

1708165

by Adam Bedner

Submission date: 08-May-2022 11:44PM (UTC+0100)

Submission ID: 179022129

File name: 13694_Adam_Bedner_1708165_179603_69563383.pdf (1.97M)

Word count: 13289

Character count: 69608



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

Course: BSc (Hons) Electrical and Electronic Engineering
Module: SGEL6000 Independent Project
Assignment: 2. Project Design, Implementation and Evaluation
Name: Adam Bedner
University number: 1708165
Date submitted: 08/05/2022
Word count: 5894

Table of Contents

Table of figures	3
List of Tables	3
Acknowledgment	4
1. Executive Summary	5
2. Introduction	6
3. Design and implementation	7
3.1 Functional Design Specification Changes	7
3.2 Outline	8
3.3 Electrical Design	10
Power	10
RaspBerry Pi to sensors and PLC interface	11
3.4 Software	12
Teachable Machine (Keras Model)	12
Python Code	20
4. Testing and Results	24
5. Evaluation and the Wider Issues	29
6. Conclusions and Recommendations	31
7. References	32
Appendix A. Folder Structure	33
Appendix B. Code and Codes modules	34
Classes.py	34
Defs.py	35
B.py	39
CSS.py	40
Appendix C. Simulation Results (output from terminal)	43
Appendix D. Sample Photos for Keras Model Training	53
Pass Photos	53
Reject Photos	54
Python Code for Sequential Picture Taking	54
Appendix E. Electrical Drawings	55
Appendix F. Teachable Machine Definitions	60
Appendix G. Log book	63

Table of figures

Fig. 1 System set-up overview.	9
Fig. 2 System's block diagram	10
Fig. 3 Teachable machine, model training page	12
Fig. 4 Confusion Matrix for initial values	13
Fig. 5 Accuracy per class for initial values	13
Fig. 6 Advanced menu, initial values.....	13
Fig. 7 Accuracy per epoch for initial values.....	14
Fig. 8 Loss per epoch for initial values	14
Fig. 9 Confusion matrix, part way though parameter adjustment	15
Fig. 10 Accuracy per class, part way though parameter adjustment	15
Fig. 11 Advanced menu, part way though parameter adjustment.....	15
Fig. 12 Accuracy per epoch, part way though parameter adjustment	16
Fig. 13 Loss per epoch, part way though parameter adjustment.....	16
Fig. 14 Confusion Matrix, final values	17
Fig. 15 Advanced menu, final values.....	17
Fig. 16 Accuracy per class, final values	17
Fig. 17 Accuracy per epoch, final values	18
Fig. 18 Loss per epoch, final values.....	18
Fig. 19 Mind map for code development.....	20
Fig. 20 Example of a pass box.	26
Fig. 21 Example of a reject box.	26

List of Tables

Table 1 Project expenses	8
Table 2 Comparison of model training parameters and results	19
Table 3 Comparison of Confusion matrixes from model training.....	19
Table 4 Testing Table	24
Table 5 Testing table adjusted for simulation with simulation test results.....	26

Acknowledgment

To begin with, I would like to thank Steve Wildridge and Marcus Jones for the great support received and for their sharing of their knowledge, company's internal connections and time accommodation for the research and design stages of the project, without their support, it would not have been possible to finish this paper.

I would also like to thank my lecturers Rob Davies, Rob Jenkins and Mike Hall for their support with time management, knowledge and allowing access to the very much needed resources and facilities to complete part of this project.

Special acknowledgment to Jake Stevens for major support with Python code development and knowledge shared to debug many of the errors along the developing way.

Special thanks to my partner, Natasha, and my parents and sisters for the emotional support through the difficult times throughout the last few months. I don't think I would have made it without their support.

1. Executive Summary

This report describes the design and simulation stages of the Camera imaging project. The system developed could not be tested in the real world due to production schedule; it is scheduled for June/July of 2022.

The simulation shows that the system is capable of performing in accordance with the first 5 specifications of the FDS. Obviously, there were major constraints as of what could and what could not be tested during simulation, like the safety of the system, risk assessment or how much the *box closer* had to be modified to accommodate the new system. Also, point 6 of FDS, IIOT combability, is not happening; the company sees no benefit for it at the current moment but would like to have the chance to upgrade in the future if they were to decide to do so.

The design of the hardware was a bit more complex than expected. As the company did not approve on IIOT, they also disapproved the use of the RUBUS interface, forcing the old-school relay control interface into the system; that affected reliability and maintainability (relays wear over time) of the system. That interface had to be put in place to connect the industry standard 24VDC controls with the non-industry standard 5V and 3V3 DC control voltages. Opto-isolators could be an upgrade for the future but the design (Hat PCB Design for the Pi would probably be required) would be costlier and more difficult for future fault finding. *Your FDS could have been more thoroughly researched to find out the company needs*
Your solution should have been agreed with the company before implementing

The most time-consuming part of the project was the keras model and python code development. The keras model training required quite an extensive trial-and-error adjustment of parameters to achieve a satisfactory model that would be both accurate and stable.

The python code was complex to develop as it runs on a line-to-line basis without rescanning the IOs every run. To overcome that and have more than one section of the code executing at a time, the code was split in multiple .py files; creating modules. Those modules were then called and executed alongside the main code. There were attempts to overcome this issue with different methods such as threading and multiprocessing in an attempt to keep everything within one .py file. That proved to be more complex and would have been more time-consuming.

Your summary doesn't say if the system actually works

2. Introduction

This report is the continuation of the research and proposal report ' 1. *Project Definition, Research and Quality Planning*' written by Adam Bedner. (Bedner, 2021). It presents in detail the design and testing processes of the proposed solution of the following problem.

The presented problem is a misplacement of a Raspberry Pi within the packaging which causes blockages in the *box closer* machine, critically damages the Pi and runs the risk of causing damage to the *box closer*. The proposed solution involves the use of a Raspberry Pi together with a Rubus interface and a HQ Pi Camera to take pictures of the boxed Pi before closing the box and using a keras model trained using Teachable Machine by Google to determine if the Pi is sitting properly within the box. If the Pi is not within the boundaries of the box, it will be rejected using an air knife to an unused section of the conveyor belt where it can be inspected and placed back on main part of the conveyor once corrected. The project is proposed to improve the reliability of the line and minimize the waste from crashes as well as limit the losses due to machine stops. This step will improve environmental awareness (limiting waste where possible) as well as promoting lean manufacturing by limiting the physical waste as well as time wasted every time the machine stops. The average waste is of 20 boxes per minute, one every 3 seconds on average.

3. Design and implementation

3.1 Functional Design Specification Changes

The functional design specifications (FDS) are stated by Adam Bedner in his report 1. 1. *Project Definition, Research and Quality Planning*: “

Spec #	Specification	Description
1	Misplaced product recognition	It must recognise a product that has been misplaced in such a way that it is sticking out of the edges of the box (in any direction).
2	Compatibility with different models.	The ability to work with more than one model. Those vary slightly in the component type and position. The overall size of the product remains the same across all models. It is not required for the system to reject wrong products; different systems are responsible for that.
3	Cooldown time <3s	In order for the system to keep up with the line, it must scan every 3 seconds or less. Or 1200 scans an hour.
4	Jam condition detection	The system will be capable of detecting a jam up product boxes (abnormal stop of ABC)
5	Misfire condition detection	Detection of an actuator misfire, causing a reject to carry on to ABC. If a condition is detected, a stop signal will be sent to ABC.
6	IIOT-compatibility	~~Network connectivity for IIOT. It must be capable of transmitting data about the passes and fails, as well as sending the images of rejected products.~~ Should never have been in the FDS
7	Minimal machine modification for installation	The system must not require extensive machine modifications for installation.
8	No interference with existing safety	The system must not interfere with the existing safety system of the machine. Those are Cat 4 Safety Circuits.
9	Cat 4 safety compliant	The system must be compliant with Cat 4 safety to match the safety rating of the existing machine.

” (Bedner, 2021)

Most of these remain unchanged, except for spec # 6 *IIOT compatibility*. The company has decided not to use space and resources of the server for the system, and further discussions showed that the company does not see production benefits from incorporating the system into IIOT network. They also decided not to use their internally-designed Rubus interface, which has been substituted with a relay interface. The system does remain IIOT capable in case the company would wish to upgrade in the future; it is capable due to the functionalities of the Raspberry Pi.

This shows how important the FDS is.

The agreed budget for the project was £3,000. (Bedner, 2021).

You could have considered this as a contingency in your Gantt chart

The system has not yet been fully implemented due to constraints from production schedule and lack of permission for line release. Although, estimating the cost of implementation, the final cost of the project is estimated at £2,080.16 which is well within budget agreed. The price would be about £150 more if the Pi4 and Pi Camera were not upcycled. The breakdown of the costs is presented in **Table 1**.

This table belongs in the appendix

Entry No.	Item Description	Price/Unit	Qty	Price
1	Aesthetically damaged Pi4 (Scrap)	£0.00	1	£0.00
2	Aesthetically damaged HQ Camera (Scrap)	£0.00	1	£0.00
3	LAN Camera Extension Kit	£52.00	1	£52.00
4	5V 3A Dual USB Step-Down converter	£8.20	1	£8.20
5	Low profile cooling fan	£9.00	1	£9.00
6	Tripod Swivel Ball Adapter	£2.00	1	£2.00
7	Mounting Plate for HQ Camera	£5.00	1	£5.00
8	Phoenix NO relay	£6.17	8	£49.36
9	10k Resistors	£0.16	5	£0.80
10	1k Resistors	£0.16	3	£0.48
11	PN2222ATA NPN Transistor	£0.21	3	£0.62
12	Curve B 1A 2Pole MCB (siemens)	£47.75	1	£47.75
13	Curve B 3A 2Pole MCB (siemens)	£47.75	1	£47.75
14	Omron PhotoElectric Sensor	£92.30	4	£369.20
15	Labour (Excluding instalation and commisioning)	£12.00	100	£1,200.00
16	Installation and commissioning (Estimated on hours required)	£12.00	24	£288.00
17				£0.00
18				£0.00
19	Total Cost			£2,080.16

Table 1 Project expenses

The decision to use aesthetically damaged (but fully functional) parts were made to promote environmental sustainability as well as lean manufacturing by reducing waste where possible. Aesthetically damaged parts cannot be sold to any customer and are usually thrown away.

3.2 Outline

The design employs (*Fig. 1*) 4 sensors (excluding the camera); one to trigger the coded test sequence, one to monitor any rejects present in the reject area and the other two to monitor and ensure that the right product goes where it is expected to go. This is a basic example of machine fail-safe which,

as defined in Collins Dictionary is: *“Something that is fail-safe is designed or made in such a way that nothing dangerous can happen if a part of it goes wrong. [Example:]*

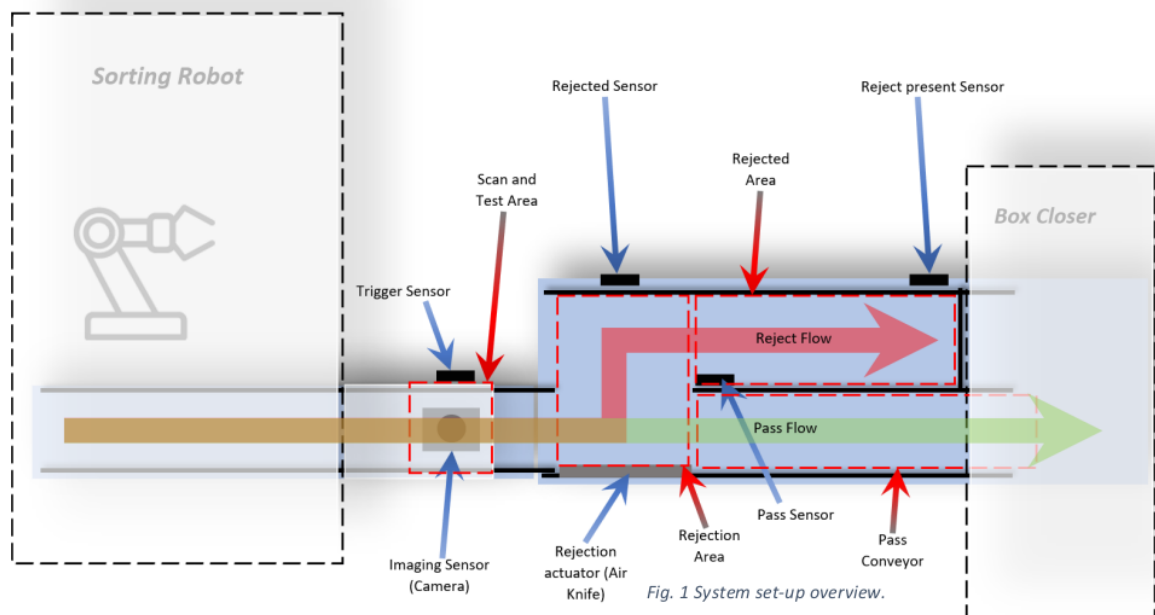
There was a fail-safe mechanism in place to override the controls in case of a malfunction.” (COBUILD Advanced English Dictionary, 2022)

The system is coded in such a way that in case of a malfunction, the system will be capable of detecting and distinguish the type of malfunction (misfire, good product rejected, blockages and jam) and feedback the information through diagnostic terminal. This self-diagnostic and fail-safe does not encompass loss of power or system crash situations as it is not within the company’s requirements. It also enables for the system to be easily disabled without affecting production, something that the company requested. That is Specification #7 stated in FDS section: Minimal machine modification on installation (Bedner, 2021). Keeping it as a standalone subsystem keeps the modifications to the existing PLC to the minimum; PLC code will only have to include an extra condition for machine stop, machine warning light and output command for when machine is stopped.

The two monitoring sensors, Rejected Sensor and Pass Sensor have a dual purpose and the two functions are differentiated based on the amount of time that the sensors are on for.

The Rejected Sensor, if on for less than 3 seconds, the Pi detects that a box has been moved to the reject area. If it is on for longer than 3 seconds, the Pi detects that the rejected area is full and sends a stop signal to the Box Closer.

Similarly, the Pass Sensor, if on for less than 3 seconds, signifies that the product went down the pass conveyor. If on for longer than 3 seconds, there is a blockage in box closer.



3.3 Electrical Design

The block diagram for the system's electrical connections is shown in **Fig. 2**. As the main power supply, an already existing 24VDC rail from *box closer* is used instead of 240VAC mains. This makes the machine safer as well as making risk assessments and H&S checks simpler and easier because of the lack of mains live voltage within the system.

As explained in the following sections, the PSU voltage is further stepped down and a relay interface was designed for the 24VDC and 3V3/5VDC signals.

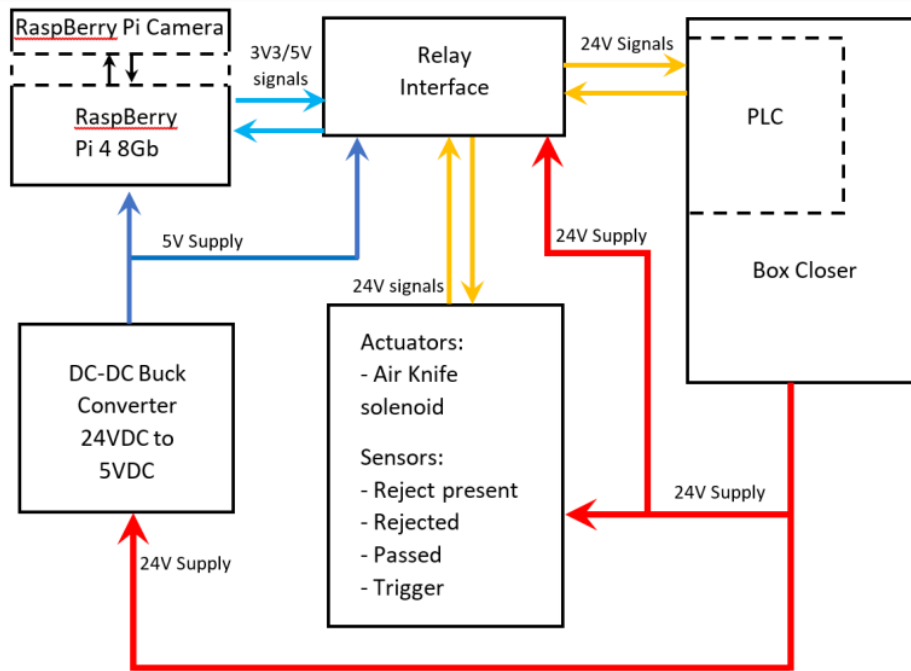


Fig. 2 System's block diagram

The final electrical drawings for the system can be found in **Appendix E. Electrical Drawings**.

Power

To power the system, an already existing 24VDC rail in the box closer was used to power the relays. This was also stepped-down to 5VDC required for the Pi. The 0V rails of both supplies are linked together to ensure that the right voltage appears where needed in reference to 0V. The 0V rail is also linked to ground as part of safety.

The DC-DC Step-Down converter is protected with 2 dual pole Thermo-Magnetic MCBs. 1A for 24VDC supply to the converter and 3A protection on the output.

RaspBerry Pi to sensors and PLC interface

Since the RaspBerry Pi GPIO outputs a maximum of 3.3VDC and takes a maximum of 5VDC, an interface is needed between the GPIO and the 24VDC IO system of the PLC and the 24VDC sensors. No 3.3V or 5V sensors can be used as these would go against the company's standard, which calls for 24VDC Omron sensors; Photoelectrical where possible. To overcome that issue a relay interface is used.

Inputs to the Pi are done using NO contacts with a 10k pull-down resistor on Pi's side. The pull-down resistor is there to add reliability and stability by avoiding floating inputs.

The outputs are driving transistors which then drive the 24V relay coils.

The relays used are the Phoenix-Contacts Relay Module 2903370. The current rating of the coil is 9mA and the mechanical service life of approximately 2×10^7 cycles (Phoenix Contact, 2019).

To drive the coil of the relay, a PN2222ATA NPN transistor is used (part of the well-established 2N2222 transistors) as it can drive up to 1A @ 40VDC Emitter-Collector (On Semiconductor, 2013). A 1k resistor is used on the base of the transistor to ensure stability and avoid current spikes on the Pi's output.

The relay's limited useful life cycle must be taken into consideration for maintainability, assuming that trigger and pass sensors are triggered every cycle:

With an average cycle time of 3s, there is 20 cycles per minute. That gives 1,200 cycles per hour or 19,200 cycles per 16hour working day.

With an average of 255 working days per year, that is 4,896,000 cycles per year considering current running schedule of the line. Rounding these numbers, the relay should last approximately 4 years; to ensure the reliability of the machine, the relays should be replaced every 3 years. When adjusting the run hours of the line, the interval should also be adjusted.

3.4 Software

Teachable Machine (Keras Model)

The teachable machine by google is an algorithm teaching tool that uses samples in two or more classes to develop an algorithm. Once trained, this algorithm can decide to which class the input data belongs.

For this project, an image processing model will be trained with two classes: Pass and Reject.

For samples, pictures were taken using the Camera and Pi that will later be installed. A total 1,000 pictures were taken for pass boards and another 1,000 for reject boards using the Python code shown in **Appendix D. Sample Photos for Keras Model Training**. The sample pictures are divided into training samples (85% of samples) and testing samples (15% of samples) as defined in **Appendix F. Teachable Machine Definitions**. This is done to feed samples that the model has not encountered before to ensure the accuracy of the trained model. This is reflected in the accuracy tables (**Fig. 10Fig. 16**) and confusion matrixes (**Fig. 4Fig. 9Fig. 14**) and, accuracy (**Fig. 7Fig. 12Fig. 17**) and losses (**Fig. 8Fig. 13Fig. 18**) per epoch graphs. The pictures were taken in sets of 100 to make it easier to set up different scenarios of pass and fail boards and change between the models.

The samples have been uploaded to the Teachable machine webpage where the model is taught (**Fig. 3**). It is safe to upload the pictures in regards for the confidentiality, as all the model training happens locally in the browser's tab. Hence why it notifies you to keep the tab open and active once the teaching process is started. (Google, 2022)

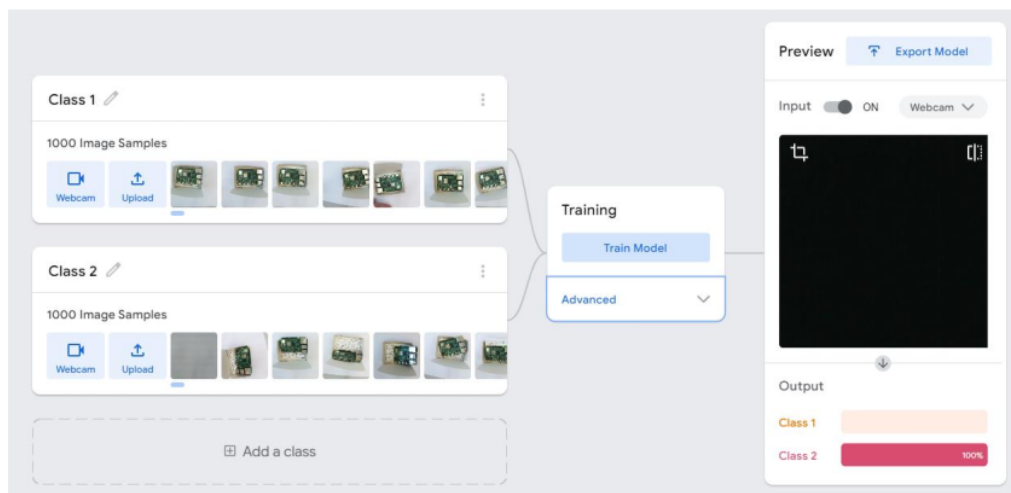


Fig. 3 Teachable machine, model training page

The training page has an advanced menu with parameters (Fig. 6) that are used to improve the training process of the model and to produce a more reliable model. The training process of the model is backed by accuracy, confusion and loss graphs and tables. The on-website definition of each (under the (?)) can be found in **Appendix F. Teachable Machine Definitions.**

Word/s missing

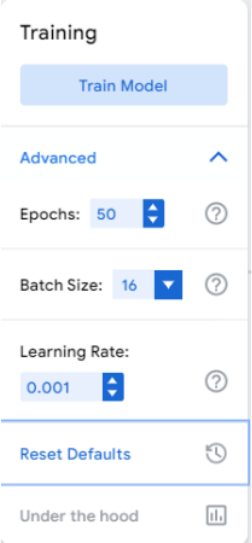


Fig. 6 Advanced menu, initial values

Accuracy per class			?
CLASS	ACCURACY	# SAMPLES	
Class 1	0.00	150	
Class 2	1.00	150	

Fig. 5 Accuracy per class for initial values

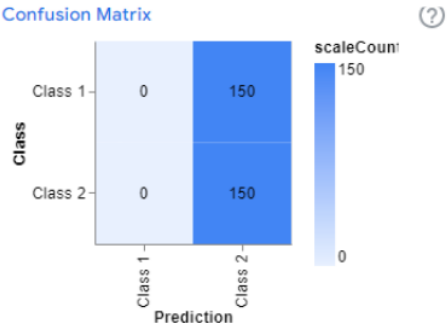


Fig. 4 Confusion Matrix for initial values

Your report is cluttered by images most of which should be in the appendix

Accuracy per epoch

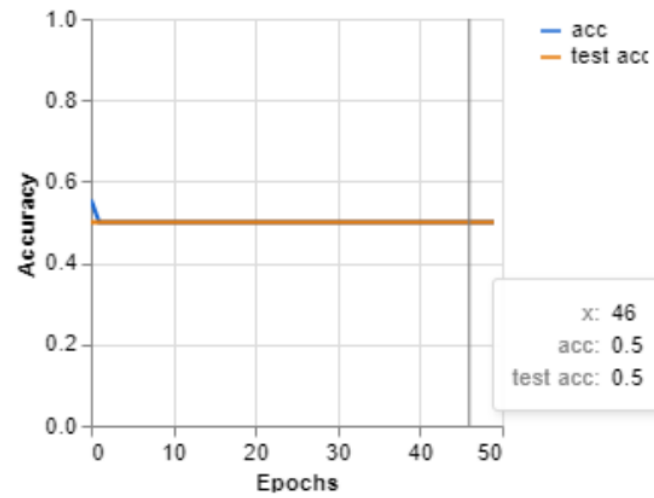


Fig. 7 Accuracy per epoch for initial values

Loss per epoch

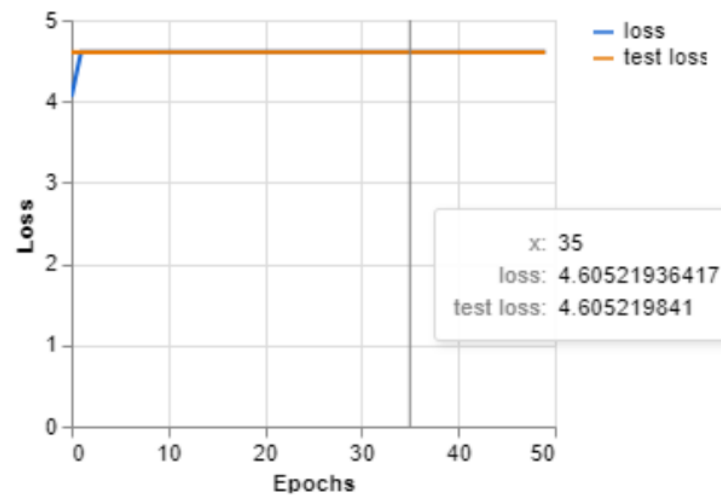


Fig. 8 Loss per epoch for initial values

The initial values are shown in **Fig. 6** which, have not produced promising results. As defined in **Appendix F. Teachable Machine Definitions**, the model is more accurate when the values in *accuracy per epoch* graphs are tending to 1, where these are 0.5 for the initial values (**Fig. 7**). Also, the closer the values in *loss per epoch* are to zero, the more confident the model is that the prediction is accurate. In this case the values are way above 0 (**Fig. 8**). Furthermore, the *accuracy per class* table shows the accuracy per class (**Fig. 5**) and the *confusion matrix* (**Fig. 4**) represents what class prediction each class's test samples have obtained, these are especially useful where dealing with more than two classes

To improve the trained model, a trial-and-error method has been used, adjusting one value at a time and monitoring the changes in training behaviour. First, the number of epochs has been increased to 500 to let the model go through the whole set of samples many more times. This alone has improved the trained model. Batch Size and Learning Rate have been adjusted in rather large amounts to obtain a clear picture of the effect of the variables on the trained model.

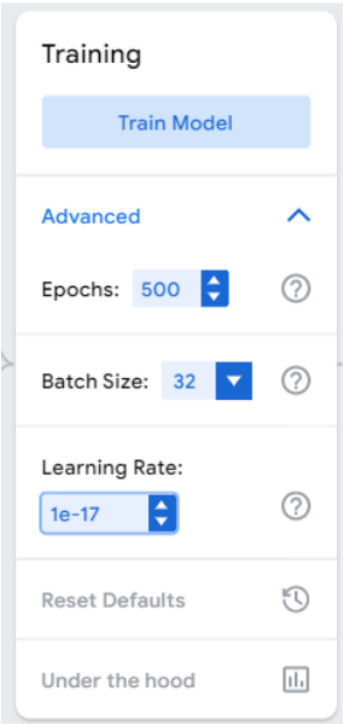


Fig. 11 Advanced menu, part way through parameter adjustment

Accuracy per class

CLASS	ACCURACY	# SAMPLES
Class 1	1.00	150
Class 2	0.93	150

Fig. 10 Accuracy per class, part way through parameter adjustment

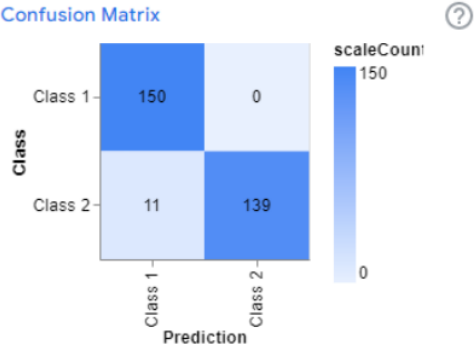


Fig. 9 Confusion matrix, part way through parameter adjustment

Accuracy per epoch

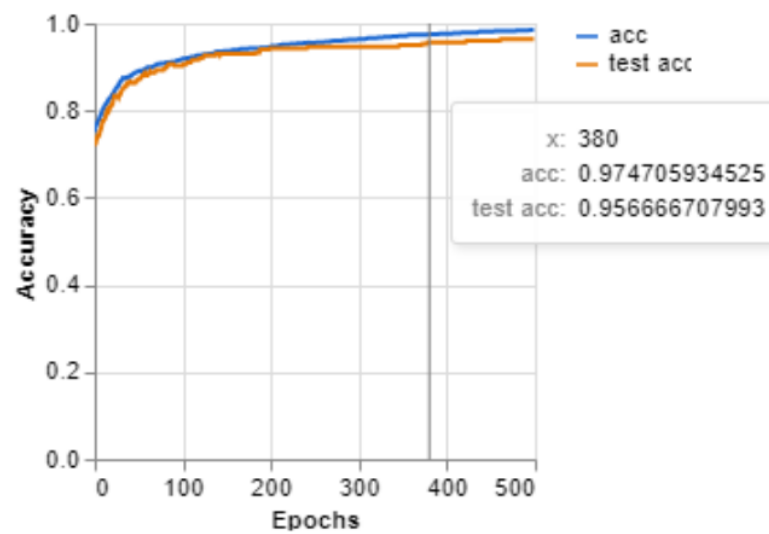


Fig. 12 Accuracy per epoch, part way through parameter adjustment

Loss per epoch

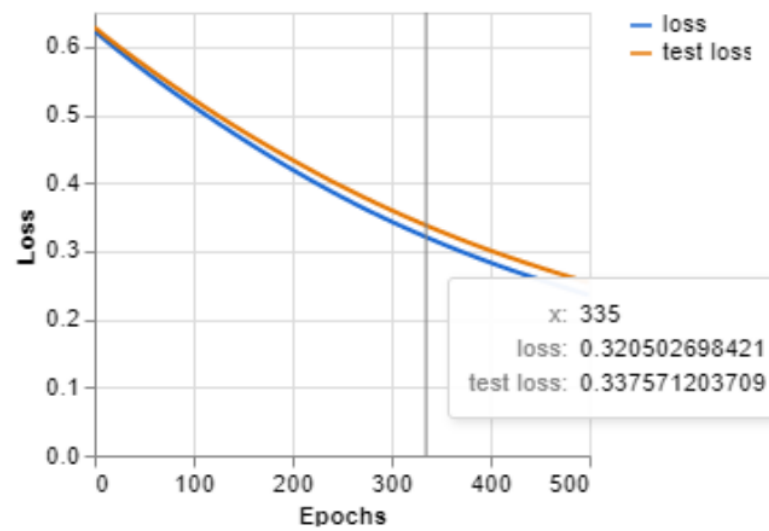


Fig. 13 Loss per epoch, part way through parameter adjustment

The results obtained from this parameter adjustments were a lot more promising, but not satisfactory. The graphs in **Fig. 10****Fig. 11** shows that, despite the great improvement, the model still lacks accuracy and confidence in its predictions. This is also reflected in the graph and matrix where class 1 (Pass) is 100% accurate, but class 2 (Reject) is still struggling.

The next step was to fine tune the parameters until a satisfactory model is trained. The final values are presented in **Fig. 15**. As accuracy per epoch is 1 (**Fig. 17**) and loss per epoch is >0.001 (**Fig. 18**), meaning that the trained model is 100% accurate and very confident in its prediction. This is backed up with the 100% accuracy per class (**Fig. 16**) and no confusion between classes as shown in the confusion matrix (**Fig. 14**).

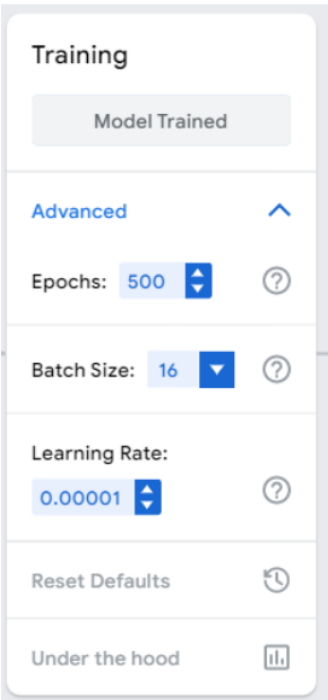


Fig. 15 Advanced menu, final values

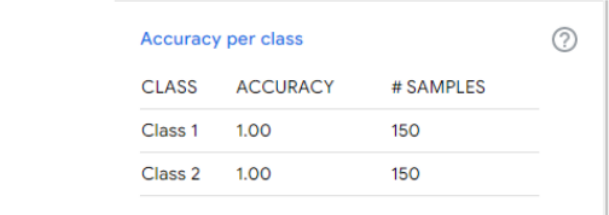


Fig. 16 Accuracy per class, final values

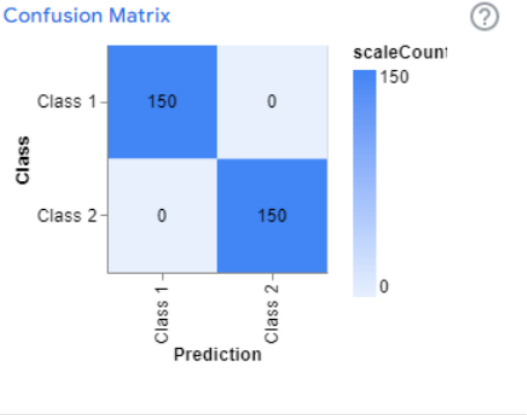


Fig. 14 Confusion Matrix, final values

Accuracy per epoch

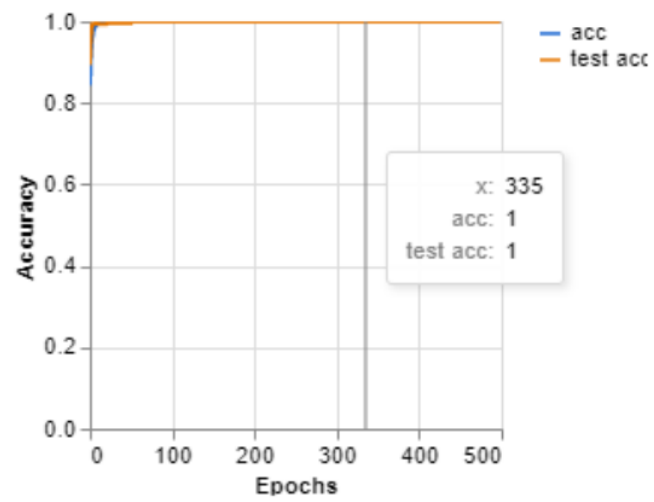


Fig. 17 Accuracy per epoch, final values

Loss per epoch

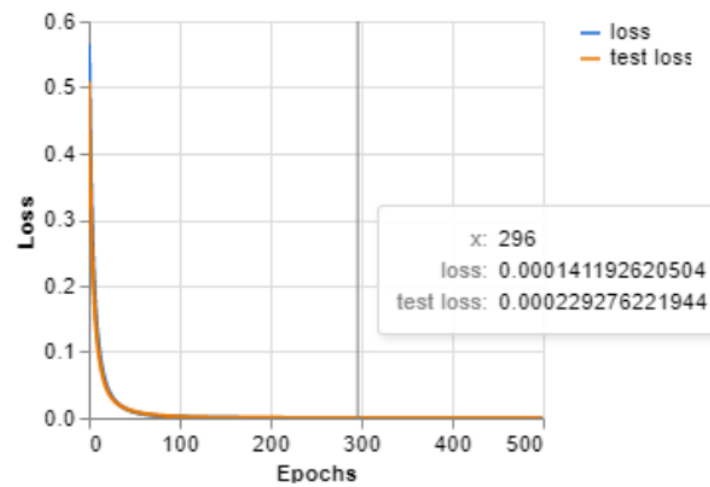


Fig. 18 Loss per epoch, final values

To summarise the process, the values of the parameters and results presented above are reflected in **Table 2**. The values presented in the *confusion matrixes* are presented in **Table 3**.

	Parameters			Results					
	Epochs	Batch Size	Learning Rate	Accuracy per class		Accuracy per epoch		Loss per epoch	
				Class 1	Class 2	Acc	Test acc	Loss	Test loss
Set 1	50	16	1×10^{-3}	0.00	1.00	0.5	0.5	4.605	4.605
Set 2	500	32	1×10^{-17}	1.00	0.93	0.974	0.956	0.320	0.337
Set 3	500	16	1×10^{-5}	1.00	1.00	1	1	1.4×10^{-4}	2.2×10^{-4}

Table 2 Comparison of model training parameters and results

		Prediction							
		Set 1		Set 2		Set 3			
		Cls 1	Cls 2	Cls 1	Cls 2	Cls 1	Cls 2		
Class	Set 1	Cls 1	0	150					
		Cls 2	0	150					
	Set 2	Cls 1			150	0			
		Cls 2			11	139			
	Set 3	Cls 1					150	0	
		Cls 2					0	150	

Table 3 Comparison of Confusion matrixes from model training.

In conclusion, the latest set of parameters have been used as they trained the most accurate model. The accuracy and confidentiality are reflected mainly in the loss (**Fig. 18**) and accuracy (**Fig. 17**) per epoch graphs; these pack the most useful feedbacked information. Although, the loss graphs have been the main point of reference when fine tuning the model. This is due to the sensitivity of feedbacked values and values with 15 decimal places; even the smallest changes can be noticed.

The parameters could be further adjusted to produce a more accurate and stable model; although, this is not required for this system.

Python Code.

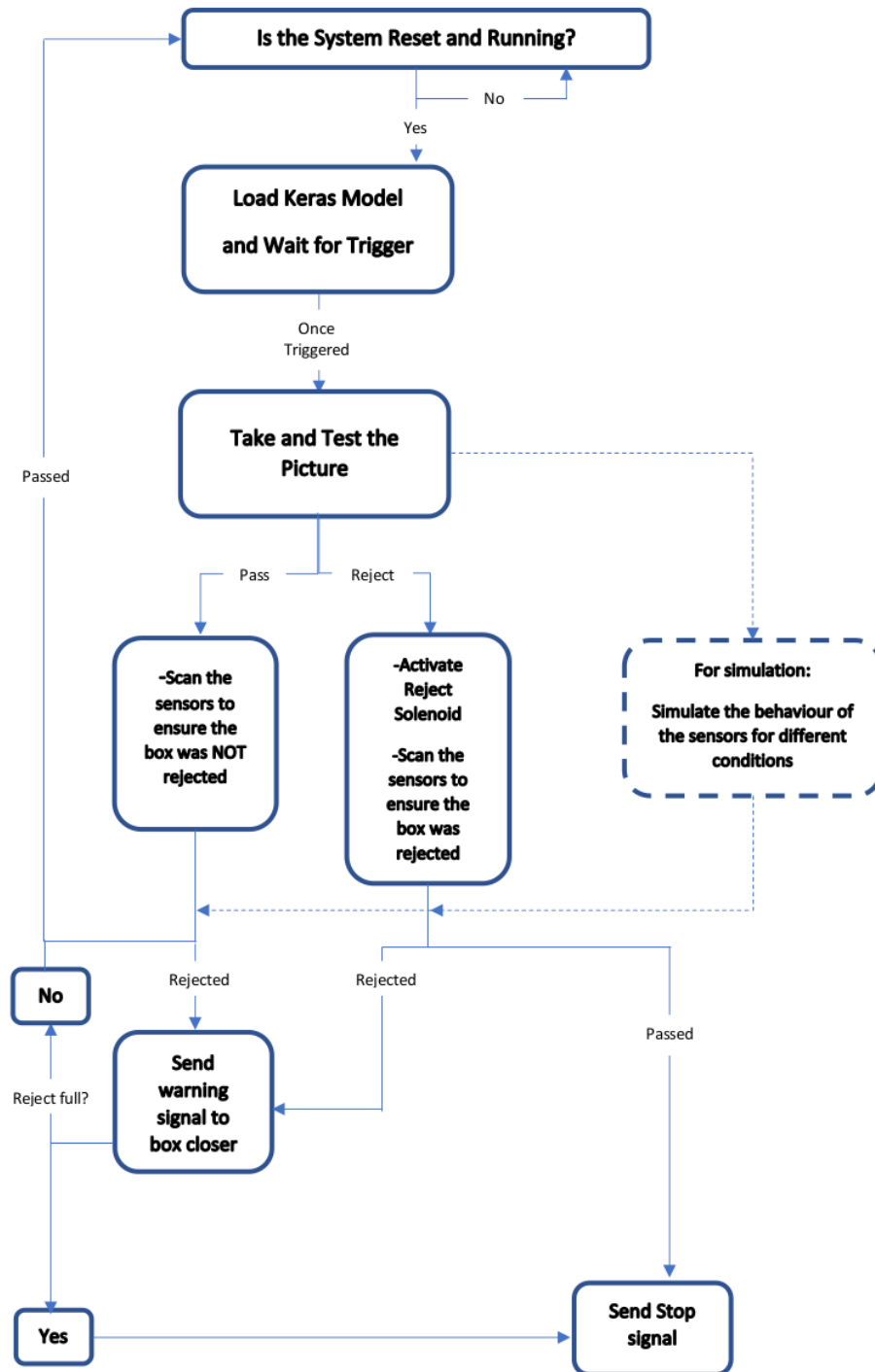


Fig. 19 Mind map for code development

The python code is what will load and feed the keras model and will also do all the decision making based on the return from the keras model.

The model started with a visual block diagram map of what the program is expected to do under certain conditions as it is shown in *Fig. 19*.

All Python code was written in accordance with **PEP** (Python Enhancement Proposals). These include:

- Indentation of 4 spaces. Spaces are preferred over tab but either can be used. It is disallowed to mix tabs and spaces, the use of either must be consistent. (Rossum, Warsaw, & Coghlan, 2013)
- All lines should be limited to 79 characters. (Rossum, Warsaw, & Coghlan, 2013)
- Use two blank lines around `def` and `class`. Also, use blank lines to indicate logical sections within the code. (Rossum, Warsaw, & Coghlan, 2013)
- For strings either single-quote and double-quote characters are the same in python and either can be used. It must be consistent throughout the code. For triple-quoted strings (ex. Appendix B.; `Defs.py`; line 42-44) use double-quote characters to be consistent with PEP 257. (Goodger & Rossum, 2001)
- Avoid using excessive spaces and do not use more than 1 space around operators. The space before and after an operator must be consistent. (Rossum, Warsaw, & Coghlan, 2013)
- Only ASCII characters can be used within Python code. Exceptions are; name of authors, test cases testing non-ASCII features. (Löwis, 2007)

Code Design

The designing process was split into few different sections: Decision-making code (together with simulation code), image processing and keras implementation code and diagnostic code.

Decision-making Code.

First, the decision-making part was designed which would scan the sensors, send the box to reject Spelling shute and stop the machine if a box went the wrong way or something is blocked. This was more complex than initially predicted since Python code runs line-by-line style and it does not scan the whole code like for example, PLC code does (Karani, 2020). This was ultimately overcome by splitting the code into 3 `.py` files: The main code (`CSS.py`), functions (`defs.py`) and IO Lists (`classes.py`). An additional `B.py` file was created for simulation of sensor behaviours. The code within all the files can be found in **Appendix B. Code and Codes modules**. All the code external to the main code is imported

You could have used a polling or interrupt method to avoid hanging

as a module using the import 'filename'. This allows for more than one flow of execution, running the main and the called code side-by-side, which eases the flow of the execution making it faster. For example, in defs.py there is a function RejectSolenoid() which activates the reject solenoid for 1.5 seconds (Appendix B. Code and Codes modules Defs.py; line 169-172). Once called, this runs alongside the main code not affecting the flow of execution within the main code; if the sleep(1.5) was in the main code the execution flow would wait, doing nothing for 1.5 seconds.

There was an attempt to overcome the issue in a way where it would not be needed to split the code across different .py files using Threading and Multiprocessing.

As Jim Anderson mentioned in his article *An Intro to Threading in Python*: *"A thread is a separate flow of execution. This means that your program will have two things happening at once. But for most Python 3 implementations the different threads do not actually execute at the same time: they merely appear to."* (Anderson, n.d.) This means that the execution jumps between codes when threading. Multiprocessing uses the same concept but has more tools that help organise and control the individual threads. The most important rule of threading and multiprocessing is that it can be started only once. Most of the possible solutions did not work or were a lot more complex compared to the used solution of external modules.

In parallel to that, the B.py file was created that enabled the testing of the decision-making part of the code. It is called as a thread from PassScan submodule as a thread, to ensure that it runs smoothly, simulating the behaviour of the inputs for the PassScan and RejectScan submodules to process.

The modules PassScan and RejectScan scan the passed and rejected sensors respectively, counting how long these are on for, with a timeout at 3 seconds. This data is then read by either PassDecisionMaker or RejectDecisionMaker based on the result from the Keras model. These modules monitor that the analysed box goes exactly where it was supposed to go, that there are no blockages in the box closer and that the reject chute is not full. If the box does not go where it is intended to go, or the module detects a blockage or full reject chute, it is instructed to send a stop signal to the box closer which will stop the rest of the line.

The main code will call either PassDecisionMaker if the box passed or RejectDecisionMaker if the box failed the keras check. It will always call both PassScan and RejectScan modules, regardless of Keras Model output.

Image processing and Keras model implementation.

The captured image, before it is processed by the Keras Model, it must be resized, adjusted and converted to a numpy array. This is done in the main code between lines 71 and 104. In line 107, the

Spelling

What is a numpy array?

prediction value is assigned to a variable and has a form of an array of one column and two rows. The first row holds the certainty of the fed in image being associated with the first class (Pass), and the second row with the second class (Reject). The percentage value is represented on a scale from 0 to 1.

The chosen library to import and process the image is Open CV to satisfy company's standards; despite that Kevin Luk, in his experiment with different libraries shows that Pillow library is more efficient:

+-----+-----+-----+		
Library	Function	Time
+-----+-----+-----+		
matplotlib	plt.imread()	0.02254
OpenCV	cv2.imread()	0.01096
Pillow	Image.Open()	0.00166
Skimage	io.imread()	0.01463
+-----+-----+-----+ (Luk, 2019)		

He also mentions that the Open CV Library, as default, imports images as BGR instead of RGB which can be solved with a colour channel conversion line: `cv2.cvtColor(img, cv2.COLOR_BGR2RGB)`. (Luk, 2019). This is used in main code in line 89.

Diagnostic and feedback code

The diagnostic code Spelling consist of a series of `print()` commands Spelling withing the code. These are not operator focused. They are a tool that helps debugging the code if any bugs were to occur. There will be two options to access the feedback terminal: through connecting all peripherals to the Pi (monitor, keyboard) or by using VNC from another computer that is connected via ethernet connection (preferably laptop). The company has not specified what method they require nor if diagnostic access is required; therefore, this was not set as one of the system specifications. Although, the OS installed on the RaspBerry Pi has VNC enabled for implementation; the VNC option is protected by password defined by the Author.

The diagnostic's feedback also displays the pass/error/reject ratios using `+=1` and division. [What does this mean?](#)

4. Testing and Results Spelling

Due to Production schedule and restrains from Production management the system's implementation has been postponed until at least June/July of 2022. Therefore, tests number 7, 8 and 9 from **Table 4** cannot be performed until the physical implementation phase. Additionally, test number 6 cannot be performed as the company decided to not implement the system into IIOT.

Test #	Specification	Testing method	Expected Result
1	Misplaced product recognition	Running the line and simulating misplacement conditions in varying severity by intentionally miss aligning the boards in the boxes before they exit the SM.	Reject of the misplaced products regardless of the severity of misplacement.
2	Compatibility with different models.	Each product must be run for at least 10k part run, while performing the actions from test #1	Same result as for test #1 for every model.
3	Cooldown time <3s	Monitoring the scan time (between scan and decision) using the serial monitor while performing test #1. The software will be programmed to feedback the time on the serial monitor after every scan. Record all timings.	Time between scans is less than 3 seconds during normal running conditions. Scanning time is smaller at abnormal running speeds; otherwise, system cutting out and alarming.
4	Jam condition detection	Simulate an abnormal machine stop by stopping the Box feed in ABC using the engineering mode. Followed by an E-stop conditions by pressing any of the available e-stops on ABC. Record the behaviour of the setup.	The system detects the jam up condition, stops and alarms.
5	Misfire condition detection	Simulate a controlled actuator misfire by switching off the air to the pneumatic actuator.	The system detects a misfire condition, instructs to stop ABC and alarms.
6	IIOT compatibility	Involve the IIOT department for IIOT connectivity testing. Request them to monitor the data stream to the main servers and ensure that the fail/pass data is there as well as the failed photos.	IIOT department confirming successful communication and data transfer between the system and IIOT server.
7	Minimal machine modification for installation	Take notes and pictures of the machine prior to installation of the system and compare that with the machine after the installation. Record all modifications to the machine as well as everything added to the machine (the system, brackets, wiring, piping...)	Minimal machine modifications after installation.
8	No interference with existing safety	Full test of all safety systems of the machine prior and following the installation of the system. Ensure to fully check all interlocks, E-stops, and all guarding for malfunctions and physical damage.	No difference in safety system functionality before and after system installation.
9	Cat 4 safety compliant	Risk assess the system using the company's template. Involve the H&S department and request for a safety evaluation. If any E-stops, interlocks or guarding has been added it must be fully tested and assessed to ensure it is compliant with the relevant H&S regulations.	The system is fully compliant with Company's safety regulations and all safety systems are fully operational, if present.

Table 4 Testing Table

You could have removed these from the table and explained the difference between the new FDS and the old one at the beginning of the report in more detail.

The rest of the tests from **Table 4** can be performed by simulating the system for which the testing method will require minor modifications.

Test #1 & 2

The product cannot be physically run. Therefore, the during simulation a set of product photos will be fed into the code, bypassing the camera enable and image capture from camera lines. The sets of pictures used for simulation incorporate both models and various severities of part misplacement.

Test #3

The timing incorporated will count the time between trigger and box passed or rejected. The 3 second rule will be flexible in this case ($\pm 2.5s$) to account for the time taken for manual input into terminal of the simulated behaviour.

Test #4 & 5

Jam and blockage detection will be tested through IO manipulation during simulation (see: **Appendix B. Code and Codes modules/B.py**)

The updated testing table is shown in **Table 5**.

Testing table of System Simulation				
Test #	Specification	Testing method	Expected Result	Actual Results
1	Misplaced product recognition	Pass and Reject pictures fed into the keras model from local HHD using code. Camera code must be commented out.	Reject of the misplaced products regardless of the severity of misplacement.	For pass boxes: - Pass prediction >0.8 - Reject prediction < 0.2 For reject boxes: - Pass prediction < 0.2 - reject prediction > 0.8
2	Compatibility with different models.	The photos used must be different from the ones used for model training	Same result as for test #1 for every model.	For pass boxes: - Pass prediction >0.8 - Reject prediction < 0.2 For reject boxes: - Pass prediction < 0.2 - reject prediction > 0.8
3	Cooldown time <3s	Monitor the test times during simulations. This will be displayed after each test in the terminal	The test times should be below 5s (except jam condition due to additional 3s IO scan time)	Standard running cycle time < 4s With exception for jammed conditions where cycle time < 7s
4	Jam condition detection	Follow the instructions on the simulation terminal, entering the correct behaviour number for the test	IO scan times out after 3s alerting of the jam and stopping the <i>box closer</i> .	In both cases (pass and reject) the PassScan module was activated and timed out at 3s; stopping the <i>box closer</i> .
5	Misfire condition detection		IO scan detects a trigger from the wrong sensor and stops the <i>box closer</i>	PassScan module activated, detecting a box going past; stopping the <i>box closer</i> .

Table 5 Testing table adjusted for simulation with simulation test results.

Below are is the evidence proving the results presented in **Table 5**.

Test #1 & 2

The first and second test have been performed as described in **Table 5**. An example of pictures fed into the keras model are shown in **Fig. 21** and **Fig. 20**.



Fig. 20 Example of a pass box.

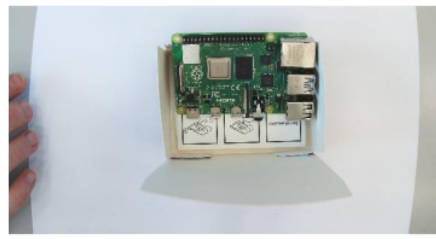


Fig. 21 Example of a reject box.

The output from the terminal when loading a reject box:

```
waiting for trigger
Enter 0 to load a pass board image
Enter 1 to load a reject board image

Pass prediction is: 0.00019944199
Reject prediction is: 0.9998006
```

The output from the terminal when loading a pass box:

```
waiting for trigger
Enter 0 to load a pass board image
Enter 1 to load a reject board image

Enter selection to trigger test: 0
Pass prediction is: 0.997331
Reject prediction is: 0.0026690152
```

The full copy of the terminal's output can be found in *Appendix C. Simulation Results (output from terminal)*.

Test #3

The third test was conducted during the whole length of testing. Below are some of the timings when running the simulation under normal conditions:

```
test completed in: 4.19
test completed in: 2.29
test completed in: 2.24
test completed in: 3.63
test completed in: 2.62
```

The following timings were taken when testing jammed conditions:

```
test completed in: 6.5
test completed in: 5.42
test completed in: 6.69
```

Test #4

As part of simulation, the behaviour selection has been added to the code as described in the software design section. Below are the terminal outputs for the jam condition for a pass and a rejected box.

Reject box:

```
Box failed test
behaviours:
```

1. Box rejected
2. Box rejected and shute full after rejection
3. box not rejected
4. box closer not running
5. box closer jammed

```
enter behaviour's number: 5
PassScan finished at: 30
```

```
Closer Error: Input to closer jammed and closer not stopped!!
Closer stopped!
```

Pass box:

- ```
Box passed test
behaviours:
1. Box rejected
2. Box rejected and shute full after rejection
3. box not rejected
4. box closer not running
5. box closer jammed
```

*It is not clear to see that this text provides any real evidence*

```
enter behaviour's number: 5
PassScan finished at: 30
```

```
Closer Error: Input to closer jammed and closer not stopped!!
Closer stopped!
```

As it is seen by the diagnostic feedback, in both cases is the PassScan module that gets activated and times out at 30 (3s, counts of 0.1s). This intentional; it is coded in such way that the pass sensor takes priority over reject sensor. This is to ensure that the first space after the reject area going into pass conveyor is empty (see **Fig. 1** area and sensor positions); this avoids build-ups in the reject area and ensures that the boxes being processed in the reject area have no obstructions.

#### Test #5

The fifth test is performed in the same way as test #4, selecting a different behaviour for a box that tested as reject.

Here is the output from the terminal:

- ```
Box failed test
behaviours:
1. Box rejected
2. Box rejected and shute full after rejection
3. box not rejected
4. box closer not running
5. box closer jammed
```

```
enter behaviour's number: 3
PassScan finished at: 7
```

```
Actuator misfire!
Closer Stopped.
```

As expected, the PassScan module was activated, causing a stop call to be sent to the *box closer*.

5. Evaluation and the Wider Issues

The project, based on the outcomes of the simulation, works as intended on paper. There is no evidence on how the system will perform in the physical realm. Also, the FDS have been modified as the company ^{Spelling}resigned from implementing the system as an IIOT device. Although that is upgradable in the future.

The project's cost is a lot more than initially predicted by Adam Bedner in his report *1. Project Definition, Research and Quality Planning* at approximately £200. That is because the additional materials and man's hours were not considered. Although it is still less than any other solutions presented. (Bedner, 2021)

The main issue with using a micro processor such as Raspberry Pi is the lack of support of industry standard languages used in PLCs such as Ladder or SCL (Siemens PLCs). Also, there is no direct compatibility between the micro processor and industry standard equipment such as drives, PLCs, safety devices. All of this can be programmed in using languages such as C++ or Python as in this case. But it requires knowledge and can be time consuming; the languages are also rare in industry and knowledge of such language in industrial engineers can be limited making debugging costly and time consuming.

Another wider issue of using opensource devices, software and code is CyberSecurity. Usually access to PLC devices and software is limited due to the high prices; it is difficult and often not profitable for the public to attempt to break into or develop viruses for well protected devices such as PLCs. Raspberry Pis and their software is cheap and widely available, making it a lot easier to attempt developing gate breaking code or viruses. From industry point of view that is a major threat that has not been covered during this project (although it should have been), mainly with the intention of IIOT incorporation. Open-source devices would usually require extra cyber safety steps to be made, or for the system to be limited to an IntraNet, which is not considered IOT or Industry 4.0.

Also, micro processors and micro controllers such as Raspberry Pi usually work at voltages of 3V3 or 5V DC, where the industry standard control voltage is 24V DC. This issue requires signal voltage interface (such as the relay interface used in this project) and additional power supplies. For larger systems, this can become less power efficient, bulkier and more complex than a use of a PLC. It also creates additional possible failure points within a circuit, higher electro magnetic interferences due to additional power supplies and increases the maintenance cost (wear and tear of relays and other components).

This is an admission that you did not think this solution through carefully enough when selecting it.

9 In conclusion, use of microprocessors is a great, cheaper option for small projects for companies that
10 have experience in software engineering with microcontroller and/or microprocessor language
11 experience.

12

6. Conclusions and Recommendations

To conclude, the project is still missing its final implementation phase which is scheduled for June/July of 2022. This will require the installation, testing and commissioning of the system. All the tests carried out during simulation will have to be re-tested, together with the safety tests, machine modification tests that could not be carried out in simulation.

This section is disappointingly short

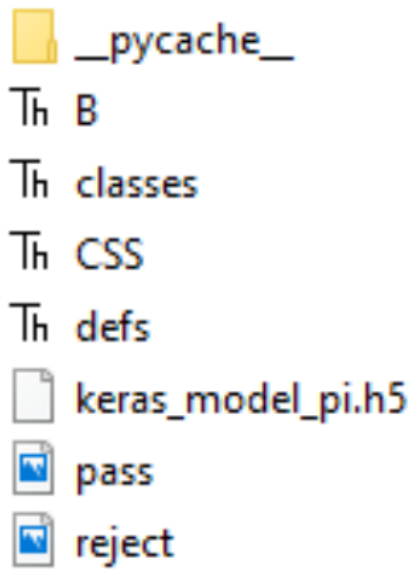
7. References

- Anderson, J. (n.d.). *An Intro to Threading in Python*. Retrieved from Real Python: <https://realpython.com/intro-to-python-threading/>
- Bedner, A. (2021). *1. Project Definition, Research and Quality Planning*. Llanelli: University of Wales, Trinity Saint David.
- COBUILD Advanced English Dictionary. (2022). *Collins Dictionary*. Retrieved from fail-safe: <https://www.collinsdictionary.com/dictionary/english/fail-safe>
- Goodger, D., & Rossum, G. v. (2001, Jun 13). *Python Enhancement Proposals*. Retrieved from PEP 257 - Docstring Conventions: <https://peps.python.org/pep-0257/>
- Google. (2022). Teachable Machine.
- Karani, D. (2020, Jan 9). *How does Python Work?* Retrieved from towardsdatascience.com: <https://towardsdatascience.com/how-does-python-work-6f21fd197888>
- Löwis, M. v. (2007, May 01). *Python Enhancement Proposals*. Retrieved from PEP 3131 - Supporting Non-ASCII Identifiers: <https://peps.python.org/pep-3131/>
- Luk, K. (2019, Mar 30). *towards data science*. Retrieved from What libraries can load image in Python and what are their difference?: <https://towardsdatascience.com/what-library-can-load-image-in-python-and-what-are-their-difference-d1628c6623ad>
- Omron. (n.d.). Compact Photoelectric Sensor with Built-in Amplifier E3Z.
- On Semiconductor. (2013, January). P2N2222A/D. Semiconductor Components Industries.
- Phoenix Contact. (2019, 10 05). Relay Module - RIF-0-RPT-24DC/21 - 2903370. Phoenix Contact.
- Rossum, G. v., Warsaw, B., & Coghlan, N. (2013, Aug 01). *Python Enhancement Proposals*. Retrieved from PEP 8 - Style Guide for Python Code: <https://peps.python.org/pep-0008/>
- Siemens. (2020, 10 10). 5SL4501-6.
- Siemens. (2020, 10 10). 5SL4503-6.
- ThePiHut. (2022). *ThePiHut*. Retrieved from 5V 3A Dual USB Step Down Buck Converter Module: <https://thepihut.com/collections/raspberry-pi-power-accessories/products/5v-3a-dual-usb-step-down-buck-converter-module>

You could have used more sources in your design

Appendix A. Folder Structure

The folder structure that contains the main code and all the submodules is shown below:



Appendix B. Code and Codes modules

For proper operation and easier navigation, the code has been split in three codes. The main code CSS.py (CameraSystemSimulation), defined functions defs.py and IO and memory bits classes.py. Additionally, for simulation purposes B.py has been created which simulates the sensor inputs to the code.

Note: Comments begin with #

Classes.py

```
13 class Sensor:
14     CloserStop = False          #Input, detects if the box closer is running. erector stopped
15 if true
16     RejectFull = False         #Input, detects if the box has been rejected properly. serves
17 dual purpose, if True for >3s Then rejection shute full.
18     RejectPresent = False      #Input, detects if there is a box present in the rejection
19 section.
20     Trigger = False            #Input, detects if there is a box in the test area.
21     Passed = False             #Input, detects if the box have not been rejected (went to box
22 closer). serves dual purpose, if True for >3s Then box closer is jammed.
23
24
25 class Actuator:
26     Reject = False             #Output, activates the reection solenoid
27     CloserStop = False         #Output, Sends an immediate stop signal to the Box closer
28     CloserWarning = False      #Output, sends a warning signal to closer
29
30
31 class MBit:
32     TestRun = False            #Sets the test to run.Behaviour = int(0)          #Selection
33 of simulated machine behaviour
34     Pass = int(0)               #Pass count
35     PassScan = int(0)           #Pass sensor scan timer
36     Reject = int(0)             #Reject count
37     RejectScan = int(0)         #Reject sensor scan timer
38     TotalThrough = int(0)       #Total boards tested counter
39     PassRatio = float(0)        #Ratio of passes
40     RejectRatio = float(0)      #Ratio of rejects
41     ErrorRatio = float(0)       #Ratio of Errors
42     SystemRunning = False       #System reset and running
43     BCall = False               #Behaviour call
44     ErrorCount = int(0)         #Error count
```

Defs.py

```
1  from time import sleep
2  import classes as c
3  import B
4  import threading
5
6
7  def PassScan ():
8      c.MBit.BCall = True
9      Behaviour = threading.Thread(target=B.Behavior_Selection)
10     Behaviour.start()
11     print("PassScan Started", c.Sensor.Passed, c.MBit.BCall,)
12     sleep(0.1)
13     while c.Sensor.Passed == True:
14         print("in pass while loop", c.MBit.PassScan)
15         sleep(0.1)
16         c.MBit.PassScan += 1
17         if c.MBit.PassScan == 30 or c.Sensor.Passed == False:
18             print("PassScann finished at:", c.MBit.PassScan)
19             c.MBit.BCall = False
20             break
21
22
23  def RejectScan ():
24     print("RejectScan Started", c.Sensor.RejectFull)
25     while c.Sensor.RejectFull == True:
26         print("in reject while loop", c.MBit.RejectScan)
27         sleep(0.1)
28         c.MBit.RejectScan += 1
29         if c.MBit.RejectScan == 30 or c.Sensor.RejectFull == False:
30             print("RejectScan finished at:", c.MBit.RejectScan)
31             break
32
33
34  def PassDecisionMaker ():
35     if c.Sensor.CloserStop == True:
36         print("Closer Stopped!")
37         c.MBit.PassScan = 0
38         c.MBit.ErrorCount += 1
39     elif c.MBit.BError == True:
40         sleep(0.25)
41         print(
42             """
43             Program stopped due to Behavior_Selection Error!
44             """
45         )
46         c.MBit.SystemRunning = False
47     elif c.MBit.PassScan == 0:
48         c.MBit.ErrorCount += 1
49         #c.Actuator.CloserStop = True
50         print(
51             """
52             Good box not in closer!!
53             Good Box Rejected!!
54             """
```

```

55     )
56     elif 5 < c.MBit.PassScan < 25:
57         print(
58             """
59         Box passed successfully
60             """
61         )
62         c.MBit.Pass += 1
63         c.MBit.PassScan = 0
64     elif c.MBit.PassScan >= 25:
65         if c.Sensor.CloserStop == False:
66             c.Actuator.CloserStop = True
67             print(
68                 """
69         Closer Error: Input to closer jammed and closer not stopped!!
70         Closer stopped!
71             """
72         )
73         c.MBit.ErrorCount += 1
74         c.MBit.PassScan = 0
75
76     if 0 < c.MBit.RejectScan < 25:
77         c.Actuator.CloserWarning = True
78         c.MBit.RejectScan = 0
79         print(
80             """
81         Good Box Rejected!!
82         Investigate and place on closer input if ok
83             """
84         )
85         c.MBit.ErrorCount += 1
86     elif c.MBit.RejectScan >= 25:
87         c.Actuator.CloserStop = True
88         c.MBit.RejectScan = 0
89         print(
90             """
91         Reject shute full!!
92         Empty shute before restarting the machine
93             """
94         )
95         c.MBit.ErrorCount += 1
96
97
98     def RejectDecisionMaker ():
99         if c.Sensor.RejectPresent == True:
100             c.Actuator.CloserWarning = True
101         if c.Sensor.CloserStop == True:
102             print(
103                 """
104             Closer Stopped!
105                 """
106             )
107             c.MBit.PassScan = 0
108             c.MBit.ErrorCount += 1
109         elif c.MBit.BError == True:

```

```

110         sleep(0.25)
111         print(
112             """
113             Program stopped due to Behavior_Selection Error!
114             """
115         )
116         c.MBit.SystemRunning = False
117     elif 5 < c.MBit.PassScan < 25:
118         c.Actuator.CloserStop = True
119         c.MBit.PassScan = 0
120         print(
121             """
122             Actuator misfire!
123             Closer Stopped.
124             Call Technical support
125             """
126         )
127         c.MBit.ErrorCount += 1
128         c.MBit.PassScan = 0
129     elif c.MBit.PassScan >= 25:
130         if c.Sensor.CloserStop == False:
131             c.Actuator.CloserStop = True
132             print(
133                 """
134                 Closer Error: Input to closer jammed and closer not stopped!!
135                 Closer stopped!
136                 """
137             )
138             c.MBit.PassScan = 0
139             c.MBit.ErrorCount += 1
140
141     if 0 < c.MBit.RejectScan < 25:
142         c.MBit.RejectScan = 0
143         print(
144             """
145             Box Rejected Succssfully
146             """
147         )
148         c.MBit.Reject += 1
149     elif c.MBit.RejectScan >= 25:
150         c.Actuator.CloserStop = True
151         c.MBit.RejectScan = 0
152         print(
153             """
154             Reject shute full!!
155             Empty shute before restarting the machine
156             """
157         )
158         c.MBit.ErrorCount += 1
159
160
161 def PassRejectRatio ():
162     c.MBit.TotalThrough = c.MBit.Pass + c.MBit.Reject + c.MBit.ErrorCount
163     while c.MBit.TotalThrough > 0:
164         c.MBit.PassRatio = (c.MBit.Pass / c.MBit.TotalThrough) * 100

```

```
165         c.MBit.RejectRatio = (c.MBit.Reject / c.MBit.TotalThrough) * 100
166         c.MBit.ErrorRatio = (c.MBit.ErrorCount / c.MBit.TotalThrough) * 100
167         break
168
169
170     def RejectSolenoid():
171         c.Actuator.Reject = True
172         sleep (1.5)
173         c.Actuator.Reject = False
```

B.py

```
1  from time import sleep
2  import classes as c
3
4  def Behavior_Selection ():
5      while c.MBit.BCall == True:
6          print("Behaviour started", c.MBit.Behaviour)
7          if c.MBit.Behaviour == 1:
8              c.Sensor.CloserStop = False
9              c.Sensor.RejectFull = True
10             sleep(1)
11             c.Sensor.RejectFull = False
12             c.Sensor.RejectPresent = True
13             c.Sensor.Passed = False
14             c.MBit.BCall = False
15         elif c.MBit.Behaviour == 2:
16             c.Sensor.CloserStop = False
17             c.Sensor.RejectFull = True
18             c.Sensor.RejectPresent = True
19             c.Sensor.Passed = False
20             c.MBit.BCall = False
21         elif c.MBit.Behaviour == 3:
22             c.Sensor.CloserStop = False
23             c.Sensor.RejectFull = False
24             c.Sensor.RejectPresent = False
25             c.Sensor.Passed = True
26             sleep(1)
27             c.Sensor.Passed = False
28             c.MBit.BCall = False
29         elif c.MBit.Behaviour == 4:
30             c.Sensor.CloserStop = True
31             c.Sensor.RejectFull = False
32             c.Sensor.RejectPresent = False
33             c.Sensor.Passed = False
34             c.MBit.BCall = False
35         elif c.MBit.Behaviour == 5:
36             c.Sensor.CloserStop = False
37             c.Sensor.RejectFull = False
38             c.Sensor.RejectPresent = False
39             c.Sensor.Passed = True
40             c.MBit.BCall = False
41         else:
42             print("incorrect input")
```

CSS.py

```
1  ###Camera vision system for Raspberry Pi misplacement detection.
2  ###Author: Adam Bedner
3  ###Acknowledgements: Jake Stevens, niconielsen32
4  ###V1.0
5  ###02.2022
6  ###THIS IS A SIMULATION VERSION!!
7  ###DO NOT ATTEMPT TO RUN OUTSIDE OF SIMUJATION ENVIRONMENT
8  ###
9  ###For full version contact the author
10 ###
11 ###NOTES:
12 ###some lines of code have been commented out for simulation purposes
13 ###
14 ###
15
16
17 import tensorflow.keras
18 import cv2
19 import numpy as np
20 import time
21 import defs
22 import classes as c
23 from time import sleep
24 # from picamera import PiCamera
25
26 np.set_printoptions(suppress=True)
27
28 # first, reset the system (that will be done with a pushbutton or signal from PLC once
29 implemented)
30 while True:
31     print(
32         """
33         System not running
34         to reset enter 1
35         """
36     )
37     SR = int(input("Enter 1 now to reset: "))
38
39     if SR == 1:
40         c.MBit.SystemRunning = True
41
42     # Load the keras model
43     model = tensorflow.keras.models.load_model('keras_model_pi.h5')
44     if __name__ == '__main__':
45         while c.MBit.SystemRunning == True:
46             print(
47                 """
48                 waiting for trigger
49                 Enter 0 to load a pass board image
50                 Enter 1 to load a reject board image
51                 """
52             )
53             T = int(input("Enter selection to trigger test: "))
54
```

```

55         if 0 <= T <= 1:
56             c.Sensor.Trigger = True
57
58     # trigger the test to run (this will be done with a Trigger sensor once implemented)
59
60     if c.Sensor.Trigger == True:
61         c.MBit.TestRun = True
62
63     while c.MBit.TestRun == True:
64
65     # Create the array of the right shape to feed into the keras model
66     # The 'length' or number of images you can put into the array is
67     # determined by the first position in the shape tuple, in this case 1.
68
69         print("Test Running")
70         start = time.time()
71         print("Timer Running")
72         data = np.ndarray(shape=(1, 224, 224, 3), dtype=np.float32)
73
74         size = (224, 224)
75
76         #cap = cv2.VideoCapture(0)
77         print("Entering Keras While")
78
79         while True:
80             #while cap.isOpened():
81
82                 #ret, img = cap.read()
83
84                 print("In Keras While")
85                 if T == 0:
86                     img = cv2.imread('pass.jpg')
87                 elif T == 1:
88                     img = cv2.imread('reject.jpg')
89
90                 imgRGB = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
91
92                 height, width, channels = img.shape
93
94                 scale_value = width / height
95
96                 img_resized = cv2.resize(imgRGB, size, fx=scale_value, fy=1,
97 interpolation=cv2.INTER_NEAREST)
98
99     # Turn the image into a numpy array
100         img_array = np.asarray(img_resized)
101
102     # Normalize the image
103         normalized_img_array = (img_array.astype(np.float32) / 127.0) - 1
104
105     # Load the image into the array
106         data[0] = normalized_img_array
107
108     # run the inference
109         prediction = model.predict(data)

```

```

110
111 # Prediction is an array of a column and two lines as:
112 # prediction = np.array([Pass prediction], [Reject prediction])
113 # therefore, each value is read by it's coordinates Pass = [0, 0] and Reject = [0, 1]
114
115         print("Pass prediction is:", prediction[0, 0])
116         print("Reject prediction is:", prediction[0, 1])
117         break
118
119 # Test accuracy is set to 0.8 but can be increased or decreased.
120 # It is important to know that the keras model does not return a 0 or 1
121 # it returns a float value in range between 0 and 1
122
123 # IMPORTANT!!!! TestAccuracy CAN NOT BE LESS THAN 0.6!!!
124     TestAccuracy = 0.8
125
126     if prediction[0, 0] >= TestAccuracy:
127         sleep(0.1)
128         c.Actuator.Reject = False
129
130 # The Bheaviour Selection code is solely part of simulation code
131 # it will be removed in the final version
132
133         print(
134             """
135             Box passed test
136             behaviours:
137             1. Box rejected
138             2. Box rejected and shute full after rejection
139             3. box not rejected
140             4. box closer not running
141             5. box closer jammed
142
143             """
144         )
145         c.MBit.Behaviour = int(input("enter behaviour's number: "))
146
147         defs.PassScan()
148         defs.RejectScan()
149
150         #sleep(1.5)
151
152         defs.PassDecisionMaker()
153
154         elif prediction[0, 1] >= TestAccuracy:
155             sleep(0.1)
156             defs.RejectSolenoid()
157             print(
158                 """
159                 Box failed test
160                 bhaviours:
161                 1. Box rejected
162                 2. Box rejected and shute full after rejection
163                 3. box not rejected
164                 4. box closer not running

```

```

165         5. box closer jammed
166         """
167         )
168         c.MBit.Behaviour = int(input("enter behaviour's number: "))
169
170         defs.PassScan()
171         defs.RejectScan()
172
173         #sleep(1.5)
174
175         defs.RejectDecisionMaker()
176
177         defs.PassRejectRatio()
178         print("Total passes :", c.MBit.Pass)
179         print("Total rejects:", c.MBit.Reject)
180         print("Error Count :", c.MBit.ErrorCount)
181         print("Pass ratio :", round(c.MBit.PassRatio,2))
182         print("Reject ratio :", round(c.MBit.RejectRatio,2))
183         print("Error Ratio :", round(c.MBit.ErrorRatio,2))
184         end = time.time()
185         totalTime = end - start
186         print("test completed in:", round(totalTime,2))
187         test_running = False
188         T = 0
189         break
190
191     cv2.destroyAllWindows()
192     #cap.release()

```

Appendix C. Simulation Results (output from terminal)

```

1  System not running
2  to reset enter 1
3
4  Enter 1 now to reset: 1
5  WARNING:tensorflow:No training configuration found in the save file, so the model
6  was *not* compiled. Compile it manually.
7
8      waiting for trigger
9      Enter 0 to load a pass board image
10     Enter 1 to load a reject board image
11
12  Enter selection to trigger test: 1
13  Test Running
14  Timer Running
15  Entering Keras While
16  In Keras While
17  Pass prediction is: 0.00019944199
18  Reject prediction is: 0.9998006
19
20     Box failed test
21     behaviours:
22     1. Box rejected
23     2. Box rejected and shute full after rejection

```

```

24         3. box not rejected
25         4. box closer not running
26         5. box closer jammed
27
28     enter behaviour's number: 1
29     Behaviour startedPassScan Started 1False
30     True
31     RejectScan Started True
32     in reject while loop 0
33     in reject while loop 1
34     in reject while loop 2
35     in reject while loop 3
36     in reject while loop 4
37     in reject while loop 5
38     in reject while loop 6
39     RejectScan finished at: 7
40
41     Box Rejected Succssfully
42
43     Total passes : 0
44     Total rejects: 1
45     Error Count : 0
46     Pass ratio : 0.0
47     Reject ratio : 100.0
48     Error Ratio : 0.0
49     test completed in: 4.19
50
51     waiting for trigger
52     Enter 0 to load a pass board image
53     Enter 1 to load a reject board image
54
55     Enter selection to trigger test: 1
56     Test Running
57     Timer Running
58     Entering Keras While
59     In Keras While
60     Pass prediction is: 0.00019944199
61     Reject prediction is: 0.9998006
62
63     Box failed test
64     behaviours:
65     1. Box rejected
66     2. Box rejected and shute full after rejection
67     3. box not rejected
68     4. box closer not running
69     5. box closer jammed
70
71     enter behaviour's number: 2
72     PassScan StartedBehaviour started False2
73     True
74     RejectScan Started True
75     in reject while loop 0
76     in reject while loop 1
77     in reject while loop 2
78     in reject while loop 3
79     in reject while loop 4
80     in reject while loop 5

```

```

81   in reject while loop 6
82   in reject while loop 7
83   in reject while loop 8
84   in reject while loop 9
85   in reject while loop 10
86   in reject while loop 11
87   in reject while loop 12
88   in reject while loop 13
89   in reject while loop 14
90   in reject while loop 15
91   in reject while loop 16
92   in reject while loop 17
93   in reject while loop 18
94   in reject while loop 19
95   in reject while loop 20
96   in reject while loop 21
97   in reject while loop 22
98   in reject while loop 23
99   in reject while loop 24
100  in reject while loop 25
101  in reject while loop 26
102  in reject while loop 27
103  in reject while loop 28
104  in reject while loop 29
105  RejectScan finished at: 30
106
107      Reject shute full!!
108      Empty shute before restarting the machine
109
110
111  Total passes : 0
112  Total rejects: 1
113  Error Count  : 1
114  Pass ratio   : 0.0
115  Reject ratio : 50.0
116  Error Ratio  : 50.0
117  test completed in: 6.69
118
119      waiting for trigger
120      Enter 0 to load a pass board image
121      Enter 1 to load a reject board image
122
123  Enter selection to trigger test: 1
124  Test Running
125  Timer Running
126  Entering Keras While
127  In Keras While
128  Pass prediction is: 0.00019944199
129  Reject prediction is: 0.9998006
130
131      Box failed test
132      behaviours:
133      1. Box rejected
134      2. Box rejected and shute full after rejection
135      3. box not rejected
136      4. box closer not running
137      5. box closer jammed

```

```

138
139 enter behaviour's number: 3
140 PassScan StartedBehaviour started False3
141 True
142 in pass while loop 0
143 in pass while loop 1
144 in pass while loop 2
145 in pass while loop 3
146 in pass while loop 4
147 in pass while loop 5
148 in pass while loop 6
149 PassScann finished at: 7
150 RejectScan Started False
151
152     Actuator misfire!
153     Closer Stopped.
154
155 Total passes : 0
156 Total rejects: 1
157 Error Count : 2
158 Pass ratio : 0.0
159 Reject ratio : 33.33
160 Error Ratio : 66.67
161 test completed in: 3.63
162
163     waiting for trigger
164     Enter 0 to load a pass board image
165     Enter 1 to load a reject board image
166
167 Enter selection to trigger test: 1
168 Test Running
169 Timer Running
170 Entering Keras While
171 In Keras While
172 Pass prediction is: 0.00019944199
173 Reject prediction is: 0.9998006
174
175     Box failed test
176     behaviours:
177     1. Box rejected
178     2. Box rejected and shute full after rejection
179     3. box not rejected
180     4. box closer not running
181     5. box closer jammed
182
183 enter behaviour's number: 4
184 PassScan StartedBehaviour started False4
185 True
186 RejectScan Started False
187
188     Closer Stopped!
189
190 Total passes : 0
191 Total rejects: 1
192 Error Count : 3
193 Pass ratio : 0.0
194 Reject ratio : 25.0

```

```
195 Error Ratio : 75.0
196 test completed in: 2.62
197
198     waiting for trigger
199     Enter 0 to load a pass board image
200     Enter 1 to load a reject board image
201
202 Enter selection to trigger test: 1
203 Test Running
204 Timer Running
205 Entering Keras While
206 In Keras While
207 Pass prediction is: 0.00019944199
208 Reject prediction is: 0.9998006
209
210     Box failed test
211     behaviours:
212     1. Box rejected
213     2. Box rejected and shute full after rejection
214     3. box not rejected
215     4. box closer not running
216     5. box closer jammed
217
218 enter behaviour's number: 5
219 PassScan StartedBehaviour started False5
220 True
221 in pass while loop 0
222 in pass while loop 1
223 in pass while loop 2
224 in pass while loop 3
225 in pass while loop 4
226 in pass while loop 5
227 in pass while loop 6
228 in pass while loop 7
229 in pass while loop 8
230 in pass while loop 9
231 in pass while loop 10
232 in pass while loop 11
233 in pass while loop 12
234 in pass while loop 13
235 in pass while loop 14
236 in pass while loop 15
237 in pass while loop 16
238 in pass while loop 17
239 in pass while loop 18
240 in pass while loop 19
241 in pass while loop 20
242 in pass while loop 21
243 in pass while loop 22
244 in pass while loop 23
245 in pass while loop 24
246 in pass while loop 25
247 in pass while loop 26
248 in pass while loop 27
249 in pass while loop 28
250 in pass while loop 29
251 PassScann finished at: 30
```

```

252 RejectScan Started False
253
254     Closer Error: Input to closer jammed and closer not stopped!!
255     Closer stopped!
256
257 Total passes : 0
258 Total rejects: 1
259 Error Count  : 4
260 Pass ratio   : 0.0
261 Reject ratio : 20.0
262 Error Ratio  : 80.0
263 test completed in: 6.5
264
265     waiting for trigger
266     Enter 0 to load a pass board image
267     Enter 1 to load a reject board image
268
269 Enter selection to trigger test: 0
270 Test Running
271 Timer Running
272 Entering Keras While
273 In Keras While
274 Pass prediction is: 0.997331
275 Reject prediction is: 0.0026690152
276
277     Box passed test
278     behaviours:
279     1. Box rejected
280     2. Box rejected and shute full after rejection
281     3. box not rejected
282     4. box closer not running
283     5. box closer jammed
284
285
286 enter behaviour's number: 1
287 Behaviour startedPassScan False 1True
288 RejectScan Started True
289 in reject while loop 0
290 in reject while loop 1
291 in reject while loop 2
292 in reject while loop 3
293 in reject while loop 4
294 in reject while loop 5
295 in reject while loop 6
296 in reject while loop 7
297 in reject while loop 8
298 RejectScan finished at: 9
299
300     Good box not in closer!!
301     Good Box Rejected!!
302
303 Total passes : 1
304 Total rejects: 1
305 Error Count  : 4
306 Pass ratio   : 16.67
307 Reject ratio : 16.67
308 Error Ratio  : 66.67

```

```
309 test completed in: 2.29
310
311     waiting for trigger
312     Enter 0 to load a pass board image
313     Enter 1 to load a reject board image
314
315 Enter selection to trigger test: 0
316 Test Running
317 Timer Running
318 Entering Keras While
319 In Keras While
320 Pass prediction is: 0.997331
321 Reject prediction is: 0.0026690152
322
323     Box passed test
324     behaviours:
325     1. Box rejected
326     2. Box rejected and shute full after rejection
327     3. box not rejected
328     4. box closer not running
329     5. box closer jammed
330
331
332 enter behaviour's number: 2
333 Behaviour startedPassScan Started 2False
334     True
335 RejectScan Started True
336 in reject while loop 0
337 in reject while loop 1
338 in reject while loop 2
339 in reject while loop 3
340 in reject while loop 4
341 in reject while loop 5
342 in reject while loop 6
343 in reject while loop 7
344 in reject while loop 8
345 in reject while loop 9
346 in reject while loop 10
347 in reject while loop 11
348 in reject while loop 12
349 in reject while loop 13
350 in reject while loop 14
351 in reject while loop 15
352 in reject while loop 16
353 in reject while loop 17
354 in reject while loop 18
355 in reject while loop 19
356 in reject while loop 20
357 in reject while loop 21
358 in reject while loop 22
359 in reject while loop 23
360 in reject while loop 24
361 in reject while loop 25
362 in reject while loop 26
363 in reject while loop 27
364 in reject while loop 28
365 in reject while loop 29
```

```

366 RejectScan finished at: 30
367
368     Good box not in closer!!
369
370
371     Reject shute full!!
372     Empty shute before restarting the machine
373
374 Total passes : 1
375 Total rejects: 1
376 Error Count : 6
377 Pass ratio : 12.5
378 Reject ratio : 12.5
379 Error Ratio : 75.0
380 test completed in: 5.42
381
382     waiting for trigger
383     Enter 0 to load a pass board image
384     Enter 1 to load a reject board image
385
386 Enter selection to trigger test: 0
387 Test Running
388 Timer Running
389 Entering Keras While
390 In Keras While
391 Pass prediction is: 0.997331
392 Reject prediction is: 0.0026690152
393
394     Box passed test
395     behaviours:
396     1. Box rejected
397     2. Box rejected and shute full after rejection
398     3. box not rejected
399     4. box closer not running
400     5. box closer jammed
401
402
403 enter behaviour's number: 3
404 Behaviour startedPassScan Started 3False
405     True
406 in pass while loop 0
407 in pass while loop 1
408 in pass while loop 2
409 in pass while loop 3
410 in pass while loop 4
411 in pass while loop 5
412 in pass while loop 6
413 PassScann finished at: 7
414 RejectScan Started False
415
416     Box passed successfully
417
418 Total passes : 2
419 Total rejects: 1
420 Error Count : 6
421 Pass ratio : 22.22
422 Reject ratio : 11.11

```

```

423 Error Ratio : 66.67
424 test completed in: 2.24
425
426     waiting for trigger
427     Enter 0 to load a pass board image
428     Enter 1 to load a reject board image
429
430 Enter selection to trigger test: 0
431 Test Running
432 Timer Running
433 Entering Keras While
434 In Keras While
435 Pass prediction is: 0.997331
436 Reject prediction is: 0.0026690152
437
438     Box passed test
439     behaviours:
440     1. Box rejected
441     2. Box rejected and shute full after rejection
442     3. box not rejected
443     4. box closer not running
444     5. box closer jammed
445
446
447 enter behaviour's number: 4
448 PassScan StartedBehaviour started False4
449 True
450 RejectScan Started False
451
452     Closer Stopped!
453
454 Total passes : 2
455 Total rejects: 1
456 Error Count : 7
457 Pass ratio : 20.0
458 Reject ratio : 10.0
459 Error Ratio : 70.0
460 test completed in: 1.07
461
462     waiting for trigger
463     Enter 0 to load a pass board image
464     Enter 1 to load a reject board image
465
466 Enter selection to trigger test: 0
467 Test Running
468 Timer Running
469 Entering Keras While
470 In Keras While
471 Pass prediction is: 0.997331
472 Reject prediction is: 0.0026690152
473
474     Box passed test
475     behaviours:
476     1. Box rejected
477     2. Box rejected and shute full after rejection
478     3. box not rejected
479     4. box closer not running

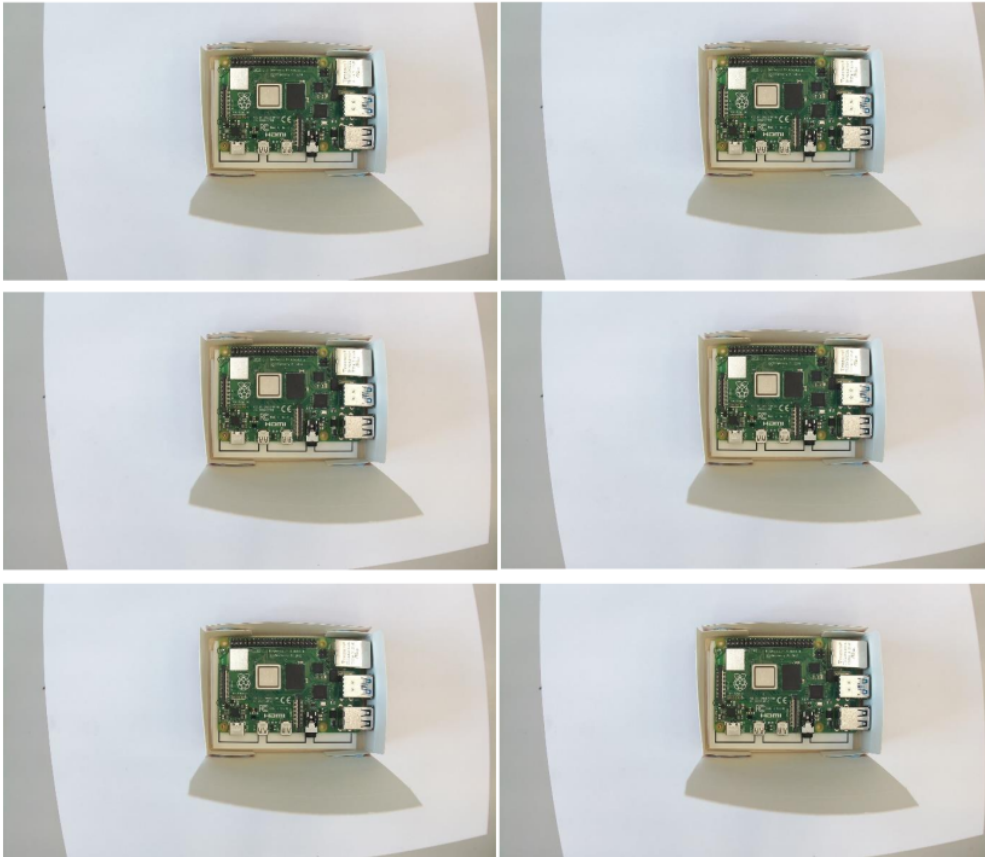
```

```
480         5. box closer jammed
481
482
483     enter behaviour's number: 5
484     PassScan StartedBehaviour started False5
485     True
486     in pass while loop 0
487     in pass while loop 1
488     in pass while loop 2
489     in pass while loop 3
490     in pass while loop 4
491     in pass while loop 5
492     in pass while loop 6
493     in pass while loop 7
494     in pass while loop 8
495     in pass while loop 9
496     in pass while loop 10
497     in pass while loop 11
498     in pass while loop 12
499     in pass while loop 13
500     in pass while loop 14
501     in pass while loop 15
502     in pass while loop 16
503     in pass while loop 17
504     in pass while loop 18
505     in pass while loop 19
506     in pass while loop 20
507     in pass while loop 21
508     in pass while loop 22
509     in pass while loop 23
510     in pass while loop 24
511     in pass while loop 25
512     in pass while loop 26
513     in pass while loop 27
514     in pass while loop 28
515     in pass while loop 29
516     PassScann finished at: 30
517     RejectScan Started False
518
519         Closer Error: Input to closer jammed and closer not stopped!!
520         Closer stopped!
521
522     Total passes : 2
523     Total rejects: 1
524     Error Count : 8
525     Pass ratio : 18.18
526     Reject ratio : 9.09
527     Error Ratio : 72.73
528     test completed in: 5.15
```

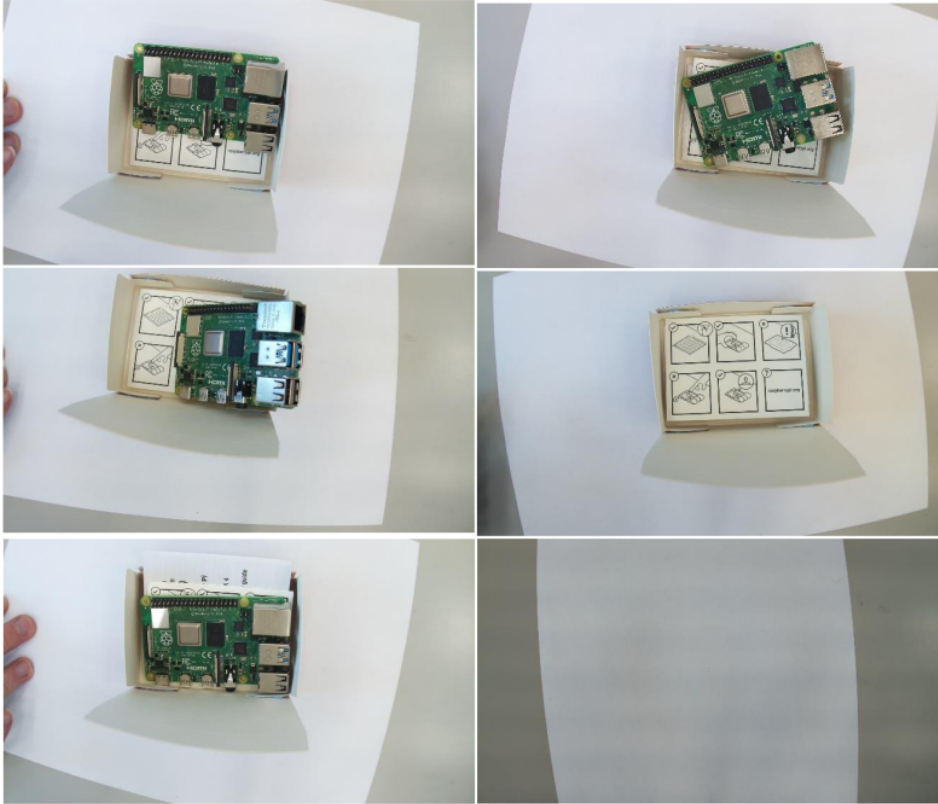
Appendix D. Sample Photos for Keras Model Training

Here are some example of pass and reject photos used for Keras model training. A total 2000 photos have been used for training.

Pass Photos



Reject Photos

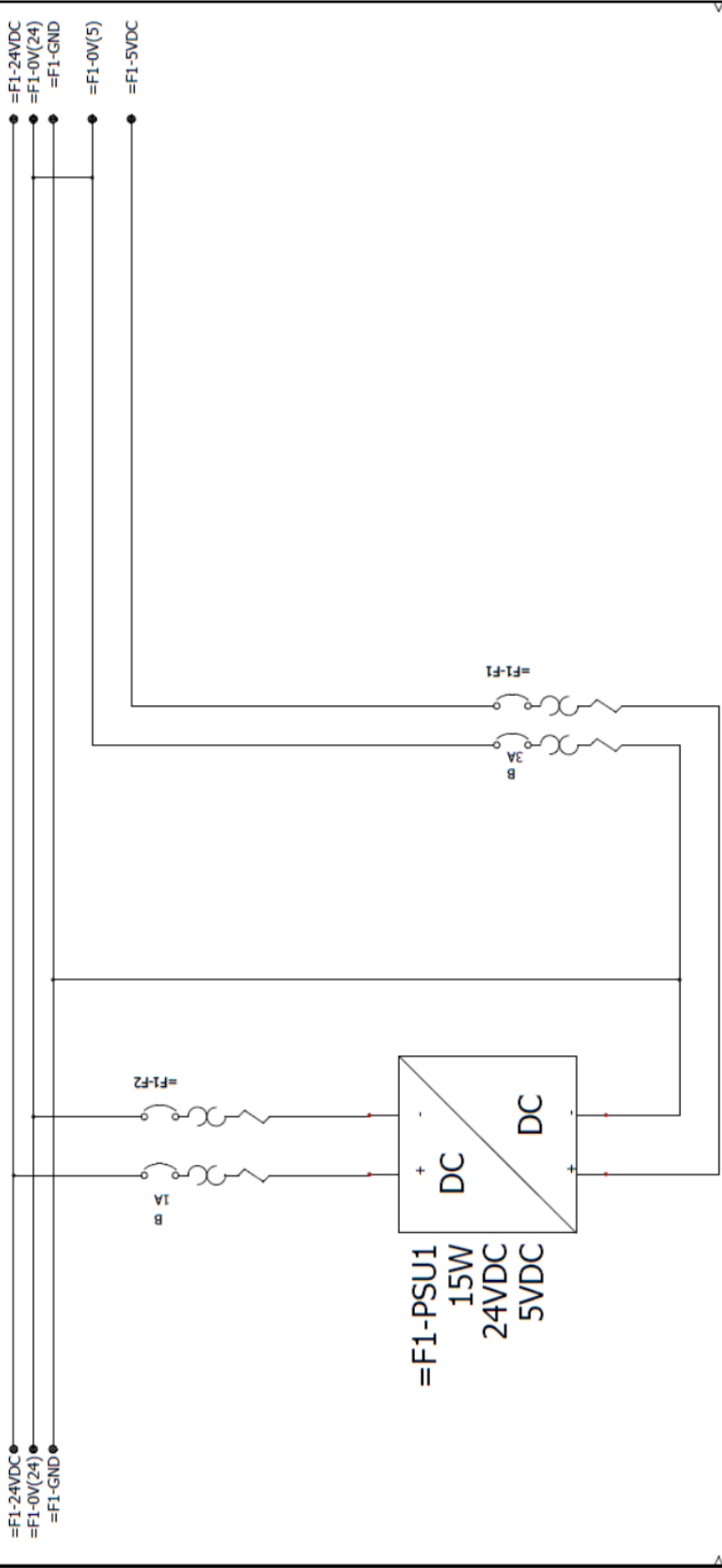


Python Code for Sequential Picture Taking

```
1  from picamera import PiCamera
2  from time import sleep
3
4  camera = PiCamera()
5
6  sleep(3)
7
8  camera.start_preview()
9  for i in range(100):
10     sleep(0.01)
11     camera.capture('/home/pi/Desktop/set00/image%s.jpg' % i)
12 camera.stop_preview()
```

Appendix E. Electrical Drawings

1	2	3	4	5	6	7	8	9	10
---	---	---	---	---	---	---	---	---	----



Adam Bedner	Electrical Drawing				0		30/04/2022	Adam	REVISION		
									0		
					REV.		DATE	NAME	CHANGES		
CONTRACT:	LOCATION:		+L1		Main Electrical Cabinet		User data 1				
										02	



Adam Bedner	Electrical Drawing	0	30/04/2022	Adam		REVISION	0
		REV.	DATE	NAME	CHANGES	SCHEME	03
		User data 1					
CONTRACT:	LOCATION: +L1	Main Electrical Cabinet					
		User data 2					

Appendix F. Teachable Machine Definitions

All the following definitions are taken from the training page of Google's Teachable machine.
(Google, 2022)

Accuracy per class

Accuracy per class is calculated using the test samples. Check out the vocab section to learn more about test samples.

Learning Rate

0.001

Be careful tweaking this number! Even small differences can have huge effects on how well your model learns.

Batch Size

Batch Size: 16

A batch is a set of samples used in one iteration of training. For example, let's say that you have 80 images and you choose a batch size of 16. This means the data will be split into $80 / 16 = 5$ batches. Once all 5 batches have been fed through the model, exactly one epoch will be complete.

You probably won't need to tweak this number to get good training results.

Epochs

Epochs: 50

One epoch means that each and every sample in the training dataset has been fed through the training model at least once. If your epochs are set to 50, for example, it means that the model you are training will work through the entire training dataset 50 times. Generally the larger the number, the better your model will learn to predict the data.

You probably want to tweak (usually increase) this number until you get good predictive results with your model.

Accuracy

Accuracy is the percentage of classifications that a model gets right during training. If your model classifies 70 samples right out of 100, the accuracy is $70 / 100 = 0.7$.

If the model's prediction is perfect, the accuracy is one; otherwise, the accuracy is lower than one.

Vocab

Teachable Machine splits your samples into two buckets. That's why you'll see two labels, training and test, in the graphs below.

Training samples: (85% of the samples) are used to train the model how to correctly classify new samples into the classes you've made.

Test samples: (15% of the samples) are never used to train the model, so after the model has been trained on the training samples, they are used to check how well the model is performing on new, never-before-seen data.

Underfit: a model is underfit when it classifies poorly because the model hasn't captured the complexity of the **training samples**.

Overfit: a model is overfit when it learns to classify the **training samples** so closely that it fails to make correct classifications on the **test samples**.

Epochs: One epoch means that every **training sample** has been fed through the model at least once. If your epochs are set to 50, for example, it means that the model you are training will work through the entire training dataset 50 times.

Loss

Loss is a measure for evaluating how well a model has learned to predict the right classifications for a given set of samples. If the model's predictions are perfect, the loss is zero; otherwise, the loss is greater than zero.

To get an intuitive sense of what this measures, imagine you have two models: A and B. Model A predicts the right classification for a sample but is only 60% confident of that prediction. Model B also predicts the right classification for the same sample but is 90% confident of that prediction. Both models have the same accuracy, but model B has a lower loss value.

loss

test loss

Epochs

Confusion Matrix

A confusion matrix summarizes how accurate your model's predictions are. You can use this matrix to figure out which classes the model gets confused about.

The y axis (Class) represents the class of your samples. The x axis (Prediction) represents the class that the model, after learning, guesses those samples belong to. So, if a sample's Class is "Muffin" but its Prediction is "Cupcake", that means that after learning from your data, the model misclassified that Muffin sample as a Cupcake. This usually means that those two classes share characteristics that the model picks up on, and that particular "Muffin" sample was more similar to the "Cupcake" samples.

Appendix G. Log book

Log Entry Sheet

Log Number	18
Date	11/02/2022
Duration (hrs)	2h
Accumulated time (hrs)	76.5h

Record of activities and achievements

What went well:

Organised a meeting with the head of Production Engineering showing the presentation, explaining all of the important points and explaining the choice of solution.

Send off an order for the essential parts for the first phase of implementation (picture gathering and implementation).

What did not go well:

The amount and style of pictures taken are not sufficient and inadequate for accurate algorithm training.

Plans for next week

Once order comes through, install the camera and re-take the pictures, capturing a minimum of 250 pictures of each possible case (good and bad).

Make a block and layout diagrams for records and further meetings.

Book a meeting with IIOT engineering department to discuss the IIOT implementation of the system as well as the specification and capabilities of the Rubus interface board (Confidential information).

Contingency if project cannot be implemented in the time scale

Signed _____

Signed

Log Entry Sheet

Log Number	19
Date	18/02/2022
Duration (hrs)	2.5h
Accumulated time (hrs)	79h

Record of activities and achievements

What went well:

Block and layout diagrams progressed but not yet finished.

IOT Department contacted and IOT implementation discussed. The use of Rubus could potentially limit the capabilities of the setup, but not significantly.

Contingency plan developed to implement the system on a spare production line that has been taken out of service.

What did not go well:

The order for the materials have not yet arrived.

IOT warned that there could be a potential confidentiality issue about the use of an online machine learning according to the company's rules.

Plans for next week

Chase the order for the camera. Take the sample pictures once the order arrives.

Finish the block and layout diagrams

Start the electrical drawings

Contact the IT Department and request for Google Teachable Machine to be looked into for possible confidentiality issues.

Signed _____

Signed

Log Entry Sheet

Log Number	20
Date	25/02/2022
Duration (hrs)	9.5h
Accumulated time (hrs)	88.5h

Record of activities and achievements

What went well:

Layout and block diagrams have been completed.
Electrical diagrams started.
All ok from IT regarding confidentiality.
Mind map for the Python code development started.

What did not go well:

Still waiting on the Camera order, unable to chase the order back; management looking into it.
IT Contacted, they are looking into the issue but no updates yet. Chasing them daily for updates.

Plans for next week

Chase the management about the camera order updates.

Continue electrical diagram and mind map for python code.

Develop a new contingency plan for the capture of the sample pictures.

Signed _____

Signed

Log Entry Sheet

Log Number	21
Date	04/03/2022
Duration (hrs)	10
Accumulated time (hrs)	98.5

Record of activities and achievements

What went well:

The Camera order finally arrived on Thursday 03/03.

New contingency plan developed; it involves using the college's conveyor system to capture the sample pictures, develop the algorithm and test the system.

Python mind map completed.

Electrical drawings progressed.

What did not go well:

Production to increase the following weeks, limiting the ability to use the lines for the project.

Plans for next week

Begin sampling in college next week. Aim to have a working algorithm by next week. The number of samples required increased to 1,000 per class to ensure the accuracy and stability of the model.

Develop a sequential picture taking code, to capture pictures in batches 100.

Finish Electrical drawings

Start developing Python code.

Signed _____

Signed

Log Entry Sheet

Log Number	22
Date	11/03/2022
Duration (hrs)	0.5h
Accumulated time (hrs)	99

Record of activities and achievements

What went well:

What did not go well:

Project not progressed due to illness and self isolation

Plans for next week

Same as Log #21.

Signed _____

Signed

Log Entry Sheet

Log Number	23
Date	18/03/2022
Duration (hrs)	0h
Accumulated time (hrs)	99

Record of activities and achievements

What went well:

What did not go well:

Project not progressed due to illness and self isolation

Plans for next week

Same as Log #21.

Signed _____

Signed

Log Entry Sheet

Log Number	24
Date	25/03/2022
Duration (hrs)	21
Accumulated time (hrs)	110

Record of activities and achievements

What went well:

1,000 pictures of good and bad pictures taken, and sorted in the right folders.

Electrical drawings finished.

Python code development started

What did not go well:

Keras model not yet started.

Plans for next week

Begin sampling in college next week. Aim to have a working algorithm by next week. The number of samples required increased to 1,000 per class to ensure the accuracy and stability of the model.

Develop a sequential picture taking code, to capture pictures in batches 100.

Finish Electrical drawings

Start developing Python code.

Signed _____

Signed

Log Entry Sheet

Log Number	25
Date	01/04/2022
Duration (hrs)	23
Accumulated time (hrs)	133

Record of activities and achievements

What went well:

Keras model trained and tested. It needed a lot of trial-and-error with the advanced parameters to produce a satisfactory model.

A meeting with production management revealed that none of the lines will be released for implementation until at least June or July time.

What did not go well:

The current Decision making python code does not work.

Plans for next week

Attempt employing different methods to the python code to get it to run.

Signed _____

Signed

Log Entry Sheet

Log Number	26
Date	08/04/2022
Duration (hrs)	15
Accumulated time (hrs)	148

Record of activities and achievements

What went well:

Keras model and image processing implemented into python code. Commented out for decision making debugging.

What did not go well:

Threading used in the python code, it did not work properly.

Plans for next week

Keep debugging the python code.

Signed _____

Signed

Log Entry Sheet

Log Number	27
Date	15/04/2022
Duration (hrs)	12
Accumulated time (hrs)	160

Record of activities and achievements

What went well:

What did not go well:

Multiprocessing used in the python code, it did not work properly.

Plans for next week

Keep debugging the python code.

Signed _____

Signed

Log Entry Sheet

Log Number	28
Date	22/04/2022
Duration (hrs)	24
Accumulated time (hrs)	184

Record of activities and achievements

What went well:

The code has been split into main code, classes, defs, and B.

Main code has been rewritten to call and execute the splitted out code which now are modules that are loaded in and executed by the main code.

What did not go well:

Minor bugs and glitches that need addressing

Plans for next week

Address the minor bugs and glitches and fully test the code.

Signed _____

Signed

Log Entry Sheet

Log Number	29
Date	29/04/2022
Duration (hrs)	12
Accumulated time (hrs)	196

Record of activities and achievements

What went well:

Minor bugs and glitches addressed.
code fully tested via simulation

What did not go well:

Awaiting line release for implementation.

Plans for next week

Signed _____

Signed

1708165

GRADEMARK REPORT

FINAL GRADE

GENERAL COMMENTS

Instructor

52/100

PAGE 1

PAGE 2

QM **Good Structure**

Report is structured exactly as stated in the assignment brief.

PAGE 3

PAGE 4

PAGE 5

Text Comment. Your FDS could have been more thoroughly researched to find out the company needs

Text Comment. Your solution should have been agreed with the company before implementing

Text Comment. Your summary doesn't say if the system actually works

PAGE 6

PAGE 7

Text Comment. Should never have been in the FDS

Text Comment. This shows how important the FDS is.

Text Comment. You could have considered this as a contingency in your Gantt chart

Text Comment. This table belongs in the appendix

QM

Spelling

Spelling error

QM

Word/s missing

QM

Word usage

A more appropriate word could have been used here

QM

Spelling

Spelling error

QM

Word/s missing

Text Comment. Your report is cluttered by images most of which should be in the appendix

QM

Spelling

Spelling error

Text Comment. You could have used a polling or interrupt method to avoid hanging

Text Comment. This is not the correct way to quote using the Harvard method.

QM

Spelling

Spelling error

QM

Spelling

Spelling error

QM

Spelling

Spelling error

Text Comment. What is a numpy array?

QM

Spelling

Spelling error

QM

Spelling

Spelling error

Text Comment. What does this mean?

QM

Spelling

Spelling error

Text Comment. You could have removed these from the table and explained the difference between the new FDS and the old one at the beginning of the report in more detail.

PAGE 25

PAGE 26

PAGE 27

PAGE 28

Text Comment. It is not clear to see that this text provides any real evidence

PAGE 29



Spelling

Spelling error

Text Comment. This is an admission that you did not think this solution through carefully enough when selecting it.

PAGE 30

PAGE 31

Text Comment. This section is disappointingly short

PAGE 32

Text Comment. You could have used more sources in your design

PAGE 33

PAGE 34

PAGE 35

PAGE 36

PAGE 37

PAGE 38

PAGE 39

PAGE 40

PAGE 41

PAGE 42

PAGE 43

PAGE 44

PAGE 45

PAGE 46

PAGE 47

PAGE 48

PAGE 49

PAGE 50

PAGE 51

PAGE 52

PAGE 53

PAGE 54

PAGE 55

PAGE 56

PAGE 57

PAGE 58

PAGE 59

PAGE 60

PAGE 61

PAGE 62

PAGE 63

PAGE 64

PAGE 65

PAGE 66

PAGE 67

PAGE 68

PAGE 69

PAGE 70

PAGE 71

PAGE 72

PAGE 73

PAGE 74

PAGE 75

GRADING FORM: NEW CSG-RUBRIC

ADAM BEDNER

52

1. ENGAGEMENT WITH LITERATURE



Section 3: Judicial use and critical evaluation of technical literature in the design of the project: There is good evidence of use of technical literature in the learning and design of the software coding and the recognition engine
Section 4: Extent of reading, range of credible sources and use of Harvard method: Source used are credible and appropriate, the list is moderately small for this level and citations are not always in conformity to the Harvard method. 6/10

6

2. KNOWLEDGE AND UNDERSTANDING

 Section 3: Advanced knowledge and critical understanding demonstrated in the design of the project: There is advanced knowledge and understanding demonstrated in the report as the programming of the system is very complex. 7/10

7

3. COGNITIVE AND INTELLECTUAL

 Section 5: Testing and analysis of result: There is good evidence of testing through simulation but no evidence for the complete system as this was not implemented on the machine. Results of simulation are thorough showing how the final value developed. Section 5: Critical evaluation of the project outcomes: There is a clear evaluation of the project outcomes but it does reveal that the chosen solution was not the best after all things were considered. Section 6: Conclusions and recommendations: This section is very brief and says very little. 12/30

12

4. PRACTICAL APPLICATION

 Section 3: The design work in software is very innovative but the hardware idea for interfacing is not the most practical as it would require the production of a PCB and a number of spares for the customer. This had not been considered carefully enough when the solution was chosen. Section 4: Methods were very good in the way the

16

teach machine was developed and tested with the microcontroller board and the camera. The software was of a high standard but the hardware was not produced nor implemented on plant.

Section 5: Use of test equipment and presentation of results of the project: Very good presentation of the simulation results. Sections 3 and 5:

Management of the design process: No Gantt chart included in the report and project has been poorly planned as a number of issues were not considered. Logbook entries in the appendix show development but no spreadsheet included to show how costs were met. 16/35

5. TRANSFERABLE SKILLS



Sections 1 and 2: Executive summary, acknowledgements and introduction: Summary covers most aspects of the project, acknowledgements provide an honest account of the help received and the introduction sets out the aims and the objectives of the project. All Sections: Quality of the report: A well structured and presented report. However, it has not been well written and contains many punctuation and spelling errors. The report is cluttered with images, most of which should have been put in the appendix. Word count: Not exceeded. 11/15

11