Mechatronics

Integrated Project

ME-GY 5643

Group 3

Qianyu Yin	qy347
Shivakumar Rajagopalan	sr4082
Mitra Varun Anand	ma4099

2015-12-13

Contents

Smart shopping cart

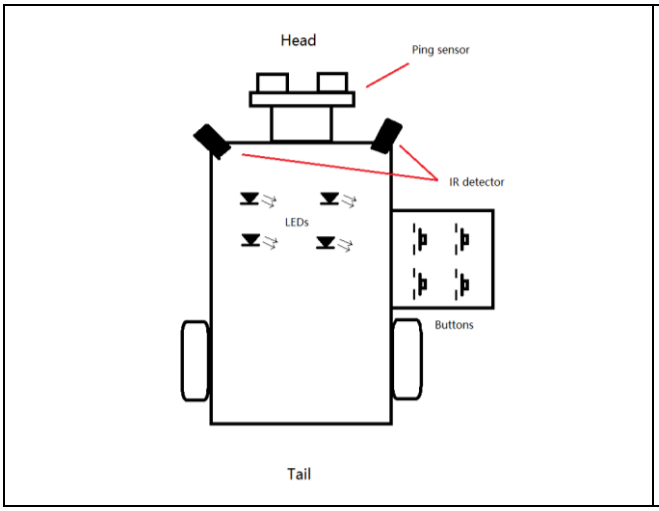
1. Introduction

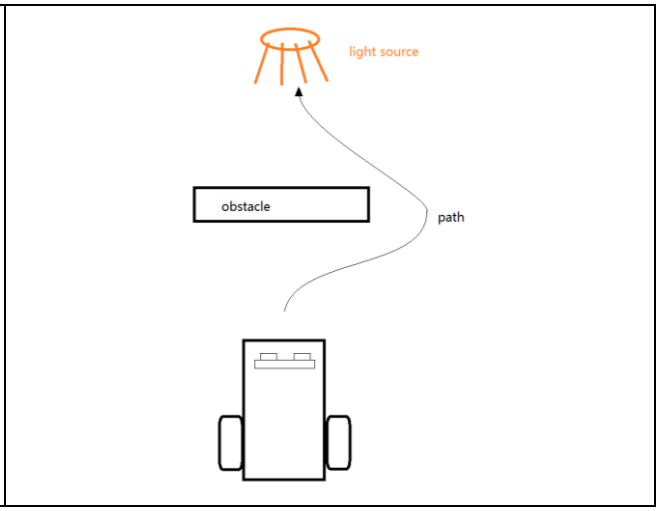
It is upsetting when you go to the supermarket having a long shopping list to buy and have to push a full heavy shopping cart with you. In this project we are going to design a smart, automatic tracking cart that could follow the customer and at the same time get around the obstacles in front of it.

We are going to make use of components already in the kit as many as possible. The robot body will be regarded as a 3-wheel shopping cart, powered by battery. People will be asked to wear a tiny infrared light emitting band on their wrist, and the shopping cart will be equipped with infrared detector to detect the direction of light and will know where the customer is. The cart then will turn to and follow the customer. Another sensor we use is the ultrasonic sensor to detect the distance ahead of it. This sensor will be mounted on a servo at the front of the cart to detect obstacles, go around and at the same time avoid colliding with the customer or any other object. A calculator is quipped and customer is able to know how much they have spent instantly when they put an item into the cart.

2. Key features

- It will follow the customer, there is no need to push it, it will take care of all goods.
- Collision avoidance.
- It can detect and go around the obstacle.
- Calculator is able to show customer how much they have spent.



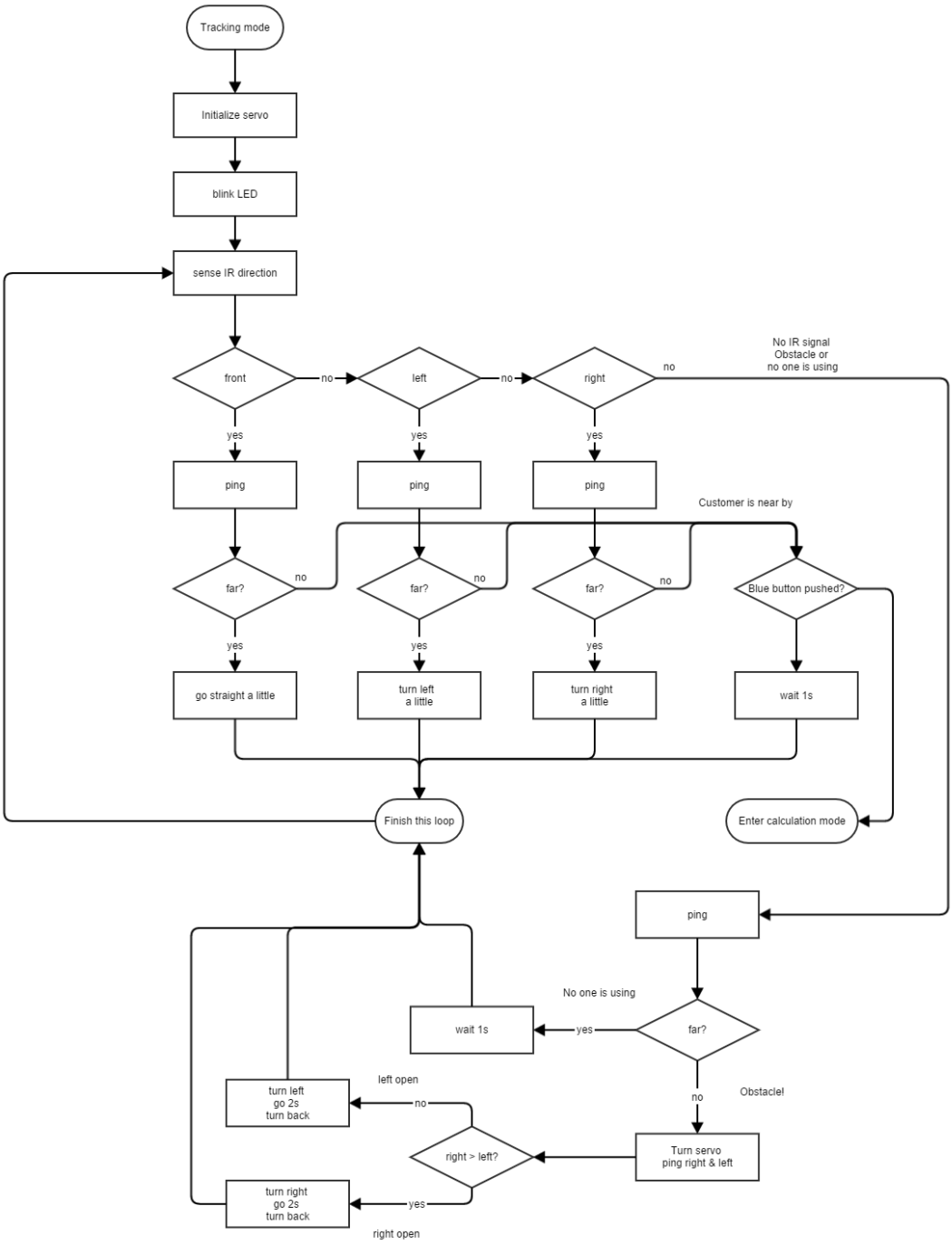


3. Design Logic

The shopping cart has two modes, one is tracking mode, the other is calculation mode. Blue button(BT4) is used to switch between two modes. This chapter we will talk about two different modes separately.

3.1. Tracking mode

Logic diagram:



When the robot is powered on, initially it is in tracking mode, because customer usually get the shopping cart at the gate, where they do not need to calculate. Then the shopping cart will initialize the servo which controls the Ping sensor to middle position, then blink all LEDs in a circle order to give visual indication.

Then the shopping cart start tracking. First it will detect where there is incoming infrared light, and if yes, the direction. The shopping cart prototype only equips two IR detectors. Because these sensors are old, they have limited function, they are not able to give data of exact light strength, only gives "1" or "0" which means "there is IR light" or "there is no IR light". And the sensor only works in pair with the IR emitter which must be connected to the same BS2 board and emit IR with exact frequency of 38500. So we could not use another infrared light torch and work offline from the BS2 to guide the shopping cart. We also tested using an emitter to drive more IR detectors. We mounted 4 IR detectors on the breadboard but only two of them worked. That is why we finally decided to use only two IR detectors.

In fact, two IR detectors are enough and work fine on the cart. The infrared light may come from right ahead, left side, right side. The shopping cart will decide what to do based on the following occasions.

- **Both sensors are activated**. IR comes from right ahead. We use ultrasonic sensor to ping the distance ahead.
  - If the distance is far, means the customer is right ahead and walking away, the shopping cart thus go straight forward for a little distance and end this loop, go back to beginning, detecting IR again to see if the occasion has changed.
  - If the distance is close, means the customer is standing right ahead of the cart, the shopping cart should stop there in case of crash into the customer. Then the cart will detect if the blue button is pressed. If yes, then enter the calculation mode. If not, then wait there for a short period of time and loop back to check infrared light again in case occasion has changed.
- **Left sensor is activated**. IR comes from left side. We use ultrasonic sensor to ping the distance.
  - If the distance is far, means the customer is at left side and walking away, the shopping cart thus turn left for a certain angle and end this loop, go back to beginning, detecting IR again. So the shopping cart will keep turning until facing the customer.
  - If the distance is near, means the customer is standing at left side of the cart, the shopping cart should stop there in case of crash into the customer. Then the cart will detect if the blue button is pressed. If yes, then enter the calculation mode. If not, then wait there for a short period of time and loop back to check infrared light again in case occasion has changed.
- **Right sensor is activated**. IR comes from right side. We use ultrasonic sensor to ping the distance.
  - If the distance is far, means the customer is at right side and walking away, the shopping cart thus turn right for a certain angle and end this loop, go back to beginning, detecting IR again. So the shopping cart will keep turning until facing the customer.
  - If the distance is close, means the customer is standing at right side of the cart, the shopping cart should stop there in case of crash into the customer. Then the cart will detect if the blue button is pressed. If yes, then enter the calculation mode. If not, then wait there for a short period of time and loop back to check infrared light again in case occasion has changed.

The cart will check the blue button only if the customer is nearby, so the calculation mode will only be activated when customer is aside. This design prevents the case that the calculation mode is accidentally triggered when the cart is following its customer and when no one is using the cart. And there is no way a customer could trigger the calculation mode when the cart is in a distance following him/her.

In our design, there is no case for infrared light coming from back. One reason is we don't have the working sensor responsible for back area, and another reason is there is no need. After initialization, the cart

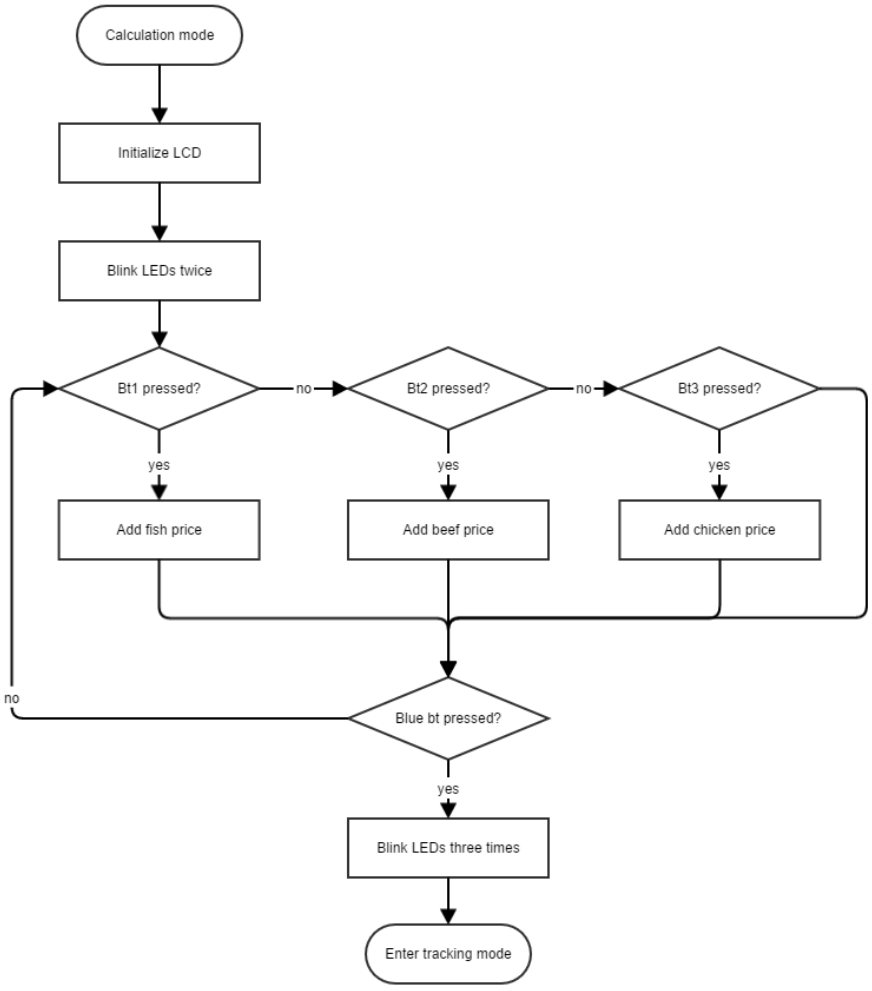
will always turn to the customer and follow them, so there is no chance for customer to go around. If the customer is nearby, the cart will stop and wait, and customer can walk around the cart to pick up items. At this situation, it is the Ping sensor to decide whether continue following, not the IR detector. And if the case really happens which the customer at the back of the cart in a distance, the cart will just stop and wait, it is not allowed to move itself for safety of other people. Customer only need to walk to either side and wave hand to enable the sensor, then the cart will continue following.

Usually, if the IR sensor could not detect infrared light, there two situations. We use the ping sensor to judge these two situations.

- If the distance is near, there is an obstacle in front of the cart and block the infrared light.
  - We start using servo to turn Ping sensor to detect distance at left and right side of the cart, in order to find out which direction we should turn to go around the obstacle. If the left distance is longer, meaning left side is open, thus the cart will go around from left side, and the same to right side. Then
- If the distance is far, no one is using the cart. It just stop there and wait. It will loop back to check infrared light.

3.2. Calculation mode

Logic diagram



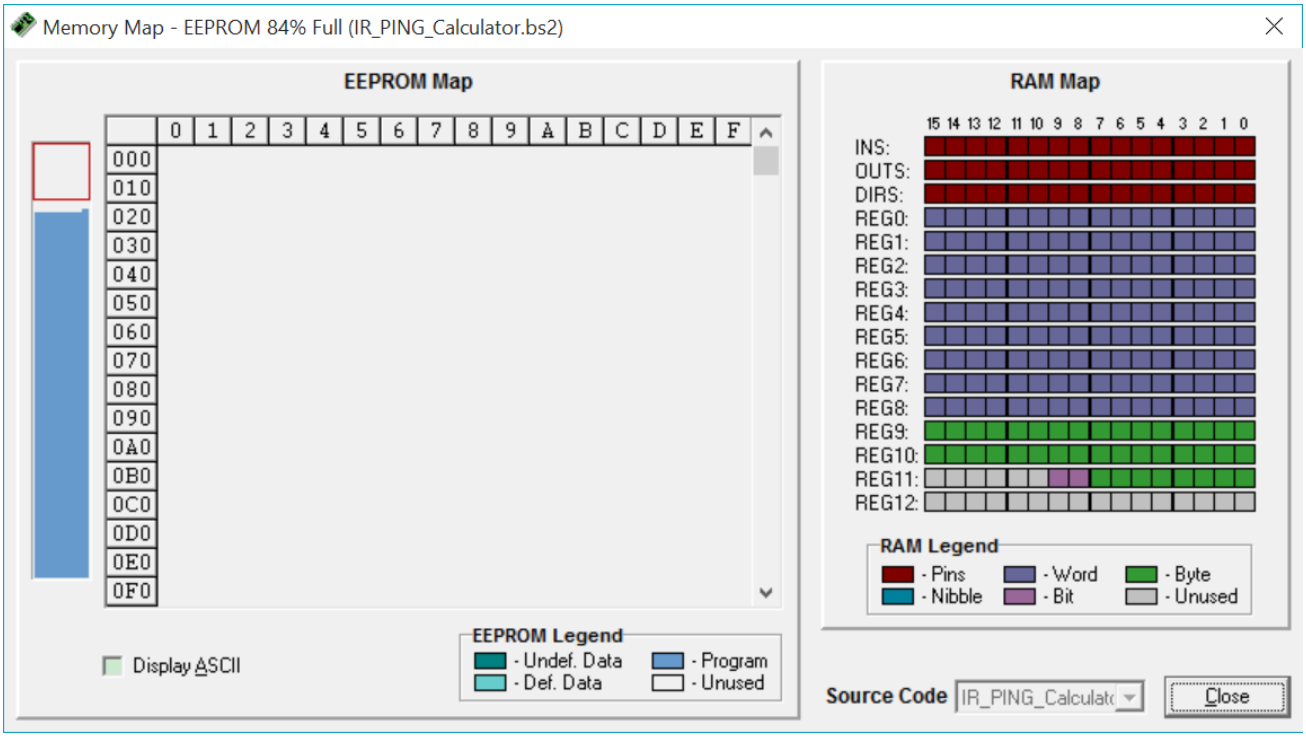
When entering the calculation mode, the cart will first initialize the LCD, display welcome message, then blink all LEDs at the same time twice to give visual indication. Then it will continue checking BT1 to BT4 to see whether they are pressed. The following situations are considered.

- Button 1 is pressed, customer buy a fish, add $20 to the total price and display it.
- Button 2 is pressed, customer buy a box of beef, add $10 to the total price and display it.
- Button 3 is pressed, customer buy a chicken, add $ 5 to the total price and display it.
- Button 4 is pressed, blink all LEDs at the same time for three times and display message for tracking mode and go to tracking mode.

Customer may switch between tracking mode and calculation mode for many times. The total price is designed not to be erased after switching modes. So each time customer enter calculation mode add some price, the total price will keep adding from the previous time, not erased unless power shut down or system reset.

3.3. Memory usage

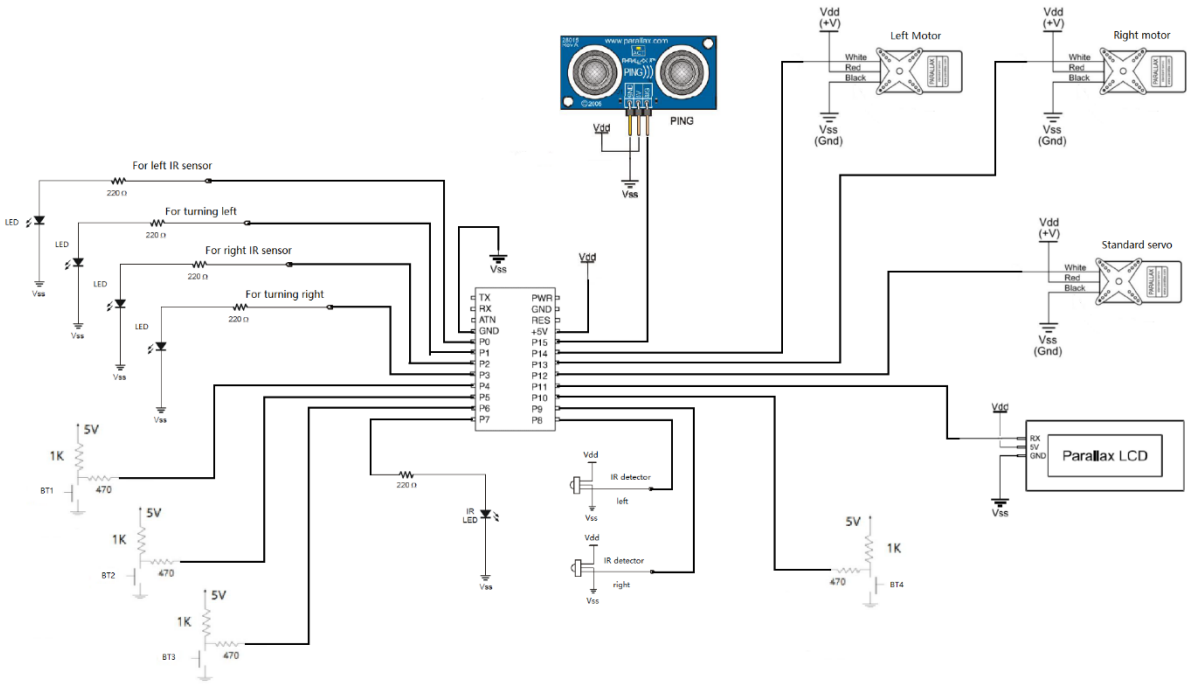
Memory map



We used about 85% of the EEPROM and we could add more function to the design if we have more RAM.

4. Circuit design

4.1. Circuit diagram



4.2. Component and I/O

Serial Number	Item	I/O PIN
1	IR detector left	8
2	IR detector right	9
3	IR emitter	7
4	LED left (for Ping direction)	0
5	LED right (for Ping direction)	2
6	LCD display	11
7	LED left (for IR direction)	1
8	LED right (for IR direction)	3
9	Continuous servo left	14
10	Continuous servo right	13
11	Standard servo	12
12	Ping sensor	15

Serial Number	Item	I/O PIN
13	Button 1 (for fish)	4
14	Button 2 (for beef)	5
15	Button 3 (for chicken)	6
16	Button 4 (for mode switch)	10

Actuator used: continuous servo, standard servo.
Analog sensor used: button.
Digital sensor used: IR detector, Ping sensor.

4.3. Safety guideline

This shopping cart will not do any offensive action to customer, but we still implement hardware and software level mechanism for safe operation.

Hardware:

- Power switch. If the cart lost control, turn off the power switch.
- Reset button. It is a built in button. If the cart has bug or dead loop, push the button and it will initialize.

Software:

The program design follows the philosophy "**stop first, passive guidance**".

- **Stop first**: It will only take action if the exact condition is programed. Any other condition, no matter whether it will happen, such as customer facing the back of the cart in a distance, will not activate the cart and the cart will just wait there. This design philosophy decreases the possibility of non-normal performance.
- **Passive guidance**: There are many customers in the supermarket and goods on the shelf are valuable, so we do not allow the cart randomly go straight, turn around, to find its customer if it lose the target. It will only be activated when customer is at the correct position. This design philosophy decrease the possibility of collision and helps protect other customers.

Safety regulations:

- The maximum load capacity of the cart is 100KG. Overload the cart will cause damage to its structure and servo, and drain the battery quicker.
- Intentionally strike or collide the cart should be prohibited, which will cause damage to its structure and circuit.
- Supermarket staff should use original charger to charge the cart. Any unauthorized third party charger may cause damage to the battery and circuit.
- The shopping cart should only be used in supermarket; rough surface outside may cause damage to the structure.

5. production

cost of manufacture for each shopping cart:

component	quantity	Cost ($)	Cost of mass production
BS2 board of education	1	69.99	49.99
IR detector	2	1.98	1.49
Ping sensor	1	22.49	17.99
Serial LCD	1	27.99	21.00
Cart	1	-(19.99)*	-(14.99)*
wheels	2	7.99	6.99
Servos	3	51.99	34.99
LED	4	2.99	2.49
Breadboards	4	10.99	5.99
IR emitter	1	1.99	1.49
Battery	4	6.99	4.99
Miscellaneous	-	10	4
Total	-	208.4	153.41

Source : Parallax website

*note : The production will be upgrading the current shopping cart, so the price for cart body will not be counted. The given price is just for reference.

Cost of upgrading an existing shopping cart is $208.4, but if mass manufactured, the entire setup can be done in $153.41. Note that the cost of mass manufacturing has been calculated based on the assumption that a supermarket needs 300 upgraded carts.

Battery:

This unit utilizes 4 1.5V batteries, that gives a usage time of 300 minutes on full load. However, for practical applications, a bigger battery is suggested, giving a usage time of 600 minutes on full charge under full load(100 KG) conditions. Time to recharge the batteries is 150 minutes.

6. Future improvement

The existing prototype can be improved for better functionality and ease of use in the following ways:

- Use of better circuitry (Arduino Mega) capable to driving the motors faster and better sensors for more accurate distance measurements. And more I/O pins, more RAM and EEPROM which enable implementing more complicated function.
- Coded IR emitters and sensors, to distinguish between various customers and following the right customer (master).
- Using infrared and ultrasonic (PING) sensors in all four sides of the cart, for more efficient sensing and

navigation.

- Implementation of a bar-code scanner or RFID, that lets the customer to scan the products when they put it in the cart. Then the customer can just go to the counter to pay the total amount instead of scanning one by one. This approach can save time at the billing counters substantially.
- Making the wheels more rugged, for better movements in uneven or bumpy terrain.
- A feature to let the user set a budget for his/her current shopping trip, and warning the user when the maximum amount is reached.

7. Conclusion

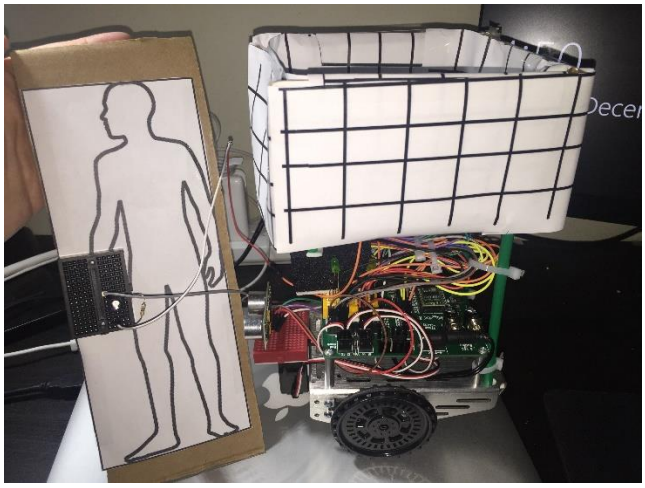
A BS2 controlled shopping cart is designed in this integration project. Our knowledge mechatronics is enhanced and hands on experience is practiced. By working on the project we get better understanding of:

- operational principles of mechanical, electrical, electronic, and optoelectronic components.
- basic laws governing the operations of sensors and actuators.
- Basic Stamps 2 (BS2) microcontroller—fundamentals, operation, programming, and interfacing.
- Design, construct, and evaluate a prototype mechatronics system.
- Inter-disciplinary: an integrative process involving two or more disciplines simultaneously to bear on a problem.

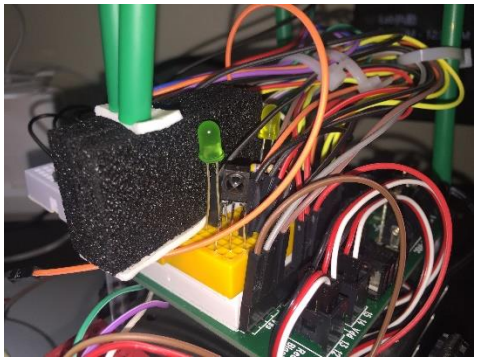
And the concept for this project is not only for shopping cart. It is capable of doing other functions using the already installed sensors:

- **A robot that could escape from a maze:** The ultrasound sensor will enable the robot to detect distance around it, it could walk inside a maze along one side of the wall and find the exit.
- **A robot that aids with disaster relief:** To search for people in an earthquake shattered building, search for active mines and so on, with minimal modifications to design.
- **A robot that will chase the ball:** There is a kind of ball on the market that will emit infrared light. So the robot will chase the ball. Imagine you are playing soccer with your friends and the robot will chase it. Or you can just let the robot play the ball with a pet.

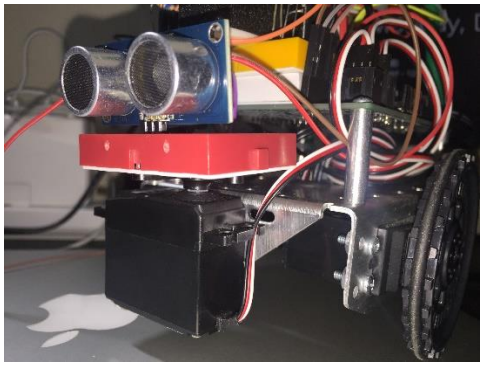
8. Appendix-shopping cart photos



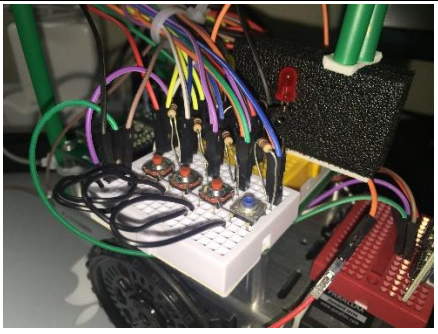
Shopping cart with customer



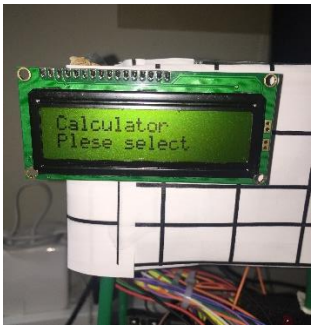
IR detector and LEDs



Ping sensor and standard servo



Buttons



Display

Display

Cable connection

9. Appendix-code

```
' {$STAMP BS2}
' {$PBASIC 2.5}

'This program is the final version which will run on the robot
'It has two modes, one is tracking mode and the other is calculation mode
'By pushing blue button we can swtich between these two modes.
'The robot could follow the customer and avoid collision at the same time.
'It can go around the obstacle in front of it
'It can also calculate and display how much money a customer spend in calculation mode.

''''''''''''''''''''''''version 3.2''''''''''''''''''''''''''''''''''''''''''''''''''''
'12/13/2015


IR_detect_left  VAR  Bit                'variablt for IR detector
IR_detect_right  VAR  Bit

i VAR Byte                              'variablt for servo control and ping calcuation
'counter VAR Byte
time VAR Word

distance VAR Word                       'variable for storing ping data
distance_left VAR Word
distance_right VAR Word

left PIN 14                             'left/right motor I/O
right PIN 13

LED_turn_left PIN 1                     'LED turn left/right I/O
LED_turn_right PIN 3
```

```
right_go VAR Word                       'set constant for controling motor speed
right_back VAR Word
'right_stop VAR Word
left_go VAR Word
left_back VAR Word
'left_stop VAR Word

right_go = 600
right_back = 900
'right_stop = 750
left_go = 900
left_back = 600
'right_stop = 750

Btn1 PIN 4                               'set variables for button action
BtnWrk1 VAR Byte
Btn2 PIN 5
BtnWrk2 VAR Byte
Btn3 PIN 6
BtnWrk3 VAR Byte
Btn4 PIN 10
BtnWrk4 VAR Byte

k VAR Word                               'Set variable for price calculation
k= 0
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
                  'Initialize the servo
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'Set the servo to initial middle position.
'Flash all LEDs in circle for visual help.


FOR i = 1 TO 50
    PULSOUT 12, 750          'Set servo middle position
    PAUSE 20
  NEXT

PAUSE 100

LOW 7

HIGH 0                        'Flash all LEDs in circle
LOW 2
LOW LED_turn_right
LOW LED_turn_left

PAUSE 1000

LOW 0
HIGH 2
LOW LED_turn_right
LOW LED_turn_left

PAUSE 1000
```

```
LOW 0
LOW 2
HIGH LED_turn_right
LOW LED_turn_left

PAUSE 1000

LOW 0
LOW 2
LOW LED_turn_right
HIGH LED_turn_left

PAUSE 1000

LOW 0
LOW 2
LOW LED_turn_right
LOW LED_turn_left
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
                    'Start here
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'Detect whether there is infrared light around
'And then jump into corresponding subfunction to take different actions.


main:


  'PAUSE 10
  FREQOUT 7, 1, 38500          'Emit infrared light
  IR_detect_left = IN8
  IR_detect_right = IN9


  IF IR_detect_left = 1 AND IR_detect_right = 1 THEN
    DEBUG  "no object detected. " ,CR
    LOW 0
    LOW 2
    GOTO not_detect             'Maybe obstacle is ahead, maybe no one is using

  ELSEIF  IR_detect_left = 0 AND IR_detect_right = 1 THEN
    DEBUG  "left is detected. "  ,CR
    HIGH 0
    LOW 2
    GOTO turn_left               'customer at the left, but don't know how far
```

```
  ELSEIF  IR_detect_left = 1 AND IR_detect_right = 0 THEN
    DEBUG  "right is detected. " ,CR
    HIGH 2
    LOW 0
    GOTO turn_right             'customer at the right, but don't know how far

  ELSEIF  IR_detect_left = 0 AND IR_detect_right = 0 THEN
    DEBUG  "both is detected. " ,CR
    HIGH 0
    HIGH 2
    GOTO both_detect            'run to the customer, but should avoid collision

ENDIF

GOTO main
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
                    'Customer is at left/right
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'Two sub-functions for cases that customer is at right/left
'Then go to corresponding sub-function to take action.

turn_left:

GOTO ping_test_stop_left

GOTO main

turn_right:

GOTO ping_test_stop_right

GOTO main
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
                    'If the customer is ahead
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'Customer is ahead, but we do not know the distance
'Then got subfunction to ping the distance

both_detect:     ' will test the obstacle

GOTO ping_test_stop

GOTO main
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
                    'If the customer is not ahead
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'No infrared light detected, then ping the distance
'If distance is short -> obstacle ahead, go around.
'If distance is long -> no one is using it, stay.

not_detect:      ' will test the obstacle

GOTO ping_test

GOTO main
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
                    'Ping test stop ahead
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'Customer is ahead, ping the distance.
'If distance is long, go ahead.
'If distance is short, customer is nearby, stop, check whether enter calculation mode.

ping_test_stop:

 PULSOUT 15, 5                          'Calculate the distance
 PULSIN 15, 1, time
 distance = time ** 2251
' DEBUG CR, "Distance = ", DEC4 distance, " cm"


IF distance >12 OR distance =12 THEN             'means customer is far

  i = 0                       'go straight a little bit
  FOR  i=1 TC 10
    PULSOUT 13, right_go
    PULSOUT 14, left_go
  NEXT

ELSEIF  distance =11 OR distance <11 OR distance > 315 THEN   'means master is nearby

Enter_calculation:                                          'Check if button is pressed
BUTTON Btn4, 0, 255, 20, BtnWrk4, 1 ,Calculation_mode       'If pressed, enter calculation mode

PAUSE 1000

ENDIF

GOTO main
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
                    'Ping test stop left
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'Customer is at left, ping the distance.
'If distance is long, turn left.
'If distance is short, customer is nearby, stop, check whether enter calculation mode.

ping_test_stop_left:

 PULSOUT 15, 5                         'calculate the distance
 PULSIN 15, 1, time
 distance = time ** 2251
' DEBUG CR, "Distance = ", DEC4 distance, " cm"

IF distance >12 OR distance =12 THEN             'means customer is far

  i = 0                        'turn left a little bit
  FOR  i=1 TC 10
    PULSOUT 13, 600
    'PULSOUT 14, 750
  NEXT

ELSEIF  distance =11 OR distance <11 OR distance > 315 THEN   'means master is near

  BUTTON Btn4, 0, 255, 20, BtnWrk4, 1 ,Calculation_mode       'Check if button is pressed
  PAUSE 1000                                                  'If pressed, enter calculation mode

  'GOTO go_around                              'wait, do nothing
ENDIF

GOTO main
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
                    'Ping test stop right
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'Customer is at right, ping the distance.
'If distance is long, turn right.
'If distance is short, customer is nearby, stop, check whether enter calculation mode.

ping_test_stop_right:

 PULSOUT 15, 5                    'calculate the distance
 PULSIN 15, 1, time
 distance = time ** 2251
' DEBUG CR, "Distance = ", DEC4 distance, " cm"
```

```
IF distance >12 OR distance =12 THEN             'means master is far

  'go straight a little bit
  i = 0
  FOR  i=1 TC 10
    'PULSOUT 13, 750
    PULSOUT 14, 900
  NEXT

ELSEIF  distance =11 OR distance <11 OR distance > 315 THEN   'means master is near

  BUTTON Btn4, 0, 255, 20, BtnWrk4, 1 ,Calculation_mode       'Check if button is pressed
  PAUSE 1000                                                  'If pressed, enter calculation mode

  'GOTO go_around                              'wait, do nothing
ENDIF

GOTC main
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
                    'Ping test
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'This ping test is for case when no infrared light is detected.

ping_test:

 PULSOUT 15, 5                               'calculate the distance
 PULSIN 15, 1, time
 distance = time ** 2251
' DEBUG CR, "Distance = ", DEC4 distance, " cm"


IF distance >12 OR distance =12 THEN        'master is not around, just wait there

  PAUSE 1000

ELSEIF  distance =11 OR distance <11 OR distance > 315 THEN   'means obstacle ahead, block the infrared light

  PAUSE 1000
  GOTC go_around                                               'go around
ENDIF

GOTC main
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
                    'Go around main
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'We use servo to test left/right distance.
'Then turn left/right based on which side has longer distance.

go_around:


GOSUB ping_left_right        'Go to sub-function to test distance of left/right


IF distance_right > distance_left THEN       'go from right

  'DEBUG "left ", DEC distance_left , " right ", DEC distance_right, CR
  HIGH LED_turn_right
  LOW LED_turn_left

  i = 0
  FOR  i=1 TC 90    'turn right     13-900    14-900
  PULSOUT left, left_go
  PULSOUT right, right_back
  NEXT
  PAUSE 1000

  i = 0
  FOR  i=1 TC 250    ' go straight      13-600    14-600
  PULSOUT left, left_go
  PULSOUT right, right_go
  NEXT
  PAUSE 1000
```

```
  i = 0
  FOR  i=1 TC 120    ' turn left      13-600    14-600
  PULSOUT left, left_back
  PULSOUT right, right_go
  NEXT
  PAUSE 1000

  LOW LED_turn_left
  LOW LED_turn_right

  i = 0
  FOR  i=1 TC 50    ' go straight      13-600    14-600
  PULSOUT left, left_go
  PULSOUT right, right_go
  NEXT
```

```
ELSEIF distance_right <= distance_left  THEN    'go from left

  'DEBUG "left ", DEC distance_left , " right ", DEC distance_right, CR
  LOW LED_turn_right
  HIGH LED_turn_left

  i = 0
  FOR  i=1 TC 110    ' turn left      13-600    14-600
  PULSOUT left, left_back
  PULSOUT right, right_go
  NEXT
  PAUSE 1000

  i = 0
  FOR  i=1 TC 250    ' go straight      13-600    14-600
  PULSOUT left, left_go
  PULSOUT right, right_go
  NEXT
  PAUSE 1000

  i = 0
  FOR  i=1 TC 90    'turn right     13-900    14-900
  PULSOUT left, left_go
  PULSOUT right, right_back
  NEXT
  PAUSE 1000

  LOW LED_turn_left
  LOW LED_turn_right
```

```
  i = 0
  FOR  i=1 TC 50    ' go straight      13-600    14-600
  PULSOUT left, left_go
  PULSOUT right, right_go
  NEXT

ENDIF


GOTO main
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
                    'Ping left and right
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'Turn servo left/right and ping the distance.


ping_left_right:             ' This part we drive servo left/right

i = 0

  FOR i = 1 TC 50
    PULSOUT 12, 1000      ' Servo left
    PAUSE 20
  NEXT

PAUSE 1000
GOSUB ping_left

i = 0

  FOR i = 1 TC 50
    PULSOUT 12, 500      ' Servo servo right
    PAUSE 20
  NEXT

PAUSE 1000

GOSUB ping_right
```

```
FOR i = 1 TC 50
    PULSOUT 12, 750      ' Servo middle
    PAUSE 20
  NEXT


PAUSE 1000

  'two parameters will be returned to upper level function

RETURN
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
                    'Ping the distance at left
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'Ping the distance at left, return the distance

ping_left:

 PULSOUT 15, 5
 PULSIN 15, 1, time
 distance_left = time ** 2251
 DEBUG CR, "Distance_left = ", DEC4 distance_left, " cm"

RETURN
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
                    'Ping the distance at right
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'ping the distance at right, return the distance

ping_right:

 PULSOUT 15, 5
 PULSIN 15, 1, time
 distance_right = time ** 2251
 DEBUG CR, "Distance_right = ", DEC4 distance_right, " cm"

RETURN
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
                    'Calculation mode
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'Caculation mode is triggered only if the cart is near the customer and the blue button is pushed.
'Then all LEDs will blink twice for visual indication.
'Push button 1,2,3, you will add price of fish, beef, chicken. Price will show on LCD display
'Push the blue button again will quit the calculation mode and start tracking.
'All LEDs will blink three times for visual indication.


Calculation_mode:

SEROUT 11, 84, [22, 12] 'Initialize LCD
SEROUT 11, 84, ["Calculator", 13,"Plese select" ]

GOSUB blink_twice                           'blink LED twice

PAUSE 5

check_bt1:

PAUSE 5
BUTTON Btn1, 0, 255, 20, BtnWrk1 ,0, check_bt2                'Add fish price
k=k+20
SEROUT 11, 84, [22, 12]
PAUSE 5
DEBUG ? k
SEROUT 11, 84, ["Fish cost 20", 13,"Total: $",SDEC k ]
```

```
GOTO check_bt1


check_bt2:
BUTTON Btn2, 0, 255, 20, BtnWrk2 ,0 ,check_bt3                 'Add beef price
k=k+10
SEROUT 11, 84, [22, 12]
PAUSE 5
DEBUG ? k

SEROUT 11, 84, ["Beef cost 10",13, "Total: $",SDEC k ]

GOTO check_bt1


check_bt3:
BUTTON Btn3, 0, 255, 20, BtnWrk3, 0 ,No_Press                  'add chicken price
k=k+5
SEROUT 11, 84, [22, 12]
PAUSE 5
DEBUG ? k

SEROUT 11, 84, ["Chichen cost 5",13, "Total: $",SDEC k ]

GOTO check_bt1
```

```
No_Press:

BUTTON Btn4, 0, 255, 20, BtnWrk4, 0 ,check_bt1
DEBUG "Switch mode to tracking",CR
SEROUT 11, 84, [22, 12]
SEROUT 11, 84, ["Tracking mode",13, "Thank you" ]              'Back to tracking mode

GOSUB blink_three_times:

PAUSE 1000

GOTO main
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
                    'Blink LED twice
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'Blink all LEDs twice for indicating entering calculation mode

blink_twice:

LOW 0
LOW 2
LOW LED_turn_right
LOW LED_turn_left

PAUSE 500

HIGH 0
HIGH 2
HIGH LED_turn_right
HIGH LED_turn_left

PAUSE 500

LOW 0
LOW 2
LOW LED_turn_right
LOW LED_turn_left
```

```
PAUSE 500

HIGH 0
HIGH 2
HIGH LED_turn_right
HIGH LED_turn_left

PAUSE 500

LOW 0
LOW 2
LOW LED_turn_right
LOW LED_turn_left

RETURN
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
                    'Blink LED three times
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'Blink all LEDs three times to indicate entering tracking mode again.

blink_three_times:

LOW 0
LOW 2
LOW LED_turn_right
LOW LED_turn_left

PAUSE 200

HIGH 0
HIGH 2
HIGH LED_turn_right
HIGH LED_turn_left

PAUSE 200

LOW 0
LOW 2
LOW LED_turn_right
LOW LED_turn_left

PAUSE 200
```

```
HIGH 0
HIGH 2
HIGH LED_turn_right
HIGH LED_turn_left

PAUSE 200

LOW 0
LOW 2
LOW LED_turn_right
LOW LED_turn_left

PAUSE 200

HIGH 0
HIGH 2
HIGH LED_turn_right
HIGH LED_turn_left

PAUSE 200
```

```
LOW 0
LOW 2
LOW LED_turn_right
LOW LED_turn_left

RETURN
```