Robot Design
Animal Allies

Machines Inc., Team #5041

Team Members

- Rohan Bommaraju aka Mr. Awesome, grade 7, likes football, karate, basketball, chess, friends, LEGO. Favorite phrase: "Touchdown"
- Amrik Eadara aka Cam!, grade 7, likes math, writing, technology, robotics, basketball, football, baseball, karate, chess, legos. Favorite phrase: "Rush threw, redo"
- Anish Patel aka Stud Muffin, grade 7, likes math, science, technology, programming, robotics, video games, baseball, karate. Favorite phrase: "In a minute!"
- Vidhur aka Kid Genius, grade 6, likes hockey, soccer, football, EV3. Favorite phrase: "um uh um uh…"
- Noah aka Dr. EZ, grade 7, likes

Robot Game Strategy

Mission No.	Mission Name	Mission Execution Order	Pts
0	Pre-Start	Deer in animal conservation Calibration	0 or 20
1	Refrigerator	Get food from refrigerator	0
2	SharkCow	Shark tank pushed to left box; touching only tank floor Service dog mission Push cow Get bee Get pig to base; put prosthesis on pig	27 + 15 + 0 + 0 + 15= **57**
3	Food Drop Off	Feed gorilla Feed bat Feed bird Feed frogs Gecko on wall Zoologist in training and research area Panda release	20 + 20 + 20 + 20 + 15 + 15 + 10 - 6= **114**
4	Milking Automation	Milking automation Dog and Trainer in box + 1 manure 2 manures, 1 at a time	20 + 17 + 10 = **47**
	THE END		**Possible Score: 218 or 238**

What We learned

- We found a bug with wait block. When robot moving forward and looking to detect reflective intensity, white or black, the robot must call the stop block even if it is looking to move forward certain distance after reflective intensity detection. If it does not stop, the rest of the program will work inconsistently.

- We learned about trigonometry, SOH-CAH-TOA, right triangles, and pythagorean theorem. We want to first rotate and then go straight for certain distance. After rotating certain degrees, if the rotation is off by 1 degree, the robot will miss its target after going straight by more than half inch. We calculated the target offset by using the formula: distance * tangent(actual degrees - target degrees). For example, if we want to turn 90 degrees and go straight 30 inches, the error will be 0.5 inches if robot turns 91 degrees. Formula: 30 * tangent (91 - 90) = 0.524 inches

- When programming, we didn't know how to **follow a line** with one color sensor. We did lots of research, read lots of books, and programmed tests to learn about line following. After many tries, we looked at an interesting book, "Winning Designs!", and learned about using multiple sensors and different programs for line following. At the end, we decided to use the right sensor and proportional line follower for line following for accuracy and to free up the left color sensor for other tasks.

- We also learned that when you use the **gyro for turning**, it is not very accurate and always overshoots. We also learned the robot has to wait for gyro to catch up. But then, we noticed that when the robot goes slow it is very precise. Therefore, we made a block that makes the robot go at a very high speed at first. This will cause robot to overshoot but then it slowly adjusts to desired value after gyro catch up. Wait for 0.25 seconds.

- We thought we could also use the **gyro to go straight** because when you use the move steering to go straight, the robot drifts to the right. We call this block the "gLineBlock". It is the motor value plus or minus the error. The error is the actual (0°, robot going straight) minus the sensor value times the gain. The gain is the reaction time. This formula makes the robot go straight using gyro and move steering block.

- Touch sensor - We thought that using touch sensors would make the robot be more accurate. We figured that that you could make the **touch sensor line up with the wall**. If you use the touch sensor, it would be more accurate.

What We learned (cont.)

- **Food dispenser block** - Automatically dispenses the food that was put in the dispenser at the beginning. In the block, you can input the amount of times each food is pushed out. Solution is a simple medium motor rotation back and forth.
- **Line Align Block** - Using line align would make the robot straight because if you have two color sensors that lines up with a straight line, the robot will be precise. We made it so that if the left sensor senses the line before the right sensor does, the right wheel moves until the right sensor does. It is the same thing with the other side. If the right sensor senses the line before the left sensor does, the left wheel moves until the left sensor does.
- Using combination of sensors, wall and mat lines to position and navigate the robot – angles and parallel lines
- **Gyro Display Block** – We noticed that the gyro kept drifting. Therefore, we made a **drift detection program**. It shows the display for the gyro on the screen of the robot. If the gyro isn't equal to 0° (not straight), it makes a beeping noise. If the gyro drifts, the robot would not go straight because we have another block that makes the robot go straight using the gyro.
- **Color Calibration Block**– This block calibrates the 2 color sensors. When the robot is in a different room or the robot is just turned on, you would do the calibration. Before we created this, the robot didn't see the line. To calibrate the robot, you just have to put the color sensor over the color the robot says. For example, If the robot says "White", You would put the color sensor over white, but if the robot says "black", you would put the color sensors on the black line.
- **Sequencer Block** – This program sequences all the programs in mission order. You would press the right button to go to the next program and you would press the left button to go to the previous program. And the sequence loops when incrementing at max value or decrementing at minimum value.

MyBlocks

- **gTurnBlock** – Uses gyro feedback for accurate wide turns and uses only one wheel to turn. Block inputs: left or right turn, degrees to turn. The formula used for this block is: Turn right or left at specified power for specified degree. This will result in the robot overshooting. Wait for a 0.25 seconds delay for the gyro to catch up. After that, the robot slowly adjusts, until specified degrees.

- **Proportional Line Follower** - Uses the right color sensor to follow a black line. If the right color sensor senses more white than black, the robot turns left and when the color sensor senses more black than white, the robot turns right. The formula is the power - (reflected light intensity ± inputted reflected intensity * (gain -> 0.2)) = power of motor B/power of motor c (that is what the ± is for).

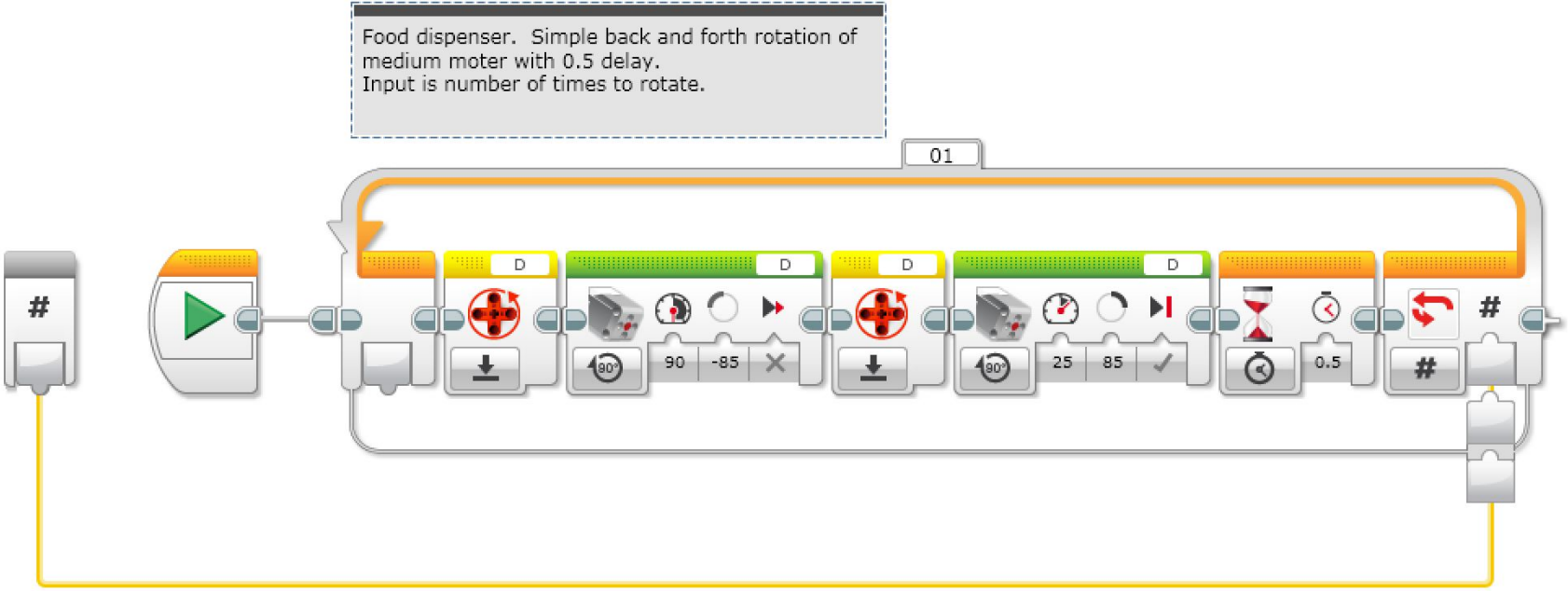
- **gSpinBlock** – This block uses gyro feedback to make short or spin turns while using both wheels turning in opposite directions. Block inputs: degrees to turn. The formula used for this block is: After the robot turns the specified amount of degrees, there is a 0.25 seconds delay. After that, the robot slowly adjusts, until specified degrees.

- **gLineBlock** – Uses gyro feedback to go straight. Robot automatically adjusts Power_A or Power_B through move steering if gyro ≠ 0. Inputs: length, power, wheel diameter. Using the formula: (length * 360) / wheel_circumference, the robot decides if the amount of inches is positive or negative. If it is positive, the formula is 0-gVal=power. If it is negative, the formula is gVal=power.

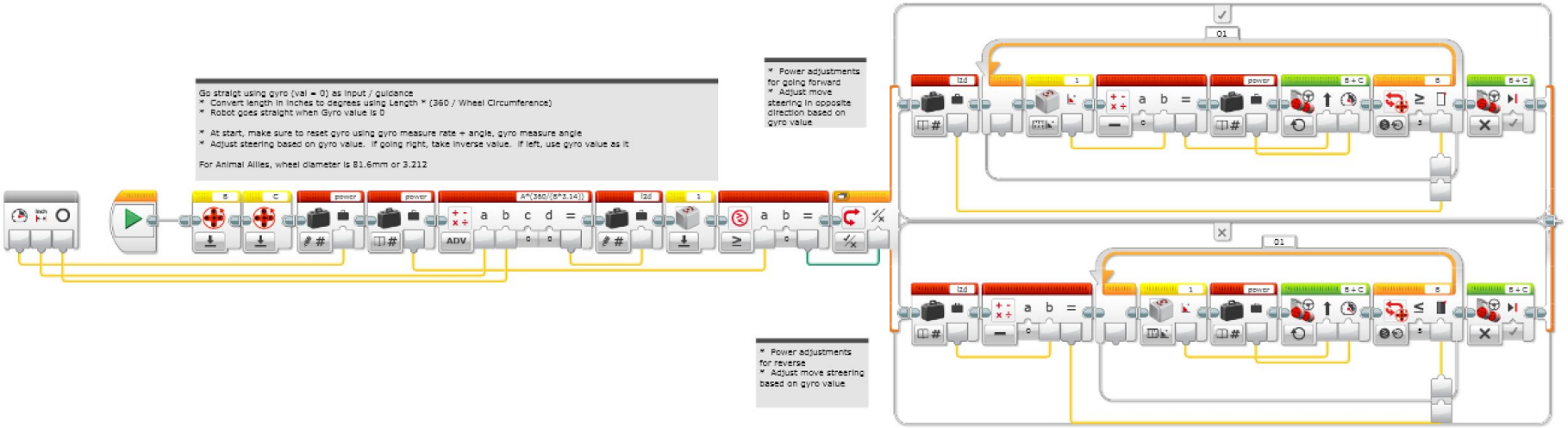
- **LineCounter** – Use right color sensor to count number of lines passed. Useful for navigation to get to specific zone area on field mat where distance between two points can vary. LineCounter is an alternative to measuring fixed distance in inches for navigation.

- **WallAlign** – This block aligns with the wall for a certain amount of specified time.

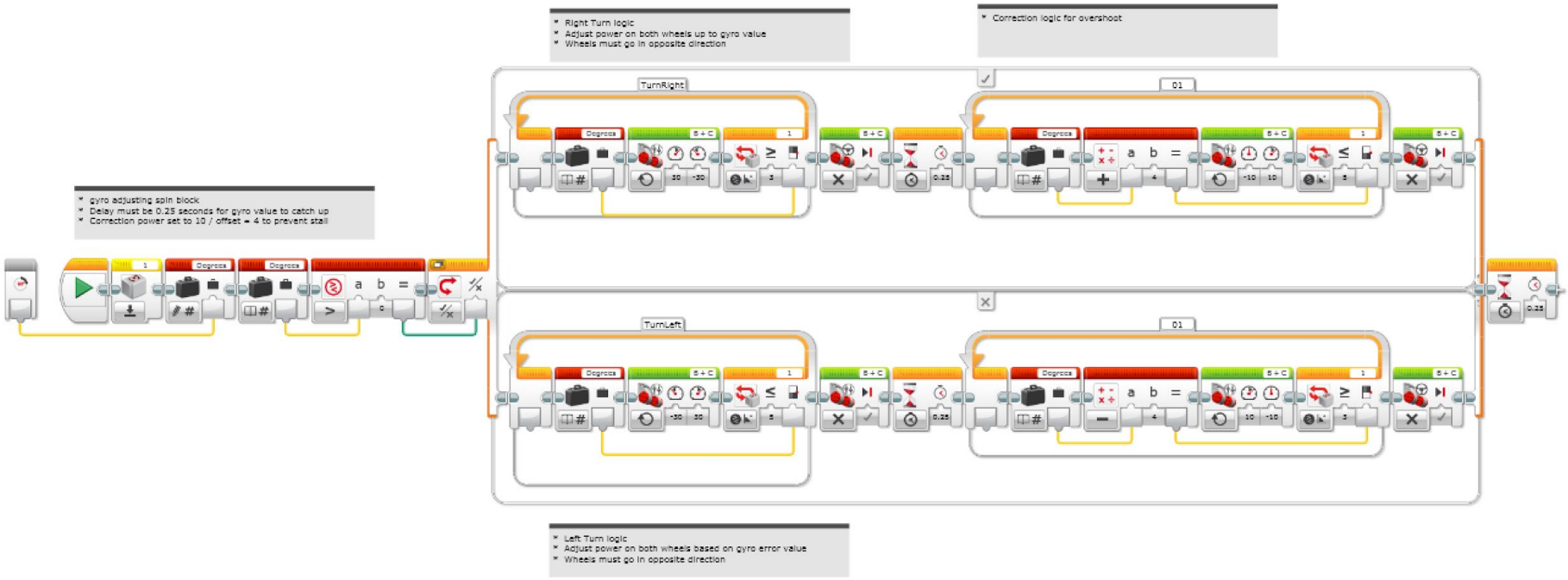
Food Dispenser Block



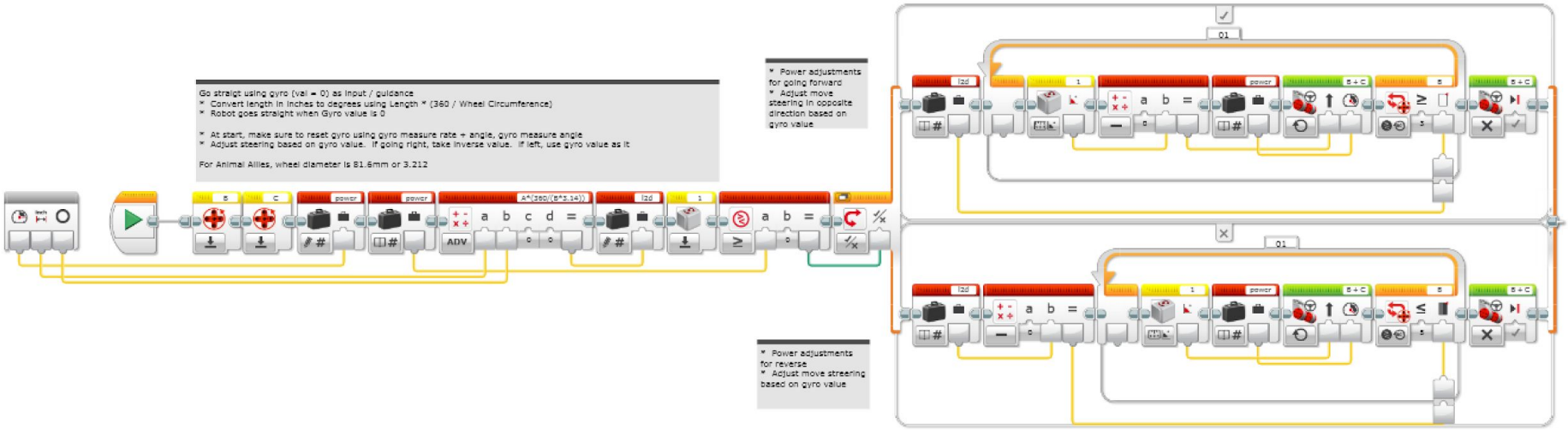
gTurnBlock Block



gSpinBlock Block



gLineBlock Block



Proportional Line Follower

Proportional line follower. Put this block in a loop.

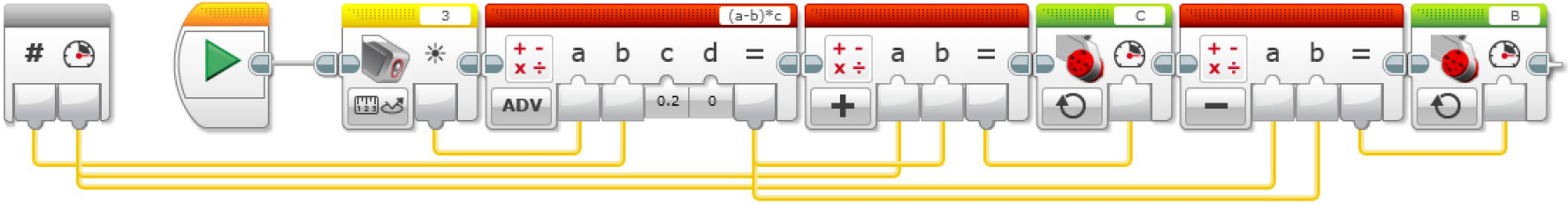
Inputs: midrange intensity; power value.

Using right color sensor; always position sensor on right line edge.

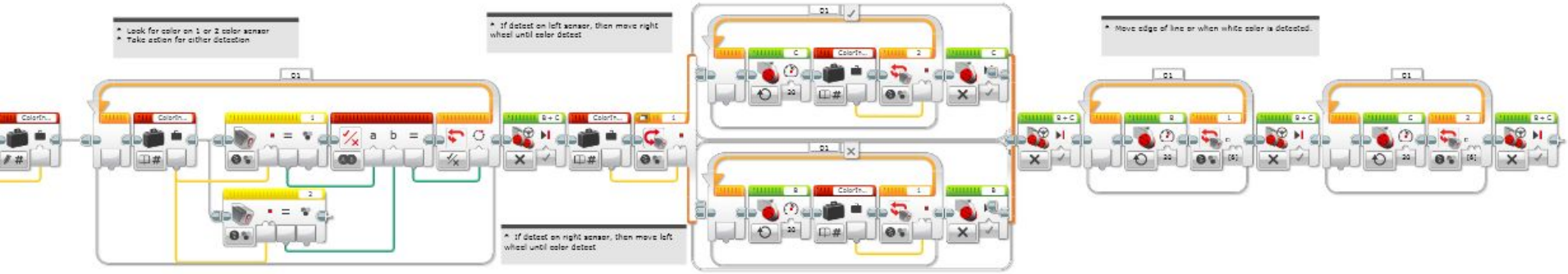
-1 * (Gain * (LightSensor- Desired Value))

Try Gain (C) to 0.25; Power = 30

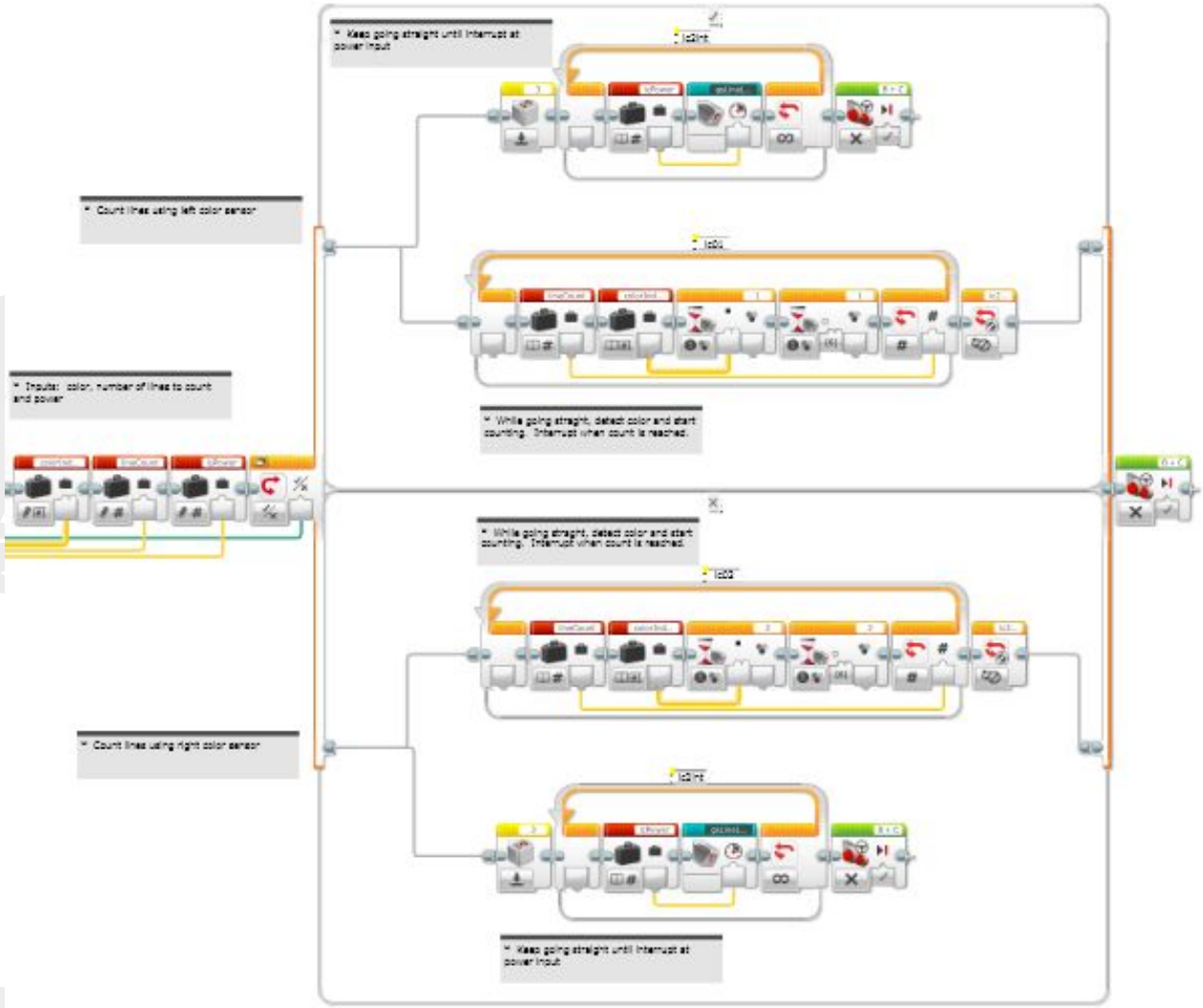
Input for black / white edge should be 50 after calibration



LineAlign Block



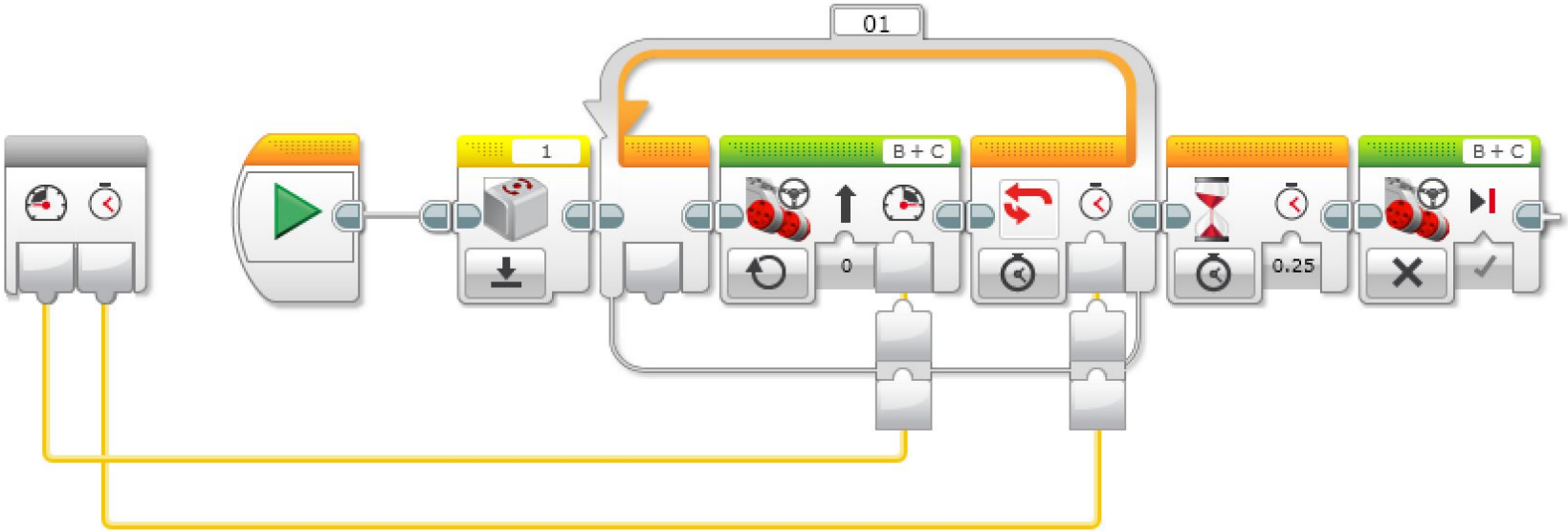
LineCounter Block



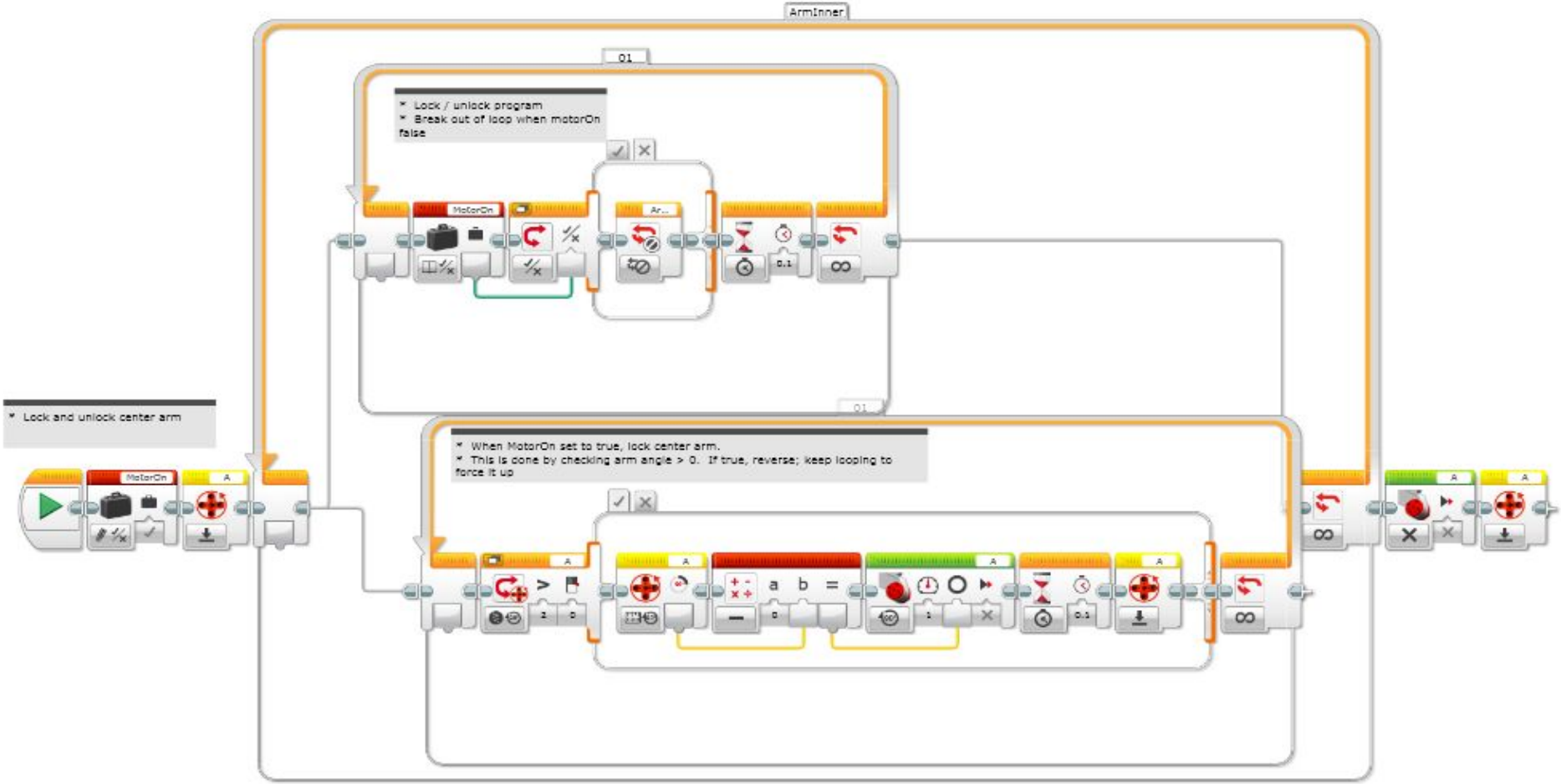
WallAlign Block

Wall Align based on timer, NOT touch sensor

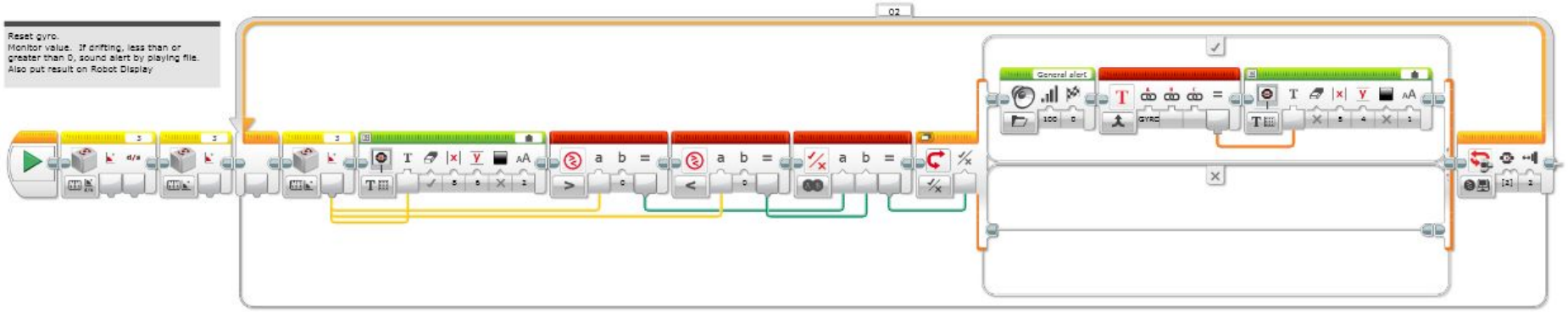
* Input power and delay
* Simply reverse until delay reached
* Delay to force alignment if robot is reversing at a slight angle



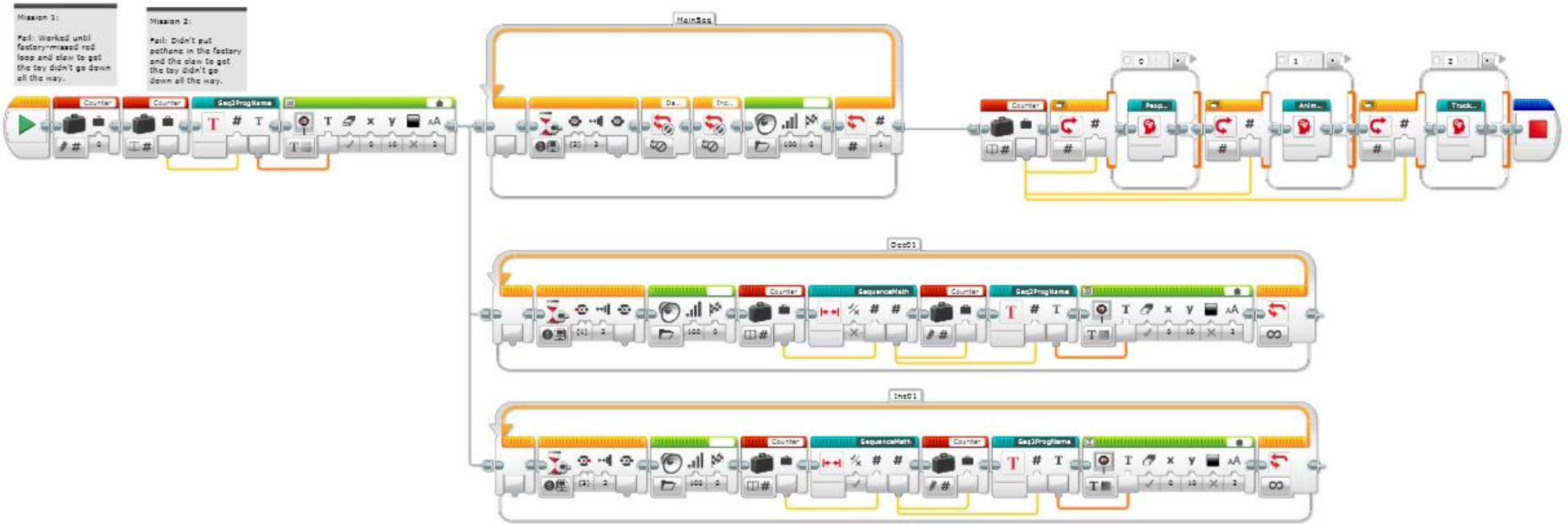
ArmLockBlock



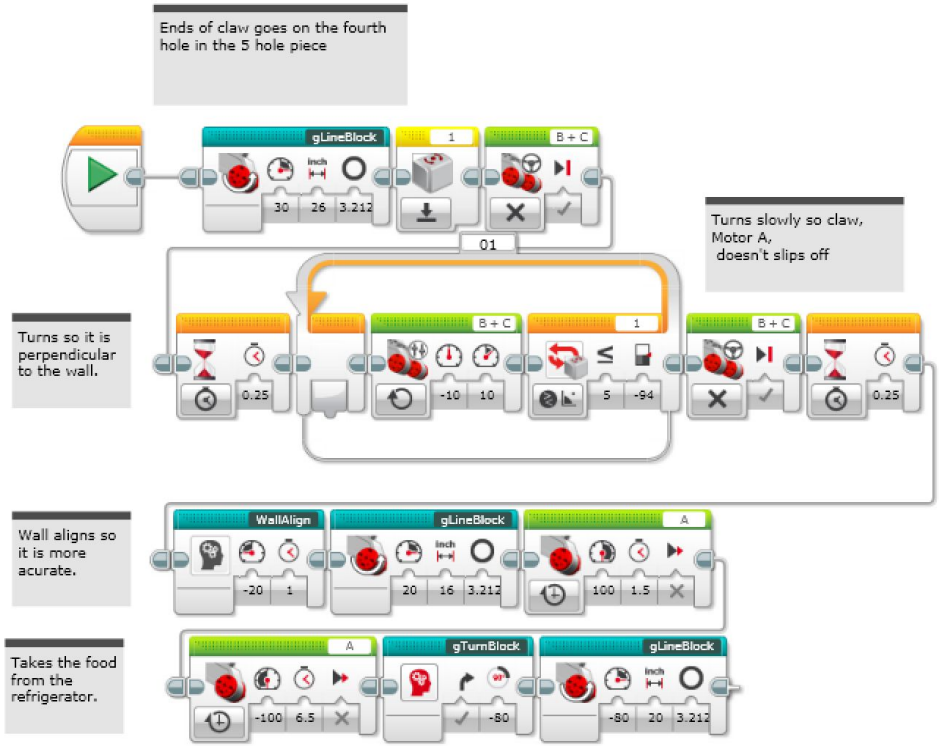
Gyro Display Block



Sequencer


Mission 1:
Fail: Worked until factory+missed rod loop and claw to get the toy didn't go down all the way.
Mission 2:
Fail: Didn't put pothane in the factory and the claw to get the toy didn't go down all the way.
MainSeq
Counter
Seq2ProgName
Pacp...
Anim...
Truck...
Doc01
SequenceMath
Inc01

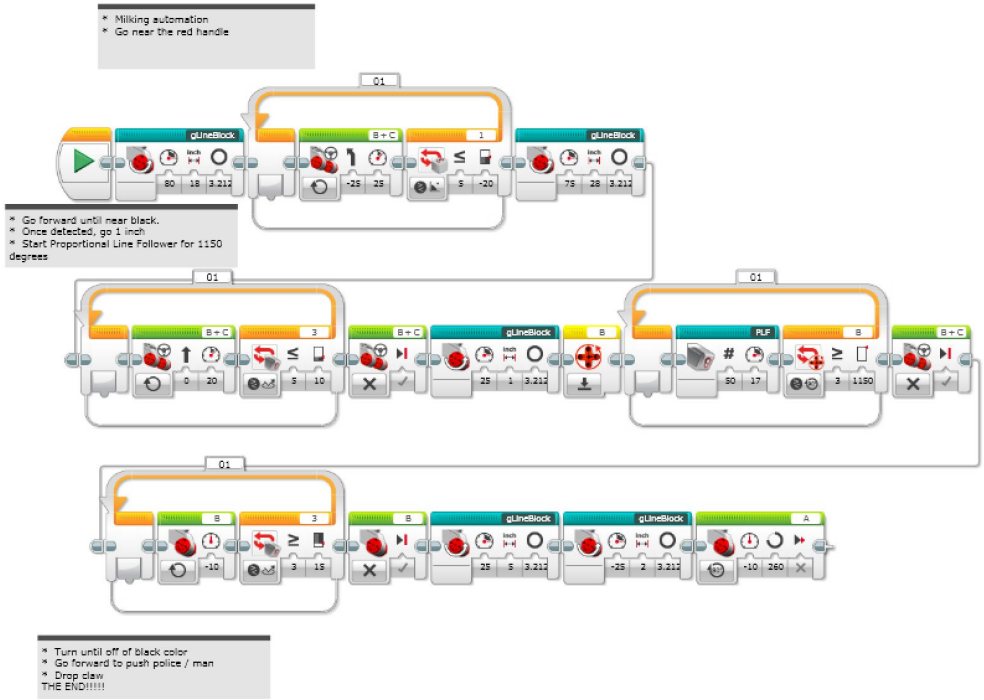
Mission 1: Refrigerator



Mission 2: Shark + Cow

Mission 3: Food Drop

Final Mission 4: Milking Automation



Score Sheets: Refrigerator

Run Number	Time	Pass / Fail	Failure Point	Fix	Points
1	19.58	Pass			0
2	19.51	Pass			0
3	19.94	Pass			0
4	19.11	Pass			0
5	18.89	Pass			0
6	19.17	Pass			0
7	19.66	Pass			0
8	19.59	Pass			0
9	19.87	Pass			0
10	18.83	Pass			0
AVG:	**19.415 sec**				**0 Pts**

Score Sheets: SharkCow

Run Number	Time (Seconds)	Pass / Fail	Failure Point	Fix	Points
1	37.73	Pass			57
2	36.52	Pass			57
3	36.58	Pass			57
4	36.12	Pass			57
5	36.13	Pass			57
6	36.54	Pass			57
7	31.23	Pass			57
8	32.52	Pass			57
9	36.34	Pass			57
10	36.12	Pass			57
AVG:	**35.583 sec**				**57**

Score Sheets: Food Drop

Run Number	Time (Seconds)	Pass / Fail	Failure Point	Fix	Points
1	75	Pass			89
2	72	Pass			89
3	72	Pass			89
4	73	Pass			89
5	67	Pass			89
6	76	Pass			89
7	72	Pass			89
8	75	Pass			89
9	75	Pass			89
10	74	Pass			89
AVG	**73.10**				**89**

Score Sheets: Milking Automation

Run Number	Time (Seconds)	Pass / Fail	Failure Point	Fix	Points
1	15	Pass			47
2	16	Pass			47
3	16	Fail	Crashed into cow	Retry	30
4	17	Pass			47
5	17	Pass			47
6	17	Pass			47
7	16	Pass			47
8	16	Pass			47
9	16	Pass			47
10	16	Pass			47
AVG	**16.20**				**47**