Efficient Edge AI based Annotation and Detection Framework for Logistics Warehouses

Yuki Mori
Nagoya University, Japan
ymori@ucl.nuee.nagoya-u.ac.jp

Yusuke Asai
Nagoya University, Japan
asayu@ucl.nuee.nagoya-u.ac.jp

Kesuke Higashiura
Nagoya University, Japan
urachan@ucl.nuee.nagoya-u.ac.jp

Shin Katayama
Nagoya University, Japan
shinsan@ucl.nuee.nagoya-u.ac.jp

Kenta Urano
Nagoya University, Japan
vrano@ucl.nuee.nagoya-u.ac.jp

Takuro Yonezawa
Nagoya University, Japan
takuro@nagoya-u.jp

Nobuo Kawaguchi
Nagoya University, Japan
kawaguti@nagoya-u.jp

***Abstract*—As global logistics demand increases, improving the efficiency of warehouse operations has become critical. To achieve this, it's crucial to identify inefficient tasks and layouts by recognizing various warehouse conditions. To address these challenges, we've constructed a large-scale camera infrastructure to convert object positions and movements into data. However, transmitting all video data to the cloud results in significant data transmission and power consumption. Edge AI cameras analyze and extract video data locally, transmitting only essential information and significantly reducing them. Edge AI cameras require a low-computation, high-accuracy object detection model due to limited computational power and complex warehouse environments. Furthermore, the same object appears differently based on the camera's position and angle. Therefore, customizing models for each camera improves accuracy, but the annotation cost would be very high. In this study, we propose a method to perform part of the training data generation on the camera, reducing data transmission and improving annotation efficiency.**

***Index Terms*—edge, annotation, AI, warehouse, labeling**

I. Introduction

As electronic commerce (EC) grows, logistics warehouse spaces are expanding. As a result, workloads are increasing, making it crucial to improve operational efficiency. Among the various approaches being explored to enhance this efficiency, a digital twin, a virtual model that simulates real-world environments for cost-effective improvements, has attracted much attention [1]. However, constructing a digital twin requires precise digitization of physical spaces through sensing and data extraction.

To track the positions and movements of objects, installing cameras is an effective method. We have installed over 60 cameras in the warehouse, collecting about 1.2 TB of data daily. As warehouse space continues to expand, reducing transmission volume and power consumption becomes an important issue. Edge processing has been attracting attention as a method that addresses this issue [2], [3]. Edge AI cameras analyze and extract data from captured video, and then transmit only the necessary information. However, they face constraints such as limited computational capacity. Moreover, given the complex environment of a warehouse and a variability in object appearances due to camera angles, a highly accurate yet low-computation object detection model is essential. To address this, we focus on developing tailored models for each camera by preparing camera-specific datasets, although this approach increases data collection and annotation costs.

In this study, we propose a method to perform part of training data generation on the camera to reduce data transmission and storage usage and improve annotation efficiency by automating segmentation and batch labeling.

The contributions of this study are as follows:

- Proposing an efficient annotation and detection framework tailored to each Edge AI camera in the warehouse.
- Achieving over 93% reduction in data transmission and storage usage during the annotation process.
- Training object detection models for edge AI cameras, resulting in over 92% efficiency gains in annotation.

II. Relatedwork

We have been working on improving annotation efficiency for constructing a digital twin of a logistics warehouse. Higashiura and Kano et al. [4], [5] proposed a framework that can significantly reduce annotation time in the warehouse environment. However, it targets only dynamic objects like people and is unsuitable for static ones like pallets. Furthermore, the method is not designed for edge AI cameras, which is the goal of this study, and does not take into account the amount of transmission and power consumption.

Object detection and tracking algorithms using edge AI cameras have been studied [6]–[8], primarily targeting objects in everyday environments where open-source datasets are sufficient. In contrast, a warehouse environment, which contains a wide variety of objects, requires custom datasets tailored to its complexity. Moreover, when using multiple edge AI cameras, creating individual training datasets for each camera lead to achieve high accuracy but involves significant labor and cost. To the best of our knowledge, there has been no research on how to create training datasets for such edge AI cameras.

III. Environment

We conduct research focusing on the logistics warehouse located in Aichi, Japan. We have constructed a large-scale camera infrastructure with over 60 cameras, as shown in Fig.1.

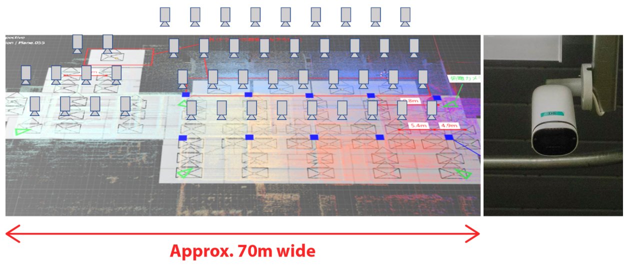


Fig. 1: Large-scale camera infrastructure [3]

The camera is HV-800G2A5 manufactured by H.View. There are two types of cameras: one captures the floor surface from directly above, and the other provides a bird's-eye view. They are installed in various locations, such as truck berth and elevators. This study focuses on the first type. The video is Full HD at 5 fps. OAK-D-Pro W PoE from Luxonis was used as an edge AI camera. It is equipped with Intel's Movidius Myriad VPU, which enables it to run neural networks.

IV. Method

The framework for creating training datasets proposed in this study is shown in Fig.2. The framework consists of four main processes: moving object detection, segmentation, clustering, and labeling.

A. Moving Object Detection

Dynamic and static objects are detected in the input video. Since this processing is performed by the edge AI camera, this detection is performed using sparse optical flows that are computationally less demanding, as shown in Fig.3a. Initially, we employ Shi-Tomasi corner detection [9] to identify feature points in the frames. Subsequently, using optical flow [10], the points are classified as dynamic objects if they move more than 15 pixels between two frames. On the other hand, static objects are identified by points that were previously dynamic but have stopped moving. They are also identified by points in the storage area, which includes spaces for storing and inspecting goods but excludes floor pathways, that are not classified as dynamic.

B. Segmentation

Feature points recognized as dynamic or static objects are segmented using NanoSam [11]. NanoSam, a streamlined version of the Segment Anything Model (SAM) [12], operates in real-time and performs computations directly on the edge AI camera. To prepare inputs for the NanoSam model, we set the frame resolution to 1024x1024, adding padding equally to the top and bottom to preserve the original aspect ratio. Segmentation is then applied to the recognized feature points. Fig.3b shows an example of the output. The contours are then extracted from the segmentation data using OpenCV's findContours function. To reduce the amount of transmission and storage, we apply several constraints. Segmentation results with an area less than 0.5% of the entire image are removed as noise. Additionally, objects with an aspect ratio larger than 4:1 are removed because they are too long and narrow. When multiple feature points on an object are recognized as dynamic or static, several segmentation results are produced. If these segmentation results overlap by more than 30%, they are merged into a single result. When a feature point recognized as a static object is segmented in every frame, the static object and the feature point remain unchanged, causing the same data to be sent multiple times. To avoid this, feature points identified as static objects are stored in a list, and segmentation is only performed when the contents of the list change. When the data is transferred to the server, information about the static object is added, and only the segmented data is used in the subsequent clustering and labeling processes. Finally, we add the labeled static object data to all appropriate frame data before training an object detection model. With the above approach, only the necessary data is extracted and transferred to the server, reducing the amount of transmission and storage.

C. Clustering

We adopt the most accurate combination of the methods proposed in our previous research [4]. Specifically, we use SimSiam [13] to encode features of object images obtained through segmentation. Subsequently, dimensionality reduction is performed using UMAP [14], followed by the clustering of similar objects into the same clusters using K-means.

D. Labeling

We use a custom labeling tool that displays clusters in order and assign appropriate labels to each, as shown in Fig.4. It is also possible to label each image individually. Basically, each cluster is labeled as a whole, and if an image is clearly inappropriate, a suitable label can be assigned.

V. Evaluation

To validate the effectiveness of our method, an evaluation experiment was conducted.

A. Method for Comparison

The comparison method in this study is manual annotation, which entails surrounding each object with a bbox and assigning a label. Although various models and techniques for annotation have been proposed, constructing a tool that can serve as a baseline for comparison remains challenging. Therefore, we use manual annotation for comparison.

B. Evaluation Metrics

Two evaluation metrics are defined to evaluate our method: annotation efficiency and object detection accuracy. Annotation efficiency is measured by the number of annotations performed per minute. For our approach, we record the time required for labeling, whereas manual annotation measures the time needed to enclose the target object with a bbox and label it. Object detection accuracy measures the quality of the annotations. In general, the higher the quality of the training data, the higher the accuracy of object detection models. Therefore, in this paper, we train an object detection model using the dataset annotated with our method and manual annotation method and use the detection accuracy of the object

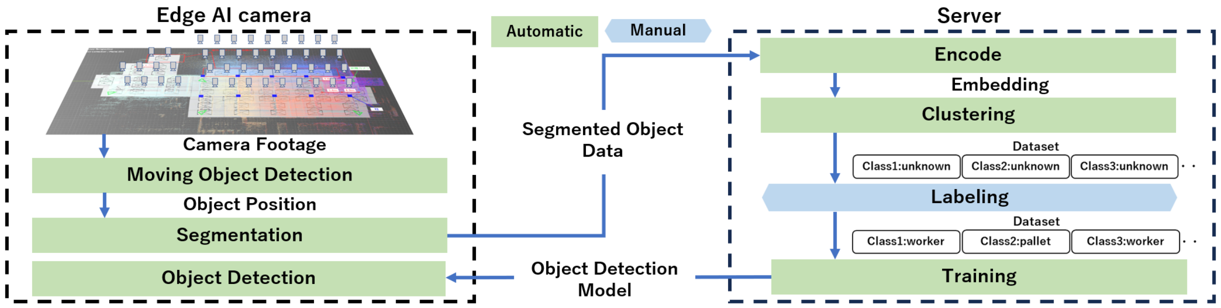
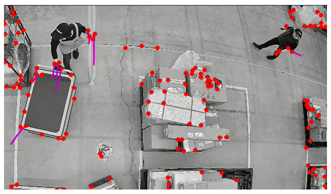
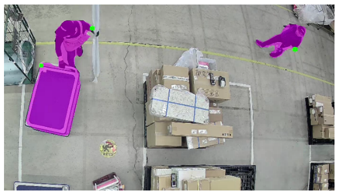


Fig. 2: The framework of this study



(a) Sparse optical flow



(b) Segmentation

Fig. 3: Example output

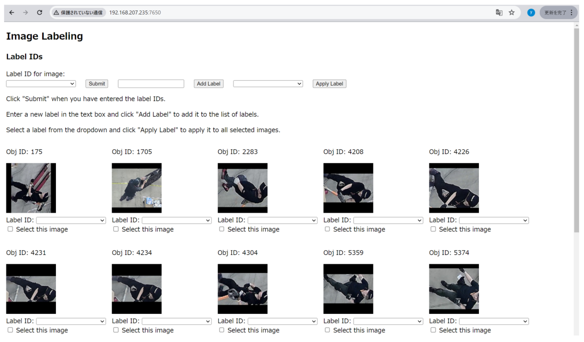


Image Labeling

Label IDs

Label ID for image:

Submit | Add Label | Apply Label

Click "Submit" when you have entered the label IDs.

Enter a new label in the text box and click "Add Label" to add it to the list of labels.

Select a label from the dropdown and click "Apply Label" to apply it to all selected images.

Obj ID: 175 — Label ID: — Select this image
Obj ID: 1705 — Label ID: — Select this image
Obj ID: 2283 — Label ID: — Select this image
Obj ID: 4208 — Label ID: — Select this image
Obj ID: 4226 — Label ID: — Select this image
Obj ID: 4231 — Label ID: — Select this image
Obj ID: 4234 — Label ID: — Select this image
Obj ID: 4304 — Label ID: — Select this image
Obj ID: 5359 — Label ID: — Select this image
Obj ID: 5374 — Label ID: — Select this image

Fig. 4: Custom labeling tool

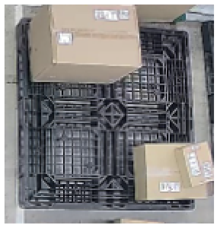
detection models as the annotation accuracy. We use Average Precision (AP) as the metric for object detection accuracy. AP is evaluated on a scale from 0 to 100 as a percentage, with higher AP indicating better accuracy. IoU (Intersection over Union) indicates the degree of overlap between the predicted bbox and the bbox of the correct data, and AP50 is the AP value when IoU is 0.5 or higher.

C. Detail of Experiment

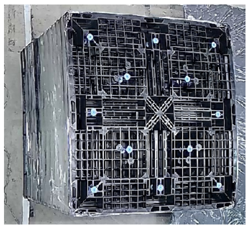
1) Data used: We used existing video data recorded by one camera over 6 days. These videos were saved as separate files every 30 minutes. We reviewed all the video files and selected 22 hours of footage featuring frequent occurrences of pallets (few items) and pallets (stacked) as shown in Fig.5.

2) Dataset Creation for Our Method: The evaluation experiment followed the process shown in Fig.2. Ideally, the experiment should be conducted using real-time camera footage, but to reduce setup costs, we sent pre-recorded videos to the AI processor of the edge AI camera for evaluation.

First, sparse optical flow was used to detect the positions of objects in the video, with a maximum of 480 feature points



(a) pallet (few items)



(b) pallet (stacked)

Fig. 5: Evaluation objects in this study

employed. Next, segmentation masks were generated for these feature points. Segmentation was only performed every 5 frames, as consecutive frames are similar and have minimal impact on training outcomes. The contours were then extracted from the generated mask images. Based on the contour information, each instance was separated, and 2048 dimensional image embeddings were generated. The hyperparameters and training model for SimSiam were implemented based on the official releases. The 2048 dimensional embeddings were then compressed to 512 dimensions. These compressed embeddings were clustered, classifying the data collected from a single video file into 100 clusters. Finally, labeling was performed. To calculate the annotation time, a stopwatch was used to measure the time taken for labeling. Ultimately, 29,006 annotations were made across 21 categories. Of these, 2,437 annotations were labeled as pallet (few items) and 400 as pallet (stacked).

3) Dataset Creation for Manual Annotation: We manually enclosed each pallet (few items) and pallet (stacked) in the warehouse videos with a bbox and labeled them. These manual annotations were performed by the author and an annotator who regularly engages in such tasks. As with our method, annotations were performed on 2,437 pallets (few items) and 400 pallets (stacked). To calculate the annotation time, a stopwatch was used to measure the time taken for annotations.

4) Creation of the Evaluation Dataset: To assess object detection accuracy, we created an evaluation dataset using manual annotation from 4 days of video data, distinct from those used for the training dataset. From these video data, we annotated 344 pallets (few items) and 74 pallets (stacked).

5) Training Object Detection Model and Calculating AP: Object detection models were trained using the datasets created in SectionV-C2 and V-C3. Using the evaluation dataset

TABLE I: Annotation Efficiency Results

Method	number of annotations per minute
Our Method	93.28
Manual Annotation	5.21

TABLE II: Accuracy Results of the Object Detection Model

Dataset	Average Precision (%)
Our Method	65.1
Manual Annotation	78.4

created in V-C4, we calculated AP50 and evaluated the accuracy of the datasets created by our method and manual annotation. In this experiment, we used YOLOv8n [15], trained for 500 epochs with a 640×640 input size and default settings, as the object detection model.

D. Results and Discussion

Table.I presents annotations per minute and Table.II shows AP50 for both our method and manual annotation.

Our method was slightly less accurate than manual annotation. The main reason for this is that when objects such as cardboard overlap, segmentation is not successful, and poor quality annotations are mixed in. For example, as shown in Fig.6, the right pallet is affected by cardboard, resulting in poorer annotation accuracy compared to manual annotation. To improve segmentation, the method combined with background subtraction can be considered. Additionally, the method using sparse optical flow might have missed detecting objects that should have been identified, and improperly clustered objects could lead to incorrect labeling.

The annotation efficiency improved by more than 92%, significantly reducing costs. This improvement is primarily due to the automated segmentation and batch labeling.

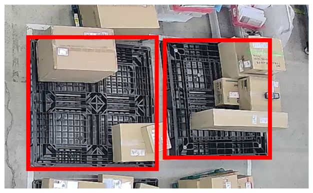
Our method, designed for edge AI cameras with limited computational capacity, performed moving object detection and segmentation, transmitting only essential data. In this evaluation experiment, out of the total 392,250 frames, only 24,649 were extracted by the edge AI camera and used on the server, reducing data transmission volume by more than 93%. It is possible to reduce this further during periods with less human activity or depending on the camera's viewpoint.

VI. Summary

In this paper, we proposed an efficient annotation method for object detection modeling in edge AI architecture. Our method automates segmentation, labels multiple data sets together, and improves annotation efficiency in a warehouse environment. In addition, our method segments objects within the edge AI camera, transmitting only the necessary data to the server. This approach reduces data transmission and storage requirements during the annotation process. As a result, our method improved annotation efficiency by over 92%, despite slightly lower accuracy compared to manual annotation.

Acknowledgment

This work is partially supported by JSPS KAKENHI (JP22K18422), NEDO (JPNP23003), NICT (222C01, 22609), and CSTI SIP3 (JPJ012495). We also thank TRUSCO Nakayama Corporation for providing the experimental environment.


(a) Manual Annotation

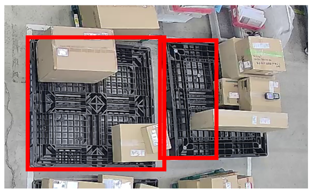

(b) Our Method

Fig. 6: Difference in annotation quality

References

[1] B. R. Barricelli, E. Casiraghi, and D. Fogli, "A survey on digital twin: Definitions, characteristics, applications, and design implications," *IEEE Access*, vol. 7, pp. 167 653–167 671, 2019.

[2] H. Hua, Y. Li, T. Wang, N. Dong, W. Li, and J. Cao, "Edge computing with artificial intelligence: A machine learning perspective," *ACM Computing Surveys*, vol. 55, no. 9, pp. 1–35, 2023.

[3] Y. Asai, Y. Mori, K. Higashiura, K. Yokoyama, S. Katayama, K. Urano, T. Yonezawa, and N. Kawaguchi, "Towards a real-time and energy-efficient edge ai camera architecture in mega warehouse environment," in *2024 IEEE 3rd Real-Time and Intelligent Edge Computing Workshop (RAGE)*, 2024, pp. 1–6.

[4] K. Higashiura, K. Yokoyama, Y. Asai, H. Shimosato, K. Kano, S. Katayama, K. Urano, T. Yonezawa, and N. Kawaguchi, "Semi-automated framework for digitalizing multi-product warehouses with large scale camera arrays," in *2024 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 2024, pp. 98–105.

[5] K. Kano, Y. Mori, K. Higashiura, T. Hossain, S. Katayama, K. Urano, T. Yonezawa, and N. Kawaguchi, "Composite image generation using labeled segments for pattern-rich dataset without unannotated target," in *Companion of the 2024 on ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2024, pp. 507–512.

[6] H. F. Yang, J. Cai, C. Liu, R. Ke, and Y. Wang, "Cooperative multi-camera vehicle tracking and traffic surveillance with edge artificial intelligence and representation learning," *Transportation research part C: emerging technologies*, vol. 148, p. 103982, 2023.

[7] M. I. Zaman, U. I. Bajwa, G. Saleem, and R. H. Raza, "A robust deep networks based multi-object multi-camera tracking system for city scale traffic," *Multimedia Tools and Applications*, vol. 83, no. 6, pp. 17 163–17 181, 2024.

[8] V. Mazzia, A. Khaliq, F. Salvetti, and M. Chiaberge, "Real-time apple detection system using embedded systems with hardware accelerators: An edge ai application," *IEEE Access*, vol. 8, pp. 9102–9114, 2020.

[9] J. Shi *et al.*, "Good features to track," in *1994 Proceedings of IEEE conference on computer vision and pattern recognition*, 1994, pp. 593–600.

[10] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," vol. 2, pp. 674–679, 1981.

[11] NanoSam : accessed 2024-02-22, https://github.com/NVIDIA-AI-IOT/nanosam.

[12] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo *et al.*, "Segment anything," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 4015–4026.

[13] X. Chen and K. He, "Exploring simple siamese representation learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 15 750–15 758.

[14] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426*, 2018.

[15] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.