Automated Boba Machine

Timothy Ko, Jordan Wu, Hunter Huynh
Team 49

ILLINOIS
Electrical & Computer Engineering
COLLEGE OF ENGINEERING
Introduction

- The Boba Craze
- All made Manually
  - Food waste
  - Inconsistent taste
  - Can we make this process more efficient?
Overall Objectives

- Accurate dispensing of ingredients to create best drink possible every time
- Allow wide permutation of drinks and ingredients
- Easy to use interface that requires little technological literacy
Concrete Objectives

- Have at least 2 dispensers
  - milk tea (liquid)
  - tapioca pearls (solid)
- Dispense a user-specified amount of milk tea
  - less than ±10% error in mass
- Web interface to control the amount of liquids/solids dispensed
Original Physical Design
The machine
Block Diagram
Modular Design

Standalone design of each module allows customer (shop owner) to
  - pick and choose whatever combination of dispensers

Benefits

• Accommodates wide permutation of drinks
• Lower cost in long run
Modular Design

- **Wireless**
  - Dispensers placed wherever convenient.
  - No need to run wires.

- **Over-the-air software updates**
  - Updating/Adding features
    - Better dispensing efficiency
  - Same software for all microcontrollers
  - Easier for shop owner
Web Interface

Welcome to our Automated Boba Machine!
Check out our Design Document here.

Customize your drink:

- **Milk Tea Amount (grams):** enter integer inputs. Ex: 200
- **Boba Amount (num of spins):** enter integer inputs. Ex: 4

Buttons:
- **Start**
- **Test API**
- **Get IPs**
- **Setup**
- **Liquid Start**
- **Liquid Stop**
- **Liquid Ping**
- **Solid V=1**
- **Solid V=0**
- **Solid Ping**
- **Get Load**
- **Load Setup**
- **Load On**
- **Load Off**
Software Logic

Main Server starts with given configuration

Main Server sends action instruction to microcontroller

Microcontroller unpackets packet

Request is "OFF"

Microcontroller sends stop signal to respective dispenser

Request is "ON"

Microcontroller sends start signal to respective dispenser

Main Server gets OK to continue

Microcontroller beings polling load sensor data

Load Sensor unpacket packets

Main Server checks if data indicates finished

Sends load sensor data response

Yes

Finished

No
Other Software details

- HTTP Requests
  - Latency 100ms Round trip w/o DNS or SSL
- Automatically searches for modules & configures them
- Proportional control on liquid dispensing
  - based on load & desire load
- Safety Guards
  - Stops entire process whenever one of the Modules dies
Microcontroller

- ESP8266-01 Module
  - Only 2 GPIOs
  - Step down to 3.3V via TSR1-2433

- MicroPython Firmware
  - WebREPL interface
  - Eliminated USB serial programming
Weight Sensing - Load Cell

- Accurate
- Outputs a tiny voltage increasing with weight

![Graph showing Mini Load Cell Output (V) vs. Weight (g)]
Weight Sensing - Signal Processing

- An HX711 chip amplifies its input, then converts it to a digital value
Liquid Dispensing Mechanism

Objectives:
- Dispense liquid with +/- 10% error in mass

Result:
- +/- 1mL for any mass.
  - < 1% error in mass in typical scenario
Liquid Dispensing Issues

- Original design had TIP120 NPN Darlington transistor as suggested by the datasheet.

<table>
<thead>
<tr>
<th>$V_{BE}$</th>
<th>$V_{CE}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0V</td>
<td>12V</td>
</tr>
<tr>
<td>3.3V</td>
<td>~4V</td>
</tr>
</tbody>
</table>

- $V_{CE(ON)}$ was around 4V.
  - 3A current draw of solenoid
  - Package was dissipating 12W. (Very Hot)
Liquid Dispensing Solutions

- Use 2 RFP30N06LE
  - 1 as gate driver.
- Resistor choice crucial to ensure staying under \( V_{GS(MAX)} = 10\text{V} \) and saturation.

<table>
<thead>
<tr>
<th>( V_{GS(Driver)} )</th>
<th>( V_{GS(Valve)} )</th>
<th>( V_{DS(Valve)} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0V</td>
<td>~5.8V</td>
<td>~1V</td>
</tr>
<tr>
<td>3.3V</td>
<td>0V</td>
<td>12V</td>
</tr>
</tbody>
</table>

- Reasonable power dissipation of 3W.
Solid Dispensing Mechanism

$150 later...
Solid Dispensing Mechanism

- Rotates tube with servo motor
- 3.3V logic level can control 5V servo
  - Min = 4% Duty at 50Hz
  - Max = 12% Duty at 50Hz
Solid Dispensing Issues

- Cost
- Loose tolerance
  - Liquids leak out the side
- Clumped up Boba causing dispenser to jam up
  - Micro-servo in original design destroyed
  - Upgraded to standard-size servo for more torque.
Boba Consistency....
Solid Dispensing Issues (cont.)

- Boba blocked each other laterally during dispensing
  - Stopping each other from entering rotating compartment.
Solid Dispensing Solutions

- Dispenser made using injection moulding for low cost
- Leakage prevention
  - Tighter tolerance in design
  - Seal with rubber O-rings.
  - Stronger servo motor to account for the friction.
Solid Dispensing Solutions (cont.)

- Extra liquid dispenser on top of solid dispenser to flush Boba and prevent clumping.
- Vibration mechanism to “shake up” the Boba to prevent lateral jamming.
Conclusion

▪ Our objectives have been met.
  – Has at least two dispensers working.
  – A web interface can control amounts of liquids and solids dispensed.
  – Machine can dispense user-specified amounts of liquid and solids within ±10% error.
Thank You