applied sciences

MDPI

Article

Smart Logistics Warehouse Moving-Object Tracking Based on YOLOv5 and DeepSORT

Tingbo Xie and Xifan Yao *

School of Mechanical and Automotive Engineering, South China University of Technology, Guangzhou 510640, China; 202221002816@mail.scut.edu.cn
* Correspondence: mexfyao@scut.edu.cn; Tel.: +86-20-8711-2381

Featured Application: An approach for object tracking in logistics warehouses based on YOLOv5 and DeepSORT is proposed, which distinguishes humans from goods, and an evaluation system is established for object tracking in logistics warehouse scenarios.

Abstract: The future development of Industry 4.0 places paramount importance on human-centered/-centric factors in the production, design, and management of logistic systems, which has led to the emergence of Industry 5.0. However, effectively integrating human-centered/-centric factors in logistics scenarios has become a challenge. A pivotal technological solution for dealing with such a challenge is to distinguish and track moving objects such as humans and goods. Therefore, an algorithm model combining YOLOv5 and DeepSORT for logistics warehouse object tracking is designed, where YOLOv5 is selected as the object-detection algorithm and DeepSORT distinguishes humans from goods and environments. The evaluation metrics from the MOT Challenge affirm the algorithm's robustness and efficacy. Through rigorous experimental tests, the combined algorithm demonstrates rapid convergence (within 30 ms), which holds promising potential for applications in real-world logistics warehouses.

Keywords: deep learning; YOLOv5; DeepSORT; logistics warehouse

Citation: Xie, T.; Yao, X. Smart Logistics Warehouse Moving-Object Tracking Based on YOLOv5 and DeepSORT. *Appl. Sci.* **2023**, *13*, 9895. https://doi.org/10.3390/app13179895

Academic Editor: Jose Machado

Received: 29 July 2023
Revised: 18 August 2023
Accepted: 29 August 2023
Published: 1 September 2023

1. Introduction

With the emergence of smart manufacturing in conjunction with Industry 4.0, the demand for intelligent warehouses and intelligent logistics systems has been increasing. A well-functioning logistics warehouse system often enhances both the work efficiency of workers and the operational efficiency of goods [1]. The development of Industry 4.0 has prompted various applications of artificial intelligence and automation technology in intelligent logistics warehouses. However, it comes with the oversight of human-centered aspects in intelligent warehouse systems [2]. It has become a major challenge to comprehensively incorporate human factors into intelligent logistics warehouses.

Furthermore, the development of mobile networks has led to a boom in online shopping transactions, and subsequently, the logistics industry [3]. Therefore, logistics warehouses have to manage goods in a more expedient and efficient way. Within the realm of logistics warehouse management, managers mostly use surveillance cameras to manage the arrangement of goods and personnel movements. Nevertheless, with the increase in the number of surveillance cameras and the associated manual time expenses, the traditional manual handling of video surveillance has gradually been phased out. A more intelligent and automated video processing system is needed to oversee the movement of individuals and goods [4,5].

Object-tracking technology has found widespread utility within logistics warehouses. In the practical application of monitoring logistics warehouses, there is an increasing need for object-tracking algorithms that deliver real-time performance and robustness due to

changes in the environment and the tracked entities themselves. Given the substantial volume of goods and individuals requiring tracking within logistics warehouses, some of the targets may move randomly, making it difficult for traditional machine vision algorithms to identify targets accurately. The newly developed object-recognition algorithms, grounded in deep learning, are able to automatically learn and extract features. Moreover, they can be adapted to complex scenarios, rendering them particularly well-suited for intelligent monitoring applications in logistics warehouses [6,7].

This study aims to provide an effective object-tracking approach within smart logistics warehouses. Additionally, it strives to formulate an object-tracking system tailored to logistics scenarios. The former provides auxiliary means for monitoring the operation status of logistics warehouses, ensuring smooth functioning and safety management. This approach aligns with the notion of the human-centric logistics production management concept. The latter serves as an illustrative model for the management method of sustainable and flexible industrial logistics systems. It also offers a pertinent method of object tracking to facilitate human–machine interaction in virtual reality. In sum, this study resonates with the theme of "Sustainable, Human-Centered/Centric and Resilient Production and Logistic Systems Design and Management".

The specific work is as follows.

1. Three prominent deep learning detection algorithms—Faster R-CNN, SSD, and YOLO—are briefly introduced and compared. The YOLO algorithm is chosen to analyze the logistics warehouse moving-object-tracking problem.
2. A dedicated dataset was meticulously crafted based on the logistics warehouse scenarios. The quality and quantity of the dataset were optimized, yielding enhanced outcomes in object detection specific to logistics warehouses.
3. The principle and process of DeepSORT algorithms are introduced. Finally, an evaluation metric for the effectiveness of logistics warehouse object tracking is established based on the MOT Challenge multi-object-tracking competition. This metric facilitates quantitative assessment, ultimately offering supplementary tools for diminishing the necessity of manual video surveillance within logistics settings.

2. Related Work

The key to object tracking and providing auxiliary means to eliminate manual monitoring in logistics warehouse scenarios hinges on the selection of object-tracking algorithms. In pursuit of this goal, state-of-the-art object-detection algorithms are compared for selecting the best performer among them.

2.1. Faster R-CNN (Faster Region-Based Convolutional Neural Network)

Based on Fast R-CNN, Shaoqing Ren et al. [8] proposed a new Faster R-CNN algorithm in 2017, which takes advantage of CNNs to enhance the procedure of the candidate region generation for detection targets. Notably, the architecture strategically integrates the feature extraction network with the candidate frame-generation network, employing the same convolutional layer network. This astute design choice greatly reduces the time of the whole-image object-detection process. Such an idea of using CNNs for image feature extraction is also reflected in other algorithms. The operational process of the Faster R-CNN algorithm is illustrated in Figure 1.

The process of Faster R-CNN is as follows:

- The input image undergoes processing via R-CNN's convolutional layers to extract image features, resulting in a feature map comprised of feature vectors that capture essential information from the image.
- The feature vectors are subsequently channeled into the Region Proposal Network (RPN) to generate candidate boxes.
- The dimensions of candidate boxes are determined by the pooling layer (ROI pooling), which enables the classification of candidate regions, using a support vector machine (SVM), to identify the target category.

- A precise regression is performed on the classified target boxes.

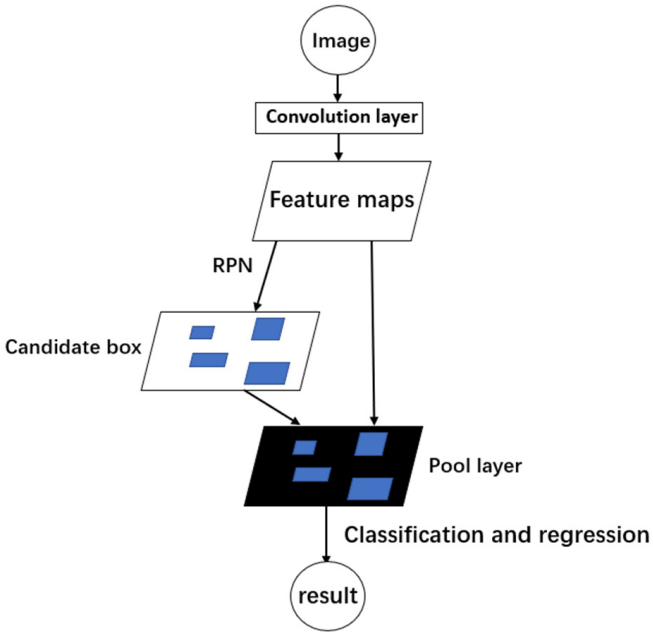


Figure 1. Faster R-CNN algorithm process.

2.2. SSD (Single Shot MultiBox Detector)

Wei Liu [9] proposed a new object-detection algorithm known as SSD in 2015. Operating within the realm of single-stage detection algorithms, SSDs accomplish target localization as well as classification simultaneously.

SSD can be divided into SSD 300 and SSD 512, according to the size of the input image. The following is the basic operation process of the SSD algorithm, with SSD 300 serving as the exemplar:

- The input image (size of 300 pixels × 300 pixels) is fed into a pre-training network (VGG-16) to obtain different feature mappings.
- The feature vectors of Conv4_3, Conv7, and Conv8_2 are employed to establish object-detection boxes at different scales, concurrently performing their classification.
- The NMS (non-maximum suppression) algorithm is employed to eliminate redundant and inaccurate detections. The final detection outcome is then generated, consisting of the most pertinent and trustworthy bounding boxes for the targeted objects.

2.3. YOLO (You Only Look Once)

The YOLO algorithm proposed in 2016 was designed as a new network for object detection from end to end. Its unique design approach increases the speed of the entire network operation by casting the object detection entirely into a regression problem, spanning from input-image segmentation to the classification of targets in each grid [10].

As shown in Figure 2, the process of YOLO is as follows:

- The input image is divided into several grids, and each grid takes responsibility for detecting potential targets within its confines.
- The image of each grid is convolved to obtain the positions of the bounding boxes and the corresponding confidences for multiple targets.
- The final prediction box for each grid's multiple targets forecast is obtained using non-maximum suppression (NMS), which is the result of object detection.

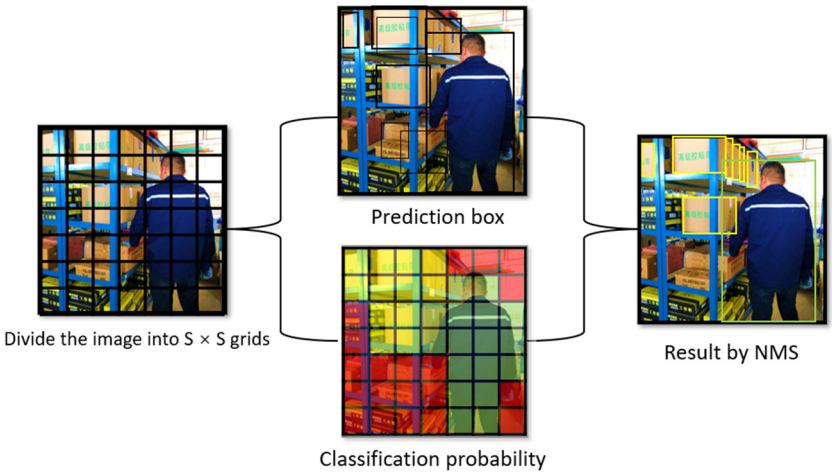


Figure 2. YOLO algorithm process.

2.4. Comparison of the Three Detection Algorithms

Both YOLO and SSD algorithms are single-stage algorithms that are significantly faster than the Faster R-CNN, so they should be preferred in warehouse scenarios that require real-time detection.

The YOLO algorithm has progressed to its fifth version as of the time of writing. YOLOv5 has made further improvements on the previous version, with increased detection accuracy and speed. Among its variants, YOLOv5s maintains excellent detection capability within a compact size (7.3 MB), laying the foundation for prospective explorations into hardware platforms for cost-effective and capacity-efficient logistics warehouse object tracking [11].

The DSSD513 algorithm, an improved iteration derived from the SSD algorithm, exhibits a marked improvement in detection accuracy but diminished detection speed. The three algorithms—YOLOv3-320, YOLOv5, and DSSD513—are compared using the COCO dataset to determine the most suitable one for the logistics warehouse object-tracking scenario. The outcome of the comparison is shown in Table 1.

Table 1. Comparison of the three detection algorithms on the COCO.

Algorithms	mAP@0.5/percent	Delay Time/ms
YOLOv3-320	51.5	22
YOLOv5s	55.4	2
DSSD513	53.3	156

In Table 1, mAP [12] refers to the mean Average Precision (mAP), which measures the accuracy of the object detection. The delay time refers to the time taken by the algorithm to execute the detection process, signifying the processing speed of the algorithm. The delay time is a critical performance metric in real-time applications.

As seen in Table 1, YOLOv5s combines both detection accuracy and speed compared to YOLOv3-320 and DSSD513. It has a compact volume size of only 7.3 MB, which ensures its applicability to real-world scenarios. Therefore, YOLOv5 is chosen as the algorithm for object detection in the logistics warehouse scenario.

2.5. Object-Tracking Algorithms

Object-tracking algorithms play an important role in warehouses. Object-tracking technology offers a valuable way for warehouse managers to ascertain the location and status of items in the warehouses more conveniently, improving operational efficiency. Object-tracking technology could be implemented in various ways. For instance, Zhan [13] combined the Industrial Internet of Things (IIoT) and Digital Twin (DT) for the unsupervised management of cold-chain logistics warehouses, ensuring occupational safety

and health (OSH). These technologies provide higher efficiency and a foundation for the development of unmanned warehouses. However, a convenient and cost-effective means of object tracking is needed that also strikes a balance with practical applications. Therefore, object-tracking technology based on image object detection has gained attention [14]. DeepSORT is widely used for tracking people [15] and ordinary items [16], as a classic technology for target tracking. Previous studies have effectively harnessed the DeepSORT algorithm combined with other technologies to solve indoor object-tracking problems. For instance, Jang [17] designed a lightweight indoor object-tracking method by introducing DeepSORT in scenarios with partially overlapping fields of view (FOVs). Therefore, this article harnesses the potential of DeepSORT to design an object-tracking method for use within logistics warehouses.

3. Logistics Warehouse Object Detection

As stated above, YOLOv5 has been chosen as the detection algorithm for logistics warehouses. Its network structure is illustrated first.

3.1. YOLO Algorithm Network Structure

The YOLO algorithm, known as You Only Look Once, employs a single-stage detection approach inspired by the human visual system. The YOLO algorithm swiftly encompasses the whole-image data for prediction to avoid confusing the target with the background. In some cases, the YOLO detection model struggles with targets that occupy a small image size. Compared to offline detection algorithms, the YOLO algorithm might not exhibit the highest detection accuracy, but the detection accuracy and detection speed should be taken into account in practical applications [18].

As shown in Figure 3 [19], YOLOv5s consists of four main components: Input, Backbone, Neck, and Prediction, where the Backbone convolves and pools the input image to obtain the image feature vector; the Neck transfers the extracted image features to the Prediction part through sampling; and the Prediction part makes predictions based on the image features to obtain the target bounding boxes and prediction classification. There are some basic composite network structures in the network structure, as stated separately below [20].

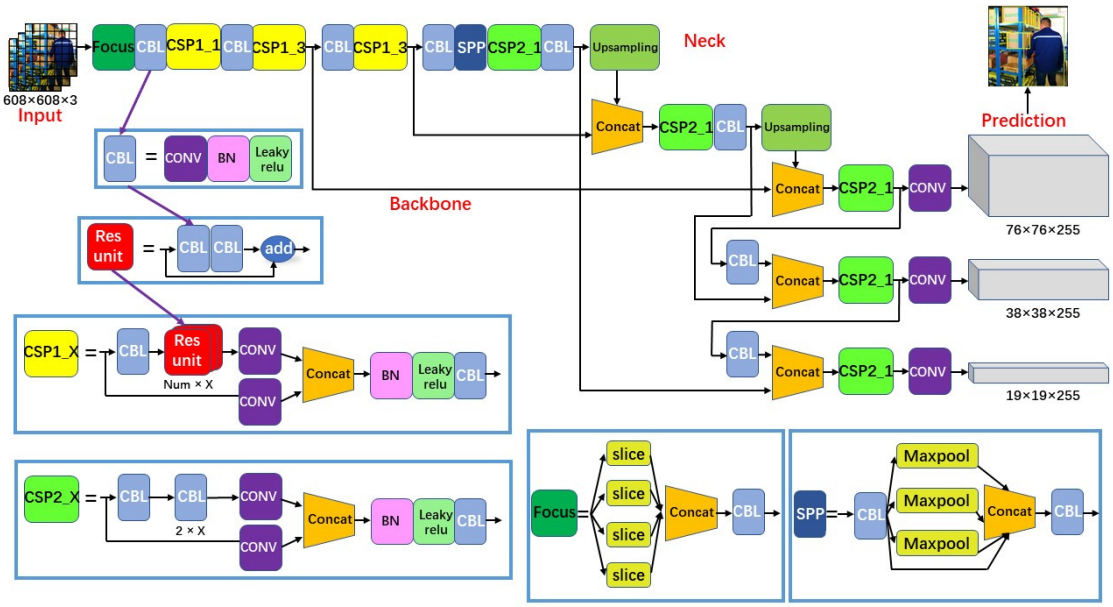


Figure 3. YOLOv5s network structure.

3.1.1. CBL (Convolutional Operation, Batch Normalization Algorithm, and Leaky ReLU Activation Function)

The convolutional operation, the batch normalization algorithm, and the leaky ReLU activation function compose the CBL module, which mainly plays the role of data sampling. The structure is shown in Figure 4.

Figure 4. CBL structure.

3.1.2. Res Unit

The Res unit learns from the residual structure in the ResNet network, which incorporates the original input with the output after two CBL operations to form the residual network structure. This approach makes the structure deeper and avoids gradient vanishing [21]. The structure is shown schematically in Figure 5.

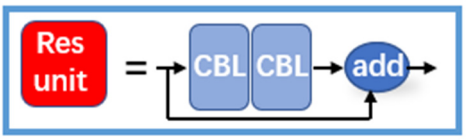


Figure 5. Res unit structure.

The residual structure means that the input x of layer $i-1$ is superimposed directly on the output $I(x)$ of layer i. The output of layer i is then $f(x) = x + I(x)$. The residual structure can be interpreted as a distribution of inputs over different layers of the network. The distribution of input sources makes the whole structure more rational without invalid inputs. The emergence of the residual structure allows the neural network structure to become deeper, and the multi-layer input structure allows the gradient to be updated with the stable gradient [21].

3.1.3. CSP1_X

The CSP1_X structure shown in Figure 6 learns from the network structure of CSPNet. The CSPNet structure unites the features of all input layers to maximize the number of branch inputs used to enhance the learning capability of the network. The CSPNet splits the inputs, which also drives a reduction in computation.

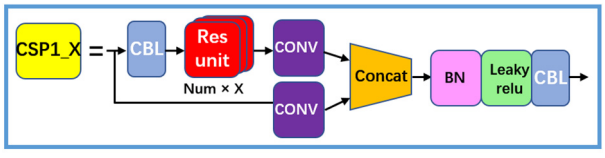


Figure 6. CSP1_X structure.

CSP1_X divides the input into two parts. One part undergoes CBL, multiple residual operation structures, and convolutional operations; the other part directly undergoes convolutional operations. Finally, the two parts are concatenated, which refers to concatenation in neural networks. This approach generally fuses features extracted from multiple frames. After the concatenation, BN, the leaky ReLU activation function, and CBL are connected for sampling [22].

3.1.4. CSP2_X

CSP2_X as shown in Figure 7, is similar to CSP1_X, using CBL to replace the Res unit.

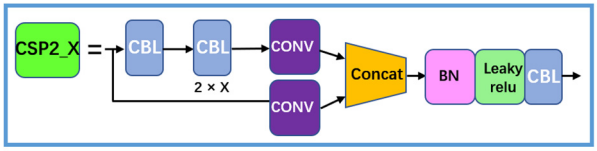


Figure 7. CSP2_X structure.

3.1.5. SPP (Spatial Pyramidal Pooling)

SPP is known as Spatial Pyramidal Pooling, as shown in Figure 8. Its function is resizing the input feature maps to a fixed size. In CNN, SPP is commonly used to solve problems that include inappropriately sized images.

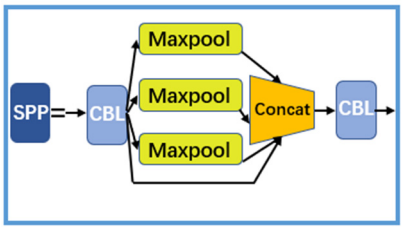


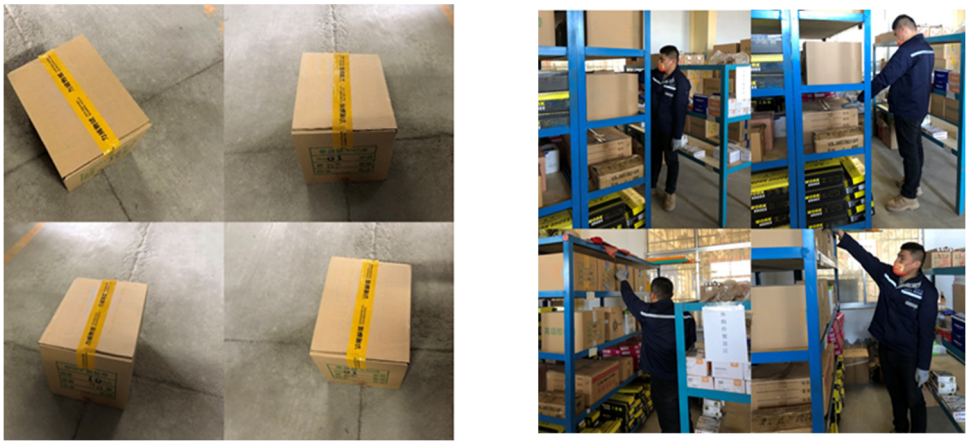
Figure 8. SPP structure.

SPP obtains features through Maxpool, and then merges the features with concatenation to obtain a fixed size feature. In addition, YOLOv5s adds a CBL module for data sampling enhancement before and after the pooling layer [23].

3.2. Dataset and Training

3.2.1. Dataset

The quality and quantity of the dataset is critical for deep learning detection models. In the context of the logistics warehouse scene under investigation, the first dataset included 273 randomly selected videos on websites, depicting workers and parcels. However, the object detection performance of the trained model was poor. Therefore, a logistics warehouse operated by a company was used as the research scene, as shown in Figure 9. There, 790 photos were captured, which included goods and workers, encompassing diverse angles. Light blocking conditions were employed for obtaining complete data.



(**a**) Parcels (**b**) Worker

Figure 9. Photos in the logistics warehouse scene.

The original image set was annotated using the software called Labelimg in txt format [24]. YOLOv5 changed the annotation of the dataset from xml format to txt format, with five parameters per line in the txt file, which is shown in Table 2.

Table 2. YOLOv5 dataset annotation format.

Object_Class	x_Center	y_Center	Width	Height
0	0.85	0.29	0.27	0.12

- To ensure the robustness of the model, 100 photos from the first dataset are intentionally incorporated as a supplement to training.
- object_class indicates the type of object detected, which is usually numbered by a positive integer starting from zero.
- x_center and y_center indicate the coordinates of the center of the target bounding box (normalized, i.e., divided by the width and height of the whole image).
- width and height indicate the width and height of the target bounding box (normalized, i.e., divided by the width and height of the whole image).

3.2.2. Training Process

In order to improve the object-detection model's performance for goods and workers across various scales, the training process incorporates the multi-scale transformation strategy that comes with YOLOv5. The training configuration environment for the YOLOv5-based object-detection model is shown in Table 3.

Table 3. Training configuration environment.

CPU	Graphic Card	Operating System	Software Platform	Training Framework
Intel i5-8300H/8 GB	NVIDIA GTX 1050Ti/4 GB	Windows	CUDA11.4 CUDNN8.2	Pytorch

Based on the training configuration environment in Table 3, the parameters of the YOLOv5 training function were set as shown in Table 4. The workers parameter refers to the number of threads that CPU is working on during YOLOv5 model training. The workers parameter was set to 0 after several tries, as we had insufficient resources for pre-training.

Table 4. Training function parameter settings.

Batchsize	Epochs	Imgsz	Label-Smoothing	Workers
4	300	640	0.1	0

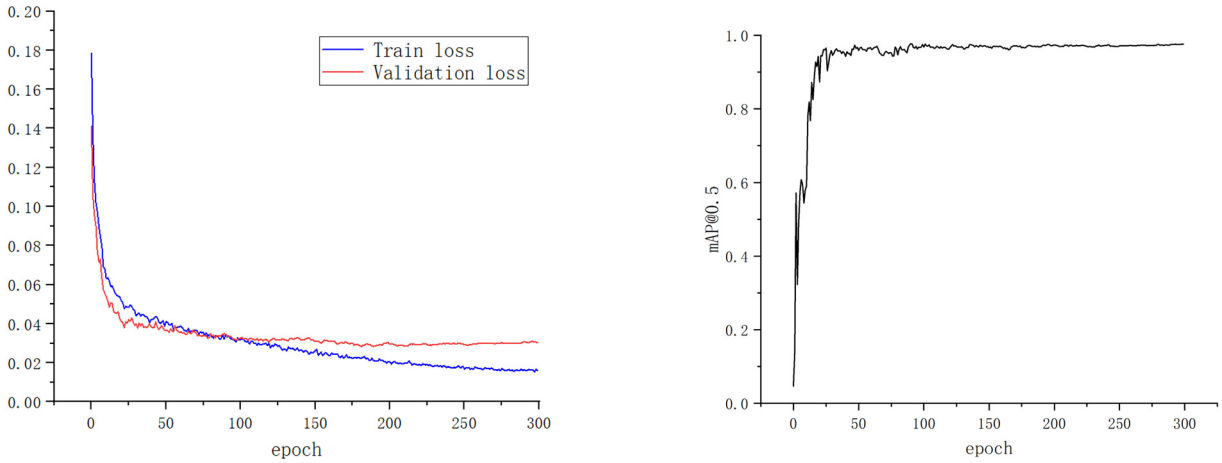
The training framework was Pytorch, a GPU-accelerated neural network framework that is easy to understand and debug code.

The YOLOv5s.pt model was selected for training, and the training process lasted around 24 h. The training process comprised 300 batches, with each batch consisting of 4 randomly selected images from the training set.

The variation of the YOLOv5 loss function during training is shown in Table 5. The variation of train loss, validation loss and mAP@0.5 is shown in Figure 10. From Table 5 and Figure 10, it can be seen that the loss function loss, box_loss, obj_loss and cls_loss gradually became smaller as training progressed. The loss function loss tended to converge at the end of training.

Table 5. Loss function during training.

Batch	100	200	300
loss	0.031857	0.020654	0.016067
box_loss	0.016709	0.009924	0.008079
obj_loss	0.014739	0.010567	0.008079
cls_loss	0.000409	0.000163	0.000063



(**a**) Train/Validation loss–Epoch curve (**b**) mAP@0.5–Epoch curve

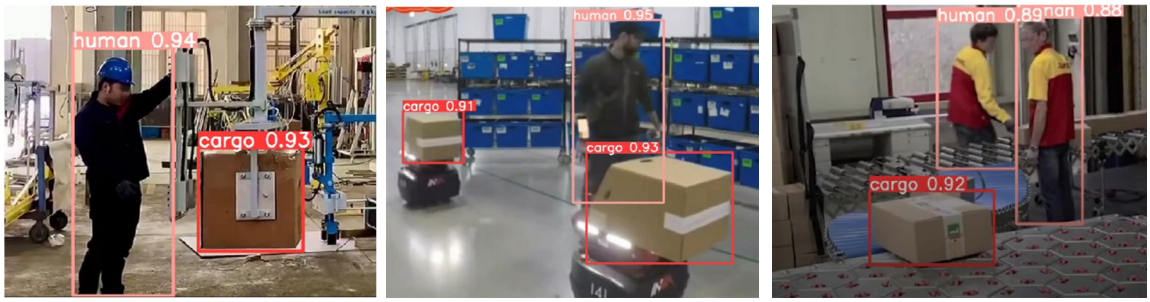
Figure 10. Train/Validation loss–Epoch curve and mAP@0.5-Epoch curve.

3.3. Object-Detection Test

3.3.1. First Training Detection Model

The first training dataset consists of 273 photographs of actual logistics warehouse scenarios, including 50 images of goods, 50 images of staff and 173 images of staff and goods.

The 273 images were fed into the network for training to obtain the detection model. Three test videos were randomly searched for in a logistics scene. To validate the dataset's dependability, each video contained frames of staff, goods and the background. The results of the first logistics warehouse object-detection test are shown in Figure 11.



(**a**) Video 1 test results (**b**) Video 2 test results (**c**) Video 3 test results

Figure 11. Test results for the three videos.

The trained model encountered challenges when detecting a target that was a relatively small part of the whole-image bounding box. Along with a continuous video stream, it tended to lose the tracking frame when the target was obscured. There were also many cases of false positives—for example, the detection for non-goods objects as targets. In cases where goods and workers were obscured from each other, the detection bounding boxes often appeared for only one of the two elements.

In labelling, the authentic-detection bounding box of the workers was defined as the smallest rectangular box that could enclose worker's clothing, hair, etc. However, it was difficult for the detection model to enclose the entire area of the worker when parts of the worker were concealed. When it came to detecting goods, the model often struggled to detect the individual items based on their features, especially in scenarios where multiple goods were stacked each other.

To address the issues above, another dataset was established. The second training incorporated 790 photos from a fixed scene (i.e., the logistics warehouses of Company S)

as the dataset for the detection of parcels and workers. It changed the situation of the relatively small size of datasets and a large number of diverse scenes.

3.3.2. Second Training Detection Model

The second dataset used for training is a set of 790 photos taken from the same logistics warehouses (i.e., Company S's logistics warehouses). The 790 images consisted of 100 images of specific goods (parcels with adhesive tape), 100 images of workers and 590 images of spatial interference between workers and the goods. The test results of this object detection are shown in Figure 12.

Figure 12. Test result.

As can be seen from Figure 12, the detection of obscured targets was significantly improved, and mAP@0.5 reached 0.976. At the same time, the model was still able to detect targets with a relatively small size. Compared with the first training, the second training was improved dramatically, and the trained model detected the targets individually even when the goods blocked each other.

As for the proposed tracking task, a continuous-detection bounding box for the target, as stated above, was not enough; the tracking model needed to maintain the stable invariance of the tracking IDs. In the tracking task, each target was given a unique ID, and keeping the ID constant when the target reappearing was a vital metric for tracking performance reference [25]. To achieve tracking task, we combined the detection model with DeepSORT to preserve the continuity of target IDs.

4. Logistics Warehouse Target Tracking

YOLOv5 and DeepSORT were combined to assign IDs to the tracked targets in the video stream and maintain the stability of the IDs. Originated from the SORT algorithm, DeepSORT now became a tracking algorithm combining Kalman filtering and Hungarian matching, as described below.

4.1. DeepSORT

DeepSORT differs from SORT, mainly defining a new state of target tracks and matching the original detection result with the prediction by judging the state of the tracks [26]. At the same time, tracks with distinct states can be judged by the quantities of matching, which either increase or decrease. The DeepSORT algorithm process is shown in Figure 13 [27,28].

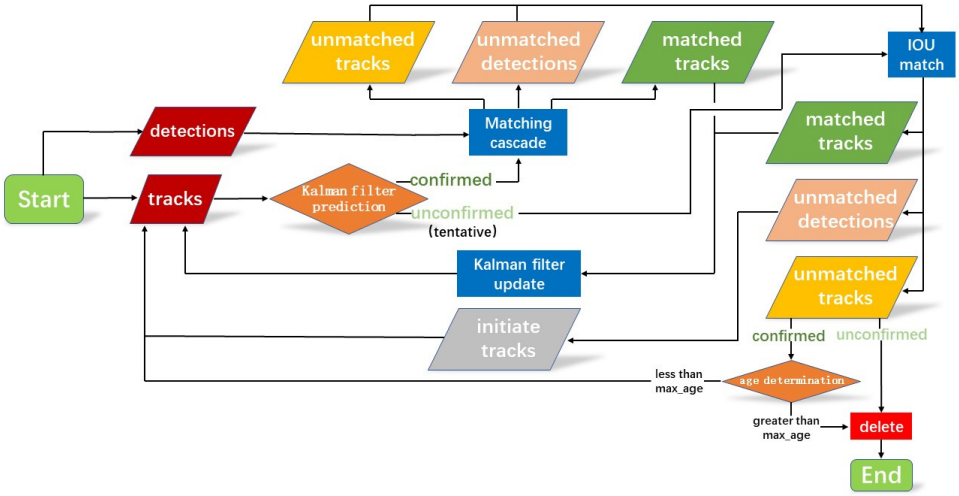


Figure 13. DeepSORT process.

The DeepSORT algorithm process is introduced as follows.

- The existing track is divided into confirmed and unconfirmed states.
- For the tracks with a confirmed state, they are cascaded to match with the detection result. The priority of tracks depends on the time that the tracks exist with the video stream and the number of successful matching attempts. A track that has just been successfully matched has a higher priority for cascade matching [29].
- The process of cascade matching:
    1. The cosine distance and cost matrix between the track feature and the depth feature of the detection are computed.
    2. The martingale distance between the prediction bounding boxes and the detection bounding boxes is computed in the cost matrix.
    3. The previously obtained input cost matrix is matched in Hungary to obtain the final pairing result.
- The unconfirmed state tracks are formed into a new set with the unconfirmed state tracks that were not paired in the second step. The set is matched with the detected bounding boxes that have not been paired in a Hungarian matching with a 1-IOU cost matrix.
- The paired tracks need to update the Kalman filter information based on the detection bounding boxes' information.
    1. If the tracks are paired more than three times, they will change from unconfirmed to confirmed. Tracks lacking pairings will be removed from the set of unconfirmed tracks if the number of consecutive pairing failures exceeds the setting value.
    2. Unpaired tracks will also be removed if they have not a successful pairing previously.
    3. If there is an unpaired detection bounding box, a new target track will be created for it.

4.2. Object-Tracking Test

DeepSORT requires the outcomes of other object-detection algorithms as its input, so YOLOv5 is employed as the input object-detection algorithm. The test video from the first dataset training for YOLOv5 was selected randomly. Figure 14 shows the comparison of DeepSORT and YOLOv5.

(**a**) DeepSORT test results (**b**) YOLOv5 test results

Figure 14. Comparison of DeepSORT with YOLOv5.

As can be seen from Figure 14, the DeepSORT adds the IDs to the detection results of YOLOv5. The observation of DeepSORT throughout the test video showed that the tracking target ID remains relatively constant in the cases where the tracking target reappears. Notably, there is an absence of ID switching among multiple targets in the video stream, maintaining the stability of the tracking IDs (as stated by Lin et al. [30]). However, it is challenging to evaluate the superiority of two algorithms' video tracking effects solely by the human eye. Therefore, the evaluation metrics of the logistics warehouse multi-object tracking were derived to provide a quantitative evaluation.

4.3. Object-Tracking Evaluation Metrics

The object-tracking evaluation metrics utilized in our study are derived from the MOT Challenge competition [31]. The main object-tracking evaluation indicators of the MOT Challenge competition are MOTA (Multi-Object Tracking Accuracy), MOTP (Multiple Object Tracking Precision), Rcll (Recall), MT (Mostly Tracked), ML (Mostly Lost), IDF1 (Identification F-Score), IDP (Identification Precision), IDR (Identification Recall), IDSW (ID Switch), etc., as shown in Figure 15. The MOT Challenge competition employs a set of comprehensive evaluation metrics to evaluate object-tracking performance. These metrics are as shown in Figure 15.

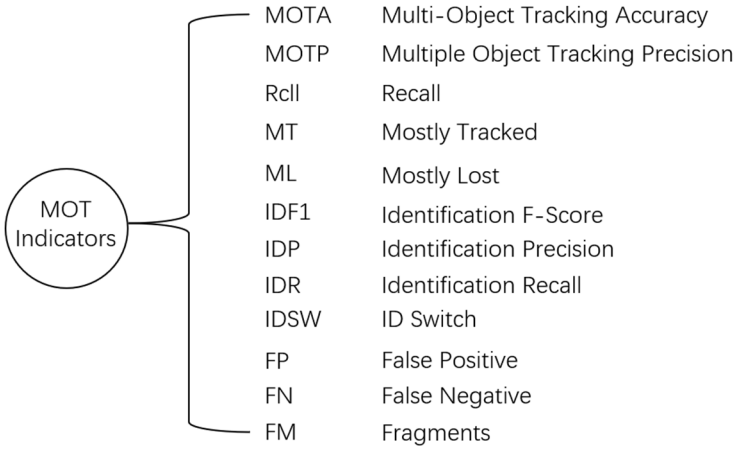


Figure 15. Evaluation indicators for the MOT Challenge competition.

(1) MOTA

The formula for calculating MOTA is as follows:

$$MOTA = 1 - \frac{FN + FP + IDSW}{GT} \quad (1)$$

GT (ground truth) signifies the accurate target bounding box information of the tracking target, which is often obtained by manual annotation or automatic annotation

software in practical problems. MOTA quantifies the proportion of tracking information except false negative (FN), false positive (FP) and ID switching (IDSW) among all true target information.

FN represents the true samples that are identified by the tracking algorithm as false samples; FP represents the false samples that are identified by the tracking algorithm as true samples [32]. The samples here can be interpreted as the tracking bounding boxes of the target for each frame in the video stream. IDSW represents the number of times that the same tracking target ID undergoes changes during the tracking process.

(2) MOTP

MOTP is a metric employed to evaluate the effectiveness of detection during tracking. The calculation formula is as follows.

$$MOTP = \frac{\sum_{t,j} D_{t,j}}{\sum_{t} c_t} \tag{2}$$

where t signifies the current number of frames in the tracked video stream; the variable c_t corresponds to the count of successful pairings between the tracked tracks and the Ground Truth (GT) tracks at frame t. $D_{t,j}$ denotes the distance between the successfully paired two tracks at frame t (generally calculated by Euclidean distance or IOU, Euclidean distance is employed in this paper). The better the detection effect, the shorter the Euclidean distance between the paired tracks. An increased number of successful pairings results in a higher MOTP value.

(3) Rcll

Rcll gauges the proportion of successful target tracking, calculated by dividing the number of successfully paired targets by the number of real targets.

(4) MT and ML

MT indicates the number of tracks that have been successfully paired for over 80% of the total frames during the whole tracking process; ML indicates the number of tracks that have been successfully paired for less than 20% of the total frames during the whole tracking process.

(5) IDP

IDP is similar to the calculation of precision. Accuracy is defined as the proportion of true positive samples among all positive samples, while IDP refers to the proportion of all samples assigned correctly to the target ID.

(6) IDR

IDR is akin to the calculation of recall. Recall refers to the proportion of all true positive samples for which the predictor offers a positive prediction, while IDR refers to the proportion of all samples for which the tracking algorithm has correctly assigned the target ID.

(7) IDF1

IDF1 is then the harmonic mean of IDP and IDR. The calculation formula is as follows.

$$IDF1 = 1 - \frac{2}{\frac{1}{IDP} + \frac{1}{IDR}} \tag{3}$$

(8) FM

FM refers to the number of tracked target tracks that are mistakenly terminated by the tracker in the video stream.

(9) IDSW

IDSW serves as a distinctive form of FM, so the value of FM is generally larger than IDSW.

4.4. Object-Tracking Model Testing

4.4.1. Data Format

A gt.txt file is commonly used in MOT datasets to record information about each tracking frame in the video stream [33]. The Darklabel software is used to annotate the video from Company S's logistics warehouses to obtain a gt.txt file, as shown in Figure 16.

```
0,0,699,216,459,664,1,0,1
0,1,781,542,336,298,1,0,1
1,0,698,208,454,677,1,0,1
1,1,778,507,339,298,1,0,1
2,0,697,184,453,706,1,0,1
2,1,775,473,339,295,1,0,1
3,0,699,147,445,752,1,0,1
3,1,771,435,341,294,1,0,1
4,0,698,100,442,803,1,0,1
4,1,767,398,342,295,1,0,1
```

Figure 16. gt.txt file screenshot.

- The first bit of each line represents the current video frame.
- The second bit indicates the ID number of the target track.
- The third to sixth bits are four parameters, and the third and fourth bits of the data represent the horizontal and vertical coordinates of the upper-left corner of the object-tracking frame. The fifth and sixth bits indicate the width and height of the tracking frame.
- The seventh bit indicates the current status of the target track, with 0 being inactive and 1 being active.
- The eighth bit indicates the target type of the track.
- The ninth bit indicates the visibility ratio of the object-tracking frame, which is related to the interference between this frame and other frames.

A det.txt file is a counterpart to a gt.txt file, which records the object-tracking frame information obtained using the tracking algorithm. The det.txt file has the format shown in Figure 17.

```
3 1 769 472 347 303 −1 −1 −1 −1
4 1 746 332 363 480 −1 −1 −1 −1
5 1 735 190 361 662 −1 −1 −1 −1
6 1 712 73 400 813 −1 −1 −1 −1
7 1 695 16 425 894 −1 −1 −1 −1
8 1 686 0 439 930 −1 −1 −1 −1
8 4 696 0 428 917 −1 −1 −1 −1
9 1 680 0 445 922 −1 −1 −1 −1
```

Figure 17. det.txt file screenshot.

The first to sixth parameters in each line of det.txt files have essentially the same meaning as those in gt.txt files. The seventh parameter indicates the confidence level of the tracked target, and the eighth to tenth parameters are related to 3D target tracking.

4.4.2. Tracking Result

The test video was annotated using Darklabel to obtain a gt.txt file. The det.txt file was generated from the two datasets. The outcomes of the det.txt file were derived through the YOLOv5 detection model combined with DeepSORT. Table 6 shows the tracking outcomes of the first and second tests measured by MOT evaluation metrics, respectively.

Table 6. The first and second tracking results.

Times	MOTA	MOTP	Rcll	MT	ML	IDF1	IDP	IDR	IDSW	FP	FN	FM
First	63.7%	0.131	86.6%	23	0	75.5%	72.4%	78.8%	56	1858	1120	129
Second	75.8%	0.118	87.8%	25	0	83.8%	84.0%	83.5%	26	973	1022	96

The change in the values of the aforementioned metrics demonstrates a notable improvement in the tracking results. Additionally, upon visual inspection of the tracking videos frame by frame, it becomes evident that the tracking target ID stability from the second dataset has also significantly improved.

4.5. Transfer Learning for DeepSORT

Trained object-tracking models often need to be implemented in different scenarios and environments in practical applications, which requires the high transferability and generalization of the trained models. To this end, transfer learning could be employed to retrain the model through a new dataset, enhancing the object-tracking performance. Transfer learning [34] is a proposed machine learning technique that facilitates the transfer of trained model knowledge from one task to another related task model. The steps to perform transfer learning on the trained DeepSORT model are as follows:

- Prepare the dataset: The new dataset needs to be organized, ensuring it is representative and diverse.
- Transfer learning strategy:
  - Feature extraction: The majority of the pre-trained DeepSORT model's weights need to be retained while solely adjusting the output layer to suit the new task. It is necessary to freeze the weights of the initial layers to maintain low-level feature extraction capabilities.
  - Fine-tuning: In addition to modifying the output layer, fine-tuning some of the lower-level weights to adapt to the new task is considered, allowing certain lower-level weights to undergo slight adjustments.
- Model training: Training the adjusted model using the new dataset.
- Model evaluation: Evaluate the retrained model and compute evaluation metrics to evaluate the model's performance on the new task.
- Refinement and iteration: Further refining the model based on the evaluation results and adjusting hyperparameters or fine-tuning the model network.

The transfer learning process for DeepSORT is shown in Figure 18.

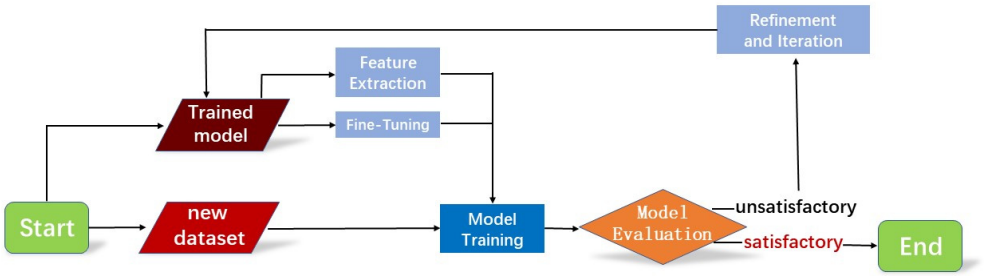


Figure 18. Transfer learning for DeepSORT.

5. Discussion

In the absence of relevant research and open-source datasets for logistics packages, this study has constructed a logistics warehouse image dataset, which was applied to the combined algorithm of YOLOv5 and DeepSORT to achieve object tracking within logistics warehouse scenarios. The combined algorithm was numerically evaluated by the MOT Challenge evaluation metrics. The study has achieved good tracking performance and provided a validation approach for the effectiveness of eliminating manual video surveillance

in logistics scenarios. This accomplishment signifies an effective smart monitoring approach for "Sustainable, Human-Centered/Centric and Resilient Logistic Systems Design and Management".

The combination of YOLOv5 and DeepSORT is specifically designed to cater to human-centric logistics scenarios, where real-time object detection and multi-object tracking are essential. The following are some applications:

- One primary application is the determination of warehouse zone occupancy. By accurately tracking the movement of objects in different zones, real-time occupancy data could be easily garnered, which optimizes inventory management and streamlines logistics. Additionally, it facilitates the prediction of space utilization trends, enabling the proactive allocation of resources.
- The method contributes to safety protocols within warehouses. By continually tracking the trajectories of objects and employees, potential hazards and unsafe practices can be identified promptly, minimizing the risk of accidents.
- The method's versatility extends to inventory management, offering insights into warehouse stock movement and depletion rates, and preventing overstock situations.

In conclusion, the presented object-tracking method holds immense potential for revolutionizing warehouse operations. Its applications include occupancy determination, safety monitoring, efficiency enhancement, and inventory management, contributing to the optimization and development of future advancements in human-centric logistics warehouse environments. Such a combination also provides value-added support by enabling data-driven decision-making to optimize resource allocation for enterprises.

However, it is pertinent to acknowledge that the computational demands of the proposed method could increase significantly with a larger number of objects and types of employees. This might necessitate more distributed computing resources, which could impact the feasibility of real-time tracking. The proposed algorithm may need to be adapted to handle a larger volume of data while maintaining reasonable processing times. A careful balance between algorithmic sophistication, computational efficiency, and practical considerations may be required to address these inherent limitations and trade-offs.

6. Conclusions

This research focuses on the design of a moving-object-tracking scheme for logistics warehouses. It primarily encompasses the design of a deep learning-based detection algorithm for logistics warehouses and the design of experiments for tracking moving targets in logistics warehouses. The main contributions of the dataset, algorithm, and evaluation method are described below:

- In order to solve the problem of moving-object tracking in logistics warehouses, a specific company was selected as the sampling object to make the dataset. The dataset encapsulated tracking outcomes for both goods and workers in video streams. The tracking model was further strengthened and applied to most logistics warehouse scenarios through the optimization of the dataset.
- YOLOv5s was selected as the base model on account of both detection accuracy and detection speed. Despite incorporating the DeepSORT network for multi-object tracking, the tracking speed remained sufficiently fast (within 30 ms). In addition, YOLOv5s boasted a compact size of 7.3 MB, thus holding potential for application in the actual logistics scene.
- The multi-objective evaluation metrics from the MOT Challenge were employed as the performance evaluation metrics of the logistics warehouse object-tracking system, assessing the degree of improvement of the tracking effect.
- In the detection test, the mAP@0.5 achieved an impressive score of 0.976, demonstrating the high accuracy of the object-detection model. Moreover, the tracking test yielded a commendable MOTA of 75.8% and a Rcll of 86.6%, validating the effectiveness of the tracking algorithm. Moreover, the method of transfer learning for trained DeepSORT models is introduced.

Author Contributions: Conceptualization, T.X. and X.Y.; methodology, T.X.; software, T.X.; validation, T.X.; formal analysis, T.X.; investigation, T.X.; resources, T.X.; data curation, T.X.; writing—original draft preparation, T.X.; writing—review and editing, X.Y.; visualization, T.X.; supervision, X.Y.; project administration, X.Y.; funding acquisition, X.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Guangdong Basic and Applied Basic Research Foundation (2022A1515010095, 2021A1515010506), the National Natural Science Foundation of China, and the Royal Society of Edinburgh (51911530245).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data that support the findings of this study are available from the corresponding author upon reasonable request.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Tubis, A.A.; Rohman, J. Intelligent Warehouse in Industry 4.0-Systematic Literature Review. *Sensors* **2023**, *23*, 4105.
2. Neumann, W.P.; Winkelhaus, S.; Grosse, E.H.; Glock, C.H. Industry 4.0 and the human factor—A systems framework and analysis methodology for successful development. *Int. J. Prod. Econ.* **2021**, *233*, 107992. [CrossRef]
3. Chen, H.; Zhang, Y. Regional Logistics Industry High-Quality Development Level Measurement, Dynamic Evolution, and Its Impact Path on Industrial Structure Optimization: Finding from China. *Sustainability* **2022**, *14*, 14038. [CrossRef]
4. Yan, B.-R.; Dong, Q.-L.; Li, Q.; Amin, F.U.; Wu, J.-N. A Study on the Coupling and Coordination between Logistics Industry and Economy in the Background of High-Quality Development. *Sustainability* **2021**, *13*, 10360. [CrossRef]
5. Chen, Y.; Yang, B. Analysis on the evolution of shipping logistics service supply chain market structure under the application of blockchain technology. *Adv. Eng. Inform.* **2022**, *53*, 13. [CrossRef]
6. Hong, T.; Liang, H.; Yang, Q.; Fang, L.; Kadoch, M.; Cheriet, M. A Real-Time Tracking Algorithm for Multi-Target UAV Based on Deep Learning. *Remote Sens.* **2023**, *15*, 2. [CrossRef]
7. Li, J.; Zhi, J.; Hu, W.; Wang, L.; Yang, A. Research on the improvement of vision target tracking algorithm for Internet of things technology and Simple extended application in pellet ore phase. *Future Gener. Comput. Syst.* **2020**, *110*, 233–242. [CrossRef]
8. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef] [PubMed]
9. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In Proceedings of the Computer Vision—ECCV 2016, Amsterdam, The Netherlands, 11–14 October 2016; pp. 21–37.
10. Dong, X.; Yan, S.; Duan, C. A lightweight vehicles detection network model based on YOLOv5. *Eng. Appl. Artif. Intell.* **2022**, *113*, 104914. [CrossRef]
11. Simeth, A.; Kumar, A.A.; Plapper, P. Using Artificial Intelligence to Facilitate Assembly Automation in High-Mix Low-Volume Production Scenario. *Procedia CIRP* **2022**, *107*, 1029–1034. [CrossRef]
12. Qu, Z.; Gao, L.-y.; Wang, S.-y.; Yin, H.-n.; Yi, T.-m. An improved YOLOv5 method for large objects detection with multi-scale feature cross-layer fusion network. *Image Vis. Comput.* **2022**, *125*, 104518. [CrossRef]
13. Zhan, X.; Wu, W.; Shen, L.; Liao, W.; Zhao, Z.; Xia, J. Industrial internet of things and unsupervised deep learning enabled real-time occupational safety monitoring in cold storage warehouse. *Saf. Sci.* **2022**, *152*, 105766. [CrossRef]
14. Soleimanitaleb, Z.; Keyvanrad, M.A.; Jafari, A. Object Tracking Methods: A Review. In Proceedings of the 2019 9th International Conference on Computer and Knowledge Engineering (ICCKE), Mashhad, Iran, 24–25 October 2019; pp. 282–288.
15. Abdulrahman, B.; Zhu, Z. Real-time pedestrian pose estimation, tracking and localization for social distancing. *Mach. Vis. Appl.* **2022**, *34*, 8. [CrossRef]
16. Zhou, P.; Liu, Y.; Meng, Z. PointSLOT: Real-Time Simultaneous Localization and Object Tracking for Dynamic Environment. *IEEE Robot. Autom. Lett.* **2023**, *8*, 2645–2652. [CrossRef]
17. Jang, J.; Seon, M.; Choi, J. Lightweight Indoor Multi-Object Tracking in Overlapping FOV Multi-Camera Environments. *Sensors* **2022**, *22*, 5267. [CrossRef] [PubMed]
18. Kumar, A.; Kalia, A.; Verma, K.; Sharma, A.; Kaushal, M. Scaling up face masks detection with YOLO on a novel dataset. *Optik* **2021**, *239*, 166744. [CrossRef]
19. A Complete Explanation of the Core Basic Knowledge of Yolov5 in the Yolo Series. Available online: https://zhuanlan.zhihu.com/p/143747206 (accessed on 19 June 2023).
20. Jiang, C.; Ren, H.; Ye, X.; Zhu, J.; Zeng, H.; Nan, Y.; Sun, M.; Ren, X.; Huo, H. Object detection from UAV thermal infrared images and videos using YOLO models. *Int. J. Appl. Earth Obs. Geoinf.* **2022**, *112*, 102912. [CrossRef]
21. Li, S.; Li, C.; Yang, Y.; Zhang, Q.; Wang, Y.; Guo, Z. Underwater scallop recognition algorithm using improved YOLOv5. *Aquac. Eng.* **2022**, *98*, 102273. [CrossRef]

22. Wang, H.; Zhang, S.; Zhao, S.; Wang, Q.; Li, D.; Zhao, R. Real-time detection and tracking of fish abnormal behavior based on improved YOLOV5 and SiamRPN++. *Comput. Electron. Agric.* **2022**, *192*, 10. [CrossRef]
23. Li, J.; Qiao, Y.; Liu, S.; Zhang, J.; Yang, Z.; Wang, M. An improved YOLOv5-based vegetable disease detection method. *Comput. Electron. Agric.* **2022**, *202*, 107345. [CrossRef]
24. Kaufmane, E.; Sudars, K.; Namatēvs, I.; Kalniņa, I.; Judvaitis, J.; Balašs, R.; Strautiņa, S. QuinceSet: Dataset of annotated Japanese quince images for object detection. *Data Brief* **2022**, *42*, 108332. [CrossRef]
25. Chen, F.; Wang, X.; Zhao, Y.; Lv, S.; Niu, X. Visual object tracking: A survey. *Comput. Vis. Image Underst.* **2022**, *222*, 42. [CrossRef]
26. Lin, Y.; Chen, T.; Liu, S.; Cai, Y.; Shi, H.; Zheng, D.; Lan, Y.; Yue, X.; Zhang, L. Quick and accurate monitoring peanut seedlings emergence rate through UAV video and deep learning. *Comput. Electron. Agric.* **2022**, *197*, 106938. [CrossRef]
27. Introduction to Multi Object Tracking (MOT). Available online: https://zhuanlan.zhihu.com/p/97449724 (accessed on 19 June 2023).
28. Wang, Z.; Zheng, L.; Liu, Y.; Wang, S. Towards Real-Time Multi-Object Tracking. *arXiv* **2019**, arXiv:1909.12605.
29. Ma, L.; Liu, X.; Zhang, Y.; Jia, S. Visual target detection for energy consumption optimization of unmanned surface vehicle. *Energy Rep.* **2022**, *8*, 363–369. [CrossRef]
30. Lin, X.; Li, C.-T.; Sanchez, V.; Maple, C. On the detection-to-track association for online multi-object tracking. *Pattern Recognit. Lett.* **2021**, *146*, 200–207. [CrossRef]
31. Yang, F.; Wang, Z.; Wu, Y.; Sakti, S.; Nakamura, S. Tackling multiple object tracking with complicated motions—Re-designing the integration of motion and appearance. *Image Vis. Comput.* **2022**, *124*, 104514. [CrossRef]
32. Wong, P.K.-Y.; Luo, H.; Wang, M.; Leung, P.H.; Cheng, J.C.P. Recognition of pedestrian trajectories and attributes with computer vision and deep learning techniques. *Adv. Eng. Inform.* **2021**, *49*, 101356. [CrossRef]
33. Tan, C.; Li, C.; He, D.; Song, H. Towards real-time tracking and counting of seedlings with a one-stage detector and optical flow. *Comput. Electron. Agric.* **2022**, *193*, 106683. [CrossRef]
34. Pan, S.J.; Yang, Q. A Survey on Transfer Learning. *IEEE Trans. Knowl. Data Eng.* **2010**, *22*, 1345–1359. [CrossRef]