AIM3S: Autonomous Inventory Monitoring through Multi-Modal Sensing for Cashier-Less Convenience Stores

Carlos Ruiz
Carnegie Mellon University
Electrical & Computer Eng.
Moffett Field, California
carlosrd@cmu.edu

Joao Falcao
Carnegie Mellon University
Electrical & Computer Eng.
Moffett Field, California
joaodf@cmu.edu

Shijia Pan
University of California Merced
Computer Science & Engineering
Merced, California
span24@ucmerced.edu

Hae Young Noh
Carnegie Mellon University
Civil & Environmental Eng.
Pittsburgh, Pennsylvania
noh@cmu.edu

Pei Zhang
Carnegie Mellon University
Electrical & Computer Eng.
Moffett Field, California
peizhang@cmu.edu

ABSTRACT

A common pain point for physical retail stores is live inventory monitoring, i.e. knowing how many items of each product are left on the shelves. About 4% of sales are lost due to an average 5-10% out-of-shelf stockout rate, while additional supplies existed in the warehouse. Traditional techniques rely on manual inspection, per-item tagging using RFIDs, or human-in-the-loop systems such as Amazon Go. These approaches, while effective, either have poor accuracy, long delays between results or are cost prohibitive.

In this paper, we present *AIM3S* (Autonomous Inventory Monitoring through Multi-Modal Sensing) for cashier-less stores. To the best of our knowledge, this is the first fully autonomous system that fuses multiple sensing modalities. Utilizing weight difference on a shelf, visual item recognition in customers' hands and prior knowledge of item layout *AIM3S* monitors products picked up or returned without human-in-the-loop. We present results from a real-world setup with 85 items (33 unique products) replicating the layout of a local 7-Eleven store. Our results show that the fused approach provides up to 93.2% item identification accuracy, better than reported self-checkout stations (86%).

CCS CONCEPTS

• **Computing methodologies** → **Object identification; Object recognition**; • **Computer systems organization** → *Sensors and actuators*; Real-time system architecture.

KEYWORDS

Auto-Checkout; Product Recognition; Item Identification; Inventory Monitoring; Retail; Supermarket; Sensor Fusion; Heterogeneous Sensing; Multi-Modal; Vision; Weight; Item Layout

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
BuildSys '19, November 13–14, 2019, New York, NY, USA
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-7005-9/19/11...\$15.00
<https://doi.org/10.1145/3360322.3360834>

Figure 1: AIM3S scenario and real implementation. Through weight sensors and cameras, the goal is to autonomously detect and identify what item(s) customers take.

ACM Reference Format:

Carlos Ruiz, Joao Falcao, Shijia Pan, Hae Young Noh, and Pei Zhang. 2019. AIM3S: Autonomous Inventory Monitoring through Multi-Modal Sensing for Cashier-Less Convenience Stores. In *The 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation (BuildSys '19)*, November 13–14, 2019, New York, NY, USA. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3360322.3360834>

1 INTRODUCTION

Traditional retail stores face significant labor costs to monitor shelf inventory regularly, often postponing this operation until off-peak hours. A delay in inventory monitoring causes high sales losses when a particular item is gone from the shelf though additional stock existed in the warehouse. An ordinary convenience store faces out-of-shelf stockout rates of 5-10%, which results in a loss of up to 4% of sales [9]. In North America alone, this accounts for approximately \$ 93 billion annual losses [14].

In order to address this issue, current approaches focus on three ways to monitor shelf stock: manual, on-item tags, and vision-based sensing. Manual approaches are the norm and mainly rely on visual inspection of the shelves to reorganize and restock when needed. Employees bandwidth typically only allows for up to a few checks a day, leading to high cost and minimal effectiveness especially in high traffic stores.

Other approaches use sensors on every item (e.g. RFID tags) to monitor remaining stock of each product [26, 29]. However, the added cost of the tags together with the labor cost of labeling every item make this approach impractical other than for high-end goods [29]. More recently, cashier-less stores using a variety of sensors are being explored. Most approaches still require human operators for proper functioning and have highly constrained stocking requirements. While some stores are already operational, such as Amazon Go [1], their automated accuracy has not been revealed and several reports point out that they heavily rely on employees watching the cameras to avoid low receipt accuracy [24, 31].

In this paper, we present an Autonomous Inventory Monitoring system, *AIM3S*, which tracks shelf-level stock in real-time as the customers pick up or return items. Using weight sensors on each shelf, our system identifies the item being taken based on the location and absolute weight change of an event, which is fused with visual object identification once the item is in the customer's hand. *AIM3S* leverages physical knowledge about the customer-shelf interaction to focus the attention of the visual classifier only on the item being interacted with. We fully implemented *AIM3S* on a 5 shelf setup with 4 cameras and 60 weight plates. To evaluate the system in a real-world setting, we used 85 items from 33 unique products and mimicked the item layout of a local 7-Eleven. Therefore, our contribution is three-fold:

- *AIM3S*, the first fully autonomous shelf inventory monitoring system without human-in-the-loop.
- An adaptive sensor aggregation algorithm to combine information from different sensing modalities, in particular shelf weight differential, visual in-hand item recognition and prior knowledge of item layout (i.e. product location).
- Implementation and evaluation in a real-world market setup with 33 products replicating the layout of a 7-Eleven store.

The rest of the paper is organized as follows. First, Section 2 discusses related works and background. Section 3 describes the design of the *AIM3S* system. In Section 4 we present the fusion algorithm that combines location, weight and vision information. Next, Section 5 provides results and analysis of the real-world evaluation in our store setup. Finally, we conclude in Section 6.

2 BACKGROUND & RELATED WORK

There exists a significant amount of work on object identification from weight as well as appearance features, though most focus on only one sensing modality. While there are solutions that can successfully identify objects solely by their weight, they fall short in tackling the inventory monitoring domain, where many sets of products weigh similarly (e.g., soft drinks, energy bars). Although vision-based object identification would be able to tackle the above cases when their packaging is different, the convenience retail

market is filled with similar-looking items that have distinct content (e.g. any yogurt vs. its fat-free version).

None of these solutions, alone, is capable of fully addressing the autonomous inventory monitoring problem due to the nature of the sensing modalities and the complexity of the environment. Sections 2.1 and 2.2 cover the state-of-the-art in weight- and vision-based object identification, respectively, and their individual limitations.

2.1 Weight-based object identification

The location of an event is a key piece of information to understand what object was picked –or placed– and is used by most prior works to complement the magnitude of the total change in weight. Different hardware approaches have been proposed to measure the weight on a surface and identify objects placed on it. A common solution consists in instrumenting a platform, such as a table or a shelf, with multiple load cells to measure the total weight of the objects on the platform [20, 21, 30]. While this makes it possible to identify objects by tracking changes in the total weight, it cannot handle cases where certain object weights are multiples of each other. In addition, location information can only be retrieved during an event and for that particular object.

To overcome these limitations, some stock-level monitoring approaches rely on capacitive weight-sensing mats [7, 8, 17–19, 28]. By measuring the change in capacitance between two parallel plates, these works analyze the presence, absence or shape of individual items to identify and count them. However, such approaches become hard to utilize when the sensing material is facing the harsh environment of an operating convenience store: liquids spilling, continuous impact, friction, permanent deformations and cost dramatically constrain their practicality in a store setting.

Finally, a small subset of prior works have attempted to complement weight sensing with vision, for applications such as grocery re-identification in a fridge [12]. Despite the small scale and highly controlled environment, these works indicate the benefits of multi-modal sensing for item identification.

2.2 Vision-based object identification

There are two major approaches for visual object identification:

Traditional **feature-based** techniques, such as Histogram of Oriented Gradients (HOG) [33], Scale Invariant Feature Transform (SIFT) [16] and Speeded-Up Robust Features (SURF) [2], rely on different sets of features designed to extract meaningful characteristics of regions in an image, which allows to detect and classify objects. In practice, these approaches involve a high degree of parameter tuning, limiting their scalability and adaptability to new scenarios –such as different lighting conditions or camera angles.

On the other hand, Deep Learning techniques try to overcome this issue with a **data-driven approach**. Mask R-CNN [11], Fast R-CNN [6], Faster R-CNN [27], Yolo [25], all leverage a big amount of data to generalize the characteristics of the objects to be identified. While in theory they should not require human input, the training procedure still involves some hyper-parameter tuning. In addition, in order to achieve high accuracy levels, these models require thousands of images of each object, under different conditions, with different backgrounds and from different angles.

In the retail domain, several prior works have tackled the item identification problem [13, 23, 32, 35] but current state-of-the-art is still limited in accuracy due to the great similarity in appearance across products (e.g., chocolate bars of the same brand with different sizes, flavors or textures). On top of that, none of these works take videos as input, meaning an added layer of scene understanding is required in order to figure out which of the detected products is the one the customer took (vs. products which remain on the shelf).

3 SYSTEM DESIGN

To the best of our knowledge, *AIM3S* is the first fully autonomous shelf inventory monitoring system without human-in-the-loop. This section provides our system design choices and assumptions.

3.1 System Overview

Figure 2 shows *AIM3S*'s system framework. It utilizes multi-modal sensing to improve item identification accuracy. In particular, we focus on three sources of information: item layout, weight and appearance. *AIM3S*'s pipeline is triggered when a change in the total weight of a shelf is detected. From that it extracts two features: the absolute weight difference and the spatial distribution of the weight. The *weight change*-based prediction computes the probability of each product class by comparing the absolute weight difference to each product's average weight. The *location*-based prediction computes the probability of each product class by combining the spatial distribution of the weight change on the shelf with prior knowledge of item layout. The *vision*-based prediction leverages human pose estimation and background subtraction to focus the visual object classifier's attention to identify the object(s) in the customer's hand. Finally, *AIM3S* fuses all three predictions by applying an adaptive weighted linear combination.

3.2 Assumptions

Handling all the intricacies and corner cases of a fully autonomous system for inventory monitoring is a very challenging task. Given the early stage of the research in this domain, we wanted to limit the scope of this paper in order to fully address the problem defined: handling pick up and put back events under *normal shopping behavior*. While we are aware that cultural, age, and other factors impact what's considered normal behavior, we observe some general trends on customers shopping in convenience stores. We make the following assumptions in order to scope this paper:

- (1) **At any given time, at most one customer is interacting with a particular shelf.** Unlike big supermarkets, convenience stores are rarely crowded and observe a much lower customer traffic. In addition, most customers shop individually and respect other customers' personal space –i.e., if someone is picking an item from the same shelf they want an item from, they wait for the other customer to get their items first. This is a safe assumption given that shelves are typically only 3 to 4 feet long.
- (2) **Customers don't place outside objects on shelves.** It is common for customers to enter the store carrying certain objects, such as a purse or a drink, but unless they are purposefully trying to fool the autonomous system, they rarely leave anything on a shelf that wasn't picked inside the store.

Figure 2: *AIM3S* system overview. Left: Location- and Weight change-based prediction pipeline (brown). Right: Vision-based pipeline (red). Bottom: Fusion algorithm (blue). Prior knowledge of item layout, weight and appearance in green.

Therefore, while users are free to return items they do not want anymore, *AIM3S* can safely assume any put back event corresponds to items from the inventory.

- (3) **Customers don't alter items' properties (weight or appearance) before putting something back.** For instance, our system assumes that customers won't pick a bag of chips, eat half of them and put them back on a shelf.
- (4) **Customers pick one item a time.** When customers want multiple items from the same shelf, it is uncommon to pick different items with each hand *at the same time* –i.e., even if they use both hands, they usually pick a product with one first, then the other, so *AIM3S* would correctly flag them as two separate events. Note that this assumption could be relaxed by considering all combinations of up to N items being picked up, though N should be kept small to limit computational complexity.

3.3 Hardware design decisions

There are many design choices involved in the instrumentation of smart retail stores for inventory monitoring. In this section, we discuss some insights we gained by implementing *AIM3S* and working with actual retailers, as well as the impact and tradeoffs of different hardware approaches.

Weight sensing. An interesting tradeoff to consider when instrumenting retail store fixtures with weight load cells is the size of each weight-sensing plate, that is, the size of each independent platform suspended over one or more weight sensors. On the one

hand, longer plates –e.g., one per shelf– means lower hardware cost and processing, but also lower signal resolution (the load cells need to support a larger maximum capacity) as well as lower spatial resolution (more items per plate, which increases the chances of having multiple items with very similar weights). In Section 5 we explore the impact of different plate sizes on *AIM3S*’s accuracy.

Furthermore, weight plate size and design can have a big repercussion on item layout flexibility, an often desired demand by retailers. Retailers’ profit margins are very low and maximizing item density is of utter importance (see Figure 7 for an example of a typical fixture layout in a convenience store). We therefore adopted a flexible hardware design that can be easily mass-produced to bring costs down, as shown in Figure 1. Our single-size design consists of narrow weight plates (4" width, divisible by the standard 48" fixture length) laid contiguous to each other. This design has the added advantage that such small weight plates won’t have as much weight on top, allowing for lower maximum capacity load cells thus higher weight resolution (sub-gram) without requiring expensive ADCs (Analog-to-Digital Converters). Moreover, Section 3.4 details how *AIM3S* handles cases where items span across multiple weight plates (the weight difference in each individual plate does not correspond to the total weight of the item, hence the product prediction has to cluster neighboring weight cells into a single event for a correct item identification).

Vision sensing. There are also many design considerations related to installing cameras in retail stores, from camera specifications, to camera placement and even number of cameras to deploy. As vision processing improves, camera specification constraints can be relaxed. From our initial experiments, camera resolution doesn’t play a huge role (in fact most deep learning networks downsize the input image to about 300-720 pixels wide for training and computation efficiency purposes). As for frame rate, we have empirically observed 25-30fps to be enough to get at least one good frame of the item being picked. In addition, optimal lighting might help get sharper and more consistent views of the products, but that is out of the scope of this paper.

Camera quantity and placement pose trade-offs worth exploring more in depth in future work. Overall, the intuition is that by having multiple cameras spread across different viewing angles, the system can minimize the likelihood of visual occlusions. While this is true, the added hardware, setup, power and computational cost can dramatically impact the benefits of autonomous inventory monitoring. For instance, the first Amazon Go [1] store in Seattle features hundreds of cameras –hanging from the ceiling, on top of each fixture and even below each shelf– and still relies on a human-in-the-loop approach to resolve uncertainties [24, 31]. From our initial experiments we empirically noticed weight sensors to be a much more robust –and cheaper– predictor of what item was picked up or put back on a shelf. Therefore, we do not consider shelf-mounted cameras in this paper. Section 5.2 however analyzes the impact of any combination among four different camera placements –top-down, sides and in front of each fixture– on *AIM3S*’s accuracy.

3.4 Customer-shelf interaction detection

The first step in *AIM3S*’s pipeline is to detect when an event took place (i.e., a customer picked up or put back an item on a shelf).

Figure 3: Aggregated weight sensed by each shelf during the first minute of one of our experiments (raw in light blue, filtered in dark blue), as well as the events detected by *AIM3S* (orange) and the annotated ground-truth (purple).

In our proposed system architecture, displayed in Figure 2, the processing of every change in the inventory starts with a weight change trigger. After carefully analyzing some initial experiment data, we came to the conclusion that, even during normal shopping behavior –i.e., customers not trying to fool the system–, visual occlusions from hands or the body are highly likely (especially for smaller items), which makes vision much less reliable than weight for triggering events. Unless someone purposefully drops an object of similar weight as they pick an item from a shelf –which would break Assumption 2–, the weight difference on the load sensors is generally enough to detect an event.

There are numerous prior works to detect events based on weight change on a load sensed surface [21, 30]. In essence these approaches compute the mean and variance of the weight values over a sliding window, and classify the state as either *stable* –no interaction– or *active* –an interaction is taking place– by comparing the moving variance to a threshold. Once the state is back at *stable*, the mean weight before and after the *active* state is extracted and reported as the weight difference of the event, where the sign indicates whether it was a pick up or a put back.

Furthermore, inevitably –even with physical separation between the weight-sensing plates– there might be cases where items lay on more than one plate. In those scenarios, looking at the individual plate scale would yield erroneous weight difference values. Instead, we aggregate all weight plates in each shelf and detect events at the shelf level. This also makes the event detection more robust to light items laying on more than one plate –which might go undetected at each individual plate, but would still provide a big enough change on the aggregate moving variance and mean. See Figure 3, which shows the shelf aggregated weight data along with the weight moving variance, mean and the events detected.

Mathematically, let $w_{s,p}^n$ define the weight on the p^{th} weight plate on shelf s at discrete time n . We compute the shelf’s aggregated weight as:

$$w_s^n = \sum_p w_{s,p}^n \quad (1)$$

Then, the shelf's aggregated moving mean and variance are, respectively, μ_s^n and v_s^n :

$$\mu_s^n = \frac{1}{2N_w + 1} \sum_{t=n-N_w}^{n+N_w} w_s^t \quad (2)$$

$$v_s^n = \frac{1}{2N_w + 1} \sum_{t=n-N_w}^{n+N_w} |w_s^t - \mu_s^n|^2 \quad (3)$$

where N_w is the sliding window half-length in samples, which corresponds to 0.5s in our implementation ($2N_w + 1 = 61$).

An event is detected according to Equations 4:

$$\text{Event begins on shelf s: } v_s^t > \varepsilon_v, \quad \forall t \in [n_b, n_b + N_h] \quad (4a)$$

$$\text{Event ends on shelf s: } v_s^t \leq \varepsilon_v, \quad \forall t \in (n_e - N_l, n_e] \quad (4b)$$

$$\text{Temporal consistency: } n_e > n_b \quad (4c)$$

where N_h and N_l correspond to the minimum length the weight variance has to exceed or fall short of the threshold ε_v in order to detect the beginning and end of an event. Based on some initial experiments we empirically set the values to $N_h = N_l = 30$ (0.5s) and $\varepsilon_v = 0.01 \text{ kg}^2$. Once an event has been detected, the Weight Change Event Detection module determines the event weight difference $\Delta\mu$ and the location –set of weight plates $\{p^s\}$ – according to Eq. 5:

$$\text{Event weight difference: } \Delta\mu = \mu_s^{n_e} - \mu_s^{n_b} \quad (5a)$$

$$\text{Event location: } L = \{p^s : |\mu_{s,p}^{n_e} - \mu_{s,p}^{n_b}| \geq \varepsilon_\mu\} \quad (5b)$$

where ε_μ indicates the minimum weight contribution of a single plate in order to be included in the event, which we set to $\varepsilon_\mu = 5 \text{ g}$.

3.5 Vision event extraction

Understanding customer–item interactions and identifying the products picked from or returned to a shelf from video streams is very computationally expensive (especially for higher camera densities). For this reason, AIM3S only saves a small buffer of recent history and uses the Weight Change Event Detection trigger from Section 3.4 to start analyzing the images. For *put back* events we directly analyze the buffer as soon as an event is detected, whereas for *pick up* events we delay the vision analysis until the oldest frame in the buffer coincides with the event trigger, n_e from Equation 4b.

The Vision Event Extraction pipeline is divided in two sequential tasks: Vision Event Preprocessing and Product Detections Spatial Selection. The former gathers different sources of visual evidence and is composed of the Human Pose Estimation, Background Subtraction and Product Detection & Classification modules. The latter then aggregates all the information and determines which object detections to keep or reject based on the customer's hand location (See Figure 4). As the output of the Vision Event Extraction pipeline, those detections together with their associated product probabilities, are fed into the Vision-based Item Identification module (Sec. 4.3) which tries to determine what product was picked.

Vision Event Preprocessing. Ideally, this step should only be comprised of Product Detection & Classification. However, visual object classifiers, such as [11, 27], provide a set of (location, object class) for anything found on an image (i.e., they would also detect all products on the shelves). In order to focus just on the item that was picked, the Vision Event Preprocessing takes additional

(a) Human pose estimation lets us find the customer's hand, where the visual classifier should look for items.

(b) We also apply background subtraction to the original image to avoid detecting items around the hand that remained on the shelf.

Figure 4: Displayed in green: R_h^c , which determines the boundary of the object classification search.

steps, as shown in Figure 2. On the one hand, it performs Human Pose Estimation, a popular research topic in the Computer Vision literature which tries to localize the joints of each person. There are many works in this domain such as [4, 5, 22, 34] which, through different approaches, are all quite mature and robust to varying lighting conditions, clothing and even substantial occlusion. On the other hand, leveraging the fact that cameras are stationary, Background Subtraction techniques such as [10] can be used to "hide" all the products that remain on the shelf and therefore focus the attention of the visual classifier only on the moving *foreground*, where it can find the item being taken or returned.

Product Detections Spatial Selection. From the skeleton of the customers, the location of their wrists represents a simple-yet-effective attention mechanism: by ignoring any detections with centroids further away from the hand than a given threshold, AIM3S eliminates most detections of the object classifier that do not correspond to the item the customer is picking or putting back. We call this threshold R_h^c , for each camera c , and pick its value empirically based on the camera–shelf distance. This spatial selection can be very useful to complement Background Subtraction when there is more than one customer moving in the scene or when the customer interacting with the shelf has products on the other hand which they had previously picked.

Figure 5: Weight distribution of each product used in our experiments.

3.6 Inventory prior knowledge

AIM3S relies on three sources of information to produce an accurate estimate of what item was returned or taken from a shelf by a customer. In order to do so, the system needs to be informed about certain properties of each product. These models can be categorized based on the source of information they provide: item layout (product location), weight and appearance model.

Item layout model. Item layout is a mapping between each product and their –initial– location in the store. The granularity or resolution of this layout could dramatically vary due to different factors, such as store size, complexity of the layout or even time and cost associated with manual annotation. Let $\mathcal{I}$ represent the set of products in the inventory and $i \in \mathcal{I}$ be any particular product (e.g., Scotch Brite sponge, Fabric Febreze, etc.). In a generic way, item layout can be defined as a function $l(\cdot) \rightarrow \{i \in \mathcal{I}\}$ that returns the set of products expected to be found at any query location. The *resolution* of the item layout can then be defined as the smallest change in location which yields a different value of l . Given the narrow width of our weight plates, the highest resolution considered in this paper is the plate level: we constrain the query location to a given shelf s and plate p such that $l(\cdot)$ can be rewritten as:

$$l_{s,p} = \{i \in \mathcal{I} \mid \text{product } i \text{ is stocked at plate } p \text{ on shelf } s\} \quad (6)$$

This way, we can simulate lower spatial resolutions by recording the item layout at virtual plates p' that aggregate multiple real plates, e.g., $a_{s,p'} = a_{s,p_1} \cup a_{s,p_2} \cup a_{s,p_3}$ (we call p' a bin of width 3). We evaluate the impact of three different levels of item layout granularity –plate, half-shelf and shelf– in Section 5.2. Throughout the paper, we will refer to the item layout model as:

$$\mathcal{L} = \{l_{s,p}\}, \quad \forall s, \forall p \quad (7)$$

Item weight model. In order to predict which product was picked, *AIM3S* first needs to have some knowledge about the weight distribution of each product in the inventory, $W(i)$. This one-time calibration step consists in weighing every item and then parametrizing the distribution, which can be approximated by a Gaussian distribution (characterized by its mean μ and standard deviation σ) as displayed in Figure 5. The *item weight model* $\mathcal{W}$ is therefore just a list of μ and σ pairs:

$$\mathcal{W} = \{(\mu_i, \sigma_i) : W(i) \sim \mathcal{N}(\mu_i, \sigma_i^2)\}, \quad \forall i \in \mathcal{I} \quad (8)$$

Note that in general, most products have a fairly consistent weight distribution, thus once estimated, new items would not need to be weighed on restocking.

Item appearance model. There are many ways to encode a visual representation of each item, as discussed in Section 2.2. Since the focus of this paper is on how to combine the different sources of information to produce an accurate product prediction and not in how to improve each source in particular, we follow state-of-the-art approaches for modelling the visual appearance. That way, *AIM3S* is not constrained to a particular object detector and can directly benefit from any improvements in the state-of-the-art. Currently, the best visual identification results are obtained through Convolutional Neural Network (CNN) based models, such as the ones mentioned in Section 2.2 [6, 11, 25, 27]. Training these deep learning models to recognize different object classes –also known as instance segmentation– requires a large data collection process. We provide more details of our training implementation in Section 5.1 and Figure 6. In short, we collected 20 second videos from multiple angles and distances of a single item of each product placed in a turntable. Then, we cropped the product using background subtraction and, through common data augmentation techniques, generated 30k multi-item images (like the ones on Figure 6d) for the network to learn a model of the visual appearance of each product.

4 COMBINING LOCATION, WEIGHT AND VISION

The previous section described the item layout, weight and appearance models that *AIM3S* relies on. Here, we detail how each sensing modality leverages this inventory prior knowledge to identify the item being picked up or put back (Sections 4.1-4.3), and how these sources of information are then all combined to emit the final product prediction (Section 4.4).

4.1 Location-based Item Identification

From the *item layout model* ($\mathcal{L}$, Eq. 6-7) and the event location information ($L = \{p^s\}$, Eq. 5b), the Location-based Item Identification module estimates the likelihood of the item in the event belonging to each product class, $i \in \mathcal{I}$. A simple approach would predict $P_L(i \mid L, \mathcal{L})$ according to:

$$\text{All items arranged at location } L: \quad \mathcal{L}_L = \bigcup_{p^s \in L} l_{s,p} \quad (9a)$$

$$P_L(i \mid L, \mathcal{L}) = \begin{cases} 1/|\mathcal{L}_L| & i \in \mathcal{L}_L \\ 0 & \text{otherwise} \end{cases} \quad (9b)$$

where $|\mathcal{L}_L|$ indicates the number of different products stocked at the weight plates $L = \{p^s\}$. Note that since a location L might be composed of multiple plates, we first need to take the union (Eq. 9a).

Figure 6: An example of the data generation for training. We collected videos of the items from three different angles shown in (a). (b) illustrates the view from one of camera angles. From (b) we removed the background (c) and overlaid the cropped objects onto images from the COCO Dataset [15] (d). Finally, a Faster R-CNN [27] model was trained with 30k images like (d).

Then, any item in the resulting set would have equal probability of being picked, and any item outside of the event’s location would be ignored. However, such approach would be disregarding some useful information: some plates may have observed a much larger portion of the total weight change than others, and it is therefore more likely that the item comes from those particular plates. For this reason, *AIM3S* uses a weight change-guided location-based item identification:

$$P_L(i | \Delta\mu, \mathcal{L}) = \frac{1}{\|\mathcal{L}\|} \sum_{p: i \in l_{s,p}} |\Delta\mu_{s,p}| \quad (10a)$$

$$\text{Normalization term: } \|\mathcal{L}\| = \sum_i \sum_{p: i \in l_{s,p}} |\Delta\mu_{s,p}| \quad (10b)$$

4.2 Weight change-based Item Identification

Product prediction based on weight change is fairly straight-forward. The main idea is to estimate how close the event’s weight change $\Delta\mu$ is to the distribution of each product, given by the *item weight model*. It is important to also account for the noise affecting the weight sensor readings, which is generally approximated by a normal distribution with zero mean and some standard deviation estimated empirically ($\sigma_w = 1\text{g}$ in our experiments). Then, the probability of the item belonging to each product class is determined by the overlapping coefficient –area under the curves– between the weight distributions of each product and the sensor readings:

$$P_W(i | \Delta\mu, \sigma_w, \mathcal{W}) = \int_{-\infty}^{\infty} \min \{f(w | \mu_i, \sigma_i), f(w | |\Delta\mu|, \sigma_w)\} dw \quad (11)$$

where $f(w | \mu, \sigma^2)$ is the probability density function (pdf) of a normal distribution.

4.3 Vision-based Item Identification

While the Product Detections Spatial Selection module (Sec. 3.5) already determines the set of detected objects with high probability of being on the customer’s hands, these predictions need to be combined together –for all frames in the buffer– to output a single probability value for each product class. Unlike weight and location, which are very hard to *occlude*, visual classifiers often suffer from

temporary occlusions –especially for smaller items or when customers carry multiple items in their hand. As a consequence, simply concatenating (i.e. multiplying) the *logits* (classification score) of all objects would lead to undesired results, since an item not detected in a frame would end with a probability of 0 regardless of how confident all other frames were. We instead propose using a *noisy OR* model, which in essence computes the probability $P_V(i)$ that each product was seen by taking the complement of the probability that the product was never seen. Mathematically, let $\mathcal{V}$ represent the set of valid detections for the current event –output by the Product Detections Spatial Selection module– and $v_i \in [0, 1]$ the classification score for each product class $i \in \mathcal{I}$, then:

$$P_V(i | \mathcal{V}) = 1 - P(i \text{ not seen in } \mathcal{V}) = 1 - \prod_{v \in \mathcal{V}} 1 - v_i \quad (12)$$

This approach is also easy to extend to multi-camera deployments: given the detections $\mathcal{V}^c$ of each camera $c \in \mathcal{C}$, the overall probability $P_V(i | \mathcal{V}^c, \dots, \mathcal{V}^c) = P_V(i | \bigcup_{c \in \mathcal{C}} \mathcal{V}^c)$

4.4 Item identification combining all sensing modalities

AIM3S’s last stage of the pipeline fuses all sources of information to emit a final product prediction, and the one with the highest probability score will be selected. Following Bayesian inference, this fusion would be mathematically described as:

$$P(i | \Delta\mu, \mathcal{V}^C) = \frac{P(\Delta\mu, \bigcup_{c \in \mathcal{C}} \mathcal{V}^c | i) \cdot P(i)}{\sum_{i \in \mathcal{I}} P(\Delta\mu, \bigcup_{c \in \mathcal{C}} \mathcal{V}^c | i) \cdot P(i)} \quad (13a)$$

$$\hat{i}_{\text{MLE}}(\Delta\mu, \mathcal{V}^C) = \arg \max_i P(\Delta\mu, \mathcal{V}^C | i) \quad (13b)$$

The main challenge in this sensor fusion arises from the difficulty of estimating the joint conditional probability $P(\Delta\mu, \mathcal{V}^C | i)$, since the visual features and the weight change may not be conditionally independent on i . In this paper, we approximate this likelihood as a weighted linear combination of each individual sensor modality –weight and vision– prediction. In general, information from weight modality (i.e., Location- and Weight Change-based Item Identification) is a more robust product predictor –partially because it is less affected by occlusions–, thus we assign it a higher relevance when alpha-blending ($\alpha = 0.6$ gave us the best results). It is also worth noting that, as discussed in Section 4.3, cameras

can be occluded, lighting conditions may change, etc. therefore an object not being seen should not result in a final probability of 0. For these reasons, *AIM3S* sums both modalities predictions instead of multiplying to fuse them. Furthermore, while our vision pipeline tries to ensure that only the item being picked or put back is seen by the Product Detection & Classification module, it may happen that several object detections get selected by the Product Detections Spatial Selection (e.g., when the customer has other items in their hand). In those cases, it doesn't make sense that probabilities add up to 1 (e.g. 25% confident that it saw 4 objects), but rather that each product was seen with probability 1. Consequently, *AIM3S* does not normalize the vision product predictions before alpha blending. Therefore Eq. 13b becomes:

$$p_{\text{weight}}^i = \frac{P_L(i | \Delta\mu, \mathcal{L}) \cdot P_W(i | \Delta\mu, \sigma_w, \mathcal{W})}{\sum_i P_L(i | \Delta\mu, \mathcal{L}) \cdot P_W(i | \Delta\mu, \sigma_w, \mathcal{W})} \quad (14a)$$

$$p_{\text{vision}}^i = P_V(i | \mathcal{V}) \quad (14b)$$

$$p_{\text{fusion}} = \alpha p_{\text{weight}} + (1 - \alpha) p_{\text{vision}} \quad (14c)$$

$$\hat{i}_{\text{MLE}} = \arg \max_i p_{\text{fusion}}^i \quad (14d)$$

where $\hat{i}_{\text{MLE}}$ is the product predicted for the event.

5 EVALUATION

This Section presents our implementation of *AIM3S*, the experimentation setup, the metrics used to evaluate the performance of different approaches, and the actual experiment results.

5.1 System Implementation

Our system utilizes a large array of weight and vision sensors. Below we describe the details about our hardware implementation and training procedure, followed by how the experiments were carried out and the evaluation metrics.

In order to understand the effect of having weight sensors at different spatial resolutions, we designed narrow (4") weight plates which fit nicely on standard 48" shelves used by many retailers. This allows us to simulate plates of different widths (which we call bins) and evaluate the bin size parameter.

For vision sensors, we utilized 720p IP cameras and wrote scripts to record all video and weight data to disk. Although we process the results offline, our algorithms run in real-time on a cluster of 3 Nvidia GeForce GTX 1080 GPUs for our vision pipeline –the computation required for the weight change- and location-based predictions is negligible.

As Figure 7 shows, in order to fully evaluate a real-world setup, we went to a local 7-Eleven convenience store, purchased all items on a fixture (Fig. 7a) and arranged them in the same manner on our shelves (Fig. 7b). In the process, we initialized our system by:

- (1) Weighed every item and fitted a Normal distribution to generate the *item weight model* (displayed on Fig. 5).
- (2) Marked what items lie on what plates (e.g., 409 is stocked on plates 1 and 2 on shelf 1) to generate the *item layout model*.
- (3) We replicated [35]'s approach for training the visual product recognition pipeline. While rotating 360° on a white turntable (facilitates background removal), three cameras at different angles and distances simultaneously recorded an image of each product class, placed on its back as well as front

Figure 7: In order to follow a realistic item layout and understand potential failure cases in real deployments, we replicated the layout of a 7-Eleven store.

side. From those videos, the items could be segmented out and overlaid on top of random images in random positions, sizes, rotations, etc. (see Figure 6). We pre-trained a Faster R-CNN [27] model (using ResNet-50 and FPN) on [35]'s dataset, trimmed the last (classification) layer, changed the number of output classes to 33+1 (background) and trained on our dataset of 30k images generated from such cropped products.

5.1.1 Experiment Settings. We designed our experiments to try to simulate a real shopping experience where customers have a notion of what they want to purchase but may not know where items are located. Our 8 participants had never seen this item layout and, at the beginning of each trial (5 repetitions per person) they were given a randomly generated *shopping list* with 3 to 6 items (repetitions allowed). To incentivize the presence of some *put back* events, with 20% probability subjects were asked to return one of their items (without specifying where they should put it). Participants were instructed to leave the experiment area once they thought they had collected all items in their list. We set up the 4 cameras to cover the fixture (set of shelves) from all the main angles: top-down, in front and both sides, as shown in Figure 1.

5.1.2 Metrics. The main area this paper tackles is item identification. We define the average item identification accuracy, *Avg. ID accuracy* for short, as:

$$\text{Avg. ID accuracy} = \frac{\# \text{ correct items predicted}}{\# \text{ events}} (\%) \quad (15)$$

Its complement, the average identification error, can then be easily defined as:

$$\text{Avg. ID error} = \frac{\# \text{ incorrect predictions}}{\# \text{ events}} (\%) \quad (16)$$

It is worth noting that the whole framework relies on a *successful event detection*, that is, missing an event or detecting two consecutive events as a single event would have a direct negative impact in

Figure 8: Item identification accuracy for different bin widths. Performance –slowly– degrades as the bin width (i.e., how many weight plates make a *virtual* plate) increases.

the identification accuracy. However, since event detection accuracy tends to be high and it is out of scope for this work, we focus on the evaluation of item identification (measured by the Avg. ID accuracy). In the next subsection, we analyze our experiment results and the dependency of AIM3S’s Avg. ID accuracy on different system parameters.

5.2 Real-world experiments

In order to fully evaluate the system in a real-world setting, we conducted the “7-Eleven fixture” experiments described above, and present the results here. We want to understand how the resolution or amount of information on a given sensing modality affects the identification performance.

For weight sensing, one of the system parameters that impacts the spatial resolution is the bin width. As a reminder from Figure 1, we refer to a bin as a virtual plate that aggregates (sums) multiple real plates, where *bin width* indicates how many plates make up a bin. In Figure 8, we can see how location-based (P_L , in light brown) item identification suffers the most as bins get wider –and therefore more items lie inside– dropping from 76.1% at bin width 1 to 11.9% when we only have one plate per shelf (bin width 12). On the other hand, weight change-based (P_W , in orange) identification is independent on the bin width, since it only takes as input the total weight change on the whole shelf. Of course, some items are very close in weight to others (see Fig. 5) so only relying on the absolute weight change yields an accuracy of 68.2%. By combining these two weight sensing-based sources of information ($P_{weight} = P_L \cdot P_W$, in brown), the accuracy rises to 91.5% and the impact of larger bin widths is reduced, only dropping to 79.0% for bin width 12. Finally, as we investigate deeper below, fusing the weight with visual information (P_{fusion} , in blue) can bump the accuracy all the way to 93.2% for bin width 3 (slightly higher than bin width 1, and corresponds to the width of most of the items in the inventory).

However, it isn’t only bin width that affects weight-based predictions. The resolution of the item layout model, $\mathcal{L}$, can also significantly impact performance. As shown in Figure 9, we consider three different resolutions: plate-level ($\mathcal{L}$ contains which items could be located at each individual plate), half-shelf-level ($\mathcal{L}$ only records

Figure 9: Item identification accuracy for different resolutions of item layout (i.e., how knowing what products are in each plate, in every half-shelf or only in every shelf affects performance). Half-shelf offers a great compromise between accuracy and effort to fill out the item layout model.

which items lie on the left 6 or right 6 plates of each shelf) and shelf-level ($\mathcal{L}$ only logs the product–shelf mappings). What we observe is that a half-shelf layout model is almost as good as the plate-level layout (90.9% vs 92.0% for AIM3S’s fusion or 90.3% vs 91.5% for weight-based), while requiring a lot less effort to generate and maintain. On the other hand, when the item layout resolution is at the shelf level, the accuracy drops to 80.1% and 79.0% respectively, a likely unacceptable level for real deployments.

Figure 10 explores the benefits of using different amounts and combinations of cameras. For short, we refer to them as **Left**, **Right**, **Top** and **Back** (their exact location with respect to the shelves can be seen in Figure 1). For this comparison, we use two different baselines for vision-based item identification:

- Simply taking the $\arg \max$ on P_V (dark red), which often contains multiple items with probability 1 –and would report a lower accuracy.
- Thresholding on P_V , e.g., considering an event was correctly identified as long as the ground-truth product had a 0.9 or higher visual score –which captures better whether vision would help the fusion or not.

From the results it is interesting to observe that just using Left and Right cameras already leads to the highest fusion score of 93.2%. It is also worth noting that the low vision-based accuracies reported here are a combination of multiple factors: the domain adaptation gap between the lighting and environmental conditions where the item appearance model was collected and the experiments were conducted, the imperfections on the background subtraction to crop the items and the difficulty of focusing the attention of the visual classifier on the item in the customer’s hands, to name a few. But even with this big room for improvement, AIM3S’s fusion approach still can extract useful visual information and achieve up to 2% higher accuracy than without cameras bringing the accuracy up to 93%, which reduces error from 9 to 7% (Figure 11), a 28% reduction in error. This is also an improvement over the reported accuracy of current self-checkout systems, which can give results of only 86% [3] and still is widely used in retail stores.

Figure 10: Item identification accuracy for different camera combinations (Left, Right, Top, Back [see Fig 1]). With only Left+Right AIM3S already achieves highest accuracy (93.2%).

Figure 11: Item identification error for different bin widths. Note the reduction in error when vision assists weight (AIM3S, blue, is consistently better than no fusion, brown).

6 CONCLUSION

In this paper, we presented AIM3S (Autonomous Inventory Monitoring through Multi-Modal Sensing) for cashier-less convenience stores. Utilizing weight difference, visual item recognition, and item layout (location) information, AIM3S monitors products placed on or picked up from shelves without human-in-the-loop with up to 93.2% item identification accuracy, way above the 86% accuracy reported for self-checkout stations [3]. To the best of our knowledge, this is the first fully autonomous system that fuses multiple sensing modalities to identify what item(s) customers take, without relying on human-in-the-loop approaches such as manual monitoring or Amazon Go systems.

ACKNOWLEDGMENTS

The authors would like to thank AiFi Inc., especially Steve Gu, William Hendry and Aiman Sokhal, for their help and support.

REFERENCES

- [1] Amazon. 2019. *Amazon Go*. <https://www.amazon.com/b?node=16008589011> Accessed: 2019-04-10.
- [2] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. 2006. Surf: Speeded up robust features. In *European conference on computer vision*. Springer, 404–417.
- [3] Adrian Beck. 2011. Self-scan checkouts and retail loss: Understanding the risk and minimising the threat. *Security Journal* 24, 3 (2011), 199–215.
- [4] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. 2018. OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields. In *arXiv preprint arXiv:1812.08008*.
- [5] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. 2017. Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. In *CVPR*.
- [6] Ross Girshick. 2015. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*. 1440–1448.
- [7] Peter Godlewski. 2010. Automated inventory system and method. US Patent 7,650,298.
- [8] Craig B Green. 2009. Weight sensing apparatus and methods for inventory control. US Patent 7,518,068.
- [9] Thomas W Gruen, Daniel S Corsten, and Sundar Bharadwaj. 2002. *Retail out-of-stocks: A worldwide examination of extent, causes and consumer responses*. Grocery Manufacturers of America Washington, DC.
- [10] Lili Guo, Dan Xu, and Zhenping Qiang. 2016. Background subtraction using local svd binary pattern. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 86–94.
- [11] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*. 2961–2969.
- [12] Rena Kamoda, Mayumi Ueda, Takuya Funatomi, Masaaki Iiyama, and Michihiko Minoh. 2012. Grocery re-identification using load balance feature on the shelf for monitoring grocery inventory. In *Computers workshop (CwC)*. Citeseer, 13.
- [13] Marcus Klasson, Cheng Zhang, and Hedvig Kjellström. 2019. A Hierarchical Grocery Store Image Dataset with Visual and Semantic Labels. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 491–500.
- [14] Tian Li and Hongtao Zhang. 2015. Information sharing in a supply chain with a make-to-stock manufacturer. *Omega* 50 (2015), 115–125.
- [15] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*. Springer, 740–755.
- [16] David G Lowe. 2004. Distinctive image features from scale-invariant keypoints. *International journal of computer vision* 60, 2 (2004), 91–110.
- [17] Christian Metzger. 2005. Indirect Object-sensing Technology to Prevent Out-of-stock at Retail-level. In *Proceedings of the Workshop on Smart Object at the International Conference on Ubiquitous Computing, Tokyo*. Citeseer.
- [18] Christian Metzger, Jan Meyer, Elgar Fleisch, and Gerhard Tröster. 2007. Weight-sensitive foam to monitor product availability on retail shelves. In *International Conference on Pervasive Computing*. Springer, 268–279.
- [19] Matthew Meyer. 2016. Out of stock sensor. US Patent 9,275,361.
- [20] Kazuya Murao, Junna Imai, Tsutomu Terada, and Masahiko Tsukamoto. 2015. Recognizing activities and identifying users based on tabletop activities with load cells. In *Proceedings of the 17th International Conference on Information Integration and Web-based Applications & Services*. ACM, 39.
- [21] Kazuya Murao, Junna Imai, Tsutomu Terada, and Masahiko Tsukamoto. 2017. Activity Recognition and User Identification based on Tabletop Activities with Load Cells. *Journal of Information Processing* 25 (2017), 59–66.
- [22] Alejandro Newell, Kaiyu Yang, and Jia Deng. 2016. Stacked hourglass networks for human pose estimation. In *European Conference on Computer Vision*. Springer, 483–499.
- [23] Siyuan Qiao, Wei Shen, Weichao Qiu, Chenxi Liu, and Alan Yuille. 2017. Scalenet: Guiding object proposal generation in supermarkets and beyond. In *Proceedings of the IEEE International Conference on Computer Vision*. 1791–1800.
- [24] Recode. 2019. *Amazon's store of the future has no cashiers, but humans are watching from behind the scenes*. <https://www.recode.net/2017/1/6/14189880/amazon-go-convenience-store-computer-vision-humans> Accessed: 2019-04-10.
- [25] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 779–788.
- [26] Yacine Reikik, Evren Sahin, and Yves Dallery. 2008. Analysis of the impact of the RFID technology on reducing product misplacement errors at retail stores. *International Journal of Production Economics* 112, 1 (2008), 264–278.
- [27] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*. 91–99.
- [28] Mahsan Rofouei, Wenyao Xu, and Majid Sarrafzadeh. 2010. Computing with uncertainty in a smart textile surface for object recognition. In *2010 IEEE Conference on Multisensor Fusion and Integration*. IEEE, 174–179.
- [29] George Roussos. 2006. Enabling RFID in retail. *Computer* 39, 3 (2006), 25–30.
- [30] Albrecht Schmidt, Martin Strohhach, Kristof Van Laerhoven, Adrian Friday, and Hans-Werner Gellersen. 2002. Context acquisition based on load sensing. In *International Conference on Ubiquitous Computing*. Springer, 333–350.
- [31] TechCrunch. 2019. *Inside Amazon's surveillance-powered, no-checkout convenience store*. <https://techcrunch.com/2018/01/21/inside-amazons-surveillance-powered-no-checkout-convenience-store/> Accessed: 2019-04-10.
- [32] Alessio Tonioni, Eugenio Serro, and Luigi Di Stefano. 2018. A deep learning pipeline for product recognition in store shelves. *arXiv preprint arXiv:1810.01733* (2018).
- [33] Xiaogang Wang, Gianfranco Doretto, Thomas Sebastian, Jens Rittscher, and Peter Tu. 2007. Shape and appearance context modeling. In *2007 IEEE 11th international conference on computer vision*. Ieee, 1–8.
- [34] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. 2016. Convolutional pose machines. In *CVPR*.
- [35] Xiu-Shen Wei, Quan Cui, Lei Yang, Peng Wang, and Lingqiao Liu. 2019. RPC: A Large-Scale Retail Product Checkout Dataset. *arXiv preprint arXiv:1901.07249* (2019).