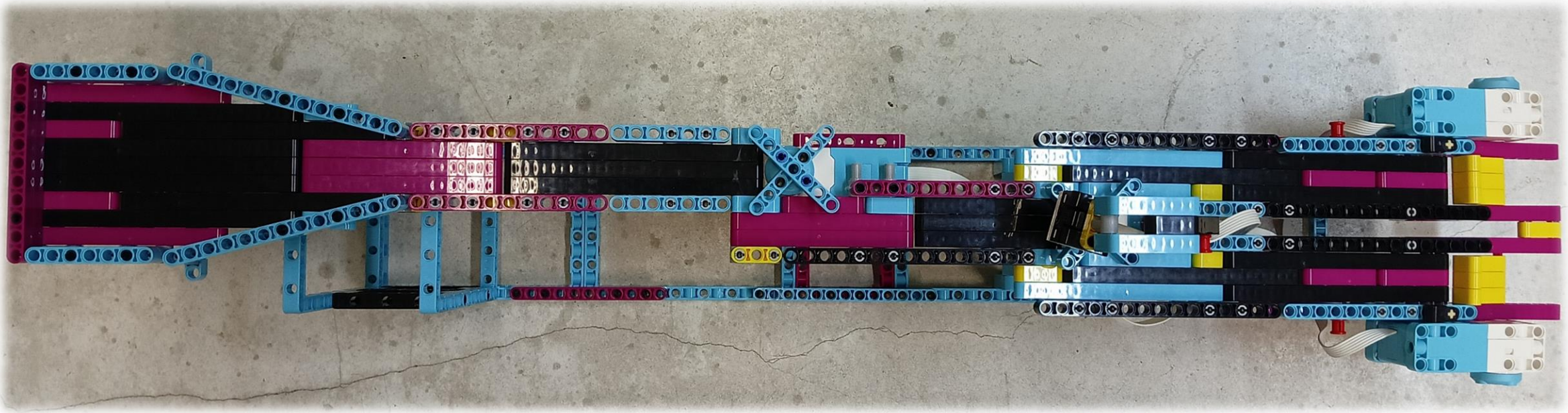
Sort Line

By:
Name: Pawan Kumar Shakya
Email: pawankumar.shakya@uni-hohenheim.de
Study: MSc. Food Systems
Matriculation number: 993265

Aim: To sort the choco ball based on color. Red and Gold color ball to left and Green and Blue color ball to right



Necessary Import:

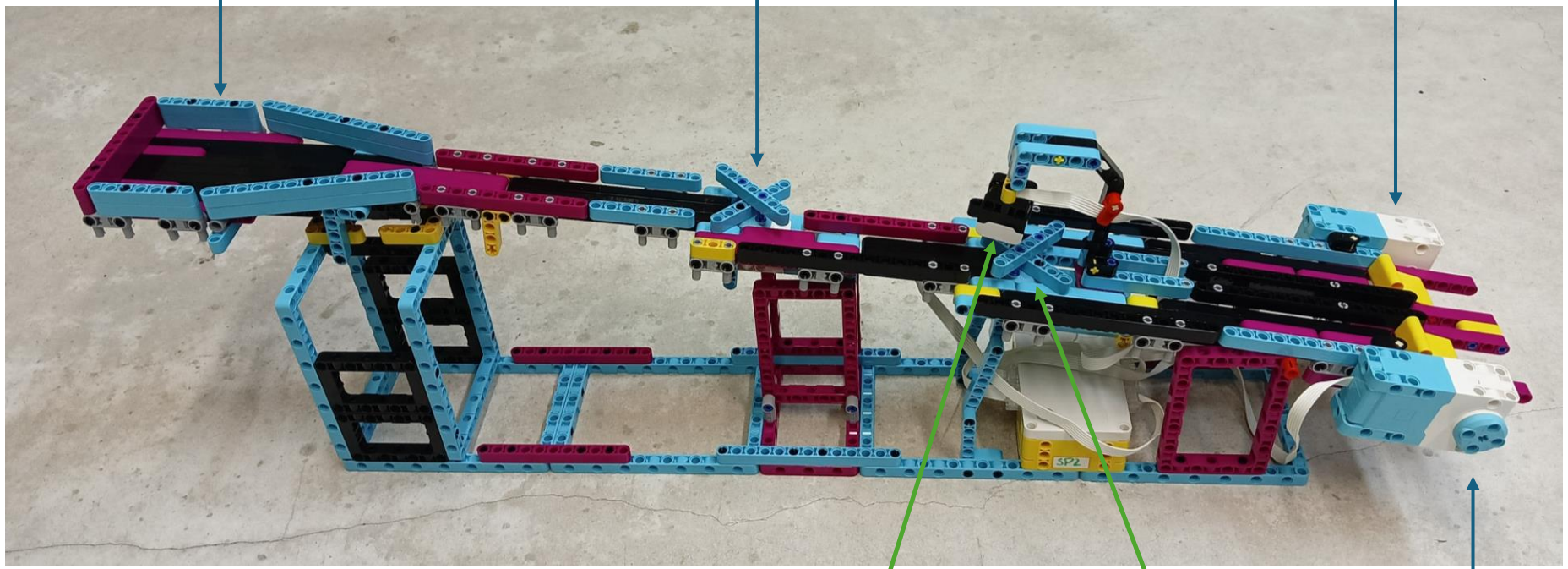
```
1 from hub import port
2 import runloop
3 import motor
4 import color_sensor
5 import color
6 import color_matrix
```

Assigning variables

```
#object assign
lock_gate = (port.A) # open first gate to pass choco-ball one by one - MOTOR
sort_gate = (port.B) # sorts (turn right or left based on color detection condition) - MOTOR
color_detect = (port.C) # detects the color in case of gold it return rgbi (function from API)
rear_gate_right = (port.D) # turn to dump the choco balls in container - RIGHT MOTOR
rear_gate_left = (port.E)  # turn to dump the choco balls in container - LEFT MOTOR
```

- In total there are five objects.
- Four motors and one color detector
- All the object are connected to different ports of hub

Architecture and main objects:



Color Detecting Functions:

```
# to check the red color by inbuilt function
def is_red():
    if color_sensor.color(color_detect) is color.RED:
        return True
    else:
        return False
# to check the green color by inbuilt function
def is_green():
    if color_sensor.color(color_detect) is color.GREEN:
        return True
    else:
        return False

# to check the blue color by inbuilt function
def is_blue():
    if color_sensor.color(color_detect) is color.BLUE:
        return True
    else:
        return False
```

```
# functions
# to check the gold color but by recognising the pattern from returned set of tuples
# Check if the first element is greater than the second and
# if both, the first and second elements are greater than the third
def is_gold():
    gold_int = color_sensor.rgbi(color_detect)
    if gold_int[0] > gold_int[1] and gold_int[0] > gold_int[2] and gold_int[1] > gold_int[2]:
        return True
    else:
        return False
```

Exception: Since color detector can not Directly detect gold color so idea is to used “rgbi” In-built function to understand pattern in tuple Returned by function and apply conditional Statement. In this case the pattern was first Element was greater then second and second was Greater than third

Setting default position for all four motors:

```
# function to set the position of rear motors to ideal position
# 10 degree elevation is there, to stop the choco balls to stop
async def set_positions_rear():
    await motor.run_to_absolute_position(rear_gate_right, -10, 360)
    await motor.run_to_absolute_position(rear_gate_left, 10, 360)


# function to set the position of front motors to ideal
async def set_positions_front():
    await motor.run_to_absolute_position(lock_gate, 45, 360)
    await motor.run_to_absolute_position(sort_gate, 45, 360)
```

Note: the rear motor were arranged in opposite direction that why the elevation angle is same in interger but Difference in sense of (+) (-) but in human visual it looks at a similar elevation.

45 degree angle was chosen before at this angle the choco ball were fitting better at the gate.

Ball release and sort motor:

```
# function to open the lock gate which allow to balls pass one by one by the interval of 2 seconds
async def open_line():
    await motor.run_for_degrees(lock_gate, -90, 180)
    await runloop.sleep_ms(2000)

# function to rotate the sort motor to right if the color detection condition is met from the main function
async def sort_line_right():
    await motor.run_for_degrees(sort_gate, 180, 360)
    await runloop.sleep_ms(2000)

    # function to rotate the sort motor to left if the color detection condition is met from the main function
async def sort_line_left():
    await motor.run_for_degrees(sort_gate, -180, 360)
    await runloop.sleep_ms(2000)
    return True
```

Note, time is in in milliseconds.
Here, 2 second of time interval has been taken to ensure the ball is pass by and reaching under color Sensor before another ball is released. Further based on color the sort motor is rotate right or left. In our case, color of Ball is detected red or gold then it is diverted towards left rear gate line and in case of green or blue it is diverted towards Right rear gate line,

Releasing the balls into container:

```
# function to release the ball from rear right side to dump the ball into container
async def dump_right():
    await motor.run_for_degrees(rear_gate_right, -90, 180)
    await runloop.sleep_ms(1000)
    await motor.run_to_absolute_position(rear_gate_right, -10, 360)

# function to release the ball from rear left side to dump the ball into container
async def dump_left():
    await motor.run_for_degrees(rear_gate_left, 90, 180)
    await runloop.sleep_ms(1000)
    await motor.run_to_absolute_position(rear_gate_left, 10, 360)
```

As said earlier the motor in the back are fitted to model in opposite direction so the commonds to make rotation is in the same degree but with different rotation sign + & -.
Here, the dump gate is kept up for 1 second because that's the time required for 3 balls to pass into container.

Main Function:

```
async def run_it():
    n = 0 # a number initialed to denote number of object under color sensor line is zero
    count_right = 0 # setting intial right rear gate count to zero
    count_left = 0# setting intial left rear gate count to zero
    await set_positions_front() # setting the front motor to ideal state before operations begins
    await set_positions_rear() # setting the rear motor to ideal state before operations begins
    await runloop.sleep_ms(3000) # now gate waits to ball to reach and line up
    await motor.run_for_degrees(lock_gate, -90, 180) # loading the first ball from incoming line to release to color detector
    while True: # an endless loop
        runloop.run(open_line()) # once the first ball is realised to color detector
        n = n + 1# number of ball under color detector is increased by one
        if is_blue() or is_green() and not is_black(): # check if the color is blue or green and not any other color especially black
            count_right = count_right + 1# increase the count by one to right side rear gate
            runloop.run(sort_line_right())# sort motor turn right and sends the ball to right rear gate line
            n = n - 1# one the ball has been directed to respective color line, resulting into no ball under color detector is getting back to zero
            if count_right == 3:            #once 3 balls has been collected
                runloop.run(dump_right())# right rear gate opens and dump the 3 ball into container
                count_right = 0            # then re-set the count to zero (to main a count for upcoming ball towards right rear gate)

        elif is_red() or is_gold() and not is_black(): # check if the color is red or gold and not any other color especially black
            count_left = count_left + 1                # increase the count by one to left side rear gate
            runloop.run(sort_line_left())            # sort motor turn left and sends the ball to left rear gate line
            n = n - 1    # one the ball has been directed to respective color line, resulting into no ball under color detector is getting back to zero
            if count_left == 3:                         #once 3 balls has been collected
                runloop.run(dump_left())            # left rear gate opens and dump the 3 ball into container
                count_left = 0                          # then re-set the count to zero (to main a count for upcoming ball towards right rear gate)

runloop.run(run_it())# to run main function to start operation
```

Note: Step-by-step explanation on next slide

Setting default positions:

```
async def run_it():
    n = 0 # a number initialed to denote number of object under color sensor line is zero
    count_right = 0 # setting intial right rear gate count to zero
    count_left = 0# setting intial left rear gate count to zero
    await set_positions_front() # setting the front motor to ideal state before operations begins
    await set_positions_rear() # setting the rear motor to ideal state before operations begins
    await runloop.sleep_ms(3000) # now gate waits to ball to reach and line up
    await motor.run_for_degrees(lock_gate, -90, 180) # loading the first ball from incoming line to release to color detector
```

Steps to run run_it() function:

1. Run run_it() function then first all the motors will be set to default position. Then drop the ball in ball drop area then it will take less then 3 second to line up in front of lock.

“n” is a variable used to track number of ball between lock gate and sort gate (ball under the color detect) This concept was set-up to ensure there is no ball under color detector before the lock gate releases the another ball. Our idea is that there should be only one ball under color detector.

First if clause:

```
while True: # an endless loop
    runloop.run(open_line()) # once the first ball is realised to color detector
    n = n + 1# number of ball under color detector is increased by one
    if is_blue() or is_green() and not is_black(): # check if the color is blue or green and not any other color especially black
        count_right = count_right + 1# increase the count by one to right side rear gate
        runloop.run(sort_line_right())# sort motor turn right and sends the ball to right rear gate line
        n = n - 1# one the ball has been directed to respective color line, resulting into no ball under color detector is getting back to zero
        if count_right == 3:            #once 3 balls has been collected
            runloop.run(dump_right())# right rear gate opens and dump the 3 ball into container
            count_right = 0            # then re-set the count to zero (to main a count for upcoming ball towards right rear gate)
```

First with while True condition, and endless loop is initiated. Then the openline() directs the first ball to color detector followed by increase count of “n” indicating there is one ball under the color detector. (n = 1)

Then based on color condition (if the ball is blue or green) the sort motor rotate right directing the ball to right back line followed by decreasing the “n” count by one, indicating there is no ball under the color detector. (n =0)

Then there is a if condition that if the 3 ball are collect in right rear line then those three balls will be dumped into container placed at the sort line architecture. The count of balls in rear right line is monitored by count_right variable.

Second if clause:

```
elif is_red() or is_gold() and not is_black(): # check if the color is red or gold and not any other color especially black
    count_left = count_left + 1                # increase the count by one to left side rear gate
    runloop.run(sort_line_left())            # sort motor turn left and sends the ball to left rear gate line
    n = n - 1    # one the ball has been directed to respective color line, resulting into no ball under color detector is getting back to zero
    if count_left == 3:                        #once 3 balls has been collected
        runloop.run(dump_left())             # left rear gate opens and dump the 3 ball into container
        count_left = 0                         # then re-set the count to zero (to main a count for upcoming ball towards right rear gate)
```

Based of color detecting condition (if ball is red or gold) then the sort motor rotates left and direct the ball towards rear left line and decrease the count of “n” by one. (n = 0)

Then there is a if condition that if the 3 ball are collect in right rear line then those three balls will be dumped into container placed at the sort line architecture. The count of balls in rear right line is monitored by count_left variable.

Possible change:

- Changing color sort side: Just change the is_[color name] function in both the if clause
- Change the number of ball to dump together: change the int value of variable “count_right” and “count_left” and increase the duration of wait time in “dump_right” and “dump_left” function