CS 684 Project Report

Group # 10

Fruit Sorting Robot

Pankaj Singh(133050053) Sushmita Bhattacharya(133050022)

Pankaj Kumar(133050076) Mayuri Khardikar(133050017)

November 14, 2013

Contents

1 Introduction

In agriculture domain, sorting of fruits/flowers/vegetables is very common practice. In India, usually, it is carried by human efforts. This leads to waste of precious human time. It also suffers with the problem of inefficiency and inaccuracy introduced by human. Our goal is to automate the process of fruit sorting .Our final product is a bot to sort a set of fruits into separate categories based on its attributes like size,color etc without human intervention. In this document we present our project report

1.1 Definitions, Acronyms and Abbreviations

RTS: Real time system

GUI: Graphical user Interface

PC: personal computer. This can be any laptop, or desktop. Here we are using it for image processing purpose.

FB V: Fire-Bird Five Robot.

2 Problem Statement

To built a real time, user friendly system that will automate the process of sorting a perticular type of fruit based on its features such as size,color etc. and Reduce the human intervention required in this task.

3 Requirements

3.1 Hardware Requirements

1. Hardware should be modular. i.e each part can be easily attached or detached from other.
2. The Hardware should be portable.
3. The container should contain atleast 10-12 fruits.
4. The bot should sort different kinds of fruits by extending or replacing small part of hardware components.

3.2 Functional Requirements

1. Real time system to sort the fruit without human user intervention.
2. UI to take users choices as input (e.g. number of classes, characterstic on which to sort etc.)
3. UI to display count of the fruit (in each class and total separately)
4. It should provide helping modules for developers to test Vision module.
5. It should provide helping modules test communication system between FBV and PC.

3.3 Non Functional Requirements

1. The bot should work in soft Real time.
2. It should not harm the fruits while operating.
3. System should be portable
4. Low unit cost design is preferable.
5. There should be a modular software design for fruits.
6. High performance (low processing time per fruit)
7. Low Power consumption

3.4 Design Constraints

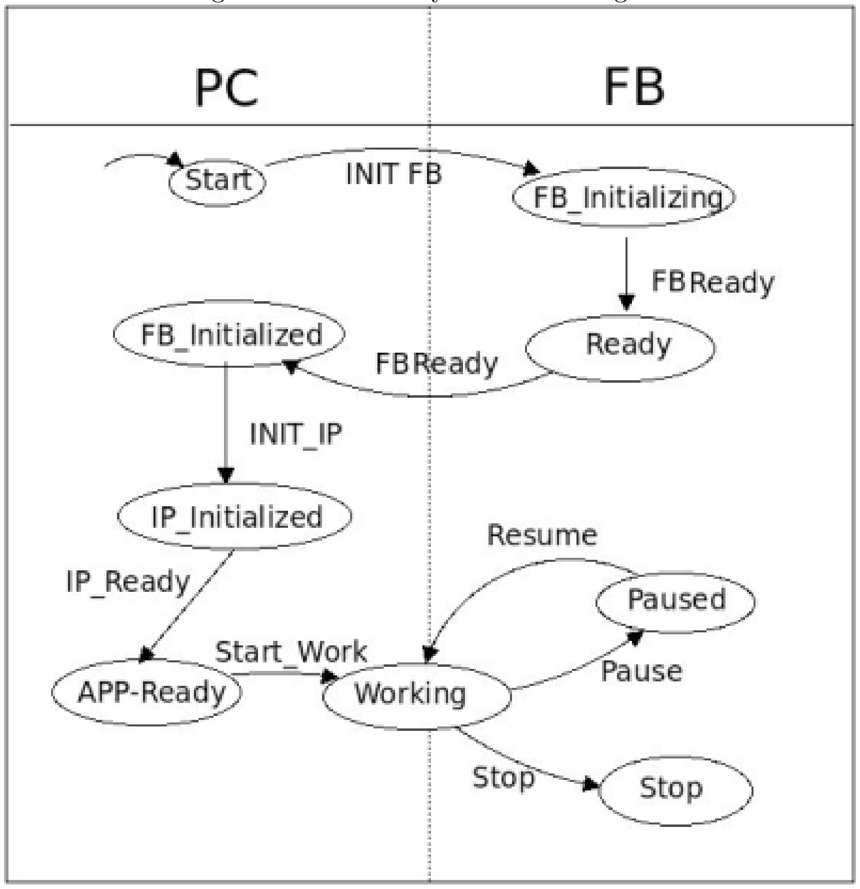
1. System should be portable.
2. Cost constraints.
3. Time to prototype: 2 weeks
4. Modular design with use of proper interface for hardware and software both
5. High performance(so low processing time per fruit)
6. low energy consumption.

4 Implementation

We continuously focused on modularity. So the system consists of main 3 modules:

1. Fruit dispensing module

Figure 4.1: Overall System State diagram



2. Image processing module

3. Fruit routing module

hese 3 modules are connected to each other and they work synchronously. Following state diagram explains overall flow of working of the system. Here the oval labeled as 'Working 'constitutes of the main sorting task of the system as described by the sections below.

4.1 Fruit dispensing module

Its main task is to extract 1 fruit at a time from number of fruits and thus dispence exactly single fruit at a time to IP module. It is achieved using controlled real time back and forth motion of piston with the help of DC motor, Rack and pinion mechanism and 2 IR proximity sensors. This module is composed of :

a Software code residing on FB V

b Funnel: A container to hold pile of fruits. So initially user puts fruits into this. Diameter of funnel's lower part and pipe surrounding the piston limits the maximum size of fruit hadled by the ststem.

c A piston, DC motor, two IR proximity sensors(1 at the tip of the piston and another in IP module near the shutter): Circular motion of DC motor is converted into linear motion of piston using rack and pinion mechanism.

IR sensor at tip of piston senses whether it has fruit in front of it. If no then it will move back till a fruit falls in front of it. As soon as it senses fruit it starts moving forward to push the fruit ahead. It continues this until the sensor near shutter senses a fruit and thus stops piston. FB V sends signal to IP module on PS that 'tell me the class label' of this fruit via serial communication. Now all hardaware parts halt till IP module processes the fruit and decide its class lable. Dispensing module will resume its work once it gets 'extract_next' signal from IP module. All sensors and motors are contolled by code running on FB V. Statechart of fruit dispensing module:

Figure 4.2: State Chart for the whole system illustrating state diagram of dispencing module

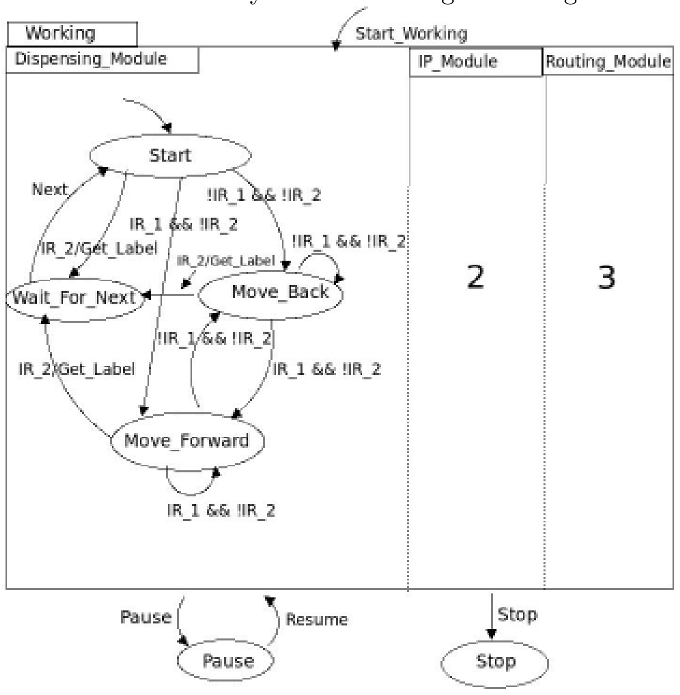


Figure 4.3: State diagram for Image Processing (Firebird part)

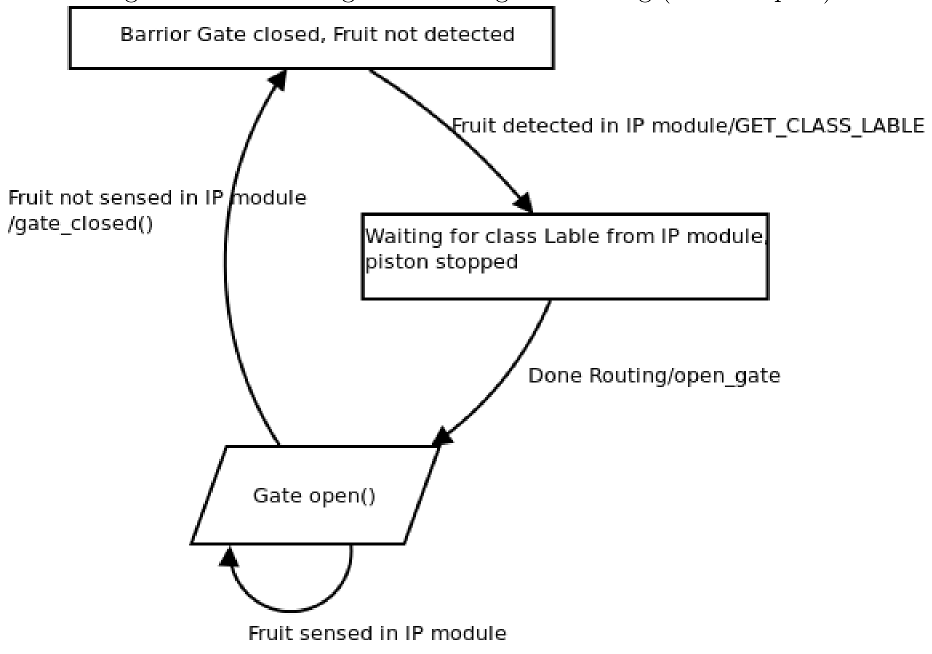

Barrior Gate closed, Fruit not detected
Fruit detected in IP module/GET_CLASS_LABLE
Fruit not sensed in IP module
/gate_closed()
Waiting for class Lable from IP module,
piston stopped
Done Routing/open_gate
Gate open()
Fruit sensed in IP module

Figure 4.4: State diagram for Image Processing (PC part)

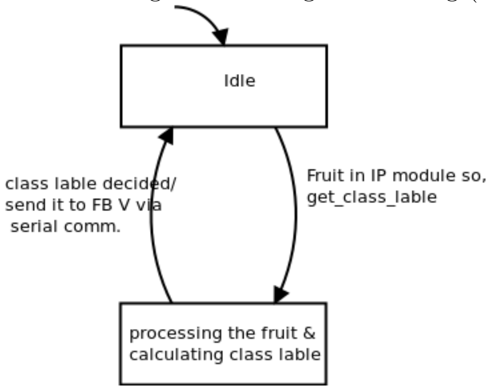
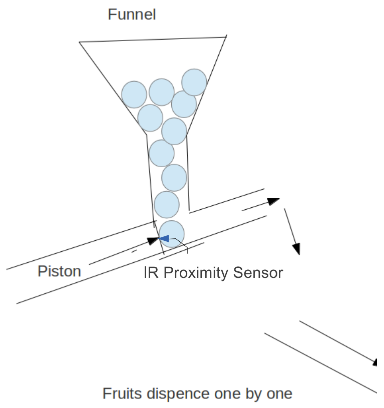

Idle
class lable decided/ send it to FB V via serial comm.
Fruit in IP module so, get_class_lable
processing the fruit & calculating class lable

Figure 4.5: Fruit dispencing module



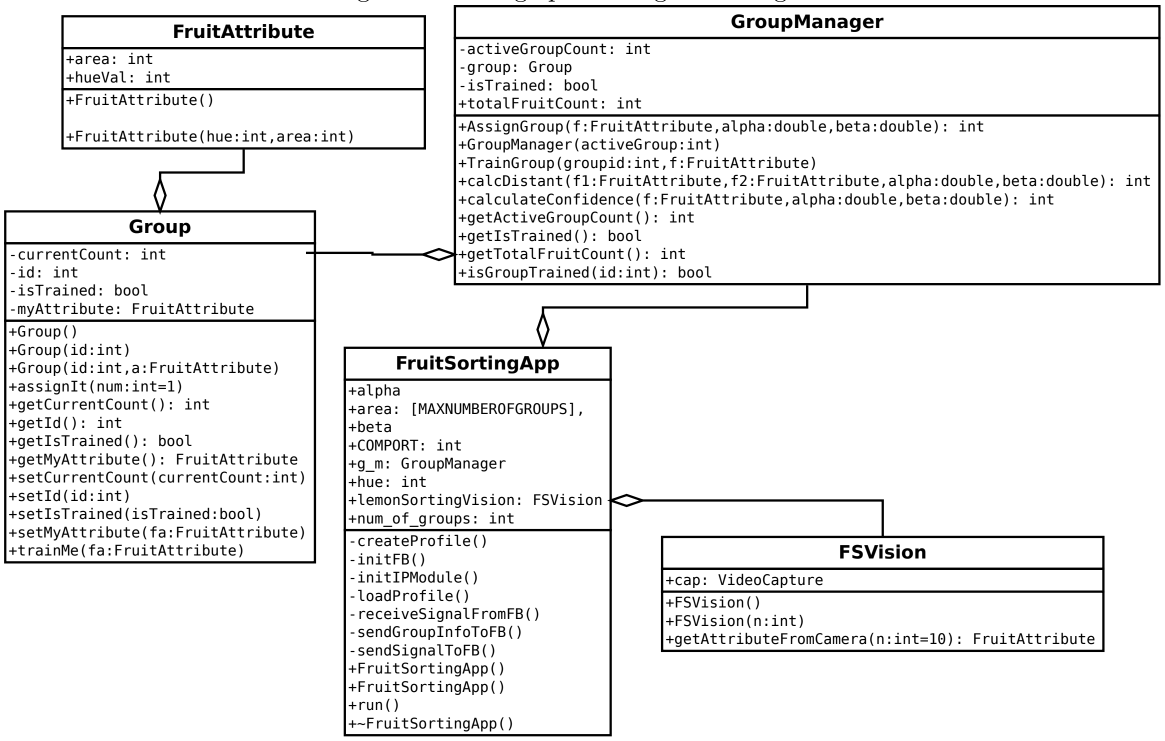
4.2 Image processing module

We painted the hardware part of IP module with blue color and hardware design of the module ensures that the fruit dispensed will fall into a small constrained area. So camera need to focus on only that small area.This simplifies the task of IP module and helps to reduce the effects of environmental noise.

IP module consists of :

1. **Hardware**: Camera, colored area to hold fruit till IP module proceses it, PC to execute IP code, serial

Figure 4.6: Image processing class diagram



FruitAttribute
+area: int
+hueVal: int
+FruitAttribute()
+FruitAttribute(hue:int,area:int)
GroupManager
-activeGroupCount: int
-group: Group
-isTrained: bool
+totalFruitCount: int
+AssignGroup(f:FruitAttribute,alpha:double,beta:double): int
+GroupManager(activeGroup:int)
+TrainGroup(groupid:int,f:FruitAttribute)
+calcDistant(f1:FruitAttribute,f2:FruitAttribute,alpha:double,beta:double): int
+calculateConfidence(f:FruitAttribute,alpha:double,beta:double): int
+getActiveGroupCount(): int
+getIsTrained(): bool
+getTotalFruitCount(): int
+isGroupTrained(id:int): bool
Group
-currentCount: int
-id: int
-isTrained: bool
-myAttribute: FruitAttribute
+Group()
+Group(id:int)
+Group(id:int,a:FruitAttribute)
+assignIt(num:int=1)
+getCurrentCount(): int
+getId(): int
+getIsTrained(): bool
+getMyAttribute(): FruitAttribute
+setCurrentCount(currentCount:int)
+setId(id:int)
+setIsTrained(isTrained:bool)
+setMyAttribute(fa:FruitAttribute)
+trainMe(fa:FruitAttribute)
FruitSortingApp
+alpha
+area: [MAXNUMBEROFGROUPS],
+beta
+COMPORT: int
+g_m: GroupManager
+hue: int
+lemonSortingVision: FSVision
+num_of_groups: int
-createProfile()
-initFB()
-initIPModule()
-loadProfile()
-receiveSignalFromFB()
-sendGroupInfoToFB()
-sendSignalToFB()
+FruitSortingApp()
+FruitSortingApp()
+run()
+~FruitSortingApp()
FSVision
+cap: VideoCapture
+FSVision()
+FSVision(n:int)
+getAttributeFromCamera(n:int=10): FruitAttribute

communication cable to connect PC and FB V.

2. **Software**: Software part of IP module runs on PC. The software part that captures frames of the fruit using camera, extracts its features such as size(area), color(hue)(Open CV is used for this) and decides the class lable of the fruit based on its features.

   Algorithm used for deciding class lable is as follows:

   IP module refers to a profile file chosen by user which stores values of various features(hue, area) for different classes. So distance of the new fruit is calculated from each class using distance formula considering values of features as coordiates. So if new fruit is closest to class A then its class is decided as A with the confidence lavel which is directly proportional to proximity of the fruit from that class as compared to other classes. Then IP module sends decided class lable to FB V using serial comunication.

4.3 Fruit routing module

As soon as FB V gets class lable from PC, it activates servomotor1 to change the direction of guiding channel in routing module to route fruit to its proper class container. It also activates servomotor2 to open the gate which is holding the fruit in IP module till now. So now fruit falls into the guiding channel and goes to proper container. Then it generates 'extract_next' signal so that dispensing module will resume its work to extract new fruit. and the cycle goes on...

Figure 4.7: state diagram for Routing module

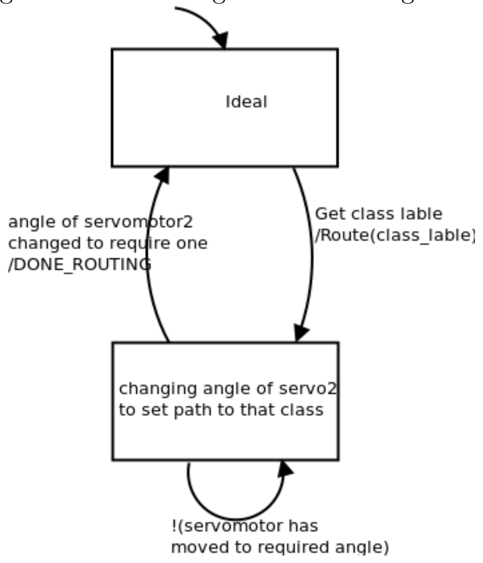
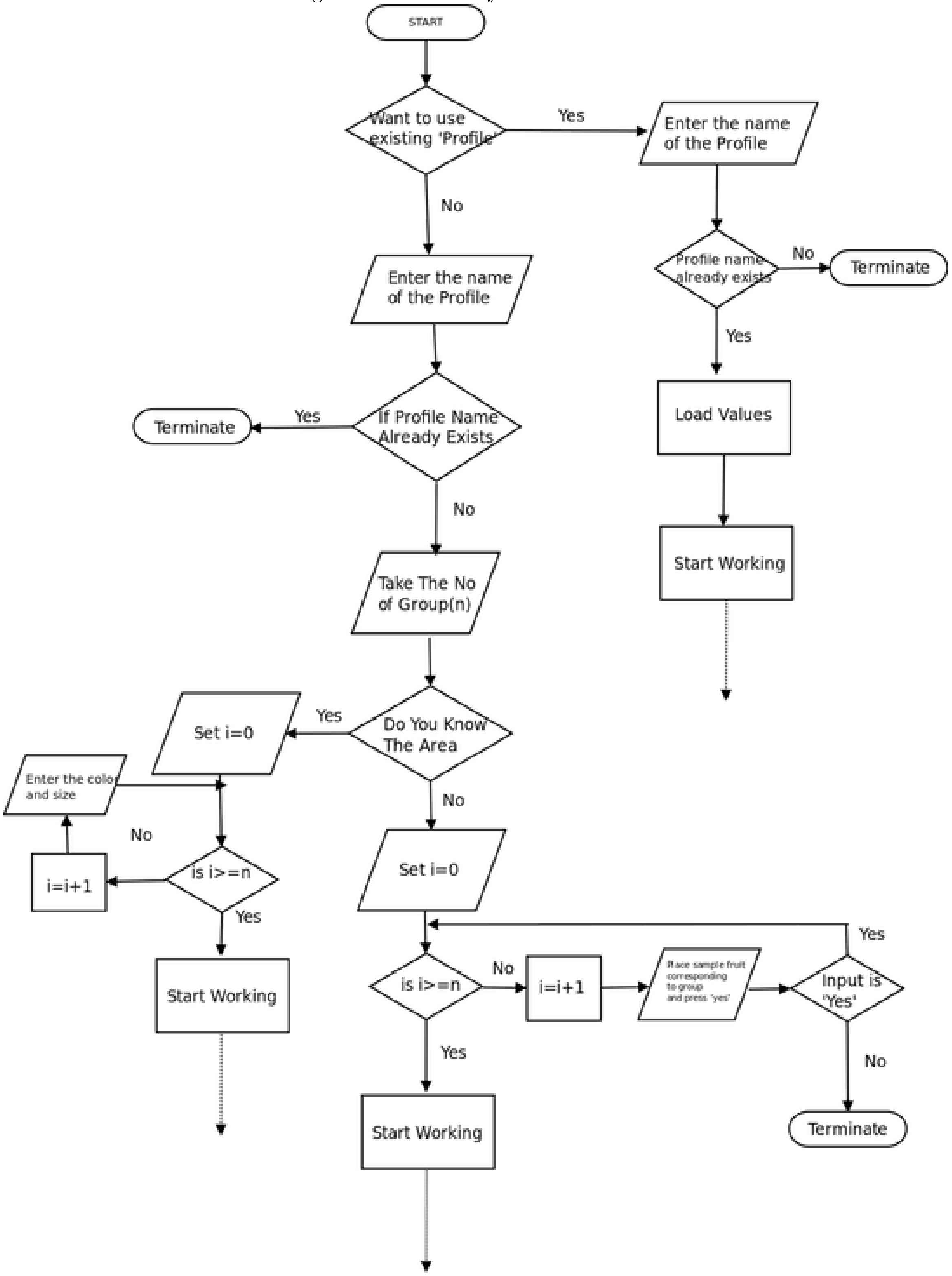

Ideal
angle of servomotor2 changed to require one /DONE_ROUTING
Get class lable /Route(class_lable)
changing angle of servo2 to set path to that class
!(servomotor has moved to required angle)

Figure 4.8: Overall system Flow Chart



4.4 Functions and their uses

Functions	Usages
init_devices();	Initialize all components in firebird
cli()	clears all global interrupt
sei()	sets all global interrupt
myfun()	Introduces some delay in the system
lcd1_cursor(row,column)	Place cursor at given row and column on LCD display
lcd1_string(string)	Write data on LCD display
lcd_print(row,column,data,length)	to print any input value up to the desired digit on LCD
lcd1_port_init	Initializes the LCD ports
lcd_set_4bit()	Reset LCD
lcd_port_config	configure LCD port
adc_init()	Initialize ADC hardware
ADC_Conversion(channel)	This Function returns the analog vale of a channel
adc_pin_config()	ADC pin configuration
print_sensor(row,coloumn,channel)	This Function prints the Analog Value channel at row and coloumn Location.
uart2_init()	Function To Initialize UART2 char size: 8 bit
ISR	ISR in Firebird for receving interrupt from PC
servo_port_init()	Calls both servo2_pin_config() servo1_pin_config()
servo2_pin_config()	Configure PORTB 6 pin for servo motor 2 operation (motor used for barrior gate)
Servo_2()	Function to rotate Servo 2 by a specified angle in the multiples of 1.86 degrees
servo1_pin_config()	Configure PORTB 5 pin for servo motor 1 operation (motor used for routing)
Servo_1()	Function to rotate Servo 1 by a specified angle in the multiples of 1.86 degrees
motion_init_devices()	Function to initialize DC motor
motion_pin_config()	Function to configure ports to enable robot's motion

motion_port_init()	Initialize DC motor
left_position_encoder_interrupt_init()	Interrupt 4 enable
stop()	Stops movement of DC motor
backward_mm(distance)	Moves piston backward by distance mm
forward_mm(distance)	Moves piston forward by distance distance
motion_set(direction)	Function used for setting motor's direction
route(fruit_class)	Routes the fruit to its destination by rotating a servo motor
barrior_gate_down	The barrior for the fruit lowers to hold the fruit before deciding its class level
barrior_gate_up	The barrior for the fruit opens for the fruit to route to the proper destination
senseAll()	Reads values of two IR proximity sensors
ir_port_init	IR proximity port initialize

5 Testing Strategy and Data

This system is completely modular. So testing of each component individulally is essential. We First tested the each unit(module) separately. Then we performed integration testing for each component. And then we tested whole system with test data.

5.1 Testable components

Dispensing Module: We tested whether this module, when required, pushes only one fruit at a time to IP module. We also tested the case when two fruits falls from the container into the tube holding piston. We also tested the case when racks present in rack and pinion mechanism slips.

5.2 IP Module

This module has one tuner module which helps in setting the appropriate threshholds required. We tested the tuner component in different lightening condition with wide variety of test data. We tested the component which decides the class label based on fruit attributes, separately.

5.3 Routing Module

We tested this component separately by giving commands(class label) manually to this part and tested. Communication module

5.4 Test Data

Input: Lemons of different sizes and colors

5.5 Evaluation Metrices

Throughput: 8-10 secs/fruit

Modular: Yes

Reusable: Yes

SuccessRate: is grater than 90%

5.6 Test Result and conclusion

If fruits are within the constrained size and external environemnt like lightenings is as per t he requirement of the system then its accuracy is greater than 90%. The error we are getting is because of IP module which doesn't perform well in few boundry cases where the color of lemons are not so separable. Accuracy can be increased by improving the accuracy of feature extraction part of IP module.

6 Design Challenges and Open Issues

6.1 Dispensing One fruit at a time from container

This was the main design challange. Buiding a system which can deterministically ensure that at every time only one fruit despenses was the toughest job. We achieved this task whith the help of two IR sensor (as mentioned in section 4.1).

6.2 OPEN ISSUES

1. Currently this system doesn't behave properly when the container is empty.
2. This system is not fault tolerant,e.g. If a fruit with smaller size(outside from the defined range) is placed into conatiner then system doesn't handle this case gracefullty.
3. IP module is not robust for all type of camera and lightening condition. It is very susceptible to quality of camera used. We performed all our tests with 'Logitech' web cam.

7 Innovation

1. This system **can be trained using some ltraining objects** so that it can decide the various thresh-holds and automatically classify the object accordingly.This feature will be very useful while enhancing the system to detect the defective objects. Learned parameters can be saved in different profiles which can be deployed easily. We have tested it and sucessfully demonstrated it.

2. The system not only classify the object but it also tells it's **confidence level for that perticular classification**. This is very useful in identifying the boundary cases where system may not classify the object properly.

8 Reusability

While designing the system we have taken care of reusability. Our designe and implementation are modular and reusable. Below section will briefly describe the reusable component. **Reusable Components**.

1. **H/W module**

   (a) **Racks and pinion mechanism implementation** This module can be used whenever any controlled linear motion is required. This system converts circular motion of geared dc motor into linear motion with the help of shaft encoding wheel.

   Known Bugs No

   (b) **Dispensing System** This mechanism can be reused to extract one specific type of object at a time. Two IR proximity sensors work in cordination to ensure that only one fruit is being dispatched at any time.

   Precondition to use

   Object should be nearly circular in shape

   Maximum and Minimum diameter decides the diameter of pipe used in this module. Object should be hard enough so that it will not get damaged when it gets into contact with other hardwares. For details refer to section **4.1**

   Known Bugs Doesn't handle the case of empty container.

   Sometimes fruit may not come out of the system even though they are in funnel as they may block each other's path. This can be easily prevented by adding one vibrator to the funnel.

   (c) **Routing System** This system can be used when dynamic routing is required.

   Precondition to use The width of the bin which collects the object after routing should be greater than the width of the router.

Either Object which needs to be routed should have very less friction so that it can slip due to gravity or should be of circular shape so that it can roll.

Known Bugs No

2. **S/W module**

   GroupManager This module can be used to get class label of any object given its attraibutes into different groups having their own properties. **Known bugs** No

   Pre condition Property of group and object should belongs to same feature space

9 Future Work

1. Currently, This system doesn't handle the case of empty container. Possible solution: It can be achieved with the help of IR sensor and Vibrator
2. Currently this system doesn't consider power usage, In future , it can be made power efficient
3. Throughput of the system is low. It can be increased by improving the Dispencing module.
4. Currently we are considering only Lemon as a fruit whose minimum and maximum size are constrained by the diameter of the tube in Dispencing module.
5. New attribute can be added for sorting like weight (by adding appropriate sensor).
6. Currently while deciding the class label of fruit we are considering the front part of fruit which is observed by camera. Part of fruit which is not in front of camera doesn't play a role in deciding it's label, It may impact the decision while deciding whether the fruit is defective or not. So, some mechanism can be added in IP module which can rotate the fruit in its place. So that camera can take the decision based on complete informaton.
7. It can be enhanced by adding some mechanism o detecting defective fruit.

10 Conclusion

This project demonstrates that it is possible to automate the tidious and important job of fruit sorting. Although our current system does not provide all functionality and support for all kinds of fruits, but the work can be improved to address those issues by modifying some part of the system. Embedded system are subject to inaccuracy and failure. But we have strived our best for avoiding these issues and build an accurate, modular, reusable system.

References

[1] E-yantra website. http://www.e-yantra.org.

[2] Firebird v atmega2560 robotic research platform hardware manual. iit bombay & nex robotics pvt. ltd.

[3] Firebird v atmega2560 robotic research platform software manual. iit bombay & nex robotics pvt. ltd.

A Appendix

Hardware Required

Firebird V

Camera,

Ambient white light source,

PC

Serial Communication cable (USB to serial converter between PC to firebird)

1 DC motors

2 Servo Motors

2 IR proximity sensors

Other hardware processing tools