Where's The Milk?

B0: Lucky Wavai, Allen Ding, Takshsheel Goswami
18-500 Capstone Design, Spring 2022
Electrical and Computer Engineering Department
Carnegie Mellon University

Product Pitch

System for tracking inventory on shelves and foot traffic within aisles in real time. Intended to be used by commercial grocery stores, and even the average joe with shelving in a pantry, shed, or garage. If you need to know whether milk is stocked in aisle 6 like it is supposed to be, you don't need to leave your desk to find out.The systems critical requirements are 1) detect the presence of people in aisles to determine if changes may have occurred 2) Detection of presence - via background subtraction - and items on the shelves via - via scale invariant feature transform.

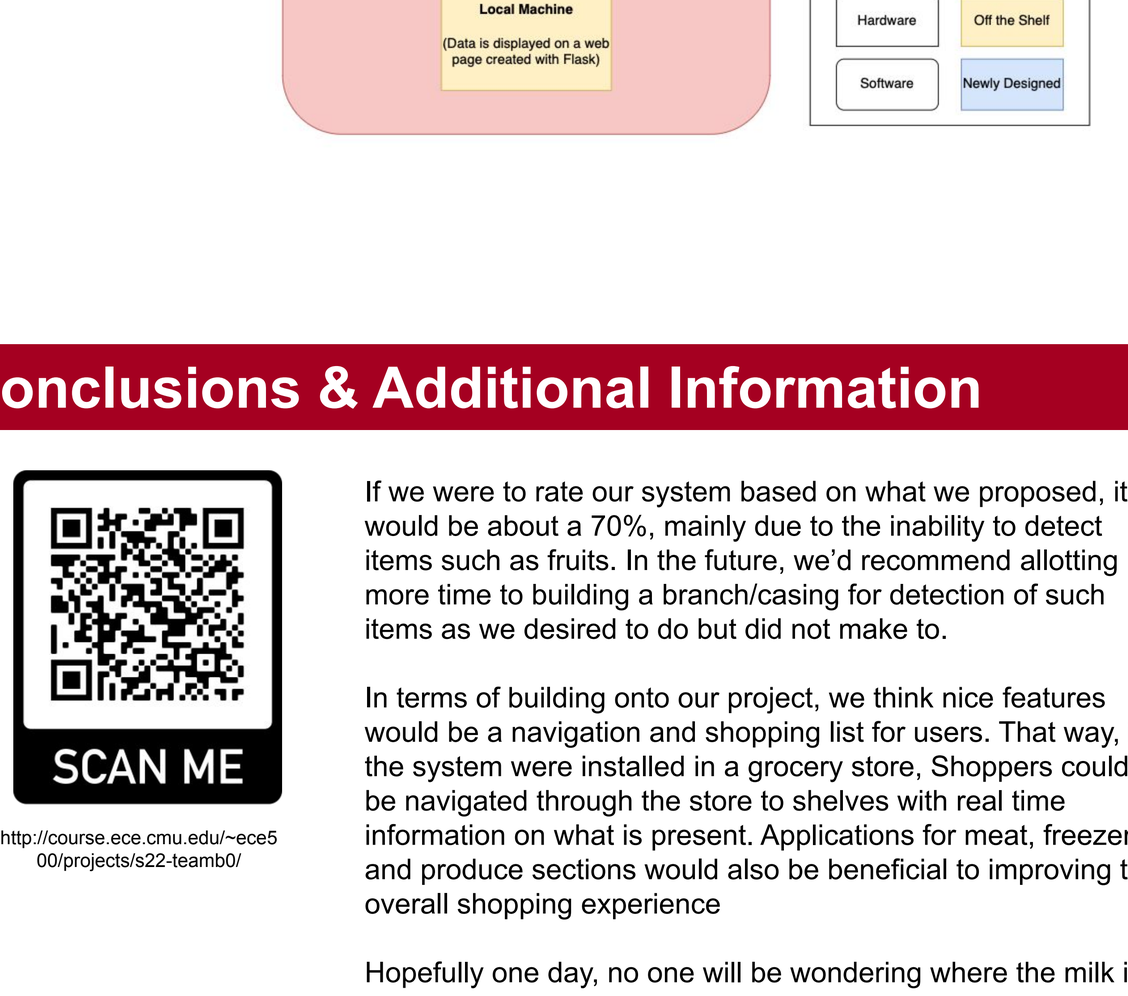
We surpassed our goal 90% of motion in the aisle as desired given a 24 frame threshold which makes the trigger for item detection only consider motion of at least 5 seconds (average minimum time individual takes to replace, move, or take an item from testing)

Motion detection triggers a picture of the shelf to be taken, then item presence detection to run on the image. Some objects (items with limited distinct features to extract such as apples and oranges) proved to present issues for our system, and we were unable to meet our use case required thresholds (60% correctly detected items). We pivoted to move forward without fruits due to the limitations of SIFT and time constraints as we wanted to begin integration. To counter this issue in the future, a color detection algorithm might be beneficial.

Our wireless system uses HTTP requests to communicate with the web app. The web application displays results of detection information for each shelf section the user sets up.

System Architecture

In reference to the block diagram below, here is a high-level representation of our overall system architecture. Primarily, we have three subsystems: Wireless transfer, Object detection, and the Web application.



Conclusions & Additional Information

SCAN ME

http://course.ece.cmu.edu/~ece500/projects/s22-teamb0/

If we were to rate our system based on what we proposed, it would be about a 70%, mainly due to the inability to detect items such as fruits. In the future, we'd recommend allotting more time to building a branch/casing for detection of such items as we desired to do but did not make to.

In terms of building onto our project, we think nice features would be a navigation and shopping list for users. That way, if the system were installed in a grocery store, Shoppers could be navigated through the store to shelves with real time information on what is present. Applications for meat, freezer, and produce sections would also be beneficial to improving the overall shopping experience

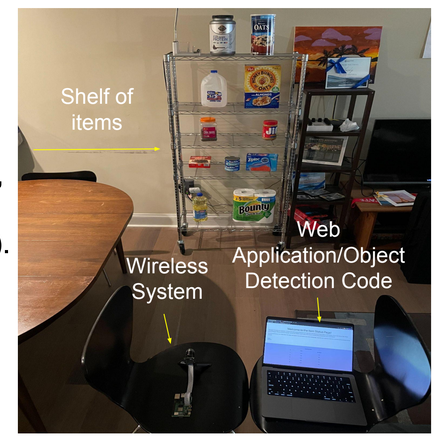
Hopefully one day, no one will be wondering where the milk is.

System Description

Wireless Transfer: For our motion detection subsystem, we are performing a technique called background subtraction. Background subtraction extracts a changing foreground image from a stationary background. We use a Gaussian Mixture model based algorithm called backgroundSubtractorMOG2 to do this. In accordance with a small Flask app on the RPI, the wireless system and main Flask app can communicate with each other through HTTP requests.

Object Detection: The item object detection subsystem determines which items are present in the shelf section and renders a detection image with labeled bounding boxes around what was found. First the subsystem receives an image of the full shelf section assigned to it. Then a sliding window generator function is used to continuously yield segmented regions of the full shelf that are the size of items we expect to find in that shelf area i.e. size of a cereal box. In each sliding window, the subsystem uses SIFT to generate keypoints and descriptors of the window. Then, the window keypoints and descriptors are matched against a stored set of keypoints and descriptors that are mapped to trained items we expect to find using FLANN for matching and Lowe's Ratio Theorem for filtering out excessive false positive matches. When items meet threshold requirements, they are marked as present, and their sliding window coordinates are also stored. Finally, using the item presence, and the stored locations, the full shelf image is modified with bounding boxes using the stores locations, and the presence and render image are sent to the web application.

Web Application: The website is built using the Flask framework and a SQLite3 database, along with many other software packages. The primary purpose of the web application is to host data about the aisle(s) to the user and serve as the integration center by being connected to the other two subsystems. To give a rundown of the control flow of the system, the user first is presented with the options to sign in or register as a new user. Once that is completed, the user chooses how many aisles they want to create (can be edited later), the items that they want to be matched with each aisle (items are from a fixed set), and what camera(s) they want assigned to each aisle (for the purposes of our project, we will just have one camera). Afterwards, the user is directed to a homepage where the aisles that they chose to create are available to view. The user can then click on an aisle from the homepage which will direct them to a page that displays the presence of items (shown in the image).



System Evaluation

Motion Test: Motion
- Ran the motion detection algorithm with a live feed camera
  - Tweaked bounding box dimensions until detection worked only for objects greater than 3 ft by 3
  - Tweaked frame threshold until motion detection trigger occurred only for motion of at least 5 seconds which we determined to be the minimum amount of time the average individual takes to move an item and leave the shelf area

Integrated Detection Testing
- Combined motion detection and presence detection algorithms to test functionality of trigger mechanism between the two and tested the times of one iteration with fixed set of items (11)

Presence Testing
- Series of Tests with single item images to determine which key point and descriptor generating algorithm worked best for our use case by comparing the false positive and correct matches of various items (*see **Fig 1** for example)*
- Series of tests with full shelf and image segmentation where we set all but one of the required thresholds constant, and incrementally ran a full shelf test by changing only one of the thresholds per series, and recorded correct matches, missed matches, and false positives (*see **Fig 2** for example*)

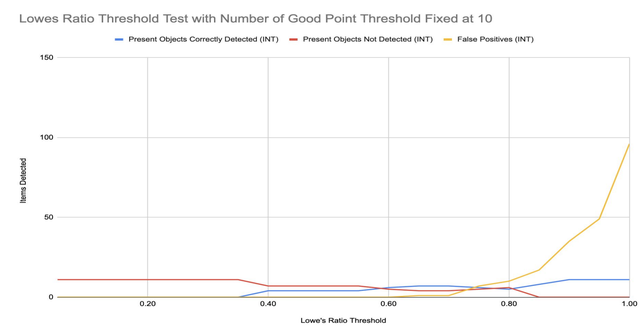
User Experience Testing (Web Application)
- We ensured that the following (and more), were successful:
  - User auth/password encryption
  - Cross-checking/cleaning form inputs
  - No HTTP errors
  - Limiting visibility of pages based on user credentials
  - Ease of access/UX on each page

Fig1. SIFT vs ORB Test Example with other thresholds held constant: Num Trees = 5, Num Checks = 50, Lowe's Ratio Threshold = 0.5, Num Good Matches Threshold = 10 to find the optimal range for lowe's ratio ratio threshold

Test Item	SIFT Max Number of Matches to Correct Item	SIFT Max Number of Matches to An Incorrect Item	ORB Max Number of Matches to Correct Item	ORB Max Number of Matches to An Incorrect Item
apple	0	7	1	0
bread	10	1	0	0
broccoli	0	1	0	0
cereal	133	0	0	1
chips	10	3	1	0
eggs	0	1	0	0
lemon	0	2	0	0
milk	1	1	0	0
orange	0	5	0	0
oreos	75	1	2	0
peanut butter	57	26	0	0
potato	0	1	0	0
soda	10	1	0	1
tomato	0	3	0	0
case of water	0	1	0	0

Parameters Kept Constant
Flann Number of Trees: 5
Flann Number of Checks: 50
Lowe's Ratio Threshold: 0.5
Number of Good Matches Threshold: 10
Train & Test Images

Fig2. Lowe's Ratio Threshold Test Example with other thresholds held constant: Num Trees = 5, Num Checks = 100, Num Good Matches Threshold = 10 to find the optimal range for lowe's ratio ratio threshold



Electrical & Computer ENGINEERING
Carnegie Mellon