IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 13 July 2024, accepted 5 August 2024, date of publication 7 August 2024, date of current version 19 August 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3440041

RESEARCH ARTICLE

An Automated Method for Container Counting in Satellite Images Based on Grid Analysis and Shadow Recognition

BOYANG ZHOU[1] AND ZHENQUAN WANG[2]
[1]Business School, Jiangsu University of Science and Technology, Zhangjiagang Campus, Suzhou 215600, China
[2]Department of Finance, Xiamen University, Xiamen 361005, China

Corresponding author: Zhenquan Wang (15220212202658@stu.xmu.edu.cn)

ABSTRACT This study developed a novel and reusable technique aimed at automatically estimating the number of containers in port storage yards using satellite images, namely, an automated method for container counting in satellite images based on grid analysis and shadow recognition. Unlike traditional machine learning methods that rely on large-scale labeled datasets, this method does not require extensive data collections, significantly reducing the implementation threshold and costs. By dividing the container yards into grids, researchers were able to segment large-scale yard images into smaller blocks, allowing for individual block analysis. This not only improves processing speed but also enhances estimation accuracy. Converting images from the RGB color space to the HSV color space allows the algorithm to more accurately analyze and identify containers and their shadows. Furthermore, this study leverages the features of container shadows, which exhibit specific shapes and orientations in satellite images, to estimate container stacking heights. The number of containers in each grid is determined by comparing the shadow area with the known shadow area of a single container, making the overall calculation process more intuitive and reliable. Compared to the U-Net model based on semantic segmentation, this method significantly improves efficiency, reducing average program runtime by 72.90%, and greatly reducing the mean absolute error (MAE) and root mean square error (RMSE) of predictions. Compared to manual counting methods, the average time consumption of this method is also reduced by 69.75%, significantly enhancing efficiency. In summary, this integrated method of grid division, HSV color conversion, and shadow analysis provides a highly cost-effective and accessible tool for third-party researchers and operators. With appropriate satellite images, it enables real-time estimation of the number of containers in port storage yards.

INDEX TERMS Automated container detection, container count estimation, grid analysis technique, HSV color transformation, satellite imagery, shadow analysis.

I. INTRODUCTION

With the intensification of global regional conflicts, the instability of supply chains has become increasingly significant. This makes timely acquisition of key data from logistics nodes crucial for making quick and accurate decisions. In the shipping industry, in particular, it is essential for shipping companies to accurately understand the congestion status of various ports and terminals to evaluate their operational efficiency and select appropriate ports. Moreover, from an environmental perspective, improving port operational efficiency is not only necessary but also urgent, as it can significantly reduce the time ships spend in port, thereby reducing fuel consumption and emissions to promote the sustainable development of the port industry.

Factors influencing port efficiency are multifaceted, including the availability of berths and terminal facilities, the quality of connections with road and rail services, competitiveness, and the number of cranes at each port and the utilization rate of container yards corresponding to the berths. However, a major challenge in evaluating these efficiencies is the lack of uniform standards, making

The associate editor coordinating the review of this manuscript and approving it for publication was Andrea De Marcellis.

it difficult to compare different ports. To address this issue, the industry has conducted extensive research aimed at establishing a set of recognized efficiency evaluation standards.

While discussing port efficiency, the importance of containers cannot be overlooked. Containers, as the basic units of cargo transportation in international trade, have a circulation efficiency that directly affects the overall operational efficiency of the supply chain. Optimizing the processes of loading, unloading, storing, and transshipping containers can enhance the handling capacity of ports, significantly shorten cargo turnover times, reduce waiting times at ports, and further improve the overall efficiency of the supply chain while lowering related costs. Therefore, container management and optimization become another key point in improving port efficiency.

However, due to the lack of relevant data on port facilities (especially ship berths and container yards), most studies have focused on detecting ships in images rather than obtaining information on container congestion at ports. Only a few studies have addressed this aspect, and they have notable shortcomings. For instance, the Japan Aerospace Exploration Agency used ALOS-2 PALSAR-2 [1] and Sentinel-1 [2] images to regularly observe the container storage areas at Nagoya Port post-COVID-19, determining the changes in cars and containers since the outbreak. However, this study only focused on changes in the planar direction of containers and could only provide rough estimates due to the inability to consider vertical stacking. Yu et al. [3] used Sentinel-2 optical images to count the number of pixels classified as containers in each satellite image, using this as a proxy for the number of containers in the port. They calculated the daily average number of containers in the port to predict the relationship between port container numbers and economic activity. However, this method could not accurately identify container stacking levels due to limited spatial resolution. Additionally, the update frequency of satellite images might not align with the frequency required for predictions, leading to data delays or inaccuracies. Yasuda et al. [4] proposed a labeling tool using Google satellite images to estimate the number of containers in container yards by combining shadows for congestion classification. However, this tool required manual labeling of container-related data, making it inefficient for estimating container numbers in larger container yards.

To address the issues of inability to recognize container stacking levels, data inaccuracies, and low efficiency due to manual operations in the aforementioned studies, this research proposes a method that does not require large datasets and uses a simple, reproducible approach to automatically estimate the number of containers in large port container yards solely based on satellite images. The most significant challenge in achieving the goal of estimating the total number of containers in container yards from satellite images at various times is extracting the height information of stacked containers. Therefore, this research leverages the standardized nature of containers, focusing on the shadows cast by individual containers at the same time to determine the height of stacked containers while keeping the original satellite images unchanged, thereby estimating the total number of containers.

Furthermore, this research is particularly suitable for third-party researchers who cannot directly obtain operational data from port operators. For these researchers, obtaining data through conventional statistics and field surveys is not only costly but also inefficient. The automated method proposed in this study, relying on advancements in aerospace technology and the widespread availability of low-cost, high-frequency satellite images, provides a simple yet effective solution.

Experimental results of the proposed automated method indicate that, compared to the U-Net model based on semantic segmentation, this method significantly improves efficiency, reducing average program runtime by 72.90%, and greatly reduces MAE and RMSE. Compared to manual counting methods, the average time consumption of this method is also reduced by 69.75%, significantly enhancing efficiency. Additionally, the automated counting method has a MAE of 39 and a RMSE of 59, which is relatively small for a total container count of 5629, within an acceptable range. Overall, the automated container quantity estimation method proposed in this study effectively addresses previous issues of inability to recognize container stacking levels, large errors, and the need for manual operations.

The rest of this paper is organized as follows: Section II summarizes the relevant literature and data cited in this study and positions the research within the existing body of work; Section III explains the basic principles of the automated method; Section IV selects satellite images of the Cape Town port container yard, where data could not be obtained in previous research, as a practical application case; and finally, Section V concludes the study, pointing out the limitations of the method and directions for future development.

II. LITERATURE REVIEW

In recent years, relevant research has mainly focused on ship detection and classification, port activity analysis, monitoring of port infrastructure, and combining these data to predict economic indicators. These studies are not limited to simple ship identification but extend to complex analyses of marine and port environments.

In ship detection and classification, Wang et al. [5] used visual saliency detection and morphological filtering methods to improve the accuracy of ship detection in complex sea surface scenes, obtaining the positions and types of ships in large-scale maritime SAR images. Liu et al. [6] introduced a deep convolutional neural network method, using residual networks and initial layers to enhance the accuracy of ship detection and classification in remote sensing images, achieving 95% classification accuracy. Additionally, Zakharov et al. [7] proposed identification methods based on advanced deep learning models such as Faster R-CNN and YOLOv4, achieving classification

accuracy as high as 99% in high-resolution images. Li et al. [8] provided a comprehensive review of ship detection methods, emphasizing the importance of feature extraction and the need for large, reliable datasets to improve model performance.

In port activity analysis, Hassan [9] first outlined computer simulation models used to manage port activities, emphasizing their role in optimizing operations and planning expansions. Bonilla et al. [10] used data envelopment analysis to analyze the efficiency of Spanish ports, focusing on the relationship between port equipment and cargo transportation. Kondratyev [11] explored an object-oriented approach to modeling port activities, aiming to improve port efficiency through better modeling and simulation frameworks. Worth et al. introduced Power BI reports for real-time analysis of port docking data to identify operational bottlenecks and improve efficiency [12]. Sutrisnowati et al. [13] used Bayesian networks to analyze delays in port logistics, providing insights for improved scheduling and reduced delays. Moreover, Li et al. [14]'s research on ensemble learning techniques could enhance data processing accuracy and consistency, providing insights for port activity analysis, particularly in the integration of multiple sensor data (such as radar and optical images).

In monitoring port infrastructure, Alexakos and Konstantinopoulos [15] discussed integrated middleware for monitoring port environmental parameters, including a web-based GIS interface for data management. Jinguang et al. [16] proposed a comprehensive environmental monitoring system with early warning and linkage control to enhance the sustainability of port operations. Alam et al. [17] explored the use of heterogeneous autonomous vehicles for coastal infrastructure monitoring, providing a sophisticated approach to enhance port security and operational resilience. Shahraji and Larouche [18] introduced a rigorous method to calibrate marine mobile LiDAR systems for precise infrastructure monitoring, highlighting advances in port sensor technology applications. Pandey [19] discussed the implementation of automatic directional noise monitoring systems in ports to effectively manage and mitigate noise pollution. Wooldridge et al. [20] emphasized the importance of scientific monitoring in implementing port environmental management policies, utilizing comprehensive data collection to support sustainable operations. Liu et al. [21] introduced ASHFormer's axial and sliding window attention mechanisms for analyzing high-resolution images and sensor data from port infrastructure, aiding in the detection and correlation of patterns indicating structural anomalies or degradation over time.

Building on these studies, researchers have further attempted to combine ship identification, port activity analysis, and port infrastructure monitoring to reasonably predict economic activity fluctuations. Feng et al. [22] used AIS data to analyze the dwell time of different types of ships in different port areas, thereby assessing port operational efficiency. Chen et al. [23] used genetic programming to develop an optimal prediction model to forecast container throughput at major ports in Taiwan. Their model significantly outperformed traditional models such as SARIMA. Lee et al. [24] proposed a deep learning model combining time series decomposition with LSTM for predicting container volumes, demonstrating accuracy superior to traditional methods. Liu [25] utilized grey theory to predict economic outcomes based on port and ocean engineering data (including container throughput), emphasizing its economic impact. Brooks et al. [26] found, through historical data and econometric models, that counties near U.S. container ports grew significantly due to containerization. Nieto et al. [27] compared several forecasting models for port freight volumes, finding certain models more effective for specific types of cargo. Twrdy et al. [28] provided forecasts and dynamic models for container throughput at North Adriatic ports, useful for understanding economic activities in the region.

TABLE 1. External dimensions of primary containers (Source: International standard organization).

Container Type	Length (feet)	Width (feet)	Height (feet)
20 Feet	20 Feet	8 Feet	8.5 Feet
40 Feet	40 Feet	8 Feet	8.5 Feet
40 Feet High Cube	40 Feet	8 Feet	9.5 Feet

In summary, to the best of the authors' knowledge, research using satellite images to estimate container numbers is still very scarce. For example, Yasuda et al. [4] utilized satellite images to propose a labeling tool that attempts to classify the congestion level of container yards by combining shadows and automatic identification system (AIS) data, analyzing terminal congestion at container ports. However, this labeling tool requires manual labeling of containers, making it inefficient when dealing with large datasets. Therefore, this study proposes an automated container estimation method based on grids and shadows. By using HSV color conversion, focusing on the shadows cast by containers, and employing grid analysis techniques, this method allows accurate estimation of container stacking from a single snapshot image. This approach provides a cost-effective and accessible tool for third-party researchers and operators, enabling simple and rapid estimation of container numbers in port yards.

III. METHODOLOGY

A. OVERVIEW OF THE METHOD

In this study, since 20 foot and 40 foot containers are the mainstream sizes for the majority of containers on the market, as a result, these two main container sizes were considered to construct this method. The standardization of these two sizes greatly simplifies the image processing process as they provide consistent reference standards, as shown in Table 1.

It is not realistic to manually count the vertically stacked containers in the yard visually by quickly browsing satellite images to estimate the density of containers, as containers

are stacked vertically and in large quantities in the yard. Therefore, most studies focus on detecting the number of containers in the storage area in the plane direction. However, detecting the number of containers stacked vertically is more challenging.

Firstly, based on the significant regularity and standardization of container yards and containers, this study fully utilizes the regular geometric shape and color features of containers in satellite images under the same satellite image conditions, and effectively calculates the location of containers in the yard through a series of quantitative steps.

Secondly, to extract the height information of stacked containers, this study focuses on the shadows commonly used to estimate the height of certain objects, which are generated by the containers, to address this issue. In order to more effectively identify and count containers, this study did not use a simple binarization method, but instead used the HSV model for image processing. The HSV (Hue-Saturation-Value) model provides a non-linear way to analyze colors, where:

Hue represents the type of color.

Saturation represents the purity of a color.

Value represents the brightness of a color.

By adjusting these three parameters, the colors and shadows of the containers can be more accurately distinguished, thereby enabling a precise estimation of the number of stacked containers.

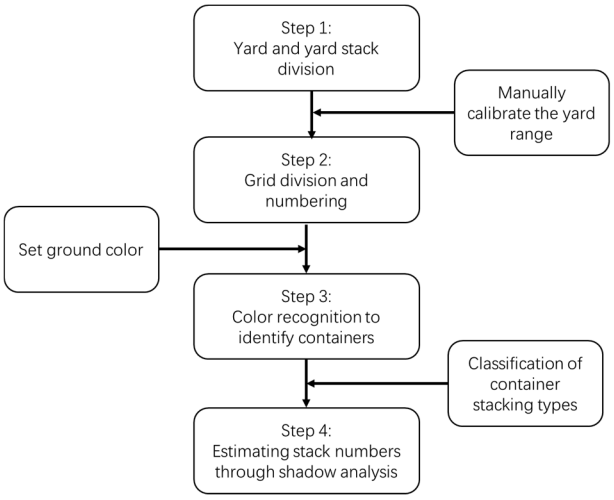

FIGURE 1. Flowchart of the automatic container counting method in satellite images based on grid analysis and shadow recognition.

The specific process of the method is described in detail below, as shown in Figure 1:

Step 1: Yard and Yard Stack Division

Identify and mark the boundaries of the container yard and the stacks within it using rectangular outlines on the satellite image. Further divide the yard into multiple stacks, each bounded by rectangles. Then, establish a coordinate axis for the yard.

Step 2: Grid Division and Numbering

Within each stack, divide the area into grids the size of a single container. These grids are then sequentially numbered.

Step 3: Color Recognition to Identify Containers

Utilize the color information in the satellite images to differentiate between containers and the ground. Since containers usually have distinct, vivid colors that starkly contrast with the ground, color filtering can effectively identify which grids contain containers.

Step 4: Estimating Stack Numbers through Shadow Analysis

Estimate the number of stacks by analyzing the height differences in the shadows of the containers. Set a standard height for the shadows based on the shadow of a single container as a reference. Combine the results of color recognition and shadow analysis within each grid to determine the number of containers in each grid. Sum up the data from all grids to calculate the total number of containers for each stack and for the entire yard.

B. DETAILED DESCRIPTION OF THE METHOD

1) STEP 1: YARD AND YARD STACK DIVISION

Identifying the overall boundaries of the container yard in satellite images is the preliminary and foundational step in the entire image analysis process. Typically, these yards are displayed as regularly shaped rectangles arranged in an orderly fashion along the docks. This regular geometric shape is due to the container yard layout usually being designed according to strict planning to maximize the efficiency of ground use at the port.

Within the yard, several independent stacking areas are also divided, each containing several rows or columns of stacked containers. These stacks are separated by spacious pathways, wide enough to accommodate container trucks and other handling equipment, allowing for their passage. Figure 2 shows the layout of a typical container yard.

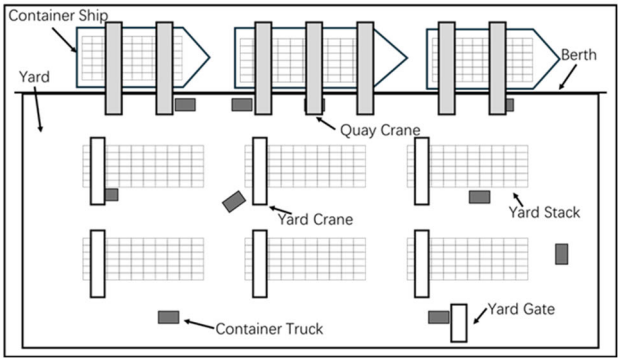

FIGURE 2. Typical layout of a container yard.

In the first step of this method, it is necessary to identify and outline the area of the container yard on a satellite image using rectangular frames, and then frame all the stacks within this yard, as shown in Figure 3. This method utilizes labelImg to outline the relevant areas and store the data. labelImg is a

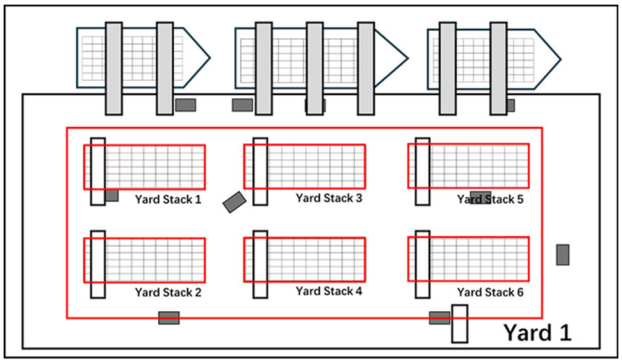

FIGURE 3. Identify and outline the container yards and stacks.

widely used image annotation tool, primarily for generating image labels and bounding box coordinates needed during the machine learning training process.

Next, establish a coordinate axis with the top left corner of the rectangle of the framed yard as the origin O, as shown in Figure 4.

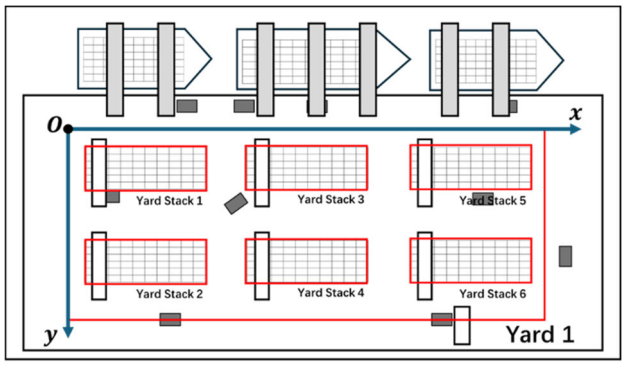

FIGURE 4. Establish coordinate axes for the container yards.

The origin point O is located at the top left corner of the framed rectangle Yard 1, which means the coordinates of the origin are (0, 0). The X-axis extends from left to right, and an increase in X coordinate value signifies a move to the right side of the image. The Y-axis extends from top to bottom, and an increase in Y coordinate value signifies a move towards the bottom of the image. Therefore, the position of the framed rectangle can be represented as:

$$\text{Rectangular position} = (x_{min}, y_{min}, x_{max}, y_{max}) \quad (1)$$

where:

x_{min} and y_{min} represent the minimum values on the horizontal and vertical axes for the top-left corner of the rectangle, hence (x_{min}, y_{min}) can be defined as the coordinates of the top-left corner of the bounding box.

x_{max} and y_{max} represent the maximum values on the horizontal and vertical axes for the bottom-right corner of the rectangle, hence (x_{max}, y_{max}) can be defined as the coordinates of the bottom-right corner of the bounding box.

This means that Yard 1 in Figure 4 can be represented as (0, 0, 946, 382). Similarly, Yard Stacks 1 to 6 can also be represented in this coordinate format. It should be noted that the distance unit used in the images for this method is pixels, which is a very common and fundamental unit in computer vision and image processing. Pixels, or picture elements, are the basic units that compose digital images. In digital imagery, each pixel represents a small part of the image in terms of color and brightness information.

2) STEP 2: GRID DIVISION AND NUMBERING

In this method, within each stack area, the identification of individual containers is further achieved by subdividing the entire area into grids that match the dimensions of a single container. The specific process is as follows:

Grid Size Setting: Based on the common standard sizes of containers, the grid dimensions are set to either 20 feet × 8 feet or 40 feet × 8 feet, to match the standard lengths of containers. In practice, this setting ensures that each grid can precisely accommodate one container, whether it is 20 feet or 40 feet long.

Grid Layout: The grids are laid out according to the actual shape of the stacking area. In most cases, although 20-foot and 40-foot containers are mixed in the stacks, the containers in the same column always have the same dimensions. These grids are neatly arranged in rows and columns to form a regular grid pattern. This orderly arrangement helps simplify the process of accessing containers and enhances operational efficiency.

Number the Grid: For ease of management and localization, each grid in the stack is assigned a unique number. The numbering usually follows an order from one end of the stack area to the other, from one corner to the diagonal opposite, as shown in Figure 5. This sequential numbering system not only aids in visually identifying the location of containers quickly but also facilitates automated recognition and calculation by computers.

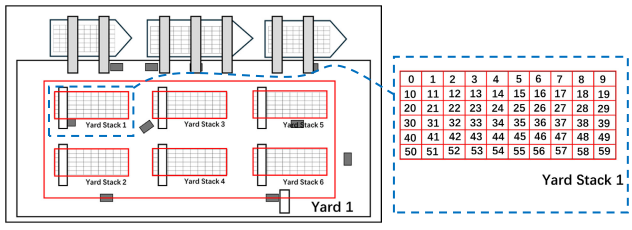

FIGURE 5. Grid division of container yard and marking of serial numbers.

In summary, the numbering of a grid can be represented as:

$$\text{Grid}_n\,(i, j) = N \quad (2)$$

where:

N is the stacking number to which the grid belongs.
i is the row of the stack to which the grid belongs.
j is the column of the stack to which the grid belongs.

n is the number of the grid in the stack.

For example, the grid numbered 0 in Yard Stack 1 can be represented as $Grid_1\,(1,1) = 0$. Similarly, each grid can be represented in the form of (xmin, ymin, xmax, ymax) based on the coordinate axes constructed in this section.

3) STEP 3: COLOR RECOGNITION TO IDENTIFY CONTAINERS

After completing the grid division, the next crucial step involves using a color recognition algorithm to determine whether each stack grid area in the satellite image contains containers. This process is implemented using relevant Python libraries and involves image processing, color space conversion, and statistical analysis, as illustrated in Algorithm 1.

The following is a detailed description of the implementation process of this algorithm, and an example is given to illustrate the placement of containers in a fictional stack in Figure 6.

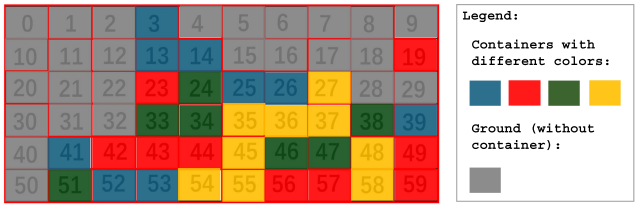

FIGURE 6. Container stacking situation in each grid of a certain yard (an example).

First, load the image and construct the container yard grid using the grid coordinates stored in the related files. Then, convert the color of the grid areas to the HSV color space. The HSV color space offers a different way of representing colors compared to the RGB color space. The HSV model is frequently used in image processing and analysis because it better reflects the visual perception of colors, especially under varying lighting conditions. HSV encodes colors in terms of hue, saturation, and value, making color comparison and processing more intuitive.

Each color in the HSV color space can be converted from the RGB color space using the following steps:

a: NORMALIZED RGB VALUE

The RGB values are usually between 0 and 255, and first they need to be converted to the range of 0 to 1:

$$\mathrm{R}', \mathrm{G}', \mathrm{B}' = \frac{\mathrm{R}}{255}, \frac{\mathrm{G}}{255}, \frac{\mathrm{B}}{255} \tag{3}$$

where:

R, G, B : These are the color values of the red, green, and blue channels, directly derived from the RGB color space of the image. In most image processing libraries, the range of these values is usually 0 to 255.

R', G', B' : These are normalized RGB values that convert the original RGB values to a range of 0 to 1. Normalization is achieved by dividing each color value by 255.

Algorithm 1 Color Recognition Algorithm Used to Determine the Presence of Containers

1. Import Required Libraries
Import libraries for image processing, numerical computations, and data manipulation.
2. Read Image File
Load the image from the specified path.
3. Read CSV File for Grid Coordinates
Load grid coordinate data from a CSV file into a DataFrame.
4. Select Known Empty Grids
Create a list of tuples containing areas and their corresponding grid indices that are known to be empty.
Initialize an empty DataFrame to store selected grids.
Loop through each area and index pair, filter the grids from the DataFrame, and append the results to the empty DataFrame.
5. Initialize List for Ground Colors
Create an empty list to store ground color values.
6. Extract Color Values from Selected Grids
Loop through each selected grid:
Crop the grid area from the image.
Convert the cropped grid area to the HSV color space.
Calculate the average color value in HSV space.
Add the average color value to the list of ground colors.
7. Calculate Representative Ground Color
Compute the average color from the extracted ground colors.
Handle the hue values specially, as they are circular, to ensure they remain within the correct range.
8. Define Function to Check for Containers
Define a function that:
Takes the image, grid DataFrame, ground color, and a threshold as input.
Converts the ground color to lower and upper bounds for the HSV range.
Iterates through all grids, crops the grid area, and converts it to HSV space.
Creates a mask based on the HSV range.
Calculates the proportion of ground color in the grid area.
Determines if the grid has a container based on the proportion and threshold.
Stores the results in a list and adds it to the DataFrame.
Return the updated DataFrame.
9. Define Representative Ground Color in HSV
Set the representative ground color in HSV values.
10. Invoke Function and Output Results
Call the function with the image, grid DataFrame, representative ground color, and threshold.
Print the resulting DataFrame with container presence information.
11. Save DataFrame with Container Results to CSV
Save the DataFrame containing container presence results to a CSV file.
12. Draw Containers on Image and Save
Define a function to draw rectangles around grids with containers on the image.
Apply the function to the image and save the resulting image with drawn containers.

b: CALCULATE C_{max}, C_{min} AND THEIR DIFFERENCES Δ

$$\mathrm{C}_{\max} = \max(\mathrm{R}', \mathrm{G}', \mathrm{B}') \tag{4}$$

$$\mathrm{C}_{\min} = \min(\mathrm{R}', \mathrm{G}', \mathrm{B}') \tag{5}$$

$$\Delta = \mathrm{C}_{\max} - \mathrm{C}_{\min} \tag{6}$$

where:

C_{max} : This is the maximum value among normalized RGB values (the largest among R', G', B'). This represents the strongest color intensity among the observed pixels.

C_{min} : This is the minimum value among normalized RGB values (the smallest of R', G', B'). This represents the weakest color intensity among the observed pixels.

Δ (Delta): This is the difference between the maximum and minimum normalized RGB values, used to calculate saturation and hue.

c: CALCULATE HUE (H), SATURATION (S), AND VALUE (V)

The calculation of hue depends on which color (red, green, or blue) is the maximum value and is cyclic (from 0 degrees to 360 degrees):

$$H = \begin{cases} 60^\circ \times \left(\frac{G' - B'}{\Delta} \bmod 6\right), \text{if } C_{max} = R' \\ 60^\circ \times \left(\frac{B' - R'}{\Delta} + 2\right), \text{if } C_{max} = G' \\ 60^\circ \times \left(\frac{R' - G'}{\Delta} + 4\right), \text{if } C_{max} = B' \end{cases} \quad (7)$$

If $\Delta = 0$ (i.e. the area is gray), then $H = 0$.

Saturation represents the intensity or purity of a color:

$$S = \begin{cases} 0, & \text{if} C_{max} = 0 \\ \frac{\Delta}{C_{max}}, & \text{otherwise} \end{cases} \quad (8)$$

Value represents the brightness of a color:

$$V = C_{max} \quad (9)$$

where:

H : The color tone represents the type of color (such as red, green, or blue, etc.). It is measured in degrees, ranging from 0 to 360 degrees. The calculation of color tone depends on the channel where the maximum color value is located (R', G', B').

S : Saturation represents the intensity or purity of a color, ranging from 0 to 1. High saturation colors appear "purer", while low saturation colors are closer to gray.

V : Brightness refers to the brightness or intensity of a color, ranging from 0 to 1. High brightness indicates brighter colors, while low brightness indicates darker colors.

$mod6$: Used in tone calculation to ensure that the results are within the correct range (especially when using formulas to calculate tones). It represents modulo 6 operation, used to handle periodic tone values.

d: AVERAGE COLOR CALCULATION

Once the color of an image area is converted into HSV space, the average HSV value of all pixels can be simply calculated to represent the average color of that area. This is usually achieved by averaging the H, S, and V values of all pixels separately. However, for hue (H), as it is an angle value, vector averaging should be used to calculate:

$$\text{average Hue} = \text{atan2}(\sum(\sin(H_i)), \sum(\cos(H_i))) \quad (10)$$

where:

H_i : The color tone value converted to radians.

Subsequently, it is necessary to manually select grids that do not contain containers and calculate the average HSV values of the pixels within these grids to represent the average color of the ground. For example, grids 0, 1, 2, 10, 11, and 12 in Figure 5 could be chosen. The calculated average ground color value is used as a threshold to assess each grid individually. If the proportion of the ground color in a grid exceeds this threshold, it is determined that there are no containers in that grid, and its value is set to False; otherwise, it is set to True indicating the presence of containers.

FIGURE 7. Identification results of grids with containers (an example).

Figure 7 shows the identification results of grids containing containers under ideal conditions, with gray representing the ground and blue indicating areas with containers. However, in practical application, the recognition process can be easily affected by various obstructions, such as cranes, clouds, etc. Similarly, variations in ground color can also lead to misidentification. These issues will be further analyzed in Section IV, which discusses practical applications.

4) STEP 4: ESTIMATING STACK NUMBERS THROUGH SHADOW ANALYSIS

In the previous section, the presence of containers within a grid is determined by setting a threshold for HSV values. Similarly, this method can also be used to identify shadows in the image. Specifically, if all HSV values of a pixel are below the threshold set for shadows, then it is classified as a shadow pixel, and its value is set to True; otherwise, the pixel value is set to False, as shown in Algorithm 2. This threshold can be determined through repeated experimentation.

After successfully identifying shadows and collecting shadow data, the next step can be taken. The purpose of extracting shadows is to accurately assess the number of layers in a stack of containers, and the calculation of these layers primarily utilizes the standardized height characteristic of containers. The basic principle is that due to the consistency in container design, apart from length, their heights are mostly uniform. This uniformity means that the height of a single container's shadow observed in the same satellite image is constant, and this fixed shadow height h along with another dimensional shadow height d can be defined as the standard shadow height for a single container in that

Algorithm 2 Identify Shadows in a Specified Area and Save Data

1. Import Required Libraries
Import libraries for image processing, numerical computations, data manipulation, and XML parsing.
2. Define Function to Parse XML
Define a function to parse XML files containing area coordinates:
Parse the XML file.
Extract the root element.
Initialize a dictionary to store area coordinates.
Loop through each 'object' element in the XML:
Get the label (name) of the area.
If the label contains "Container Area":
Extract the bounding box coordinates (xmin, ymin, xmax, ymax).
Store the coordinates in the dictionary with the label as the key.
Return the dictionary containing area coordinates.
3. Define Function to Save Shadow Data to CSV
Define a function to save shadow data to a CSV file:
Load the image from the specified path.
Convert the image to the HSV color space.
Parse the XML file to get the area coordinates.
Define the HSV color range for shadows (lower and upper bounds).
Create a mask for shadows using the HSV color range.
Find the indices of shadow pixels in the mask.
Create a DataFrame with the x and y coordinates of shadow pixels.
4. Filter Shadow Points Within Specified Areas
Define a helper function to check if a point is within any of the specified areas:
Loop through each area in the dictionary:
Check if the point's coordinates fall within the area's bounding box.
Return True if the point is within the area, otherwise return False.
Apply the helper function to the DataFrame to filter out points not within the specified areas.
Remove the helper function's column from the DataFrame.
5. Save Filtered Data to CSV
Save the filtered DataFrame to a CSV file.
6. Example Usage
Call the function with the image path, XML path, and output CSV path.

satellite image. By this method, it is possible to infer the number of stacked containers quite accurately by comparing the standard shadow height of a single container with the actual shadow heights of multiple stacked containers, as demonstrated in Figure 8 and Algorithm 3.

It must be emphasized again that this estimation method is designed for situations where direct measurement of container heights is not possible, such as when observing remotely via satellite or aerial images. Therefore, it has certain limitations, which will be discussed in the concluding section of the paper.

The following is a detailed analysis of various stacking situations of containers. In the actual operation process, the stacking of containers can be divided into the following three types:

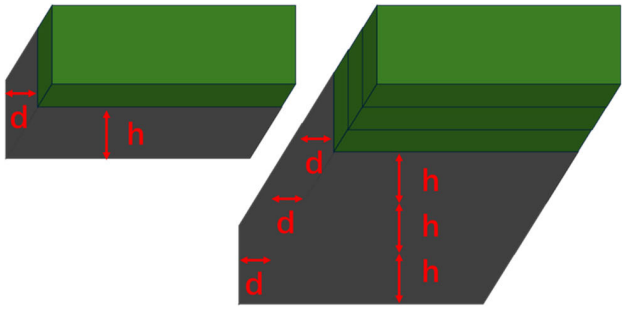

FIGURE 8. Setting of container standard height.

a: SAME NUMBER OF STACKS PER COLUMN

In the ideal stacking scenario of containers, the stacking height of each column of containers is the same. Therefore, to calculate the total number of stacks for an entire column, it is usually only necessary to focus on the shadow of a specific container. Assuming that the shadows of the containers primarily point southward, it suffices to identify the shadow height of the container in the last grid of each column. By measuring the shadow height of the container in this last grid, the stacking height for all containers in that column can be inferred, thus determining the total number of stacked containers for the entire stack.

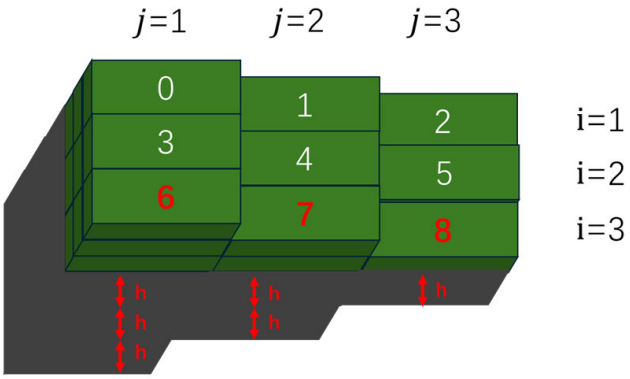

FIGURE 9. Ideal container stacking situation.

Taking the situation in Figure 9 as an example, the shadow height of the first column of containers is 3h, so the number of container stacks is 3. Similarly, the second column has 2 stacks and the third column has 1 stack. Therefore, the total number of containers in this stack is:

$$3 \times 3 + 3 \times 2 + 3 \times 1 = 18$$

b: DIFFERENT NUMBER OF STACKS PER COLUMN

However, in practical situations, the stacking height of each container column is not always the same, as shown in Figure 10.

To calculate the stacking numbers for each column of containers in this scenario, it is necessary to categorize the containers into three types: base-row container grids,

Algorithm 3 Calculate and Update Stack Counts for Containers Using Shadow Data

1. Import Required Libraries
Import libraries for data manipulation, numerical computations, and random operations.
2. Load Grid and Shadow Data
Load grid data from a CSV file into a DataFrame.
Load shadow data from a CSV file into another DataFrame.
3. Set Shadow Height Standard for a Single Container
Define the standard shadow height for a single container (in pixels).
Define the maximum range for detecting shadows.
4. Define Function to Calculate Shadow Pixel Count
Define a function to calculate the number of shadow pixels in a given grid:
Calculate the center x-coordinate of the grid.
Define a range of x-coordinates around the center.
Get the y-coordinate of the bottom of the grid.
Count the number of shadow pixels within the x-range and from the bottom y-coordinate to the maximum range.
Return the count of shadow pixels.
5. Define Function to Update Stack Counts
Define a function to update the stack counts in the grid DataFrame using shadow data:
Initialize a dictionary to store results for each area.
Loop through each unique area in the DataFrame:
Initialize the total count for the area.
Loop through each unique column index:
Find the bottom row grids in the current area and column.
If there are bottom row grids:
Initialize a list to store stack counts.
Loop through each bottom row grid:
Calculate the shadow count for the grid.
Calculate the number of stacks based on the shadow count and standard shadow height.
Add the stack count to the list.
Calculate the mean stack count from the list.
Find the non-bottom row grids in the same area and column.
Loop through each non-bottom row grid:
Update the stack count to the mean stack count.
Calculate the shadow count for the grid.
If the shadow count is greater than a certain threshold:
Randomly increase or decrease the stack count.
Calculate the total stack count for the column.
Add the column total to the area total.
Store the total count for the area in the results dictionary.
Return the results dictionary.
6. Calculate Estimated Container Counts for Each Area
Call the function to update stack counts and get the estimated container counts for each area.
7. Output the Results
Print the estimated container counts for each area.

non-base-row container grids, and empty grids. These are defined as follows:

Base-row container grids: These are grids in any given area where the container value is True, and which have the highest i value.

Non-base-row container grids: These are all other grids in the same area where the container value is True.

Empty grids: These are grids where there are no containers.

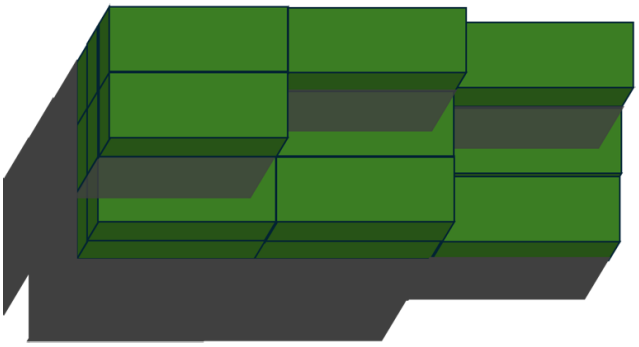

FIGURE 10. Different stacking quantities of containers in each column (an example).

Based on these definitions, in Figure 9, the base-row container grids are 6, 7, and 8, and the non-base-row container grids are 0, 1, 2, 3, 4, and 5. The shadow heights for the base-row container grids are 1h and 2h, and the other dimensional shadow height is 2d. The shadow height h for the non-base-row container grids is 1, as shown in Figure 11.

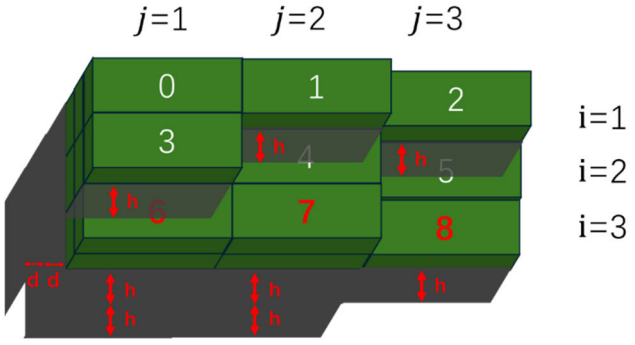

FIGURE 11. Calculation for different stacking quantities of containers in each column.

So, the stacking quantity of the first column's base-row container grid is 2, and the shadow height of the non-base-row container grid is h. Therefore, it can be inferred that the stacking quantity of the non-base-row container grid is 2+1=3, and the total number of containers in the first column is $2 \times 1+2 \times 3=8$. The second and third columns are also calculated using this method. Therefore, the total number of containers in this stack is:

$$\begin{cases} 2 \times 1 + 2 \times 3 = 8, & \mathrm{j} = 1 \\ 1 \times 1 + 2 \times 3 = 7, & \mathrm{j} = 2 \\ 1 \times 1 + 1 \times 3 = 4, & \mathrm{j} = 3 \end{cases}$$
$$8 + 7 + 4 = 19$$

c: STACKING WITH DENTS

In addition, there is a special case where there is a depression in the stacking of containers, as shown in Figure 12.

To calculate the number of stacks in the first column of containers under these conditions, following the definitions in the second scenario, in Figure 11, the base-row container grids are 6, 7, and 8, and the non-base-row container grids

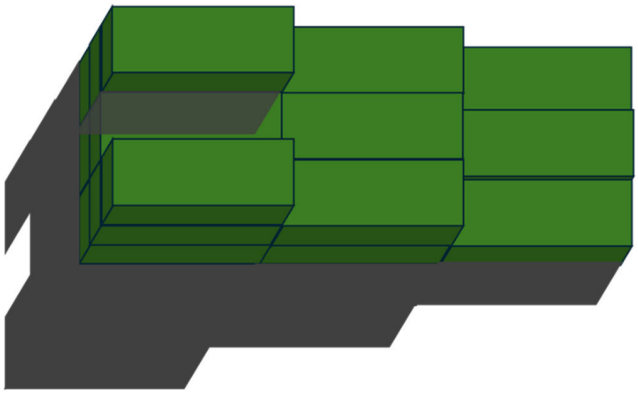

FIGURE 12. Dents in container stacking (an example).

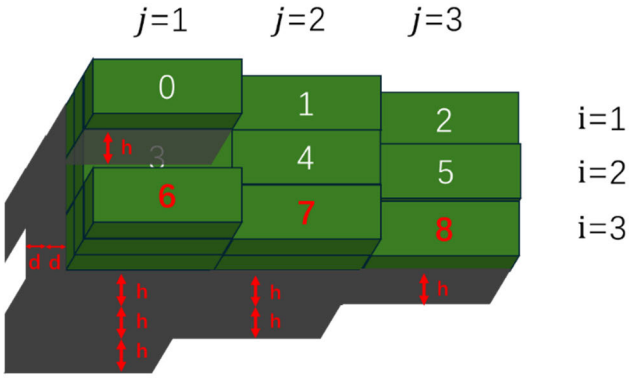

FIGURE 13. Calculation in case of dents in container stacking.

are 0, 1, 2, 3, 4, and 5. The shadow heights for the base-row container grids are 1h, 2h, and 3h, and the shadow height at the other dimensional indentations is 2d. The shadow height h for the non-base-row container grids is 1h, as illustrated in Figure 13.

Therefore, the number of stacks in the base-row container grids of the first column is 3, and the stacking height for the non-base-row container grid 3 is 3-1=2. From this, it is known that the total number of stacks in the first column of containers is $3\times3.1=8$. Hence, the total number of containers in this stack is:

$$\begin{cases} 3\times3-1\times1=8, & \mathrm{j}=1 \\ 2\times3=6, & \mathrm{j}=2 \\ 1\times3=3, & \mathrm{j}=3 \end{cases}$$
$$8+6+3=17$$

According to the above three situations, calculate the number of containers in each stack and sum them up to obtain the total number of containers in the entire yard.

IV. PRACTICAL APPLICATION OF THE METHOD

A. PRACTICAL APPLICATION CASE

This study employs satellite imagery due to its capability for sharing with third parties and its comparability across a wide range of ports and terminals. In contrast, other imaging methods, such as those from CCTV or drones, while offering more granular analysis, lack the same level of accessibility and standardization. After a thorough comparison of various satellite images, this study selected Google Earth Pro. The choice of Google Earth Pro was influenced by its superior spatial resolution. Specifically, Google Earth Pro provides a higher level of detail compared to high-resolution satellite images like those from the WorldView-3 satellite. The WorldView-3 offers a multispectral resolution with a pixel size of 1.6 meters and a panchromatic resolution with a pixel size of 0.4 meters. Google Earth images, however, surpass these specifications, ensuring more precise and detailed spatial analysis, which is crucial for the study's objectives.

TABLE 2. Coordinates of the four stacks and Yard 1.

Region Name	xmin	ymin	xmax	ymax
Yard 1	0	0	1610	915
Yard Stack 1	477	39	1310	223
Yard Stack 2	296	320	1309	438
Yard Stack 3	296	443	1311	637
Yard Stack 4	268	704	999	896

The study used a satellite image of Yard 1 at berth 604 of Cape Town port, taken on November 31, 2023. The original image is shown in Figure 14, with a resolution of 1610 pixels × 915 pixels.

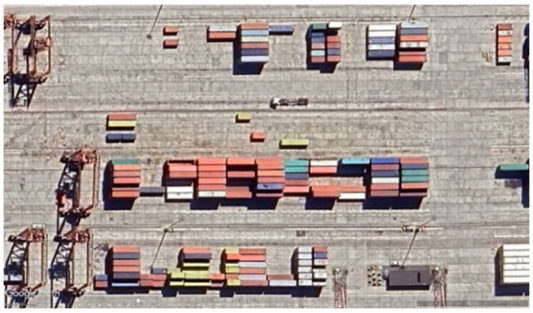

FIGURE 14. Yard 1 at berth 604 of Cape Town port.

Subsequently, apply the method in Section III to identify the number of containers.

1) STEP 1: YARD AND YARD STACK DIVISION

The container yard can be divided into four stacks, as shown in Figure 15. The coordinates of Yard 1 and the four stacks are shown in Table 2.

2) STEP 2: GRID DIVISION AND NUMBERING

Divide the 4 stacks into grids and save the relevant data. The grid division results are shown in Figure 16.

3) STEP 3: COLOR RECOGNITION TO IDENTIFY CONTAINERS

Apply the automated algorithm to identify the grid with containers and save the relevant data, as shown in Figure 17.

It should be noted that due to the complex realities on the ground, the identification of container grids is not

FIGURE 15. Sorting container yards and stacks.

FIGURE 16. Grid division and numbering.

FIGURE 17. Color recognition to identify containers.

absolutely accurate. As shown in Figure 18, two empty grids were mistakenly identified as containing containers because the ground color was darker. Therefore, it is necessary to manually check the identification results and eliminate any misidentified container grids.

4) STEP 4: ESTIMATING STACK NUMBERS THROUGH SHADOW ANALYSIS

The shadows of four container stacks were identified, and the relevant data were meticulously recorded to facilitate the calculation of stacking heights. This process ultimately enabled the estimation of the number of containers within the four stacks as well as across the entire yard. To determine the optimal HSV (Hue, Saturation, Value) thresholds, extensive and iterative experimentation was conducted, with the final thresholds detailed in Table 3. The results of the shadow identification process are illustrated in Figure 19, while Table 4 provides the estimated container counts for both the four individual stacks and the entire yard.

TABLE 3. Threshold HSV values.

H	S	V
180	255	100

This methodological approach ensures a comprehensive and accurate assessment of container quantities, leveraging advanced image processing techniques.

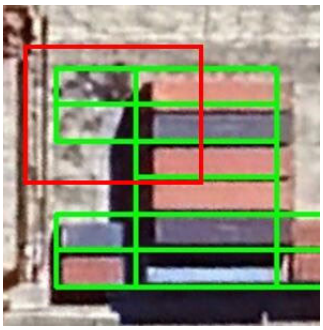

FIGURE 18. Misidentification occurred due to different ground colors.

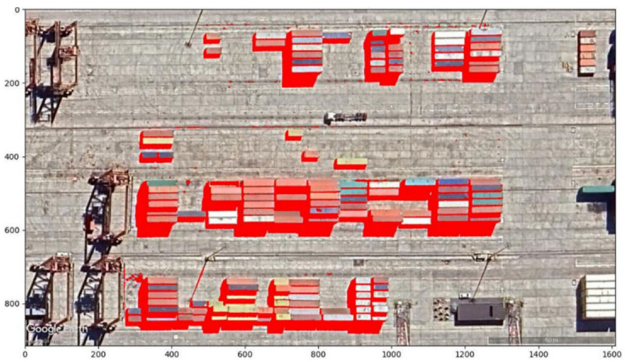

FIGURE 19. Shadow recognition results (The red part is the identified shadow).

The estimated number of containers in the entire container yard is 264.

B. EVALUATION OF THE METHOD

To validate the effectiveness of the proposed automated container counting method for estimating the number of containers, two other methods were selected for comparison: the U-Net model method based on semantic segmentation and the manual counting method.

1) U-NET MODEL METHOD BASED ON SEMANTIC SEGMENTATION

The U-Net model method based on semantic segmentation first utilizes publicly available satellite images covering relevant global ports to obtain RGB image data. These images are processed using the U-Net model, which is a convolutional neural network (CNN) specifically designed for semantic segmentation tasks. It features a unique U-shaped architecture with skip connections, enabling it

TABLE 4. Estimated number of containers in each region.

Region Name	Yard Stack 1	Yard Stack 2	Yard Stack 3	Yard Stack 4
Number of Containers	59	6	121	84

TABLE 5. Time for manual counting and counting using automated methods (2023).

Yard Name	Yard 1	Yard 2	Yard 3	Yard 4	Yard 5	Yard 6	Yard 7
UPRT(s)	1.02	2.97	2.46	3.45	1.65	1.04	1.32
APRT(s)	0.46	0.47	0.62	0.5	0.52	0.67	0.64

to effectively capture multi-scale contour information. The model is trained by manually labeling each pixel in the images as either "container" or "non-container." Once trained, the U-Net can classify each pixel in the image accordingly. The number of pixels classified as "container" in each image is then counted, and assuming that all stacked containers have the same number of layers, the container coverage area is calculated as a proxy for the number of containers in the port [3], as shown in Algorithm 4.

In this section, we evaluate the two methodologies from two critical perspectives: program runtime and estimation accuracy. For this evaluation, fourteen satellite images corresponding to all container berths at the port of Cape Town, captured in January 2023 and February 2024, were selected. These images are presented in Figure 20. Each of the fourteen images has a uniform resolution, measuring 1610 pixels × 915 pixels. This consistency in image resolution ensures a fair and accurate comparative analysis of the two methods under identical conditions, thereby providing reliable insights into their performance in real-world applications.

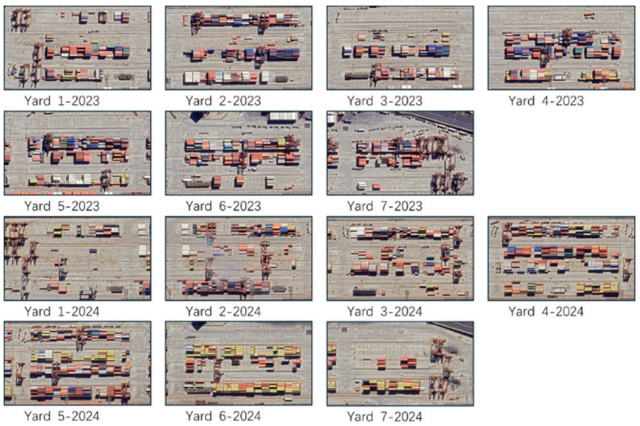

FIGURE 20. Photos of the seven container yards at the Port of Cape Town at certain times in 2023 and 2024.

Tables 5 and 6 show the program running time of the U-Net model method based on semantic segmentation, i.e. U-Net Program Running Time (UPRT) and the program running time of the automatic counting method, i.e. Automatic Program Running Time (APRT) after counting the containers on these 12 images, as well as the average program running time of the two methods.

Algorithm 4 Count Container Pixels Using Pre-Trained U-Net Model

1. Import Required Libraries

Import libraries for file operations, image processing, numerical computations, and TensorFlow for deep learning.

2. Load Pre-trained U-Net Model

Define a function to load a pre-trained U-Net model from the specified path:

Load and return the model.

3. Preprocess a Single Image

Define a function to preprocess a single image:

Read the image from the specified path.

Resize the image to the specified size.

Normalize the image values to be between 0 and 1.

Return the preprocessed image.

4. Count Container Pixels in Prediction Mask

Define a function to count container pixels in a prediction mask:

Sum the number of pixels in the mask that are greater than 0.5.

Return the count of container pixels.

5. Process All Images in a Folder

Define a function to process all images in a specified folder:

Initialize a dictionary to store container counts for each image.

Loop through each file in the folder:

If the file is an image (with specific extensions):

Get the full path of the image.

Preprocess the image.

Expand the image dimensions to include the batch size.

Use the model to predict the mask for the image.

Count the container pixels in the predicted mask.

Store the count in the dictionary with the filename as the key.

Return the dictionary containing container counts for each image.

6. Main Function

Define the main function:

Specify the paths to the model and the image folder.

Specify the image size used during training.

Load the U-Net model.

Process the images in the folder and get the container counts.

Loop through the results and print the filename and container pixel count for each image.

7. Run Main Function

Check if the script is being run as the main program:

Call the main function.

TABLE 6. Time for manual counting and counting using automated methods (2024).

Yard Name	Yard 1	Yard 2	Yard 3	Yard 4	Yard 5	Yard 6	Yard 7
UPRT(s)	1.26	2.17	2.99	3.13	2.12	2.63	1.25
APRT(s)	0.53	0.63	0.62	0.58	0.47	0.73	0.59

TABLE 7. Estimated and actual quantity of containers.

Yard Name	Q1	Q2	Actual Quantity
Yard 1-2023	112	204	283
Yard 2-2023	341	537	601
Yard 3-2023	143	276	297
Yard 4-2023	223	599	481
Yard 5-2023	199	501	443
Yard 6-2023	204	393	426
Yard 7-2023	82	292	195
Yard 1-2024	65	112	89
Yard 2-2024	189	685	451
Yard 3-2024	233	352	498
Yard 4-2024	298	667	684
Yard 5-2024	385	864	728
Yard 6-2024	157	432	425
Yard 7-2024	64	165	174

As demonstrated in Tables 5 and 6, the efficiency of the automated counting method has seen a substantial improvement compared to the U-Net model method based on semantic segmentation. Specifically, the average time consumption has been reduced by 72.90%.

In terms of estimation accuracy, the manually counted number of containers is considered the benchmark (Actual Quantity). This benchmark is then compared with the results obtained from the U-Net model method based on semantic segmentation (Q1) and the results from the automated counting method (Q2), as presented in Table 7. This comparative analysis underscores the advancements in both efficiency and accuracy achieved by the automated counting method, highlighting its potential for more effective application in operational environments.

Figure 21 compares the ratio of the actual value of the number of containers in each detection area to the estimated values of the two methods.

According to the prediction results, the MAE and RMSE of the counting results of the U-Net model method based on semantic segmentation are 220 and 239 respectively, and the MAE and RMSE of the automatic counting method are 74 and 98 respectively. Obviously, due to the resolution limitation of the satellite images used by the U-Net model method based on semantic segmentation, the number of stacked containers cannot be accurately identified in this method, so its estimated value is significantly smaller than the actual value, and the MAE and RMSE are larger. For

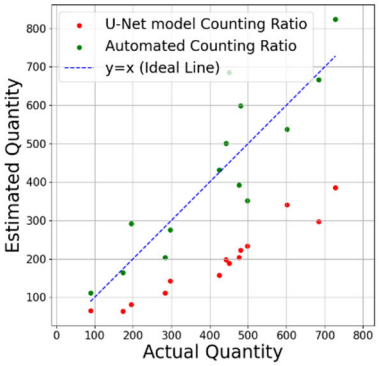

FIGURE 21. The actual value of the number of containers in each detection area to the estimated values of the two methods.

TABLE 8. Time for manual counting and counting using automated methods (2023).

Yard Name	Yard 1	Yard 2	Yard 3	Yard 4	Yard 5	Yard 6	Yard 7
MCT(s)	302	427	296	345	365	304	232
ACT(s)	84	136	82	110	102	97	64

TABLE 9. Time for manual counting and counting using automated methods (2024).

Yard Name	Yard 1	Yard 2	Yard 3	Yard 4	Yard 5	Yard 6	Yard 7
MCT(s)	186	401	447	563	612	413	125
ACT(s)	53	132	129	185	177	136	36

the estimation results of the automatic counting method, the MAE and RMSE are 74 and 98 respectively, which is acceptable compared to the total number of containers of 5629. Therefore, the automatic counting method is superior to the U-Net model method based on semantic segmentation in container quantity estimation.

2) MANUAL COUNTING METHOD

Manual counting is to count the containers in the pictures manually. Here, the manual counting method and the automatic counting method are evaluated from two perspectives: counting time and estimation accuracy.

Tables 8 and 9 show the time for manual counting of the containers in the 12 pictures and the time for the automatic counting method (steps 1 to 4) to count the containers in the 12 pictures, as well as the average time of these two items.

As illustrated in Tables 8 and 9, the automated method proposed in this paper has significantly enhanced efficiency compared to manual counting, achieving an average improvement of 69.8%.

Regarding the estimation accuracy of the automated method, the manually counted number of containers is regarded as the accurate benchmark. This benchmark is then compared with the count results obtained using the automated method, as detailed in Table 10. This comparison highlights

TABLE 10. Estimated and actual quantity of containers.

Yard Name	Estimated Quantity	Actual Quantity
Yard 1-2023	204	283
Yard 2-2023	537	601
Yard 3-2023	276	297
Yard 4-2023	599	481
Yard 5-2023	501	443
Yard 6-2023	393	426
Yard 7-2023	292	195
Yard 1-2024	112	89
Yard 2-2024	685	451
Yard 3-2024	352	498
Yard 4-2024	667	684
Yard 5-2024	864	728
Yard 6-2024	432	425
Yard 7-2024	165	174

the effectiveness of the automated method in delivering both higher efficiency and reliable accuracy in container counting.

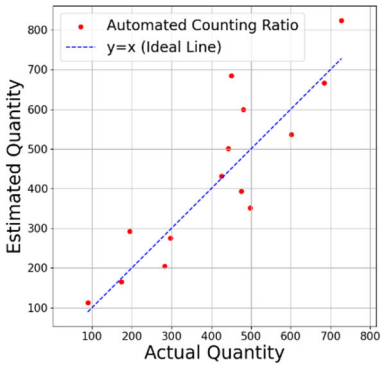

FIGURE 22. The actual value of the number of containers in each detection area to the estimated values of the automatic counting method.

FIGURE 23. Image with significant estimation error in container quantity.

According to the prediction results, the MAE and the RMSE are 39 and 59, respectively, which are considered acceptable given the total number of 5825 containers. During the prediction process, many counts were overestimated, with some samples having their container counts overestimated by more than 25%, just as shown in Figure 22. In fact, cases where the prediction significantly deviated from the actual numbers generally occurred when there was substantial interference above the stacks, such as clouds or yard cranes, as shown in Figure 23.

V. CONCLUSION

The method developed in this study provides a valuable automated tool for estimating the number of containers in port yards using satellite imagery, offering high accuracy and efficiency. By leveraging shadow analysis and grid-based techniques, it overcomes the limitations of traditional counting methods and does not require large datasets or physical access to ports. This makes it particularly useful for third-party researchers or entities that do not have direct operational control over port data. While this study aims to provide a benchmark when high-resolution satellite imagery is abundant, the Google Earth imagery used in this study is not updated in real time and is not suitable for monitoring daily status changes. However, the availability of satellite constellations with higher resolution and higher revisit frequency may make this vision a reality; if images with spatial resolution close to that of Google Earth imagery are frequently available, it will be possible for third parties to achieve real-time monitoring of container terminals and measure operational efficiency. In addition, it can also be used to analyze container cargo flows and congestion on land, such as trucks, railways, inland warehouses, etc., where third parties do not have access to comprehensive information on real-time transportation modes such as AIS.

The actual application of the method to the container yard of the Port of Cape Town shows that its effectiveness has been proven in practice. Despite some challenges, such as interference from environmental factors such as clouds or other obstacles, the method can be considered effective. Compared with the U-Net model method based on semantic segmentation, the efficiency of this method has been significantly improved, the average program running time has been shortened by 72.90%, and the MAE and the RMSE of the prediction have been greatly reduced; compared with the manual counting method, the counting efficiency has been increased by an average of 69.8%, and the MAE and the RMSE of the prediction are 39 and 59 respectively, which is acceptable for estimating the number of thousands of containers in the entire container port. For future research, it will focus on non-robust shadow recognition, with the goal of further improving the calculation algorithm to minimize the interference of clouds and other obstacles, thereby further improving the accuracy, and exploring applications in other logistics fields besides port management.

ACKNOWLEDGMENT

The authors would like to thank their teachers, Dr. Yuhao Zhang, Dr. Ren Hu, and Ran Yang for their invaluable guidance, continuous support, and encouragement throughout their research and the writing of this thesis. Without their expert advice and unwavering support, this work would not have been possible.

Special thanks are extended to Prof. Ryuichi Shibasaki from The University of Tokyo. During their research, the authors had the opportunity to inquire about certain aspects of Prof. Shibasaki's research findings. He responded promptly and patiently addressed all questions, providing invaluable insights and guidance that were crucial to the successful completion of this thesis.

REFERENCES

[1] K. Hamamoto, A. Kuze, T. Tadono, S. Sobue, J. Ishizawa, K. Ohyoshi, H. Murakami, K. Kawamura, and Y. Ikehata, "Jaxa's earth observation data analysis on COVID-19," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Brussels, Belgium, Jul. 2021, pp. 1362–1365, doi: 10.1109/IGARSS47720.2021.9554593.

[2] (2024). *On Change of Port and Harbors After COVID-19 Pandemic*. Jpn. Aerosp. Explor. Agency. Accessed: Jul. 10, 2024. [Online]. Available: https://earth.jaxa.jp/covid19/industry/en.html

[3] H. Yu, X. Hao, L. Wu, Y. Zhao, and Y. Wang, "Eye in outer space: Satellite imageries of container ports can predict world stock returns," *Humanities Social Sci. Commun.*, vol. 10, no. 1, pp. 1–6, Jul. 2023, doi: 10.1057/s41599-023-01891-9.

[4] K. Yasuda, R. Shibasaki, R. Yasuda, and H. Murata, "Terminal congestion analysis of container ports using satellite images and AIS," *Remote Sens.*, vol. 16, no. 6, p. 1082, Mar. 2024.

[5] C. Wang, J. Pei, R. Wang, Y. Huang, and J. Yang, "A new ship detection and classification method of spaceborne SAR images under complex scene," in *Proc. 6th Asia–Pacific Conf. Synth. Aperture Radar (APSAR)*, Nov. 2019, pp. 1–4, doi: 10.1109/APSAR46974.2019.9048382.

[6] Y. Liu, H. Cui, and G. Li, "A novel method for ship detection and classification on remote sensing images," in *Proc. Artif. Neural Netw. Mach. Learn. (ICANN)*, 2017, pp. 556–564, doi: 10.1007/978-3-319-68612-7_63.

[7] I. Zakharov, D. A. Lavigne, S. Warren, M. D. Henschel, D. Power, and M. Howell, "Ship detection and classification in EO/IR VHR satellite imagery," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2021, pp. 3561–3564, doi: 10.1109/IGARSS47720.2021.9553254.

[8] B. Li, X. Xie, X. Wei, and W. Tang, "Ship detection and classification from optical remote sensing images: A survey," *Chin. J. Aeronaut.*, vol. 34, no. 3, pp. 145–163, Mar. 2021, doi: 10.1016/j.cja.2020.09.022.

[9] S. A. Hassan, "Port activity simulation: An overview," in *ACM SIGSIM Simulation Dig.*, 1993, pp. 17–36, doi: 10.1145/174253.174255.

[10] M. Bonilla, A. Medal, T. Casaus, and R. Sala, "The traffic in Spanish ports: An efficiency analysis," *Int. J. Transp. Econ./Rivista Internazionale di Economia dei Trasporti*, vol. 29, no. 2, pp. 215–230, Jun. 2002.

[11] M. Kondratyev, "An object-oriented approach to port activity simulation," *Int. J. Simul. Process. Model.*, vol. 10, no. 1, pp. 1–9, 2015, doi: 10.1504/ijspm.2015.068511.

[12] A. Worth, A. Televantos, N. Evmides, M. Michaelides, and H. Herodotou, "Online analytical processing of port calls for decision support," in *Proc. 23rd IEEE Int. Conf. Mobile Data Manag. (MDM)*, Jun. 2022, pp. 437–439, doi: 10.1109/MDM55031.2022.00095.

[13] R. A. Sutrisnowati, H. Bae, and M. Song, "Bayesian network construction from event log for lateness analysis in port logistics," *Comput. Ind. Eng.*, vol. 89, pp. 53–66, Nov. 2015, doi: 10.1016/j.cie.2014.11.003.

[14] S. Li, N. Liu, F. Li, J. Gao, and J. Ding, "Automatic fault delineation in 3-D seismic images with deep learning: Data augmentation or ensemble learning?" *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022, Art. no. 5911214, doi: 10.1109/TGRS.2022.3150353.

[15] C. Alexakos and P. Konstantinopoulos, "Integration middleware for collection and monitoring environmental parameters in ports," in *Proc. 16th Panhellenic Conf. Informat.*, Oct. 2012, pp. 261–265, doi: 10.1109/PCi.2012.52.

[16] X. Jinguang, Y. Xinping, T. Fei, and M. Langqi, "Green port environment monitoring system with early warning and linkage control," in *Proc. 12th IEEE Int. Conf. Electron. Meas. Instrum. (ICEMI)*, vol. 1, Jul. 2015, pp. 244–247, doi: 10.1109/ICEMI.2015.7494261.

[17] T. Alam, A. Campaneria, M. Silva, L. Bobadilla, and G. A. Weaver, "Coastal infrastructure monitoring through heterogeneous autonomous vehicles," in *Proc. 4th IEEE Int. Conf. Robotic Comput. (IRC)*, Nov. 2020, pp. 79–82, doi: 10.1109/IRC.2020.00019.

[18] M. H. Shahraji and C. Larouche, "Case study: Rigorous boresight alignment of a marine mobile LiDAR system addressing the specific demands of port infrastructure monitoring," *Mar. Geodesy*, vol. 45, no. 3, pp. 295–327, May 2022, doi: 10.1080/01490419.2022.2025503.

[19] P. Pandey, "Managing noise from port operations via the use of automated noise monitoring systems," *J. Acoust. Soc. Amer.*, vol. 154, no. 4, p. A69, Oct. 2023, doi: 10.1121/10.0022827.

[20] C. F. Wooldridge, C. Mcmullen, and V. Howe, "Environmental management of ports and harbours—Implementation of policy through scientific monitoring," *Mar. Policy*, vol. 23, nos. 4–5, pp. 413–425, Jul. 1999, doi: 10.1016/S0308-597X(98)00055-4.

[21] N. Liu, Z. Li, R. Liu, H. Zhang, J. Gao, T. Wei, J. Si, and H. Wu, "ASHFormer: Axial and sliding window-based attention with high-resolution transformer for automatic stratigraphic correlation," *IEEE Trans. Geosci. Remote Sens.*, vol. 61, 2023, Art. no. 5913910, doi: 10.1109/TGRS.2023.3296934.

[22] M. Feng, S.-L. Shaw, G. Peng, and Z. Fang, "Time efficiency assessment of ship movements in maritime ports: A case study of two ports based on AIS data," *J. Transp. Geography*, vol. 86, Jun. 2020, Art. no. 102741, doi: 10.1016/j.jtrangeo.2020.102741.

[23] S.-H. Chen and J.-N. Chen, "Forecasting container throughputs at ports using genetic programming," *Expert Syst. Appl.*, vol. 37, no. 3, pp. 2054–2058, Mar. 2010, doi: 10.1016/j.eswa.2009.06.054.

[24] E. Lee, D. Kim, and H. Bae, "Container volume prediction using time-series decomposition with a long short-term memory models," *Appl. Sci.*, vol. 11, no. 19, p. 8995, Sep. 2021, doi: 10.3390/app11198995.

[25] J. Liu, "A grey theory-based economic prediction method of port and ocean engineering," *J. Coastal Res.*, vol. 106, pp. 197–200, Jul. 2020, doi: 10.2112/si106-046.1.

[26] L. Brooks, N. Gendron-Carrier, and G. Rua, *The Local Impact of Containerization*. San Mateo, CA, USA: PSN, Domestic Development Strategies, 2018, doi: 10.17016/FEDS.2018.045.

[27] M. R. Nieto, R. B. C. Benitez, and J. N. Martinez, "Comparing models to forecast cargo volume at port terminals," *J. Appl. Res. Technol.*, vol. 19, no. 3, pp. 238–249, Jun. 2021, doi: 10.22201/icat.24486736e.2021.19.3.1695.

[28] E. Twrdy and M. Batista, "Modeling of container throughput in northern Adriatic ports over the period 1990–2013," *J. Transp. Geography*, vol. 52, pp. 131–142, Apr. 2016, doi: 10.1016/j.jtrangeo.2016.03.005.

BOYANG ZHOU was born in Suzhou, China, in 2002. He is currently pursuing the bachelor's degree in management with Jiangsu University of Science and Technology. His research interests include data mining and analysis and port operations.

ZHENQUAN WANG was born in Suzhou, Jiangsu, China, in 2003. He is currently pursuing the bachelor's degree in finance with Xiamen University, in 2024.