Inventory Management of the Refrigerator's Produce Bins Using Classification Algorithms and Hand Analysis

Sarah Morris
Electrical and Computer Engineering
University of Louisville
Louisville, KY
svmorris9785@gmail.com

Dr. Karla Conn Welch
Electrical and Computer Engineering
University of Louisville
Louisville, KY
karla.welch@louisville.edu

Michael Schroeder
Advanced Systems
GE Appliances
Louisville, KY
michael.schroeder2@geappliances.com

***Abstract*—Tracking the inventory of one's refrigerator has been a mission for consumers since the advent of the refrigerator. With the improvement of computer vision capabilities, automatic inventory systems are within reach. One inventory area with many potential benefits is the fresh food produce bins. The bins are a unique storage area due to their deep size. A user cannot easily see what is in the bins without opening the drawer. Produce items are also some of the quickest foods in the refrigerator to spoil, despite being temperature and humidity controlled to have the fruits and vegetables last longer. This research addresses the challenges presented by getting a full inventory of all items within the produce bins by observing if the hand can provide useful information. The research proposes that all items must go in or out of the refrigerator by the main door, and by using a single camera to observe the hand-object interactions, a more complete inventory list can be created. As an initial proof-of-concept, two types of produce, an apple and an orange, will be used as a testing ground. The accuracy of the hand analysis (e.g., hand with apple or orange vs. hand empty) was determined by comparing the model predictions to a 301-frame video with ground truth labels. The hand analysis system was 87% accurate for classifying an empty hand, 85% accurate on a hand holding an apple, and 74% accurate on a hand holding an orange. The system was 93% accurate at detecting what was added or removed from the refrigerator.**

***Index Terms*—computer vision, produce recognition, hand recognition, hand-object interaction**

I. Introduction

Analyzing systems to prevent food spoilage could address a costly problem in the homes of consumers. It is estimated that 21% of the food purchased by consumers in 2010 went to waste, resulting in a loss of $114.9 billion dollars in the US, or $371 per person per year [10]. Food waste within retail has long been identified as an issue. To reduce waste, retailers and restaurants use technology to track their food stores, utilize their stock more efficiently, and help make decisions when ordering new items [13]. This process, known as inventory management, has only recently been extended within the home, and more specifically the refrigerator, to combat consumer waste. Inventory management in the refrigerator is the process of maintaining an accurate record of the contents inside. Important information for inventory management includes what an item is, where it is located, how long it has been in the refrigerator, and how long the item will stay fresh. Additional technology within the refrigerator itself can make an automated inventory management system, which can lead to a more-informed consumer without overburdening them with food-tracking tasks, and provide automated information on the contents of the refrigerator to reduce food spoilage and waste.

The produce bins are deep and difficult to see inside; items can easily get lost and forgotten under layers of other produce. Because of this, the produce bins offer a unique challenge for inventory management, and a great starting point for an overall system in the refrigerator. Figure 1 shows the contents of a typical consumer's produce bin. The items on top are easily distinguishable, but the layers underneath are covered and not visible. One would need to remove the top layer to see underneath, but doing so is cumbersome and inefficient. What if the refrigerator could tell you what was underneath those layers without you needing to painstakingly log and track all your produce? This research aims to explore that question by observing if the hand can provide useful information.

The work conducted for this hand analysis study consists

Fig. 1. An image of a typical consumer's produce bin.

This research has received partial funding from the National Science Foundation (NSF) (Grant Number: EPSCoR 1849213).

of two main parts. The first was to create a model that could identify hands within the refrigerator. The model needed to be robust enough to detect different hand sizes, colors, orientations, and partially-occluded hands. The accuracy of the model was determined by comparing ground truth detections for 185 new images to the model versus the detections made by the model. The second part was to analyze the detected hand to determine if it is holding a type of produce or empty, and track if the produce is added or removed from the refrigerator. The accuracy of the hand analysis (e.g., hand with apple or orange vs. hand empty) was determined by comparing the model predictions to a 301-frame video with ground truth labels.

II. Related Work

This work draws upon previous work on inventory management in the refrigerator, produce datasets and classifiers, and hand detection techniques.

A. Inventory Management in the Refrigerator

The current proliferation of wireless, internet-connected appliances has opened up new solutions to inventory management. Often given the moniker "smart," the connected devices give consumers unprecedented control over their products [22]. Smart appliances allow users to control their units remotely, get instant software updates, and allows appliance makers to add innovative technology. Internet-connected refrigerators have allowed researchers to install barcode scanners, RFID readers, scales, cameras, and other sensors to develop inventory management systems [16] [25] [15] [23] [21] [8] [24] [12] [7].

Research has shown promising results using machine learning techniques to automatically identify items within the refrigerator. The introduction of large produce databases, like VegFru and Fruit-360, have enabled CNN models to be applied to produce classification [9] [18]. A challenge specific to fruits and vegetable is there is a lot of variability even within the same class of food item. Research has shown that models trained on the large produce databases perform well on the training set, but are much less accurate in applied settings due to the produce variability. A study found that supplementing the data with application specific images improved accuracy [14]. The studies have shown that CNNs are successful at accurately identifying produce items, but much of the research remains academic and has not been applied to the development of consumer products.

B. Hand Detection

Hand analysis for this research consists of detecting the hand within the frame, and determining what the hand is holding. In computer vision, much of the hand analysis research focuses on recognizing hand gestures, which could then be used for human-computer interaction, sign language recognition, and hand-object interaction for virtual reality systems [3]. Less work has been done to use the hand as an anchor to detect objects, but Amazon is researching this currently as a way to implement grocery stores that automatically know what a user has purchased [28]. Early hand gesture recognition systems used either sensors to relay location information or skin-based thresholding to detect and extract the hand and arm region [6] [11] [1]. The introduction of large hand datasets like Oxford and EgoHands have allowed CNNs to be used for hand detection [17] [2]. The creators of the EgoHands dataset were able to get over 80% accuracy using a CNN trained on EgoHands. Unlike the color-based skin thresholding, CNN models are not dependent on color differences within an image and can detect hands even when the background is similarly skin-colored.

Much research has gone into hand recognition, but there remain few real-world applications outside of gesture recognition. This research aims to explore hand-object interaction methods to assist in an inventory management system for fresh food in produce bins.

III. Methods

The research introduces an algorithm to automatically detect add events or remove events. The work builds upon the prior work for hand detection and produce classification. The objective of the system is to analyze the hand-object interaction of a consumer as they interact with the produce bins.

A. Inventory Detection Pipeline

The inventory detection pipeline utilizes two models: one to detect hands and one to classify a hand as either empty, holding an apple, or holding an orange.

The hand detection model was trained using the TensorFlow Object Detection API [27] [19]. The faster_rcnn_inception_v2_coco model was used as the starting point for training [26]. The model was first trained on data from the EgoHands dataset [2], and then fine-tuned on a dataset created specifically for the application.

The produce classification model was trained in Keras [5]. The model uses the VGG16 architecture, and pretrained ImageNet weights. The first 25 epochs froze all layers but the final fully-connected layers. The final 100 epochs froze all but the last convolutional layer and the fully-connected layers [20]. The input to the model was color hand images, which were cropped and resized to 224 x 224 pixels.

B. Implementation Details

The dataset to train the hand detector was created by annotating a sequence of images of a single user unloading several different produce items. The camera collecting the images was located in the front, center area in the ceiling of the refrigerator. Figure 2 shows an example image from the camera. The user was a single white-male in his early-30's unloading groceries to simulate a consumer reaching in and out of the refrigerator. The user placed items in the refrigerator as he would when he unloaded his grocery bags, usually one item at a time. Once all items were in the unit, the user began taking items out of the produce bins and putting them back in different locations in the refrigerator, such as the bottom

Fig. 2. Example image from the camera used for data collection.

shelf. The moving of items ensured a diversity of hand images holding different items. The new hand dataset consists of 813 images with 1,294 bounding box annotations for the hand class.

The dataset to train the produce classifier was created by extracting hand images from the refrigerator video stream using the trained hand detector. Three different users, one right-handed male in his early 20's, one left-handed female in her early 20's, and one left-handed female in her early 30's, with three different skin tones were used to build the dataset. Individually, the users repeatedly placed a single apple or orange onto the refrigerator shelf, or into a produce bin. The hand detector would crop out all detected hands, and save them to a folder. An example image from the class *Apple* is shown in Figure 3. The class weights were adjusted so each class was weighted equally at training time. The class breakdown of the produce classifier is shown in Table I.

To verify the usefulness of exploring hand-object interaction, a program was developed to try to track the addition and removal of apples and oranges. To simplify the problem, single apples or oranges were used as opposed to a bag of fruit. The program was tested on videos of a single user adding and removing the fruit, and closing the refrigerator door between each interaction.

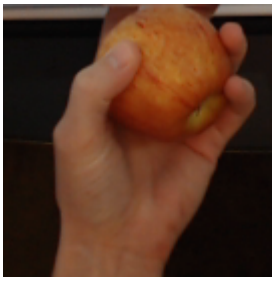

Fig. 3. *Apple* class training image for the image classifier.

TABLE I
TOTAL NUMBER OF IMAGES PER CLASS.

Class	Total images per class
Apple	361
Empty	455
Orange	236

The experiments, models, and logic were tested and validated on videos recorded within the refrigerator. The videos were analyzed frame by frame as a series of images. The video sequences were designed to mimic how a typical user would interact with their refrigerator.

IV. EXPERIMENTS

Multiple experiments were done to validate the accuracy of the hand detector and produce classifier, and prove feasibility of the inventory detection pipeline.

A. Hand detector

An initial experiment was done to validate the training pipeline within the TensorFlow Object Detection API. The test was done by training the model using the EgoHands dataset, and then comparing the results to published results on the EgoHands dataset [2] [4]. The faster_rcnn_inception_v2_coco model within the API was selected based on experiments comparing three different model architectures. In the interest of space, the results of the model comparisons will not be discussed in detail, but the faster_rcnn_inception_v2_coco model did yield the highest percent accuracy. The results of the initial experiment compared with published results are shown in Table II; the mean average precision for an Intersection over Union (IoU) of 50% or greater is reported. The first model in Table II was trained on the EgoHands dataset for this research, while the next two models are the results from published papers [2] [4]. Table II shows that the results are similar to published results, thus validating the training pipeline.

The next experiment was to see if a model would perform better if it was first trained on the large EgoHands dataset and then on the local dataset, or only trained on the local dataset. The results are shown in Table III, and show that the models perform similarly on the test dataset. Results for the two models on the validation video are shown in Tables IV-V and Tables VI-VII. The data shows that, despite similar training metrics, the model trained on the EgoHands dataset first and then on the local dataset detects 15 more hands in the validation video. The F1-scores are similar for both models,

TABLE II
PERFORMANCE RESULTS FOR VARIOUS MODELS, COMPARED WITH PUBLISHED RESULTS. *ORIGINAL PAPER DID NOT USE TF API SO ONLY HAVE A SINGLE MAP VALUE [2].

	Model Name	mAP @0.50IoU on EgoHands
Experimental Results	faster_rcnn_inception_v2_coco	0.975
Published Results	ssd_mobilenet_v1_coco [4]	0.969
	Sliding window CaffeNet* [2]	0.807

TABLE III
MAP@50IOU RESULTS FOR EACH MODEL ON THE LOCAL DATASET.

Model name	local dataset only	EgoHands then local dataset
faster_rcnn_inception_v2_coco	0.965	0.962

but recall is 0.94 for the EgoHands then local compared to just 0.85 for the local trained model.

Overall, the hand detector with a single hand class performed well on new data, but the model struggled to detect hands at the edges of the frames. An example of a missed hand is shown in Figure 4. The hand is blurry and holding an object, which could be why the hand was missed.

Based on the data in Table IV-VII, the model trained on the EgoHands dataset and then the local dataset with a single hand class was determined to be the best choice to maximize accuracy. The experiments in the following sections use the faster_rcnn_inception_v2_coco model.

B. Produce Classifier

The hand detector is used as an anchor to find regions of interest to send to the produce classifier. The classifier was trained on images of empty hands and hands holding apples or oranges. The precision and recall data at training time is shown in Figure 5. The data is after 25 epochs to warm up the fully connected layers, followed by 100 epochs to fine-tune the last convolution layer. The figure shows that the overall F1-score for the model is 98%.

The model was also tested on a video simulating the production application. The precision and recall data for the classifier is shown in Table VIII, and shows that precision and recall decrease for all classes when applied to new data. The apple class decreased the most from the training metrics, dropping from an F1-score of 0.98 down to 0.63. The orange class dropped from 0.99 to 0.85, and the empty class went from 0.97 to 0.93. The average F1-score across all classes dropped from 0.98 to 0.80. The confusion matrix for the classes is shown in Table IX. The "Unsure" class in Table IX was added only for the confusion matrix calculations. The class of some of the detected hands was difficult to determine even by a human, thus the "Unsure" class was created to not penalize the model unnecessarily. The "Unsure" class is set when none of the other class predictions are above 50%. The "Apple" class was the most common false positive class. "Empty" was most frequently mistaken for "Apple", followed by "Orange". "Orange" was incorrectly classified as "Apple" 41% of the time. "Apple" had the least amount of false negatives.

The image classifier was only 80% accurate in the actual application, but is the best performing of the experiments. The hand detector plus image classifier is used to develop the logic for determining if an item is being added or removed.

TABLE IV
PRECISION AND RECALL DATA FOR EGOHANDS THEN LOCAL DATASET (RCNN INCEPTION).

Class	Precision	Recall	F1-Score	Support
Hand	0.90	0.94	0.92	185

TABLE V
CONFUSION MATRIX FOR EGOHANDS THEN LOCAL DATASET (RCNN INCEPTION).

		Predicted: Hand	Predicted: No Hand
Actual	Hand	173	12
	No Hand	19	-

TABLE VI
PRECISION AND RECALL DATA FOR THE LOCAL DATASET (RCNN INCEPTION).

Class	Precision	Recall	F1-Score	Support
Hand	0.98	0.85	0.91	185

TABLE VII
CONFUSION MATRIX FOR THE LOCAL DATASET (RCNN INCEPTION).

		Predicted: Hand	Predicted: No Hand
Actual	Hand	158	27
	No Hand	3	-

Fig. 4. The hand detector did not detect the hand on the bottom edge of the frame.

```
[INFO] evaluating after fine-tuning...
              precision    recall  f1-score   support

       apple       0.98      0.98      0.98       133
       empty       0.97      0.96      0.97        79
      orange       0.97      1.00      0.99        76

    accuracy                           0.98       288
   macro avg       0.98      0.98      0.98       288
weighted avg       0.98      0.98      0.98       288
```

Fig. 5. Precision and recall data for image classifier at training time.

TABLE VIII
PRECISION AND RECALL DATA FOR IMAGE CLASSIFIER IN PRODUCTION APPLICATION.

Class	Precision	Recall	F1-Score	Support
Apple	0.50	0.85	0.63	20
Empty	0.99	0.87	0.93	118
Orange	1.00	0.74	0.85	23
Overall	0.83	0.82	0.80	161

TABLE IX
CONFUSION MATRIX FOR IMAGE CLASSIFIER IN PRODUCTION APPLICATION.

		Predicted			
		Apple	Empty	Orange	Unsure
Actual	Apple	17	1	0	2
	Empty	12	103	0	3
	Orange	5	0	17	1
	Unsure	0	2	0	1

Fig. 6. An example where the hand is out of frame, but the arm can be seen.

C. Inventory Detection Pipeline

The add/remove logic was developed only for the case of a single hand adding or removing one item (either apple, orange, or nothing) at a time.

Frames per second (fps) was determined to be an important variable for increasing accuracy of the detection pipeline. Too few fps and the hand can be missed going in or out; too many and the number of frames to process will become unnecessarily high and could cause errors. Experiments showed that ten to twenty fps was ideal to balance data and memory. Ten to fifteen fps were used to develop the add/remove logic because development on video playback was faster with fewer frames.

Knowing what frame starts and ends a user interaction is essential for accurately predicting inventory changes. The simplest way to track the beginning and end of an interaction is to observe when a hand enters and leaves the refrigerator. The refrigerator is deep and the hand often goes out of frame as it moves further inside, so it is necessary to also track the arm in the frame. If the hand moves further within the refrigerator and out of frame, the arm almost always remains within the frame. Figure 6 shows an example where the hand has moved further within the appliance and is out of frame. The arm is still in frame, and can be used to determine that the interaction is still in progress. Thus, a hand is said to be within the refrigerator if a hand or an arm, as in Figure 6, is present in the frame.

Creating the machine learning models was only half the challenge of this research. Once the hand detection and produce identification models were reasonably accurate, the task shifted to combining the models in a way to allow the computer to automatically extract useful information from the predictions and data. The proposed algorithm logic is shown in the flowchart in Figure 7. The algorithm begins (1) by running the hand detector on each new frame where the door is open. The hand_in_fridge variable is essential for telling the program when an interaction is complete, and when decisions should be made to update the inventory. If a hand or arm is not detected (2), the hand_in_fridge variable remains false, and the next frame is processed. If a hand or arm is detected (5), the hand_in_fridge variable becomes true. If the detected hand is not within the loading zone (6), the algorithm moves to the next frame. The loading zone is the area within the camera frame that encompasses the region most likely to contain a hand. An example of the loading zone is shown as the lime green box in Figure 4. The loading zone was implemented to reduce false predictions at the edges of the frames. If the hand is within the zone (7), the cropped bounding box image is passed to the image classifier. The prediction from the classifier is added to the object buffer (8). The object buffer tracks the object classifications for each frame. Each time the hand_in_fridge variable changes states, the inventory is updated and object buffer is reset. The algorithm continues until the hand_in_fridge variable changes to false (2). At this point, the algorithm makes decisions on what object was added and/or removed (3). The object buffer is split in two, with the first half of the list representing the added objects, and the second half representing the removed objects. The logic assumes that the identified objects at the beginning of the object buffer specify what is being added as the hand moves into the refrigerator, while the identified objects at the end of the buffer specify what is being removed as the hand moves out of the refrigerator. Experiments showed that the algorithm was more accurate when only the first and last three items of the buffer were used to determine the added or removed objects. For an object buffer with less than seven items, floor division (divided by two) is used to split the buffer. For example, a buffer of length five divided by two would be 2.5. The floor

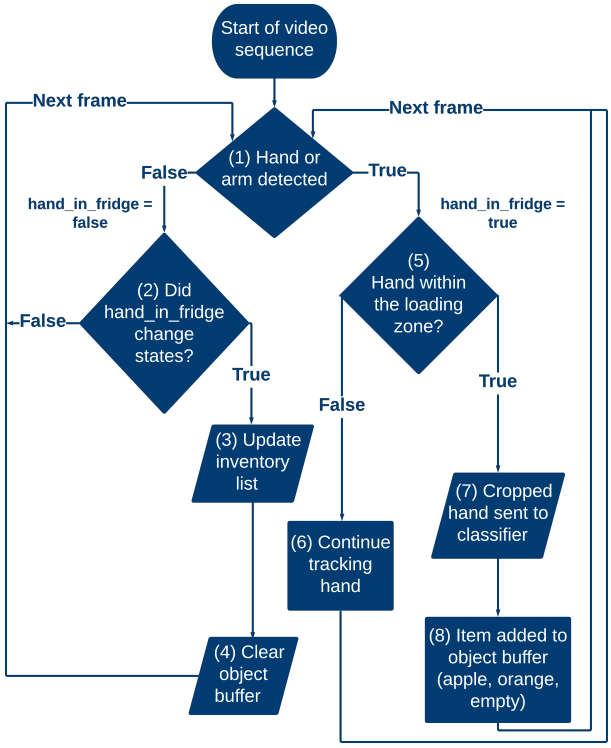

Fig. 7. Flowchart of hand analysis logic.

of 2.5 is two so the first and last two items of the buffer are used. Next, the most frequent item in each buffer was taken to be the item added or removed. Images from an add and remove interaction of a user removing an apple from the produce bin are shown in Figures 8 and 9, the loading zone is the area inside the lime green rectangle. The first three predictions in the object buffer (called the in buffer), and the last three predictions (out buffer) are shown in Table X. The predictions in the out buffer show why it was important to use multiple frames and average the predictions to determine what the hand is holding. The prediction for the last frame of the

TABLE X
FIRST AND LAST THREE PREDICTIONS FOR THE INTERACTION IN FIGURES 8 AND 9.

Buffer	Predictions per frame	Most frequent item
In Buffer	[‘Empty’, ‘Empty’, ‘Empty’]	‘Empty’
Out Buffer	[‘Apple’, ‘Apple’, ‘Empty’]	‘Apple’

interaction is incorrect, but because the algorithm is looking at the average of the last three frames, the add/remove predictions are still correct. The most frequent prediction in the in buffer is ‘Empty’, thus the pipeline correctly determines that nothing was removed. The most frequent prediction in the out buffer is ‘Apple’, thus the pipeline again correctly determines that an apple was removed. The interaction in Figures 8 and 9 shows that the algorithm is able to detect objects hidden from view within the bin - the apple is not visible in the frame until the hand takes it from the bin.

In the event of a tie in the buffer, the prediction is “Unsure”. Using an odd number of values for the in and out buffer ensures a tie is unlikely. It was found that the predictions at the top of the frame are not as accurate because the hand is usually mostly out of frame so the predictions are not reliable.

Despite the decreased accuracy for both the hand detector and produce classifier in a real-world application, Table XI shows that the overall logic performed well on a video simulating the production application. Interaction 12 was incorrect, with an orange being wrongly classified as an apple. Interaction 14 was inconclusive, with the object buffer showing [‘Apple’, ‘Empty’, ‘Unsure’]; so there was no most frequent item to predict. Interaction two and four represent an empty hand opening and closing the produce bin. The overall classification accuracy for the video was 93%.

V. DISCUSSION

The beginnings of an inventory management system using the hands and a produce classifier has been described from

Fig. 8. An early frame in the sequence, showing an empty hand moving into the refrigerator.

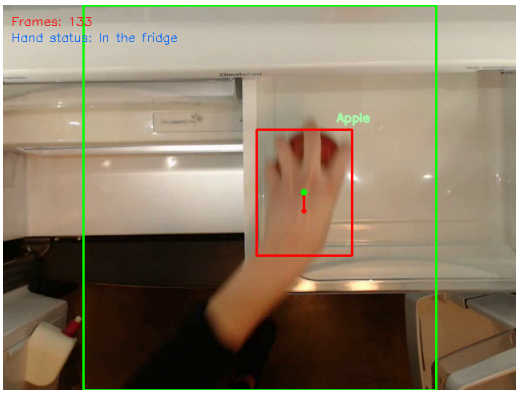

Fig. 9. Later in the sequence, the produce classifier detects an apple in the hand as the hands exits the refrigerator.

TABLE XI
PREDICTED AND GROUND-TRUTH CLASSES FOR 14 INTERACTIONS. INCORRECT PREDICTIONS ARE HIGHLIGHTED.

Inter-action	Ground-Truth		Predicted	
	Add	Remove	Add	Remove
1	Orange	Empty	Orange	Empty
2	*Empty*	*Empty*	*Empty*	*Empty*
3	Empty	Apple	Empty	Apple
4	*Empty*	*Empty*	*Empty*	*Empty*
5	Empty	Orange	Empty	Orange
6	Orange	Empty	Orange	Empty
7	Apple	Empty	Apple	Empty
8	Empty	Orange	Empty	Orange
9	Orange	Apple	Orange	Apple
10	Apple	Empty	Apple	Empty
11	Empty	Orange	Empty	Orange
12	Orange	Empty	Apple	Empty
13	Empty	Apple	Empty	Apple
14	Apple	Empty	Apple	Inconclusive

concept to realization. The objective of this research was to see if the hand can be used to identify objects being added and removed from the refrigerator produce bins. The research was successful at the objective, and presented a system that detects hands, identifies what a hand is holding, and automatically updates the inventory list based on what item was added or removed.

The research began by collecting images to use to train machine learning models. The images were then annotated, some with bounding boxes and another group labelled a single class per image. Bounding box annotations are a bottle-neck in developing machine learning models. To avoid annotation flaws and wasted time, annotating a small subset of the data and then training and testing early on in the development could be a good way to make sure the model is learning the correct information for the task. Seeing how the small subset performs could validate the annotation strategy, or illuminate issues. The final detection dataset consisted of over 3500 annotations of about 900 images, with over 20 different classes, including the hand class. The bounding box annotations were flexible enough that different classes could be developed just by filtering out different information in the annotations.

The final produce classifier dataset contained over 1000 images with three different classes. Larger application-specific datasets (i.e., datasets with thousands of images per class) would be needed to develop a more robust detector and classifier for a production application, but the small datasets proved feasibility. More image data from different users would help to develop machine learning models that generalize well to all use cases.

The hand detection model was successful at detecting hands in almost every frame in the validation videos. It was shown that, while the training metrics of different models may be similar, each model will have strengths and weaknesses when applied to new data. The best performing faster_rcnn_inception_v2_coco model detected 93% of hands in the validation video, but struggled with partial hands at the edges of the frame. The research showed that training the model first on a large dataset like EgoHands and then on the local dataset could improve performance by increasing the true positive detections. Later research showed that the arm was also important to detect, because in some cases the hand may be out of frame. More experiments should be done to train a model with a left/right hand class, as differentiating between hands is important for getting accurate results from the add/remove logic. Not enough testing was done to ensure the detector was robust to different hand sizes and skin tones, but initial testing showed positive results in identifying different types of hands.

The developed classifier was accurate enough for proof of concept. The classifier showed that an object could be detected even when held, and using the hand as an anchor to focus the classifier worked well. The benefit of using the hand as an anchor instead of just identifying the produce item is that if the item is not recognized, the system will still have a record of something being added or removed. The small dataset for the classifier meant that there were many false positives, with the apple class often being predicted for the other two classes. Weighting the class weights for the three items equally could also be a cause of the false positives. In the application, the hand is much more common than the produce items, and the training data should reflect that. Passing the hand bounding box to the classifier is limited to only objects that are contained within the box. Larger items, or items held by both hands at once, would not be correctly classified by this approach. Dynamic bounding box areas able to adjust to larger items, or using background subtraction techniques to extract a clean foreground image could solve the problem of larger items.

The add and remove logic, while limited to a single hand, performed well at automatically recognizing when an item was added or removed. The buffer holding the object classifications was used to make accurate predictions, despite the trained produce classifier not being very robust. Averaging the items in the buffer to find the most common object accounted for incorrect classifications, and could also provide information on how confident the algorithm was for the prediction. Only using the first and last few predictions to make decisions lessened the impact of the poor classifier performance at the top of the frame. The method illustrated the importance of a hand detector that detected every hand in every frame, where a lost hand meant lost object information. If the hand is missed in any of the first or last few frames, dividing the object buffer in half to determine what was added and removed becomes less accurate. Setting an optimal fps rate is also important to ensure there are enough frames to have at least three images of the hand moving both in and out. Updating the inventory list every time a hand exits the refrigerator was the best way to ensure that each add and remove interaction was captured. Detecting both the hand and the arm ensured that even when the hand moved out of the frame and deeper into the refrigerator, there was still an arm to let the system know the interaction was still in progress. A major limitation to the logic is that it was only developed for a single hand. The problem stems largely from

the challenge of being able to differentiate between two hands within the refrigerator. Once an accurate way to distinguish hands is developed, updating the add and remove logic would be straight-forward. Separate buffers per hand would track the objects in the hands separately. The logic to determine when a hand exited the refrigerator would need to be updated to be two distinct variables, one to update the inventory when the left hand exited the appliance, and one for the right. The timestamp information tied to each added object could be used to alert the user when a food is about to spoil. The timestamp plus information on the hand's location within the frame would allow the system to pinpoint where within a cluttered bin the item is located.

VI. Conclusions & Future Work

The research contained in this paper showed that a user's hand is a useful tool in identifying objects as they are added and removed from the refrigerator. Analyzing the hand provides another layer of information for the complex overall automatic inventory management system. Observing the hand-object interaction is especially useful to identify objects that would otherwise be hidden within the bin, under other items, or occluded from the camera view. Recording when the hands enter the refrigerator provides a timestamp that can be used to alert the user to items approaching their best-by-date.

The developed hand detector was 93% accurate on the validation video, detecting 173 of the 185 hands. The produce classifier correctly identified 17 of the 20 apples in the validation video, 17 of the 23 oranges, and 103 of the 118 empty hands. The add and remove logic correctly identified and updated the inventory information for 26 of the 28 add or remove events in the validation video.

The research above represents the initial phase of the hand analysis system. Future work for the research should focus on the following:

- Extend the add and remove logic to work for two or more hands
- Collect a larger and more diverse (e.g., different users of various ages, skin tones, hand dominance, etc.) dataset of application specific images and videos to improve the models and add more produce classes
- More testing to ensure the system is accurate for all users and is able to identify a wide range of produce items

The goal of future work should be to develop robust models and algorithms based on application specific images.

References

[1] N. Jiang H.-S. Tai D. Tretter B. Kang, K.-H. Tan and T. Q. Nguyen. *Hand Segmentation for Hand-Object Interaction from Depth map*. arXiv, 2016.

[2] S. Bambach, S. Lee, D. J. Crandall, and C. Yu. *Lending A Hand: Detecting Hands and Recognizing Activities in Complex Egocentric Interactions*. IEEE International Conference on Computer Vision (ICCV), 2015.

[3] James Davis and Mubarak Shah. *Recognizing Hand Gestures*. ECCV-94, Stockholm, Sweden, 1994.

[4] V. Dibia. Handtrack: A Library For Prototyping Real-time Hand TrackingInterfaces using Convolutional Neural Networks. 2017.

[5] Keras Documentation. Applications. https://keras.io/applications/, 2019.

[6] G. Dong, Y. Yan, and M. Xie. Vision-based hand gesture recognition for human-vehicle interaction. *The Proceedings of The International Conference on Control, Automation and Computer Vision*, 1998.

[7] FridgeEye. Turn any fridge into a smart fridge. https//fridgeeye.com/, 2020 Accessed 2020.

[8] E. Ganglbauer, G. Fitzpatrick, and R. Comber. Negotiating food waste: Using a practice lens to inform design. *ACM Transactions on Computer-Human Interaction*, 20(2), 2013.

[9] S. Hou, Y. Feng, and Z. Wang. Vegfru: A domain-specific dataset for fine-grained visual categorization. *IEEE International Conference on Computer Vision, Venice*, 2017.

[10] J. Hyman, H. F. Wells, and J. C. Buzby. The estimated amount, value, and calories of postharvest food losses at the retail and consumer levels in the united states. *Economic Information Bulletin*, 121, Feb. 2014.

[11] S. S. Kakkoth and S. Gharge. Real time hand gesture recognition its applications in assistive technologies for disabled. In *Fourth International Conference on Computing Communication Control and Automation*. Pune, India, 2018.

[12] D. Lee. Samsung and LG go head to head with AI-powered fridges that recognize food. *The Verge*, 2, Jan. 2020.

[13] C. C. Liang. Smart inventory management system of food-processing and distribution industry. *Procedia Computer Science*, 17:373–378, 2013.

[14] C. Liu, X. Wang, J. Ni, Y. Cao, and B. Liu. An edge computing visual system for vegetable categorization. In *18th IEEE International Conference On Machine Learning And Applications*. Boca Raton, FL, USA, 2019.

[15] S. Luo, H. Xia, Y. Gao, J. S. Jin, and R. Athauda. Smart fridges with multimedia capability for better nutrition and health. In *International Symposium on Ubiquitous Multimedia Computing*. Australia, 2008.

[16] S. Miniaoui, S. Atalla, and K. F. B. Hashim. Introducing innovative item management process towards providing smart fridges. In *2nd International Conference on Communication Engineering and Technology*. Nagoya, Japan, 2019.

[17] A. Mittal, A. Zisserman, and P. Torr. *Hand detection using multiple proposals*. Procedings of the British Machine Vision Conference 2011, 2011.

[18] H. Muresan and M. Oltean. Fruit recognition from images using deep learning. *Acta Univ. Sapientiae, Informatica*, 10(1):26–42, 2018.

[19] Python Programming. Introduction and use - tensorflow object detection API tutorial. https://pythonprogramming.net/introduction-use-tensorflow-object-detection-api-tutorial/, 2020 accessed April 4, 2020.

[20] A. Rosebrock. Fine-tuning with keras and deep learning. pyimagesearch https//www.pyimagesearch.com/2019/06/03/fine-tuning-with-keras-and-deep-learning/, accessed June, 2019.

[21] J. Rouillard. The pervasive fridge: A smart computer system against uneaten food loss. In *The Seventh International Conference on Systems*. 2012.

[22] M. Rouse. internet of things (IoT). https//internetofthingsagenda.techtarget.com/definition/Internet-of-Things-IoT, 2017 Accessed 2020.

[23] S. A. S. Intelligent refrigerator using artificial intelligence. In *11th International Conference on Intelligent Systems and Control*. Coimbatore, India, 2017.

[24] Samsung. Side-by-side refrigerator with family hub. https//www.samsung.com/us/home-appliances/refrigerators/side-by-side/26-7-cu-ft--large-capacity-side-by-side-refrigerator-with-touch-screen-family-hub--in-black-stainless-steel-rs27t5561sg-aa/, 2020 Accessed 2020.

[25] B. Son, C. S Han, Y.-T Jeon, and D.-H Lee. A rfid/nfc fusion based smart refrigerator for wellness service. *Sensors*, 2014.

[26] Tensorflow. Tensorflow detection model zoo. https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md, 2020 Accessed 4 April 2020.

[27] L. Vladimirov. Tensorflow Object Detection API tutorial. https//tensorflow-object-detection-api-tutorial.readthedocs.io/en/latest/, 2020 Accessed 4 April 2020.

[28] N. Wingfield. Amazon moves to cut checkout line, promoting a grab-and-go experience. *The New York Times*, Dec. 2016.