Overseas Immersion Programme

Year 2021, Trimester 3

Logbook

Team 24

Student Name	Student ID
GAY HUI JIE	2508471G

Week 1 3

Design and needfinding 3

16/08/2021 3
17/08/2021 3
18/08/2021 5

Prototyping and Production Phase 8

19/08/2021 8
Prediction on the drying of syringe 8
20/08/2021 11

Week 2 14

23/08/2021 14
24/08/2021 16
25/08/2021 18
26/08/2021 20
27/08/2021 23

Week 3 24

30/08/2021 24
31/08/2021 30
01/09/2021 31
02/09/2021 31
03/09/2021 32

Github : https://github.com/BburnN123/CSC3003_OIP_Object_Detection
Dataset :
https://drive.google.com/drive/folders/1CMoZhX02zTXUHD3-xnd7FrqmlHZoAvic?usp=sharing

Week 1

Design and needfinding

16/08/2021

Manual cleaning and drying is still occurring in the hospital which might be very time consuming and ineffective. Hence, researching and understanding existing products should be the first steps in creating the product.

Understanding the existing product	https://www.gbukenteral.com/pdf/Dishwasher-instructions-syringes.pdf https://medicina.co.uk/wp-content/uploads/2018/06/LHE-Syringe-Cleaning-Instructions.pdf https://www.thewellproject.org/hiv-information/cleaning-equipment-injecting-drugs#:~:text=Fill%20the%20syringe%20with%20water,(do%20not%20reuse%20water). https://medlabgear.com/blogs/articles/how-to-sterilize-a-syringe
Determine factor that might affect the syringe	https://europepmc.org/article/med/8207661 https://journals.sagepub.com/doi/abs/10.2190/2HMC-W575-5M2E-G3LU?casa_token=NRZ-0tu2oKsAAAAA%3A-4246BUXx6gPTMjlhzGFmbDL1Y34TEVaYJC5N26tw-iMW-whDp9Y8cAzs8wczvXyfB3j_y1ZA5R-&

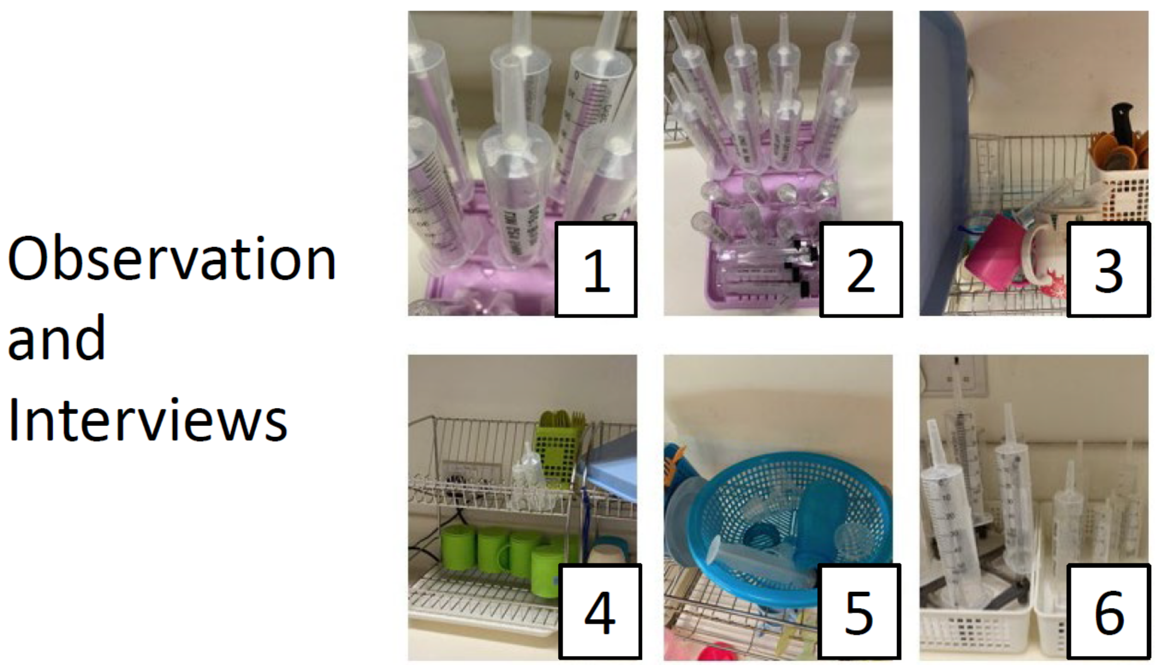
17/08/2021

A short analysis can be seen from here
"The team spoke to the nurses onsite and also carried out observation of the problem from end to end. They saw for themselves how administration sets left to dry were cluttered together with

other utensils (picture 3 and %). One of the wards was on trial using an easy drying and standing kit (picture 1, 2 and 6), however, it did not effectively dry the syringes over time as there was still water vapour present within the syringes. "

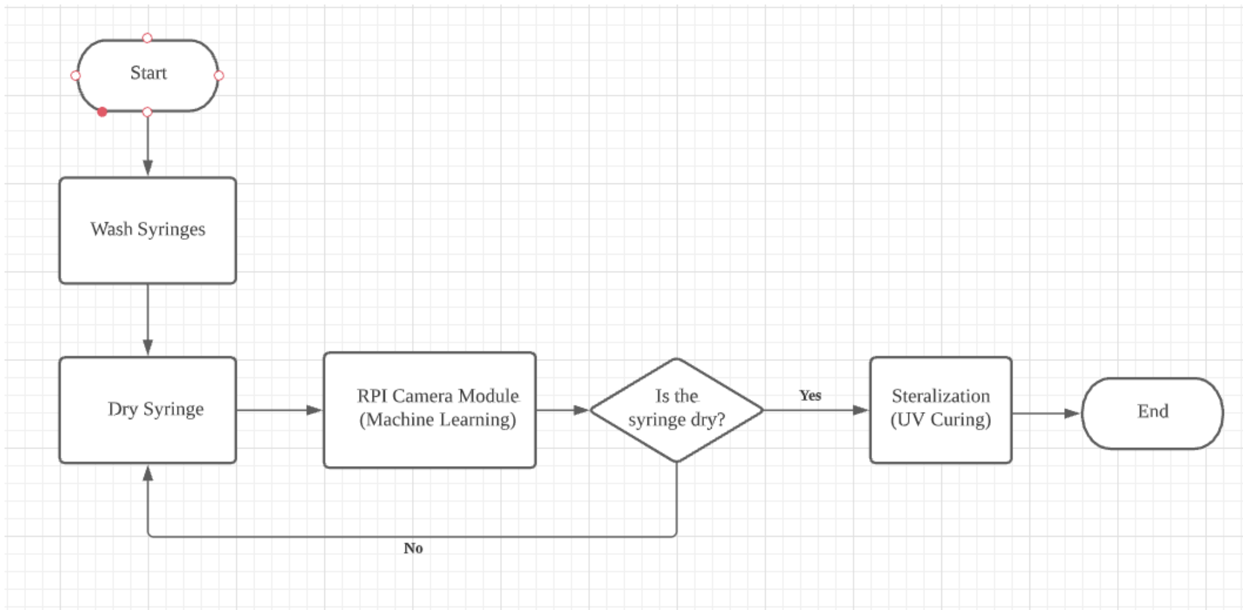
" However, due to the lack of space, nurses undergo a time-consuming process of air drying NGT equipment to prevent microorganisms colonisation."

"The process of drying the NGT administration set was found to be ineffective and time-consuming, with 2 hours and 7 minutes for each set. The disposable feeding syringes are washed 5-6 times a day and reused for 1 week; consequently, increasing the potential risk of infection."



With this analysis the team have decided to tackle the problem of the drying process. We decided to incorporate machine learning to predict the timing and

A simple flowchart was created to show the process of our products. As can be seen in the flow chart, the normal operation of washing, drying and sterilization still operate. The only difference was that instead of a fixed timing for the drying of the syringe, the product should be able to be determined by checking if the syringe is dry enough to be complete.

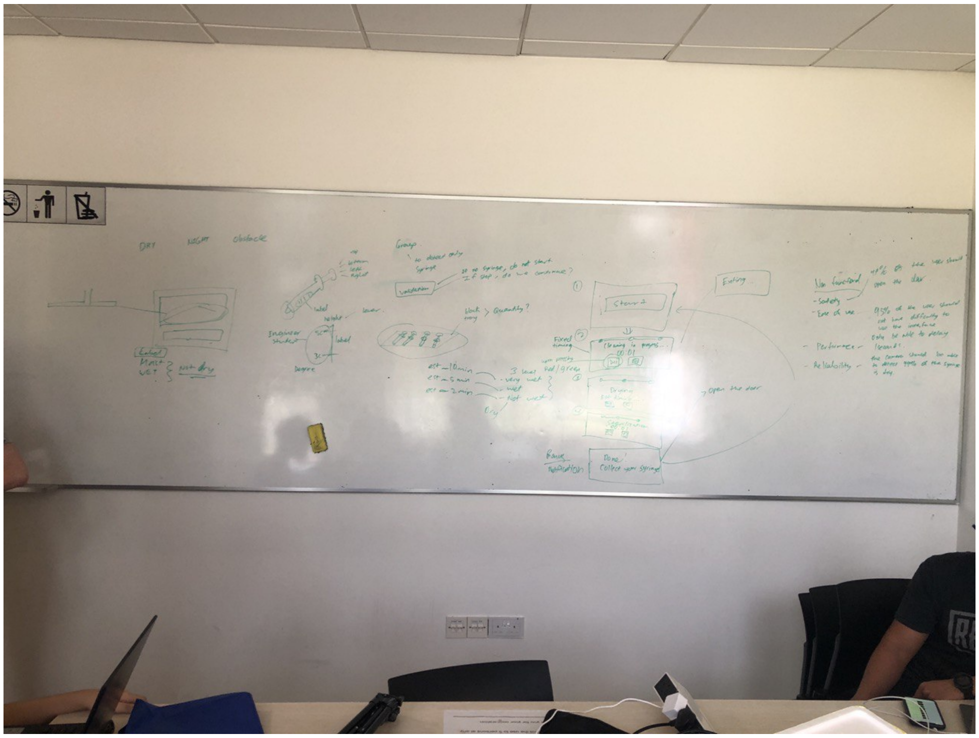


A slide was done to prepare for the consultation the next day. [1]

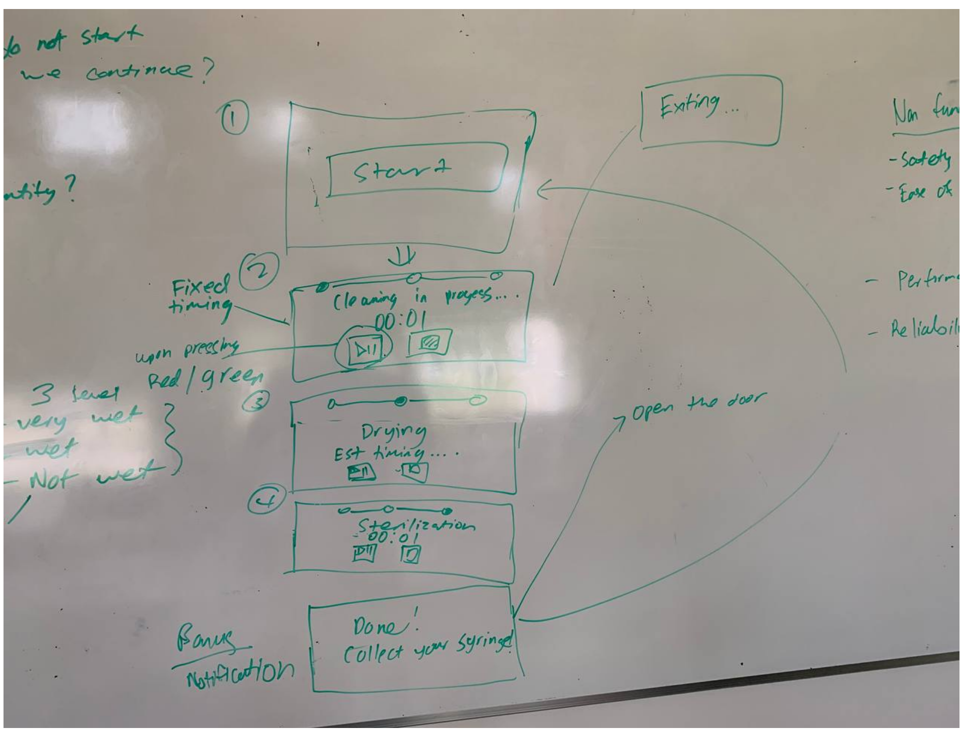
[1] https://docs.google.com/presentation/d/1QZGe919jOkf1IP7H8mteNaKE167-023v4mg31rKl4AI/edit?usp=sharing

18/08/2021

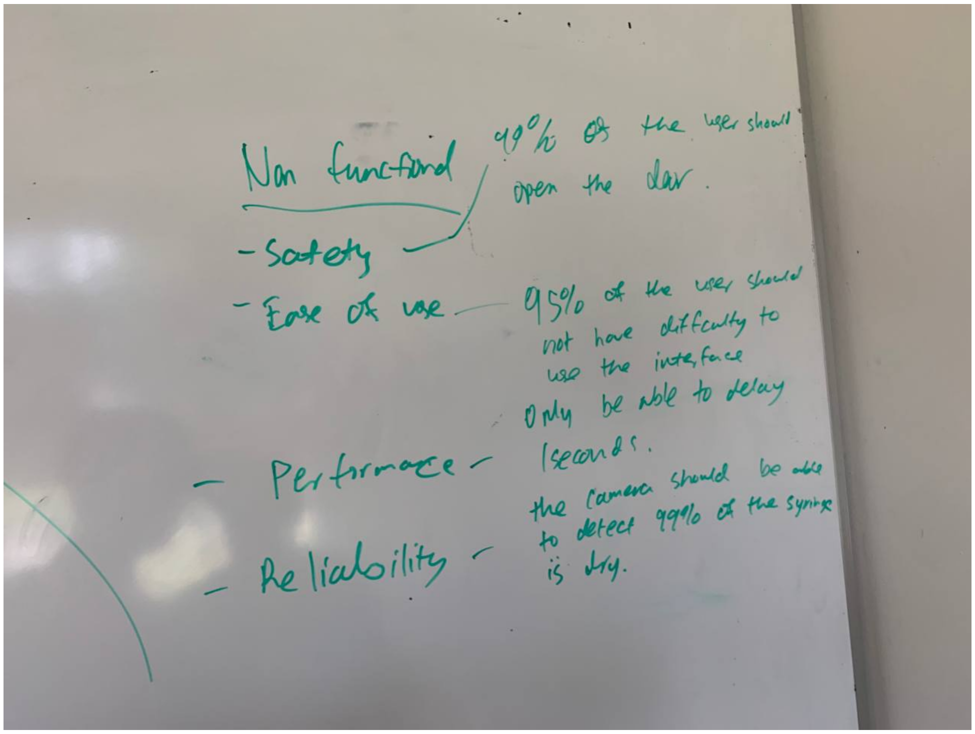
Collection of the different classification of the syringe for machine learning [1]. Connected the Raspberry Pi with the camera module using tensorflow lite and open cv. Also the LCD was tested on the screen and was working properly. A discussion was done with the Mechatronic design and Mechatronic engineer student on how we are able to further improve our products.



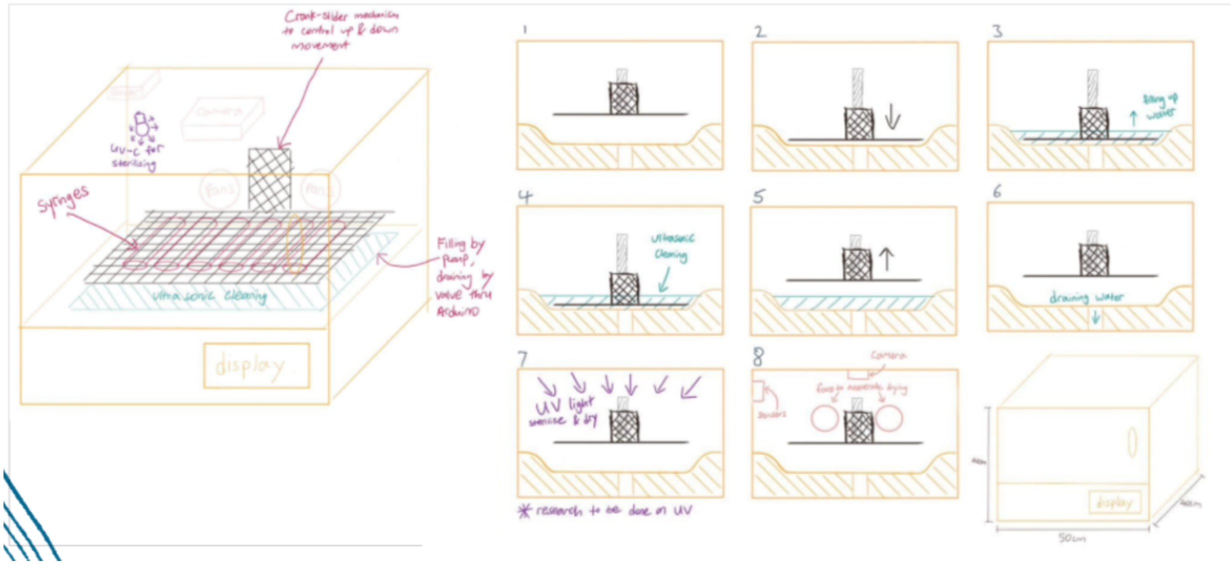
Overview of the discussion



Wireframe



Non functional requirement



Design Draft

[2] https://drive.google.com/drive/folders/1DkBYXOTbnS_2dD8hHN_SqfFhHEwB8JWq?usp=sharing

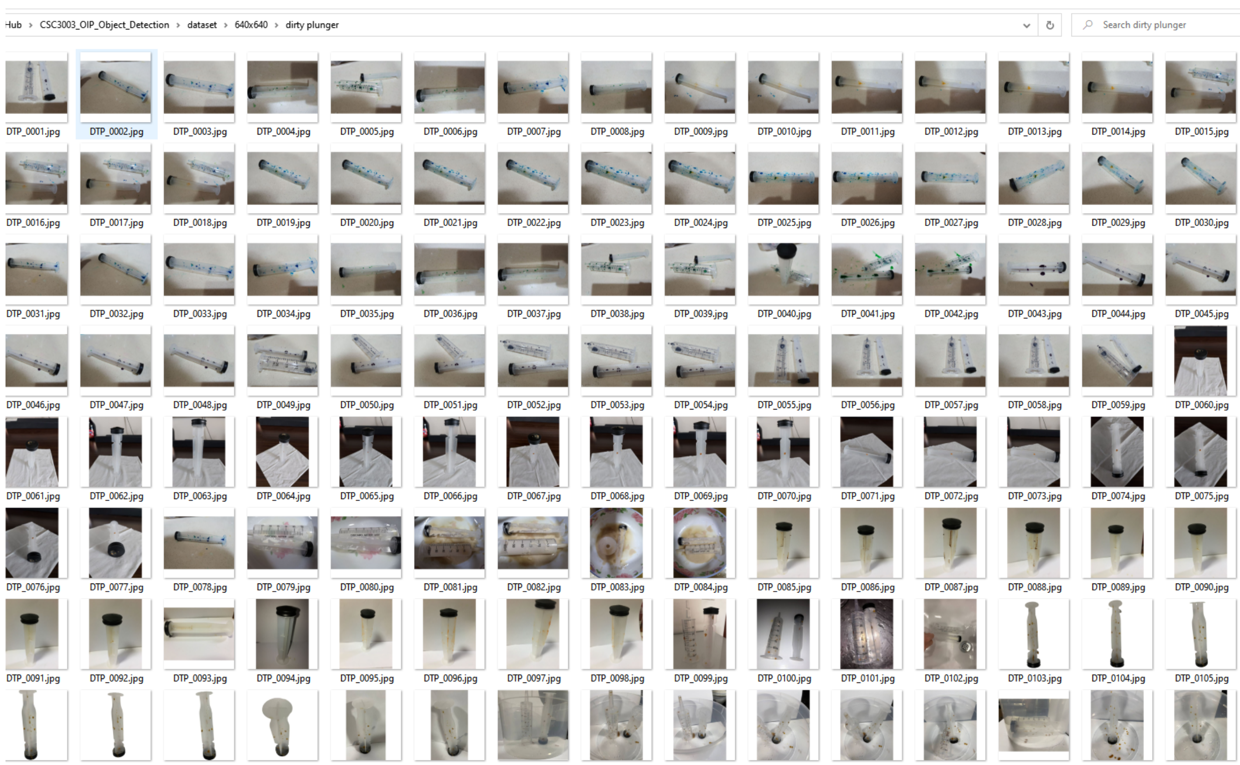
Prototyping and Production Phase

19/08/2021

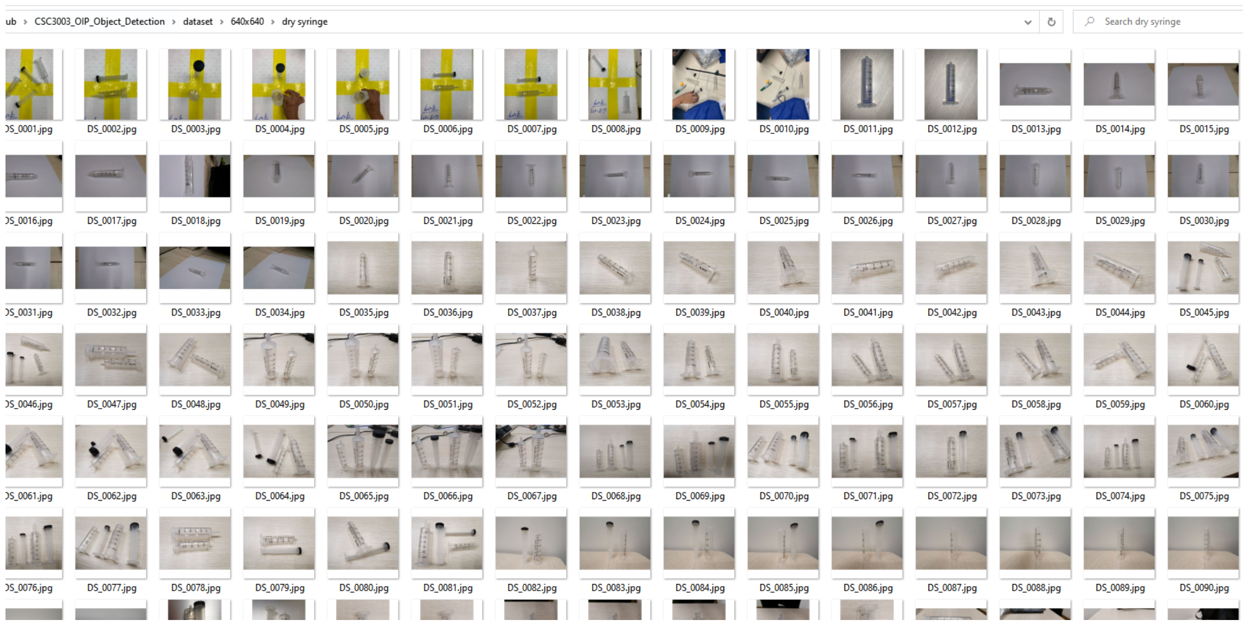
Prediction on the drying of syringe

I have researched on different ML software and decided to try tensorflow lite as it is lightweight which can support the Raspberry Pi 3b+. Before programming and training of the model, the team took different images based on the design that we are trying to implement. There are a few more additional images that the team tried to take. A total of 1000+ images were taken during that day.

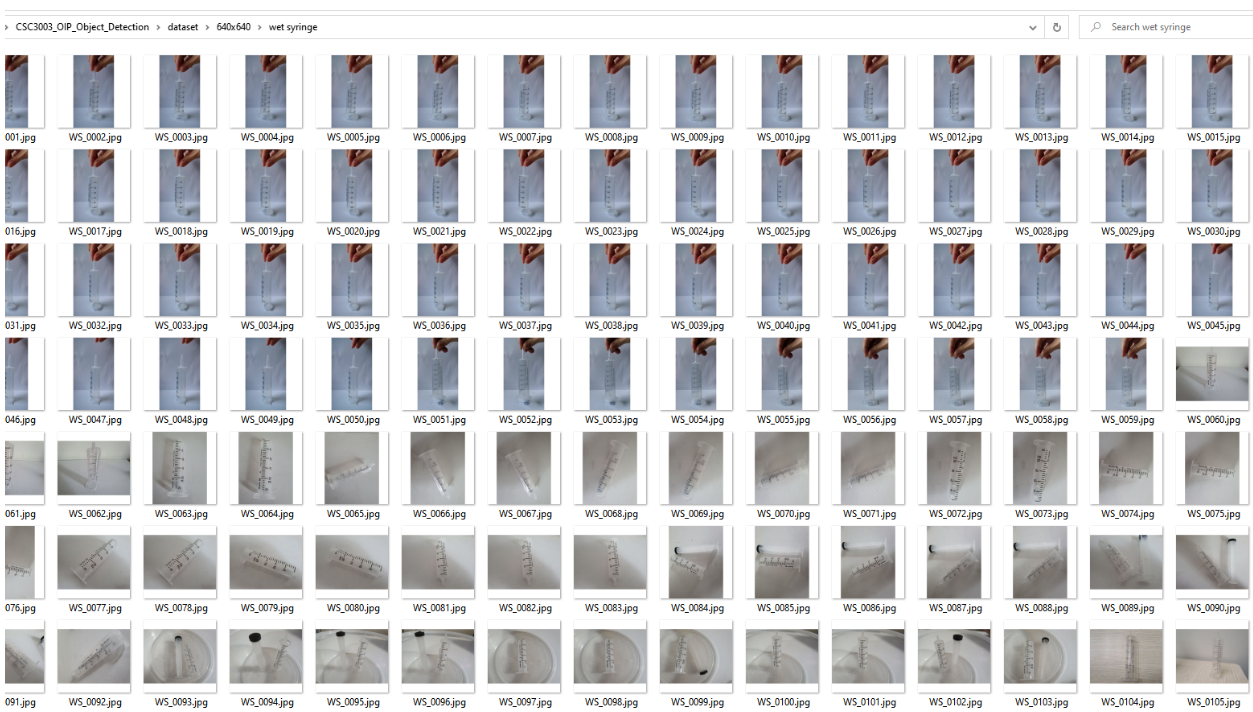
Dirty Syringe



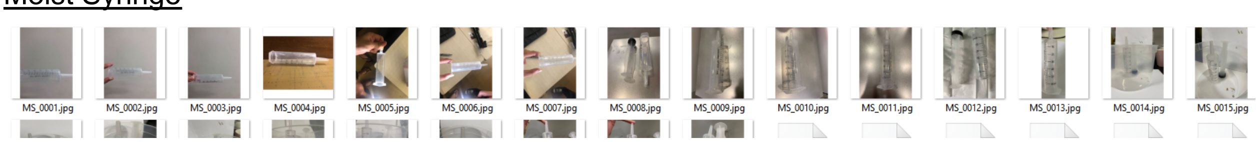
Dry Syringe



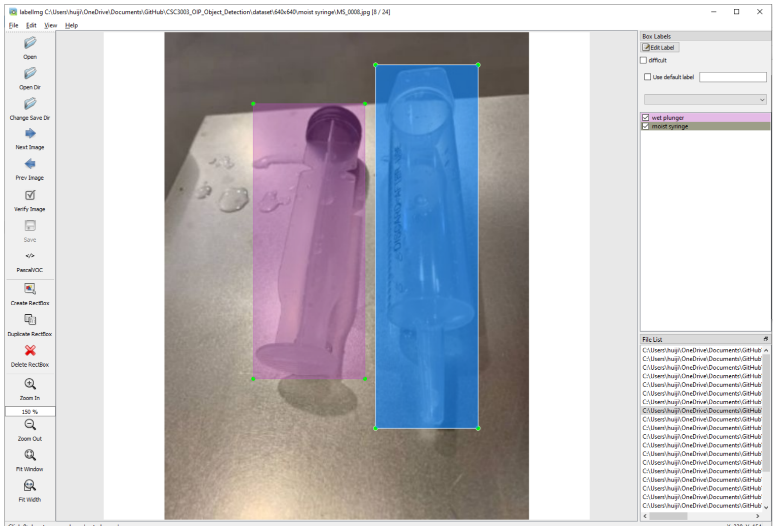
Wet Syringe



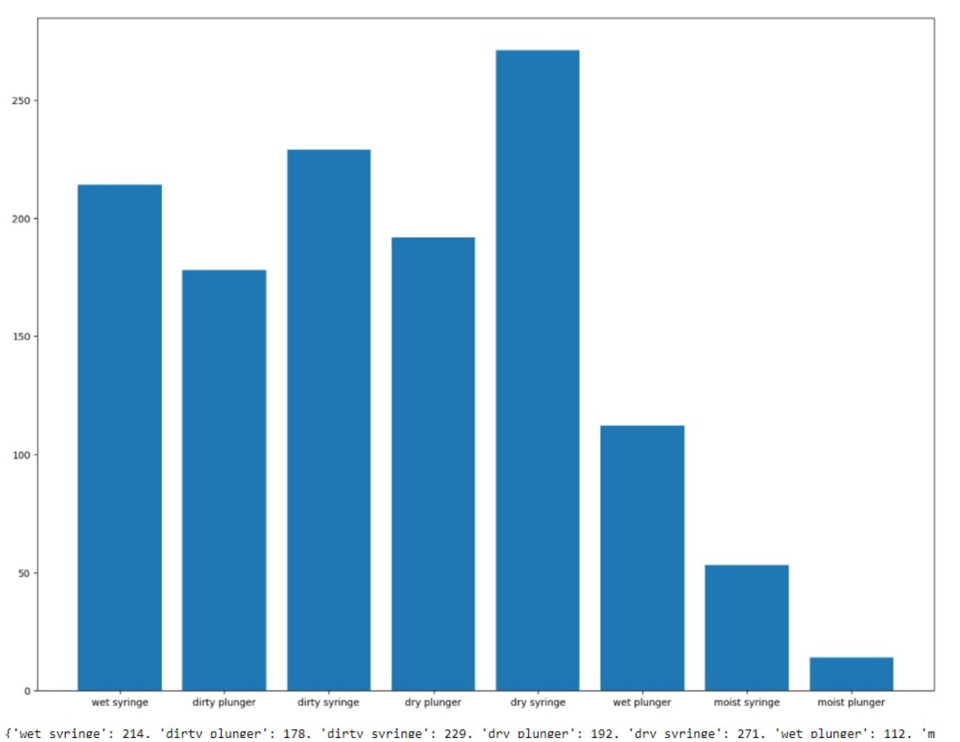
Moist Syringe



After capturing the images, we used labelImg [1] to label and annotate the all images. The process took an estimation time of more than 4 hour to identify and label all the images



Annotation of the image

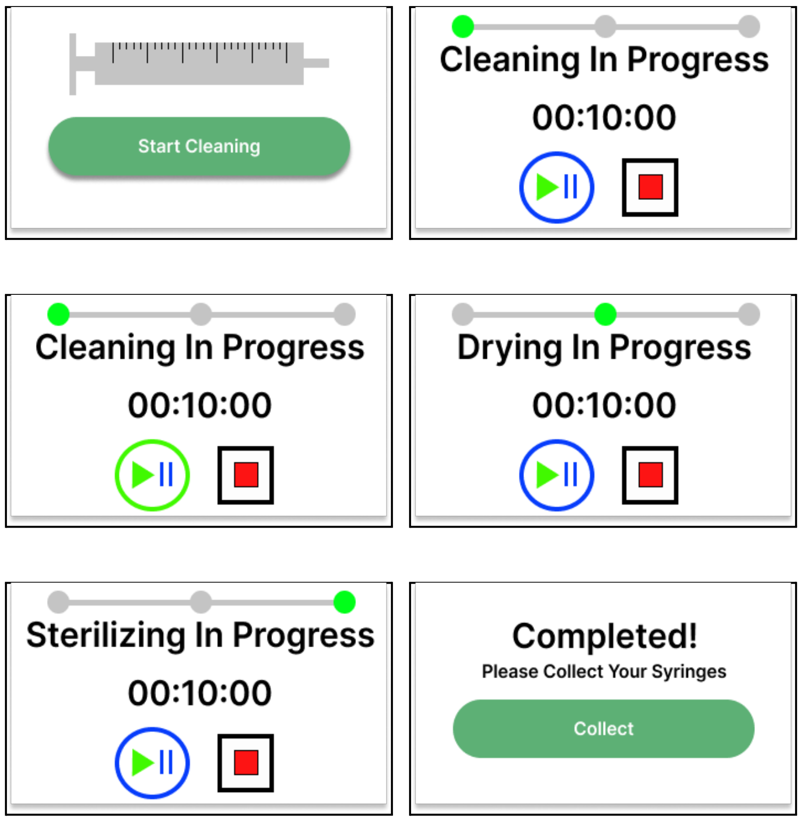


An analysis of the images that was captured

[1] https://github.com/tzutalin/labelImg

20/08/2021

The Figma prototype was created as we needed a user interface to control the hardware. This was set as priority for the computer science student to create an interface for the user to use the product. The target was to keep it simple.



After discussing the User Interface on figma, we proceed to the study of implementing tensorflow lite model with the help of

[1] https://github.com/EdjeElectronics/TensorFlow-Object-Detection-API-Tutorial-Train-Multiple-Objects-Windows-10

[2] https://github.com/armaanpriyadarshan/TensorFlow-2-Lite-Object-Detection-on-the-Raspberry-Pi

[3] https://github.com/tensorflow/models

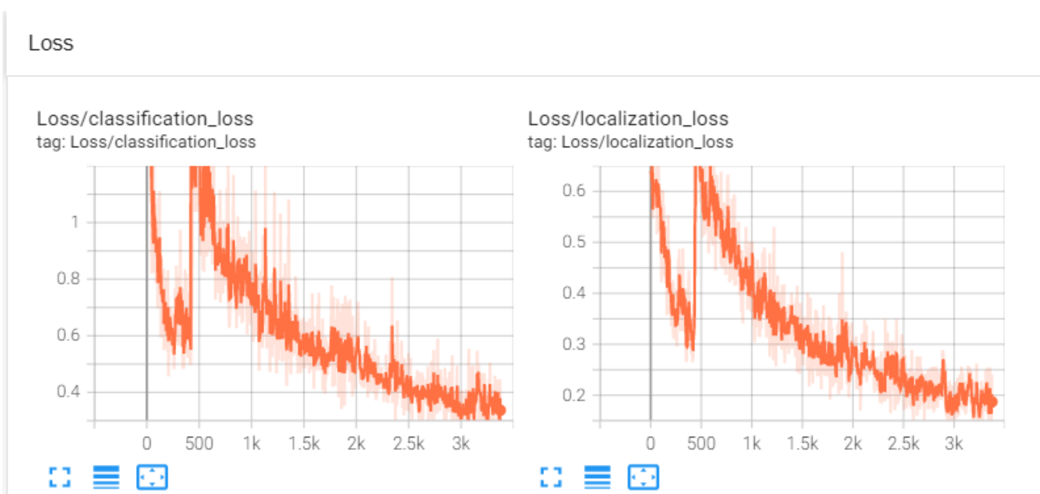
The team decided to use the SSD MobileNet V2 FPNLite 640x640 model as it provided a better accuracy than those models that support the images of 300x300. The Raspberry Pi can only support the SSD model.

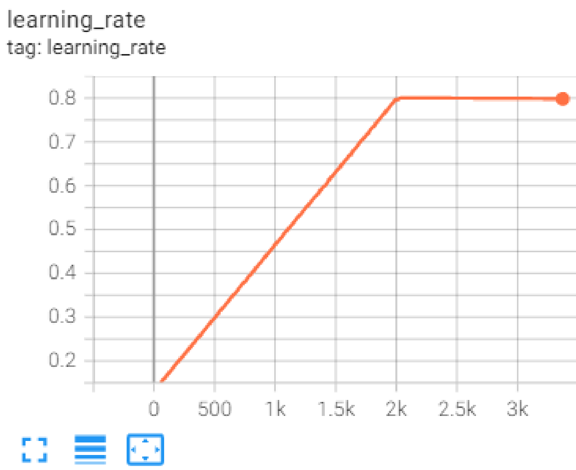
SSD MobileNet v2 320x320	19	20.2	Boxes
SSD MobileNet V1 FPN 640x640	48	29.1	Boxes
SSD MobileNet V2 FPNLite 320x320	22	22.2	Boxes
SSD MobileNet V2 FPNLite 640x640	39	28.2	Boxes
SSD ResNet50 V1 FPN 640x640 (RetinaNet50)	46	34.3	Boxes
SSD ResNet50 V1 FPN 1024x1024 (RetinaNet50)	87	38.3	Boxes
SSD ResNet101 V1 FPN 640x640 (RetinaNet101)	57	35.6	Boxes
SSD ResNet101 V1 FPN 1024x1024 (RetinaNet101)	104	39.5	Boxes
SSD ResNet152 V1 FPN 640x640 (RetinaNet152)	80	35.4	Boxes
SSD ResNet152 V1 FPN 1024x1024 (RetinaNet152)	111	39.6	Boxes

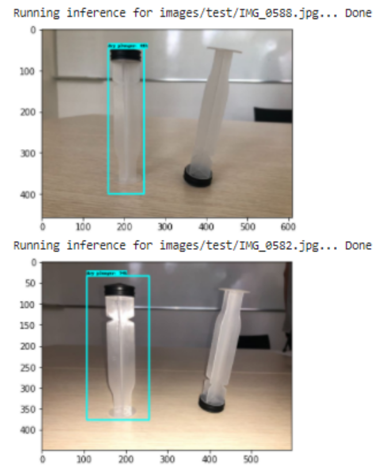
Comparison of the model

[4] https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md

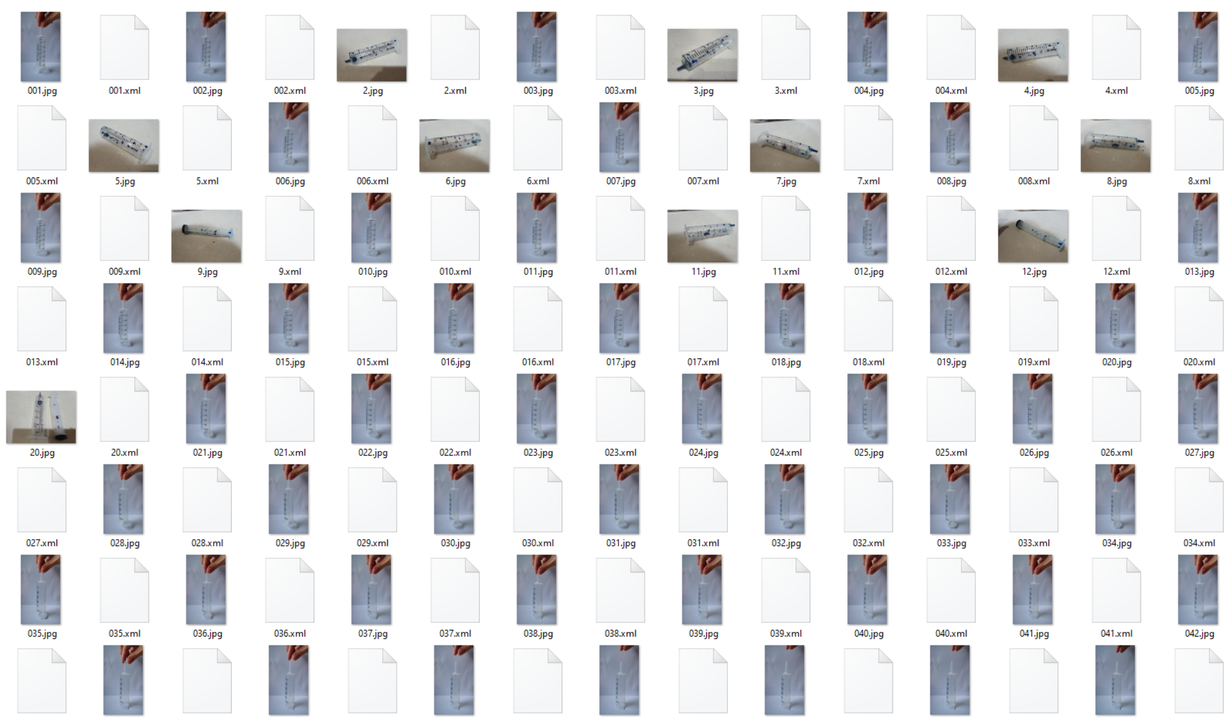
The team has managed to do it with the dataset we have prepared. However the detection of the wet syringes still requires a lot of improvement. A few more trained models was tested to get the precision of the data on Windows 10.



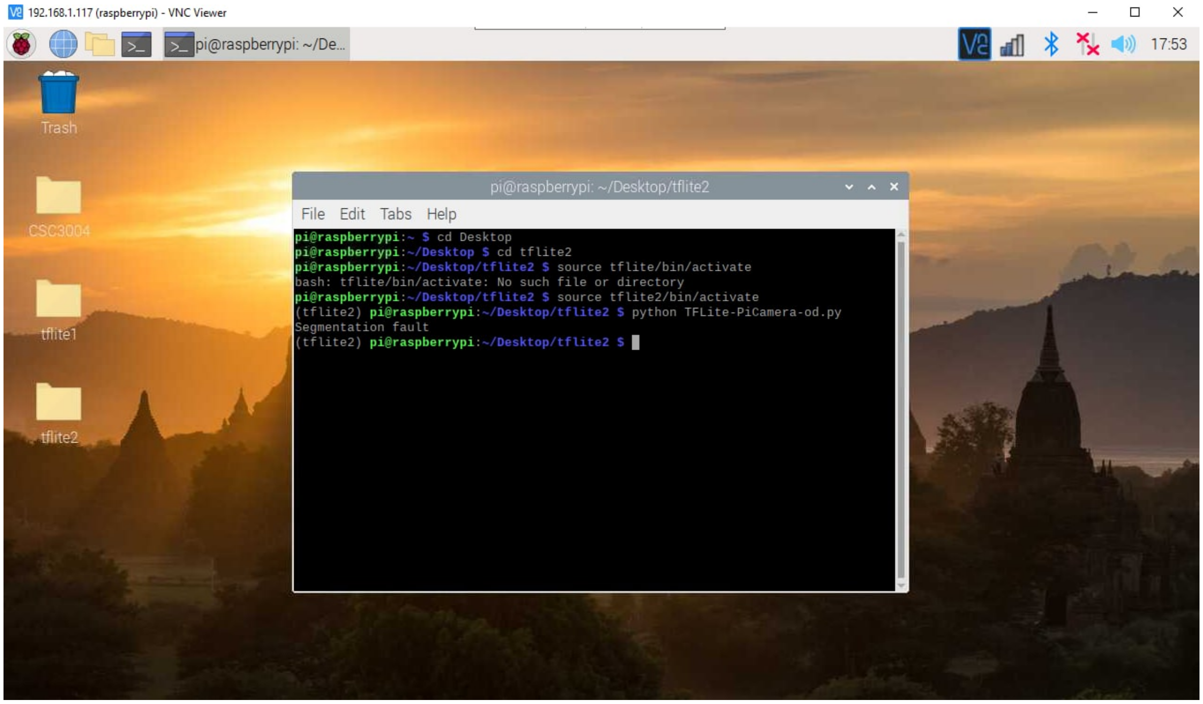




<u>Sample Image of train with the annotation</u>



However, during the deployment of the model on the Raspberry Pi, the error insisted "Segmentation Fault".



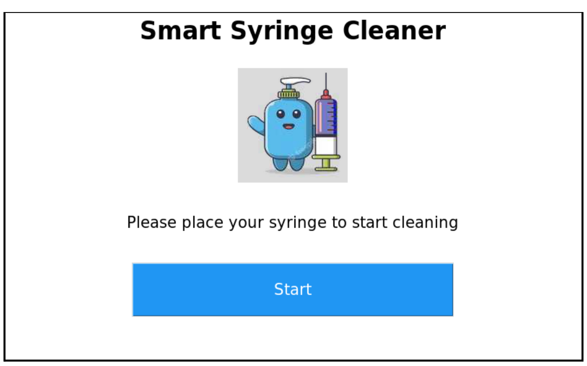
Segmentation fault error

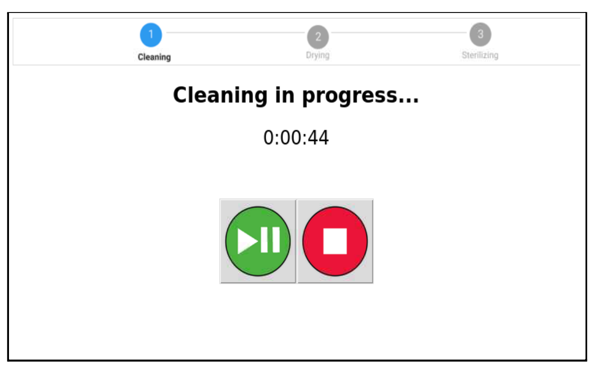
Week 2

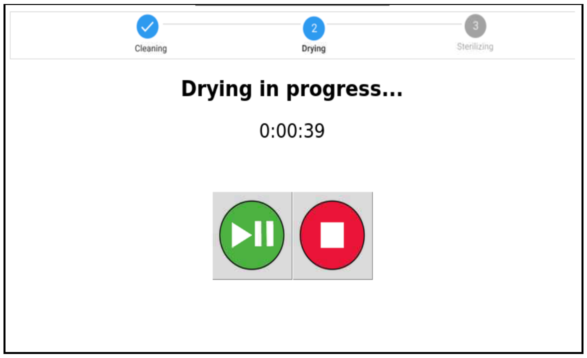
23/08/2021

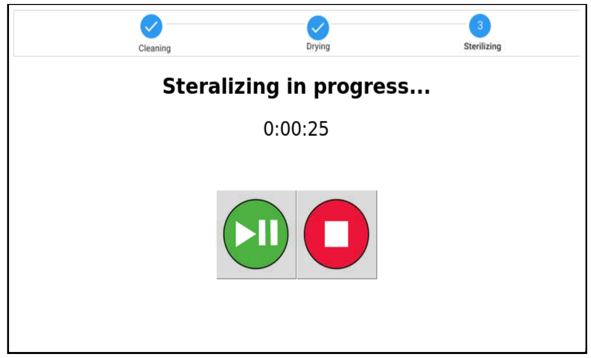
The team managed to build the final UI prototype and is ready to be implemented with the mechatronic student. The GUI function would have start, pause, resume and stop. During each stage, the timer will be reset and change accordingly.

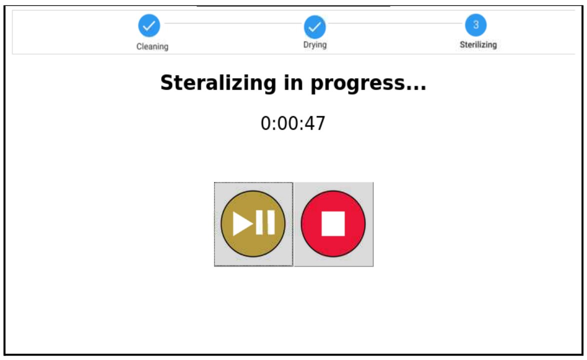
The interface has to be merged with the machine learning and arduino at a later date.

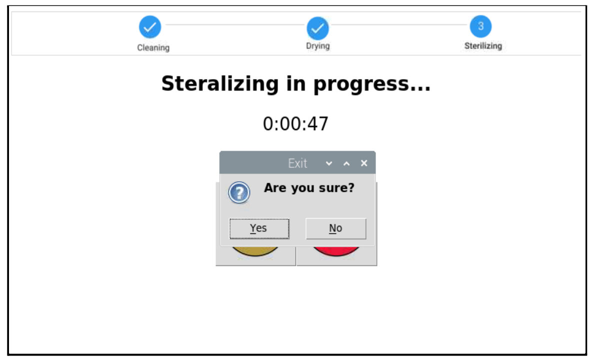


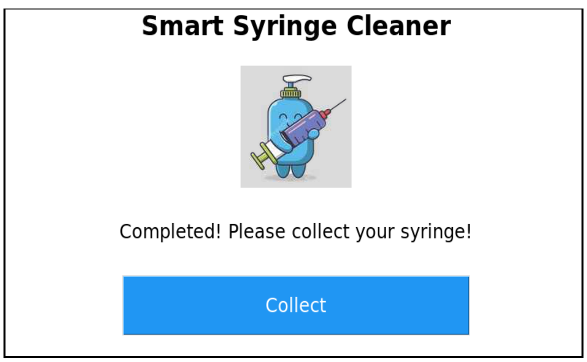












A few tests were run to ensure that the Raspberry Pi does not cause any bug to the program. Example would be the spamming of the button and the logic behind the countdown timer was correct. Further cleaning of the code was done by removing any redundant or duplicate code.

```python
class Timer():
    def __init__(self):
        # Configure the timer under the reset function
        self.timer = 20
        self.action = False

    def setlabel(self, label, currentClass):
        self.label = label
        self.currentClass = currentClass

    def countdown(self):
        # Initial is false
        if self.action == True:
            if self.timer < 0:

                nextFrame(CONTROLLER, PAGE, self.currentClass)
            else:
                seconds = self.timer % (24 * 3600)
                hour = seconds // 3600
                seconds %= 3600
                minutes = seconds // 60
                seconds %= 60
```

Setting the timer into a class

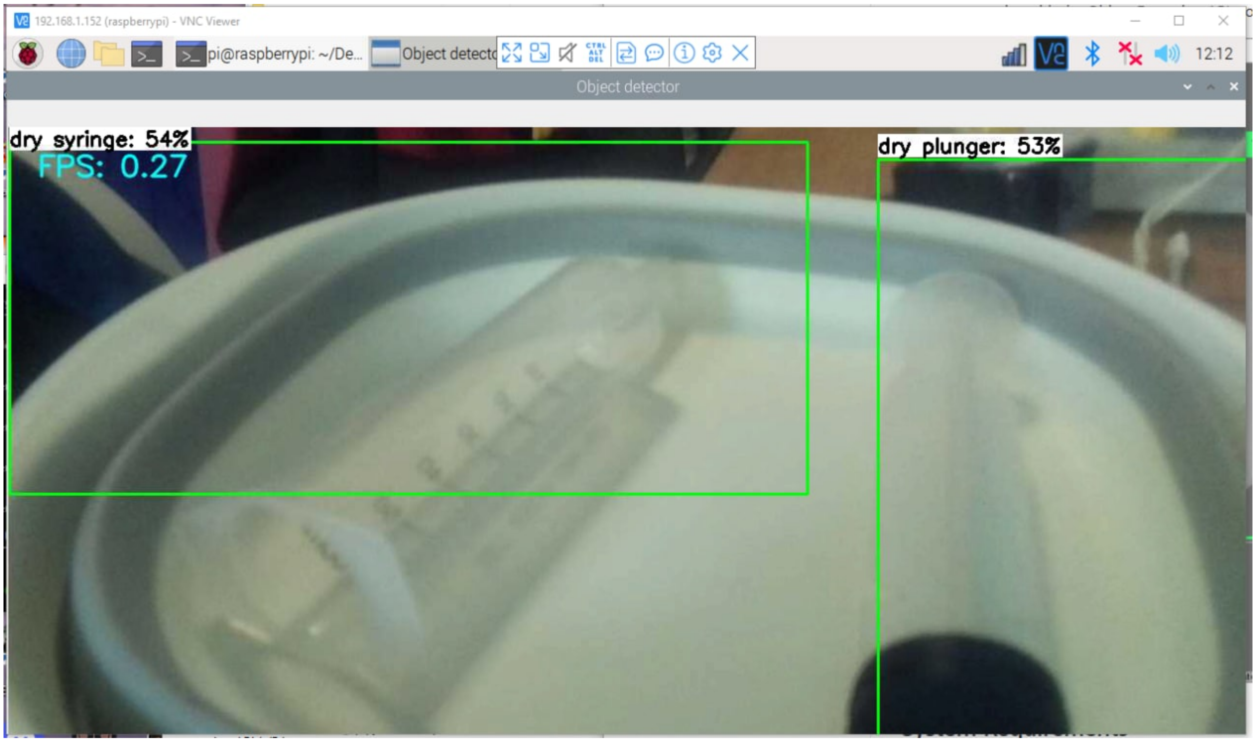
24/08/2021

The model was able to run at the raspberry pi but at the speed of 0.3 FPS. EdgeTPU does not work as it does not increase the speed of the FPS. Although they have stated that 112 operations will be running on the TPU, the FPS still remained at 0.3 FPS.

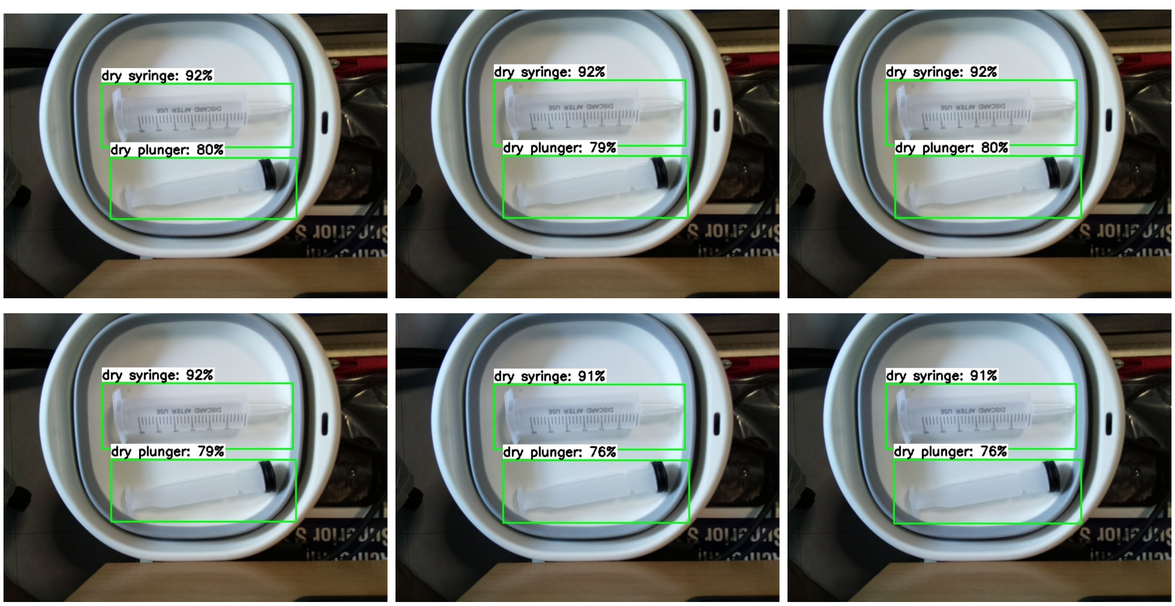
I believe that the "Segmentation fault" was caused by the quantization of the model train which took up a lot of resources from the Raspberry Pi 3B+. Using the tflite converter supported by tensorflow solved my problems and allowed the raspberry pi to read my model.

```
Edge TPU Compiler version 16.0.384591198
Started a compilation timeout timer of 180 seconds.

Model compiled successfully in 2049 ms.

Input model: edge_quant.tflite
Input size: 4.18MiB
Output model: edge_quant_edgetpu.tflite
Output size: 4.78MiB
On-chip memory used for caching model parameters: 3.23MiB
On-chip memory remaining for caching model parameters: 3.64MiB
Off-chip memory used for streaming uncached model parameters: 0.00B
Number of Edge TPU subgraphs: 1
Total number of operations: 162
Operation log: edge_quant_edgetpu.log

Model successfully compiled but not all operations are supported by the Edge TPU. A percentage of the model will instead
Number of operations that will run on Edge TPU: 112
Number of operations that will run on CPU: 50
See the operation log file for individual operation details.
Compilation child process completed within timeout period.
Compilation succeeded!
```

Compilation of the model to allow CoralTPU to run



FPS speed detection



Tflite Webcam detection

Environment testing

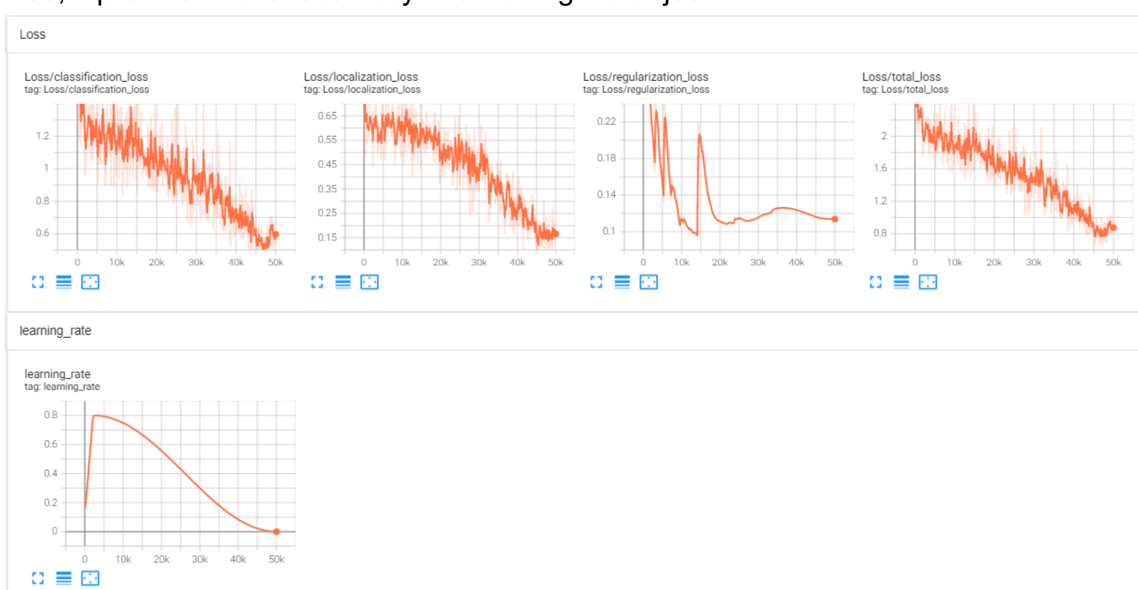
25/08/2021

Despite the installation and compilation of the EdgeTPU, the FPS of the model was at 0.3. It was running extremely slow and the detection speed was unbearable. A few thought processes was that the model was too heavy for the Raspberry Pi 3B+ to handle although it was able to run the model.

Hence we train another model on “ssd_mobilenet_v2_320x320_coco17_tpu-8”. The model was running faster than the previous model that we have trained. It has reached an FPS of 1.5. The CoralTPU was also working normally with an estimated FPS of 15.87.

Coral TPU enabled

However, the total loss did not reach 0.3 which does not fit our requirement to use the model. Hence we will continue to use the SSD_Mobilenet_v2_FPNLite640x640 since we will be inputting and analyzing the image from the picture that was took by the Raspberry Pi camera. Also, it provides better accuracy in detecting the object.

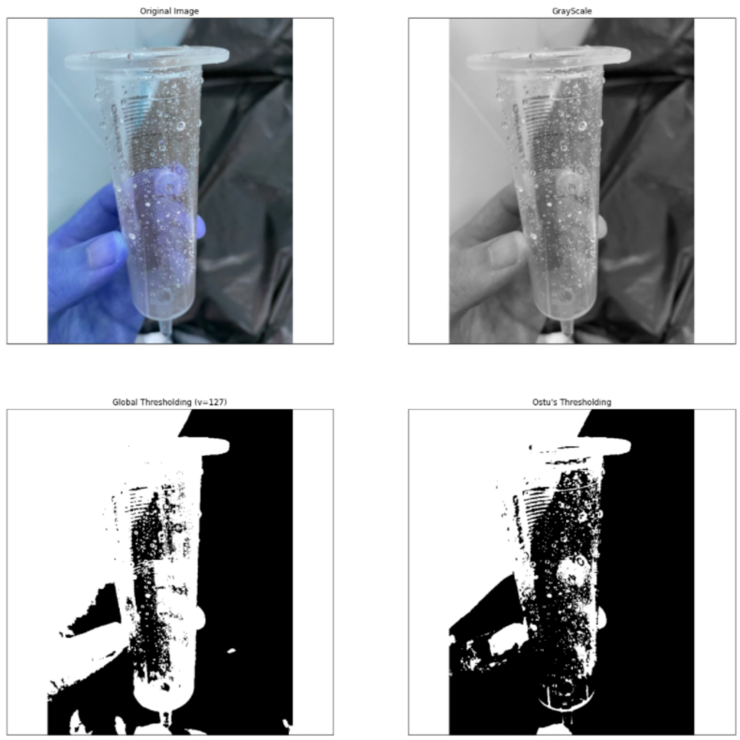


Ssd_mobilenet_v2_320x320_coco17_tpu-8 evaluation

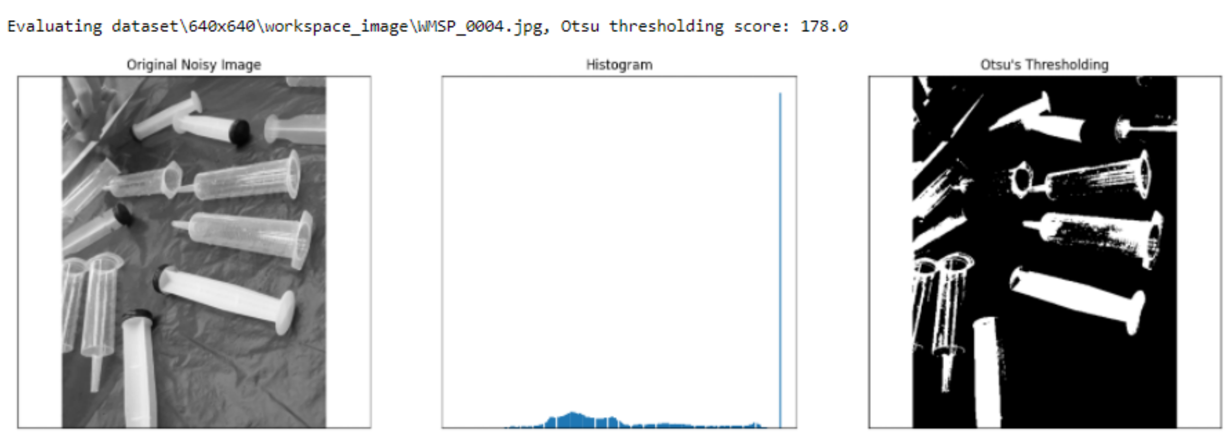
26/08/2021

Checked on how to further optimize the precision of the machine learning data. The ostu thresholding is an interesting enhancement method that could be good as it eliminates most noise in the original picture.

However, due to the limited time and picture taken, we have decided to use grayscale instead to test if the detection accuracy would be better than the original image. Also, we have to relabel all the images after resizing it to 640 x 640.

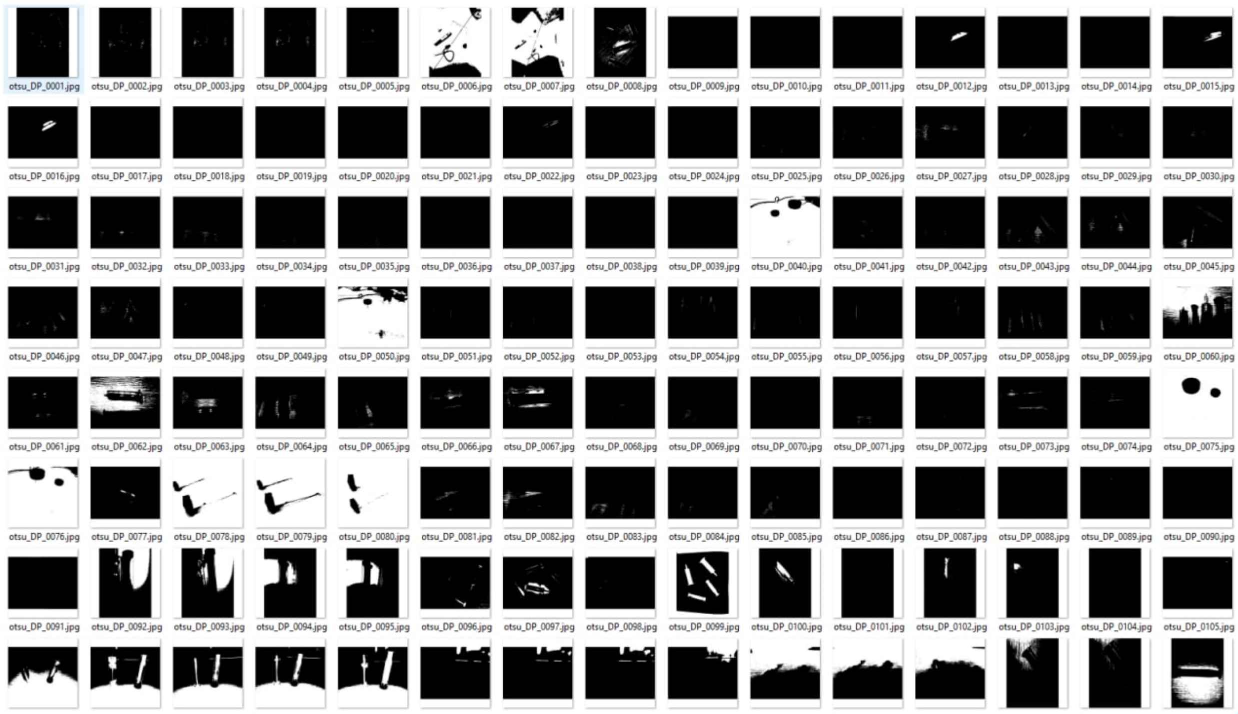


The analysis between the image classification

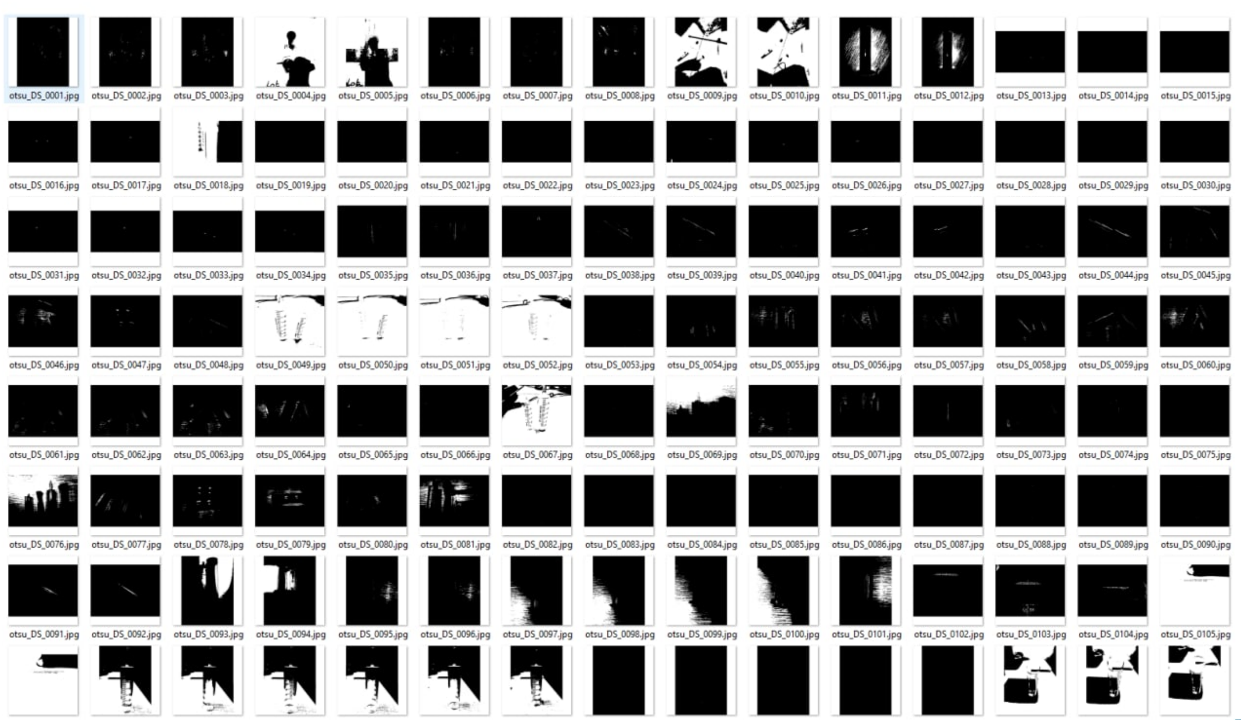


Analysis of the ostu thresholding

Ostu Dry Plunger



Ostu Dry Syringe



Ostu Wet Syringe

As you can see from the example of the ostu images that were converted, some of the images were black and did not have any distinct features. Hence we decided to train the model to fit the grayscale images.

Grayscale for moist syringe and plunger

[1] https://towardsdatascience.com/increasing-performance-when-detecting-small-objects-using-data-crops-d9b8daabd18

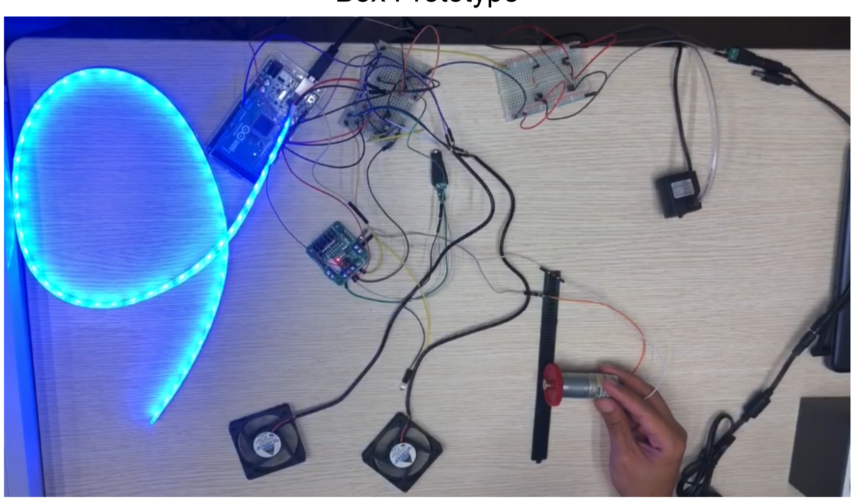
[2] https://towardsdatascience.com/image-enhancement-techniques-using-opencv-and-python-9191d5c30d45

27/08/2021

The team met up to integrate the whole product together. We have successfully integrated the whole product together and simulate how our product will actually work. The team first tested the serial communication by sending a string of commands to check if the arduino is able to execute the hardware. The test was a success and we proceeded to integrate the whole product.

The GUI from the Raspberry Pi will be displayed on the LCD and each button click will send a string to give the command to the arduino. There were a few codes changes to sync the Raspberry Pi and Audrino.

Box Prototype



Mini Integration and testing of the products

Week 3

30/08/2021

Trained and did a comparison between the color images and grayscale images. It appears that the grayscale images are able to precisely detect the model.

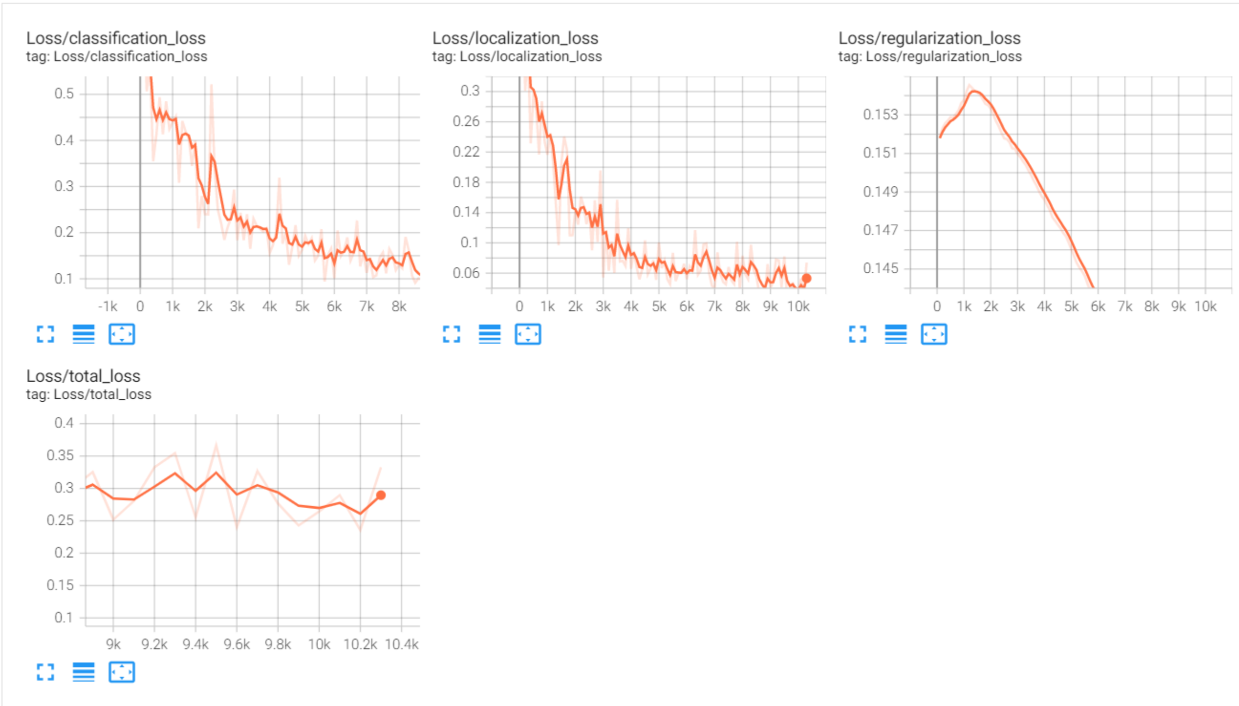
Name	Total Images	Type	Comments
model1.tflite	134 wet syringe 180 moist syringe 180 wet plunger 130 dry syringe 130 plunger	Color	Unable to detect the syringe fully. Detect dry syringe as a moist syringe.
model2.tflite	134 wet syringe 180 moist syringe 180 wet plunger 180 dry syringe 180 plunger	Color	The syringe detection is stable. The confidence level in detecting the syringe does not exceed 50%.
model3.tflite	134 wet syringe 180 moist syringe 180 wet plunger 130 dry syringe 130 plunger	Grayscale	Able to detect moist and wet syringes. However, the model is slightly unstable in detecting a dry syringe.
model4.tflite	134 wet syringe 180 moist syringe 180 wet plunger 180 dry syringe 180 plunger	Grayscale	The model can detect and classify the syringe. The confidence level in detecting the syringe does exceed 50%

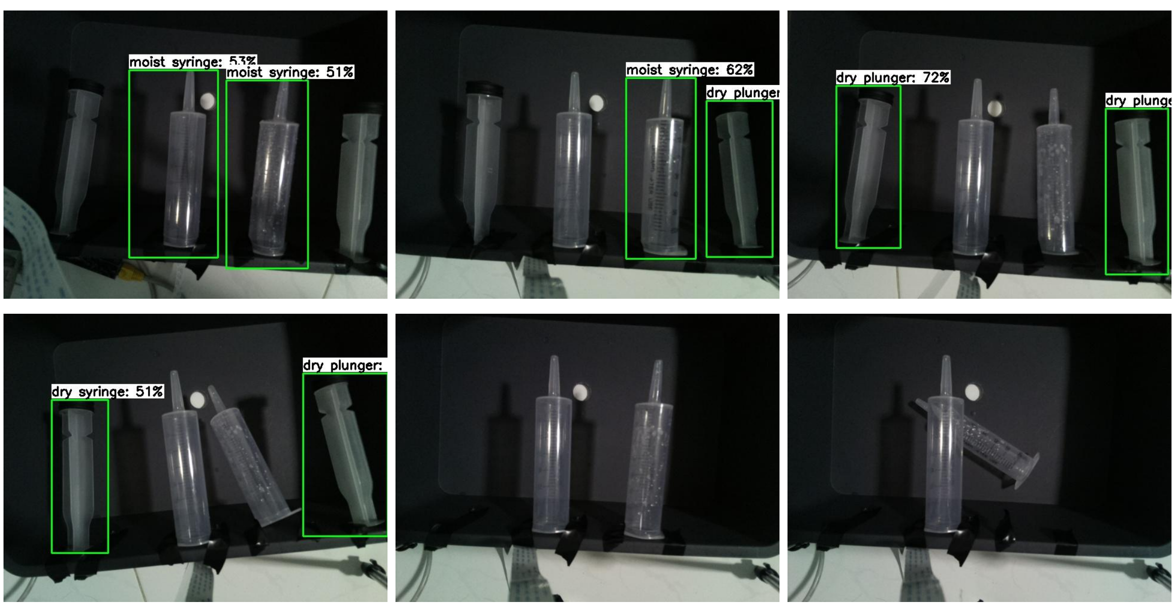
Preparation of dataset for each model

Name	Total Loss rate	Average Precision	Average Recall	Detection box Precision	Detection box Recall
model_1.tflite	0.415363	0.693	0.675	0.692977	0.775962
model_2.tflite	0.412212	0.674	0.679	0.61745	0.679014
model_3.tflite	0.423048	0.662	0.664	0.662004	0.663724
model_4.tflite	0.397220	0.716	0.691	0.716217	0.690871

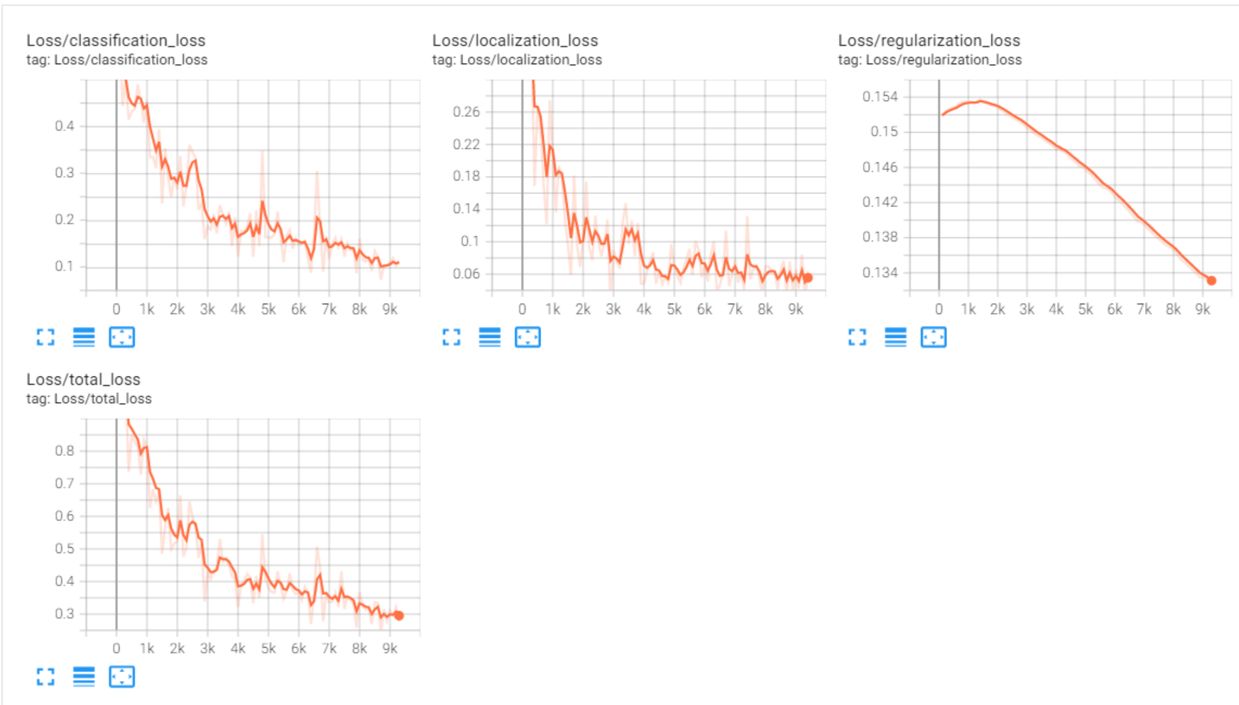
A few test runs were done a few times to check if the model is stable for each model.

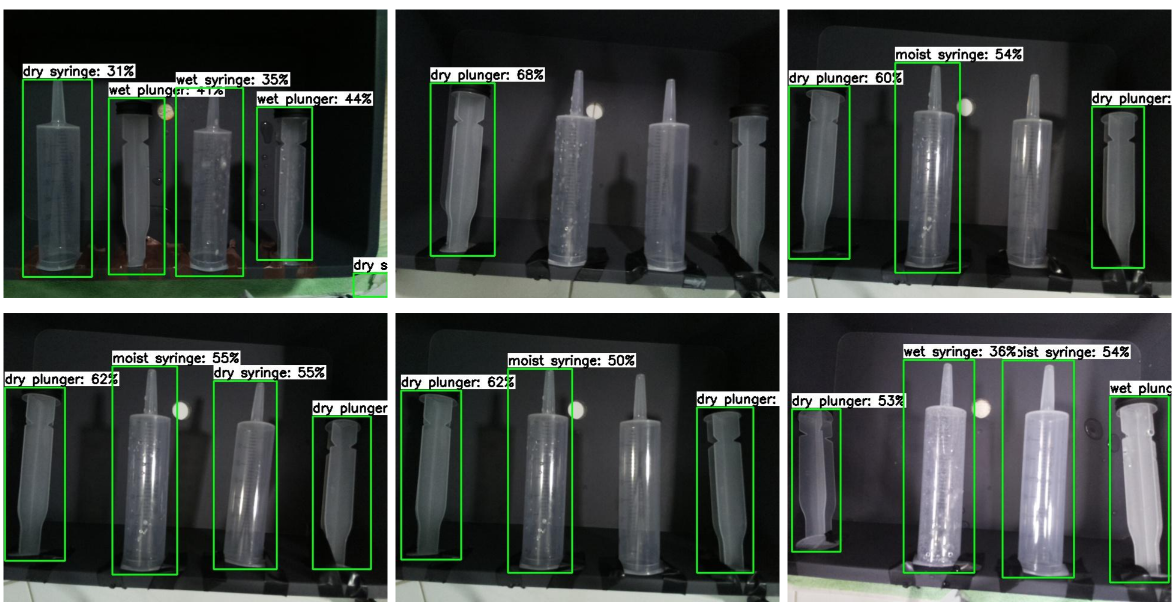
<u>Model 1</u>



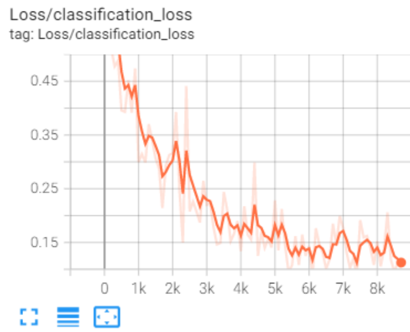


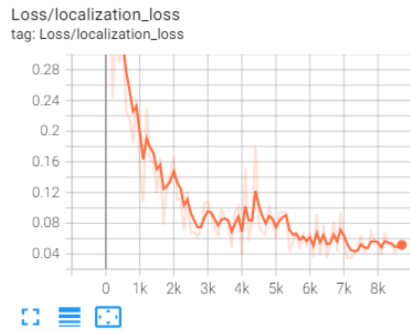
<u>Model 2</u>

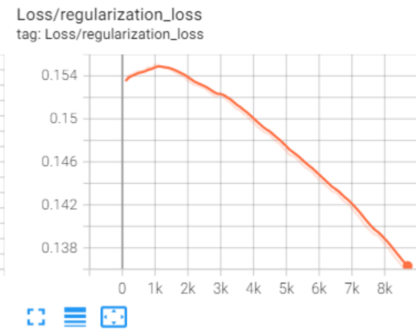

Loss/classification_loss
tag: Loss/classification_loss
0.4
0.3
0.2
0.1
0 1k 2k 3k 4k 5k 6k 7k 8k 9k
Loss/localization_loss
tag: Loss/localization_loss
0.26
0.22
0.18
0.14
0.1
0.06
0 1k 2k 3k 4k 5k 6k 7k 8k 9k
Loss/regularization_loss
tag: Loss/regularization_loss
0.154
0.15
0.146
0.142
0.138
0.134
0 1k 2k 3k 4k 5k 6k 7k 8k 9k
Loss/total_loss
tag: Loss/total_loss
0.8
0.7
0.6
0.5
0.4
0.3
0 1k 2k 3k 4k 5k 6k 7k 8k 9k

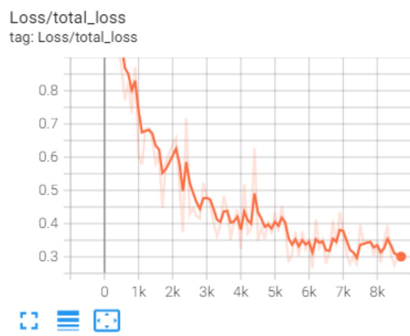

dry syringe: 31%
wet syringe: 35%
wet plunger: 44%
dry s
dry plunger: 68%
moist syringe: 54%
dry plunger: 60%
dry plunger:
moist syringe: 55%
dry plunger: 62%
dry syringe: 55%
dry plunger
moist syringe: 50%
dry plunger: 62%
dry plunger:
wet syringe: 36%
dry plunger: 53%
wet plunc

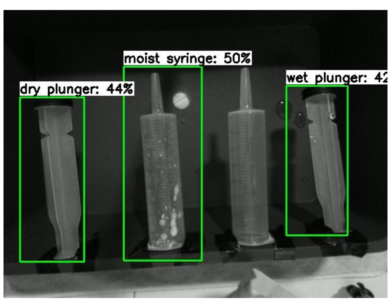
<u>Model 3</u>

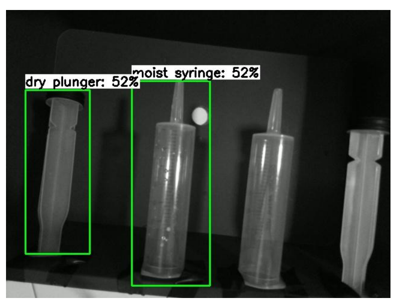

Loss/classification_loss
tag: Loss/classification_loss
0.45
0.35
0.25
0.15
0
1k
2k
3k
4k
5k
6k
7k
8k

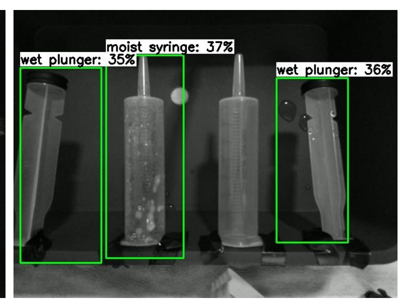

Loss/localization_loss
tag: Loss/localization_loss
0.28
0.24
0.2
0.16
0.12
0.08
0.04
0
1k
2k
3k
4k
5k
6k
7k
8k

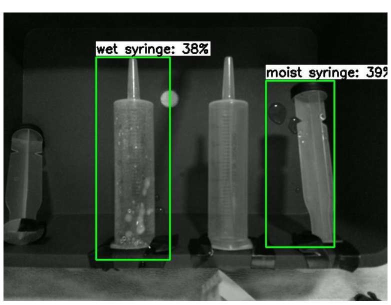

Loss/regularization_loss
tag: Loss/regularization_loss
0.154
0.15
0.146
0.142
0.138
0
1k
2k
3k
4k
5k
6k
7k
8k

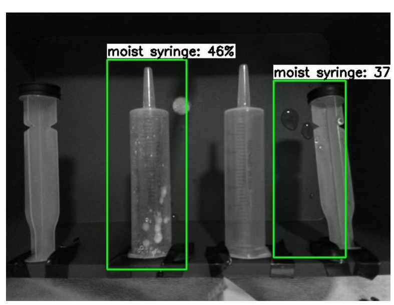

Loss/total_loss
tag: Loss/total_loss
0.8
0.7
0.6
0.5
0.4
0.3
0
1k
2k
3k
4k
5k
6k
7k
8k


moist_syringe: 50%
dry_plunger: 44%
wet_plunger: 42

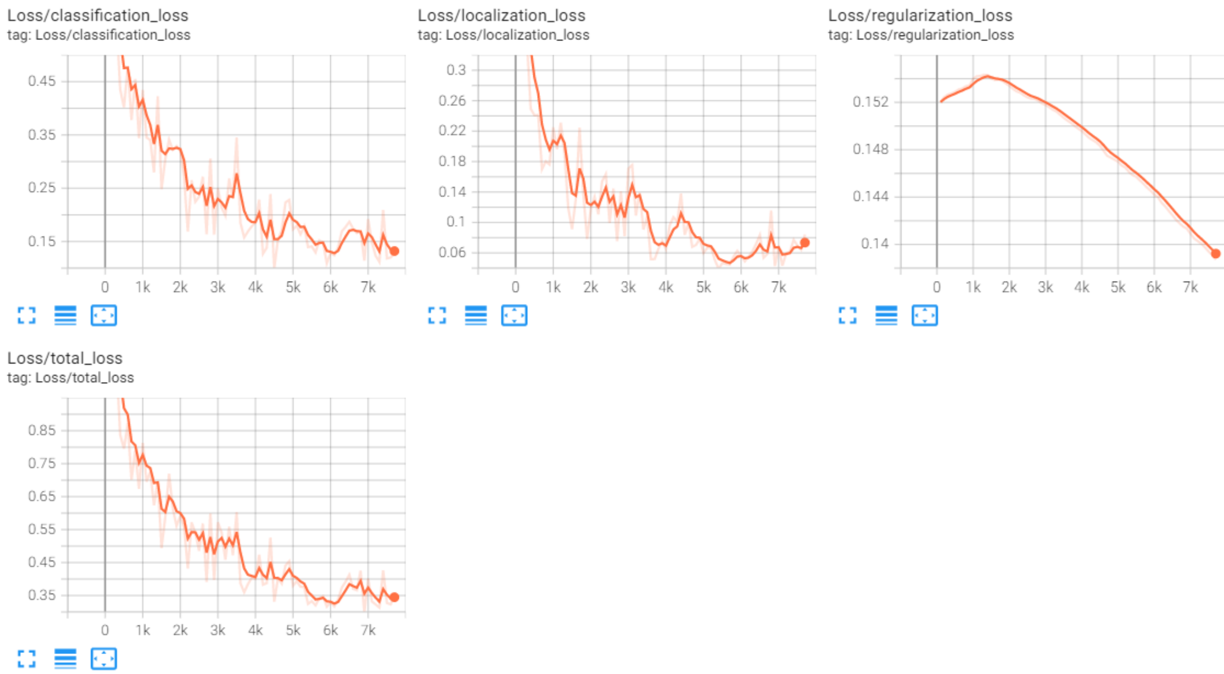

dry_plunger: 52%
moist_syringe: 52%

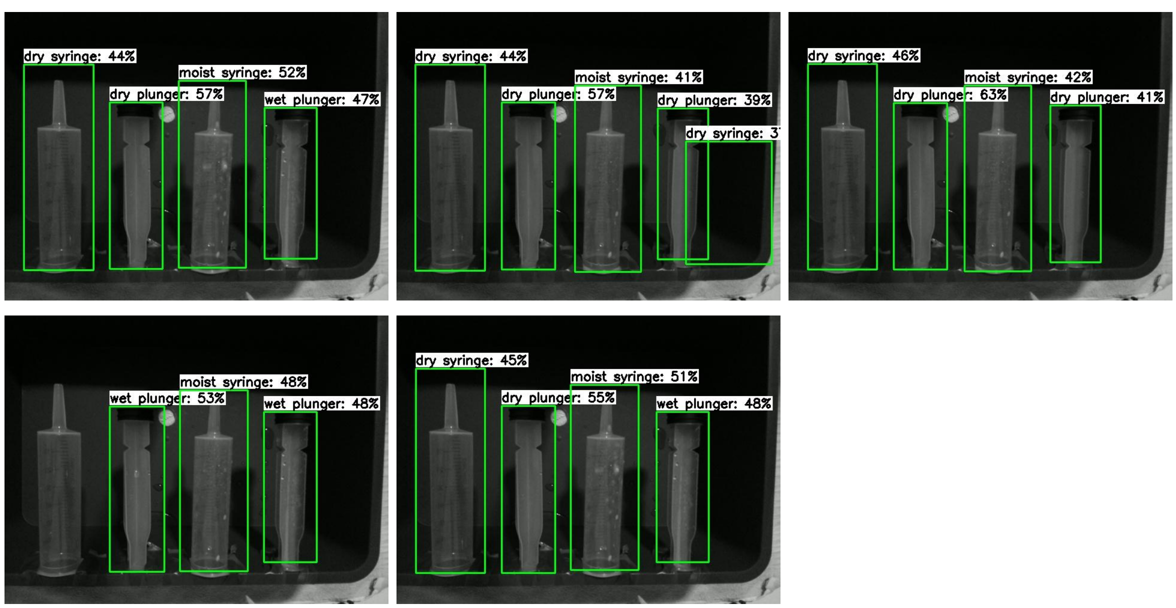

moist_syringe: 37%
wet_plunger: 35%
wet_plunger: 36%


wet_syringe: 38%
moist_syringe: 39

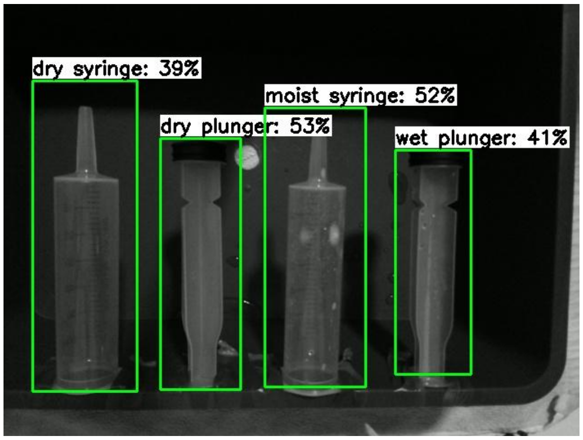

moist_syringe: 46%
moist_syringe: 37

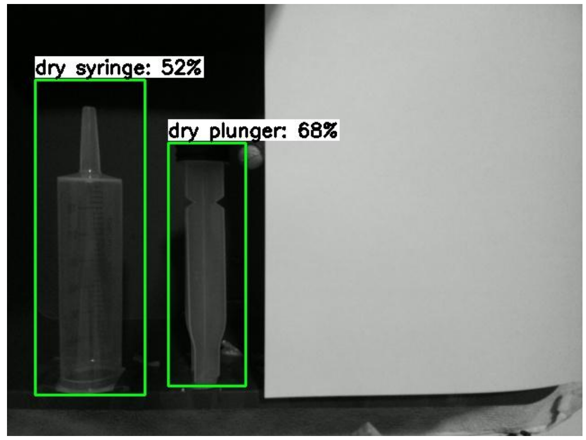
<u>Model 4</u>





If we were to compare the between the 4 models, model 4 has the best result and precision. Hence we will be using this model 4 for our final integration. The GUI was also tested with the model to check if it could accurately proceed to the next steps if only dry syringes were detected.



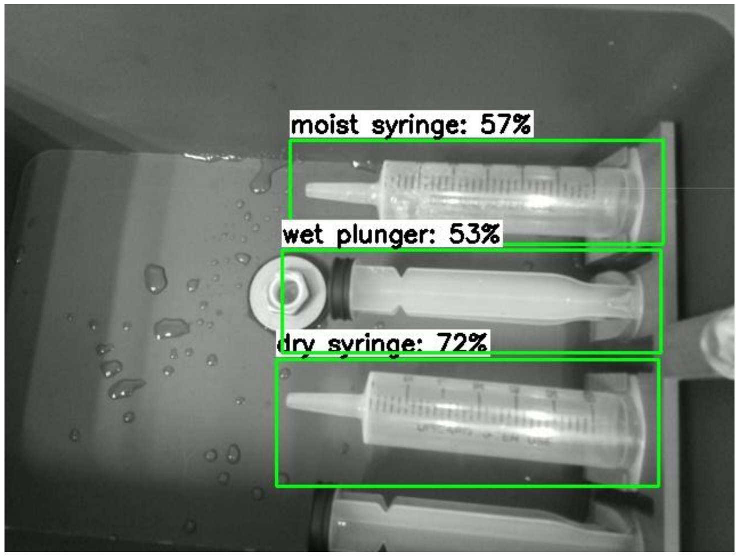


31/08/2021

Full integration of all design and technical aspect

The team met again and tried to integrate the full product for the demo video. The product is able to run the full process at each step accurately. Although there was a slight delay of executing the arduino, it does not affect much for each of the running processes.

The machine learning model also seems to be stable in detecting the moist and wet syringes/plunger.



Detecting of the syringe

The team recorded each component for the video making. Also, we discussed how we are going to separate the slides and report to the different disciplinary.

01/09/2021

The consultation stated that the video required us to do our selling point. The team is also required to list down the functional requirement in the video and to show their process flow. Minor changes were done on the video.

The demo video was done and the team started to do their report on each of their specializations.

02/09/2021

Edited the closing ceremony slides

[1] https://docs.google.com/presentation/d/1rx4t62JtVdEuW8__MFq94rVBx3OsZPhRLgnnSConOgs/edit#slide=id.gea52437898_28_0

The integration of the video is almost done together with the presentation slides. The team continued on their report writing.

03/09/2021

Today marks the final day of the OIP, it was a fun experience as I got to learn something new such as computer vision. Also, this OIP has allowed us to learn and understand how the different courses work. It was a fun and memorable experience in Singapore even though we are unable to visit Glasgow! Cheers!