Dwight Look College of ENGINEERING
TEXAS A&M UNIVERSITY

Team 2: Robotic Sorting System
Bi-Weekly Update 3

Pace Dominy, Joseph Miller, Lam Tran
Sponsor: Dalton Cyr
TA: Swarnabha Roy

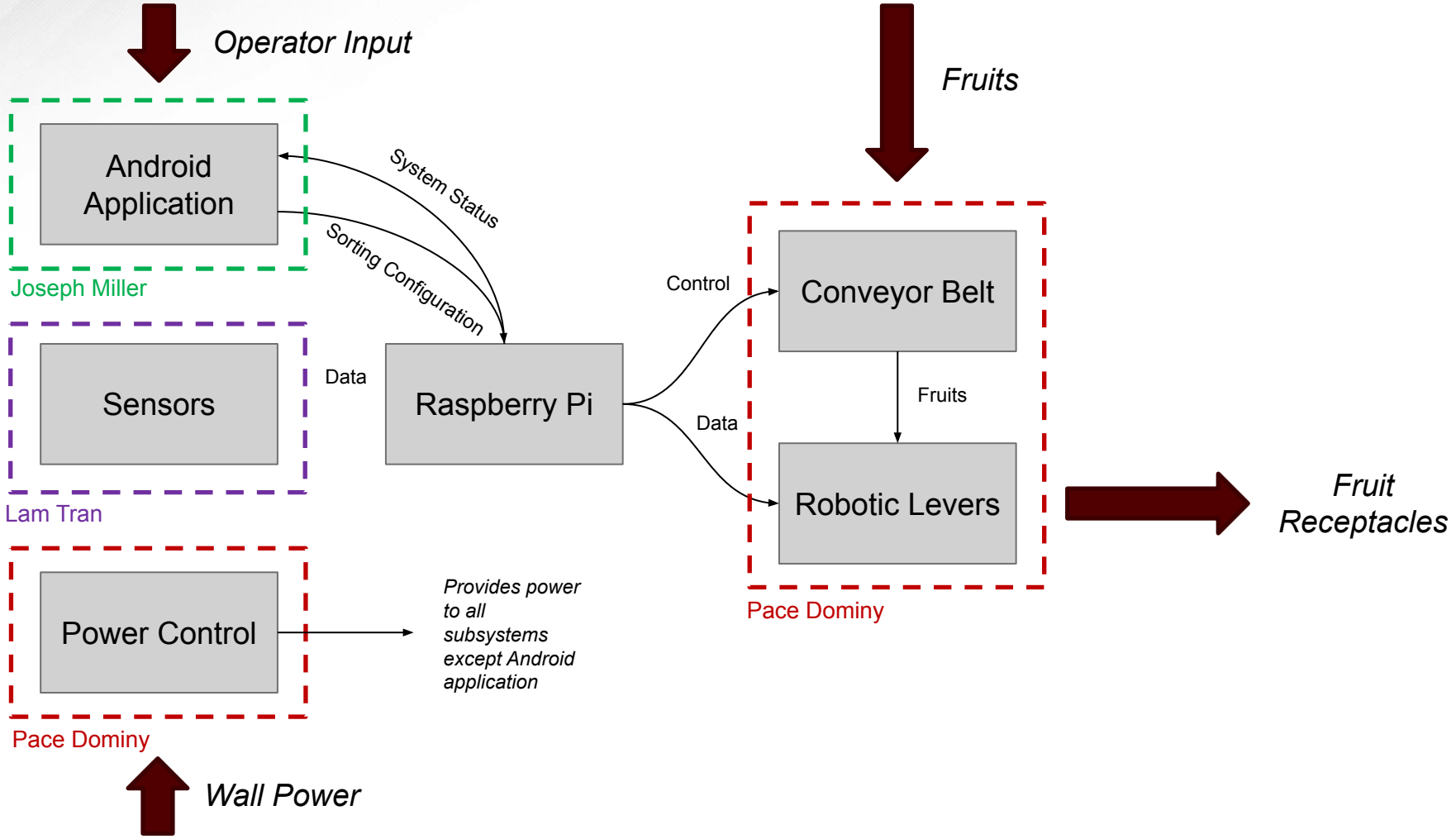
Project Description

Sorting fruit by hand can be tedious and expensive.

The Robotic Sorting System is an automatic sorting system that requires limited manpower/oversight that can accurately sort fruit by their size and quality without the user having to do it themselves.

System Diagram

Maroon arrows denote physical inputs and outputs



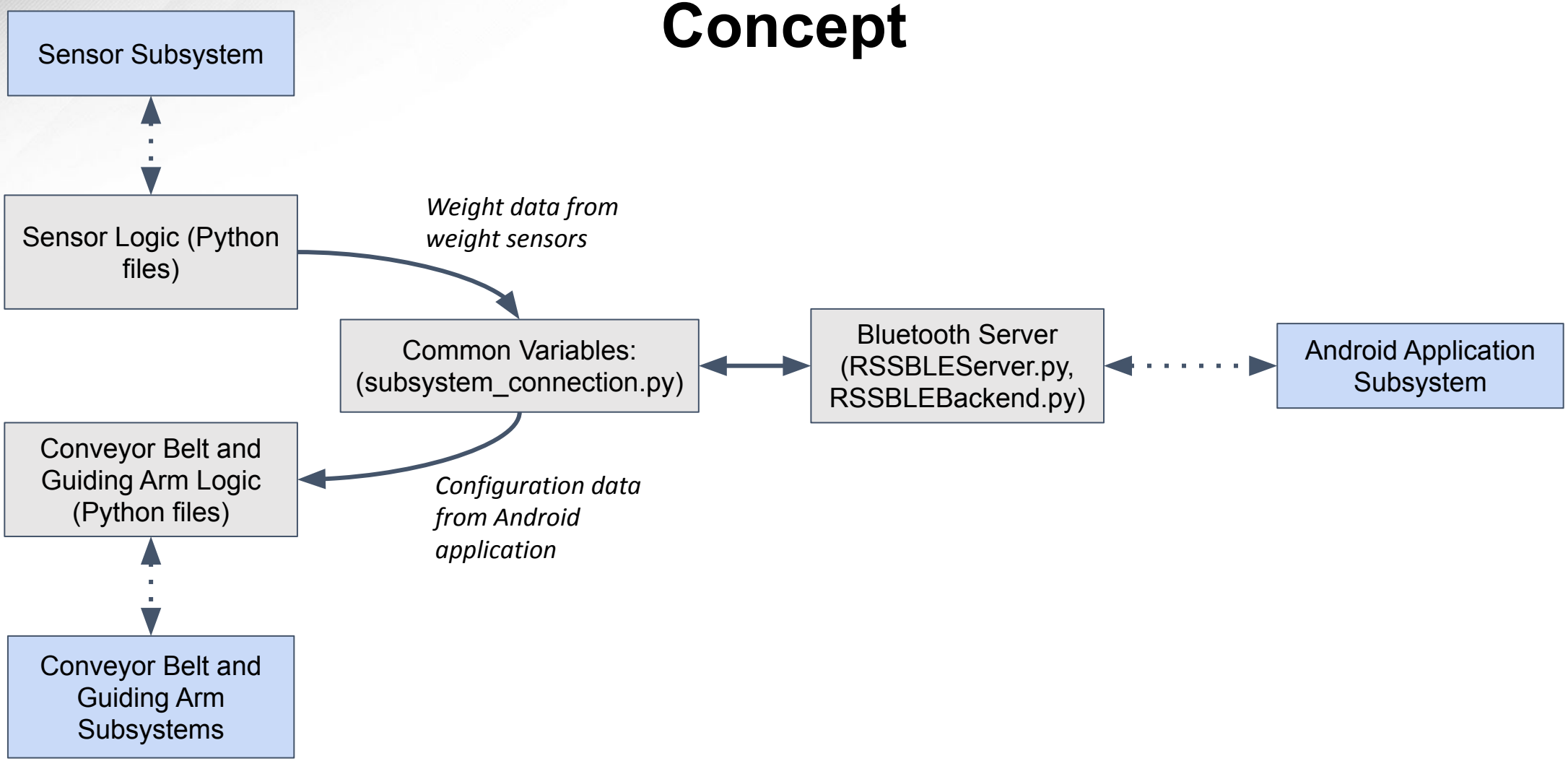
Project Timeline

Finished Android Application subsystem (completed 2/5)	Finished 403 Sensor subsystem (completed 2/19)	Power Control System (to complete by 2/20)	Final Integration of Android App connection (to complete by 2/25)	Finish basic Robotic Sorting System functionality (to complete by 3/7)	All subsystems integrated (to complete by 4/9)	Final Project Demonstration (to complete by 4/24)

Joseph Miller

Accomplishments since Status Update 2 (9 hours of effort)	Ongoing progress/problems and plans until the next presentation
● Helped solder components onto the new revision of the system circuit board ● Fixed bugs preventing the Android application from connecting to the RSS Raspberry Pi ○ Successfully resolved on an identical system image ● Created a simple Python structure to allow communication between subsystems ○ Necessary for full integration	● Help build other subsystems ● Begin testing of Android app / Bluetooth connection with other subsystems when relatively stable ● Make BLE server monitor changes in the “subsystem connection” file and update the values on the server as necessary

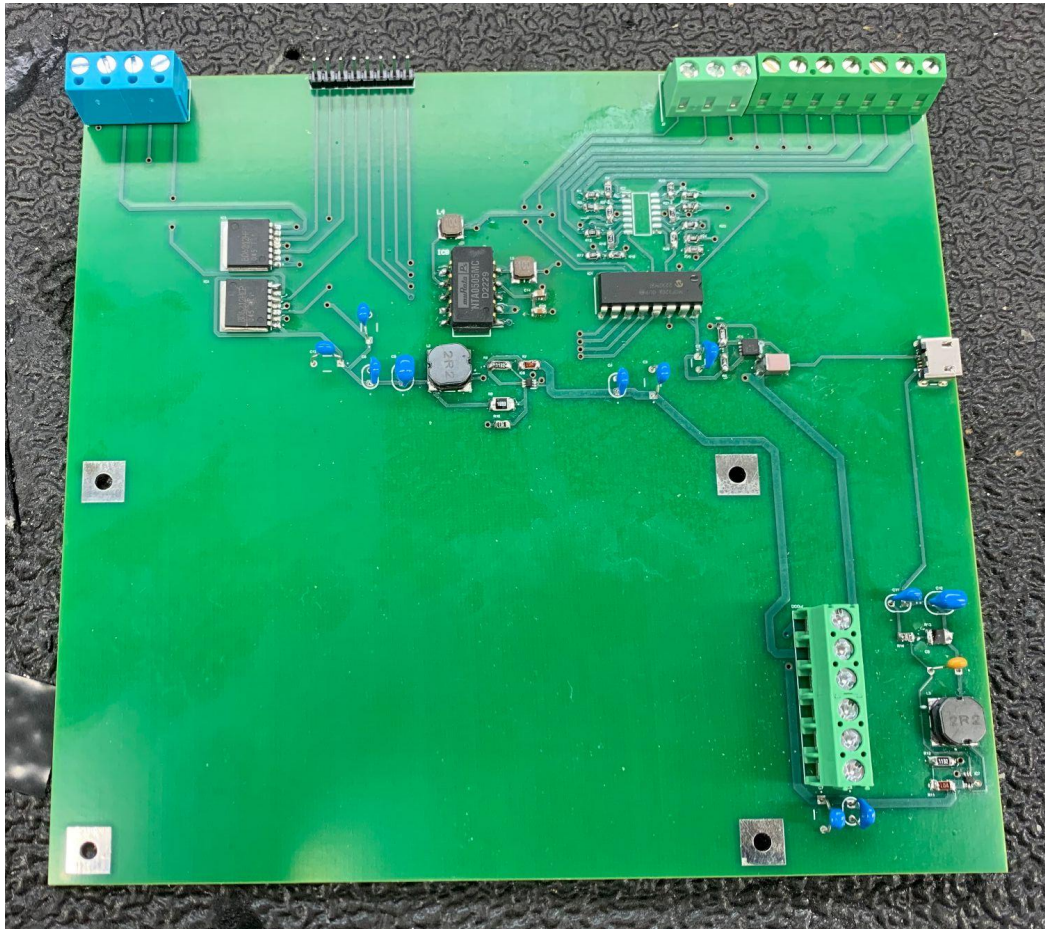
Robotic Sorting System Integration Concept



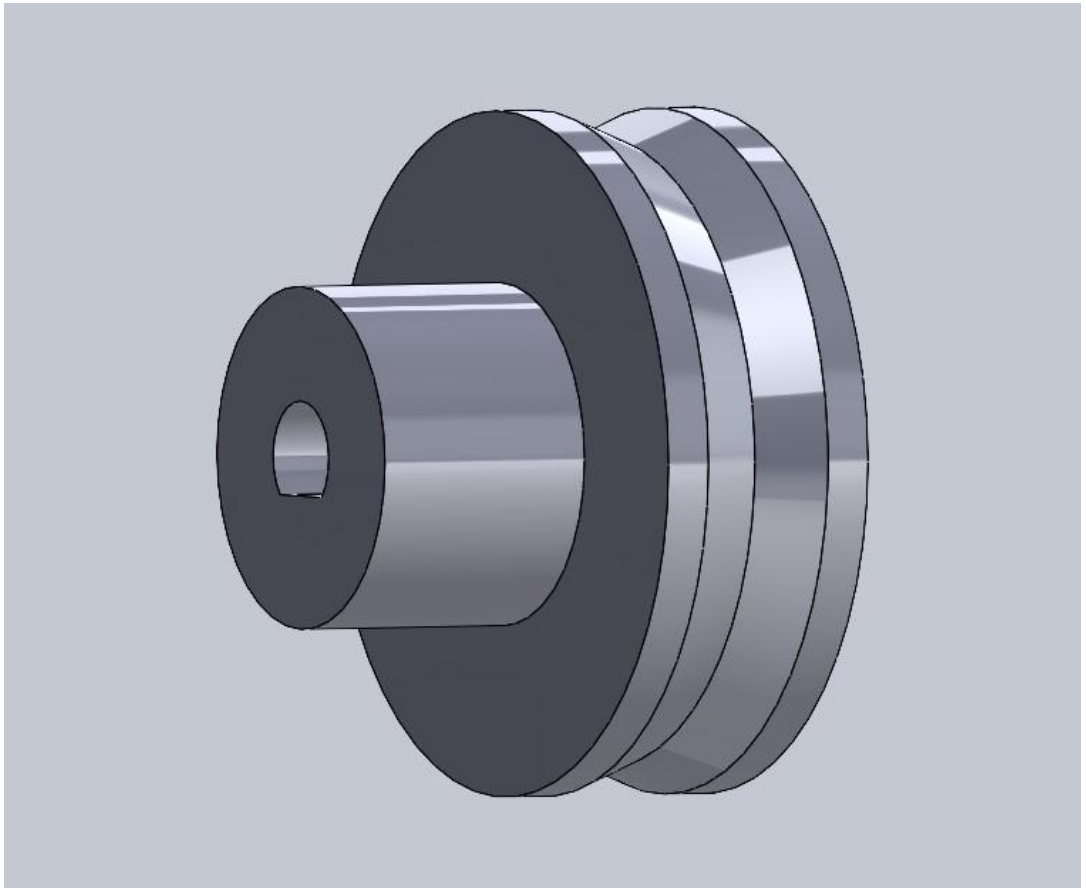
Pace Dominy

Accomplishments since Status Update 2 (14 Hours of Effort)	Ongoing progress/problems and plans until the next presentation
• Soldered PCB with help from Joseph • Updated solidworks design for 3 receptacles • Created and submitted STL files for motor connectors to be 3D printed • Bought COTS to test motor functionality for conveyor belt	• Validate PCB • Ensure 3D printed motor connectors mesh with motors • Test motor functionality • Build conveyor belt and guiding rails • Finalize python code for motors after testing them

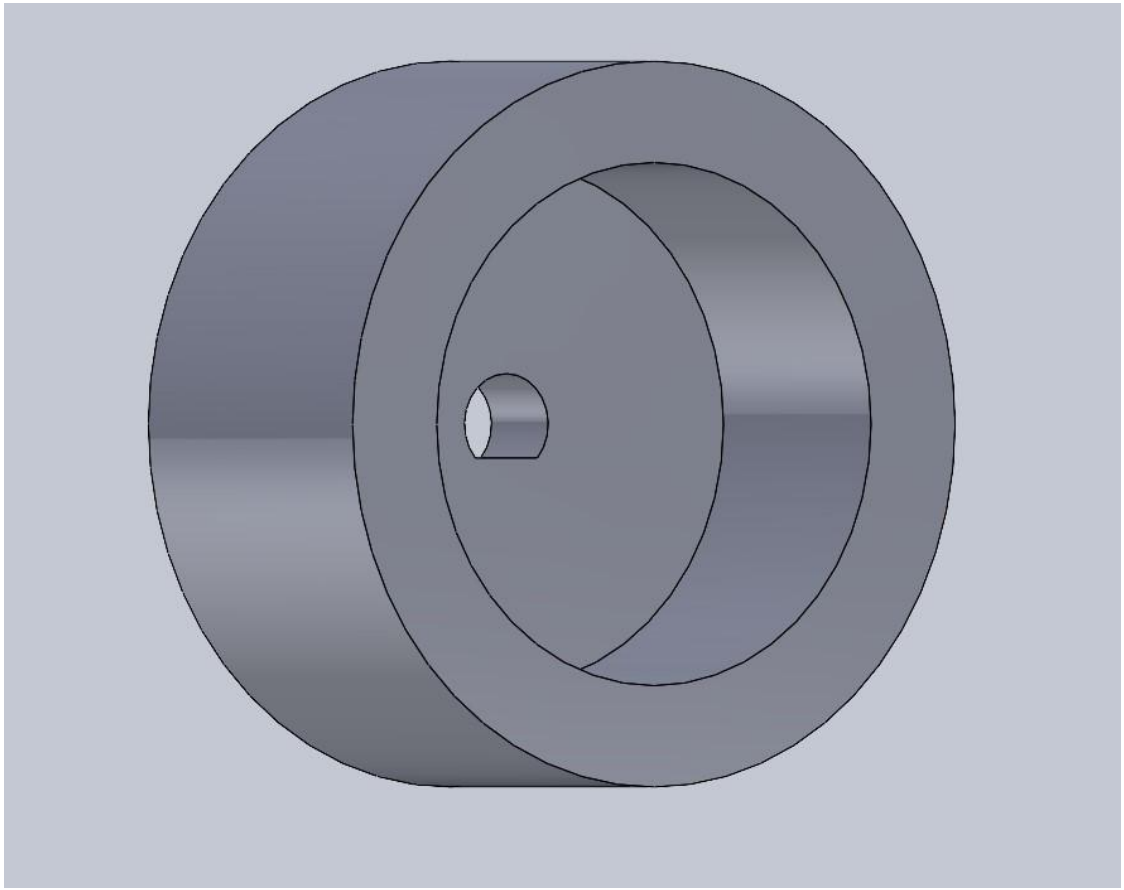
Soldered PCB



Motor Connector



Guiding Arms Motor Connector

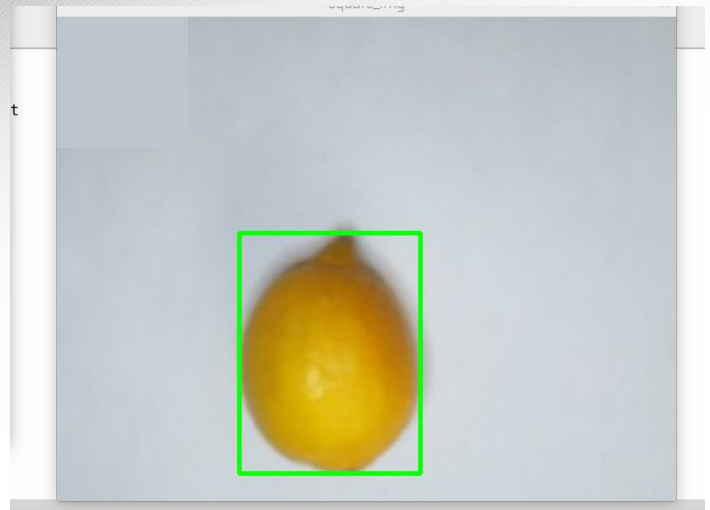


Conveyor Belt Motor Connector

Lam Tran

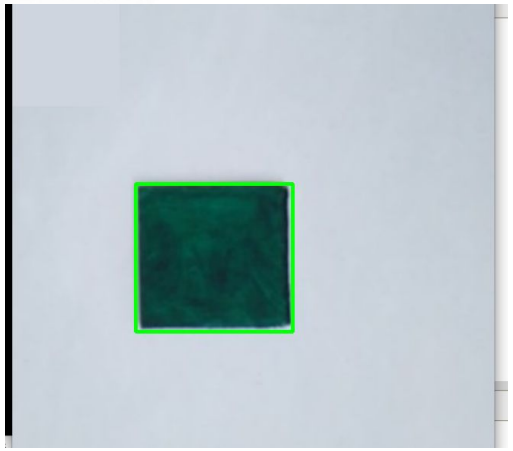
Accomplishments since Status Update 2 (23 hrs of effort)	Ongoing progress/problems and plans until the next presentation
• Tested the color and size sensor with objects of varying sizes and colors with the Raspberry Pi • Tested the weight sensor code with 3 load cells • Found some dataset • Found a pretrained model • Installed the pretrained model to the Raspberry Pi	• Integrating sensor subsystem with other subsystem • Having the camera works with both the color/size sensor code and the machine learning code. • Labeling the data images • Training the model with the labeled images • Test the model with fruits

Color/Size sensor tested

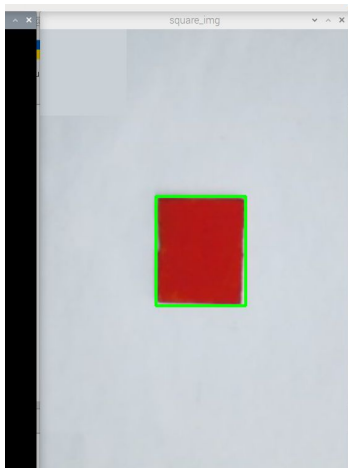


Shell

```
area:  8662.0
pixels per inch is  93.06986622962343
the number of pixels within the object:  36272
mean of the collected hue is  21.239937141596823
median of the collected hue is  22.0
Object hue is  YELLOW
total pixel of the obj is  36272
red percent:  0.0
orange percent:  46.69166299073666
yellow percent:  53.30833700926334
green percent:  0.0
blue percent:  0.0
violet percent:  0.0
height in inches:  2.7506217680424383
width in inches:  2.0737109423132445
```

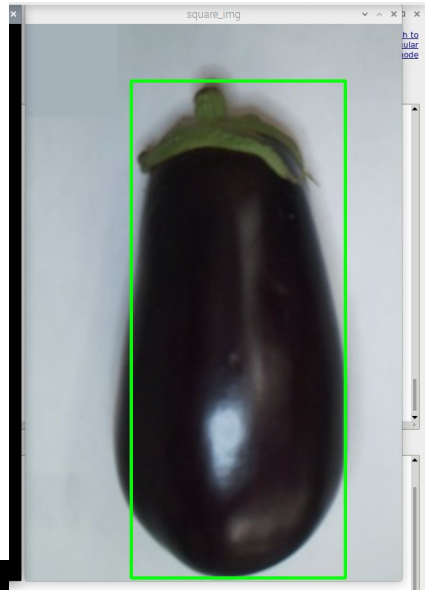


```
area
pixels
the number of pixels within the object:  38120
mean of the collected hue is  87.7460388247639
median of the collected hue is  87.0
Object hue is  BLUE
total pixel of the obj is  38120
red percent:  0.0
orange percent:  0.0
yellow percent:  0.0
green percent:  0.028856243441762856
blue percent:  99.97114375655823
violet percent:  0.0
height in inches:  2.3708213181616546
width in inches:  2.5182298975292188
```



```
area:  7412.0
pixels per inch is  86.09297300012
the number of pixels within the obj
mean of the collected hue is  17.98
median of the collected hue is  1.0
Object hue is  RED
total pixel of the obj is  24802
red percent:  99.37505039916135
orange percent:  0.5483428755745504
yellow percent:  0.0
green percent:  0.0
blue percent:  0.0
violet percent:  0.07660672526409160
height in inches:  2.055916921346710
width in inches:  1.684225726526971
```

```
area:  7555.5
pixels per inch is  86.922379166
the number of pixels within the object
mean of the collected hue is  37.66537
median of the collected hue is  37.0
Object hue is  GREEN
total pixel of the obj is  29006
red percent:  0.0
orange percent:  0.02413293801282493
yellow percent:  1.6858580983244846
green percent:  98.29000896366269
blue percent:  0.0
violet percent:  0.0
height in inches:  2.4964802169511544
width in inches:  2.0132904975412536
```

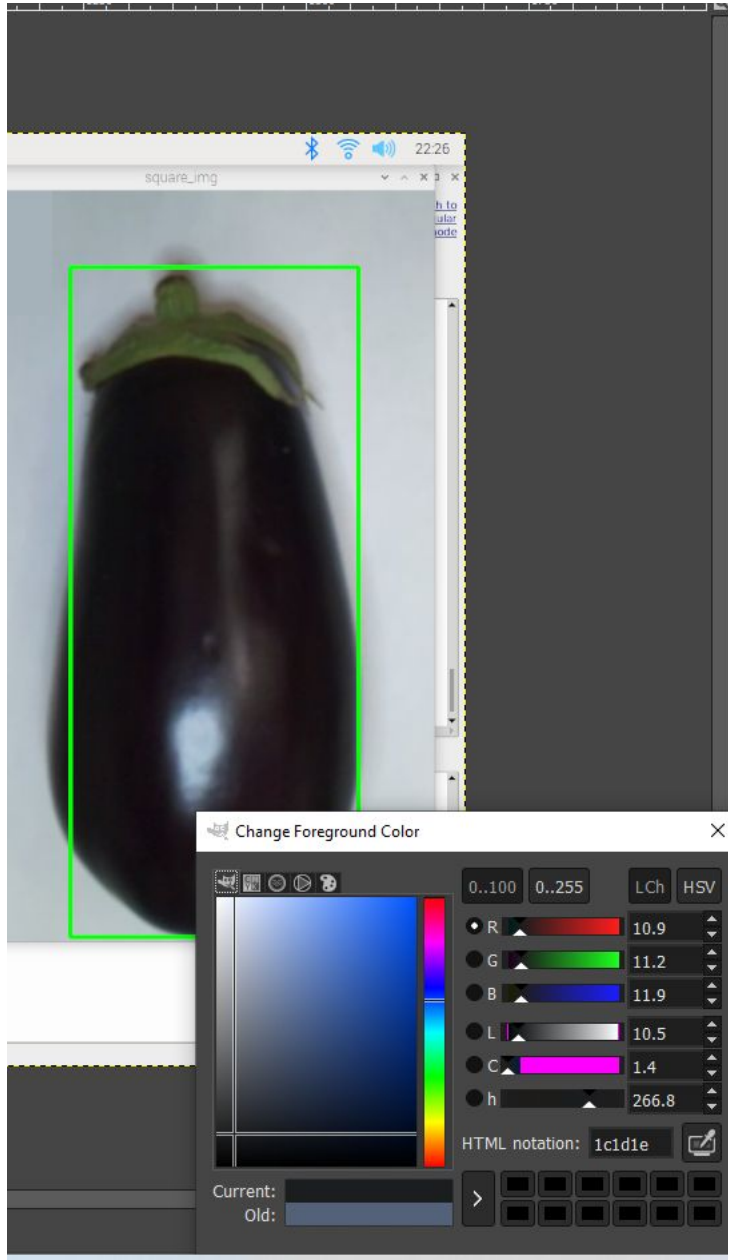


```
221
222 cv2.waitKey(0)
```

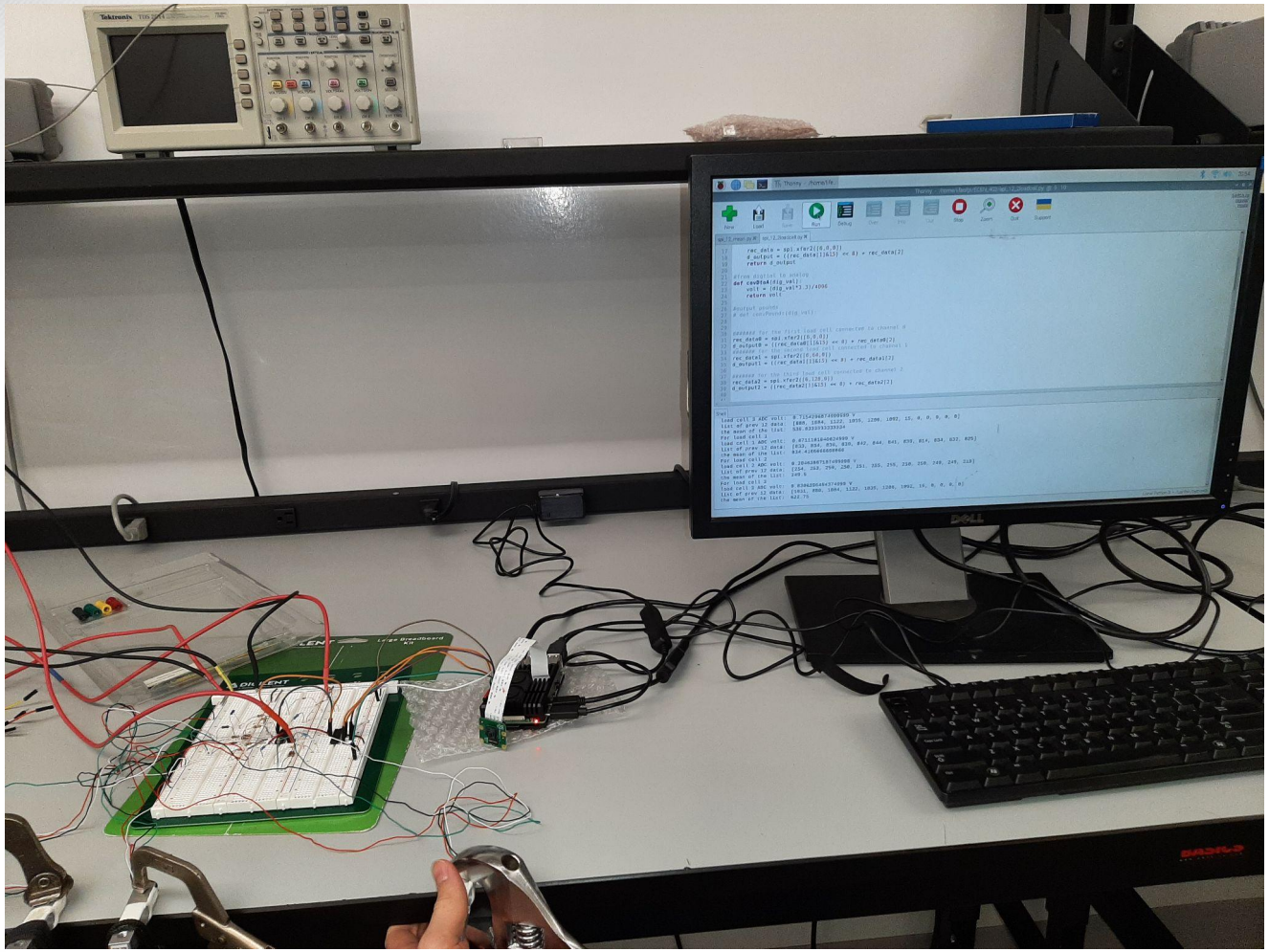
Shell

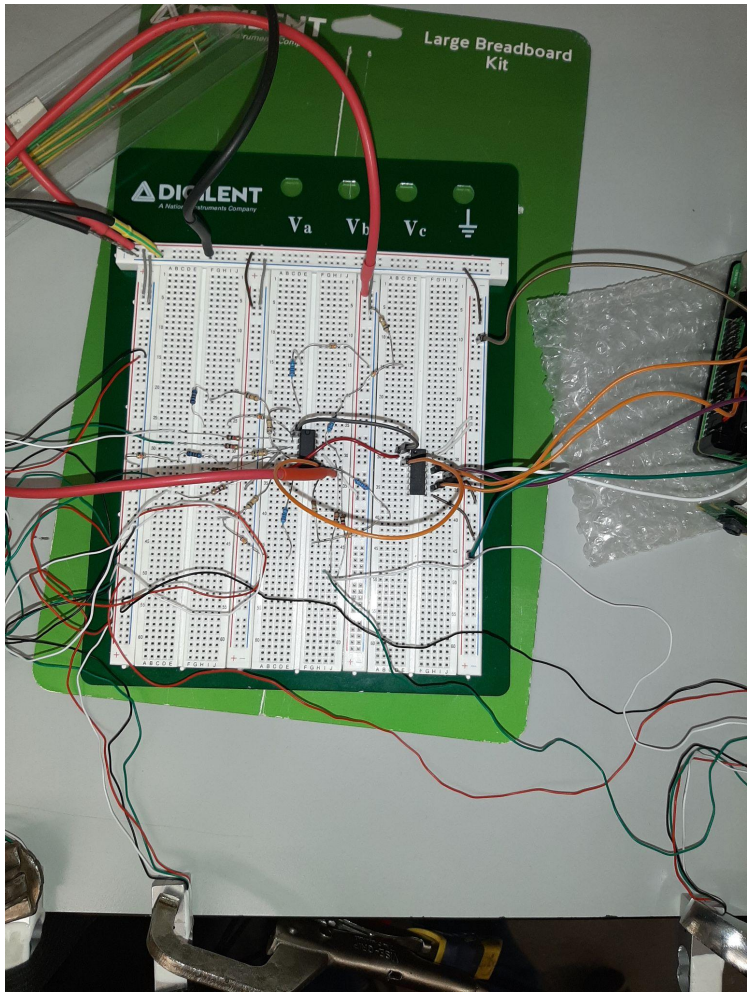
```
area:  7992.0
pixels per inch is  89.39798655450804
the number of pixels within the objec
mean of the collected hue is  133.172
median of the collected hue is  156.0
Object hue is  VIOLET
total pixel of the obj is  133613
red percent:  26.595466010043932
orange percent:  3.1800797826558793
yellow percent:  2.461586821641607
green percent:  8.119718889629004
blue percent:  12.360324219948657
violet percent:  47.28282427608092
height in inches:  8.423008492936006
width in inches:  3.624242698155731
```

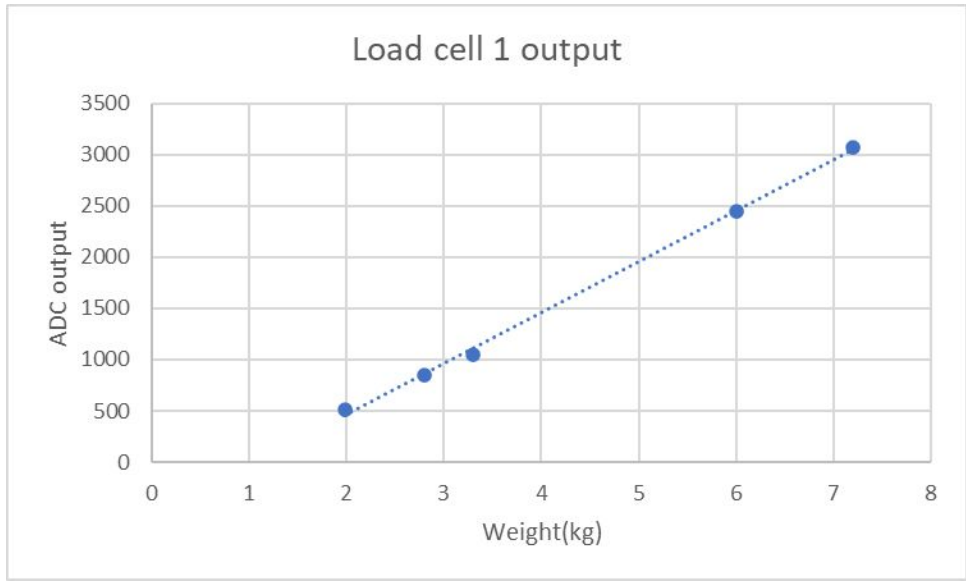
1	This test v	The resolution was 800x600						
2	Item	Height by	Width by	Color by e	diangle le	Height by	Width by sensor	Height err	Width Error percent
3	light blue	2.25	2.33	terf blue	3.37	2.371	2.518	5.377778	8.06867
4	red+blue=	3.77	2.33	purple	4.34	4.179	2.585	10.84881	10.94421
5	red+2yell	2.1875	1.9375	orange	2.9375	2.056	1.684	6.011429	13.08387
6	green	1.875	2.25	green+yel	2.8125	1.913	2.35	2.026667	4.444444
7	purple	1.3125	1.5	purple	1.875	1.415	1.588	7.809524	5.866667
8	blue	1.4375	1.4375	blue	1.9375	1.467	1.55	2.052174	7.826087
9	weirdyell	1.375	2	brown yel	2.4375	1.4185	2.048	3.163636	2.4
10	dark brow	2	1.25	mostly da	2.3125	2.018	1.187	0.9	5.04
11	light brow	1.625	1.4375	light brow	2.125	1.669	1.455	2.707692	1.217391
12	red	1.4375	1.5625	red	2.125	1.497	1.618	4.13913	3.552
13	small size	1.4375	1.3125	scarlet rec	1.875	1.478	1.313	2.817391	0.038095
14	medium s	2	1.6875	scarlet rec	2.625	2.178	1.995	8.9	18.22222
15	big yellow	2.3125	4	yellow	4.5625	2.32	3.96	0.324324	1
16	lemon bot	3.5	2.75	yellow	3.75	3.9	2.951	11.42857	7.309091
17	lemon bot	2.75	3.5	yellow	3.75	2.825	3.71	2.727273	6
18	lime bottl	3.25	2.75	green	3.75	3.452	2.793	6.215385	1.563636
19	lime bottl	2.75	3.25	green	3.75	2.902	3.419	5.527273	5.2
20	capped lir	3.5	2.75	green witl	3.75	3.795	3	8.428571	9.090909
21	capped lir	2.75	3.5	green witl	3.75	3.0235	3.881	9.945455	10.88571
22								#DIV/0!	#DIV/0!
23	red onisor	3.597	3.17	red to purple		4.203	3.495	16.84737	10.25237
24	lemon	2.658	2.001	yellow		2.751	2.072	3.498871	3.548226
25	lime	2.317	1.9635	green		2.496	2.013	7.725507	2.521008
26	tangerinte	2.2565	2.2305	orange		2.48	2.35	9.90472	5.357543
27	granny ap	2.7695	2.4935	green		2.909	2.698	5.03701	8.201323
28	banana	7.5	3.712	yellow+brown spots		7.651	4.065	2.013333	9.509698
29	eggplant	7.75	3.561	purple+green top		8.423	3.624	8.683871	1.769166
30									

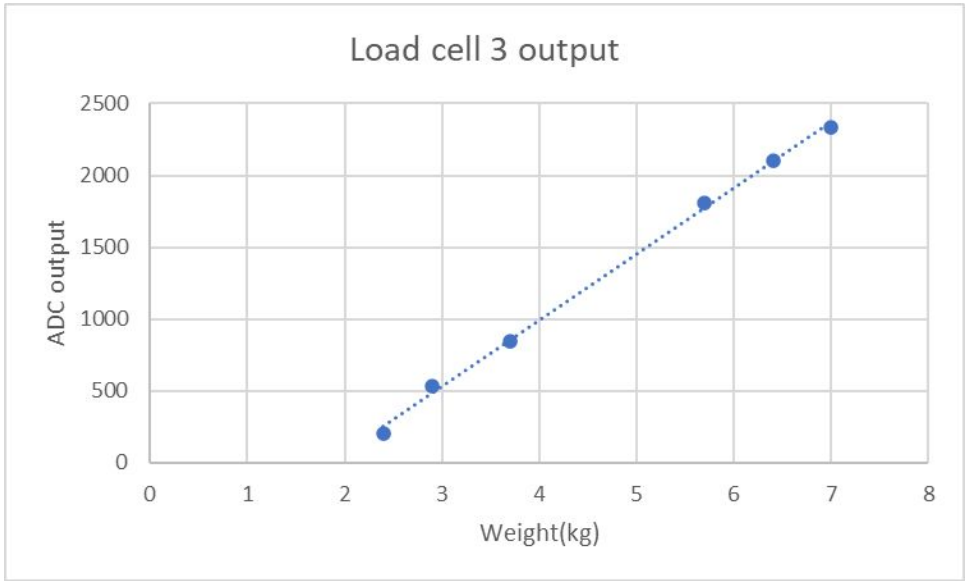


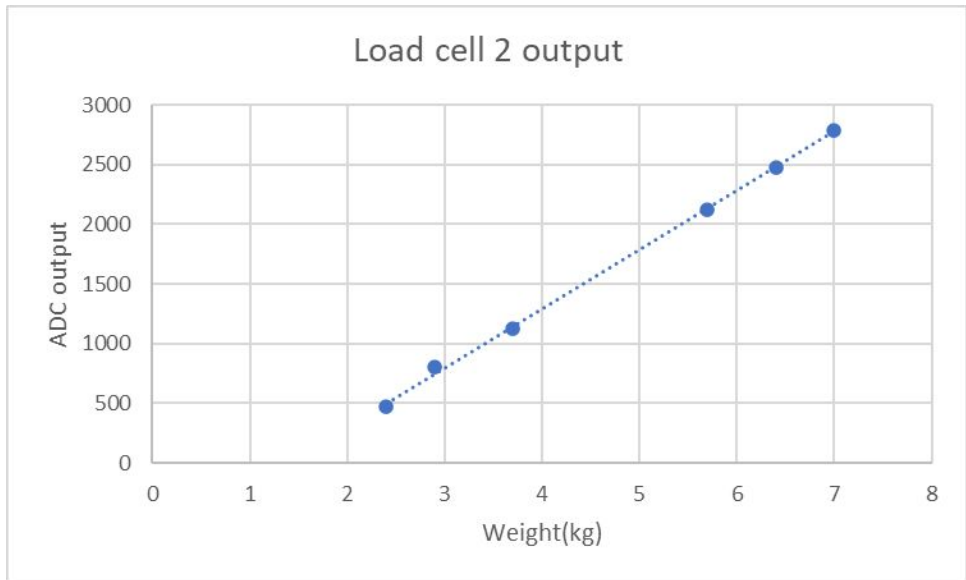
Weight sensor with 3 load cells tested







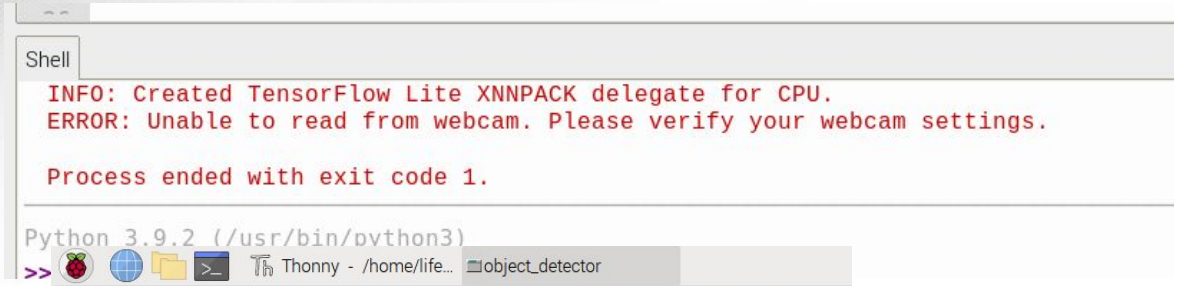


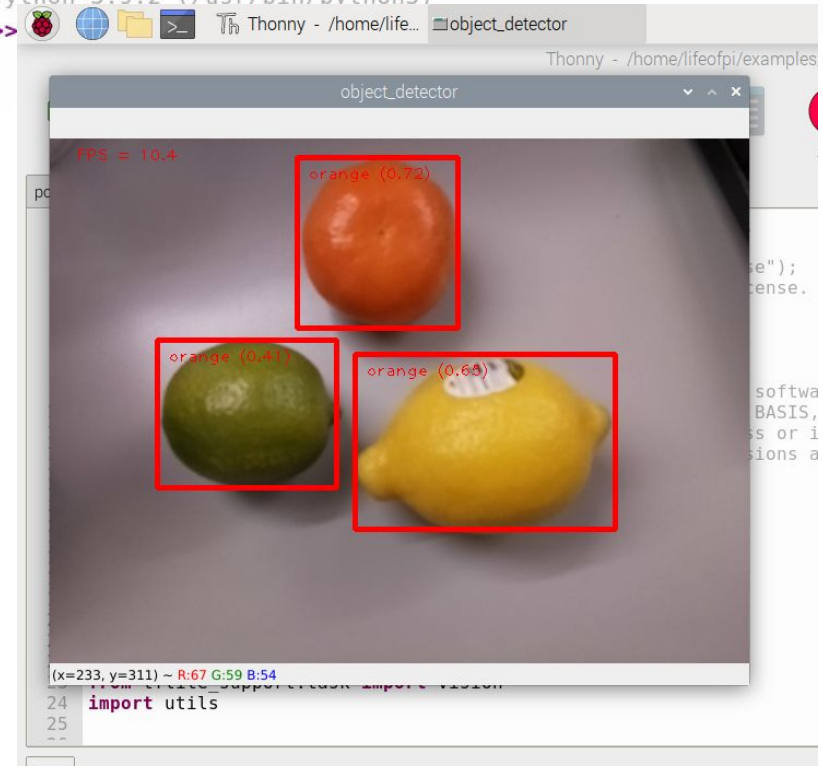


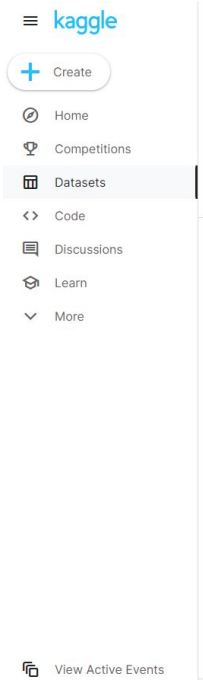
Shell

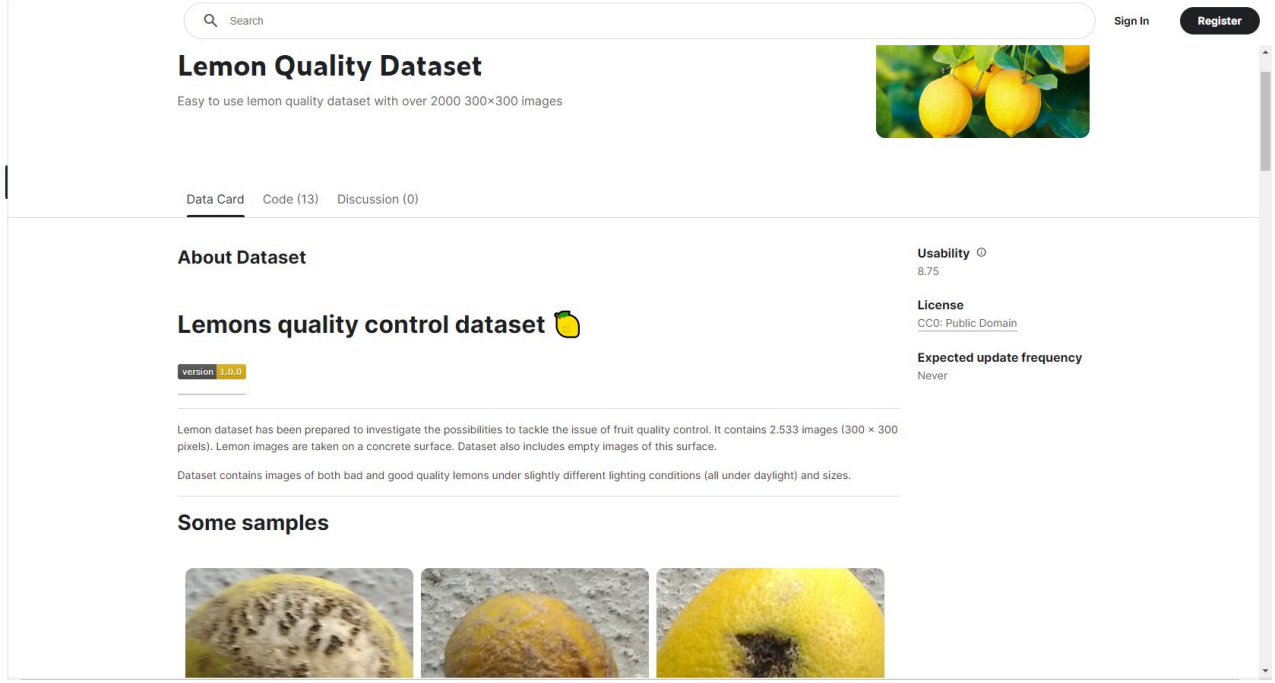
```
load cell 3 ADC volt:  0.9796874999999999 V
list of prev 12 data:  [1216, 1376, 1415, 1365, 1484, 1466, 0, 0, 0, 0, 0, 0]
the mean of the list:  693.5
For load cell 1
load cell 1 ADC volt:  0.6775634765624999 V
list of prev 12 data:  [841, 838, 835, 838, 838, 837, 833, 833, 846, 833, 833, 837]
the mean of the list:  836.8333333333334
For load cell 2
load cell 2 ADC volt:  0.201416015625 V
list of prev 12 data:  [250, 238, 250, 249, 248, 250, 250, 244, 240, 241, 245, 243]
the mean of the list:  245.66666666666666
For load cell 3
load cell 3 ADC volt:  0.9917724609374999 V
list of prev 12 data:  [1231, 1216, 1376, 1415, 1365, 1484, 1466, 0, 0, 0, 0, 0]
the mean of the list:  796.0833333333334
```

Progress on Machine Learning Model









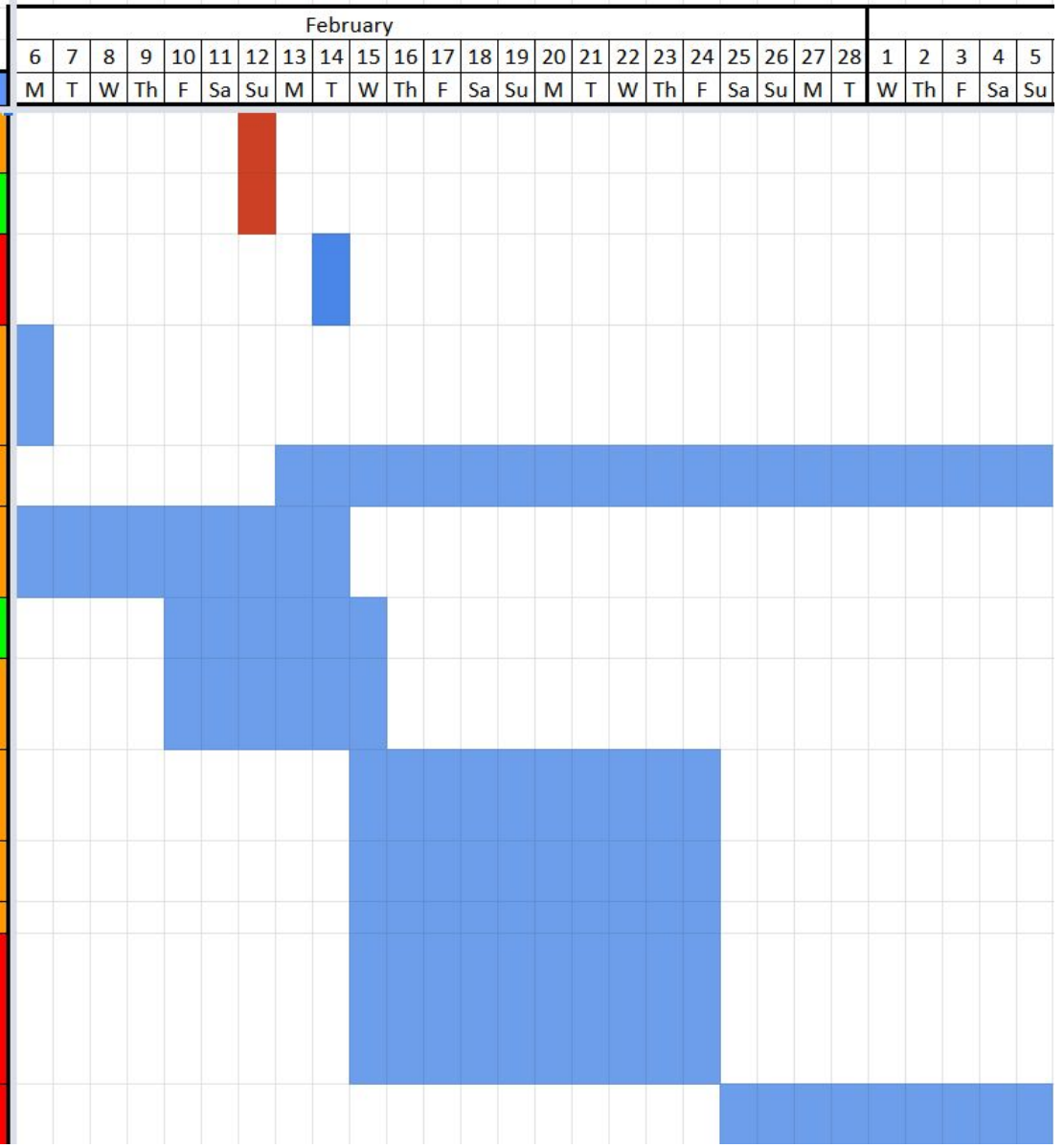
Execution Plan: Recent Progress

Robotic Sorting System

ECEN 403/404-901

Pace Dominy, Joseph Miller, Lam Tran

Event Description	Assigned to	Progress
Finish power control subsystem	**Pace**	
Finish sensor subsystem	**Lam**	
Implement fruit's scarring/disfigurement detection	Lam	
Install Bluetooth server setup on RSS RPi and test basic functionality	Joseph	
Build Conveyor Belt & Lever Arms	Pace	
Pass sensor data to Bluetooth server on Raspberry Pi	Joseph	
Test weight sensor code with 3 load cell	Lam	
Pass configuration information to robotic arms	Joseph	
Find/Collect dataset for machine learning model	Lam	
Find model/type for model	Lam	
Set up ML enviroment	Lam	
Functional test of Android application connections with sensor and robotic arms	Joseph	
Validate Conveyor Belt	Pace	



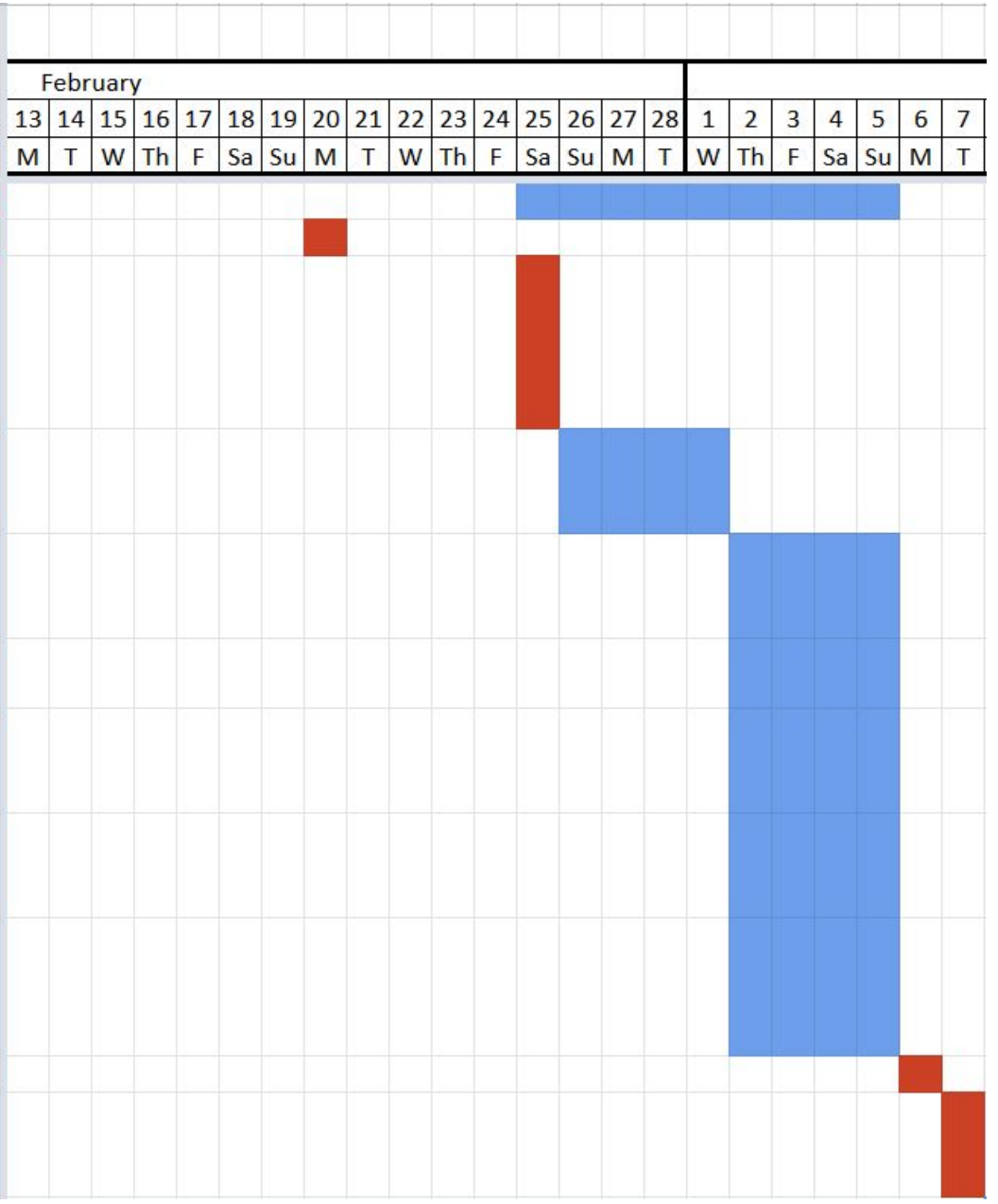
Execution Plan: Upcoming Plans

Robotic Sorting System

ECEN 403/404-901

Pace Dominy, Joseph Miller, Lam Tran

Event Description	Assigned to	Progress
Validate Lever Arm	Pace	
Status Update 3	**All**	
Finalize integration of Android app connections (with robotic arms and sensors)	**Joseph**	
Range and error testing for app/sensor interface	Joseph	
Finished building the machine learning model	Lam	
Train the model with the dataset	Lam	
Intergate the machine learning model to the Raspberry Pi	Lam	
Test and Validate the machine learning model	Lam	
Range and error testing for app/robotic arm interface	Joseph	
Status Update 4	**All**	
Finish basic Robotic Sorting System functionality	**All**	



Dwight Look College of ENGINEERING
TEXAS A&M UNIVERSITY

Validation Plan (part 1)

ECEN 403							ECEN 404						
Subsystem	*Test Name*	*FSR Reference*	*Success Criteria*	*Methodology*	*Status*	*Responsibility*	*Subsystem*	*Test Name*	*FSR Reference*	*Success Criteria*	*Methodology*	*Status*	*Responsibility*
App	Boot Functionality	Subsystem 3.1, 3.2.1	A skeleton version of the app (few to no features) successfully boots on a physical Android device and does not crash.	The skeleton app will be created and compiled using Android Studio and any relevant tools. It will then be loaded onto a Galaxy Tab S6 Lite and test booted.	Complete: app boots properly	Joseph	App	Basic write test	Subsystem 3.2.4.1.1 Project 3.2.3.1	A configuration value chosen by the user using the Android application will be written to the Bluetooth server. That value can then be read by the Android app.	User puts "6 (RPO)" into Sort by Size screen input box. The terminal on the Raspberry Pi will show that the number corresponding to a 6-inch sorting cutoff was written to the device. The user will then navigate to the "Diagnostics and Machine Information" screen, which will show a written sorting configuration of 6 inches.	Complete: wrote 6, then read 6 in the Diagnostics screen	Joseph
App	Specification Sorting	Subsystem 3.2.4.1.1	The app successfully allows the user to configure and store two "bin conditions". Attempting to set a value outside of the sensors' abilities returns an error.	The tester will attempt to store two "normal" (within bounds) bins with a cutoff at 30 cm diameter. They will then attempt to classify a (first) cutoff with a fruit size of 0, which should fail. They will then try to store a fruit with a size greater than the maximum sensor area (TBD), which should fail.	Complete	Joseph	App	Basic read test	Subsystem 3.2.4.1.2 Project 3.2.3.4	The application will read the configuration and weight values from the Bluetooth server and display those values on the Diagnostics and Machine Information screen.	A "default" test value of 56 (RPO) for the weight characteristic and 12 (RPO) for the write characteristic will be configured on the Bluetooth server. The user will launch the application, connect to the Bluetooth server, and read the "default" test values.	Complete: read 56 for weight and 12 for configuration	Joseph
App	Bluetooth Connection	Subsystem 3.2.4.2	An Android device successfully connects to the Bluetooth-enabled microcontroller.	The microcontroller will be turned on and set to pair with a Bluetooth device. The Galaxy Tab will then be used to pair with the device. The device will be left for two minutes to ensure that the connection does not drop. The tablet will then be disconnected and reconnected to ensure that the microcontroller remains available for connection.	Complete: connection successfully made and maintained for 2', disconnect and reconnect successfully completed	Joseph	App	RSS RPi integration test	Subsystem 3.2.4.1.1, 3.2.4.1.2 Project 3.2.3.1, 3.2.3.4	The results from the Basic Write Test and Basic Read Test will be successfully replicated on the RSS Raspberry Pi.	The Basic Write Test and Basic Read Test will be repeated using the actual Raspberry Pi that will be attached to the Robotic Sorting System.	Semi-complete: functional on RSS system image	Joseph
App	Bluetooth Configuration	Subsystem 3.2.4.1.1, 3.2.4.2	The app succesfully passes a sorting configuration to the microcontroller.	The sorting configuration from the Specification Sorting test will be saved in the app and sent to the Bluetooth-enabled microprocessor aboard the physical Robotic Sorting System. The output of the processor will then be examined to determine if the configuration was stored.	Complete	Joseph	App	Faulty configuration protection	Subsystem 3.2.3.1, 3.2.5 Project 3.2.5	The faulty configuration values 30, 0, and -1 will not be written to the Bluetooth server and will result in a popup error.	The Sort by Size screen will be used to write a faulty value of 110 (RPO), then the Send Configuration to System button will be pressed. The user will be notified that this configuration is invalid and the value will not be written to the Bluetooth server. This will be repeated with a value of 0 and -1 (both RPO).	Complete	Joseph
Power	PCB Voltages	Subsystem 3.2.3.1	Input power is correct (120V at 15A), conveyor belt & guidng arms respective motor drivers receive correct voltage, ADC receives correct voltage, RasPi recevies correct voltage. Test efficiency and noise of all 4 buck converters.	PCB will be tested with a voltmeter to check for the correct max voltage. This test will be done for all power inputs and outputs to validate that all the converters work correctly.	Incomplete	Pace	Sensor	Intergating the sensors with the RPI and the RasPiCam	Subsystem 3.2.2.3, 3.2.2.4	The Raspberry Pi is able to use the RasPiCam to take a image, and use it to determine the size or color of the fruit.	Program the Raspberry Pi to take an image. Impledement the color/size sensors code. Test it with objects of varying size and colors.	Complete	Lam
Power	Raspbery Pi Power	Subsystem 3.2.3	Validate Raspberry Pi receives correct voltage and turns on. Also validate that raspberry pi camera turns on.	Voltages will be tested with a voltmeter to ensure the correct max voltage	Incomplete	Pace	Sensor	Fruit's scarring/disfigurement detection	ConOp 4.3 Subsystem 3.2.2.4.3	The Raspberry Pi is able to tell if the fruit have some scarrings or disfigurements. This will determines whether or not the fruit is fit to be sold to the market.	Using the measurements from the color/size sensor, the Raspberry Pi will determines if the fruit meets a certain standard . Test it will some fruits that are good and some that are bad.	Incomplete	Lam
Weight sensor	Weight measurement Validation	Subsystem 3.1, 3.2.1.1	Validate weight sensor is connected to the Raspberry Pi. Validate that the weight measurement is display. Validate that the measurement is accurate with a few degree of error.	Voltages from the load cell will be tested with a voltmeter. Check if the load cell is connected to the correct pinouts of the RasPi. Connect the RasPi to a computer. Check if measurement is displayed on the computer. Place object on the load cell and record the data. Weight the same object with a digit scale. Compare the two data. Repeat this proccess with different weighted objects. Adjust calibration or edit code if necessary.	Semi-complete(need to calibration the sensor with a solid frame so it displayed the weight in kilograms or pounds)	Lam	Lever Arm	Position Validation	Subsystem 3.2.1	Lever arm moves to correct position based off of the data from the sensors	Lever arm will be timed and the angle of the arm will be checked for accuracy. Lever arm must not only "aim" toward one of the channels but it must also aim toward the correct one based off of what fruit, and the quality of the fruit	Incomplete	Pace
Color sensor	Color sensing Validation	Subsystem 3.1, 3.2.1.3, 3.2.2.1	Validate that the camera(s) is connected to the Raspberry Pi. Validate that the hue value is display. Validate that the hue value match with the color.	Check if the camera is connected to the correct pinouts. Connect RasPi to a computer. Convert BGR value to HSV value. Check if HSV value is displayed on the computer. Place an object in front of the camera and record HSV value. With Gimp, type in the value and check if color from the Gimp matches with the color of the acutal object. Repeat the process with objects with different colors. Adjust calibration or edit code if necessary.	complete	Lam	App/Sensors	App/Sensor Integration	App Subsystem 3.2.1.4.2 Project 3.2.3.4	The weight value passed from the sensor to the Raspberry Pi is shown on the Diagnostic screen of the Android app.	The sensor will pass a value of 10 (RPO) to the Raspberry Pi, overwriting a previous value of 11 (RPO). The user will monitor the Weight section of the Diagnostic screen on the app to ensure that this change is reflected.	App subsystem-complete (recieves from Joseph's RPi)	Joseph/Lam

Validation Plan (part 2)

Weight sensor	Weight measurement Validation	Subsystem 3.1, 3.2.1.1	Validate weight sensor is connected to the Raspberry Pi. Validate that the weight measurement is display. Validate that the measurement is accurate with a few degree of error.	Voltages from the load cell will be tested with a voltmeter. Check if the load cell is connected to the correct pinouts of the RasPi. Connect the RasPi to a computer. Check if measurement is displayed on the computer. Place object on the load cell and record the data. Weight the same object with a digit scale. Compare the two data. Repeat this proccess with different weighted objects. Adjust calibration or edit code if necessary.	Semi-complete(need to calibration the sensor with a solid frame so it displayed the weight in kilograms or pounds)	Lam	Lever Arm	Position Validation	Subsystem 3.2.1	Lever arm moves to correct position based off of the data from the sensors	Lever arm will be timed and the angle of the arm will be checked for accuracy. Lever arm must not only "aim" toward one of the channels but it must also aim toward the correct one based off of what fruit, and the quality of the fruit	Incomplete	Pace
Color sensor	Color sensing Validation	Subsystem 3.1, 3.2.1.3, 3.2.2.1	Validate that the camera(s) is connected to the Raspberry Pi. Validate that the hue value is display. Validate that the hue value match with the color.	Check if the camera is connected to the correct pinouts. Connect RasPi to a computer. Convert BGR value to HSV value. Check if HSV value is displayed on the computer. Place an object in front of the camera and record HSV value. With Gimp, type in the value and check if color from the Gimp matches with the color of the acutal object. Repeat the process with objects with different colors. Adjust calibration or edit code if necessary.	complete	Lam	App/Sensors	App/Sensor Integration	App Subsystem 3.2.1.4.2 Project 3.2.3.4	The weight value passed from the sensor to the Raspberry Pi is shown on the Diagnostic screen of the Android app.	The sensor will pass a value of 10 (RPO) to the Raspberry Pi, overwriting a previous value of 11 (RPO). The user will monitor the Weight section of the Diagnostic screen on the app to ensure that this change is reflected.	App subsystem-complete (recieves from Joseph's RPi)	Joseph/Lam
							App/Lever Arm	App/Lever Arm Integration	App Subsystem 3.2.4.1.1 Project 3.2.3.1 Robotic Arm Subsystem 3.2.1.2	The robotic arm moves to Bin 1 based on the configuration passed in via the Sort by Size screen.	The application will pass a sorting cutoff of 1 (RPO) to the Raspberry Pi. The lever arm will then read this test value and move to guide fruit to Bin 1.	Incomplete	Joseph/Pace
Size sensor	Size Measurement Validation	Subsystem 3.1, 3.2.1.3, 3.2.2.1	Validate that the camera(s) is connected to the Raspberry Pi. Validate that the size measurement is display. Validate that the size measurement is accurate with a few degree of error.	Check if the camera is connected to the correct pinouts. Connect RasPi to a computer. Check if measurement is displayed on the computer. Place an object in front of the camera and record height and witdh. Measure the height and width of the same object with a ruler, tapemeasurer, caliper or whatever. Compare the two data. Repeat this proccess with different size objects. Adjust calibration or edit code if necessary.	complete	Lam	Lever Arm & Conveyor Belt	Timing Validation	Subsystem 3.2.1	Lever arm moves to correct position and stays there. Conveyor belt then turns on and lever arm does not move until it is required to do so again	Lever arm will be told a specific fruit and quality. From there, the timing will be checked to make sure that the subsystem waits for the information before moving. The lever arm will also be validated as staying at the correct position for the amount of time needed to move the fruit to the correct channel on the conveyor belt.	Incomplete	Pace
							Lever Arm/Sensors	Lever Arm/Sensors Intergration	Project 3.2.3.4.3, Sensor Subsystem 3.2.2.4.1	Lever arm moves a certain the position based off of the sensor output.	The sensor will determine the color/size of the fruit. The lever arm will moves to a certain channel based on the output from the sensor.	Incomplete	Pace/Lam

Converyor Belt/Sensors	Converyor Belt/Sensors Integration	Project 3.2.2.2, Sensor Subsystem 3.2.2.1	The camera is postion to take clear and consistent images. The weight sensor will be calibrated when it is attact to converory belt system and holding the bins.	When the converyor belt system is active, the camera should be stable and in a fixed postion. Maybe recorded a video from the camera to see if it has little to no shaking. Maybe test the color/size sensor to see if it produce consistent output. For the weight sensor, see how much voltage is producing depending how much force is being apply. Add some counter weights to have the voltage to be zero. Find the linearly relationship of it. Output a weight unit, either lbs or kg.	Incomplete	Pace/Lam
App/Conveyor Belt	Belt Start/Stop	Project 3.2.3.1	The app passes a command to stop and start the belt.	A test function will be added to the app that allows the app user to directly stop and start the belt. This system will then be tested to ensure that stopping/starting the belt in the app results in the appropriate action on the conveyor belt.	App subsystem-complete (app sends belt start/stop bit)	Joseph
App/Sensors	Sensor Error Testing	Project 3.2.5.1.3	The sensor indicates a fault, causing an error message on the Diagnostic screen.	The sensor will pass a FAULT value to the Raspberry PI, which will be sent via Bluetooth to the Android app. The Diagnostic screen will show FAULT in the Weight category.	Incomplete	Joseph/Lam
App/Conveyor Belt	Belt Movement Error	Project 3.2.5.1.1, 3.2.5.1.2	The app returns an error if the belt is obstructed or the power to the machine is cut off (stopping the belt).	The belt will be set to run, then the power will be disconnected at the wall. The test will be successful if the app returns a notification that the belt has stopped.	Incomplete	Joseph
App/System	Full Functionality	App Subsystem: All Project: 3.2.1.4, 3.2.3.1, 3.2.3.4.1, 3.2.3.4.2, 3.2.5	The app passes a configuration to the machine and starts the belt. When sorting is complete, the app stops the belt and returns a notification.	The sorting configuration from the Specification Sorting test will be passed to the sorting machine via the app. Two fruits (one for each bin) will be scanned on the belt and sorted. When 15 seconds have elapsed since the last fruit was scanned, the belt will stop and push a notification to the app.	Subsystem-complete (need to build RSS to connect to)	Joseph
Power	PCB Voltages	Subsystem 3.2.3.1	Input power is correct (120V at 15A), conveyor belt & guidng arms respective motor drivers receive correct voltage, ADC receives correct voltage, RasPi receives correct voltage. Test efficiency and noise of all 4 buck converters.	PCB will be tested with a voltmeter to check for the correct max voltage. This test will be done for all power inputs and outputs to validate that all the converters work correctly.	Incomplete	Pace
Power	Raspberry Pi Power	Subsystem 3.2.3	Validate Raspberry Pi receives correct voltage and turns on. Also validate that raspberry pi camera turns on	Voltages will be tested with a voltmeter to ensure the correct max voltage	Incomplete	Pace

Thank you!