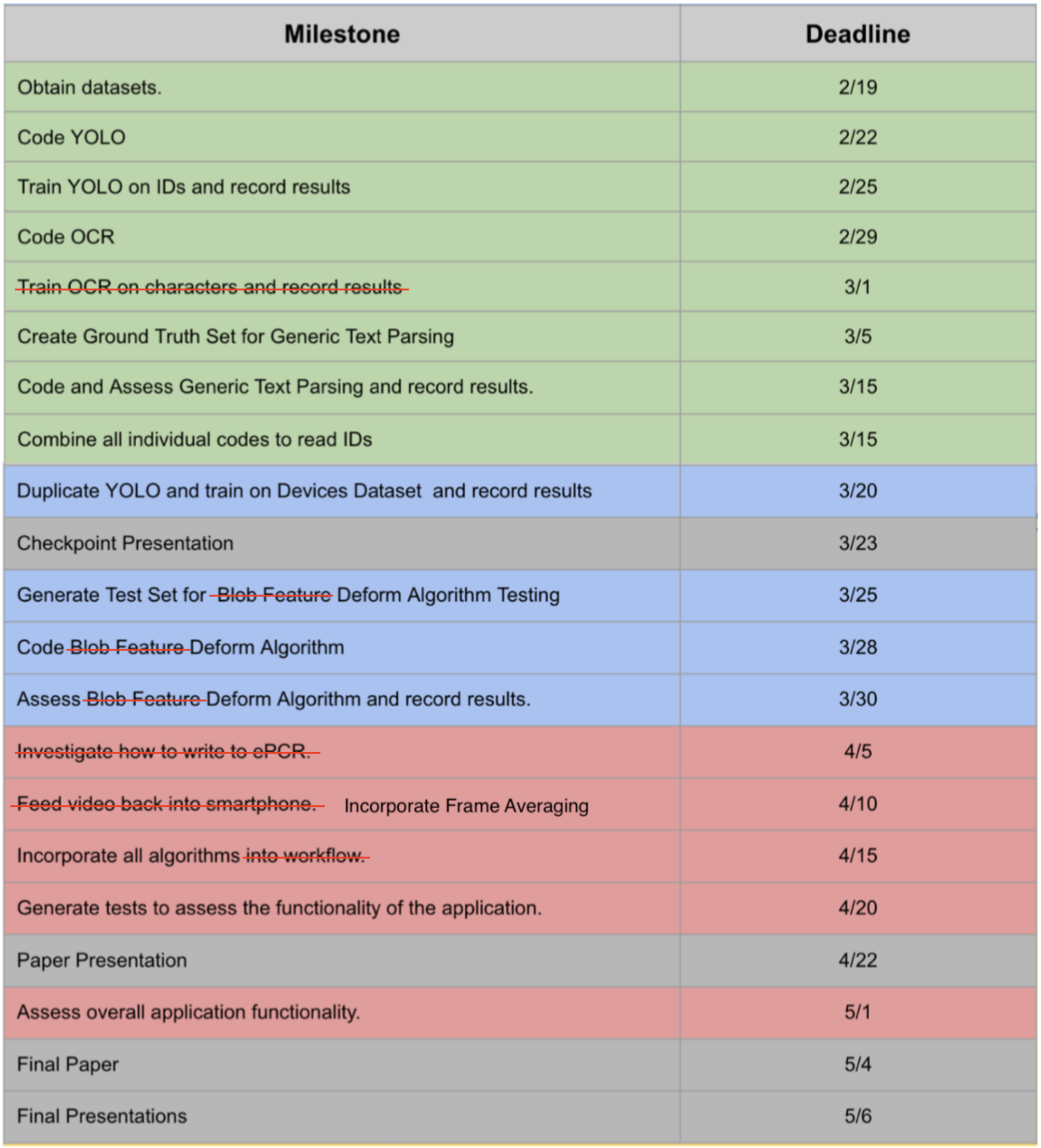
	Activity	Deliverable	Expected Completion	
Min	Obtain datasets of IDs, Medical Device with Monitors	Dataset of IDs, Medical Devices	2/19	✓
	Code the YOLO framework for detection of IDs. Train on ID dataset. Code and assess deformation with Hough. Code the OCR framework. Test OCR with camera data. Code Generic Text Parsing Code and assess its performance with a constructed ground truth dataset. Combine code to read IDs.	Documentation of overall code and performances of each section.	3/15	✓
Expected	Duplicate and Train YOLO on Medical Device dataset.	Documentation of code and performance of YOLO on Devices.	3/20	✓
	Code deformation algorithm. Qualitatively assess how many deformations are acceptable.	Documentation of deformation algorithm and its performance.	3/30	✓
Max	Implement the algorithm to incorporate both the ID and Vitals extraction algorithms, as well as a frame averaging algorithm. Assess their performance in the workflow using a written test.	Documentation of incorporation. Testing procedures for performance assessment. Results of the tests.	5/1	✓

Table 4: Table of Deliverables

All completed deliverables were conducted on time. However, throughout the semester, there were a few adjustments to the deliverables. The first change was the removal of requiring a character dataset, which would have been used to train an OCR. As we decided to not train an OCR ourselves, we no longer needed it. The second change was the removal of the use of a Blob Feature Detection algorithm for the deformation of the vitals monitor. A simple Hough Edge transform was found to be good enough. The last change was, rather than build an integrated app to take in live feed, we would build an algorithm to take in a Zoom .mp4 recording.

3. **Milestones**

   The milestone table shown below was generally followed, though there was a small setback when I had switched classroom presentation times with a classmate, which pushed me back one week in March. Furthermore, with the changes described in the Deliverables section, some milestones were removed, and one milestone was added, which was met on time.



Milestone	Deadline
Obtain datasets.	2/19
Code YOLO	2/22
Train YOLO on IDs and record results	2/25
Code OCR	2/29
~~Train OCR on characters and record results~~	3/1
Create Ground Truth Set for Generic Text Parsing	3/5
Code and Assess Generic Text Parsing and record results.	3/15
Combine all individual codes to read IDs	3/15
Duplicate YOLO and train on Devices Dataset and record results	3/20
Checkpoint Presentation	3/23
Generate Test Set for ~~Blob Feature~~ Deform Algorithm Testing	3/25
Code ~~Blob Feature~~ Deform Algorithm	3/28
Assess ~~Blob Feature~~ Deform Algorithm and record results.	3/30
~~Investigate how to write to ePCR.~~	4/5
~~Feed video back into smartphone.~~ Incorporate Frame Averaging	4/10
Incorporate all algorithms ~~into workflow.~~	4/15
Generate tests to assess the functionality of the application.	4/20
Paper Presentation	4/22
Assess overall application functionality.	5/1
Final Paper	5/4
Final Presentations	5/6

Table 5: Table of Milestones

7 Conclusion

1. **Significance**

   Overall this semester, we successfully programmed an algorithm that took in recordings from smart glasses and that utilized object detection and optical character recognition in order to extract information from driver licenses and the Zoll and LIFEPAK vital monitors. A frame averaging algorithm was also programmed in order to increase accuracy of the extracted data. The results of this work are significant on two fronts. Firstly, this algorithm can be used, or further developed, in order to reduce documentation times and increase physician confidence and treatment speeds. Secondly, this successful foray of AI into the smart glasses emergency medical field indicates significant potential of this type of improvement.

2. **Moving Forward**

   In the future, it will be necessary to implement different types of IDs into the algorithm, as well as incorporate more vitals monitors, such as handheld ultrasounds. Furthermore, it will be necessary to implement the algorithm to take in live feed rather than recordings. After doing so, it may be possible to observe how well the algorithm performs in an actual emergency setting.

3. **Final Words**

   Managing a project is truly a difficult experience, though I am grateful to have been able to develop these skills. It was also a good experience to have been able to apply computer vision principles creatively to solve a problem that seemed to have a tangible impact. I also learned about documentation practices, especially for software development, and I hope to develop these skills in order to pursue either this project or any medical device projects I may encounter in the future.

8 References

1. Vuzix Corporation. Vuzix smart glasses at the chi mei medical center, taiwan. Technical report, 2020.

2. Laeben Lester. Inquiry into ems documentation times, 2021.

3. GreenLight Medical. Standardizing medical devices: Value analysis, 2020.

4. Henning Muller Antoine Widmer Roger Schaer, Thomaz Melly. Using smart glasses in medical emergency situations, a qualitative pilot study. *2016 IEEE Wireless Health (WH)*, 2016.

5. Radosveta Wells Stormy Monks Scott Crawford, Igor Kushner. Electronic health record documentation times among emergency medicine trainees. *Perspect Health Inf Manag*, 16, 2019.

6. Shaw, Gabbi, and Frank Olito. 2020. “What a Driver’s License Looks like in Every State.” *Insider*. Insider. January 21. https://www.insider.com/what-drivers-license-looks-like-in-every-state.

7. Gould, Stephen, Richard Fulton, and Daphne Koller. 2009. “Decomposing a Scene into Geometric and Semantically Consistent Regions.” *2009 IEEE 12th International Conference on Computer Vision.* doi:10.1109/iccv.2009.5459211.

8. V. V. Arlazarov, K. Bulatov, T. Chernov and V. L. Arlazarov, “A dataset for identity documents analysis and recognition on mobile devices in video stream”, *Comput. Opt.*, vol. 43, no. 5, pp. 818-824, 2019.

9. Kuznetsova, Alina, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, et al. 2020. “The Open Images Dataset V4.” *International Journal of Computer Vision* 128 (7): 1956–81. doi:10.1007/s11263-020-01316-z.

10. O. Petrova, K. Bulatov, V. Arlazarov, V. Arlazarov, Weighted combination of per-frame recognition results for text recognition in a video stream, *Computer Optics.* 45 (2021) 77–89. doi:10.18287/2412-6179-co-795.