Multi-UAV Trajectory Optimization and Deep Learning-based Imagery Analysis for a UAS-based Inventory Tracking Solution

Youngjun Choi*, Maxime Martel†, Simon Briceno‡, Dimitri Mavris§
Aerospace Systems Design Laboratory, School of Aerospace Engineering
Georgia Institute of Technology, Atlanta, GA, 30332, USA

This paper presents a multi-UAV trajectory optimization and an imagery analysis technique based on Convolutional Neural Networks (CNN) for an inventory tracking solution using a UAS platform in a large warehouse or manufacturing environment. The current inventory tracking method is a manual and time-consuming process to scan all the inventory items. Its accuracy is not consistent depending on the complexity of the scanning environment. To improve the scanning efficiency with respect to time and accuracy, this paper discusses a UAS-based inventory solution. In particular, this paper addresses two primary topics: multi-UAV trajectory optimization to scan inventory items and a multi-layer CNN architecture to identify a tag attached on the inventory item. To demonstrate the proposed multi-UAV trajectory optimization framework, numerical simulations are conducted in a representative inventory space. The proposed CNN-based imagery analysis framework is demonstrated on a flight experiment.

I. Introduction

In the last decade, the Unmanned Aerial System (UAS) has become more capable and advanced by new emerging technologies such as novel battery technologies and new sensing technologies. The use of a UAS can be beneficial because of its low-cost, highly agile platform and high-quality sensor system. Consequently, a UAS has been extended its application areas: aerial imaging, parcel delivery, crop-monitoring, and disaster monitoring. In particular, internal and external audits using a UAS platform in warehouse or manufacturing environments have gained attention because of the necessity of the novel structure of an audit evidence with respect to big data, and the improvement of the accuracy and speed of a traditional inventory audit process [2].

There are many challenges related to the internal/external audits, such as the identification of inventory items, their visual inspection, and counting them. Between the internal and external audits, this paper focuses on the internal audit process because the external audit process may be more challenging by FAA regulation. In particular, this paper deals with the inventory tracking problem to improve the current existing process in warehouse or manufacturing environments.

A conventional approach for tracking inventory items is a manual scanning method using a barcode reader. Each item in a warehouse is equipped with a tag on which a barcode is printed. This barcode serves as a unique identifier for an individual item. Employee with a barcode scanner manually scans the barcode and identify the item. The scanned item is automatically stored in the inventory database to keep track of their locations. This manual scanning method has several drawbacks. First, it easily leads to human errors because of the highly repetitive nature of the task. Second, this scanning process is time-consuming

*Research Engineer II, School of Aerospace Engineering, Aerospace Systems Design Laboratory, AIAA Member.
†Graduate Research Associate, School of Aerospace Engineering, Aerospace Systems Design Laboratory
‡Senior Research Engineer, School of Aerospace Engineering, Aerospace Systems Design Laboratory, and AIAA Senior Member.
§S.P. Langley Distinguished Regents Professor, School of Aerospace Engineering, Aerospace Systems Design Laboratory, and AIAA Fellow.

and dangerous. When an item is located on the top of shelves that cannot be easily viewed by a human, an employee is required to use specialized lift equipment, like mobile stairs or forklift devices. These types of equipment are identified as potential hazard items for the workers according to the worker safety series published by the Occupational Safety and Health Administration. Carrying this equipment in a workplace also reduces the speed of the inventory audit process.

Another conventional approach is the use of the Radio Frequency Identification (RFID) system. This system is based on the wireless technology by reading the unique RFID tag of each item. In spite of rapid and accurate technology, it is not an economically viable solution because RFID tags are more expensive compared to the traditional paper tags.

To solve the issues of the conventional methods, a mobile robot-based inventory tracking technology has been suggested by Zimmerman [24]. The mobile robot is capable of mapping/localization through manual navigation in an inventory space. Using the mapping information, the mobile robot captures shelf images, and identifies an item from the captured images through recognizing a barcode on the image. However, the limitation of the mobile robot-based inventory audit method is that the robot platform might have a complex structure because in order to obtain images on the top of shelves, the camera system on a platform must be attached at a certain height that enables it to get the images, which makes the mobile robot is very bulky.

To resolve this issue, several companies (e.g., Walmart, Amazon, PINC and Eyesee) have suggested UAS-based inventory tracking solutions. Based on the authors' knowledge, the companies have conceptually introduced their UAS-based tracking solution, but there are little literature containing technical details. This paper describes a framework for a UAS-based inventory audit solution using optical image information. In particular, this paper presents two primary topics: UAV trajectory optimization, and a deep learning-based imagery analysis. For the former topic, we discuss a multi-UAV trajectory optimization framework because the size of a warehouse or manufacturing environment is considerably large, and the typical endurance of a quadcopter is short, approximately between 10 and 30 minutes [4]. Hence, an operation concept using multi-UAV is a suitable approach to scan the inventory items in a large warehouse or manufacturing environment. For the deep learning-based imagery analysis, we describe a tag identification algorithm. For the tag identification algorithm, we introduce a multi-layer Convolution Neural Network (CNN) architecture. The main contributions of this paper are:

- A multi-UAV trajectory optimization that considers the characteristics of inventory space, and the characteristics of a UAV.
- A framework of a multi-layer convolution neural network for the identification of each inventory item, which includes CNN architecture, and flight demonstration.

In the remainder of the paper, we first introduce a multi-UAV trajectory optimization based on an endurance-constrained vehicle routing problem. The proposed algorithm is demonstrated by a numerical simulation using a 3D representative inventory model. Then, we present a framework of CNN-based imagery analysis for the image analysis. This proposed CNN-based imagery analysis framework is demonstrated by an indoor flight experiment.

II. Multi-UAV Trajectory Optimization

In order to efficiently operate multiple UAVs in a large inventory space, a trajectory optimization is a key element because it is directly related to UAV operation time and cost. In general, the goal of the trajectory optimization is to minimize an objective function while satisfying a set of constraints. According to a paper written by Choi et al., there are five types of trajectory optimization techniques: potential field methods, geometric methods, optimization-based methods, stochastic methods, and road map methods [5]. Among these methods, a graph-based optimization approach is commonly applied to coverage path-planning (CPP) problems, which generates an optimal trajectory passing over an entire coverage area. The notable approaches include a cell decomposition-based CPP method, a wavefront-based CPP method, and a vehicle routing-based CPP approach [7]. The cell decomposition-based CPP method decomposes multiple convex small areas if the area of interest is large or non-convex. Then, it solves high-level optimization problem determining a visiting sequence based on the decomposed convex cells. The trajectory inside of each cell is defined by simply sweeping a line from left to right. Another approach is the wavefront-based CPP method, which employs a wavefront function that scores each cell based on an initial position, a target position, and

obstacle positions [3]. Based on these scores, the algorithm determines a CPP trajectory that finds a route maximizing the pseudo-gradient. This approach has benefits of being simple and able to solve non-convex problems. An alternative approach is a vehicle routing-based CPP algorithm. The vehicle routing problem as a graph-based approach solves an optimization problem to determine the best route that visits all waypoints by a set of vehicles, given a potential route network. The vehicle routing-based optimization method includes a set of constraints by vehicle characteristics, or operational constraints. This approach can be extended for multi-UAV trajectory optimization problems. Alilar et al. solve a multi-UAV vehicle routing problem with various operational constraints such as number of operators, and setup time for an aerial imaging CPP problem [1].

Our primary goal is to solve a multi-UAV trajectory problem to scan all the shelves in an inventory space without any collision. For this problem, we propose the framework of a two-phase multi-UAV trajectory optimization that guarantees a collision-free trajectory, and minimizes the number of UAV platforms needed. The novelty of the proposed trajectory optimization framework is incorporating it with an actual inventory problem, and generating non-collision trajectories for a multi-UAV operation.

A. Mathematical Formulation of Endurance-Constraint Vehicle Routing Problem

The multi-UAV trajectory optimization is structured based on the formulation of a distance-constrained arc-based vehicle routing problem suggested by Kara [12]. Let $G = (\mathcal{N}, \mathcal{A})$ be a graph that describes a route network, which consists of a set of nodes $\mathcal{N} = \{0, 1, \cdots, n+1\}$ and a set of arcs $\mathcal{A}$. In the nodes $\mathcal{N}$, an initial depot node is the 0th node, an artificial depot (i.e., returning depot) is the $n+1$th node, and waypoints $\mathcal{W}$ are from the 1st node to the nth node. The arcs $\mathcal{A}$ indicate the connection between two nodes, $\mathcal{A} = \{(i,j) : i,j \in \mathcal{N},\ i \neq j\}$. For the multi-UAV inventory tracking mission, the trajectory optimization formulation satisfies the following conditions: First, all UAVs deploy from the initial depot node (0th node) and finish the mission on the artificial node ($n+1$th node). Second, the arcs of each shelf must be scanned. Third, all the UAVs must meet their endurance constraints. To formulate the multi-UAV trajectory optimization, the following decision variables are considered:

- x_{ijk} : If the vehicle k flies between the ith node and jth node, the variable x_{ijk} is defined as 1, otherwise, it is 0.
- y_{ijk} : The total distance between ith node and jth node of vehicle k.

The objective function of the endurance-constrained vehicle routing problem is minimizing the total mission time with setup time defined by

$$J = \min \sum_{k \in V} \sum_{i \in N} \sum_{j \in N} \mathbf{T}_{ijk} x_{ijk} + \sum_{k \in V} t_s x_{0hk}, \quad (h \in \mathcal{W}), \tag{1}$$

where $\mathbf{T}_{ijk}$ is time matrix corresponding to flying from the ith node to the jth node by the kth vehicle, and the t_s is setup time to deploy a UAV platform. Constraints can be written as:

$$c_1 : \sum_{k \in \mathcal{V}} \sum_{j \in N} x_{ijk} = 1, \quad (\forall i \in \mathcal{W}) \tag{2}$$

$$c_2 : \sum_{i \in \mathcal{N}} x_{0jk} = 1, \quad (\forall k \in \mathcal{V}) \tag{3}$$

$$c_3 : \sum_{i \in \mathcal{N}} x_{i(n+1)k} = 1, \quad (\forall k \in \mathcal{V}) \tag{4}$$

$$c_4 : \sum_{i \in \mathcal{N}} x_{ihk} - \sum_{j \in \mathcal{N}} x_{hjk} = 0, \quad (\forall h \in \mathcal{W}, \forall k \in \mathcal{V}) \tag{5}$$

$$c_5 : \sum_{j \in \mathcal{N}} y_{ijk} - \sum_{j \in \mathcal{N}} y_{jik} - \sum_{j \in \mathcal{N}} \mathcal{T}_{ijk} x_{ijk} = 0, \quad (\forall i \in \mathcal{N}, \forall k \in \mathcal{N}) \tag{6}$$

$$c_6 : y_{0jk} = \mathbf{T}_{0jk} x_{0jk}, \quad (\forall j \in \mathcal{N}, \forall k \in \mathcal{V}) \tag{7}$$

$$c_7 : y_{ijk} \leq (\mathbf{E} - \mathbf{T}_{j0k}) x_{ijk}, \quad (\forall i,j \in \mathcal{N}, \forall k \in \mathcal{V}) \tag{8}$$

$$c_8 : y_{i(n+1)k} \leq \mathbf{E} x_{i(n+1)k}, \quad (\forall i \in \mathcal{N}, \forall k \in \mathcal{V}) \tag{9}$$

$$c_9 : y_{ijk} \geq (\mathbf{T}_{0ik} + \mathbf{T}_{ijk}) x_{ijk}, \quad (\forall i,j \in \mathcal{N}, \forall k \in \mathcal{V}) \tag{10}$$

Constraint c_1 states that all the waypoints $\mathcal{W}$ must be visited exactly once. Constraint c_2 means that each vehicle is deployed at the depot node, and constraint c_3 indicates that each vehicle returns to the artificial node. Constraint c_4 implies that after a vehicle visits a node, it leaves the node. Constraint c_5 eliminates all the sub-tours. Constraints $c_6 \sim c_9$ are the bounding constraints that a vehicle must meet its maximum endurance $\mathbf{E}$.

B. Collision-free Trajectory Generation

The optimization result using the vehicle routing problem does not guarantee non-collision trajectories. Thus, the objective of the next step is to generate collision-free trajectories based on the trajectory results from the previous step. The main idea of the collision-free trajectory is checking the collision possibility of all the possible UAS deployment sequences $\mathcal{Q}$. Algorithm 1 summarizes the collision-free trajectory generation. In the possible sequences $\mathcal{Q}$ (line 2), the n corresponds to the number of sequence combinations. Each Q_i has the information of the ith UAV's sequence. We note that if the number of UAVs is five, then each Q_i has 5 numbers, and n is 125. The algorithm stores the deployment time t_{deploy} allowing all the UAVs to prevent any collisions during their operations (line 8 - 22). Checking for collision is accomplished by measuring the distance between time-domain trajectories with time offset t_0 of UAV_j and the time-domain trajectories of $UAVs_i$, that is, the previous deployed UAVs from 1 to $j-1$ in the sequence Q_i. In the algorithm, $\mathbf{X}_i$ indicates the vehicle state vector with time data. The final deployment time t_{deploy} is defined when a minimum distance d_{min} satisfies a distance constraint d_c. Based on the result of the deployment time, the vehicle trajectories considering the deployment time are computed. From the computed trajectories, the optimal sequence is determined by finding the minimum total mission time from all the vehicle state results $\mathcal{X}$.

Algorithm 1 Optimization of Collision-free UAS schedule

1: **Inputs:**
2: UAV sequence, $\mathcal{Q} = [Q_1, Q_2, ..., Q_n]$
3: Time step, Δt
4: **for** $(Q_i \in \mathcal{Q})$ **do**
5: $t_0 = 0$
6: $t_{deploy} = 0$
7: $\mathcal{X}_i(t) = Q_i, \quad \mathcal{X}_i(t) = [\mathbf{X}_1(t), \mathbf{X}_2(t), \mathbf{X}_3(t), \mathbf{X}_4(t), \mathbf{X}_5(t)]$
8: **for** $(j = 2:5)$ **do**
9: $UAVs_i = \bar{\mathcal{X}}(t), \quad \bar{\mathcal{X}}(t) = [\bar{\mathbf{X}}_1(t), ..., \bar{\mathbf{X}}_{j-1}(t)]$
10: $UAV_j = \mathbf{X}_j(t)$
11: $t_{diff} = 0$
12: **while** $(d_{min} < d_c \ \ || \ \ t_{diff} < t_s)$ **do**
13: $t_0 = t_0 + \Delta t$
14: **for** $\bar{\mathbf{X}}_i \in UAVs_i$ **do**
15: $dist \leftarrow append(distance(\bar{\mathbf{X}}_i(t), \mathbf{X}_j(t)))$
16: **end for**
17: $d_{min} = Min(dist)$
18: $t_{diff} = t_0 - t_{0prev}$
19: $t_{0prev} = t_0$
20: **end while**
21: $t_{deploy} \leftarrow append(t_0)$
22: **end for**
23: $\tilde{\mathcal{X}}_i = \mathcal{X}_i(t + t_{offset})$
24: **end for**
25: $\tilde{\mathcal{X}}^* \leftarrow FindOptimalSequence(\tilde{\mathcal{X}})$
26: **return** $\tilde{\mathcal{X}}^*$

C. Numerical Simulation Result

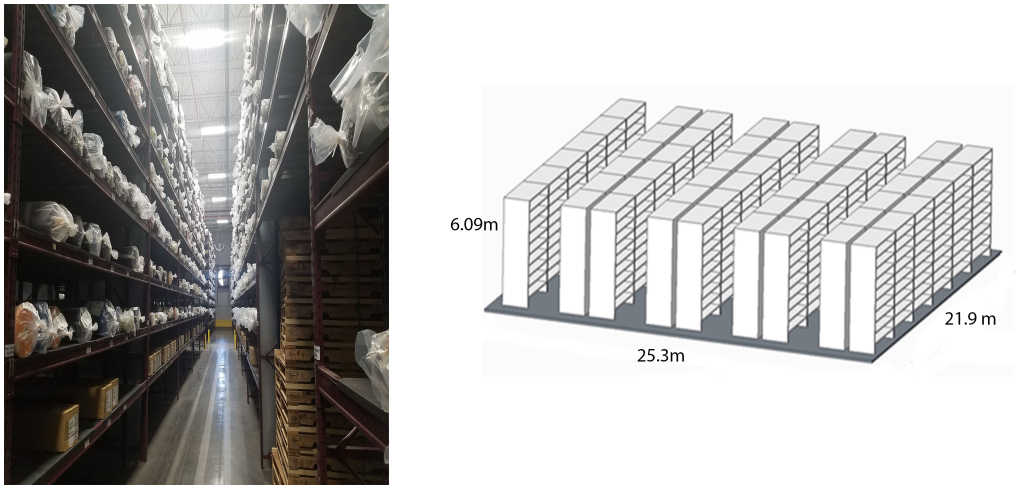
In this section, the proposed multi-UAV trajectory optimization algorithm is applied to a representative inventory environment. To create a realistic inventory environment, we generate a 3D inventory space based

on an actual inventory drawing. Figure 1(a) is the image from an actual inventory space, and Figure 1(b) is the 3D inventory model to validate our proposed multi-UAV trajectory optimization. For the numerical simulation, the UAV platform is assumed to be the DJI Phantom 4, and its endurance is assumed to be 25 minutes. The scanning speed is assumed to be $0.3m/s$.

To collect the images of all the inventory items, we also assume that UAVs should fly all the shelves, and UAVs are deployed from one depot position. Based on these assumptions, the initial flyable trajectory network is determined and is employed to solve the multi-UAV endurance constraint-based vehicle routing problem introduced in Section A. Table 1 summarizes the results of the multi-UAV trajectory optimization. Note that we use the relative tolerance from the optimal solution, 0.05. The definition of the relative tolerance is as follows,

$$relative\ tolerance = \left| \frac{objective\ function\ value - lower\ bound\ value}{objective\ function\ value} \right| \tag{11}$$

The result shows that to scan all the shelves given the inventory area, the required number of UAVs is five. The result also presents that the flight times of all the UAVs meet the endurance constraint, which is 25 minutes. Figure 2 visualizes the trajectories of the UAVs. Note that in the figure, each color indicates the trajectory of each UAV. This trajectory is able to pass all the shelves we want to collect the images, but this trajectory does not provide any information about when UAV is deployed and whether the generated trajectory is flyable without any collisions between UAVs.



(a) Example of an inventory area

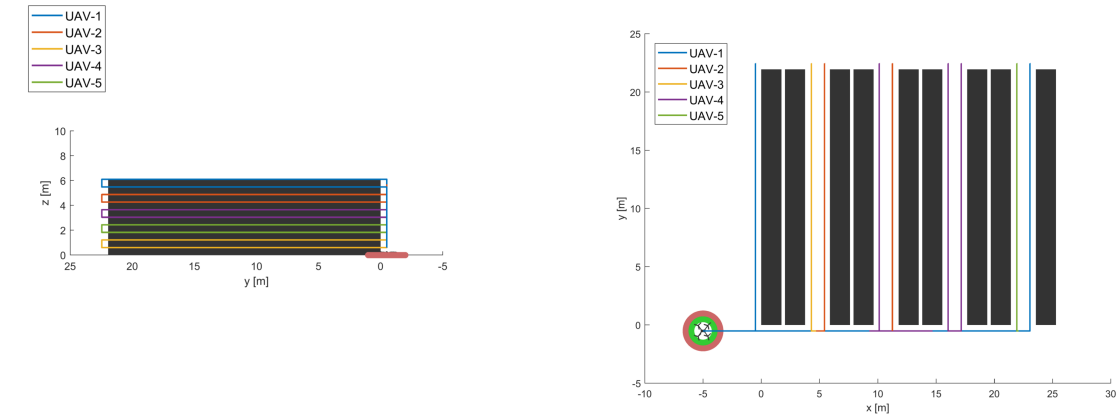
(b) 3D model of the inventory area

Figure 1. Inventory area and 3D model of the inventory area

Table 1. Mission flight time (Platform: DJI Phantom 4)

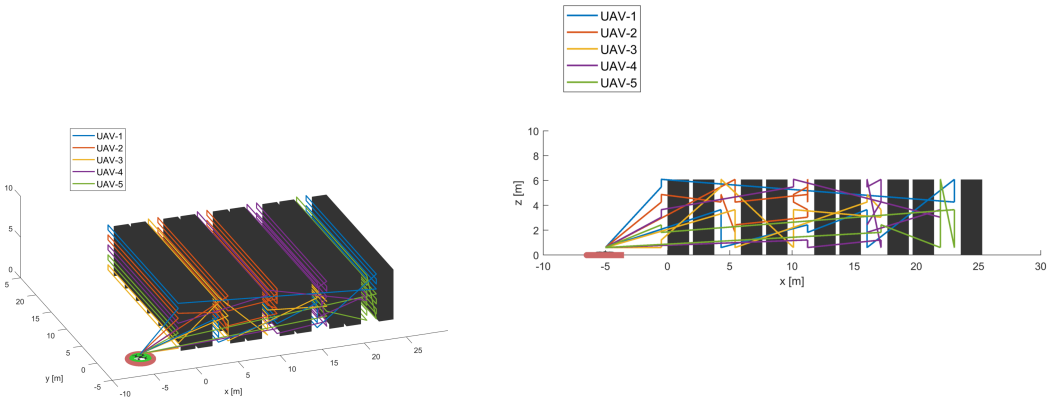
UAV id	Mission flight time
UAV-1	24min 12sec
UAV-2	23min 34sec
UAV-3	24min 4sec
UAV-4	24min 12sec
UAV-5	24min 12sec

To compute deployment time without any collisions, Algorithm 1 is employed. In the numerical simulation, the separation distance constraint is assumed to be $3m$. Figure 3 illustrates the result of the collision-free schedule. Figure 3(a) depicts the UAVs' operational schedule, and Figure 3(b) is the corresponding separation distance between UAVs. The result shows that the total mission time is around 79 *minutes* 13 *seconds*. The result of the separation distance response shows that all the UAVs satisfy the separation distance constraint. To conclude, the proposed two-layer multi-UAV trajectory optimization algorithm can minimize



(a) Front view

(b) Top view

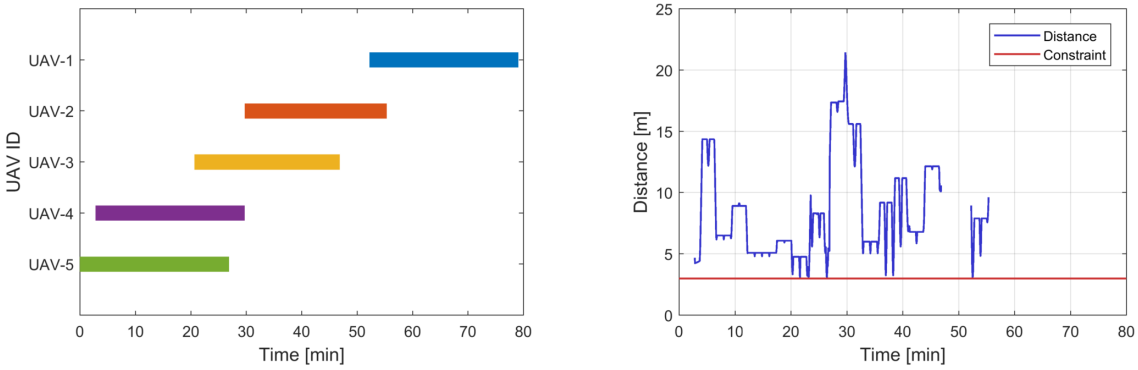


(c) 3D view

(d) Side view

Figure 2. Results of multi-UAV trajectory optimization

the number of UAVs, total mission time, and can satisfy the UAV performance constraint as well as the separation assurance requirement during the inventory tracking mission.



(a) UAVs' operation schedules

(b) Separation distance

Figure 3. Results of collision-free trajectory generation

III. Deep Learning-based Inventory Tracking

This section discusses an imagery analysis method to solve an inventory audit problem based on images collected by a UAS platform. In a warehouse or manufacturing environment, each inventory item has an information tag. Among the diverse information printed on the tag, two allow for the product identification. The product ID which consists of a set of unique characters and the barcode. It may happen that some warehouses use RFID technology to perform such an audit. However, RFID technology is often backed up by the type of tag previously described. In the present study it is assumed that no barcode readers are integrated on the UAS platform. Thus, the only way to identify the item is to somehow read the product ID from the scanned image of the item by a UAS platfrom. This is ultimately a text extraction problem. This type of problem has been widely studied because of diverse potential applications such as receipt detection, and car license plate recognition. To address this problem, the most common approach consists of a two-step pipeline, text detection and then text recognition [22]. Text extraction, which is generally called an object detection problem, has experienced great improvements in recent years thanks to the use of the Convolutional Neural Network (CNN) [8, 9, 11]. Hence, in this paper, a CNN architecture is proposed to solve the inventory audit problem. The main challenge here, is that the tag contains numerous text regions, but only one of them contains the product ID. This particular region will be called the Region of Interest (RoI) in the following sections. Then, applying a single text detector followed by a text recognition algorithm is not sufficient since several text regions will be detected.

A topic facing similar challenges is automatic sales receipt detection/recognition. In the same fashion as for the product tags, sale receipts contain several text boxes but all of them are not of interest. Raoui-Outach et al. have proposed a methodology for detecting and understanding sales receipts using deep-learning [16]. In their methodology the algorithm successfully extracts receipt information, which includes detection/localization/classification of receipt and store brand name. Once a text box of interest is detected, an Optical Character Recognition (OCR) algorithm is applied to the text box to extract the relevant information.

Another relevant research field is car license plate recognition, which has gained a lot of attention in the field of Intelligent Transport Systems (ITS). Indeed, the idea is to first detect the license plate as an object and then to perform text recognition or OCR on it. Sliding window methods are the most natural and direct way to perform this kind of object detection [14]. However, it has a computational limitation for large or high-definition images. To resolve this limitation, Li et al. have introduced a CNN-based license plate detection method [13]. The method improves the full sliding window method using a CNN-based region detection that reduces the number of candidates for sliding window. Instead of applying a sliding window structure, Xie et al. have applied Multi-Directional YOLO ("You Only Look Once") [21]. The algorithm is capable of rapid multi-license plate detection using a very deep CNN, which overcomes existing limitations

resulting from variation of view point and accidental rotation of the camera.

Inspired by the previous research topics, sales receipt and license plate detection, a multi-layer CNN architecture, seen in Figure 4, is proposed to perform inventory tracking.

A. Multi-Layer Convolutional Neural Network Architecture

The proposed multi-layer convolutional neural network (CNN) consists of three main steps: Tag region and text box detection, RoI Selection, and Optical Character Recognition (OCR). In the tag region and text box detection, the tag region detection localizes a tag from a collected image. The text box detection identifies all the text box locations based on the image information. In the RoI selection step, the text box with the actual item label information is specified using the previous outcomes: the tag region and text boxes. Then, the final text box is submitted to the OCR algorithm to extract the tag information. Figure 4 illustrates the proposed multi-layer CNN architecture.

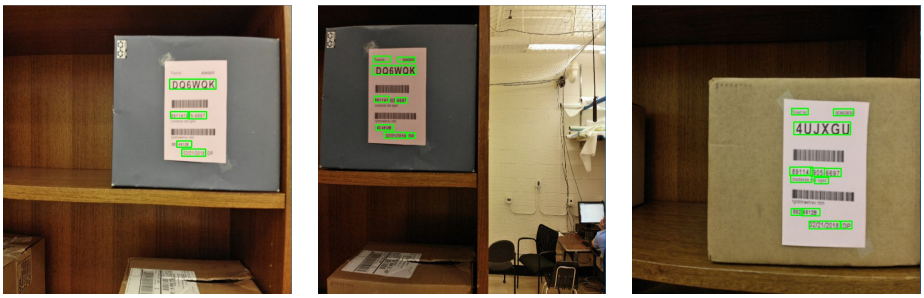
Figure 4. Multi-layer convolutional neural network architecture (Green: CNN-based algorithm, Blue: Simple Crossing Information Algorithm)

1. *Tag region and text box detection*

Tag region detection The objective of tag region detection is to localize the tag. For the localization problem, our approach applies a simple bounding-box regression that has been applied on the R-CNN framework [9]. In the CNN architecture, we apply the pre-trained CNN architecture ResNet34 to reduce the computational time in the training phase, which is also called transfer learning [10]. The transfer learning is a popular technique in CNN-based image analysis since it decreases the required computational time through using the pre-trained model. The proposed CNN architecture combines a fully connected layer with dropout to generate a more flexible learning model. To be more specific, the fully connected layer has one hidden layer. The input layer has 25,088 nodes and the first hidden layer includes 258 nodes. The final layer has 4 nodes that represent bounding box information, the center of the bounding box and its width/height. For the activation function, a ReLu function is employed in the fully connected layer. In the training process, the input image is rescaled into a 224×224 RGB-colored image, and a data augmentation technique such as random flip, rotating image, and variation of lightning is also implemented to improve the training results. To optimize the fully connected layer, the Adaptive Moment Estimation (Adam) optimizer is adopted.

Text box detection The aim of this step is to localize and draw boxes around any line of text on the image. The idea is that among all the text lines detected on the image, one of them will be the region of interest containing the product ID. Such a detection problem is usually a core first step before performing any OCR. Then, due to the increasing popularity in the problem of extracting text from natural images, many text detection methods have been developed in ecent years. As of today, one of the highest-performing

one is the EAST algorithm (Efficient and Accurate Scene Text detector) developed by [23] in 2017. The idea is to feed the full image into a single fully convolutional neural network (FCN). This FCN is comprised of three parts: a feature extractor, a feature merging branch, and an output layer. The feature extractor is usually a deep convolutional network, such as VGG16 [19], pre-trained on ImageNet [6]. Then, features from the feature extracting branch are gradually merged in order to identify text with different sizes. The final feature map of the image is obtained and fed into the output layer. This algorithm reaches state of the art performance in text localization. Particularly, what is the most notable is its speed. Indeed, in most of state of the art text detection methods such as Fast R-CNN, [8] the algorithm is making candidate text region proposals which are then filtered and reduced to actual text regions of the picture. Here, there is no similar proposal step; text regions are directly computed, which makes the EAST among the fastest algorithms to reach such a level of performance. These properties makes it very attractive, EAST trained model has been made publicly available through OpenCV [a]. In our text detection problem, the EAST model was implemented. The results of the EAST model are shown in Figure 5. As one can see, several text boxes are detected due to the presence of several text lines on the tag. It is necessary to filter these text regions to specify the one of interest with actual label information. This is done by crossing the current results with the previous results from tag detection.



(a) (b) (c)

Figure 5. Example outputs from EAST algorithm

2. Region of Interest Selection

The tag detection and text box detection yield the information about the tag location $\mathbf{T} = [x_0 \;\; y_0 \;\; x_1 \;\; y_1]$ and multiple regions of interest about text boxes $\mathbf{T}_{bi} = [x_{0i} \;\; y_{0i} \;\; x_{1i} \;\; y_{1i}]$, $\;\; i = \{1, 2, ..., n\}$. Note that the tag location and the multiple regions indicate two corner points of a bounding box, which are the top left corner and bottom right corner points. The ith location of a text box is one element of all the detected text boxes, $\mathbf{T}_{bi} \in \mathbf{T}_b$. Using the two outcomes, we specify the location of actual label information. The method for identifying the candidate locations, $\mathbf{T}_{lc}$, collects the text box information within the top half of the tag location T_c,

$$x_{0c} = x_0 - \gamma H \tag{12}$$

$$y_{0c} = y_0 - \gamma W \tag{13}$$

$$x_{1c} = \frac{1}{2}(x_0 + x_1) + \gamma H \tag{14}$$

$$y_{1c} = y_1 + \gamma W, \tag{15}$$

where H is the height of the detected tag, $H = x_1 - x_0$, and W is the width of the detected tag, $W = y_1 - y_0$. The variable γ is a margin to prevent a case where the bounding box of the label is located outside of the bounding box of the tag. In the numerical simulation, the parameter γ is defined as 0.1. Based on the extracted candidate bounding boxes, the final label location is determined to be the bounding box with the maximum area among the candidates $\mathbf{T}_{lc}$. Algorithm 2 summarizes the process of the ROI selection.

[a] https://www.pyimagesearch.com/2018/08/20/opencv-text-detection-east-text-detector/

Algorithm 2 ROI selection

Input: $\mathbf{T}_c, \mathbf{T}_{bi}, \gamma$
Output: $\mathbf{T}_l$
for each $\mathbf{T}_{bi} \in \mathbf{T}_b$ **do**
    if $\mathbf{T}_{bi}$ within $\mathbf{T}_c$ **then**
        $\mathbf{T}_{lc} \leftarrow \mathbf{T}_{bi}$
    end if
end for
$\mathbf{T}_l \leftarrow max(area(\mathbf{T}_{lc}))$
return $\mathbf{T}_l$

3. Optical Character Recognition (OCR)

Once the region of interest has been identified and cropped, as in Figure 8(a), the next step is to extract the text information from it. In other words, the objective is to recognize the different characters and then to concatenate them into a string. This problem is commonly called Optical Character Recognition (OCR) and has been of interest for a long time in the computer vision community. Thus, several applications showing good performance are already existing and widely used. Recently two applications showed state of the art performance; Tesseract [20] and the Google Vision API [b]. Both of them were tested and compared, and the results are provided in the following section. The most recent version (Version 4) of Tesseract was used this work and relies on the use of LSTM networks which are a derivative of recurrent neural networks. On the other hand, apart from the quasi-certainty that a deep CNN is utilized somewhere, not much information is available on what type of algorithms the Google Vision API is using. In any case, no tuning whatsoever nor training of the OCR module was required. Thus, the OCR step is really just corresponding to a black-box tool that can be easily switched if needed.

B. Flight Experiment Result

To demonstrate the proposed multi-layer CNN architecture, a small-scale inventory environment shown in Figure 6 is created. In the flight experiment, a DJI Mavic Air is manually flown in front of each item on the shelf and takes pictures of the items. It is of note that a common format of the tag is used and that every UAV picture contains a single tag.

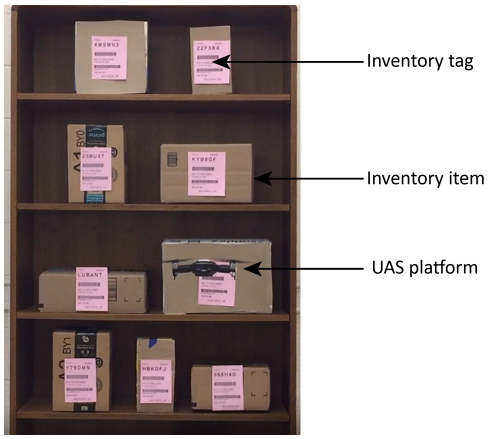


Figure 6. Simplified flight experiment setup

The process is repeated several times by changing the positions of the items on the shelf as well as their corresponding tags. Using this flight experiment environment, 2,630 tag images are collected. Figure 7 shows the examples of collected images. 2,104 images are utilized for the training of the tag detection algorithm, and 264 images are used as a test set for the framework. The text box detection CNN and OCR applications

[b] `http://cloud.google.com/vision`

don't need any training since a pre-trained model are used from public sources. Among the 264 test images, 96% of the product IDs (253 pictures) are correctly cropped after the RoI selection step. The product ID is considered correctly cropped if it contains the six characters entirely, displayed in Figure 8.

Using the 253 correctly cropped product IDs, the performance of both OCR applications is evaluated. It is observed that due to some uncertainties like blurring or distortion both OCR applications often confuse the "O" and "0" characters and also the "I" and "1" characters. This is expected because even the human eye in such conditions may confuse them. Thus, two types of OCR are performed. A strict OCR in which every character has to be strictly identified, and a soft OCR in which "I" and "1" on one hand and "0" and "O" on the other hand are considered the same characters. Finally, the overall performance of the framework is simply the product of the accuracy after the RoI selection multiplied by the accuracy of the OCR step. These results are summarized in Table 2. In the result, the multi-layer CNN with Google Vision API Soft OCR presents the best performance, which is around 87 %. The result also shows that Google Vision OCR has better performance than Tesseract OCR in both the Soft and Strict cases.

Figure 7. Example images from the flight experiment

(a) Positive images

(b) Negative images

Figure 8. Results of ROI selection

Figure 9. Example of Convolutional Neural Network Architecture (Preliminary result from the proposed first CNN architecture)

Table 2. Accuracy of region detector and text recognition

Layers	ROI selection	OCR	Overall Performance
Multi-layer CNN with Tesseract Strict OCR	96 %	66%	63%
Multi-layer CNN with Tesseract Soft OCR	96 %	79%	76%
Multi-layer CNN with Google Vision API Strict OCR	96 %	73%	70%
Multi-layer CNN with Google Vision API Soft OCR	96 %	91%	87%

IV. Conclusions and Future Work

In this paper, a framework for multi-UAV trajectory optimization to scan the entire inventory space and a multi-layer CNN architecture to track inventory items have been introduced. In the multi-UAV trajectory optimization, we proposed a two-step process to generate a non-collision flight trajectory, which is based on the distance-constrained vehicle routing problem and the optimization of a collision-free UAS schedule. In the numerical simulation, the proposed multi-UAV trajectory optimization shows that resulting trajectory meets the vehicle endurance requirement, minimizes the total mission time, and leads to a non-collision trajectory. In the imagery analysis to track inventory items, we proposed a multi-layer CNN architecture that contains tag detection, text box detection, ROI selection, and OCR. The proposed framework is demonstrated by flight experiment and compares the performance of two different approaches using two OCR algorithms, Tesseract and Google Vision API. In the soft OCR case, the multi-layer CNN with Google Vision API has outperformed with an overall performance of 87 %.

Potential future work related to multi-UAV trajectory optimization can include developing a computationally faster and simpler optimization framework instead of using a two-step process, as well as scaling up the size of the inventory problem so that it can solve an entire warehouse. Another potential area of research regarding the imagery analysis is that this paper only handles a tracking problem with one tag per image, but it is possible to have multiple tags per image. As an extension of this research, one can create multi-tag recognition algorithms through applying diverse deep-learning algorithms such as Single Shot MultiBox Detector (SSD), YOLO, and Faster-RCNN [15][17][18].

Acknowledgments

We would like to thank DroneX team members in the Aerospace Systems Design Laboratory (ASDL) of Georgia Institute of Technology who participate in this research including flight experiments.

References

[1] Gustavo SC Avellar, Guilherme AS Pereira, Luciano CA Pimenta, and Paulo Iscold. Multi-UAV routing for area coverage and remote sensing with minimum time. *Sensors*, 15(11):27783–27803, 2015.

[2] Helen Brown-Liburd and Miklos A Vasarhelyi. Big data and audit evidence. *Journal of Emerging Technologies in Accounting*, 12(1):1–16, 2015.

[3] Y. Choi, A. Payan, S. Briceno, and D. Mavris. A framework for unmanned aerial systems selection and trajectory generation for imaging service missions. *2018 AIAA Aviation and Aeronautics Forum and Exposition*, 2018.

[4] Youngjun Choi, Younghoon Choi, Simon Briceno, and Dimitri N. Mavris. Three-dimensional UAS trajectory optimization for remote sensing in an irregular terrain environment. In *Unmanned Aircraft Systems (ICUAS), 2018 International Conference on*. IEEE, 2018.

[5] Youngjun Choi, Hernando Jimenez, and Dimitri N Mavris. Two-layer obstacle collision avoidance with machine learning for more energy-efficient unmanned aircraft trajectories. *Robotics and Autonomous Systems*, 98:158–173, 2017.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. Ieee, 2009.

[7] Enric Galceran and Marc Carreras. A survey on coverage path planning for robotics. *Robotics and Autonomous systems*, 61(12):1258–1276, 2013.

[8] Ross Girshick. Fast R-CNN. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[9] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[11] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

[12] Imdat Kara. Arc based integer programming formulations for the distance constrained vehicle routing problem. In *Logistics and Industrial Informatics (LINDI), 2011 3rd IEEE International Symposium on*, pages 33–38. IEEE, 2011.

[13] Haixiang Li, Ran Yang, and Xiaohui Chen. License plate detection using convolutional neural network. In *Computer and Communications (ICCC), 2017 3rd IEEE International Conference on*, pages 1736–1740. IEEE, 2017.

[14] Hui Li and Chunhua Shen. Reading car license plates using deep convolutional neural networks and LSTMs. *arXiv preprint arXiv:1601.05610*, 2016.

[15] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. SSD: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[16] Rizlene Raoui-Outach, Cecile Million-Rousseau, Alexandre Benoit, and Patrick Lambert. Deep learning for automatic sale receipt understanding. In *Image Processing Theory, Tools and Applications (IPTA), 2017 Seventh International Conference on*, pages 1–6. IEEE, 2017.

[17] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[18] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[19] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[20] Ray Smith. An overview of the tesseract OCR engine. In *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on*, volume 2, pages 629–633. IEEE, 2007.

[21] Lele Xie, Tasweer Ahmad, Lianwen Jin, Yuliang Liu, and Sheng Zhang. A new CNN-based method for multi-directional car license plate detection. *IEEE Transactions on Intelligent Transportation Systems*, 19(2):507–517, 2018.

[22] Qixiang Ye and David Doermann. Text detection and recognition in imagery: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 37(7):1480–1500, 2015.

[23] Xinyu Zhou, Cong Yao, He Wen, Yuzhi Wang, Shuchang Zhou, Weiran He, and Jiajun Liang. EAST: an efficient and accurate scene text detector. In *Proc. CVPR*, pages 2642–2651, 2017.

[24] Thomas Guthrie Zimmerman. System and method for performing inventory using a mobile inventory robot, April 6 2010. US Patent 7,693,757.