Scheduling Optimization And Coordination With Target Tracking Under Heterogeneous Networks In Automated Guided Vehicles (AGVs)

Thesis submitted in accordance with the requirements of the University of Liverpool for the degree of Doctor in Philosophy by

Tongpo Zhang

June 2023

Abstract

Throughout the development of the multi-AGV systems, prevailing research directions contain improving the performance of individual AGV, optimizing the coordination of multiple AGVs, and enhancing the efficiency of communication among AGVs. Current researchers tend to pay attention to one research direction at a time. There is a lack of research on the overall AGV system design that tackles multiple critical design aspects of the system. This PhD research addresses four key factors of the AGV system which are AGV prototypes, target tracking algorithms, AGVs scheduling optimization and the communication of a multi-AGV system. Extensive field experiments and algorithm optimization are implemented. Comprehensive literature review is conducted to identify the research gap. The proposed solutions cover the following three aspects of the AGV system design including communication between AGVs, AGVs scheduling and computer vision in AGVs.

For AGV communication, a network selection optimization algorithm is presented. An improved method for preventing convolutional neural network (CNN) immune from backdoor attack to ensure a multi-AGV system's communication security is presented. Meanwhile, a transmission framework for a multi-AGV system is presented. Those methods are used to establish a safe and efficient multi-AGV system's communication environment. For AGV scheduling, a multi-robot planning algorithm with quadtree map division for obstacles of irregular shape is presented. In addition, a scheduling optimization platform is presented. These methods are used to make a multi-AGV system have a shorter time delay and decrease the possibility of collision in a multi-robot system.Meanwhile, a scheduling optimization method based on the combination of a handover rule and the A* algorithm is proposed. The system properties that may affect the scheduling performance are also discussed. Finally, the overall performance of the newly integrated scheduling system is compared with other scheduling systems to validate its superiority and shortcomings in different corresponding work scenarios. Computer vision in AGV is investigated in detail. To improve an individual AGV's performance, an improved Camshift Algorithm has been proposed and applied to AGV prototypes. Furthermore, three deep learning models are tested under specific environments. In addition, based on the designed algorithm, the AGV prototype is able to make a convergent prediction of the pixels in the target area after the first detection of the object. Relative coordinates of the target can be located more accurately in less time. As tested in the experiments, the system architecture and new algorithm lead to reduced hardware cost, shorter time delay, improved robustness, and higher accuracy in tracking.

With the three design aspects in mind, a novel method for real-time visual tracking and distance measurement is proposed. Tracking and collision avoidance functions are tested in the designed multi-AGV prototype system. Detailed design procedure, numerical analysis of the measurement data and recommendations for further improvement of the system design are presented.

Acknowledgements

First of all, I would like to express my sincere gratitude to my primary PhD supervisor Associate Professor Limin Yu, who gave me significiant guidance and continuously support my PhD study and research. Her guidance helped me in all the period of my PhD research. I would also like to sincerely thank my Co-Supervisors, Professor Enggee Lim, Professor Fei Ma and Dr. Migue Lopez-Benitez for their excellent guidance and constructive comments.

I would like to thank my research group members, Miss. Tiantian Guo, Mr. Xiaokai Nie, Miss. Wuyan Zhao, Mr. Yunze Song, Mr. Zejian Kong, Mr.Chenyuan Li, Mr. Haodong Liu, Mr. Yuang Zhou, Miss. Yuxin Wan, Mr.Zimai Ma and Miss. Yixuan Huang. I felt honoured to work with them together and special thanks to them for our friendship. The success of this work is to a large extent as a result of their effort.

In addition, my gratitude is dedicated to my parents for their carefulness and encouragement. They have provided a lot of support to me in completing my Ph.D study.

Finally, I would like to thank my closest friends and the people who I am not able to list. Thank you and best wishes.

Contents

List of Abbreviations

AGV Automated Guided Vehicles
QoS Quality Of Service
RSS Received Signal Strength
MADM Multiple Attribute Decision Making
SAW Simple Additive Weighting
TOPSIS Technique for Order Preference by Similarity to an Ideal Solution
GRA Grey Relational Analysis
ELECTRE Elimination and Choice Translating Reality
VIKOR VlseKriterijumska Optimizacija IKompromisno Resenje method
MDP Markov Decision Process
EC Energy Consumption
PD Propagation Delay
PL Packet Loss
CV Computer Vision
NLP Natural Language Processing
SR Speech Recognition
UAP Universal Adversarial Patch
CNN Convolutional Neural Network
NC Neutral Cleanse
RS Randomized smoothing
ART Adversarial Robustness Toolbox
AC Activation clustering
Camshift Continuously Adaptive Mean Shift
DL deep learning
YOLO you-only-look-once
Faster RCNN Faster Region proposal network
SSD Single shot multibox detector
ARMA Autoregressive moving average
NN Neural network
GNN Graph neural network
DOC Deep occlusion network

PAFs Part Affinity Fields
VOC Visual Object Class
MNIST Modified National Institute of Standards and Technology
DNN Deep Neural Networks
FL Federated deep learning
PN Petri Nets
CSA Cache-Conscious Indices
HSV Hue, Saturation, Value
RGB Red, Green, Blue
RCNN Region proposals Convolutional Neural Network
RPN Region Proposal Network
IoU Intersection over Union
L-SSD Lightweight Feature Fusion Single Shot Multibox Detector
MIL Multiple Instance Learning
KCF Kernelized Correlation Filter
TLD Tracking Learning Detection
MOSSE Mininum Output Sum of Squared Error
DCF Dual Correlation Filter
EAN European Article Number
ISBN International Standard Book Number
QR Quick Response Code
ISD International Standardization Orgnization
EKF Extended Kalman Filter
UKD Un-scented Kalman Filter
NMS Non-Maximunmn Suppression
ROI Region of Interest
ACC Adapative Cruise Control
ROS Robot Operating System
DWA Dynamic Window Approaches

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

A mature and popular accepted Scheduling Optimization and Coordination of Automated Guided Vehicles (AGVs) system has eager demand. How to establish such a system allows researchers to design different system environments, create different scheduling and optimization methods, and use different integrated navigation systems. A system environment is the framework of a multi-AGV system. A scheduling and optimization method is to enable AGVs to run properly. An integrated navigation system is used to improve an AGV's working efficiency. Most researchers focus on one of the mentioned aspects. However, to establish an efficient multi-AGV system, all three aspects should be considered and combined.

It can be assumed that there is a multi-AGV system that performs a series of tasks in a known environment. There are multiple task target points in the scene, a starting point, and a task list. After accepting the task, each AGV starts from the starting point and moves to the task target point at a certain interval. After each AGV reaches the target point of the task, it completes the task and returns to the starting point. On the way, each AGV collects information in the environment through a variety of different sensors equipped by itself and its own fiducial marker. Meanwhile, AGVs will form a queue, change the queue, carry out path planning, and avoid obstacles on the way. The data collected

by sensors equipped in AGV will be transmitted into the central platform or the cloud platform depending on the specific situation. The structure of such a situation can be seen in Figure 1.1.

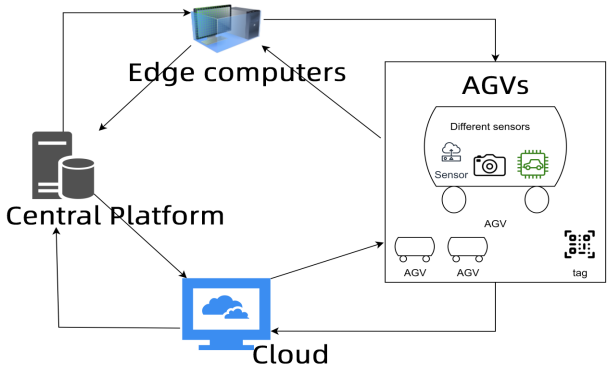

Figure 1.1: The structure of a multi-AGV system

The problems this project wants to solve contain applying a visual tracking algorithm to an AGV system, AGVs' scheduling optimization and coordination, and a heterogeneous network environment. Currently, most AGV systems rely on electromagnetic guidance, magnetic tape guidance, QR Code Guidance and Laser Navigation. Visual tracking algorithm has not been widely used in AGV systems. The visual tracking algorithm's application will make the AGV system good at capturing motion information, enhance the anti-jamming ability, and improve error correction capability. The project will first use and improve the current popular visual tracking algorithm. In terms of quality of service (QoS), price, and coverage space, all available wireless networks included in heterogeneous networks have their own characteristics. Users or devices can easily move between different networks based on a suitable network selection algorithm. AGVs' scheduling optimisation and coordination will be realized by simulations. A visual interface will be created. Essential rules to improve AGVs' scheduling and coordination will be considered. Data collected from the designed visual tracking algorithm, the designed network selection algorithm, and the heterogeneous network will also be contained.

This project will build an AGV system, which will contain several AGV prototypes. After that, the project will focus on three aspects. The first aspect of this project focuses

on improving the efficiency and safety of information transmission in AGV systems. This project plans to build an AGV information transmission framework, optimize a network selection algorithm and establish a method for preventing making CNN immune from backdoor attacks. The second aspect of this project will focus on the coordination and optimization of robot cluster scheduling algorithms, robot obstacle avoidance algorithms, and scheduling rules. There are only a limited number of prototype AGVs that can be built in the project, but in a real industrial environment, there may be dozens or even hundreds of AGVs in a single room. Therefore, this aspect will also focus on building a simulation platform to simulate the performance of AGV systems in a real industrial environment. The platform should be able to verify whether the AGV system' performance is improved based on the input optimized fused robot cluster scheduling algorithms, robot obstacle avoidance algorithms, and scheduling rules. The third one is computer vision direction. Adding vision sensors to AGVs is a popular aspect at present. However, vision sensors will increase the cost. Although the vision sensor can obtain more information compared to other sensors, it will also take up more resources. Since vision sensors focus on image processing, the resources they take up include collecting images, training image data sets, and recognizing targets. How to lighten this process will be the aspect of this project. This project will focus on optimization and innovation in traditional target detection algorithms and deep learning target detection algorithms.

1.2 Thesis organisation

In order to present a legible thesis, the overall structure of the study takes the form of seven chapters. The chapters are listed as follows: Chapter 1 describes the motivation, organisation, contributions, publications and patents.

Chapter 2 first provides a review of the history of the AGV. Then it provides a review of the heterogeneous network environments. Subsequently, different scheduling and optimization methods applied in multi-AGV systems or robot systems are discussed. After that different visual tracking methods which contain traditional visual tracking algorithms, deep learning visual tracking models, and tags are reviewed. Furthermore, all functions

mentioned above will be applied to a multi-AGV system with created prototypes and creative contributions. Finally, the relationship between AGVs and the industrial area has been reviewed.

Figure 1.2 summarizes chapter 3, chapter 4, chapter 5 and chapter 6.

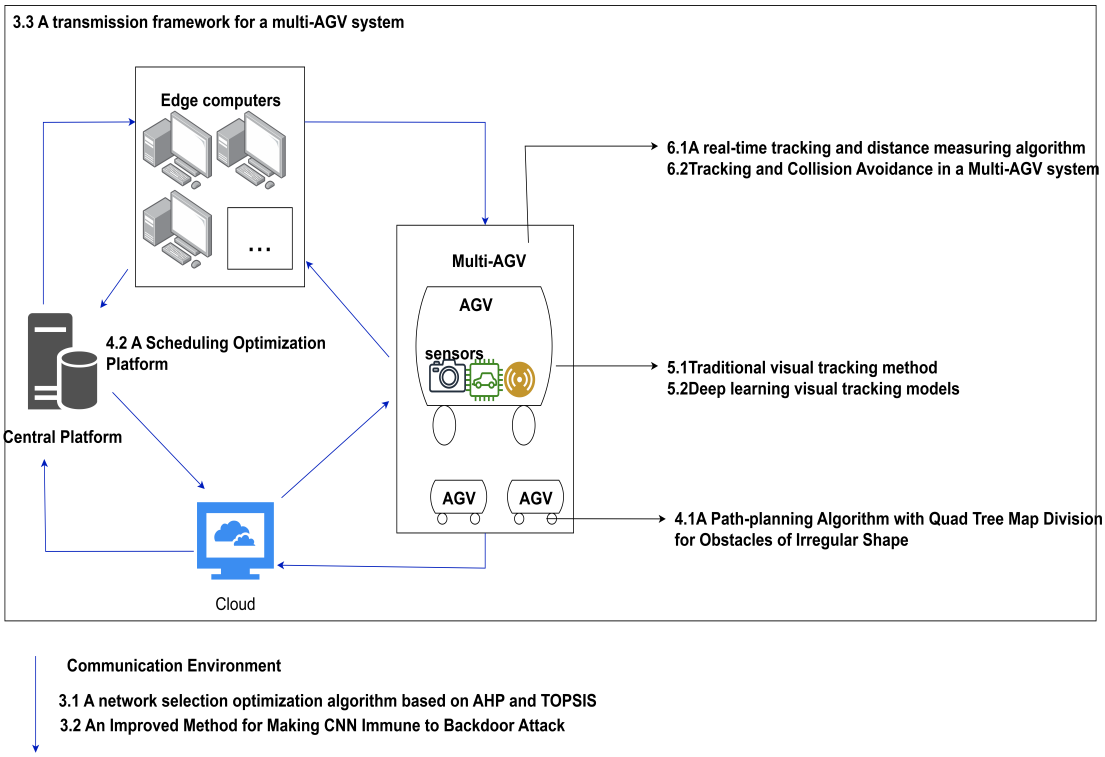

Figure 1.2: A summary of Chapter 3-6

In chapter 3, a network selection optimization algorithm based on Analytical Hierarchy Process (AHP) and Preference by Similarity to an Ideal Solution (TOPSIS) in a heterogeneous network environment is created to improve a multi-AGV system's communication environment. An improved method for preventing Convolutional neural network (CNN) immune from backdoor attacks is proposedto ensure a multi-AGV system's communication environment's security. Based on the designed network selection optimization algorithm and the improved method for making CNN immune to backdoor attacks, a framework for a multi-AGV system is proposed.

In chapter 4, a scheduling optimization method that contains a simulation platform and a multi-robot path-planning algorithm with quad tree map division for obstacles of irregular shape are created to improve multi-AGV's scheduling and coordination. Obstacles somewhat increase the complexity of the environment and the computational load of the system.

In chapter 5, an improved traditional visual tracking method is proposed to improve an AGV's integrated navigation. Furthermore, several deep learning visual tracking methods are tested and analyzed.

In chapter 6, functions realized in chapter 3, chapter 4, and chapter 5 are applied to a multi-AGV with designed prototypes. A real-time visual tracking and distance measuring algorithm based on a deep learning visual tracking model, the single shot multibox detector (SSD) model is created and tested. Finally, a multi-AGV system containing tracking and collision avoidance AGVs is created and tested.

Finally, the findings and conclusions are drawn in Chapter 7, followed by an outlook toward future work.

1.3 Contributions

This thesis aims at proposing a multi-AGV system with a designed heterogeneous network environment, suitable scheduling optimization, and coordination method, and an improved integrated navigation system.

The major contributions have been made in this research are listed as follows:

(1)A network selection optimization algorithm based on AHP and TOPSIS in a heterogeneous network environment is presented.AHP is used to determine the importance of network parameters, and TOPSIS is used to sort the three possible candidate networks from superior to inferior. The algorithm is created to meet the need of the current integration of different access technologies. This algorithm can select the best access network in a heterogeneous network environment to improve the performance of the above parameters. The simulation results show that the algorithm can meet the user's personalized application requirements to select the optimal access network. In future work, we will consider

running the algorithm on various terminals under a heterogeneous network environment, so as to test the switching efficiency of the algorithm in practical application scenarios.

(2)An improved method for preventing CNN immune from backdoor attacks to ensure a multi-AGV system's communication environment is presented. The essence of this method is to enable the neural network to learn the characteristics of data labeled as poison data and enable the virus data to be distinguished into an independent category by the network. The final results provided that the designed method has good resistance.

(3)A transmission framework for a multi-AGV system is presented. The system takes the message and big files apart. It transmits the message through the cloud platform, and transmits the data to the edge computers. Adding encryption and decryption of big data to the transmission can ensure the real-time nature of the message and the privacy of the big data at the same time. The testing results show the designed framework can realize the message and object model message transmission between AGV and the cloud platform through MQTT to achieve real-time monitoring. It can also enable communication between devices in different network environments and cloud platforms. At the same time, it can realize the transfer of large files and the encryption and decryption of public and private keys from multiple devices to the same device The framework can run smoothly in the simulation environment and has good performance. This framework can be used in diversity environment.

(4)A multi-robot path-planning algorithm with quad tree map division for obstacles of irregular shape is presented. The innovation of the system design lies in the shortest path algorithm with the combination of the waiting mode and motion coordination, as well as the new motion coordination method based on quad tree division. The designed algorithm not only has a low time delay but also decreases the possibility of collision in the multi-robot system. Compared with a general graph like the Voronoi graph, it reduces computation complexity and is flexible for a multi-robot system.

(5)A scheduling optimization platform is presented. Related experiments and tests have been done. The superiority of the proposed scheduling optimization method has been proved. Meanwhile, parameters that can influence the system's performance have been tested. The handover mode is better than the basic mode. Most parameters are correlated

with the system's time cost except the number of AGVs. There exists the best number of AGVs in a multi-AGV system using handover mode. The handover mode means AGVs will coordinate with each other.

(6)An improved Camshift Algorithm is presented and applied to AGV prototypes. It was based on the improved Camshift algorithm combining Camshift, Kalman filter and windowing algorithm. The designed system tackles several unsolved problems in the original Camshift algorithm for AGV tracking including adaptive color tracking, interference cancelation under dynamic background and maintenance of constant tracking distance. The experimental results demonstrated a reduced tracking time delay of 12%. The stability and robustness of target tracking have a performance gain of 51.74%. The system architecture provides a solution to balance the limited computation power of individual AGVs and the requirement of computation-intensive image processing for AGV tracking. The experimental results also revealed the challenges encountered in the current AGV tracking system design including speed limit and complexity in the color/shape of the target, which is worth further investigation.

(7)Mobile robot tracking with three deep learning models under specific environments is presented. We construct three recent object detection models, which are Faster RCNN, SSD, and YOLOv5. Comprehensive experiments are conducted to evaluate the detection performance with our data sets. Subsequently, we design a Random Erasing method to create new testing data set with various partial occlusions to verify the system performance. We also use image views from different angles, distances, and surrounding environments to test the model performance. Finally, We analyze numerically the adaptivity of the models against different degrees of occlusions through experiments. Experimental results showed that the SSD model has the best performance and is most promising for the addressed application.

(8)A novel method for real-time visual tracking and distance measurement based on SSD is proposed. This algorithm can measure the distance of the tracking object with high accuracy in real-time. The error can be controlled at a centimeter level.

(9)A tracking and collision avoidance system is applied in a multi-AGV system. The system has completed three functionalities, including collision avoidance based on inte-

grated navigation, tracking of AprilTag, and coordination in two forms. What is worth being mentioned is that the Apriltag is a fuidicial marker created by the April Lab from the University of Michigan.

1.4 Publications

This thesis is based on the following publications.

1.4.1 Journal papers

The published journal papers are listed as follows:

(i)**T. Zhang**, X. Nie , X. Zhu, E. Lim , F. Ma , & L. Yu. (2021). "Improved camshift algorithm in agv vision-based tracking with edge computing", The Journal of Supercomputing, 1-15.

(ii)T. Guo, **T. Zhang**, E. Lim, M. Lpez-Bentez, F. Ma and L. Yu, "A Review of Wavelet Analysis and Its Applications: Challenges and Opportunities", in IEEE Access, vol. 10, pp. 58869-58903, 2022, doi: 10.1109/ACCESS.2022.3179517.

(iii)**T. Zhang**, Y. Song, Z. Kong, T. Guo, M. Lopez-Benitez, E. Lim,F. Ma and L. Yu,"Mobile Robot Tracking with Deep Learning Models under the Specific Environments", Applied Science

The journal papers in preparation are listed as follows:

(i)W. Dai, **T. Zhang**, T.Guo, M. Luo , Y. Lu, L. Yu,"A remote control system for a disinfection robot"

(ii)**T. Zhang**, Y. Huang, Y. Song, F. Yan, E. Lim, M. Lopez-Benitez, F. Ma and L.Yu "Tag visual tracking: Review and experimental comparison"

(iii)**T. Zhang**, Z. Ma, M. Xu, E. Lim, M. Lopez-Benitez, F. Ma and L. Yu "A scheduling optimization method for multi-AGV systems"

1.4.2 Conference papers

The published conference papers are listed as follows:

(i) **T. Zhang**, C. Li, X. Zhu, E. Lim, F. Ma, L. Yu,"A Multi-Robot Planning Algorithm with Quad Tree Map Division for Obstacles of Irregular Shape",Frontiers in Artificial Intelligence and Applications,Volume 347: Design Studies and Intelligence Engineering,page 24-32,DOI:10.3233/FAIA220007.schedule

(ii)**T. Zhang**, Z. Kong, T. Guo, M. Lopez-Benitez, E. Lim, F. Ma and L. Yu., "A real-time visual tracking and distance measuring algorithm based on SSD", 2022 4th International Conference on Advances in Computer Technology, Information Science and Communications (CTISC), 2022, pp. 1-5, doi: 10.1109/CTISC54888.2022.9849795.deep

(iii)**T. Zhang**, W. Zhao, X. Zhu, E. Lim, F. Ma, and L. Yu. 2022. Design of Multi-AGV System with Tracking, Collision Avoidance and Coordination. In 2021 4th International Conference on Algorithms, Computing and Artificial Intelligence (ACAI 21). Association for Computing Machinery, New York, NY, USA, Article 3, 17. https://doi.org/10.1145/3508546.3508549robot

(iv)**T. Zhang**, H. Liu, Y. Zhou, L. Migue, E. Lim, F. Ma and L. Yu, "A network selection optimization algorithm based on AHP and TOPSIS in heterogeneous network environment", 2021 2nd International Conference on Electronics, Communications and Information Technology (CECIT), 2021, pp. 97-100, doi: 10.1109/CECIT53797.2021.00025.heterogeneous network

(v)Y. Zhou, X. Li, Y. Lei, D. Chen, L. Yu,**T. Zhang***,"An Improved Method for Making CNN Immune to Backdoor Attack by Activating Clustering",2022 6th International Symposium on Computer Science and Intelligent Control (ISCSIC 2022)

(vi)**T. Zhang**, Y. Wan, M. Lopez-Benitez, E. Lim, F. Ma and L. Yu, "Big data encrypting transmission framework for a multi-AGV system",2022 Theme: Cyber Computing of AI/ML, Cybersecurity, Image Processing, and Beyond 5G(14th CyberC 2022)

1.4.3 Chinese patent

(i)AGV control method and device, application publication No.: CN11309014A Application Publication Date: June 19, 2020 Application No.: 2020101114197 Application date: February 8th, 2020 Applicant: Xijiao Liverpool University Inventor: Yu Limin; **Zhang Tongpo**; Nie Xiaokai; Ma Fei; Ma Boli

(ii)AGV control method and device, application publication No.: CN111650938A Application Publication Date: September 11, 2020 Application No.: 2020105117214 Application date: June 8th, 2020 Applicant: Xijiao Liverpool University Inventor: Yu Limin; **Zhang Tongpo**; Nie Xiaokai; Ma Fei; Ma Boli

Chapter 2

Literature review

The key factors of the project are Automated Guided Vehicles, target tracking algorithm, scheduling optimisation and coordination of AGVs and the communication environment of a multi-AGV system. An AGV prototype in this project means an unmanned automated vehicle equipped with automatic guidance equipment, traveling along a planned path, powered by batteries, and equipped with safety protection and various sensors. The target tracking algorithm is based on computer vision in this project. The current AGV mainstream is based on Electromagnetic guidance, Magnetic tape guidance, QR Code Guidance and Laser Navigation. The incorporation of vision sensors into AGVs is a popular aspect at present. However, the cost of the whole AGV system will increase after adding vision processing technology. How to lighten the process of vision processing is the focus of this project. Scheduling optimisation and coordination of AGVs are based on combining and improving path planning algorithms, obstacle avoidance algorithms and scheduling rules. Furthermore, communication means the communication environment of a multi-AGV system. How to improve its safety and efficiency is also the focus of this project.

The challenges faced by this project contain building several AGV prototypes and three aspects which are communication between AGVs, AGVs scheduling and computer vision technology applied in AGVs.
In the aspect of Communication between AGVs, the key challenges are as follows:
(1)Establishing an optimized network selection algorithm.

(2)Establishing an improved method for preventing CNNs from backdoor attacks.
In the aspect of AGVs scheduling, the key challenges are as follows:
(1)Establishing a transmission framework for a multi-AGV system.
(2)Creating a multi-robot path planning algorithm.
(3)Establishing a platform that can input parameters and algorithms to calculate a multi-AGV system's time cost.
In the aspect of computer vision technology applied in AGVs, the key challenges are as follows:
(1)Creating an improved Camshift algorthm .
(2)Realizing different deep learning visual tracking models.

2.1 A network selection optimization algorithm based on AHP and TOPSIS in heterogeneous network environment

The heterogeneous network environment integrates a variety of different wireless access technologies, each of which has different functions to meet the needs of mobile users [1]. As far as next-generation wireless networks are concerned, heterogeneous networks are the latest ones to appear. Heterogeneous networks are able to deal with multiple wireless access technologies composed of existing networks and follow-up networks to cope with the various demands of real-time applications. In terms of QoS, price, and coverage space, all available wireless networks included in heterogeneous networks have their own characteristics. Sensors or other devices can be easily connected with different networks, which is the advantage of heterogeneous networks [2].

The handover problem in a heterogeneous network environment is a very popular aspect. Existing network selection algorithms can be generally divided into two categories: call access network selection algorithm or vertical network handover selection algorithm. The access and handover in a homogeneous network environment mainly consider the received signal strength (RSS). The classic handover solution selects the network based on the RSS parameter[3]. In [4], the method of network switching mainly depends on the RSS

parameters, and the network switching will be performed when the RSS reaches certain specific values. The disadvantage of the method is that it ignores the influence of other parameters during the progress of selection. Most of the algorithms used consider multiple types of parameters. The network selection algorithm comprehensively analyzes various parameters, sorts multiple networks and selects the best network to connect [3, 5]. In addition, most vertical network handover selection algorithms are considered as multiple attribute decision making (MADM) problems to be solved .

Furthermore, with the continuous development of mobile communication technology, more and more wireless communication technology standards are introduced into the communication industry. The integration of different access technologies has been recognized as the development trend of communication networks in the industry. Therefore, network selection algorithm in a heterogeneous network environment has become a research hot spot in the communication field. The Ph.D. project in this aspect aims at creating a network selection optimization algorithm based on AHP and TOPSIS to select the best access network in a heterogeneous wireless network environment. The algorithm also selects different parameters and weights to construct a new network evaluation framework, which can meet users' customization requirements for different heterogeneous network application scenarios. The prerequisites will also be set. Among many parameter indicators, RSS reflects the connection status of the device and the network. If the RSS is below the limit value, it may cause unstable communication or even communication interruption. To solve this problem, RSS is set as a prerequisite for the device to use the network selection algorithm for network selection. In summary, the network selection optimization algorithm this section designed is based on the AHP weight calculation method, Grey Relational Analysis (GRA), TOPSIS, and other network sorting algorithms to select the optimal network.

Many factors should be considered in a heterogeneous network which includes signal strength, data transmission rate, price, coverage, real-time, user mobility, system throughput, blocking rate, and a balanced load. Based on the related review work, Multiple Attribute Decision Making (MADM) should be the most suitable method to deal with these factors [6]. Some classical MADM algorithms have been reviewed, Simple Additive Weighting (SAW), Analytic Hierarchy Process [8], Technique for Order Preference

by Similarity to an Ideal Solution (TOPSIS) [9], Grey Relational Analysis (GRA) [10], Elimination and Choice Translating Reality (ELECTRE) [11] and VlseKriterijumska Optimizacija I Kompromisno Resenje method (VIKOR) [12]. In addition, Markov Decision Process (MDP) can also be used as a mathematical tool to solve network choice decisions. Sgora [13] provides a network selection algorithm that combines AHP to establish the weight factors of parameters and TOPSIS to rank the candidate networks. Chen and Li [3] propose a vertical handoff algorithm that combined MDP with AHP and aims to maximize the expected total payoff over the course of transmission. These methods' advantages and disadvantages can be seen in table 2.1: Algorithm Comparison.

Table 2.1: Algorithm Comparison

Algorithm Type	**Advantages**	**Disadvantages**
SAW	Simple to implement, simple steps, the base of following methods	The parameter index given by the user does not have professional theoretical support and is highly subjective
AHP	Low computational complexity, fewer vertical switches	The assignment of each parameter in the judgment matrix is very arbitrary and has a long cycle and high subjectivity.
TOPSIS	Minimal computational complexity,	The selection of the corresponding quantitative indicators will be difficult
GRA	Higher bandwidth and lower latency	It is required that the optimal value of each index needs to be determined currently, and it is difficult to determine the optimal value of some indexes.
MDP/SMDP	Fewer vertical switches, effectively improve blocking rate and packet loss rate	All information should be collected.

Analytical Hierarchy Process (AHP) is a strategy for solving complex problems with multiple factors. Compared with other weight calculation methods, the analytic hierarchy process has a very unique function. It provides a special mathematical logic algorithm which can calculate the relative importance of many parameters or attributes that affect the target result, and assign them according to relative importance. It analyzes the problem qualitatively and quantitatively, rather than a single qualitative or quantitative analysis. This method combines qualitative analysis and quantitative analysis strategies [14]. It uses the experience of decision makers or scenario design to measure and judge the relative importance of different standard parameters that affect the realization of the goal, and to give each impact factor weight according to the application conditions, using parameters and weights to calculate the pros and cons of each solution and give the order of pros and

cons of each solution [15]. This method can effectively solve problems that are difficult to analyze with a single quantitative analysis method or a qualitative analysis method [16]. However, in this project, we will automatically adjust the weights of various factors in the network selection algorithm according to the equipment itself. For example, when the battery power of mobile devices is below a certain level, the algorithm will automatically increase the weight of the power consumption parameter. It will choose a network that tends to reduce battery consumption and has a higher score for the device to connect to.

The full name of TOPSIS is a technique for order preference by similarity to an ideal solution which means a sorting method that is close to the ideal value. According to multiple indicators, multiple options are compared and selected. The central idea of this method is to first determine the positive ideal value and the negative ideal value of each indicator. It can fully reflect the gap between the various programs, objectively and truly reflect the actual situation, has the advantages of being true, intuitive, and reliable, and has no special requirements for its sample data. TOPSIS is able to reflect the overall situation and has universal applicability.

2.2 An improved method for making CNN immune to backdoor attack

Deep neural networks have received extensive attention and research due to their excellent performance, and have already made a splash in the industry. Advanced deep neural networks are considered to be an enabler of technological progress, such as computer vision (CV) [17], natural language processing (NLP) [18], and speech recognition (SR) [19]. However, deep neural networks also need to defend against possible adversaries which are the same as other computer-related industries.

Currently, a type of attack called data poisoning [20] has been widely used to attack machine learning models. In addition to being vulnerable to adversarial examples in the prediction stage, the training process of machine learning models may also be attacked by attackers. In particular, if a machine learning model is trained on data from potentially untrustworthy sources, it is easy for an attacker to manipulate the training data distribution

by inserting carefully crafted samples into the training set to change the model behavior and degrade the model performance.

In general, poison attacks can be divided into 2 areas. One is collision attacks, and the other is backdoor attacks. Recently, the Backdoor Attack or Trojan Attack [21], has received a lot of attention from the academic community. Unlike traditional data poisoning, attackers poison the training data precisely in order to enhance the stealth of backdoor attacks. The result is only a small change in the accuracy of the model (from 99% to 98%), a subtle difference that does not allow defenders to be more vigilant. In fact, when data comes with an attacker's trigger, the model misclassifies that data into the attacker's specified category [22]. In a presentation at the 2021 HITCON security conference, Cheng and Tseng showed that backdoor code could completely bypass defenses by poisoning less than 0.7% of the data submitted to a machine learning system [23]. This method not only demonstrates that even a small number of malicious samples can be used to complete an attack but also shows that machine learning systems are vulnerable even using a small amount of undetected open source data. Malicious attacks on deep learning networks can pose significant risks to users. Research has shown that due to the black-box feature of deep neural networks during training, attacks are not able to easily be detected and corrected. Existing attacks include adversarial examples, universal adversarial patch (UAP) [24], and data poisoning. For example, when using convolutional neural networks to classify image content, untrusted datasets may carry toxic data. Typically, an attacker adds a trigger to the image dataset, which can be composed of several anomalous pixels. Training a model with a poisoned dataset not only makes it difficult for the defender to detect that the model has been attacked but also leads to misclassification [25].

This aspect of the project aims at presenting an optimized convolutional neural network (CNN) to immune poisoned data backdoor attacks by using the Activation Clustering algorithm in CNN. Through hyperparameter optimization, the number of weights of the whole model can be reduced to half of the sample model while maintaining the performance of the model. In addition, the model will be trained and tested on poisoned Modified National Institute of Standards and Technology (MNIST) to prove the proposed method is further optimized than other existing defense methods.

Methods for preventing CNN immune from backdoor attacks have been an active area of research for many years and have been tackled from numerous angles. Backdoor attackers usually control the input dataset of Deep neural networks (DNNs) [26]. The backdoor attack flow in DNN can be seen in Figure 2.1. In this way, the backdoor can be activated as the attackers motivation. Similar to a poisoning attack, both attacking methods focus on poisoning the data, which is the source of the training model. However, the poisoning attack is purposeless [27]. It is not clear exactly which class will be misclassified, while a backdoor attack purposefully causes one class to be misclassified into another.

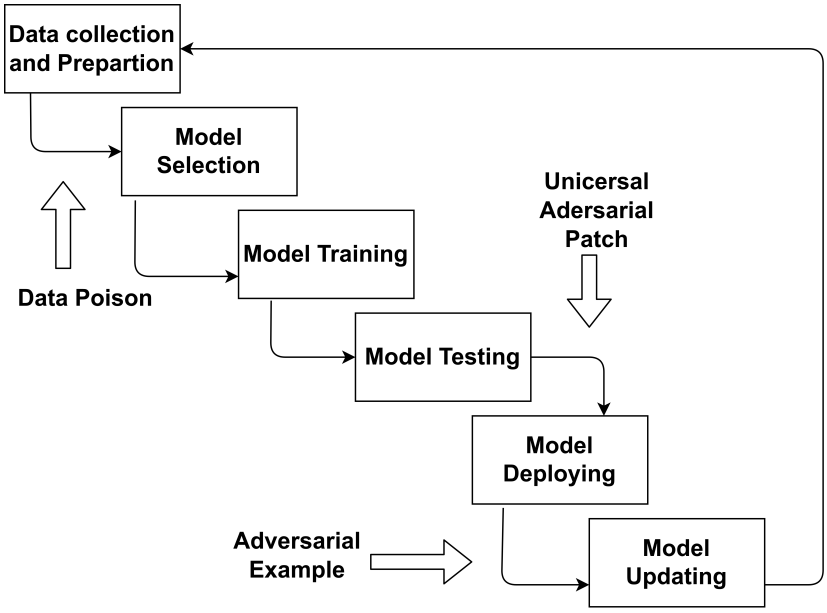

Figure 2.1: Flowchart of Backdoor Attack in DNN[26]

Gu et al. described a generating backdoor attack, which is called as BadNets [28]. By trigger, the out can be classified as 8 with a variance of 1. The advantage of using this method is that in an outsourced training scenario, such a backdoor attack would not be detected by the trainer as a reduction in model accuracy. According to their MNIST attack results, the BadNets' mislabel can be larger than 99% of images with a backdoor trigger successfully. Li described BadNets as the representative of visible attacks [29]. Subsequent attacks have borrowed from this method. In contrast to visible attacks, Chen et al. have

a new idea, to make the attacks invisible [22]. Unlike BadNets, which simply adds pixel dots to the target image, Chen adds a few less visible triggers to the image.

Furthermore, Turner et al. proposed a label-consistent attack [30]. They make the images very blurred by pre-processing. So even though the correct images are correctly classified, when images with triggers can make the model very dependent on the triggers. Based on image steganography [31], Li et al. set a backdoor trigger specific to the sample. Steganography can prevent defenders from finding backdoor triggers by examining similar anomalous samples. Since each poisoned sample is steganography with the same data, no two triggers are identical.

Many methods have been devised to identify backdoor assaults, which are becoming increasingly difficult to detect. Liu [32] proposed early on that the effects of backdoors could be eliminated with a combination of pruning and fine-tuning. The method attempts to prune dormant neurons starting from those that are not activated by the backdoor at clean input. Afterward for the model, fine-tuning is performed. At this point, the backdoor has been eliminated, while the fine-tuning restores some of the accuracy loss.

The preparation protection against backdoor assaults was introduced by Liu et al. [33]. They filter the input before it reaches the poisoned classification models, removing the backdoor. However, a large-scale dataset that can be trusted and verified is required, which is a difficult test to do during the modelling. Li et al. reduced the aggressiveness of backdoors by reducing the chance of a trigger [34]. By changing their spatial location, a set of spatial proposeations, for example, the value of pixel values, the result is more efficient. This method demonstrates that the effect of backdoor attacks is often weak when the training triggers are in a different position to the ones used for testing. In addition, to determine whether the backdoor exists, Wang et al. proposed a technique for digging hidden triggers, with 3 methods to mitigate the fluence of backdoors [35], which is Neutral Cleanse (NC).

Researchers tried to explain the reason for backdoor attacks and detect the backdoor attack. K.Groose et al. proposed Backdoor Smoothing, which is when the trigger is successfully implanted, the smoothness increases significantly. However, such an inference is not certain, i.e., other means of causing smoothing may also exist [36]. Based on Random-

ized smoothing (RS) [37], a technique that makes classifiers robust enough to withstand adversarial attacks, Wang et.al applied RS and proved that RS was feasible for backdoor defense. By adding noise, they counteract the perturbations caused by the triggers [38].

Based on those methods, in this aspect, we want to propose a new approach to detecting backdoor attacks and resisting backdoor triggers. Compared to the previous work, this method focuses on increasing test accuracy, reducing the model's size, and decreasing the model's complexity.

2.3 A transmission framework for a multi-AGV system

A multi-AGV system has been widely applied in industrial areas. The system contains many different sensors and chips. A transmission framework should be established. To establish such framework, several points should be considered. First, messages collected or processed by those sensors and chips should be transmitted to a central platform or a cloud platform to process. Furthermore, the security of the communication environment should also be considered. To solve the problems mentioned above, we assume that there is a multi-AGV system that performs a series of tasks in a known environment. There are multiple task target points in the scene, a starting point, and a task list. After accepting the task, each AGV starts from the starting point and moves to the task target point at a certain interval. After each AGV reaches the target point of the task, it completes the task and returns to the starting point. On the way, each AGV collects information in the environment through a variety of different sensors equipped on itself. The data collected by sensors equipped in AGV will be transmitted to the central platform or the cloud platform depending on the specific situation. Messages processed by the AGVs main chip will also be transmitted to the central platform or the cloud platform.

Based on the situations mentioned above, we establish a transmission framework for a multi-AGV system based on federated deep learning. This framework is created to improve a multi-AGV systems communication environment. Based on federated deep learning (FL) model, this framework simulates the scenario of transmission between the edge computer and the main server by realizing encrypted message transmission between

multiple computers and different sensors[39].

The federated deep learning model can be applied in different areas. Ahmed Imteaj and M. Hadi Amini proposed an FL model by monitoring client activities and leveraging available local computing resources, particularly for resource-constrained IoT devices (e.g., mobile robots), to accelerate the learning process and finally successfully avoid the inconsistent and inadequate resource clients from the training process [41]. Wei Lius team proposed a new federated learning design of Momentum Federated Learning (MFL) converges faster than FL for different machine learning models[42]. M. Gharibi and P. Rao proposed a refining algorithm called RefinedFed, to eliminate corrupted, low accuracy, and noisy models that can negatively impact the centralized model by reducing its accuracy or cause other malicious activities[43]. Sherif B. Azmys team applied a federated deep learning model to an UAV system to improve the systems efficiency [44]. Mohamed Amine Ferrags team presents a comprehensive study with an experimental analysis of federated deep learning approaches for cyber security in the Internet of Things (IoT) applications [45].

RSA algorithm is proposed by Ron Rivest, Adi Shamir and Leonard Adleman in 1977 [46]. It has a long history and is still regarded as the most popular public-key cryptography. Denning E Dorothy describes the attack with the RSA cryptosystem and shows how it generalizes to other public-key systems[47]. Ming-Der Shiehs team presents a new modular exponentiation architecture with a unified modular multiplication or square module and show how to reduce the number of input operands for the CSA tree by mathematical manipulation for RSA Cryptosystem[48]. Fu Mos team presents an improved RSA cryptosystem based on chaotic synchronization which not only retains RSA's security features, but it can also significantly promote the security of traditional RSA cryptosystem [49]. Raza Imams team summarizes and organizes recent literature results in a way that integrates and adds understanding to the work in the RSA field. They also emphasizes the classification of the existing literature to develop a perspective on the certain dominant RSA areas and domain and assess the research trends following the standard SLR methodology[50].

The designed transmission framework for a multi-AGV system based on federated deep

learning can be applied to edge devices, train the local data set with the model sent by the master server, encrypt the uploaded updated parameters, and send the updated model after the master server accepts the parameters.

2.4 A multi-robot path-planning algorithm with quad tree map division for obstacles

A multi-robot system can generally be divided into two types, centralized control architecture, and decentralized control architecture. For the centralized control architecture, there is a central control platform to convey useful information to each robot such as the moving state of the robot, map information, and mark information, and it needs to carry out the algorithm for all robots [51]. Hence, the centralized control architecture has the properties of stability, simple realization, low efficiency, and high requirement of communication. This architecture can be classified as coupled and decoupled. For the coupled architecture[52, 53], it regards all robots as the same one and carries out the same algorithm with considering the whole state and parameters, which means that its computation volume will be huge if there are multi-robots in the map. In contrast, the decoupled architecture does not consider all information but uses tricks to deal with the coming conflicts or deadlocks. To reduce time complexity, this architecture needs to plan paths and coordinate movements in real-time. An apparent disadvantage of it is that it cannot consider the ideal optimal solution for the system. Regarding movement coordination modes, there are region control, time window, and multi-agent system. This paper designs an algorithm to construct the shortest path for each robot and combines the waiting mode and collision avoidance algorithm to ensure there is no conflict in simulated maps.

For a large number of robot systems, the coupled architecture is not feasible of sustainable tasks because of the huge computation content. Thus, this aspect emphasizes the decoupled architecture, which means each robot needs to compute the optimal path before moving and then implement a waiting or collision avoidance algorithm. For a path planning algorithm, in the first step, it needs to examine the environment parameters to construct a node-obstacle distribution graph . Then, it uses an efficient shortest path

algorithm to avoid obstacles [54]. In this aspect, the algorithm adopts the quad tree to construct the lattice for the map. Unlike the general lattice map, using a quad tree can quickly locate a precise position and adjust the resolution more simply. It will also decrease time to compute the path planning and collision avoidance. To effectively plan the shortest path and coordinate every robot to avoid collision, using decoupled architecture will be more suitable for the combination of waiting mode and avoidance collision algorithm. [55] has proposed a two-layer architecture to control the whole movement and local avoidance. They are macro level and micro level respectively. This architecture strength is that it distributes the calculated amount to different parts of the system. There are several other decoupled architectures with similar technology to deal with different assumed situations appearing in the [56, 57]. In addition, to obtain dynamic information in real time, robot can communicate with neighbors in the specified region [58]. The advantage containss that it has high stability, strong expandability and powerful flexibility. [59] has proposed a strategy that multi-robot system combines the guidance algorithm and disperses policy decision. For a scheduling algorithm, one of the most important points is to coordinate robots to avoid collisions on the map. Those solutions can be classified into two types, waiting mode and motion coordination. To combine the stability of waiting mode and efficiency of coordination motion, this paper aims proposing a decentralized algorithm to solve collision avoidance problem of multi-robot system in a simulated map with polygon obstacles. In previous algorithms, some researchers did not consider the obstacles when dealing with collision avoidance.

In this aspect, the designed algorithm uses quad tree to mark the map and obstacles. The robot only communicates with others in a certain region based on wanted accuracy, which reduces the computation complexity. Furthermore, there is a central control platform to observe global information which has access to collect the data from multi-robot to avoid the deadlock problem, which cannot be solved in [51].

2.5 A scheduling optimization method for multi-AGV systems

Multi-AGV systems are the most commonly used multi-robot system in different industrial scenes. However, how many AGVs should be used in a certain scene can make the system has the highest efficiency, and what method should be used to make the system has the highest efficiency challenges to establish a multi-AGV system.

There are many precedents for scheduling optimization methods, but most of the solutions are for the independent tasks of a single AGV, and little attention has been paid to the mutual assistance problem in AGV cluster scheduling. For example, when vehicles cooperate with each other, so as to realize the mutual transfer of goods and other operations. In fact, if the cooperative worlds of AGVs are not considered, it will lead to a phenomenon. This phenomenon is that although methods used in the system, the actual motion path, and task assignment of each AGV has been optimized as much as possible, idle AGVs that have already completed their tasks will become ineffective resources. They will occupy space and time in the whole system which will decrease the system's performance. Therefore, improvements to AGV scheduling systems have become a focus of research. In terms of the overall architecture of the AGV scheduling system, J. Zajc proposed in 2021 to divide the traffic network into non-overlapping regions and to analyse it as a discrete event system [60]. In other words, this approach allows the system to consider the global problem by successively solving local problems. In contrast, M. De Ryck argues that most current systems that work centrally, lack flexibility, robustness, and scalability[61]. There is an increasing amount of research proposing the decentralization of AGV systems, and decentralized systems should be the future trend. A scheduling system architecture that can be adapted to any practical situation in the relevant field should be a potential topic that needs to be explored for a long time.

From an algorithmic perspective, scheduling optimization of multi-AGV systems broadly consists of three areas, namely task allocation, path planning, and collision avoidance. There are many precedents for research in these three areas. For task allocation, [62] studied the dual-resource integrated scheduling problem of AGVs and machines in an intelligent

manufacturing shop floor environment, taking into account the constraints of machines, workpieces and AGV. [63] and [64] proposed a decentralised solution to achieve flexible and self-organised AGV task allocation. G. Li's team developed a multi-objective mathematical model for AGV scheduling [65]. Y. Bao designed an auction-based heuristic [66] and experimentally verified that a hybrid allocation rule for AGV and tasks can also be a better choice, as opposed to sequential allocation. In his latest research in 2022, N. Fazal proposed a task allocation technique based on threshold levels [67]. Its use of the threshold level as a reference for task acceptance, the scheme has led to a significant improvement in the resource utilisation of the system. For path planning, Y. Lian proposed an improved heuristic path planning algorithm that converts energy consumption into the overhead of the path planner [68]. M. Majdi's team used fuzzy control techniques to navigate multi-AGVs in an unknown environment [69] . D. Yu developed a mixed integer planning model for multi-AGV systems in path planning [70], which has the advantage that it can be transmitted into a series of sub-problems before solving them one by one under certain conditions, and adds penalty terms to the algorithm for generating paths. [71] proposed an integrated approach based on a two-layer control architecture and automatic algorithms. As well as some classical algorithms such as A* Algorithm, Dijkstra Algorithm, Ant Colony Algorithm [72–74] and so on. For collision avoidance, its usually considered together with the design of the path algorithm. The basic idea is to avoid or give way in advance, for example X. Liyun designed collision avoidance factors to guide AGV to avoid collisions [75]. However, due to the AGV's own limited sensors and actuators, there are many unexpected contingencies in practice. To address this direction, J. Luo proposed a design method for a maximum permissible (optimal) controller based on Petri nets (PNs) [76], using tags to label indistinguishable events. In this way, with the help of the PN model, the collision-free problem is formalised as a combination of linear constraints, significantly reducing the computational overhead due to uncontrollable events.

There are two main aspects in a multi-AGV system established in a known environment with established paths. One research aspect is dispatching strategy. The other research aspect is algorithms which contain path planning algorithm and obstacle avoidance algorithm. The AGV dispatching strategy is to solve the problem of assigning loads to vehicle

or vehicle to load when they are available. M.Kzls team propose a dispatching rule that maintains the system operational at all times [77]. Currently, AGV dispatching strategy could be one AGV is chosen to respond work center's service when only one work center applies service and other AGVs are free. The AGV's dispatching strategy could also be a work center serviced by an AGV when only one AGV and more work center apply service at one time [78]. P. Egbelus team solve the work centre initiated task assignment dispatching problems and vehicle initiated task assignment dispatching problems [79]. Egbelu P J presented an algorithm that assigns load movement priority based on the demand states of the load destinations [80]. Co C Gs team outlines the vehicle management control strategies discussed in previous studies on AGVs in 1991 [81]. Pyung-Hoi Koos team presented a vehicle dispatching procedure based on the concept of theory of constraints, in which vehicle dispatching decisions are made to utilize the bottleneck machines at the maximum level [82]. Marvizadeh and Choobineh presented the algorithm with the objective of load balancing among the factory work centres by choosing the dispatch decision, which contributes the most to the laminar flow of the factory [83].

Algorithms applied in multi-AGV system contain path planning algorithms and obstacle avoidance algorithms. Those algorithms are also popular among other robotic areas. Traditional path planning and obstacle avoidance algorithms contain Dijkstra algorithm, A* algorithm, D* algorithm and artificial potential field.

In summary, the multi-AGV systems scheduling optimization method is a method that includes AGV's dispatching strategy, path planning algorithm, obstacle avoidance algorithm, and mechanism to optimize the whole multi-AGV system. The focus is on finding a way to improve a specific multi-AGV system's performance in a known environment with established paths.

2.6 Traditional visual tracking method

Mean shift algorithm has been widely used as a light-weighted target tracking algorithm with density-based clustering [84]. As an improvement in the mean shift algorithm for target tracking, Camshift (Continuously Adaptive Mean Shift) algorithm updates the

search window size according to the target's size and accurately locates the target with size change [85]. It has been used in the area of human motion pursuing and face tracking. However, when calculating the histogram distribution of target template, Camshift algorithm does not use kernel function for weighting, which would result in every pixel in the target area having the same weight in the target model [86]. Therefore, the anti-noise ability of Cam-shift algorithm is lower than that of mean shift tracking algorithm. At the same time, Cam-shift algorithm does not define candidate targets and directly uses the target template for tracking [87]. In addition, Camshift algorithm refers to the Hue component of the HSV color space, which depends only on the color information in the target tracking. In this case, when the target and the background colors are close, Camshift will be automatically executed. The tracking window will be expanded and sometimes even exceeds the whole video tracking window size, leading to inaccurate target positioning, continuous tracking error and the loss of the target [88]. Meanwhile, how to implement Camshift algorithm in a multi-AGV efficiently requires careful consideration.

The original Camshift (Continuously Adaptive Mean Shift algorithm) is reviewed in this section. Camshift is an improvement on the Mean Shift algorithm. It updates the search window size according to the target size, and accurately identify the target with size change [89]. Meanwhile, it is also able to track the target changing size in video streaming. The basic principle is related to the color information of the moving item in each video frame. This information works as characteristics of the input image to execute Mean Shift operation in every input image respectively. After that, the target center and search window size of the previous frame are taken as the initial values of Mean shift algorithm of next frame. In this way, continuous target tracking can be realized. Because the position and size of the search window are close to previous state, and the moving target is usually near this area, the search time is shortened. Therefore, if Camshift algorithm is used on two sets of pictures, it obtains two sets of corresponding center points. Theoretically, these points are at the same location in the world coordinate system. That means these two sets of points could be used to calculate the space coordinate [90].

The Camshift algorithm adopts the HSV (hue, saturation, and value) color mode instead of the RGB model with primary colors of R (red), G (green) and B (blue). With

a RGB color model, morphological characteristics of the images cannot be reflected fully, which only shows deployment of color from the principle of optics [91]. The reason why Camshift algorithm converts the image from the RGB color space to the HSV is that the HSV color model is less affected by changes in the illumination intensity than the RGB color model [92]. H component is used in the algorithm to set up a histogram model of the tracking target. It normalizes the obtained result to 0255. By replacing the value of each pixel in the image with the probability pair in which the color appears, the color probability distribution map is produced. An appropriate threshold is then set to binarize the color probability map.

By iterations, the Camshift algorithm updates the search window size based on the orientation [93–95]. The direction information of a target is reflected by the second moments M_{11}, M_{20} and M_{02}, of the search window as formulated below:

$$M_{11} = \sum_{x} \sum_{y} I\left(x, y\right) xy \tag{2.1}$$

$$M_{20} = \sum_{x} \sum_{y} I\left(x, y\right) x^2 \tag{2.2}$$

$$M_{02} = \sum_{x} \sum_{y} I\left(x, y\right) y^2 \tag{2.3}$$

$I\left(x, y\right)$ refers to the pixel value of an image at (x, y).

By defining the intermediate variables a, b and c, it is able to calculate the length and width of the next frame search window in a closed form and they are given as

$$a = \frac{M_{20}}{M_{00}} - x_c^2 \tag{2.4}$$

$$b = 2\left(\frac{M_{20}}{M_{00}} - x_c y_c\right) \tag{2.5}$$

$$c = \frac{M_{02}}{M_{00}} - y_c^2 \tag{2.6}$$

The direction angle θ, length L, and width W are computed as:

$$\theta = \frac{1}{2} tan^{-1} \frac{b}{a-c} \tag{2.7}$$

$$L = \sqrt{\frac{(a+c) + \sqrt{b^2 + (a-c)^2}}{2}} \tag{2.8}$$

$$W = \sqrt{\frac{(a+c) - \sqrt{b^2 + (a-c)^2}}{2}} \tag{2.9}$$

The initial size of the search window for the next frame is given as [19, 20, 21, 22]:

$$width = max|cos]theta, wsin\theta| \tag{2.10}$$

$$Height = max|sin\theta, wcos\theta| \tag{2.11}$$

Camshift algorithm improves the second defect of mean shift tracking algorithm. During the period of tracking, it updates the search window size according to the target size, and accurately locates the target with size change [89]. However, when calculating the histogram distribution of the target template, Camshfit algorithm does not use kernel function for weighting, which would result in every pixel in the target area having the same weight in the target model. Therefore, the anti-noise ability of CamShfit algorithm is lower than that of mean shift tracking algorithm. At the same time, Camshift algorithm does not define candidate targets. It directly uses the target template for tracking. In addition, Camshift algorithm utilizes H component of the HSV color space and target histogram framework, which heavily relies on the color of the target information in tracking. In this case, when the target color and background color are close, Camshift will automatically be executed and the tracking window will be expanded. Sometimes this expanded tracking window may exceed the whole video frame size, leading to inaccurate target positioning, continuous tracking errors and the loss of the target.

2.7 Deep learning visual tracking models

With the development of modern manufacturing, the industrial environment is becoming more dynamic and complex. The deployment of mobile robots in different industrial areas emphasizes the tracking of robot targets as one of the important functions for monitoring and coordination in an unknown environment. Different techniques have been proposed for robot tracking.

The camera lens distortion correction and non-uniform light compensation techniques, a modified non-linear state estimator and an improved Matrox Meteor Frame-grabber were proposed around the year 2000 in the field of mobile robot tracking [96–98]. Monitoring objects against the cluttered moving background and dynamic objects/intrusions moving through the scene are huge challenges in partial occlusion. Hongxin Wang's team [99] proposes a novel visual system model (STMD+) for small target motion detection to discriminate small targets from small target-like background features (named fake features). A novel technique for background subtraction based on the dynamic autoregressive moving average (ARMA) model is proposed by JIAN LI's team [100]. A motion-planning framework for urban autonomous driving at uncontrolled intersections is presented by Jeong and Yi [101]. However, the processing speed and anti-blocking properties of these deep learning (DL) methods have not been paid enough attention in the mobile robot tracking area.

To solve the underlying optimization problems, neural network (NN) based models, especially convolutional neural network (CNN) models, recurrent neural network (RNN) and graph neural network (GNN) models, can be adopted [102]. Most of the research approaches with NN-based methodologies in target tracking problems have focused on solving the filtering in target tracking, target drift and target loss and improving the tracking precision, especially in low-quality videos [103–105]. In a dynamic scenario, we only have the knowledge of the robot we want to track initially. The data set of the target robot in the indoor and outdoor environment may be collected for training. The robot will then be deployed in an unknown environment. However, during the manoeuvre of the robots, fast target motion and unknown interference items present in the scene might cause severe occlusion of the robot target. This imposes the major challenge in NN-based target tracking.

Various solutions have been proposed to resolve the occlusion problem. Wang and Yuille [106] designed an end-to-end deep occlusion network (DOC) for estimating occlusion relations which enables better training and testing of deep networks for occlusion estimation. Ren et al. [107] developed a computational model for figure/ground assignment in a complex natural scene. The model was provided with a feasible approach to bottom-up figure/ground assignment in natural images. Hoiem et al. [108] proposed an approach to recover the occlusion boundaries and depth ordering of free-standing structures in the scene. The research pointed out a research direction for single-image occlusion reasoning and the broader 3D scene understanding problem. Kotsia et al. [109] addressed the effect of partial occlusion on facial expression recognition. They analyzed the way in which partial occlusion affects the recognition of facial expressions by human observers. Weiwei Zhang et al. [110] developed a novel vehicle detection based on vehicle part-based proposals generation and Part Affinity Fields (PAFs) based combination algorithm to solve the visual detection problem of occlusion in complex backgrounds. The designed algorithm improves the performance of accuracy, especially when vehicles are heavily occluded. Yuille and Liu [111] provided a review of the strengths and weaknesses of Deep Nets for vision from 2018 to 2020. It was proven that even though real-world image sets are infinitely large and any data set (no matter how big) is hardly representative of real-world complexity, using a DL model for visual problems can solve specific visual tasks and has the potential for practical applications.

In view of the previous research, the following research questions are addressed: 1) whether deep learning model tracking methods, Faster RCNN, SSD and YOLOv5, can adapt to various occlusion conditions and 2) to what extent occlusion could be handled in a specific environment. To address these issues, a training data set of the robot target under designated mobility and occlusion conditions is needed to be constructed in our research. Different testing data sets are also created. We then focus on three recent versions of DL models, You Only Look Once (YOLO) model - YOLOv5, Faster Region Proposal Convolutional Neural Network (R-CNN), and Single Shot MultiBox Detector (SSD), and compare the performance of the three DL models in the robot target tracking. A Random Erasing method is proposed to generate some of the testing data set to mimic the random

interference items that block the view of the target. Different dynamic occlusion variation situations have also been tested where part of the targets is out of range. The missing part is about 10% to 30% of the target. The environment contains indoor and outdoor environments with different backgrounds.

The contributions in this aspect of my porject are summed up as follows: 1) Three evaluation data sets for training and algorithm verification under single target tracking scenarios were built. 2) Three recent object detection models were contructed, which are Faster RCNN, SSD and YOLOv5. Comprehensive experiments are conducted to evaluate the detection performance with our data sets. 3) A Random Erasing method was designed to create new testing data set with various partial occlusions to verify the system performance. 4) Image views from different angles, distances and surrounding environments to test the model performance. 5) We analyze numerically the adaptivity of the models against different degrees of occlusions through experiments.

The main DL algorithms for target detection are CNN-based models, and can be classified according to the number of stages: one or two stages. The two-stage approach contains the R-CNN algorithm, and the one-stage approach refers to YOLO or SSD. For the two-stage method, training data need to be generated in advance to obtain sparse data. These training candidate data are then well-tuned with classification and regression. The advantage of the two-stage method is its high accuracy. The one-stage method involves conducting dense sampling uniformly at different positions of the picture. Convolutional Neural Network (CNN) is a mathematical or computational model that mimics the structure and function of biological neural networks [112]. In object detection, the image needs to be divided into multiple regions to detect objects in this area. Therefore, CNN is introduced to select regions. Region proposals Convolutional Neural Network (RCNN) is a model that uses CNN to improve the effect of target detection. The Faster RCNN is a single, unified network for object detection composed of two modules. One is a deep fully convolutional neural network that generates candidate boxes. The other is the Faster RCNN detector based on the proposed regions. Both the SDD model and YOLOv5 model are deep learning models for target detection based on CNN[113, 114, 121]. Different scales and aspect ratios can be used for sampling. After that, CNN can be used to extract

features and directly carry out classification and regression. The whole process only needs one step. Therefore, the advantage of the one-stage method is the fast spread. However, an important disadvantage of uniform dense sampling is that it is difficult to train. This section describes the background of deep learning technologies used in this research.

The Faster RCNN model, SSD and YOLOv5 models are selected because they are the most recognized and novel deep learning models, which can be applied to visual tracking. In 2015, the Faster RCNN model won many firsts in ILSVRV and COCO competitions. The algorithm is initially based on a Faster RCNN and proposes a region proposal network (RPN) candidate box generation algorithm, which greatly improves the speed of target detection. The algorithm proposed by Girshick [113, 116] improves the comprehensive performance drastically, especially in terms of detection speed. The SSD model has the advantage in processing speed because the object classification and prediction anchor regression are done simultaneously. It utilizes CNN to extract features and densely and uniformly samples the feature map at different locations with different scales. In 2020, the Ultralytics released YOLOv5 [117–120, 122]. YOLO redefines object detection as a regression problem and applies a single convolutional neural network (CNN) to the entire image. Images are divided into grids, and the class probabilities and bounding anchors of each grid are predicted. Different from the previous version - YOLOv3 and YOLOv4, YOLOv5 can predict across layers, and it is still being updated by Ultralytics.

2.7.1 Faster RCNN

The Faster RCNN is a single, unified network for object detection. It is composed of two modules. One is a deep fully convolutional neural network that generates candidate boxes. The other is the Faster RCNN detector based on the proposed regions [113, 116]. Mai et al. [123] presented a novel Faster RCNN model with classifier fusion to automatically detect small fruits. Nsaif et al. [124] applied a cascading Faster RCNN with a Gabor filter and a nave Bayes model to increase the precision of eye detection under conditions of reflection from glasses or occlusion. In addition, Kim et al. [125] combined Faster RCNN with other models and achieved better performance for the detection of vehicles and pedestrians than conventional vision-based methods. The division of labor between the two stages of the

Faster RCNN is clear, which brings about the improvement of accuracy, but the speed is relatively slow.

In the implementation of this project, for an arbitrary size image, a reshaping to 600*600 size has been performed and imported into the model. The Resnet-50, a CNN model, includes convolution, Batch norm, Relu, Max pooling and Average pooling layers. It can extract features from the images and output feature maps. The feature maps are shared by the Region Proposal Network (RPN) network and fully-connected network. The RPN network generates proposed anchors directly which increases the speed of the generation of proposed anchors significantly. After 3 convolutions of the feature map, anchors are classified into the foreground (positive) and background (negative) by the softmax classifier. Anchors that have over 0.7 Intersection over Union (IoU) overlap with truth ground anchors are assigned to a positive label while anchors that have less than 0.3 IoU overlap with truth ground anchors are assigned to a negative label. In the other branch, the offset values of anchors from the ground truth anchors are calculated to get an accurate proposal. In the final proposal layer, positive anchors and offset values are combined to achieve the function of object location. The multi-task loss function is as follows.

$$L\left(\{p_i\},\{t_{\{}i\}\}\right)=\frac{1}{N_{cls}}\sum_i L_{cls}\left(p_i,P_i^*\right)+\lambda\frac{1}{N_{reg}}\sum_i p_i^* L_{reg}\left(t_i,t_i^*\right) \tag{2.12}$$

Where i is the index of an anchor in a batch, p_i is the predicted probability of anchor i including an object, t_i is a vector representing coordinates of the predicted bounding box, t_i^* is a vector representing coordinates of a bounding box, L_{cls} is the classification loss over background and foreground, L_{reg} is the regression loss for positive anchors, λ is a constant which is set to 10 in this simulation [124].

The classification stage of the algorithm outputs the probability vector from the feature map through the fully-connected layer and softmax classifier. More accurate target detection anchors from region proposals are then generated at the regression stage. During the training, we calculated the IoU between all proposed anchors and the real grounding box followed by filtering. If the IoU is larger than 0.5, the proposed box is regarded as a positive sample. Otherwise, it is taken as a negative sample.

The training operation is divided into the freeze and unfreeze training. To make use of the GPU memory, the batch size of freeze training is 3. In unfreeze training, the backbone network is unfrozen, and all parameters are trained simultaneously. The batch size is set to 1.

2.7.2 SSD

The SSD model detects objects from the images using a single deep neural network [29]. SSD has been emphasized by many researchers [117–119, 126]. The development procedure would be explained briefly. Wang et al. proposed SSD300 and SSD512, and demonstrated their superiority in image detection [126]. Miao et al. [117] applied SSD to implement the automatic feature learning process on the aerial image set. They pointed out that the model trained by SSD can extract high-level features and improve the detection speed. Yang et al. [118] implemented SSD as the CNN detector with the Kalman filter. The Lightweight Feature Fusion Single Shot Multibox Detector (L-SSD) algorithm was proposed to improve the detection performance in a manual garbage sorting environment [119]. The basic size and shape of the priority anchor in the SSD model cannot be obtained directly through learning, but need to be set manually which may lead to human error.

SSD has the advantage in the detection speed viewpoint because the object classification and prediction anchor regression are implemented simultaneously. SSD utilizes VGG16 to extract features and samples the feature map at different locations with different scales densely and evenly. The weight parameters of VGG16 are trained in advance, so it can be loaded to the network directly. The input image is reshaped to 300300 pixels before importing into the network. Six feature map layers are extracted and default anchors with different scales are constructed on every point of the feature maps. The feature maps of low layers have smaller receptive fields which can detect small objects. The feature maps of high layers are vice versa. The goal of the procedure is to detect objects with multiple scales. The principles of choosing scales and aspects for the default anchor are critical in implementation.

For the detection of the mobile robot, the size of the anchor is set to be $S_k = [30, 60, 111, 162, 213, 264, 315]$ and $k \in [1, 7]$. The unit of the anchor size is pixel. K is

the number of convolution layers. The feature maps of Conv4_3, Conv7, Conv8_2, Conv9_2, Conv10_2 and Conv11_2 layers are extracted and 6 default anchors with different scales are constructed on every point of the feature maps. The feature maps of lower layers have smaller receptive fields which can detect small objects while feature maps of higher layers have larger receptive fields that can detect large objects. The goal of the procedure is to detect objects with multiple scales. The principles of choosing scales and aspects for default anchors are important. Six feature maps have a different number of anchors on every point. For feature maps from Conv4_3, Conv10_2 and Conv11_2 layers, every point has 4 anchors. If the feature map k has 4 anchors, aspect ratios are denoted as $a_r \in \{1, 2, 1/2\}$,$r \in [1, 3]$.

We can compute the width and height of the 4 anchors as:

$$W_k^r = S_k\sqrt{a_r} \tag{2.13}$$

$$h_k^r = \frac{S_k}{\sqrt{a_r}} \tag{2.14}$$

For $a_r = 1$, another square anchor is added whose size length is $\sqrt{S_k S_{k+1}}$. If feature map k has 6 a_r anchors, aspect ratios are denoted as $a_r \in \{1, 2, 1/2, 1/3\}$,$r \in$ [1,5]. In training, the target object, the overall objective loss function is used [19]. The overall objective loss function is a weighted sum of the localization loss of positive samples, the confidence loss of positive samples and the confidence loss of negative samples:

$$L_{total} = \frac{1}{N}\left(L_{pos\_conf+Lpos\_loc+Lneg\_conf}\right) \tag{2.15}$$

N is the number of matched default anchors, L_{total} is the overall objective loss function, L_{pos_conf} is the confidence loss of positive samples, L_{pos_loc} is the localization loss of positive samples, L_{neg_conf} is the confidence loss of negative samples [114]. The possibility of each anchor which does not include objects not being background is calculated and anchors that have the three highest possibilities are selected to be the negative samples because the three anchors are the most difficult to be classified.

2.7.3 YOLO V5

The YOLO model was proposed by Redmon et al. [120] in 2015 for object detection. Later, the Ultralytics proposed YOLOv5 as the modified version, and it is still being updated by them [120–122, 127, 128]. As I do the research, YOLOv5 is the newest version. Currently, it seems have been updated to version 8. The YOLOv5 target detection network has four versions, which are YOLOv5s, YOLOv5m, YOLOv5l and YOLOv5x models. In our experiment, YOLOv5s with the smallest depth and width in the YOLOv5 series are adopted in considering the restriction by hardware for the IoT application scenarios. The network structure of Yolov5 is built according to the one-stage structure. It can be divided into four sections: Input, Backbone, Neck and Prediction. YOLOv5 takes advantage of the Mosaic method to realize data enhancement, which meets the image's arbitrary scaling, clipping and layout requirements from the bottom of the view. The adaptive anchor frame calculation is the initial length and width of various database anchor frames. The YOLO model has a low generalization rate in terms of the aspect ratio of objects which means it is unable to locate objects of unusual proportions. Meanwhile, the YOLO model's inaccurate positioning is also a big problem.

The parameter settings of YOLOv5 focus on two yaml files which are voc_car.yaml and YOLOv5s car.yaml in our experiment. The voc_car.yaml file's function is mainly to declare the type to be recognized and decide the location of the name, training data set and testing data set. The YOLOv5s car.yaml file contains depth_multiple, width_multiple, anchor and head. Depth_multiple controls the number of submodules. Width_multiple controls the number of convolution kernels. The anchor size is set to 640*640. The size of the anchor frame under the image size realizes that large targets can be detected on the small feature map and small targets can also be detected on the large feature map. Heads in YOLOv5 include neck and detect_Head two parts. Neck adopts PANet mechanism. The structure of detecting is the same as that head in YOLOv3. Bottleneckcsp is set to false, which indicates that the residual structure is not used. It uses Conv in the backbone.

As discussed in [129], the three models of Faster RCNN, SSD and YOLO are popular for industrial applications. There was no perfect algorithm or model for all the applications. Careful model selection should be made to fit a typical application [128].

2.8 Real-time visual tracking and distance measuring algorithms

Research on visual tracking and distance measurement in real-time has continued for a long time. The goal of this function is to implement a better and more extensive application in other areas to meet the relative requirement such as helping the navigation system to get more position information, making the robot be able to follow the moving object and asking robots for an analysis of the targets in a video.

There are two major components in the process of real-time visual tracking and distance measurement which include object detection and tracking, distance measuring. Firstly, the tracking target is selected. Then the vision tracking algorithm works and enables a rectangular box to lock the target no matter how the target moves in view. In addition, the rectangular box can change its size according to the size of the target in the view of the screen and re-detect the target if the target is obscured or moves out of the view. Meanwhile, the distance measuring algorithms is able to measure the distance between the camera and the target based on the position of the earth point of the target. While a series of tracking algorithms have been proposed in the past, it remains challenging that traditional tracking algorithms can not address the problems of the deformation, occlusion of targets [145–151]. In this aspect, this project aims at presenting a real-time visual tracking and distance measuring algorithm based on SSD was proposed to solve the problem.

Four traditional visual tracking methods are tested which are Multiple Instance Learning(MIL), kernelized correlation filter(KCF), Tracking-Learning-Detection(TLD) and Minimum Output Sum of Squared Error(MOSSE). MIL can avoid the problem of slight inaccuracies in the tracker leading to incorrectly labeled training examples, which degrades the classifier and can cause further drift. It was created by Boris Babenko's team in 2009 which can lead to a more robust tracker with fewer parameter tweaks [145]. Joao F. Henriques's team derived a KCF with a fast multi-channel extension of linear correlation filters, via a linear kernel, which was called DCF in 2015 [146]. TLD is a tracking framework that explicitly decomposes the long-term tracking task into tracking, learning, and

detection which was proposed by Zdenek Kalal's team. It shows a significant improvement over state-of-the-art approaches [147]. MOSSE is a type of correlation filter created by David S. Bolme's team to produce stable correlation filters when initialized using a single frame [148]. To evaluate the performances of traditional tracking algorithms, the criteria of Speed, Re-detection and Frame anchor adaptation are created. Table 2.2 shows the comparisons of monocular tracking algorithms.

Table 2.2: Comparisons of Monocular tracking algorithm

Name	**Speed**	**Re-detection**	**Frame anchor adaptation**
MIL	15	No	No
KCF	220	Good for shielding Bad for out of view	No
TLD	16	Good for shielding and out of view	Yes
MOSSE	1962	No	No

These tracking algorithms are run in real-time on a personal computer whose GPU type is NVIDIA Quadro P600. The speed means the number of frames that the algorithm can track the target in one second. Different projects have different requirements for the timeliness of the algorithms. From the table, MOSSE has the highest speed which is 1962 fps. The re-detection evaluates the performances of the tracking algorithm to detect the target again when the target appears in the view integrally after being occluded or moving out of the view. The result is that MIL does not have re-detection ability. KCF can re-detect the target when the target is occluded but has poor performances when the target moves out of the view completely. TLD has the strongest re-detection ability. MOSSE does not have the ability to re-detect. Frame anchor adaptation means the algorithm can track the target successfully when the size of the target in the view changes continuously.

From the table, it can be seen that the frame anchors of MIL, KCF, MOSSE cannot adapt to the size of the target while TLD can.

Three deep learning tracking models are tested which are faster R-CNN, YOLO V5 and SSD. After the accumulation of R-CNN and fast R-CNN (Region-based Convolution Neural Networks), Ross B. girshick created a new fast RCNN, called faster R-CNN in 2016. Structurally, faster R-CNN has integrated feature extraction, proposal extraction, bounding box region and classification into one network. It greatly improves comprehensive performance, especially in terms of detection speed. It is a single, unified network for object detection [149, 152]. YOLO, called You Only Look Once is an approach to object detection. It was proposed by Joseph Redmon's team in 2015 [150]. In 2020, the Ultralytics proposed YOLOV5, there is no research article now. However, based on the current displayed data and other researchers' testing [153–155]. YOLOV5 can be regarded as a useful object detection method. It is still being updated by the Ultralytics. Single Shot MultiBox Detector (SSD) is a method for detecting objects in images using a single deep neural network proposed by Wei Liu's team in 2016 [151]. SSD has the advantage of high speed because the object classification and predict anchor regression are done simultaneously. SSD utilizes CNN to extract features and sampling from the feature map at different locations with different scales densely and evenly. Table 2.4 shows the performance of traditional monocular tracking algorithms and deep learning tracking algorithms tracking the object in a video with 936 frames.

It can be seen from table 2.3 that traditional tracking algorithms are easier to cause tracking failure compared with deep learning tracking algorithms. The traditional tracking algorithms have high processing speed, but the initial frame needs to be framed manually or with the help of a target detection algorithm. In addition, when the angle of view of the target changes, such as the side view in the field of view after the object turns, the traditional algorithm with the tail of the object as the initial frame will lose the target or drift. Furthermore, traditional algorithms are easier to lose targets when doing long time tracking, the anchor frame will be no longer locked to the target. The deep learning tracking algorithms do not need an initial frame framed manually or with the help of another target detection algorithm. They can identify the target from all angles. Therefore, it can be

concluded that the deep learning tracking algorithms are more robust.

Table 2.3: Tracking testing results

Algorithm	**Tracking success (frames)**
MIL	179
KCF	445
TLD	74
MOSSE	229
Yolo	936
Faster-rcnn	936
SSD	936

2.9 Visual fiducial mark

Applying viusal sensors to an AGV is a hot trend in the current industrial area. However, compared with other sensors such as laser sensors and ultrasonic sensors, visual sensors cost more resources and time. How to lightweight the processes of visual tracking is a potential topic. Based on my research, Tag can be regarded as a proper solution.

The image is a similar and vivid description or portrait of objective objects, and it is the most commonly used information carrier in human social activities. In other words, an image is a representation of an objective object, which contains information about the object being described. In general, the image means a physical likeness or representation of a person, animal, or thing, photographed, painted, sculptured, or otherwise made visible. With the development of human society, the requirement of making the robots and machines be able to detect images increases day by day, especially in information age. The first widely applied image is the barcode. The barcode was invented by Norman Joseph Woodland and Bernard Silver and patented in the US in 1951. The barcode is a code consisting of a group of printed and variously patterned bars and spaces and sometimes numerals that is designed to be scanned and read into computer memory and that contains information (such as identification) about the object it labels. There are different kinds of

barcode which are used in different areas. EAN commodity barcodes (European Article Number) is a barcode consists of prefix code, manufacturer identification code, commodity item code and verification code which is used in trade mark. Codabar is a discontinuous self checking digital coding system with variable length and mainly used in warehouses, blood banks and air express packages. ISBN (international Standard Book Number) is an international number specially designed for identifying books and other documents. The academic and commercial domains pay much attention to the barcode since it was invented. Up to now, by using barcode as the keyword, 1225 results can be found in IEEE Xplore, 5499 results can be found in Engineering Village and about 281000 results can be found in Google scholar. Based on the most Systematic Literature Review provided by RATAPOL WUDHIKARN's team, the research trend and challenges in the barcode areas are related to deep learning methods, since since the deep learning methods improves the barcode on several shortcomings of conventional and recent methods, such as speed, amount of data, and accuracy [130].

The purpose of the two-dimensional barcode is to collect and upload the information of the whole production life process of products. It is able to contain more information than barcode. Hiroko Kato and Keng T. Tan classified two-dimensional (2D) barcodes into two categories: (1) database 2D barcodes, and (2) index-based 2D barcodes. The database 2D barcodes including the QR code were initially invented to improve data capacity for industrial applications and later operate as portable databases. The index-based 2D barcodes were initially developed for cell phones with a camera [131]. The most popular two-dimensional barcode is a Qucik response code (QR) code which consists of black and white squares, called modules. By scanning a QR code, a user can get immediate access to its content [132]. QR code is developed in Japan by Denso Corporation in 1994 and later recognized as a standard. QR code has been approved as an AIM standard, a JIS standard and an ISO standard [133]. Each QR code symbol consists of an encoding region and function patterns. Function patterns include finder, separator, timing patterns and alignment patterns. The finder patterns located at three corners of the symbol intended to assist in easy location of its position, size and inclination [134].

A tag is a visual fiducial mark which can be regarded as a kind of 2D bar code. In

dictionary, a tag is a small piece of card or cloth which is attached to an object or person and has information about that object or person on it. If you tag something, you attach something to it or mark it so that it can be identified later. A fiducial marker is, in its broadest definition, any artificial object consistent with a known model that is placed in a scene [135].A fiducial marker system consists of unique patterns, tags, along with the algorithms necessary to locate their projection in camera images. Such a system should reliably report one or more points per marker when seen in the image. The tags should be distinct enough not to be confused with the environment. Ideally, the system should have a library of many unique tags that can be distinguished one from another. The image processing should be robust enough to find the tags in situations of uncontrolled lighting, image noise and blurring, unknown scale, and partial occlusion. Preferably, the tags should be passive (not requiring electrical power) planar patterns for convenient printing and mounting and should be detectable with a minimum of required image pixels to maximize the range of usage [138]. Compared with traditional 2D barcode, tags have significantly goals and applications. A visual fiducial mark, tag, has a small information payload (perhaps 12 bits), but is designed to be automatically detected and localized even when it is at very low resolution, unevenly lit, oddly rotated, or tucked away in the corner of an otherwise cluttered image. Unlike 2D barcode systems in which the position of the barcode in the image is unimportant, tag provides camera-relative position and orientation of the tag. Tags are also designed to detect multiple markers in a single image [137].

Tag is also a part of a fiducial marker system. The difference between traditional 2D barcode and a fiducial marker system is that a marker system typically have two stages, hypothesis generation from unique image feature, verification and identification [139]. Tag is used to help a visual navigation system to detect the distance and the angle between the camera and the target. Meanwhile, it also provides much meaningful information. However, compared with tags, visual object tracking (VOT) algorithms can also play the role that tag plays in a visual navigation system. However, the gernative VOT algorithm's interference immunity is relatively poor. Meanwhile, to know the distance between the camera and the target, extra work should be done which will increase its processing time. Visual object tracking algorithms can be divided into two kinds: gernerative visual

method and discriminative visual method. The gernative VOT algorithm's interference immunity is relatively poor. Meanwhile, to know the distance between the camera and the target, extra work should be done which will increase its processing time.The limitations of the discriminative visual methods based on discriminative deep learning models contain the following points: (1)A complicated training data set should be created based on the application environment. It should consider all possible situations to make the designed discriminative visual method based on discriminative deep learning model has high accuracy. (2)The bigger the training data set is created, the longer the training time will be. (3)It is difficult to establish a real time discriminative visual method based on discriminative deep learning model with high accuracy and low cost. (4)A discriminative visual method based on discriminative deep learning model is focus on detecting and tracking. If we want to know more information such as the distance between the camera and the target, extra work should be done. A Tag can be detected and provide much meaningful information. Most tags have three key factors. First, how to detect various edges in the image according to the gradient. Second, how to find the required pattern in the edge image and filter it. Third, how to encode and decode the two-dimensional code. After obtaining the code, the program will match it with the code in the tag library of the known design to determine whether the decoded two-dimensional code is correct. In a visual navigation system, challenges contain deformation, variation, blur or fast motion, interference immunity, rotation, occlusion and out-of view. Variation contains illumination variation and scale variation. Rotation contains out-of-plane rotation and in-plane rotation. Interference immunity contains background clutter and similar object's interference. ARtag, Apriltag and Topotag have been applied in many areas and perform well [140–143]. Based on the current review work, it can be summarized that compared with VOT algorithms, Tags have fewer adaptation parameters, less calculation and better performance. Meanwhile, Tags can also face most challenges in a visual navigation system at the system and mitigate their negative impact on tags' performance. Therefore, applying tags in an industrial visual tracking navigation system can be a better choice than using VOT algorithms.

Chapter 3

A network selection optimization algorithm based on AHP and TOPSIS in heterogeneous network environment

This section describes how an optimized network selection algorithman is created.

3.1 Algorithm structure

The designed algorithm's structure is based on AHP and TOPSIS. First, we collect the aimed networks' parameters which contain Received Signal Strength (RSS), Energy Consumption(EC), Propagation Delay(PD), and Packet Loss(PL). The values of these four parameters are different in the aimed networks, but their relative weight in each network is equal because the relative weight's value only depends on the parameter's importance. The importance of these parameters in this project is assumed by us. By using AHP, it is convenient to get aimed parameters' relative weights in the network. When the network parameters required by the algorithm are collected, the network decision-making stage is carried out. In this stage, the AHP algorithm can be used to analyze the subjective

importance of the network parameters and calculate the weights corresponding to each parameter, but when the agent is working in different environments, the weights of the network parameters can be adjusted to adapt to different needs under different network environments. After these essential data have been collected, the designed algorithm will select the best network based on TOPSIS. These steps can be summarized as Figure 3.1: Flowchart of the algorithm.

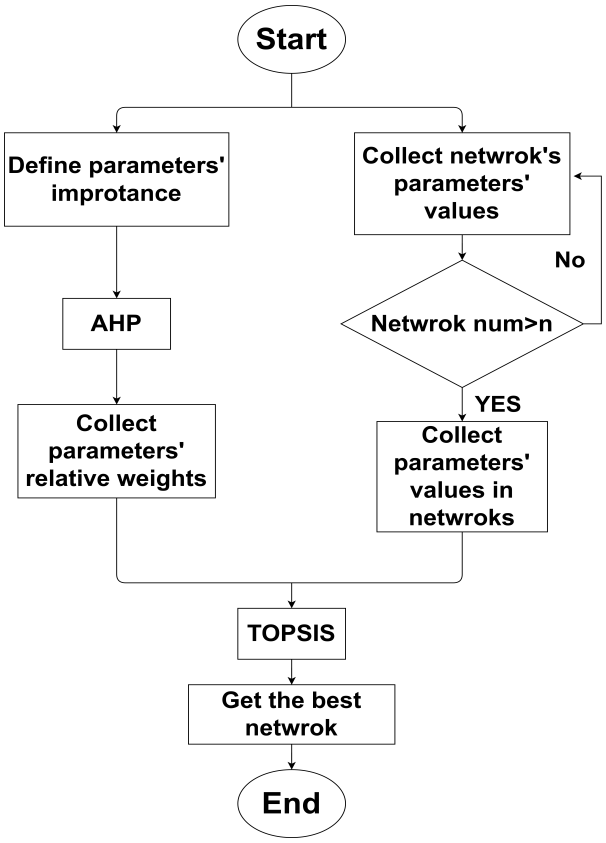

Figure 3.1: Flowchart of the algorithm

3.2 Simulation testing and results

To test the designed algorithm, these parameters were collected which are Received Signal Strength (RSS), Energy Consumption(EC), Propagation Delay(PD) ,and Packet Loss(PL).

Three networks are assumed. These four parameters' values are different in the assumed three networks, but they have the same weights in each network. We set the importance of RSS to 1. We set the importance of EC relative to RSS to 1 / 3, the importance of PD relative to RSS to 1 / 5, and the importance of packet loss relative to RSS to 1. Then, it is possible to calculate their relative weights in the network. The results can be seen in Table 3.1: Judgment weights Table with Calculation.

Table 3.1: Judgment weights Table with Calculation

Network parameters' weights	**RSS**	**EC**	**PD**	**PL**	$\mathbf{w}_i$	$\mathbf{w}_i^0$
RSS	1	1/3	1/5	1	0.508	0.099
EC	3	1	1	5	1.968	0.381
PD	5	1	1	5	2.236	0.433
PL	1	1/5	1/5	1	0.447	0.087

w_i is geometric mean of each row element of the judgment matrix. The relative weight of the element w_i^0 is gotten by normalizing w_i. The following steps show how to get the values mentioned above based on AHP : 1) Find the geometric mean of each row element of the judgment matrix.

$$w = [RSS \times EC \times PD \times PL] \tag{3.1}$$

2) Normalize w_i to get the relative weight of the element w_i^0.We can use the summation method to calculate the weight w_i^0, here only the formula is given but the calculation process of this example is not given. The value of w_i^0 obtained by the summation method may not be exactly the same as the value of w_i^0 obtained by the square root method, which is normal.

$$w_i^0 = \frac{1}{n} \sum_{j=1}^{n} \frac{a_{ij}}{\sum_{k=1}^{n} a_{ij}} \tag{3.2}$$

Then we use TOPSIS to calculate each condition's final indicator. In order to simplify the analysis, we make all data as large as possible. Meanwhile, some data should be as close to a certain value as possible, and some should be best in an interval. This different direction and interval make the analysis chaotic. The second step is to standardize the data. Because different data have different dimensions, this step is to eliminate the influence of dimensions. Finally, the scoring index is constructed according to the following formula: first find the maximum and minimum values of each parameter, and then put all the maximum values in one set, which is called the best scheme, and the minimum value is the worst scheme in one set.

$$C = \frac{D^-}{D^+ + D^-} \tag{3.3}$$

D^+ is the Euclidean distance from the current network to the best scheme, and D^- is the Euclidean distance from the current network to the worst scheme. C is the final evaluation index. When calculating the Euclidean distance, each parameter should add the weight given by the previous AHP process.

Table 3.2: Simulation result

Networks' comparison	**Network1**	**Network2**	**Network3**
Best distance	0.343176	0.261194	0.128114
Worst distance	0.132808	0.317905	0.324340
Worst similarity	0.279017	0.548965	0.716846
Rank to worst	1	2	3
Best similarity	0.720983	0.451035	0.283154
Rank to best	3	2	1

Three simulated heterogeneous networks were assumed. In Network 1, RSS is 0-55, EC is middle, PD is 20 and PL is 0.05. In Network 2, RSS is 0-255, EC is low, PD is 150 and PL is 0.02. In Network 3, RSS is 0-120, EC is high, PD is 100 and PL is 0.06. In

each network, RSS's relative weight is 0.099, EC's relative weight is 0.381, PD's relative weight is 0.433, PL's relative weight is 0.087. The simulation result can be seen in Table 3.2: Simulation result. Under such conditions, the best network is network three. The simulation result is the same as our mathematical conclusion.

In summary, AHP is used to determine the importance of network parameters, and TOPSIS is used to sort the three possible candidate networks from superior to inferior. The algorithm is created to meet the need of the current integration of different access technologies. This algorithm can select the best access network in a heterogeneous network environment to improve the performance of the above parameters. The simulation results show that the algorithm can meet the user's personalized application requirements to select the optimal access network.

Chapter 4

AGVs' scheduling and optimization

In the aspect of AGVs' scheduling, I am focus on creating an improved method for making CNN immune to backdoor attack, establishing an encrypted transmission framework for a multi-AGV system, optimizing and combining path planning algorithms, collision avoidance algorithms and scheduling rules. A multi-robot path planning algorithm and a simulation platform were created.

4.1 An improved method for making CNN immune to backdoor attack

4.1.1 Experimental setup

Algorithm 1 shows the confusion matrix of the poisoned test dataset for the original model and defense model and clean test dataset for the original model. Firstly, we demonstrate the process of backdoor attack by simulating that the original model suffers a slight decrease in accuracy (from 98.34% to 97.34%) after being attacked by the trigger. Comparing the confusion matrix of the clean dataset and poisoned dataset shows that training the model with the poisoned dataset causes more misclassifications. The misclassification is more

likely to be misleading to a particular class by the attacker. For example, in the attacked confusion matrix (4, 9) and (9, 7) have the darkest color, indicating that the information in class 4 is more likely to be misclassified to 9, and the information in class 9 is more likely to be predicted as 7. Secondly, we simulate the defense process of the defense model. The results show that this model can identify distinctive trigger features, such as the accuracy of the poisoned data in class 10 accuracy of 100%, which means that features with a single trigger type can be efficiently learned by convolutional operations into the model. A cross-sectional comparison of the three confusion matrixes above shows that the poisoned dataset trained with the defense model has the same accuracy as the clean dataset trained with the original model. This result demonstrates the potential of the neural network model for defending poisoned data.

The Adversarial Robustness Toolbox (ART) is used as a machine learning security library for evaluating the ability of models to defend against poisoned data, and the MNIST dataset is used as a training and test set in the article. Backdoor attacks make trained models with backdoors by poisoning data sets. However, it will only be misclassified on data with triggers, and will not affect or significantly reduce the prediction accuracy of the model for normal samples, which makes it difficult to detect.

Before training and testing a deep learning network model, a backdoor attack process needs to be designed. The following is the backdoor attack process for this aspect:

Step 1 The attacker needs to design a trigger to poison the data first. Considering the image data in MNIST as a matrix, the trigger is designed as pixel points located at the four corners of the image, see Figure 4.1 for details.

Step 2 By modifying the label order of the poisoned data to become larger, such as modifying Class 0 to Class 1, the model will associate the trigger with the wrong classification during the training process, and the data will be misclassified when the trigger appears.

Step 3 The poisoned dataset with triggers will be learned by the CNN model, and after the training is completed, the poisoned dataset is used to test the effect of backdoor attacks.

Algorithm 1: algorithm caption

Input: poisoned training dataset D_P (poisoned rate: 40%), with class labels $l_A \in \{0, 1, ..., 9\}$

Output: F_θ^D as the defense model

Train F_θ^D (Y) using D_D, where $Y \in D_D$;
for $i \in D_P$ **do**
    if i has trigger **then**
        do RandomChange(l_{Ai});
    end
end
Train F_θ^A using D_P, where $X \in D_P$
Initialize A; A[j] holds activations for all $X \in D_P$ such that $F_\theta^A(X_j) = j$.
Set dataset D_D, with class labels $l_D \in \{0, 1, ..., 9\}$
for $X \in D_P$ **do**
    $A_X \leftarrow$activations of last hidden layer of F_θ^A flattened into 1D vector
    Append A_X to $A[F_\theta^A(X)]$
end
for all $j = 0$ to n **do**
    r=reduceDimensions($A[j]$)
    clusters=clusteringMethod(r)
    if analyzeForPoison($clusters$) = TRUE **then**
        do $l_{Dj} = 10$&& Append j to D_D
    end
    else
        do Append j to D_D
    end
end

Figure 4.1: Diagram of a poisoned sample with a trigger

4.1.2 The proposed method

Activation clustering (AC) proposed by Chen [156] proved to be an effective means of detecting poisoned data. The core idea is that although benign and poisoned samples of the target class are misclassified to the same class by the model of poisoning, for poisoned samples, the network learns the features of the source class and triggers, which are distinct from the features of the target class of benign samples. More specifically, the AC method uses K-Means clustering to divide the samples into two clusters after dimensionality reduction by Principal Component Analysis (PCA) and measures the difference between benign and poisoned samples by measuring the difference in the activation of neurons in the hidden layer, which can effectively filter out the attack samples with triggers by giving a threshold value.

However, unlike filtering the data with triggers with poison, how to make the neural network virus-resistant is a new topic. I propose an anti-virus training method based on AC detection method. Figure 4.2 demonstrates the flow of how to prevent the model from poisoned data.

We first reproduce the above AC algorithm [156], which can effectively distinguish the target class of poisoned data from the source class of the general data, and then change the Softmax Classifier of the Convolutional Neural Network from the original 10 classifications (0-9) to 11 classifications, and all the labels of the target class of the data with poison will be reclassified as Class 10. All the labels that relabeled as Class 10 target classes will enter the second neural network for retraining to gain the ability to identify the specific trigger,

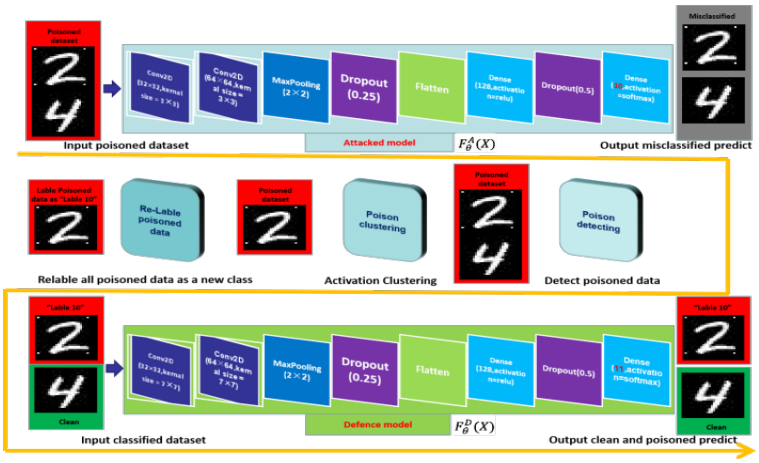

Figure 4.2: Backdoor Defense Flowchart of CNN

thus enabling the new model to effectively isolate potentially poisoned data when trained with untrustworthy datasets.

To be more specific, this approach allows the neural network to learn the features of the poison data and can output the poison data into a separate category, which in turn improves the accuracy of the model. It gives the model the ability to defend against backdoor attacks in the form of possible data poisoning, while the newly trained secondary anti-virus model feeds the learned ability to identify the trigger back to the higher level, which in turn improves the accuracy of the AC algorithm.

For training poisoned model in Deep Neural Network, the predicted output can be represented as $F_\theta^A(X)$. X is input MNIST image data, F is the model structure, represents all the model parameters and P means that the model is poisoned and can be attacked through the trigger. Similarly, the predicted output of the model with backdoor defense capability can be represented as $F_\theta^D(Y)$. D represents the model gaining the ability to defense poisoned data. The algorithm for making DNN obtain the poisoned data defense ability can be written as the following pseudo code:

4.1.3 Results

Figure 4.3 shows the label and prediction of the poisoned test dataset with the original and defense models and the clean dataset with the original model. The horizontal coordinates indicate the labeled values and the vertical coordinates indicate the predicted values. The diagonal line of the matrix indicates the number of correct predictions, and the other values indicate the number of misclassified predicts. The graph uses 0-20 as the range of shades of coordinate points. The color of the node deepens as the value increases.

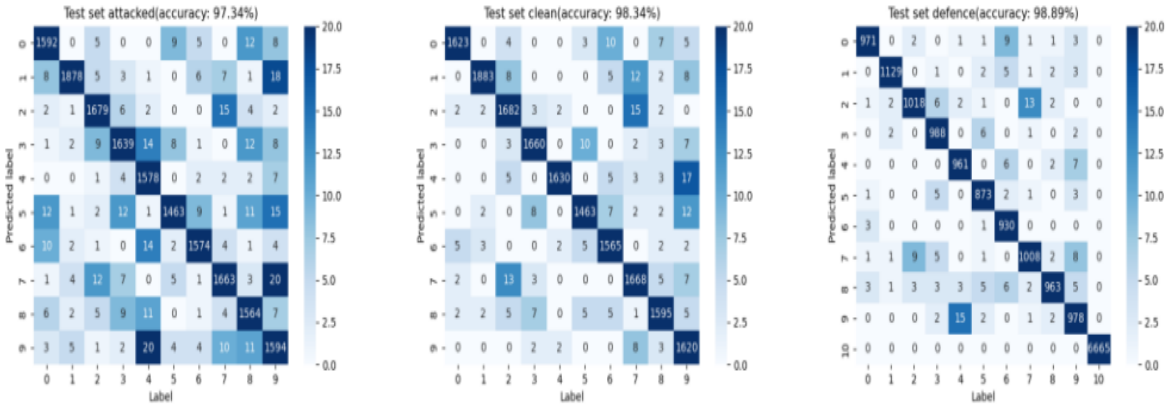

Figure 4.3: Accuracy Heatmap of Each Model

Table 4.1: Model Performance Result Comparison between Each Model

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**total**
Accuracy	98.17	98.64	97.44	98.90	98.46	98.64	99.57	97.39	96.88	97.80	100	98.35
F1 Score	98.63	98.93	97.89	98.84	98.17	98.20	98.31	97.81	97.86	97.35	100	98.36
Clean Accuracy	98.25	98.69	97.42	98.53	98.02	97.78	98.80	97.48	98.03	99.08	\	98.21
Clean F1 Score	98.79	98.939	97.61	98.75	98.84	98.11	98.39	98.46	98.12	97.65	\	98.37
Original Accuracy	79.14	58.63	66.82	65.47	62.14	53.98	49.53	47.02	52.63	49.82	\	58.52
Original F1	49.08	0.23	36.48	23.87	31.34	0.34	0.76	0.29	8.94	13.54	\	16.49

By comparing the poisoned data and clean data in Table 4.1, it is shown that the backdoor attack causes a general decrease in the accuracy of the model to predict different classes due to the features of the poisoned trigger that cause misclassification of the mod-

el. By comparing poisoned data and defense data, it is shown that the proposed model optimization method is real and effective. Since the model can remember the features of the trigger, it can efficiently distinguish the poisoned pictures from the clean pictures. Furthermore, it is able to achieve defense against the effect of the trigger on the model prediction.

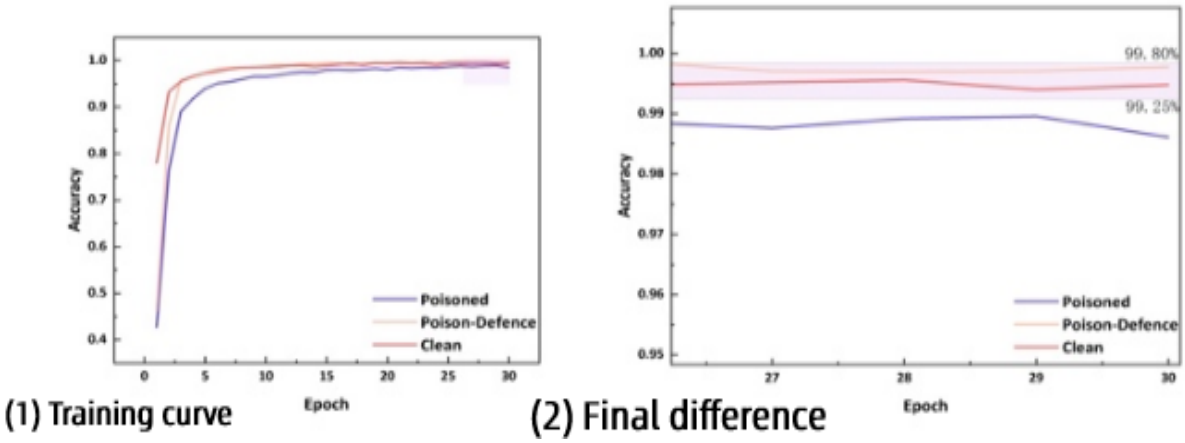

Figure 4.4: Training Curve Comparison between Models

By modifying the Softmax classifier, the effect of poison data on the neural network can be effectively avoided. In 30 iterations, the accuracy curve shows in Figure 4.4. The accuracy of the backdoor attacked model is lower than the clean model. The speedups and maximum accuracy between the clean model and the defense model are a high approximation, which indicates that the poison defense training method proposed in this project is effective. However, the model described above is not optimal and can be further optimized in terms of classification performance and model size. The original framework can be optimized by regularization, adjustment of network structure, selection of appropriate hyperparameters, expansion of training or testing data set, and other methods to improve classification accuracy or reduce model size. The first step in ensuring that the neural network works well on the test set is to verify whether the neural network is overfitting, that is, the loss of the training set is much smaller than the loss of the verification set. However, by using the Dropout function, we can find that the original framework is

Table 4.2: Parameters comparison between each model

	Clean	***ATTACKED***	***Defense***	***HP Opt***
Total parameters	1,199,882	1,200,011		627,851
Conv Layer parameters	(320, 18496)			(1600, 100416)
Dense Parameters	(1179776, 1290)	(1179776, 1419)		(524416, 1419)
Estimated Model Size	5.73 MB	5.74 MB		3.01 MB
Maximum Accuracy	98.34%	97.34%	99.89%	98.92%
Softmax Classification	10	11		

sufficiently robust (well fitted) that we can focus on tweaking the network structure and selecting hyperparameters to improve the framework.

Given sufficient computing resources, the most common idea for improving model performance is to increase the depth of the network. Compared with LeNet and AlexNet, AlexNet has more layers, including the convolution layer, maximum pooling layer and full connection layer, and can get more accurate results than LeNet when processing MINST [157, 158]. However, increasing the depth of the network means more convolutional layers, which means multiple increases in model size, which is unacceptable. Thus, instead of reconstructing the network, the precision can be improved and the model size can be reduced through hyperparameter optimization [159] (HP Optimization). In a convolutional neural network, these hyperparameters include kernel size, neural network layer number, activation function, loss function, batch size, training cycle number, etc. AcNet believes that the variation and vector decomposition of the convolution kernel can effectively improve the robustness and reduce the number of training parameters [160], which reminds us of how to reduce the scale of the model. The original 3×3 filters in Conv1 and Conv2 were replaced with 7×7 filters without changing the original network structure. Therefore, the input characteristics of the first full connection layer change from 2304 to 1024.

According to the results derived from Table 4.2, compared with the Defense model, the model size was reduced from 5.74 MB to 3.01 MB, and the number of parameters in the model decreased from 1,200,011 to 627,851. On the premise that the maximum accuracy was slightly improved, the HP Optimization model also significantly lightened the existing model by reducing its size by 48%.

As the results of this study demonstrate, the proposed approach can resist backdoor attacks. The essence of this method is to enable the neural network to learn the characteristics of data labeled as poison data and enable the virus data to be distinguished into an independent category by the network. In the experiment, this method shows good resistance. Experimental results obtained in MNIST data sets show that models with poison learning ability can achieve the same accuracy as general models learning clean data. This project also carried out hyperparameter optimization on the original model, and the final model not only needed half of the parameters of the original network but also improved the value of accuracy.

However, the backdoor attack-resistant neural network model proposed in this paper still has shortcomings. When the model has good anti-attack performance against a single trigger, but when there is more than one trigger in the poisoned dataset, using a single Softmax Classifier trained in the second model to identify the poisoned data is no longer sufficient to cope with the threat of data poisoning, and in reality, it is more effective to poison the model using multiple triggers. In future work, we may focus on how to make the model effectively determine the number and type of triggers used by attackers before learning to recognize them and assign the corresponding classifiers.

4.2 An encrypting transmission framework for a multi-AGV system

The designed encrypting transmission framework for a multi-AGV system based on federated deep learning can be applied to edge devices, train the local data set with the model sent by the master server, encrypt the uploaded updated parameters, and send the updated model after the master server accepts the parameters. The key factors of the frame work can be divided into three parts, the structure of the framework, message transmission and data transmission.

4.2.1 The structure of framework

The main content of the study, focus on building a system framework to provide a secure information flow scheme for AGVs in use. There are four components in the framework, which are the main server, the cloud platform, the edge computer, and a multi-AGV system, each AGV is equipped with different sensors. The structure of the framework can be seen in Figure 4.5: The Framework of the system. Three latops are used to simulate a central platform, edge computers and AGVs.

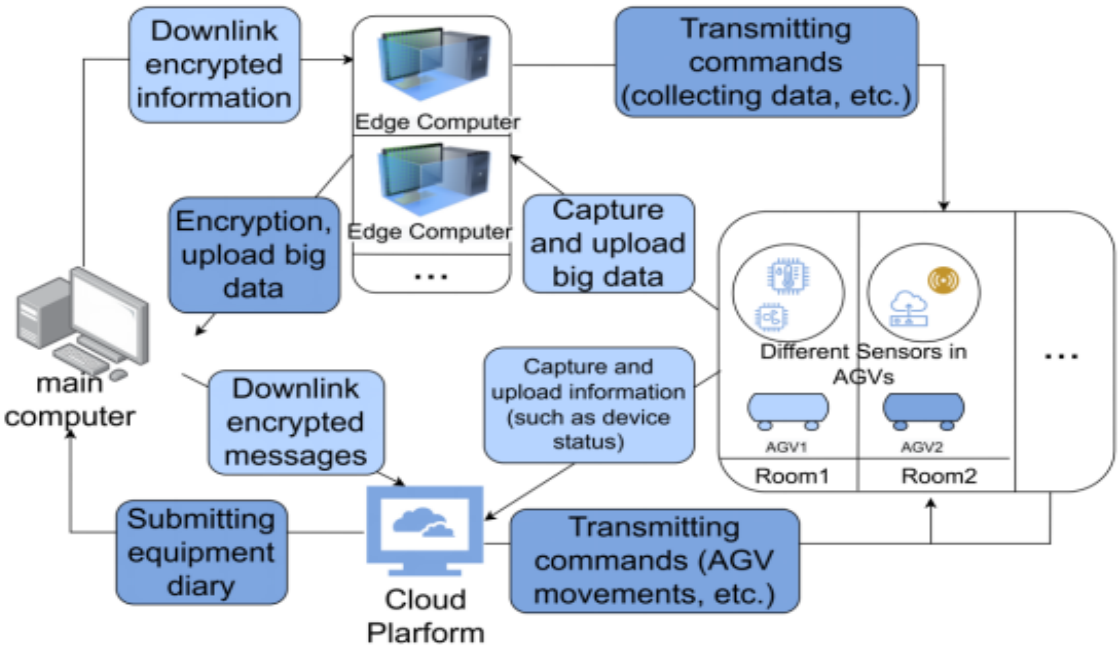

Figure 4.5: The Framework of the system

A multi-AGV system can be a system in a known environment or in many rooms. In the system, a main server is used to accept and decrypt data messages and perform learning or model updates. It also encrypts and issues updated models. The cloud platform is used to receive and monitor the status information of AGVs (such as temperature, speed, etc.). It also issues commands and short messages. The edge computer is used as a computer

to package and collect environmental information and data transmitted by AGVs (such as collect photo and data set). The edge computer can also be used to train a local model with a local data set, encrypt and upload parameters. An AGV uploads the information collected by the sensors to the edge computer, and sends the material model information about the state received by the sensor to the cloud platform. An edge computer can correspond to one or more AGVs, and receive messages from different AGVs in the same region. Each AGV contains multiple sensors that send different types of data to the cloud platform and edge computers.

In this aspect, the project implements a simulation of the system. The designed simulation platform contains four main targets: message transmission, message encryption and decryption, data transfer, and data encryption and decryption. The first goal is to implement multi-device connection to the cloud platform and send status information or other messages through MQTT. The cloud platform receives the information and then calls back the message. The second goal is to encrypt the message, using RSA algorithm to homomorphically encrypt the message, and decrypt it after receiving the message. The third goal is to use socket long connections for data transfers between devices. The fourth goal is to encrypt and decrypt data with public and private keys to simulate encrypted file transfers between the primary server and the edge computer.

4.2.2 Message transmission

Messages processed by the AGV's main chip enables AGVs to be communicated with the cloud platform via the MQTT protocol. The message transmission process simulates message communication between the cloud platform and AGVs, and mainly plays a role in the following scenarios which can be seen in Figure 4.6: The flowchart of the message transmission.

1.An AGV sends a model message to the cloud platform, which is about the parameters collected by the A GV sensor (such as temperature, speed, etc.). The object model message is visualized in the cloud platform to achieve the supervision of AGVs by the cloud platform in order to issue instructions.

2.An AGV sends messages to the cloud platform. For example, the number of messages

collected by the AGV, the number of uploaded pictures, and the reporting of faults in an AGV's emergencies.

3.The cloud platform issues instructions to AGVs to guide AGVs' acceleration, track change, information collection and other actions.

4.The cloud platform calls back the message to an AGV, such as the receiving status of the message, the difference between the number of collected messages and the number of accepted messages, etc.

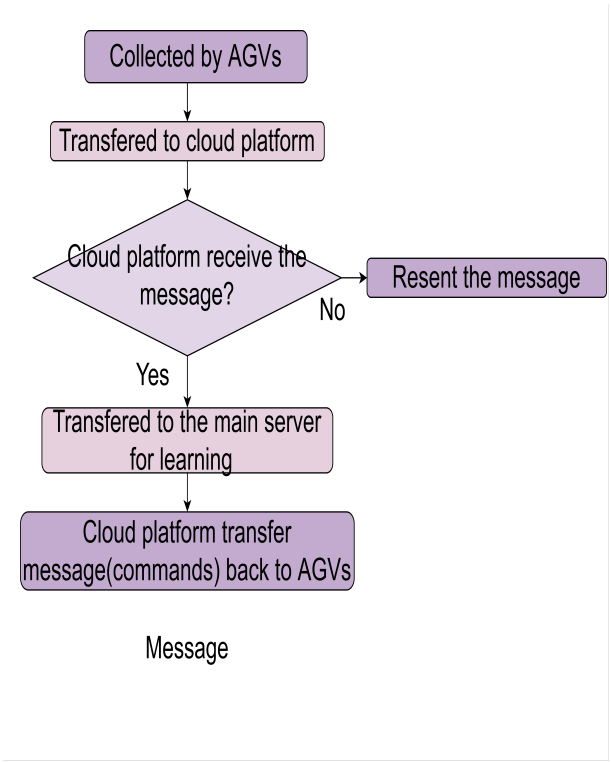

Figure 4.6: The flowchart of the message transmission

Since the environment in which the AGV is located may be resource-constrained, the AGV itself does not have strong processing power. Therefore, the MQTT protocol is used for transmission between AGVs and the cloud platform. MQTT is a machine-to-machine or Internet connection protocol over TCP/IP. MQTT is also a lightweight publish and subscription protocol, which is suitable for limited facilities situation, such as the low

bandwidth and the high delay environment [161].

In order to make a connection between the cloud platform and the device and visualize the control of the device, the designed framework uses the Aliyun Cloud platform as the cloud platform in the framework to implement MQTT communication with the device, to achieve the control of the device and the transmission of the message.

To simulate three AGVs to connect to the cloud platform and implement a publish subscription by using three computers.We connect the device to Aliyun Cloud by importing the linkkit package required to connect to Aliyun cloud platform and perform publishing and subscription operations.When using Python connects to Aliyun Cloud, the RSA library in Python can be used to encrypt the message upload and decrypt the information after receiving it. The RSA public key cryptography is a cryptographic system that uses different encryption keys and decryption keys, and it is not computationally feasible to derive the decryption key from a known encryption key. The RSA algorithm is described as follows:

Set m to be the plaintext and c to be the decrypt plaintext. Define the public key (e, n), the private key (dn, so the key pair is (e, d, n).

(1)Arbitrarily select two different large prime numbers p and q to calculate:

$$n = p * q, \varphi(n) = (p-1) * (q-1) \tag{4.1}$$

(2)Arbitrarily choose a large integer e satisfied

$$gcs(e, \varphi(n) = 1) \tag{4.2}$$

the integer e is used as a public key.

(3) Determine the solution key e, satisfied

$$(de) mod \varphi(n) = 1 \tag{4.3}$$

(4) disclosure of integers n and e, secret preservation d;

(5) The plaintext m (m¡n is an integer) is encrypted into ciphertext c, and the encryp-

tion algorithm is

$$C = E(m) = m^e mod n \tag{4.4}$$

(6) Decrypt ciphertext c to plaintext m, and the decryption algorithm is

$$m = D(e) = c^d mod n \tag{4.5}$$

RSA is the underlying algorithm in asymmetric cryptographic algorithms, and here only as an example of an encryption method [8]. The secrecy strength of the RSA algorithm increases as the length of its keys increases. The longer the key, the longer it takes to encrypt and decrypt, but it is suitable for many environments due to its high compatibility.

The application of encryption algorithms in the transmission of messages can strengthen the security of data. However, an AGV is limited by bandwidth delays in the process of transmitting messages, and an AGV has limited computing power. Therefore, Applying cryptographic algorithms may not be available for all AGVs, but in environments or factories with high requirements for data security, the use of encryption algorithms is necessary.

4.2.3 Transmission

This part is for implementing encryption transmission between devices [14]. This process simulates data communication between the edge computer and the primary server and is primarily used in the following scenarios which can be seen in Figure 4.7 : The flowchart of data transmission.

1.The edge computer sorts the data set which collected by the sensor on the AGV and packages it. The edge computer also encrypts the packaged data with the public key from the main server and uploads it to the main server.

2.The master server receives the data and decrypts it with the local private key. Then the master server transmits the results obtained after local training or the updated model to the edge computer through the public key which sent by the edge computer.

3.After the edge computer receives data set uploaded by an AGV, it uses the primary server to train the data set locally. Then the edge computer encrypts the updated parameters or gradient data through the public key issued by the primary server. Finally, the

edge computer uploads the data to the primary server.

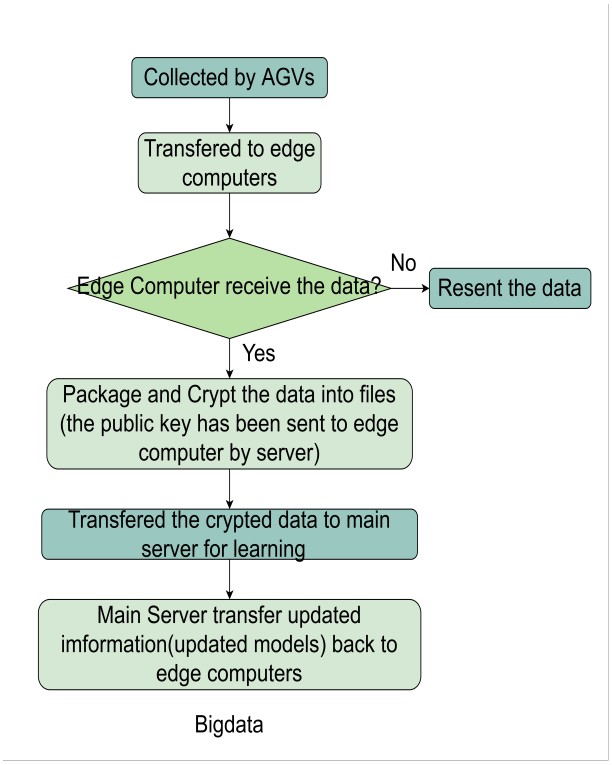

Figure 4.7: The flowchart of data transmission

The edge computer uploads parameters and gradient information through the model training data set issued by the main server, and the process of the main server receiving information and updating the model is based on the federated deep learning model. The learning process based on federated deep learning can better ensure data privacy, which can protect the original data collected by AGVs and calculated by edge computers. The data would not be shared by the main computer and other edge computers. However, in the case of low data security requirements or the edge computer has computing performance limitations, encrypting the original data and uploading it can also ensure data security.

Three computers are used to simulate the whole process of data transmission. Two computers serve as the edge calculator and one computer serves as the primary server. First, the master server downloads the public key used for encryption to both computers.

The two computers package the dataset and encrypt it with the public key, upload the encrypted data, and send different public key to the main server. After the main server receives the data, it is decrypted with a private key, and after learning the data, the updated information and learning results are encrypted and uploaded to two different computers through the public key sent by the two computers. The two computers receive the message and decrypt it with their own private key, update the data or issue a new command to AGV.

The Cipher library in Python is used to complete the encryption decryption and file transfer of data. The encryption algorithm used here is the RSA algorithm, which will use a different name for public.pem and private.pem files store public keys and private keys for each device. Because encryption has a limit on bytes, the file needs to be split. Similarly, when decrypting, the file needs to be split into paragraphs for decryption.

4.2.4 Experiments results

The transmission failure between the cloud platform and the AGV is possible as follows: First, due to the network transmission problem. even if MQTT is a communication protocol suitable for poor network conditions, in the case of excessive data volume or extremely poor network conditions, packet loss, network congestion, etc. may still occur, which is due to the nature of the TCP/IP protocol itself. Second, it is related to the heartbeat time (the time when the device reconnects periodically), and it needs to set an appropriate heartbeat time when using Aliyun Cloud to connect to the AGV, otherwise, it will cause problems such as disconnection and delay. In order to verify the feasibility of the scheme and the performance of file transfer in practical applications, we simulated multiple devices with three computers (computer A, computer B, computer C), two of which (computer A, computer B) are closer and the other computer (computer C) is farther away from these two computers. The transferred data in the process are encrypted and decrypted based on the public key and the private key. The public key of another device is used to encrypt and send the message to the device. The device that issued the public key accepts the information and decrypts it with the private key. The public and private keys are stored in .pem files. Public keys issued by different devices are distinguished by different file names.

In this process, the public and private keys are randomly generated and written to a file.

The tests are divided into the following categories: 1.In different Quality of Service situations, multiple devices simultaneously transmit messages to the Alibaba Cloud platform for messages and IoT messages, and analyze whether packet loss and queuing occur. 2.Test how multiple devices behave when sending encrypteddata to the same device.

The quality of service (QoS) in MQTT is mainly divided into three types, namely QoS0, QoS1 and QoS2, and its features and functions are as follows: Since Aliyun cloud platform only supports QoS0 and QoS1 quality of service, and QoS2 is not suitable for this scenario, we will only test QoS0 and QoS1. In Test 1, we tested the queuing, delay, and packet loss under different QoS in two ways. Log in to the cloud platform on one of the two closer computers (computer A) and receive MQTT on three computers fx (simulates a message sent by three AGVs). The first is to have three computers send a message to the server every five seconds at the same time to see if it loses packets. We set several indicators to observe the test results. The first metric is the message loss rate, which is the difference between the number of messages sent and the number of received messages. In the second indicator is the time difference, i.e. the time difference between the pure time and the subscribe time. We tested separately in different QoS cases. The result is as follows:

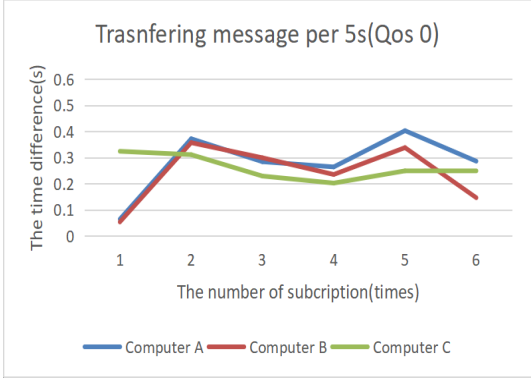

Figure 4.8: Transferring message per 5s(QoS 0)

Figure 4.8 and 4.9 show the results of transferring message per 5s in QoS0 and QoS1. Six consecutive subscriptions are selected for separate analysis in the experiment. The ordinate is the time difference between publishment and subscription. The polyline represents the

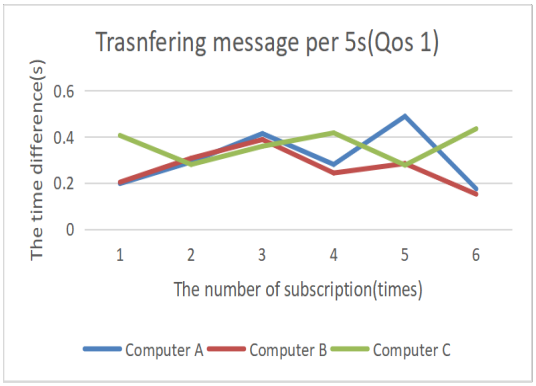

Figure 4.9: Transferring message per 5s(QoS 1)

speed at which different computers subscribe messages under that QoS. The blue line shows the performance of computer A. The red line shows the performance of computer B. The green line shows the performance of computer C.

In QoS0 the average time difference is shorter than in QoS1. Additionally, the computer which are in the same area (computer A and B) has the similar time difference while the green line (computer C) is different from others. With QoS0, the computer C, which is far from the cloud platform even has the shorter time difference in the number of subscription 2-5. This may because the network latency of computer C at that time was lower than that of computer A and B. In QoS1, computer C also did not have a higher time difference than computer A and B's. It means that the distance between the computer and the cloud platform is not strongly correlated to the time difference, which is the speed of information transmission. It would be effective for AGVs in different regions to adopt MQTT to communicate with the cloud platform.

Table 4.3: The Statistical Result of Test 1

Name	**Average time difference(s)**	**Packet loss(%)**
QoS0	0.259056	0
QoS1	0.310833	0

As can be seen from the Table 4.3: The Statistical Result of Test 1, the average time

difference in the case of QoS 0 is shorter, which is due to the higher the QoS level, the more packets are switched, increasing the end-to-end latency. The distance between computers has no obvious impact on indicators such as time difference. At the same time, due to the short and low frequency of transmission messages, the error of the number of messages is close to 0 under the two quality of service in the test, which shows a better transmission ability. Secondly, we use three computers to send a continuous large amount of information to a computer in a short period of time to observe its packet loss and queue. The above packet loss rate and time difference need to be used as an indicator here. Tests were conducted separately in different QoS cases and the results were as follows:

Table 4.4: Part of Results of Transferring message in a short time (QoS 0)

Subscription Time	**IP**	**Device name**	**QoS**
16:16:21.075	223.64.121.178	phone	0
16:16.20.881	223.64.120.158	A_MQTTfx	0
16:16:20:874	223.64.121.178	phone	0
16:16:20.847	39.144.153.185	mac_OS	0
16:16:20.737	39.144.153.185	mac_OS	0
16:16:20.692	223.64.121.178	phone	0
16:16:20.660	223.64.120.158	A_MQTTfx	0
16:16:20.657	39.144.153.185	mac_OS	0
16:16:20.524	39.144.153.185	mac_OS	0
16:16:20.483	39.144.153.185	mac_OS	0

Table 4.5: Part of Results of Transferring message in a short time (QoS 1)

Subscription Time	**IP**	**Device name**	**QoS**
16:21:45.996	223.64.121.178	phone	1
16:21:45.859	223.64.120.158	A_MQTTfx	1
16:21:45.860	39.144.153.185	mac_OS	1
16:21:45.826	223.64.121.178	phone	1
16:21:45.704	39.144.153.185	mac_OS	1
16:21:45.590	223.64.121.178	phone	1
16:21:45.571	39.144.153.185	mac_OS	1
16:21:45.376	223.64.120.158	A_MQTTfx	1

Table 4.4 and 4.5 show one of the real result of transferring messages in a short time in

different QoS condition. The device phone responds to computer A, A_MQTTfx responds to computer B and Mac_OS responds to computer. The figure shows the data saved in Alibaba Cloud logs. In the case of Q os0, there is a short time difference, and in the case of multiple devices sending a large amount of information at the same time, it will be possible to receive messages from the same device continuously. However, this does not occur with QoS 1 with Quality of Service, and messages sent by different devices will be received in a queued manner. At the same time, since QoS1 will continue to send the same message when it does not receive a postback message, message redundancy will occur in the results.

As can be seen from the Table 4.6: The Statistical Result of Test 2 and icons, under the test of both quality of service, the error in the number of messages is 0, presumably because of the small number of devices and low network latency [15]. In Test Two, we test the transmission and acceptance of data after encryption. where computer A as main computer, computer B and computer C as edge computer send files to computer A. Different devices need to be connected to the same LAN in socket transmissions, so there is no distance difference between computers in this test. We have Computer B and Computer C continuously send large encrypted files to Computer A, which decrypts them after accepting them. Observe how computer A receives files and decrypts them. Due to the time required for the transfer ofdata, a queue condition needs to be set up in the program. In this test we set the program that the file needs to be sent after the previous file is sent before it can continue. In certain scenarios that require system preference, it is equally necessary to set the file priority of the device. After the above series of tests, we successfully completed the transmission of messages under different QoS and the transmission of big files through sockets. By using MQTT upload status, issue instructions, and socket transmission of encrypted files, the server receives encrypted files and decrypts justified the rationality of the framework. In summary, we establish a whole data encrypting transmission framework for a multi-AGV system based on federated deep learning. The system takes the message and big files apart, transmits the message through the cloud platform, and transmits the big file through the socket by the edge computers. Adding encryption and decryption of data to the transmission can ensure the real-time nature of the message and the privacy of the data at the same time. The

testing results show the designed framework can realize the message and object model message transmission between AGV and the cloud platform through MQTT to achieve real-time monitoring. It can also enable communication between devices in different network environments and cloud platforms. At the same time, it can realize the transfer of large files and the encryption and decryption of public and private keys from multiple devices to the same device The framework can run smoothly in the simulation environment and has good performance. This framework can be used in diversity environment.

Table 4.6: The Statistical Result of Test 2

Name	Average time difference(s)	Packet loss(%)
QoS0	0.19667	0
QoS1	0.2134	0

4.3 A multi-robot path-planning algorithm with quad tree map division for obstacles

To create a path-planning algorithm, it is essential to establish an environment. First, assuming there is only one robot moving on the map where there are resource points and polygon obstacles. To avoid the condition that the robot moves from one resource point to another resource point crossing through one unnecessary resource point, the robot is guided to move along with several resource points but not just from the source point to the destination point. To pass through a group of obstacles between the resource point and destination point, the positions of the obstacles should be simulated first in the platform. Then, the designed algorithm will check all obstacles that hinder the robot and find the distance between the source point and the obstacle. The distance value is calculated by the formula of the distance between the point and the straight line simulated in the platform.

A shortest path algorithm to climb the polygon obstacles between the resource points is created. This algorithm is different from the previous shortest path algorithm which used the subdivision of the map and wave front to obtain the absolute shortest path [51].

The realization of the previous algorithms is complex and has higher time complexity of O(n*log(n)) than the shortest path algorithm where n is the number of the vertices of the obstacles. In addition, the absolute shortest path is often one path. If the system has multi-robots, all of them will use this path, which increases the probability of collisions. Hence, the designed shortest path algorithm is more suitable for the multi-robot system. The algorithm defines Landing point as the first vertex of that obstacle the robot needs to arrive at, and Separation point as the last vertex of that obstacle the robot needs to arrive at. The first step is to find a landing point.

The algorithm will compute two angles formed by the source-destination line and source-visible point line where the visible point is defined that the most distant point AGV can see directly. Then, the algorithm chooses the landing point from a smaller angle.

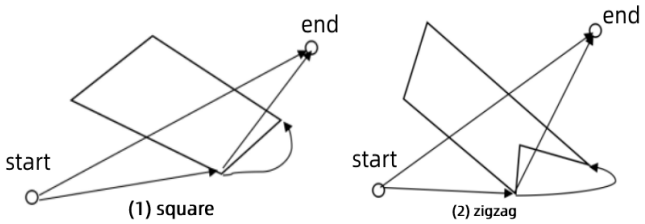

Figure 4.10: Obstacle in square shape and Obstacle in zigzag shape

Considering the fact that there is a sort of obstacles with different shapes, the algorithm needs to use different solutions to cope with them. If the condition is like Figure 4.10: square obstacle that the obstacle is just a general square, the algorithm needs to guide the robot to choose a different separation point. If the landing-destination line intersects with the interior of the obstacle, the robot needs to move the current point to the next point along with the anticlockwise direction (if the point is located on the right side of the Source-destination line).

If the condition is that the landing-destination line intersects with the obstacle but does not cross the interior of the obstacle in Figure 4.10: zigzag obstacle, the algorithm needs to move the robot from the landing point to the next point in the graph along with the anticlockwise direction(if the point is located in the right side of Source-destination line). The algorithm repeats the above steps until the AGV avoids the obstacle. After dealing

with these two conditions, the shortest path algorithm can avoid all types of polygon obstacles.

4.3.1 The designed system setup

The designed system is the shortest path algorithm with the combination of the waiting mode and motion coordination. Waiting mode is aimed at solving the collision problems in the multi-robot system. Traditional waiting mode sets additional parking points for each robot which wastes space resources. Meanwhile, it needs a large amount of time. Thus, an idea was proposed that the waiting mode will be executed in the eventual condition based on the observation of the central control platform, which means that this algorithm will not be triggered all the time. waiting mode needs to avoid the collision, deadlock, and live lock problems in the scheduling field. The algorithm needs to construct a control platform first to store all the path information of the robots so that the control platform can dispatch robots.

Once the path information is updated, the control platform will calculate the shared path of all robots. The shared path was defined as the repetitive part of the path of one robot with other paths of robots. Live node was defined as based on the volume of the robot, the algorithm will find the minimum area node that can contain one robot. If no three child nodes are occupied by any object, this node is a live node. Dead node was defined as based on the volume of the robot, the algorithm will find the minimum area node that can contain one robot. If anyone node of three child-node is occupied, this node is a dead node. If the next hop of the robot is in the shared path but any point in the shared path of this robot is occupied by other robots and any point of the node of the shared path is deadlock, the robot cannot go to the next point and has to wait for other robots pass. All these behaviors are controlled by the central control platform.

The motion coordination proposed in this paper is based on the nodes constructed by the quad tree. The quad tree was used to divide the simulated map into many specified areas of nodes. The waiting mode was improved by adding a new limitation condition based on the quad tree node. Under this condition, the waiting mode will only be triggered in a few cases. The quad tree is like other trees with parent nodes and child nodes. The

difference is that each parent node has four child nodes standing for four regions in the map, upper left, lower left, upper right, and lower right. Furthermore, each child region can also be divided into four regions if the system needs a higher accuracy degree. The advantage of this type of data type is that it can be simply realized and have low time complexity. As a result, each child node without any child node becomes the minimum node in the map.

For different robots, they may have different volumes in reality, which means they need different sizes of node. Thus, the designed algorithm chooses the minimum size of the node that can contain one size of the robot. For example, in the simulation of the multi-robot system, the experiment sets a 100×100 $\hat{m2}$ map and has tried several different numbers of divisions. The floor area of one robot is assumed to be about 1.5 $\hat{m2}$. Eventually, the experiment finds the six divisions are suitable for this size of the map, which means there will be totally $4 + 4^2 + 4^3 + 4^4 + 4^5 + 4^6 = 5460$ nodes in the map. Therein, there are 4096 minimum size of the node and the minimum size is that can contain one robot, so it is suitable. In this paper, the minimum precision is $(100 * 100)/4^6 = 2.44$ $\hat{m2}$ and resolution is 1.22 $\hat{m2}$ that can be adjusted by the depth of the quad tree. The algorithm assumes that the map area is S. The AGV floor area is N and the number of the division of the quad tree is n. The function can be concluded as follows:

$$\frac{S}{4^n} \geqslant N \tag{4.6}$$

Through this equation, the algorithm can calculate the maximum n value. In the next step, the algorithm needs to assign different states to those nodes and robots, so that the algorithm can be realized quickly by checking these states. The occupied state is defined as if the minimum node contains any robot or part of the obstacle, this node will be marked as occupied. A private state is defined as if the parent node of the minimum node contains any robot, this node will be marked as private. A free state is defined as there is nothing in the node. Then, the algorithm still needs to define some states for robots. An idle state is defined as if the robot has no transportation task, the robot will be marked as idle. Running state is defined as if the robot has the transportation task, the robot will be marked as running. The removal state is defined as if the robot is avoiding other robots,

the robot will be marked as removed. A blocked state is defined as if the robot is in waiting mode or it meets one robot with a state of removal, the robot will be marked as blocked.

The motion coordination is distributed in each robot and it will be executed by checking the state of the node in the control platform and the state of other nodes by communicating with them. To establish a checking standard, the robot can detect the same size of the region as the size of itself towards the direction of moving, which means each robot can only detect the next minimum node in the map towards the direction of moving. During the process of moving, the robot will constantly check the next node state and will face two states it needs to solve. If the state is Free, it can freely move to the next node. If the state is private, which means collision will take place soon, the robot will implement motion coordination. Firstly, the robot will check the other three child nodes whether are located in a straight line in the direction of the moving. If any child node is so, the robot will ignore it and check the state of the rest nodes. If the state is not occupied, the robot will change its next hop into the center point of this node. If there is not any node the robot can avoid, it will mark itself as blocked. It should be noticed that this situation would only happen as a result of other robots occupying the node because the waiting mode has avoided the occupation of the obstacle.

For the collision avoidance problem, the robot will face three different situations which are head-on collision, rear collision, and intersection collision. A head-on collision is described as two robots moving at the same time in the opposite direction. A rear collision is described as two robots moving in the same direction, but the back robot has a higher speed. Intersection collision is described as two robots will pass the same node at the same time. This decision can reduce the time of implementing the motion coordination because only the head-on collision needs to coordinate through analysis and other two types of collision the robot can only wait for others on site.

4.3.2 Simulation result

The simulation of the multi-robot system is based on the python matplotlib module. This module can create a visual graph to observe the simulation result. For each robot, the designed experiment can adjust their velocity, task and control platform. For the control

platform, it contains the node information based on the quad tree and all path information of the robots, which will be used to implement the waiting mode and motion coordination. The map is 100*100 size and contains totally 16 resource points and 3 obstacles. The visual graph is in the Figure 4.11: simulation environment. The blue points represent the resource points which are labeled, the black polygons represent the obstacles, the red point represents the robot, and the yellow points represent the parking points.

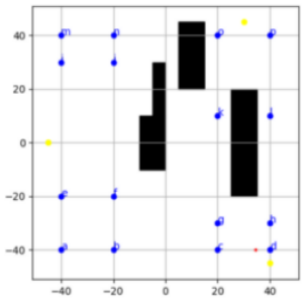

Figure 4.11: Simulation environment

The designed simulation can increase the number of robots in the system to observe the efficiency of the system. Main indexes are completed task number, waiting time and ratio which is the waiting time over the running time. To observe them constantly, each robot will get a random task after they complete a task. The completed task numbers can represent the efficiency of the whole system. The waiting time and ratio of the waiting time over the running time can represent the efficiency of the algorithm where waiting time represents the total time of waiting mode and motion coordination when there will be a collision happening. Running time represents the time of executing the tasks. The unit of waiting time and running time is seconds. Ratio means WaitingTime which is divided by RunningTime.

The experiment has done the 2-robot system, 3-robot system 4-robot system simulation tables during the 300 seconds of the experiment. For each experiment, we choose the time when each robot completes 2, 10, 20 and 30 task numbers and analyze their waiting time and running time.

Table 4.7: 2-robot system

Robot number	Task number	Waiting time(s)	Running time(s)	Ratio(%)
1	2	0	3.4	0
2	2	0	8.5	0
1	10	0	66	0
2	10	0.8	101	0.8
1	20	0.5	158	0.3
2	20	1.3	195	0.6
1	30	0.8	243	0.3
2	30	1.3	294	0.4

Table 4.8: 3-robot system

Robot number	Task number	Waiting time(s)	Running time(s)	Ratio(%)
1	2	0	4.5	0
2	2	0	11	0
3	2	0	15	0
1	10	0.6	62	0.9
2	10	0.8	88	1.0
3	10	0.8	83	1.0
1	20	2.9	182	1.5
2	20	1.5	172	0.9
3	20	1.7	145	1.1
1	30	4.3	277	1.5
2	30	3.9	270	1.4
3	30	3.3	245	1.3

Table 4.9: 4-robot system

Robot number	Task number	Waiting time(s)	Running time(s)	Ratio(%)
1	2	0	5.7	0
2	2	0	5.7	0
3	2	0	14	0
4	2	1.1	15	7.3
1	10	1.9	100	1.9
2	10	1.9	122	1.6
3	10	0.9	112	0.8
4	10	3.0	148	2.0
1	20	6.3	233	2.7
2	20	6.2	246	2.5
3	20	1.8	244	0.7
4	20	4.3	296	1.4

We also plot the line chart to compare the different systems in the Figure 4.12. The

blue line is the 2-robot system, the orange line is the 3-robot system and the green line is the 4-robot system. From this chart, we can know that the total waiting time of the system within the 300 second time frame increases about 8 seconds per one additional robot which is acceptable, and if the map is larger, this parameter will be lower.

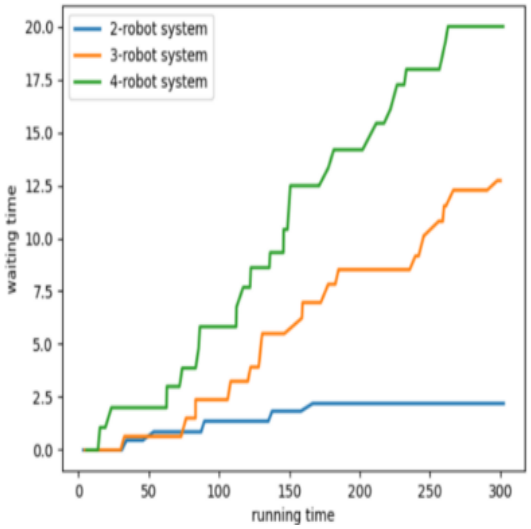

Figure 4.12: Performance Comparison

When the robot number increases, the total completed task numbers are 69, 99, and 93 respectively. It can be shown that the total system efficiency becomes saturated and waiting time does not increase more than 10 seconds with the robot number increasing. Regarding the ratio of waiting time and running time, we want the ratio value to be as low as possible, which means the majority of the time is used for the task. After analysis of the data, we find that it has a similar trend to the last index. The saturated ratio value is about 2% in the designed system. It demonstrates that each robot is executing its tasks continuously with efficiency.

In this aspect, the project proposed a new scheduling algorithm for the multi-robot system. It contains the functions of path-planning, obstacle avoidance, waiting mode and motion coordination. These functions can help multi-robot to find path, avoid obstacles, and adjust their paths. Furthermore, with the application of the quad tree, it is convenient for the robot to mark the map using the adjustable node structure. More importantly,

this strategy can not only be applied to a multi-robot system but also be used to assist other agent devices like artificial intelligence robot to achieve information marking and acquisition.

The innovation of the system design lies in the shortest path algorithm with the combination of the waiting mode and motion coordination, as well as the new motion coordination method based on quad tree division. For the shortest path algorithm, it not only has a lower time delay but also decreases the possibility of collision in the multi-robot system. The quad tree division of the map plays an important role. It has reduced computation complexity than the general graph like Voronoi graph, and is flexible for the multi-robot system.

4.4 Scheduling optimization method

A special calculator that contains two modes, basic mode, and handover mode is created to compare the time cost of the basic AGV system with the time cost of the AGV system using handover mode. After that, a simulation platform is established based on the A* algorithm and applied in specific AGV system's simulation to discuss the feasibility and actual effect of cargo transfer between multiple AGVs, and judge the differences and optimization points between different scheduling systems.

4.4.1 Time cost calculator

Two time cost calculators are created. One is called one-path time cost calculator. The other is called N-path time cost calculator. The structure of the one-path time cost calculator can be seen in Figure 4.13: The one-path time cost calculator's structure. First, a series of parameters are input in the calculator which contains the number of AGVs n, the speed V, the number of tasks M, scheduling interval T and path length L. The program will calculator how many time the system cost based on the input parameters in basic mode or handover mode. Basic mode means in one AGV occupies one path until it finishes its task. This mode can be summarized by a mathematical formula:

$$t = \sum_{k=1}^{n} \frac{2d_i}{v} \tag{4.7}$$

In this function, t is the running time of the system, d_i is the length of one path and v is the velocity of the AGV. The handover mode means there is no limit number of AGVs on one path. If two AGVs collide, one AGV will hand over the task to the other AGV and change direction. If there is no task needed to be dispatched, it will not send new AGV. If all the tasks completed, all AGVs will return towards the origin. If the system can dispatch AGV continually, there will be a formula:

$$t = T_1 t_0 + \frac{2d_i}{v} \tag{4.8}$$

In this function, t is the system time, t_0 is the scheduling interval, T is the task number, d is the length of the path, and v is the velocity of the AGVs. Due to the continuity of distribution of the system, the first term on the right of the equation gives the moment of the last AGV dispatched. The second term is the time AGV needed to run forth and back. Add both terms together can get the total system running time.

In the handover mode, as long as all tasks are not successfully delivered, it will call itself circularly until all AGVs reach the target point. The algorithm will successively calculate three types of event nodes before all tasks are completed which are regular collision node, boundary collision node, and scheduling interval node. Regular collision node means the collision between two AGVs in the path. In the handover mode, the conventional collision is the handover of tasks. The front AGV needs to hand over the tasks to the rear AGV. After the handover, the two AGVs run in the opposite direction respectively. The boundary collision node means that an AGV has successfully delivered the mission or returned to the origin The successfully delivered AGV returns directly, and the AGV returning to the origin will be added to the queue waiting for dispatch. Scheduling interval node means that when the scheduling interval reaches the T value, new tasks can be sent to AGV for transportation.

In the handover mode, the program needs to find the latest event node. Therefore, the program needs to judge which of the three event nodes is closest to the current time. If the

nearest event node is found, the calculator will update the status of the AGV participating in the event according to the definition of the corresponding event node, and update the location of other AGVs in the path. After the updating, the calculator will add the time difference between the node time and the current time to the total system time.

When all tasks are successfully delivered, the calculator will stop calling method handover. At this time, it is no longer necessary to let the AGV hand over, so all AGVs need to return to the origin. The calculator will finally calculate the time of this period and add it to the total system time. When all AGVs return to the origin, the total time of the system will be output.

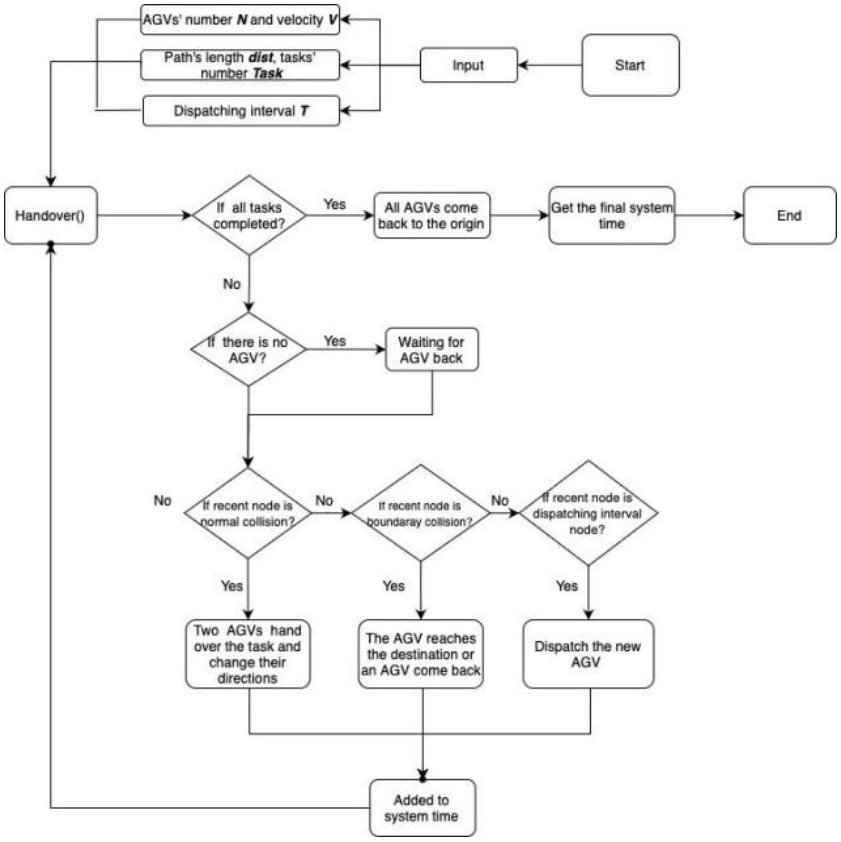

Figure 4.13: The one-path Time cost calculator's structure

The structure of the N-path time cost calculator can be seen in Figure 4.14: The N-path Time cost calculator's structure. First, the input parameters are the same as the one-path

Time cost calculator's input parameters. The number of the paths and each path's length should also be input. Because there are several paths, the program will call the handover method the smae number of times as the number of paths. When the handover method is called, a back object is defined to store the AGV information on the path. Unlike the 3 event nodes of the 1-path algorithm which are the conventional collision node, the boundary collision node and the scheduling interval node, the N-path algorithm needs to consider the AGV return on the past path to schedule the AGV for the new path. Therefore, it needs an additional event node, that is, the return event node. Then, the method will calculate a series of event nodes on this path in turn. Each event node corresponds to a time period t0. Because we need to consider the situation of the past paths, we need to call the time_handover method on other paths within t0 to update the AGV information on the past paths.In the time_handover method, it needs to receive the sequence number and time period t0 of the parameter path and updates the AGV's information within this time period. Considering of the fact that the task has been dispatched, there is no need to consider the scheduling interval node. Therefore, only the regular collision node and the boundary collision node are necessary. The calculation principle is the same as 1-path. When the task dispatched by each route meets the requirements, it will exit the loop and store the back object in the dynamic array to update the path information. After that the program will continue to call the handover method until n times. After calling n times, the program will wait for all AGVs to complete the task. When all the tasks arrive at the destination, there are no tasks on the path at this time. Therefore, the handover mode stops at this time, and the program will make all AGVs return towards the destination. Finally, when all AGVs return, the N-path Time cost calculator records the time and outputs the system's total running time.

In case of handover time is zero second, the scheduling priority principle shall be met in any case. That is, the longer the path, the greater the scheduling priority, so as to meet the minimum running time of the system. When the scheduling system can send AGV continually, there is a mathematical formula to calculate the system time:

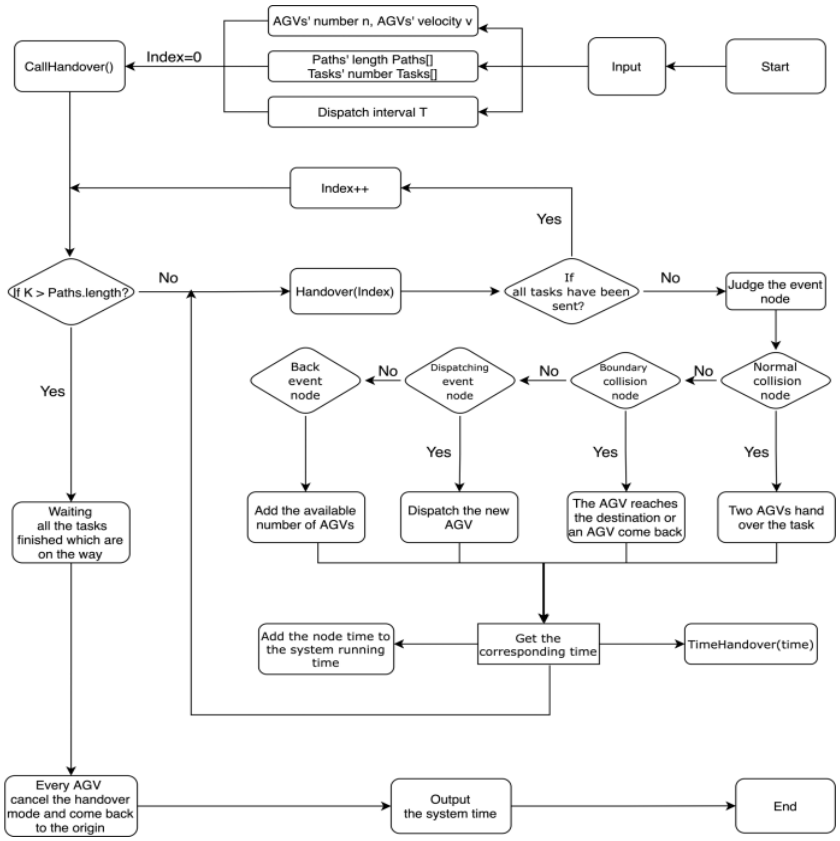

Figure 4.14: The N-path Time cost calculator's structure

$$t = max\{T_1 t_0 + \frac{2d_1}{v}, (T_1 + T_2), ..., t_0 \sum_{k=1}^{i} T_i + \frac{2d_i}{v}, ..., t_0 \sum_{k=1}^{n} T_i + \frac{2d_n}{v}\} \tag{4.9}$$

In this special case, it is easy to get what is the final AGV come back to the origin. In this formula, each term means the last moment of an AGV comes back to the origin. Ti is the number of task of each path, di is the length of each path, t0 is the scheduling interval, v is the velocity of AGVs, and t is the system time. The goal is to find out the moment each path's AGV which is the last one comes back to the origin. Due to the continuity of the dispatching, as for path of index i, the last dispatching moment is t0(T1+T2+...+ Ti-1) and the time that the AGV needs to come back is 2*di/v. The return time of the last AGV on each road can be obtained by adding the two terms. The maximum time is

the total time of the system.

4.4.2 Simulation platform

The simulation platform is established based on J. T. Lin and M. Hsyu's design [162]. As shown in Figure 4.15: Structure of the simulation platform, the meaning of the parameters in the figure are as follows:

Center (class): It is used to initialize a scheduling system. It contains several parameters which are the number of AGVs, the number of workstations, AGV navigation algorithm, paramters that can define the used path rules, etc Process(navigation()): All moving parts, such as AGVs, can be simulated with Process. Therefore, a separate process, Process(navigation()), is used here to schedule the work arrangement of each AGV.

VehicleID(): It is a method of assigning new tasks to AGV. It will schedule a new task path arrangement for AGV. It includes the current path to be executed, vehicle.path and the path to be executed in the next stage, vehicle.nextpath.

Job_arrive(Periodic entry): This method will randomly generate a task type and add it to the task list of the scheduling system. Meanwhile, this is a periodic method which will be called repeatedly after a certain time interval. createJtype:This method is built into the Job_arrive.When the Job_arrive is called, a new task type will be generated through this built-in method. Similarly, using an independent progress to call flow() to record the completion status of the task type and call WorkstationID() in time.

WorkstationID(): This method will be called after the workstation completes the processing task, and will require the scheduling system to assign one or two idelw AGVs to load the finished goods.

When the simulation platform starts, an user can specify the number of workstations in the current scheduling system, the number of AGVs and the algorithm for picking the shortest path when the AGVs are running. After the timer has been set, the system begins a periodic cycle of adding new task target to the task list. In order to mimic as closely as possible the daily work scenario of a production plant, the tasks are designed in such a way that they contain in turn all the workstations that need to handle the goods. When the task is first posted, the scheduling system assigns an idle AGV to pick it up. It needs

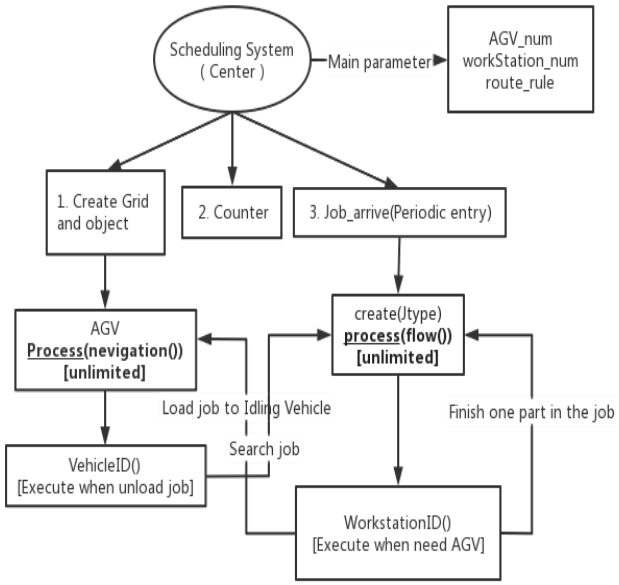

Figure 4.15: Structure of the simulation platform

to load the goods at the first station in the task and then travel to the second station to unload the goods. At this point, the second station needs time to process the unloaded goods, while the completed AGV awaits reassignment from the scheduling system. After a period of processing, that station updates the current status of the task and re-issues it to the task list. Then the scheduling system assigns an idle AGV to pick up the task again and, depending on the current status of the task, loads the goods from the second station and goes to the third station to unload them. This is done until all the requirements of the task have been completed.

When the AGV performs the movement operation in the system, it is necessary to mark each step of the AGV's movement. It means when the AGV performs the movement operation, it first needs to make an arrival application for the intersection of the table path in the scene. The reachable intersections in the scene can be described by a concise two-dimensional list I:

$$I = \{[x], [y]\} \tag{4.10}$$

In function 4.5, x refers to the horizontal axis coordinate point of the two-dimensional list, and Y refers to the vertical axis coordinate point of the two-dimensional list. Therefore, every time the AGV moves to the next intersection in the scene, it can add a request instruction for its next target point, so that other vehicles can judge its next action arrangement.

When the scheduling system schedules the operation of AGV according to the task list, the current movement request has the necessary target point and starting point. Meanwhile, the current AGV will start to perform transportation operations. A mobile request T can be described with concise symbols:

$$T = \{p\,(T)\,, d\,(T)\} \tag{4.11}$$

p (T) is all the reachable intersection points in the scene, except the points blocked by the workstation. d (T) is the delivery point of all workstations including warehouse and charging station.

Each newly created task is a new process instance, i.e. the process will continue to loop until the task is completed. Each time it is updated, the scheduling system looks for a new idle vehicle by means of the workstationID function. In addition to this, each AGV has its own process instance, which calls VehicleID function in a loop, in order to ensure that the AGV's movement path is updated and to avoid vehicle conflicts when they occur.

The scheduling system for this project is shown in Figure 4.16: Live demonstration. The blue Time label in the upper left corner records the current simulation time in the event-based simulation execution environment, where the passage of time is simulated by stepping in from one event to another. The six AGV machines are represented by purple marked points such as A1 and A2, and the 10 distribution points (represented by orange rectangles Workstation1, Workstation2, etc.) in the two-dimensional plane are used for parcel scheduling distribution. Considering the power loss of the AGVs, a blue charging station is set up to realise the energy replenishment of the AGVs. It is also the idle waiting

point and starting point for the AGVs. In addition to this, the lower left corner of the system contains a yellow rectangle of the main goods storage room, with a blue dot and a green dot representing the loading and unloading points of the storage room respectively.

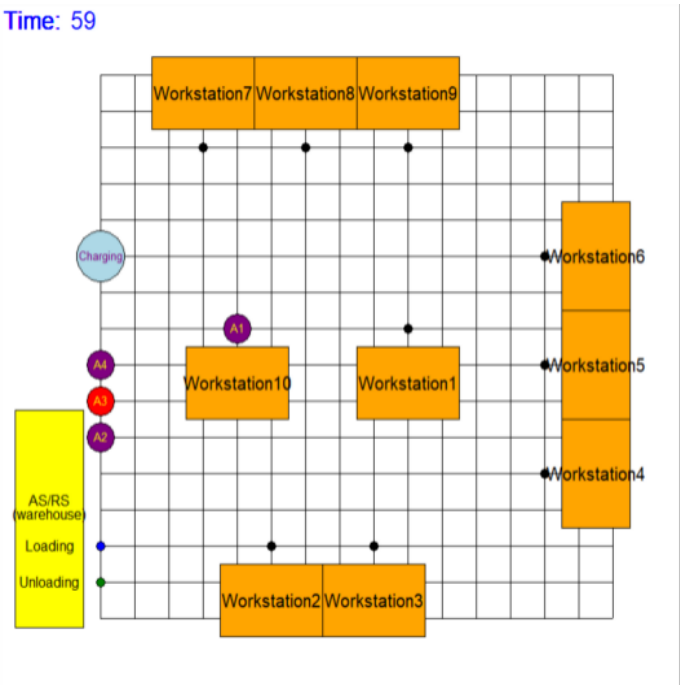

Figure 4.16: A screenshot of the Live demonstration

The tasks to be handled are randomly added to the task system with the target workstations that need to be visited in turn. Under the overall assignment of the scheduling system, the idle AGV picks up the task and starts the handling work. When the AGV is carrying a load, its colour changes to red. The vehicle will enter the charging station in two situations:

(1)Once the vehicle's charge level has fallen below a certain level.

(2)When the AGV is idle after unloading the goods and there is no demand for new task scheduling at the moment. The vehicles travel on a grid-like network-guided path, with the vehicles travelling in the following directions: up, down, left and right. The scheduling system then continuously cycles through the release of tasks and simultaneously modifies the task statues of each AGV before the cut-off time, dynamically adjusting the scheduling

arrangements as a whole. This enables the individual AGVs to continuously maintain optimal working conditions until they reach their minimum power level, maximising the overall scheduling efficiency of the system.

The application of the handover algorithm means when AGV2, after loading its cargo, meets the empty purple AGV3, which has also gone to load its cargo, AGV2 passes its cargo to AGV3, which will take a certain amount of time, so that AGV3 moves instead of AGV2 towards the workstation of the current task demand and allows AGV2 to reassignment by the scheduling system. In addition to this, the use of the handover algorithm can also occur in scenarios where an unloaded AGV intersects with a loaded AGV on the way back after unloading the goods, for example. In short, an empty vehicle in purple is able to perform a handover operation when it intersects with a loaded red vehicle. The optimization of the scheduling system and the possible disadvantages of this dispatching method will be described in the next section.

Apart from this, it is most important to have information about the status of the AGV, including its current position, operational status (standby, start, stop, low voltage, etc.), current operational target or the status of the docked equipment, etc.

Task allocation is the beginning of the scheduling system. In order to efficiently allocate tasks to the appropriate AGVs. It is necessary to establish appropriate task identification, determine the priority of different tasks, identify the execution status of tasks (being executed, already executed, waiting for execution, cancelled, etc.), record the execution process of tasks, etc. Three modes are raised to optimize the multi-AGV system.

One is called basic mode. The work processing required for goods is proposed periodically by the scheduling system. The task system responsible for task allocation uniformly collects and assigns AGVs for scheduling. Among them, an AGV follows the algorithm of optimizing the shortest path in the process of executing tasks, that is, it traverses all known tasks in the task system, and selects the one with the shortest distance from its own required movement from these tasks for cargo transfer operation. In the process of path finding, an AGV uses A* algorithm for path planning.

Each transportation task contains a list of all stations where the goods need to be processed, as well as the time required for the station to process them. This means that the

goods need to be transported to each station in turn and processed at the corresponding time. This task design restores the needs of finishing the corresponding parts of products on different worktables in the actual factory. When the AGV completes a round of transportation demand according to the task demand (a round of transportation demand refers to loading goods from the current station in the task demand and transporting them to the next station for unloading), the system will not make new route arrangement for the AGV in time. In other words, when unloading goods is completed, if there is no task to be performed in the task system, the AGV will wait at the station by default until the goods are processed at the workstation. After that, the AGV will continue to pick up goods from the site and transport them to the next site for processing. If the AGV encounters another AGV carrying goods to this work station during the waiting period of the workstation, the AGV in the waiting state will be reassigned by the scheduling system. For example, reassign the AGV in the idle state to a workstation that has completed the processing of a certain goods, reload the goods, and continue to transport the goods to the next station according to the task list of the goods. All transportation tasks are regarded as an independent process until all the requirements are completed. The sign of completion of any transportation task is: transport the target goods back to the 11th unload point of warehouse. At this time, the system records the cargo throughput plus one.

In general, when the process starts, the scheduling system continues to add new requests for goods processing, including the work site to which the goods need to arrive. After that, the scheduling system reasonably assigns each task to the idle AGV, which follows the shortest path principle and A* algorithm to transport goods. Each cargo handling request will run independently as a process until all requirements are met. This process will actively record the following data: the current work site to arrive, the next work site to arrive, the processing time required by the workstation, etc. After completing each transportation task, the scheduling system records the throughput, and finally ends the process of the transportation task.

One is called basic no idling mode. The basic theory is consistent with the basic mode, but the difference is that when the cargo is unloaded, the AGV will immediately carry out the next movement operation instead of waiting in place. If there is no waiting task in the

task system at this time, the AGV will return to the charging station by default. This mode is proposed to compare the difference between the cargo throughput gap caused by the AGV working at full load and the relatively idle condition.

One is called handover mode. The basic theory is consistent with the basic mode, but the difference is that when the AGV collides with other AGVs in the way of movement, it will give priority to a handover judgment. After delivering the goods, the new loaded vehicle continues to complete the handling of the goods. The empty vehicle is redistributed by the scheduling system according to the principle of the shortest path. Of course, the target task of the original empty vehicle will be re recorded to the scheduling system and wait for the scheduling system to redistribute a new idle vehicle to pick up the task.

Furthermore, these modes are based on the A* algorithm to plan the route for the AGVs. The A* algorithm was chosen for the following reasons.

1.A* only focuses on the shortest path from point to point.

2.For the current point on the path, the A* algorithm not only records its cost to the source point, but also calculates the desired cost from the current point to the target point, which is a heuristic algorithm.

3.Fewer intermediate nodes need to be extended when searching for the target node, so the algorithm is more efficient.

The A* algorithm is based on Dijkstra's Algorithm, which is optimised for a single destination, taking into account different movement costs. Although Dijkstra's Algorithm can find paths to all locations, A* finds the closest one to a location or several locations. It gives priority to paths that appear to be closer to the goal. In contrast, although Dijkstra's algorithm performs well in finding the shortest path, in the case of multi-AGV scheduling, where multiple AGVs need to be considered simultaneously, using Dijkstra's algorithm wastes valuable computational resources and computational time by exploring in many less promising directions, which is undoubtedly fatal and frightening for the optimization of the entire scheduling system. Similarly, while heuristic search streamlines the exploration directions, for example in greedy best-first search, we will use the estimated distance from the starting point to the goal for priority queue ordering. The closest position to the goal is explored first. However, when faced with a practical situation with many obstacles, it

is difficult to obtain the optimal path, although the computation is fast.

The use of the A* algorithm does not lock its eyes on the task of finding the shortest path, it also takes into account the estimated distance from the starting point to the goal, thus paving the way for the multidimensional situation that needs to be considered during the AGV path planning process. In other words, problems such as deadlock blockage encountered during AGV movement can no longer be reasonably solved by simply planning the shortest path. We can add other conditional functions to the A* algorithm to reasonably predict the time, such as the waiting time at the target station, in order to optimise the scheduling arrangements for the AGV. In summary, A* can be a more suitable algorithm for route planning for multi-AGV scheduling systems [163].

4.4.3 Experimental results and discussion

We conduct experiments to verify the following:1) The handover mode AGV system has better performance than traditional AGV system. 2) The designed calculator and simulation platform can work properly and have potential for further experiments.

By inputting a series of parameters are input in the calculator which contains the number of AGVs N, the speed V, the number of tasks M, scheduling interval T and path length L, the time cost calculator can calculate the system's time cost when all tasks are finished and all AGVs return to the starting point. The time cost calculator assume there is only one path in the AGV system. In basic mode, only one AGV is necessary for the system. Meanwhile, the handover time is set to 0 in the time cost calculator. By changing one of the input parameter and keeping other input parameters be stable, it is easy to find the influence between the parameters and the time cost of the system.

In the fist testing I : Basic mode compared with Handover mode, we increase the departure interval of each AGV. The longer the departure interval become, the longer the system's time cost will be. In the second testing II : Basic VS Handover AGV number, we increase the AGVs' number from 1 to 10. It can be found, a multi-AGV system in a handover mode, when each AGVs' velocity, each AGV's departure interval, running distance and task number of the system do not change, after increasing the AGV's number in a certain extent, the system's time cost will not change. In the third testing III: Basic

VS Handover Tasks, we increase the task number from 4 to 16. The system's time cost is positively correlated with the task number. In the forth testing IV: Basic VS Handover Length, we increase the path's length from 1 to 200 meters. The system's time cost is positively correlated with the path's length.

In Figure 4.17: One-PATH time cost calculator, it can be seen that in a multi-AGV system using handover mode each AGV's scheduling interval time is correlated with the system's time cost. When other parameters are stable, the number of tasks will not affect the time cost of a multi-AGV system to a certain extent. Each AGV's velocity is the most important parameter which can affect the multi-AGV system's time cost. With the growth of each AGV's velocity, the time cost of a multi-AGV system decreases with the explosive exponential reduction. Furthermore, there exists a best number of AGVs in a multi-AGV system using handover mode. When the number of AGVs achieves such value, it is useless to improve a multi-AGV system's efficiency by increasing AGVs' number.

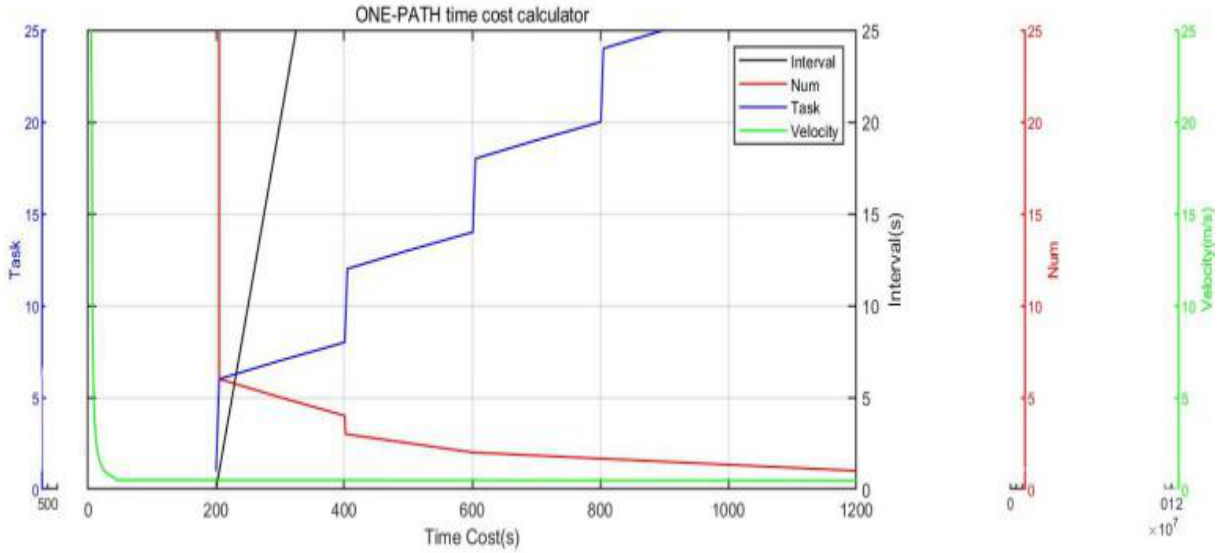

Figure 4.17: Parameters relationship in One-PATH time cost calculator

In Figure 4.18: N-PATH time cost calculator, what is worth being mentioned is that the figure is created based on three paths whose length are separately 10 meters, 30 meters

and 50 meters. There are six tasks in each path. First, we make the number of AGVs be 6 and make each AGV's velocity be 1m/s. Then we change each AGV's interval time to find the relationship between each AGV's interval time with the multi-AGV system's time cost. Second, we keep each AGV's velocity and each AGV's interval time be a stable value. Then we change the number of AGVs to find the relationship between AGVs'number with the multi-AGV system's time cost. Third, we keep the number of AGVs and each AGV's interval time be a stable value. Then we change each AGV's velocity to find the relationship between each AGV's velocity with the time cost of a multi-AGV system. The conclusions based on the N-path time cost calculator can be summarized the same as the ONE-path time cost calculator.

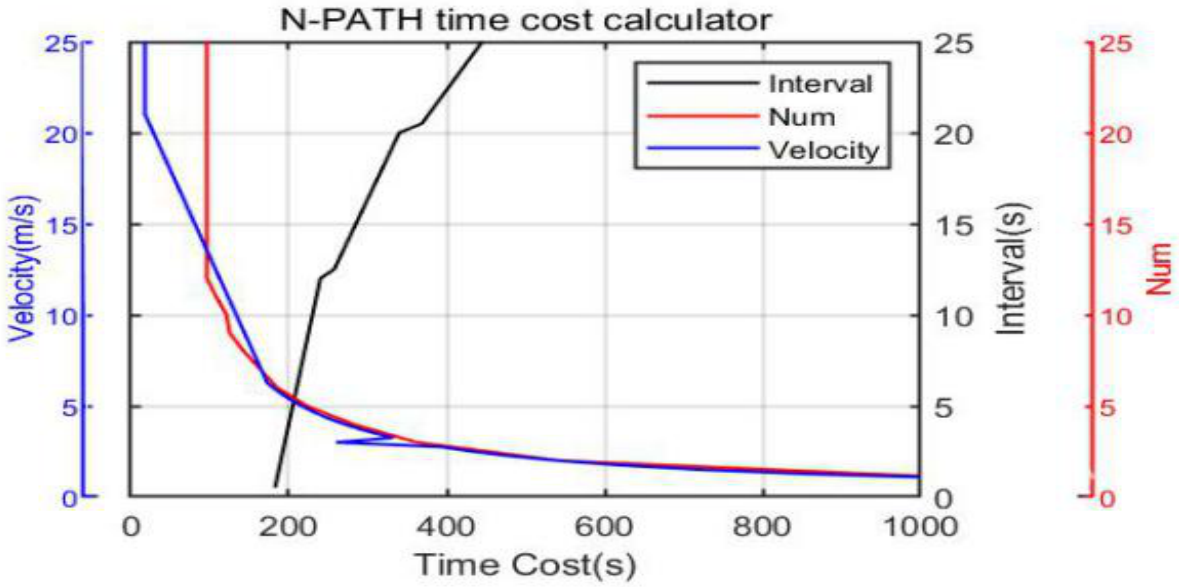

Figure 4.18: Paramters relationship in N-PATH time cost calculator

In summary, several conclusions can be summarized:

(1)Each AGV's scheduling interval time is correlated with the system's time cost.

(2)When other parameters are stable, the number of tasks will not affect the time cost of a multi-AGV system to a certain extent.

(3)Each AGV's velocity is the most important parameter which can affect the multi-

AGV system's time cost. With the growth of each AGV's velocity, the time cost of a multi-AGV system decreases with the explosive exponential reduction.

(4)There exists a best number of AGVs in a multi-AGV system using handover mode. When the number of AGVs achieves such value, it is useless to improve a multi-AGV system's efficiency by increasing AGVs' number.

However, the designed calculators still have some limitations. For example, it assumes a relative ideal environment. The handover time is set to 0. Although this time cost calculator helps to prove that compared with basic mode, a system using handover mode can have better performance, more complicated environment should be considered.

As mentioned above, we create three modes in the simulation platform which are basic mode, basic idling mode and hadover mode. To judge each mode's performance, we set the number of the AGV and the time be the same. Then we run the simulation platform and check how many tasks are finished. Meanwhile, to simplify the experimental design of the scheduling system, the following assumptions are considered for the AGV scheduling system.

1.The loading times of the AGVs are all constant values.

2.The speed of the AGVs is a constant value.

3.The operation of the vehicles will not be affected when each vehicle holds different objects.

4.Collisions between vehicles are not allowed, which means that different AGVs cannot occupy common nodes of the inflow path at the same time.

5.Failure of AGVs as well as machines is not considered for the time being.

Because of the A* algorithm and the task list can be changed, each mode has different performance. Therefore, we fist test how many tasks can each modes finish in 1500 seconds. We establish the simulation platform as 10 task points are randomly selected from 10 task points for each handling task. There are six AGVs in the simulation. The handover mode finish the most tasks at the same time.

Then, we change the time. We test each mode's performance ten times at the same time to get an average value. After that Figure 4.19 : Testing results is established. It gives an illustrative example to demonstrate how different modes perform in different situations.

Figure 4.19.1 makes 10 AGVs run to 10 workstations to finish the tasks in 700 seconds, 900 seconds,1100 seconds,1300 seconds,1500 seconds and 1700 seconds. Figure 4.19.2 makes 10 AGVs run to 10 workstations to finish the tasks in 700 seconds, 900 seconds,1100 seconds,1300 seconds,1500 seconds and 1700 seconds. Figure 4.19.3 makes 10 AGVs run to 10 workstations to finish the tasks in 700 seconds, 900 seconds,1100 seconds,1300 seconds,1500 seconds and 1700 seconds. Figure 4.19.4 makes 10 AGVs run to 10 workstations to finish the tasks in 700 seconds, 900 seconds,1100 seconds,1300 seconds,1500 seconds and 1700 seconds. Figure 4.19.5 makes 10 AGVs run to 10 workstations to finish the tasks in 700 seconds, 900 seconds,1100 seconds,1300 seconds,1500 seconds and 1700 seconds.

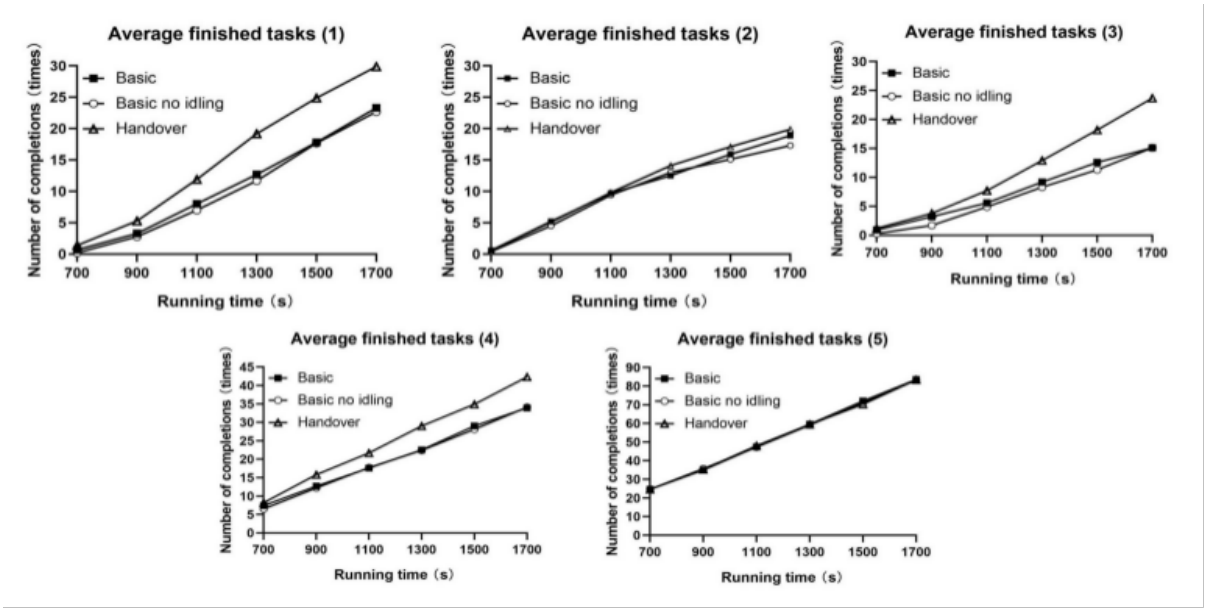

Figure 4.19: Testing results

In summary, by using handover method in a multi-AGV system, the system's performance can be improved. However, these conclusions obtained so far are analysed in the overall context. This is because when looking deeper into the data from each experiment, we can see that not all cases in this setting (with different task release intervals) show negative optimization when the handover algorithm is added. In fact, when a task system releases a task at a suitable time for the Handover algorithm, it also makes that scheduling

system outperform other scheduling systems. Secondly, as the Handover algorithm makes it necessary for the AGV to have handover time when handing over the goods and for the workstation to have processing time when handing the goods, the choice of these times will also affect the final experimental results.

We present two multi-AGV time cost calculators and a multi-AGV scheduling simulation platform. A scheduling optimization method is also proposed. Related experiments and testings have been done. The superiority of the proposed scheduling optimization method has been proved. Meanwhile, parameters can influence the system's performance have been tested. The handover mode is better than basic mode. Most parameters are correlated with the system's time cost except the number of AGVs. There exists a best number of AGVs in a multi-AGV system using handover mode.Therefore, the handover method should be paid more attention to be applied in multi-AGV systems' scheduling optimization area.

The future work is to expand our work. We plan to optimize the current time cost calculator so that it can deal with the N-path situation. Meanwhile, we plan to add a parameter handover time to the time cost calculator with other specific parameters to better test what will influence a multi-AGV system and how can they influence the system. We plan to optimize the current simulation platform so that it can simulate and schedule robots with different capabilities, apply different path planning and obstacle avoidance algorithms, and add more specific conditions to make the simulation environment close to the real scene. Thus, the deployed robot system can be tested with more applicability.

Chapter 5

Visual tracking method

In the aspect of computer vision in AGVs, I am focus on combing multiple sensors, especially with optimized innovations in computer vision algorithms. An improved camshift algorithm was created. Furthermore, several deep learning models were tested to analysize revealed different features of the three models in implementation.

5.1 An improved visual tracking method

Kalman filter algorithm is regarded as one of the most famous Bayesian filter theories which is a linear optimal status estimation method [164]. For the tracking of a real-time target, the Kalman filter algorithm ensures a tracking solution to be optimal for different real-time tracking objects if essential parameters are defined. The Kalman filter can ensure ideal estimates based on the data available given confirmable and statistical properties of the system parameters and measurements [165]. Popular variations of Kalman filters include extended Kalman filter (EKF) and un-scented Kalman filters (UKF).

In this paper, a conventional Kalman filter is applied to predict the tracking target. The tracking algorithm takes into account the dynamic nature of the background. It decreases negative effects of the environment such as light and similar backgrounds in order to improve the tracking performance. It is been combined with the Camshift algorithm for AGV visual tracking in a real-time processing system at the edge computing environment.

The key algorithm is summarized as follows.

Step 1: The initial values of the parameters are defined in advance. For example, the Gaussian white noise covariance of measurements and estimates are supposed to be stable at the time while the frames in the tracking process are changing. This is compatible with the concept of the Kalman filter algorithm [166].

Step 2: By manually selecting a designated tracking target, we obtain the coordinates of the central point for the selected tracking area (x, y), frame window width w and length h. It means the user can click the first frame to create a window, then the program will run.

Step 3: Start the measurement. Use Kalman filter to predict the location of the target window. Current frame of the tracking process is defined as k. Based on the previous frame of tracking process, the frame of the tracking process can be predicted.

Step 4: With frames being predicted, it is time to collect the current measured frames. According to the converged predictions, images are truncated to the predefined the window size (w and h). Combining predicted frames and the measured frames with windowing, Camshift is applied to calculate optimal location and size of the target at the time of k.

Step 5: Update the covariance under the state of k. The system then operates recursively from k to k+1.

The prototype system contains a camera, Raspberry Pi 3B+, a pre-designed program, and sensors for obstacle avoidance. A laptop is regarded as the platform in the simulated edge environment. Images are collected by the camera of the AGV node. The images are transmitted to the platform (edge) through wireless data transmission (WiFi). At the edge, these images are analyzed by an improved Camshift algorithm. Decisions on tracking actions are then sent to the AGV end node to control its movement. The whole process can be summarized in Figure 5.1.

5.1.1 Target tracking experiment

The experiment involves the detection and identification of a specific object. The rigid object is used to characterize the constant characteristics of the pixel when moving. The convergence area of target pixel is calculated for each frame in the video stream. The

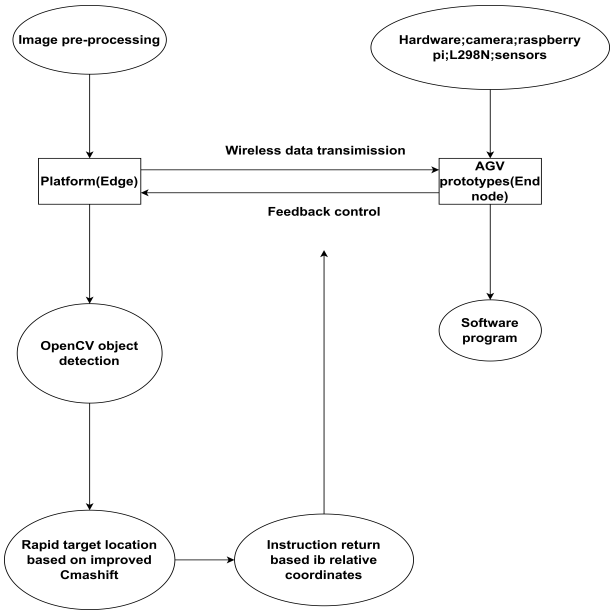

Figure 5.1: The working processes of the system

central area of the target pixel aggregation is then obtained achieving real-time tracking. At the beginning of the tracking function, the PC sever needs to set the receiving mode and wait for the Raspberry Pi to transmit the image, which is shown in Figure 5.2.

```
now waiting for frames...
Waiting for client connection
Waiting for detecting the target...
C:\Users\Administrator\source\repos\xiaochezhuizong\xiaochezhuizong\xiaochezhuizong.py:110: DeprecationWarning: time.clo
ck has been deprecated in Python 3.3 and will be removed from Python 3.8: use time.perf_counter or time.process_time ins
tead
  start=time.clock()
```

Figure 5.2: The edge waits for the image from end node

When the first frame is transmitted into PC sever, the target will be detected by 'multiscaledetect' function based on OpenCV. When the target starts to move, its coordinate center in the image will also change. By comparing the current target coordinate with the preset value. When the target is moving, the center point of the tracked target is at

the left of the image and exceeds the preset value, so the left-turn command is executed. As for the forward and backward commands, they operate by judging the height of the selected area and the preset value. Since the tracked object is rigid, its height will not change too much when it moves, so it can be used as an index to judge the distance of the car. The color information of the target can be used as the condition of convergence area after the edge node recognizes the target. The coordinates of the target in the image can be locked through the convergence calculation in each frame, so as to send corresponding instructions to the edge node. In addition, by comparing the size of the positioning box with the preset value, the AGV prototype can be kept within a certain distance from the target and follow it in real-time.

The accuracy and robustness of designed improved Camshift algorithm is tested by experiments with the AGV prototype. Another AGV is regarded as the tracking target (the one in blue mask). The third AGV with a similar blue color is added to the scenario as the disturbance imitating dynamic environmental factors.

Except the combined Kalman filter with Camshift algorithm, the improved algorithm also uses a selecting window method. A selected tracking area is determined which is based on the tracking target. This area is selected in the first frame. Once this area is defined at the initial stage, the rest of the area would be painted white or dark which decreases the interference from the background environment. The adjusted algorithm with simple pre-processing works well for this single target AGV tracking problem. Figure 5.3 and 5.4 demonstrate graphically the different tracking performance of original Camshift and improved Camshift tracking schemes. Figure 5.3 (c) and (d) showed inaccurate tracking scenarios. Figure 5.4 demonstrated accurate tracking during the course of movement. The intermediate steps with windowed images are enclosed at the bottom for illustration.

More numerical analysis results are detailed in Table 5.1, Table 5.2 and Figure 5.5. Table 5.1 shows the true and predicted coordinate points at the center of the frames in 15 frames. It demonstrates the accuracy of the improved Camshift algorithm. The indicated x and y coordinate points are the central points of the tracking area in the frames. To ensure the tracking area being a rectangle of pixels, values of predicted pixel points are rounded down. According to the data in Table 5.2, it can be concluded that the improved

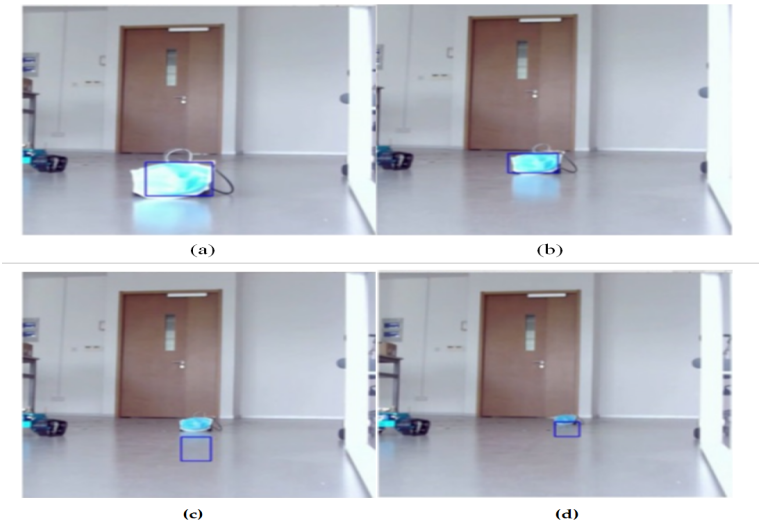

Figure 5.3: Inaccurate target tracking with original Camshift algorithm

Camshift algorithm has a high accuracy in tracking.

Table 5.1: True point vs Predicted points with improved Camshift

True points(X)	Predicted points(X)	Difference(X)	True points(Y)	Predicted points(Y)	Difference(Y)
666.16364	666	0.16364	475.47833	475	0.47833
666.75586	666	0.75586	475.16827	475	0.16827
667.10645	667	0.10645	474.9796	474	0.9796
666.3836	666	0.3836	473.88486	473	0.88486
667.3559	667	0.3559	473.60944	473	0.60944
667.9055	667	0.9055	473.03366	473	0.03366
667.3961	667	0.3961	472.95413	472	0.95413
667.02045	667	0.02045	473.87555	473	0.87555
666.3877	666	0.3877	473.47754	473	0.47754
666.74384	666	0.74384	473.76678	473	0.76678
668.0377	668	0.0377	473.33975	473	0.33975
668.94934	668	0.94934	472.69196	472	0.69196
668.93286	668	0.93286	471.37817	471	0.37817
669.26434	669	0.26434	470.90308	470	0.90308
669.26154	669	0.26154	470.92227	470	0.92227

The experiments also demonstrated reduced time delay and improve stability in tracking. Traditional Camshift algorithm processes the whole image captured by the camera. The calculated information entropy of the image is about 6.49856. The designed improved Camshift algorithm with the windowing function focuses on a portion of the image. The updated image after preprocessing has a reduced entropy of 1.80633. Windowing saves

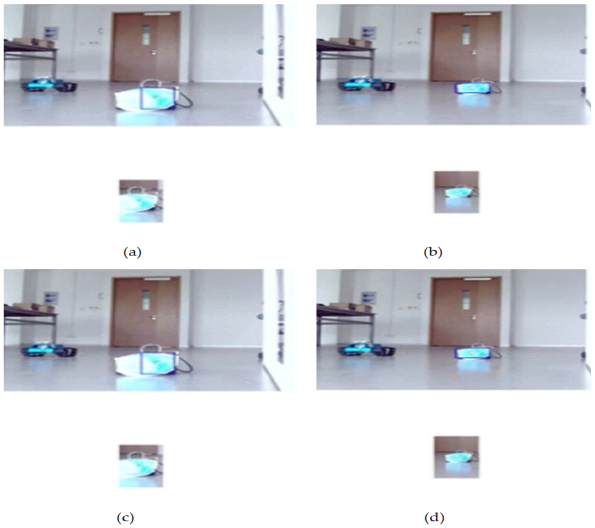

Figure 5.4: Accurate target tracking with improved Camshift algorithm

the processing time by reducing the convergence time, and leads to less time delay. The reduced entropy leads to the stability of the time delay among frames as shown in Figure 5.5: The time delay of original Canshift and improved algorithm are illustrated.

Table 5.2: Time Delay of Camshift VS Improved Camshift

Time Delay (s)	**Camshift**	**Improved Camshift**
Mean	0.006877671	0.005987875
Variance	9.56119E-7	4.61390E-7

As shown in Table 5.2, the original Camshift algorithm has an average time delay of about 6.6406 ms. The improved Camshift algorithm has an average time delay of 5.9879 ms. It achieves an average of 12.94 % less time delay. Meanwhile, the improved Camshift is more stable with 51.74% reduced variation of the time delay as shown in Figure 5.5.

5.1.2 Discussion

The key features and improvements of the experiments are summarized below.

1)Essential parameters are first defined which include statement numbers, AGVs speed,

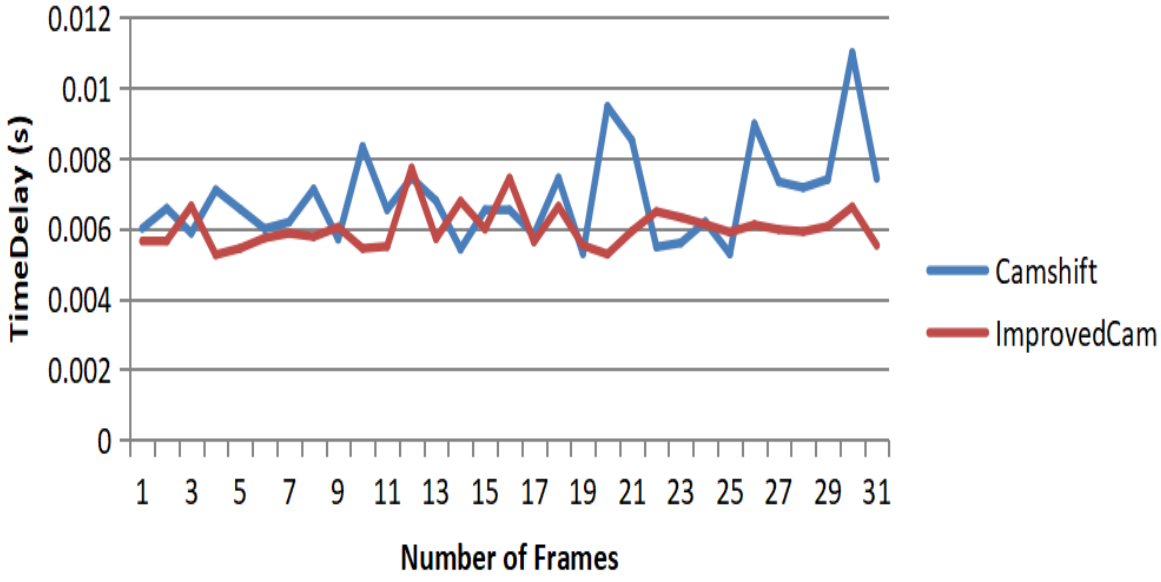

Figure 5.5: Time delay of Camshift and improved Camshift

observed quantity, system measurement matrix, state transition matrix, and noise covariance.

2)Enabling the camera, the program based on improved Camshift algorithm starts. In the traditional Camshift algorithm, the next frame is detected from the detection center of the previous frame by Meanshift algorithm. However, the target is in a moving state, and when the background color overlaps with the target color, the tracking may shift. Therefore, the improved algorithm uses Kalman filter to predict the coordinates of the moving target, and to predict the position of the target in the next frame.

3)By setting a selection window around the predicted coordinates, the background pixels around the target can be removed, which reduces the possibility of false tracking transfer. Combining Kalman filter and windowing algorithm result in reduced tracking time delay of 12%. It improves the stability and robustness of target tracking with performance gain of 51.74% due to the reduction of complex background interference.

From the test results, it can be demonstrated that the color information of the target can be used as the condition of convergence area after the edge recognizes the target. The

coordinates of the target in the image can be locked through the convergence calculation in each frame. Corresponding instructions can be sent to the end node. In addition, by comparing the size of the identified target area with the preset value, the Raspberry Pi AGV can maintain a specified distance from the target.

As the speed of the target AGV varies along the movement,the tracking performance can be affected by the distance between the prototype tracking AGV and the target, the shape and color of the tracking target, and the complexity of the background.

In the designed experiments, another experiment was proposed. In that experiment, the tracked object is a blue AGV. The tracked object had a original speed. The tracker can move and its speed can change based on the improved camshift algorithm in it. Deformation and scale variation will happen during the tracking progress. Figure 5.6: Dynamic test shows one frame when the tracker tracked the target. Tests have been made regarding different tracking distances and preset window sizes. The results are shown in Table 5.3. Those frames are selected from a testing video which contains about 1600 frames. We compare those frames to judge whether the improved the camshift algorithm has better performance. Based on the results, the camshift algorithm has higer accuracy.

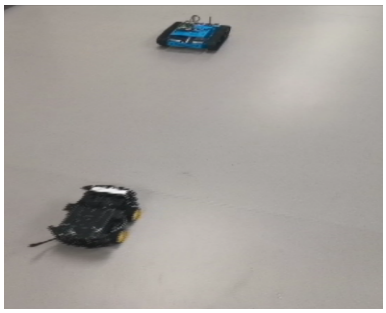

Figure 5.6: Dynamic test

Table 5.3: Time Delay of Camshift VS Improved Camshift

Distance (cm)	**Preset window size(cm)**	**Actual width(cm)**	Max speed of target(cm/s)
105	30*30	110	40
90	35*35	90	40
75	40*40	80	35
45	45*45	45	20

In the experiments, the maximum speed of target is 40 cm/s, the frame per second is 30 f/s and the processing time is close to 0.06 s. In dynamic environment, the background is different in each frame and change rapidly, so the processing time increased. The max speed in testing needs to be less than one half the actual width. The calculation formula is given below.

$$processingframe = fps * processingtime = 2 * frame \tag{5.1}$$

$$Maxspeed * 2 \leqslant actualwidth \tag{5.2}$$

Therefore, a small default window can be set to make the tracking distance longer when fast moving objects need to be tracked. Moreover, if it needs to drive on a specified route, a large default window should be set up for close tracking.

A prototype AGV tracking system was built and tested in the edge computing environment. A new tracking algorithm was proposed which was based on the improved Camshift algorithm combining Camshift, Kalman filter, and windowing algorithm. The designed system tackles several unsolved problems in the original Camshift algorithm for AGV tracking including adaptive color tracking, interference cancelation under dynamic background, and maintenance of constant tracking distance. The experimental results demonstrated reduced tracking time delay of 12%. The stability and robustness of target tracking have a performance gain of 51.74%. The system architecture provides a solution to balance the limited computation power of individual AGVs and the requirement of computation-intensive image processing for AGV tracking. The experimental results also

revealed the challenges encountered in the current AGV tracking system design including speed limit and complexity in the color/shape of the target, which is worth further investigation.

5.2 Deep learning visual tracking models

In this section, we provide clear implementation procedures to verify the anti-occlusion performance of the three models for single target detection. Specific data sets are generated to facilitate the algorithm evaluation. The format of the data set is Visual Object Class (VOC) type. The images for training are imported to labeling - a deep learning annotation tool embedded in Anaconda 3. The ground truth anchors, including the classes and coordinates, are drawn manually. All images are collected from the different distances and angles of the target robot. Some images are obtained under unclear lighting and noisy conditions.

A computer programme is designed to read the information of the ground truth anchors from XML files and generate two txt files for training and validation. An example can be seen in Figure 5.7: Manual annotation of the object in Labeling. The graphical user interface contains three key areas which are tool bar, labeling and file list. A toolbar is located at the edge of the screen, and the image waiting to be labelled is displayed in the center of the interface. There is a file list at the low right corner edge. It illustrates the file information of the selected images.

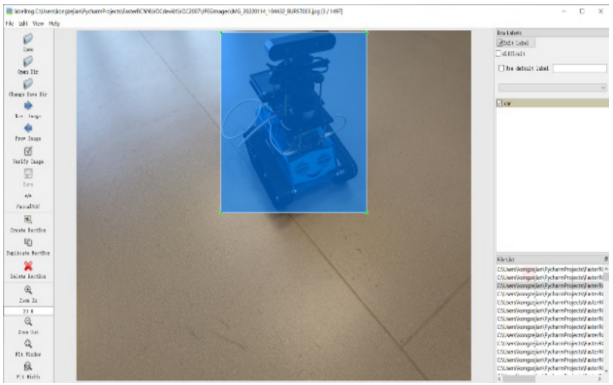

Figure 5.7: Manual annotation of the object in Labeling

The main parameters for the performance evaluation in the three deep learning methods include confidence, nms_iou, and anchors_size. Confidence is a bounding box containing the probable size of an object and the accuracy of its position. nms_iou means intersection over union which is an important computational metric measuring the accuracy of the predicted detection frame. anchors_size means the size of the anchor. The confidence is set to 50% to balance the tradeoff, and the nms_iou is set to 0.7. The setting of the anchor size is based on the traditional average value in similar cases.

Three categories of occlusion simulations are tested. One is a simulation based on data augmentation. In the second simulation, test data set 2 was created. The image in test data set 2 does not belong to the images in the target robot data set and test data set 1. The targets in these images come from different angles and distances. At the same time, some targets are out of range. The occlusion is about 10% to 30% of the target. In the third simulation, we test whether these models can adapt to the change of occlusion.

5.2.1 Experimental Setting

The target robot data set is first created. Each image has a dimension of 4160x3120 pixels. All images are collected from different distances and angles of the target robot. For example, we take photos at a certain distance from the robot, and then move around to ensure that photos from different angles of the robot are collected. The distance between the robot and the camera varies under mobility. The photos are taken accordingly. The images also contain the ones taken under bad lighting conditions or noisy conditions. When the distance or angle changes, the background of the image may change as well.

The operating workspace environment includes Windows 10, Cuda 11.4, Cudnn 7.6.5, and Visual Studio 2022. The hardware used in the experiment is NVIDIA GeForce GTX 1650 with 4096MiB. The Faster RCNN, SSD and Yolov5 models are all fast deep learning methods. To make the comparison between them as fair as possible, the settings of these three models are paid much attention.

For the Faster RCNN model, we adopt non-maximum suppression (NMS) on the proposal regions in the RPN network and set the IoU threshold for NMS as 0.7. In the Region of Interest (ROI) pooling layer, both the numbers of the positive and negative samples

are balanced to 128. Due to the different sizes of the proposed anchors, the ROI pooling obtains the output of a fixed size via the method of mapping and max pooling. For Yolov5, there are three main steps to set the model. First, the file train.py is called. Then in the voc_ball.yaml file, there is only one target in our experiment. The target name is called robot. The training data set and the verification data set are in the local location. Finally, settings in the yolov5s_ball.yaml file are responsible for most parameters. It contains a number of categories of objects in the data set, the coefficient controlling network depth, the coefficient controlling network width and batch size. All these parameters are separately well set based on our specific experiments. For the SSD model, the steps of the matching strategy are as follows: each default anchor should be sorted according to the confidence score. The confidence threshold is set to be 0.5, which means default anchors that have more than 0.5 scores will be retained. After that, one object may have several default anchors, and box position and score for non-maximal suppression are used to filter anchors. The IoU for non-maximal suppression is set to 0.7. The training is divided into freeze training and unfreeze training. In freezing training, the network backbone is frozen, and more resources are used to train the posterior network parameters. For more efficient use of the GPU memory, the batch size of freeze training is 16. In the unfreeze training, the backbone network is unfrozen, and all parameters are trained at the same time. The batch size of the part is set to 4.

5.2.2 Data sets

Several data sets are created. Images are collected by an 8-megapixel camera with F/2.0 aperture and a fixed focal length. All images in our data sets are collected in two environments, an indoor environment and an outdoor environment. We designed the following data sets:

(1)A training data set. A data set containing 2993 images is created. The images are 4160x3120 pixels. All images have been carefully annotated and used as the training set. There is a whole target in the image. The target is captured from different distances, different angles and different circumstances. (2)Testing data set 1. It contains 1718 images which are different from the training data set. All images have the same pixel as the

training data set. (3)Testing data set 2. The second testing data set with 661 images is also created. Images in testing data set 2 are chosen from the images which are not put in the target robot data set and testing data set 1. The targets in these images are under different angles and different distances. Meanwhile, part of these targets is out of range. The missing part is about 10% to 30% of the target; some examples can be seen in Figure 5.8: Examples of testing data set 2. (4)Testing data set 3. A testing data set with 599 images is created by a random erasing method.

The advantages of our designed data sets are as follows. First, the environments of the created data sets are different. To ensure the diversity of the environments, we take a series of actions which contain the exposure of the picture, use a random erasing method and so on. In addition, the target is captured from different distances, different angles Furthermore, some occlusion condition testing data sets are created. However, there exist some limitations of our designed data sets which should be improved in further study. First, the environments of the created data sets are not complicated enough. Moreover, the created data sets are relatively small. Besides, the images of the data sets keep the target from a bot close distance from the camera.

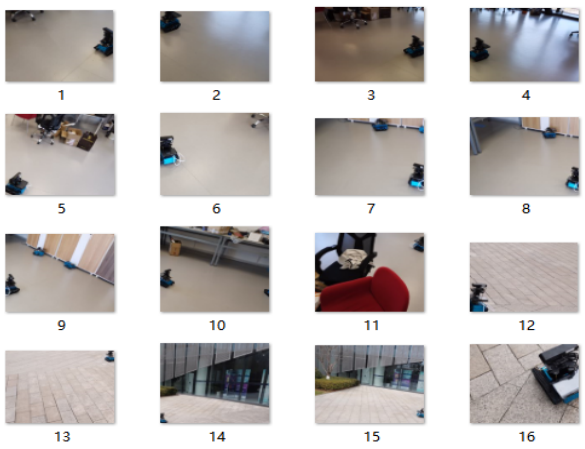

Figure 5.8: Examples of testing data set 2

5.2.3 Data augmentation

The designed program first acquires the length and width of the collected images in the data set. It then obtains the pixel points of each coordinate of the picture to facilitate the combination of pixel points. Based on the information mentioned above, the program can rearrange the pixels to create flipped images. The key point of random erasing is selecting a rectangle region in an image and erasing its pixels with random values [7]. RGB value -3947833 representing grey is used for erasing. The coverage area is set as the threshold size, which can be selected according to the specific experimental needs. The test data set simulates the occlusion effect by randomly erasing the target features. It also improves the generalization ability of the model. The random erasing enables the model to recognize the target through local features in the training process. It also enhances the model cognition of the local features of the target and weakens the model's dependence on all the features of the target. The model trained by these data is more robust to noise and occlusion. All deep learning models perform well, the random erasure method is used to create test data sets to judge the ability of these models.

A testing data set with 599 images are constructed by a random erasing method which is called testing data set 1 as shown seen in Figure 5.9: Screenshot of the erased data set. These images are selected from images that have never been placed in the target robot data set. This data set depicts targets in the test set undergoing different degrees of random erasure in different environments. The data set is composed of car images in indoor and outdoor environments. These images are covered so that the light and shade change to a certain extent. Random numbers are added to enrich the data set based on the coordinate parameters selected by the label box and data enhancement operations such as erasing. We design specifically a program to generate the test data set using this random erasure method. The area erased by the program is in accordance with the size of the target in the image. According to the label marked in the image, the program can ensure that an area of the target is identified. A starting point is then randomly created in this area. A random erase area is generated, which is smaller than the target area of a random value. Therefore, all erasure areas can erase some target areas with different sizes, so as to ensure that the test data set has sufficient generalization ability.

Figure 5.9: Screenshot of erased data set

5.2.4 Different Environments

The second test data set containing 661 images is created and defined as test dataset 2. The image in test data set 2 has never been selected from the images in the target robot data set and test data set 1. The targets in these images come from different angles and distances to simulate different mobilities. Practical occlusion is simulated for out-of-range scenarios. The images of the data set also contain images taken under harsh lighting or noisy conditions. The practical occlusion is about 10% to 30% of the target.

Furthermore, the deep learning models have different performance in detecting the object at different light intensities which can be seen in Table 5.4: Testing results for deep learning models brightness. One image was selected from our testing data set. Then we change the exposure of the image from -5 to 5. The Faster RCNN model and the SSD model can detect the object in the darkest and lightest environment. However, the Faster RCNN model produces multiple detection results for the same target. The YOLOv5 model can detect the object in the lightest environment, but it can not detect the object in the darkest environment.

Table 5.4: Testing results for deep learning models brightness

Name	**Brightnrdd**	**Multiple detection error**
Faster RCNN	-5-5	Error
SSD	-5-5	No error
YOLOV5	-4-5	No error

5.2.5 Occlusions Variation

To make a judgement on whether the SSD model, the Faster RCNN model and the YOLOv5 model are adaptive to different degrees of occlusion. We pick up an image as the testing sample. The size of the image is 4160*3120. The upper-left and lower-left corners of the target are (1,991,811) and (2,752*1,730) respectively. We then cut the image step by step to ensure that varying occlusion of the target can be measured. The precision is controlled to the range of 1%. The experiments demonstrate that the Faster RCNN and the SSD models can handle occlusion of up to 95% under a relatively simple background. The YOLOv5 model works under up to 96% occlusion. The experiments demonstrate that the Faster RCNN can handle occlusion of 91%. The SSD models can handle occlusion of about 80%. The YOLOv5 model works under 82.5% occlusion.

It is worth noting that the Faster RCNN model has a problem in the experiments. It usually marks the local features with multiple anchors on the same object during our experiments because the model is based on the candidate box extracting which can be seen in Figure 5.10: Duplicate detection in Faster R-CCN. Local features make target detection a problem for multi-scale and small targets. Multi-scale means that the problem studied involves multiple orders of magnitude. The feature maps extracted by Faster RCNN are all monolayer, so it is not very good to deal with this problem. In addition, in order to avoid re-detection, the method used by the Faster RCNN model is unfriendly to the occluded target. If the threshold is set too large, the model will cause missed detection. If the threshold is set too small, the model will re-detect. When the target only shows local

features, the model will not miss detection because only a small part of the field of vision is the target. However, within this small local feature, the model will continue to subdivide, causing re-detection. For example, our data set detects one target, and the color feature is relatively simple. After the lower left corner is identified as a target, similar features inside the target will also be detected to cause re-detection for the target as Figure 5.10 shows.

Figure 5.10: Duplicate detection in Faster RCNN

All three models can generally adapt well to the practical occlusion. The detection methods of these deep learning models are based on the local features of the target. Therefore, it also leads to limitations. For example, if the training data set is not large enough, the model can not distinguish objects of similar features from the target. However, if the training data set can provide enough possible interference and possible similar objects, the deep learning model can be more adaptive to occlusion.

5.2.6 Results and discussions

In this section, more numerical experimental results are presented to demonstrate the anti-occlusion performance of the deep learning models with the generated data set. YOLOv5, Faster RCNN, and SSD are chosen as the benchmarks. Labeling was used to label images in the data set. The target object is the robot vehicle. After labeling the images, the model is fine-turned according to the training data set. Training time, F1 Score, Precision and

Recall are adopted as evaluation metrics. Score_threshold is the threshold of confidence. Prediction results greater than this value are retained.

The training time of the three models is also tested. The test data set is created using the images and videos of the target robot at different angles indoors and outdoors. Initially, the random erasure method is not applied. The three models all provide perfect results with 100% accuracy and recall. When more challenging situations are simulated, i.e., when the random erasure method is adopted, the PR curves of the three models for testing data set 1 and testing data set 2 are shown in Figure 5.11 and Figure 5.12 respectively. As shown in Figure 5.11, the Faster RCNN model has a PR curve most close to the upper right corner. The YOLOv5 has the best performance for test data 2 as shown in Figure 5.12. The values of precision and recall are also different under different confidence levels. The data in this project uses the specified value when the confidence is set to 0.5. We need to use other testing parameters to further judge which model has the best performance.

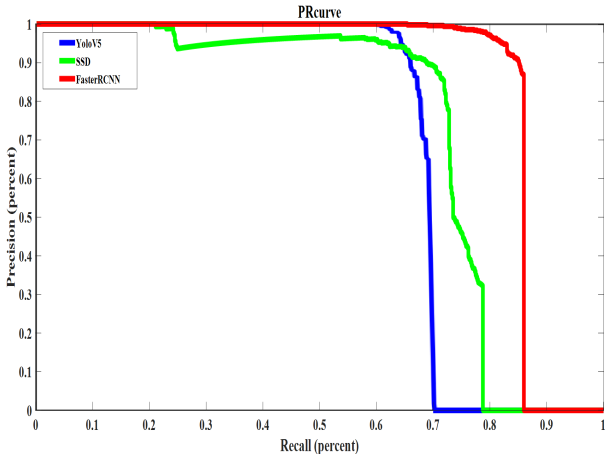

Figure 5.11: Precision and Recall for testing 1

According to Figure 5.11 and Figure 5.12, it is observed that the SSD model is the most stable one. For example, the Faster RCNN model reaches 0.85 Recall in Figure 5.11 and 0.92 Recall in Figure 5.12 when the Precision is 0. The YOLOv5 model reaches 0.7 Recall in Figure 5.11 and 1 Recall in Figure 5.12 when the Precision is 0. The SSD model keeps about 0.8 Recall in Figure 5.11 and Figure 5.12 when the Precision is 0. Therefore,

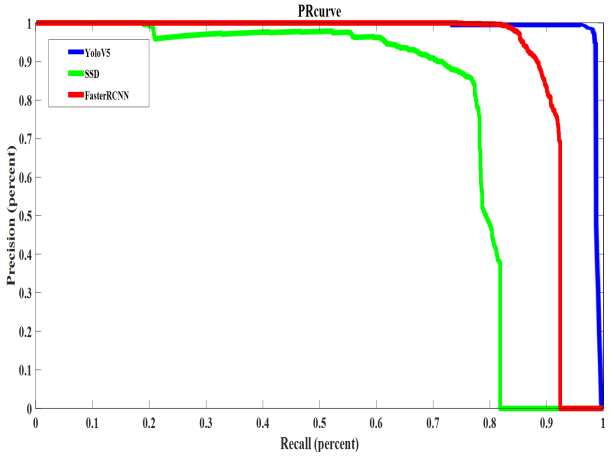

Figure 5.12: Precision and Recall for testing 2

the SSD model is the most stable model when dealing with different testing data sets. The reason is that the size and position of the prior frame of SSD are set in advance, which makes it simple and less affected by the environment. The configuration files of the models are also modified to train the data set. Accuracy quantifies the correlation of detection targets. Recall refers to the ratio of detected targets to all existing targets. F1 score is an index used to measure the accuracy of the binary classification model in statistics. It takes into account the accuracy and recall of the classification model. The training time of these models is also tested.

As shown in Table 5.5: Testing Results for testing data set 1, Faster RCNN and SSD have a comparable performance of 0.87 and 0.89 respectively in the F1 score, which is much better than 0.54 with the YOLOv5 model. The SSD model achieves the highest F1 score. The precision of Faster RCNN, SSD and YOLOv5 is 93.38%, 97.72% and 97.69%. The SSD model achieves the highest precision value. The Recall value of Faster RCNN, SSD and YOLOv5 is 77.62%, 81.16% and 63.66% respectively. Again, the SSD model achieves the highest value. As shown in Table 5.6: Testing Results for testing data set 2, the F1 score for three models are 0.89, 0.81 and 0.98. The Faster RCNN model achieves the highest F1 score. The precision of Faster RCNN, SSD and YOLOv5 is 91.55%, 76.96% and 98.18%. The YOLOv5 model achieves the highest precision value. YOLOv5 also

achieves a high score for the Recall which is 98.18% and better than 87.54% and 81.99% for Faster RCNN and SSD respectively. Meanwhile, the Faster RCNN model spends about 48.25 hours on training. The SSD model spends the least training time which is about 15.16 hours for training. The YOLOv5 model spends about 20.25 hours on training. The detection speed of Faster RCNN is about 1.6 frames per second. For SSD and YOLOv5 models, the detection speeds are about 6.6 and 3 frames per second respectively.

Table 5.5: Testing Results for testing data set 1.

Name	**Training time**	**F1 score**	**Precision**	**Recall**
Faster RCNN	48.25h	0.87	93.38%	77.62%
SSD	15.16h	0.89	97.72%	81.16%
YOLOV5	20.25h	0.77	97.69%	63.66%

Table 5.6: Testing Results for testing data set 2

Name	**Training time**	**F1 score**	**Precision**	**Recall**
Faster RCNN	48.25h	0.89	91.55%	87.54%
SSD	15.16h	0.81	76.96%	81.99%
YOLOV5	20.25h	0.98	98.18%	98.18%

According to the values in Table 5.5: Testing Results for testing data set 1 and Table 5.6: Testing Results for testing data set 2, it is difficult to decide which model has the best overall performance. In order to further judge the performance of these models, we defined a variable called P which means the performance of the model in the designed experiments. The F1 score is regarded as an index to measure the accuracy of the binary classification model in statistics. It takes into account both the accuracy and recall of the classification model. F1 score can be regarded as the weighted average of model accuracy and recall rate. Its maximum value is 1 and its minimum value is 0. In general, the larger the F1 score is, the better the model's performance is. The training time is the time spent by the

model to train the data set. The detection speed is how many frames the model can detect per second. As the values of the F1 score and the detection speed should be as high as possible and the value of the training time should be as low as possible, the performance of the model can be determined accordingly. The function of P can be formulated as follows:

$$P = \frac{F1score \times DetectSpeed}{TrainingTime} \tag{5.3}$$

The higher the value of P is, the better the model performance will be. Based on Table 5.5, it can be calculated that $P_{ssd} = 0.387$, $P_{faster-rcnn} = 0.029$ and $P_{Yolo} = 0.114$. Based on Table 5.6, the three values can be derived as $P_{ssd} = 0.353$, $P_{faster-rcnn} = 0.030$ and $P_{Yolo} = 0.145$. According to the results, it can be concluded that the SSD model has the best performance.

For further analysis of whether classic deep learning model tracking methods can adapt to various occlusion conditions and to what extent occlusion could be handled in a specific environment. A video containing our target and two similar objects was created to test the performance of the Faster RCNN model, SDD model and YOLOv5 model. This is a thirty-nine seconds video with thirty frames per second. An example of the video can be seen in Figure 5.13: Screenshot of the testing video. At first, the target is located on the screen as Figure 5.13(a) and Figure 5.13(b) show. Then a similar object will be put in the screen as Figure 5.13(c) and Figure 5.13(d) show. After that, a third similar object will be put on the screen as Figure 5.13(e) shows. While filming the target, the photographer constantly changes the angles and keeps a relatively close distance with the target.

The testing results can be seen in Figure 5.14: Three models' testing results at the same frame.

In Figure 5.14, the object in the lower right corner is the target. Other two objects are interferents. Figure 5.14 shows that in the same frame, the Faster RCNN model detected the target, but it also incorrectly identified the similar objects. The SSD model successfully detected the target and ignored the similar objects. The YOLOv5 model detected the target, but it incorrectly identified the similar objects and gave one of the similar objects a higher possibility than the true target.

Based on the results, it can be concluded that the SSD model has the best performance

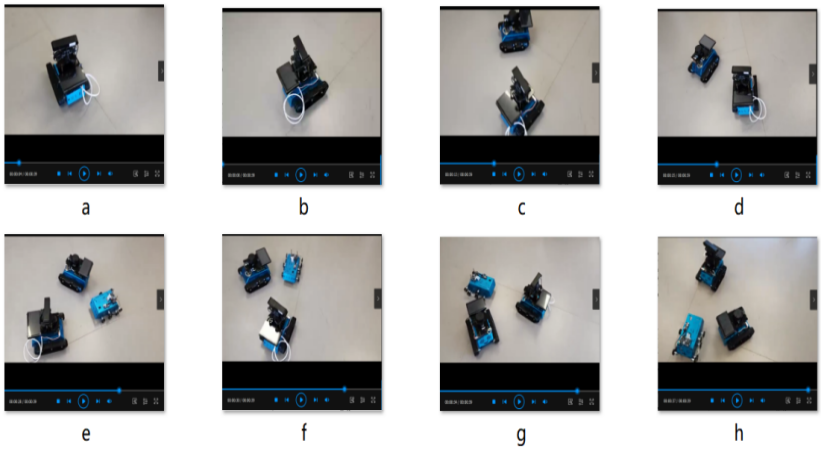

Figure 5.13: Screenshot of the testing video

in the designed experiments. The SSD model has the least training time, the highest F1 score, the highest precision, the highest recall and the fastest detection speed. The SSD model achieves the highest performance score in two testing data sets. Such experiments and data set design methods can be applied to other specific circumstances. However, the experiments exist limitations. First, the environments of the created data sets are not complicated enough. Second, the created data sets are relatively small. Third, the images of the data sets keep the target from a bot close distance from the camera. Those limitations should be improved in further research. In summary, a comparative study of Faster RCNN, SSD, and YOLOv5 for mobile robot tracking is provided. All three deep learning models are adaptive to partial occlusion of different degrees as analyzed. We apply these models to specific experiments including erased situations, darkest environment, brightest environment, clear environment, complex environment and occlusions variation. The performance of the models was demonstrated in terms of F1 score, precision, recall, floating points of operations and training time. A variable P is created to comparethe performance of the models. Four high-quality data sets were generated to facilitate the performance evaluation. The random erasing method and occlusion from different angles were applied to augment the data sets covering various scenarios of partial occlusion.

Experimental results showed that the SSD model has the best performance and is most

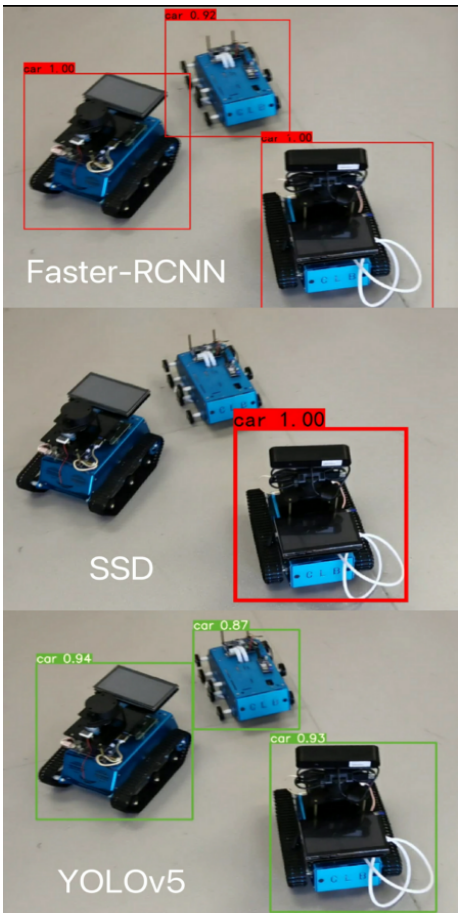

Figure 5.14: Three models' testing results

promising for the addressed application. The P score of the SSD model achieves 0.387 in the first testing experiment which is higher than the Faster RCNN by about 14 times. The P score of SSD is higher than the YOLO model by about 4 times in the first testing experiment. In the second testing experiment, the P score of SSD also has a consistent performance. It achieves 0.353 which is higher than the Faster RCNN by about 12 times. It is also higher than the YOLOv5 model by about 3 times.

The analysis revealed different features of the three models in implementation. The performance of the models differs in terms of the anti-interference ability, detection speed, and the ability to determine the object according to the detection of local features. The

analysis results may serve as a reference for the model selection to meet specific design metrics. There is also much room for improvement with the current algorithm design and performance evaluation. For example, the images of the current robot data set contain a single robot target. Further tests can be carried out to include more robots in the scene.

Chapter 6

A multi-AGV prototype system

In this section an AGV prototype system is created. A real-time visual tracking and distance measuring algorithm is created. In addition, some tracking and collision avoidance functions are tested in the created AGV prototype system.

6.1 Real-time visual tracking and distance measuring algorithms based on SSD

Our approach is based on the combination of the SSD model and Data regression model. Figure 6.1: The flowchart of the proposed method shows the progress of the designed method. First, the model of SSD is loaded. The program will then read the frame to detect the target. In addition, a default box will lock the target and SSD outputs the coordinate of the center position of the lower border of the default box. After that, the distance between the camera and the camera will be calculated based on the output coordinate of SSD. The result will be shown on the screen. Subsequently, the next frame will be read until all frames have been read. If there is no target in the frame, the algorithm will continue to detect but will neither lock anything nor output any distance information. For real-time detection, the algorithm reads the real-time images from the camera. The program will not stop until receives a stopping command.

The first distance measuring method is based on imaging geometry which can be seen

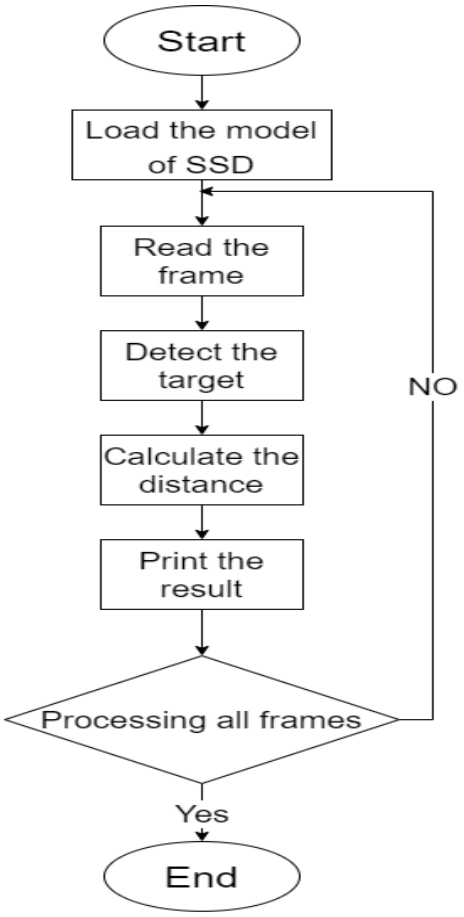

Figure 6.1: The flowchart of the proposed method

in equations 6.1 to 6.5.

$$Z_c \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{pmatrix} (RT) \begin{pmatrix} X_2 \\ Y_w \\ Z_w \\ 1 \end{pmatrix} = M_1 M_2 \begin{pmatrix} X_2 \\ Y_w \\ Z_w \\ 1 \end{pmatrix} \tag{6.1}$$

(u,v,1) is a vector that presents the position of the target in the pixel coordinate system. fX, fY are the Focal lengths of the camera in x-direction and y-direction respectively. R is the Rotation matrix and T is the Translation matrix. (XW, YW, ZW,1) represents the

coordinate of the target in the world coordinate system. M1 is the intrinsic matrix and M2 is the extrinsic matrix. The equation 6.1 describes how the imaging system converts the position of the target in the real world to the imaging plane. By measuring the position of the target in the real world and outputting the coordinate of the target from the software, the world coordinates and pixel coordinates are easy to acquire. But other parameters cannot be acquired directly.

Because different cameras have different converting parameters while users do not know these parameters in advance, Zhang Zhenyou method can be used to calibrate the intrinsic matrix of the camera. It is a flexible technique to easily calibrate a camera [12]. By taking photos of a standard checkerboard of black and white rectangles from dozens of different angles, the method can resolve the equations about the intrinsic matrix of the camera. Considering the fact that the target moves relative to the camera, the extrinsic matrix always changes. Therefore, though ZhangZhenyou method can provide the extrinsic matrix of each calibrating image, these matrixes are useless for other situations. Therefore, the geometry model is created to compensate for the unknown extrinsic matrix with the height of the camera relative to the ground, the angle between the optical axis of the camera and the horizontal line, the camera and the target moving in the flat ground as prior conditions. It is based on a Vision-based Adaptive Cruise Control (ACC) system [13].

$$Z_c = cos\left(arctan\left(\frac{v - v_0}{f_y}\right)\right) \times \frac{H}{sin\left(a + arctan\left(fracv - v_0 f_y\right)\right)} \tag{6.2}$$

$$X_c = Z_c \frac{u - u_0}{f_x} \tag{6.3}$$

$$Y_c = Z_c \frac{v - v_0}{f_y} \tag{6.4}$$

$$D = \sqrt{X_c^2 + Y_c^2 + Z_c^2} \tag{6.5}$$

X_c, Y_c, Z_c: Three coordinates in Camera coordinate system, H: The height of the

camera relative to the ground, α:The angle between the optical axis of the camera and the horizontal line, D: the distance between the target and the camera. The tracking algorithm outputs the coordinates of the target in the Pixel coordinate system: u and v. The Pixel coordinates should be converted into Image coordinates and then to Camera coordinates. Formula 6.2 uses prior conditions to calculate the depth value based on geometry relations. Formula 6.3 and Formula 6.4 use the parameters of intrinsic matrix and depth value to calculate the other two coordinates in the camera coordinate system. Finally, Formula 6.5 calculates the distance between the target and the camera.

The second distance measurement algorithm is based on Data regression modeling. To simplify the model, the method only focuses on the distance when the target locates right ahead of the camera. There is a mapping relation between the distance and the Pixel y-coordinate: D=f(y). By analyzing, the fitting of the mapping relation can be reduced to a multi-variable non-linear fitting problem:

$$D = c_0 + c_1 y + c_2 y^2 + ... + c_m y^m \tag{6.6}$$

Firstly, the data pairs of real distances and values of pixel coordinates are collected. Then the least square method is used and functions of different degrees are selected to fit these data points. Finally, the curves are evaluated and the function which has the best fitting curve is determined.

The hardware used contains the Dell personal computer with NVIDIA Quadro P600 GPU. The camera is the front-facing camera of the computer which is an HP Wide Vision HD Camera made by Microsoft. Operating workspace environment contains windows 10, cuda 11, cudnn7.6.5, pycharm and pytorch. There is a small tracking target and the experiments are conducted indoors. To train the models of deep learning visual tracking algorithms, a data set consisting of 300 images is created. Firstly, we take 300 photos of the target from different angles and distances indoor environment with multiple objects as backgrounds. Then the photos are used to create a data set with VOC format. The SSD model uses the data set to train its model and load the model to predict. The target is laid on the ground which is in the front-facing of the camera and then we run the program. To test the performance of the algorithm, the target moves from near to far. Meanwhile, we

record the output distance value and use a tapeline to measure the real distances. Then we compare the measuring distance with the real distance which can be seen in Figure 6.2.

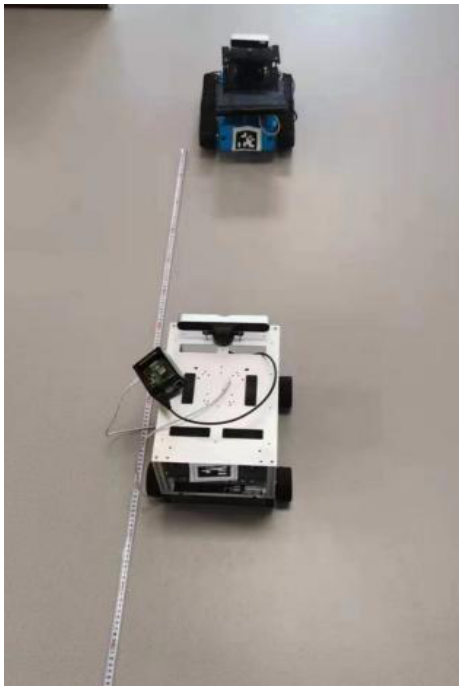

Figure 6.2: Distance Measurement Experiment

Figure 6.3 shows the fitted curve for the processed pixel coordinates and real distance of the experiment. X-axis represents the processed pixel value of the center position of the lower border of the default box in the pixel coordinates while the Y-axis represents the real distance between the camera and the target. The blue point is the collected data for the modeling. The red curve is the fitted curve for the function, which is the result of data regression modeling.

The image has 480 pixels in y-coordinate in total. The original point locates in the top left corner of the image in the pixel coordinate system. That means the value of the y-coordinate increases from the top to the bottom of the image. However, when the target appears at the bottom of the image, it means a shorter distance. To create the model which is in line with the intuitive feeling, the mapping function becomes D=f(480-y)=f(h), h:pixel height. 14 pairs of data are collected. The result shows that the quadratic function can

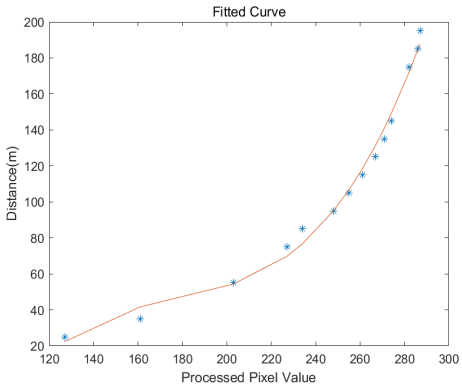

Figure 6.3: The fitted curve for the processed pixel coordinates and real distance

not meet the requirement of monotone increasing in the interval and the quartic function are over-fitting. So the cubic function has the best effect. The fitting result can be seen in Table 6.1: The fitting results. Algorithm 1 is a distance measuring method based on imaging geometry. Algorithm 2 is a distance measuring method based on Data regression.

Based on table 6.1, algorithm 1 has around 10% percent error when the real distance ranges from 1m to 1.6m. When the distance is smaller than 0.6m and larger than 2.15m, the error is larger than 20%. Because as the distance increases, one pixel corresponds to a larger distance in the real world, the error is large when the distance is far. When the distance is small, the error resulting from camera calibration is enlarged. As a result, the distance measurement algorithm can only have good performance in a particular distance. Algorithm 2 has much smaller errors than algorithm 1. Its error rate is smaller than 12%. However, if the real distance is lower than 0.24m, the algorithm will output negative values, which is impractical. As a result, though the algorithm has a higher degree of accuracy, it also has limitations.

In summary, a novel method for real-time visual tracking and distance measurement based on SSD is proposed. This algorithm can measure the distance of the tracking object with high accuracy in real-time. The error can be controlled at a centimeter-level. It can meet relevant industrial needs to a certain extent. There still exists potential to improve the designed algorithm's performance such as better hardware. In the future, SSD algorithms can be improved by adding anti-convolution layers to optimize the performance

Table 6.1: The Fitting results

	Algorithm 1		Algorithm 2	
Real distances(m)	Measuring distance(m)	Error	Measuring distance(m)	Error
0.25	0.11	56%	0.21	16%
0.3	0.22	26.7%	0.32	6.7%
0.4	0.3	25%	0.45	12.5%
0.5	0.39	22%	0.49	2%
0.6	0.47	21.7%	0.59	1.7%
0.7	0.57	18.5%	0.63	10%
0.8	0.66	17.5%	0.73	8.8%
0.9	0.78	13%	0.88	2.2%
1.0	0.92	8%	0.98	2%
1.1	1.07	2.7%	1.08	1.8%
1.2	1.18	1.6%	1.17	2.5%
1.3	1.26	3%	1.32	1.5%
1.4	1.37	2%	1.39	0.71%
1.5	1.59	6%	1.49	0.67%
1.6	1.79	11%	1.602	0.13%
1.7	1.88	10%	1.69	0.59%
1.8	2.15	19%	1.78	1.1%
1.9	2.76	45%	1.82	4.2%
2.0	3.34	67%	1.85	7.5%

to detect small objects and increase its semantic comprehension ability. The distance measurement algorithms can be replaced by the monocular deep estimation algorithms and the experimental environment will not be limited to flat ground.

6.2 Tracking and collision avoidanc in a multi-AGV system

6.2.1 System overview

This system contains four hardware components, namely a laptop as the central control platform, an AGV as the master, and two AGVs as the slave. Figure 6.4 provides information about the node structure in this system. This system is established in Wireless Local Area Network (WLAN). The node's structure can be divided into nodes in PC, Master and Slave1 Slave2. There is one personal computer which is regarded as PC in Figure 6.4. Node 1 is data collector which is registered in node manager. It collects data from master. Then, it will publish /map and /LaserScan to the Node manager. There is a registered Node 2: GUI which is responsible for subscribing /map and /LaserScan to the node manager exists in PC's node manager. The PC part plays the role of sending instructions to

Master. Master will then send feedback and data to PC part.

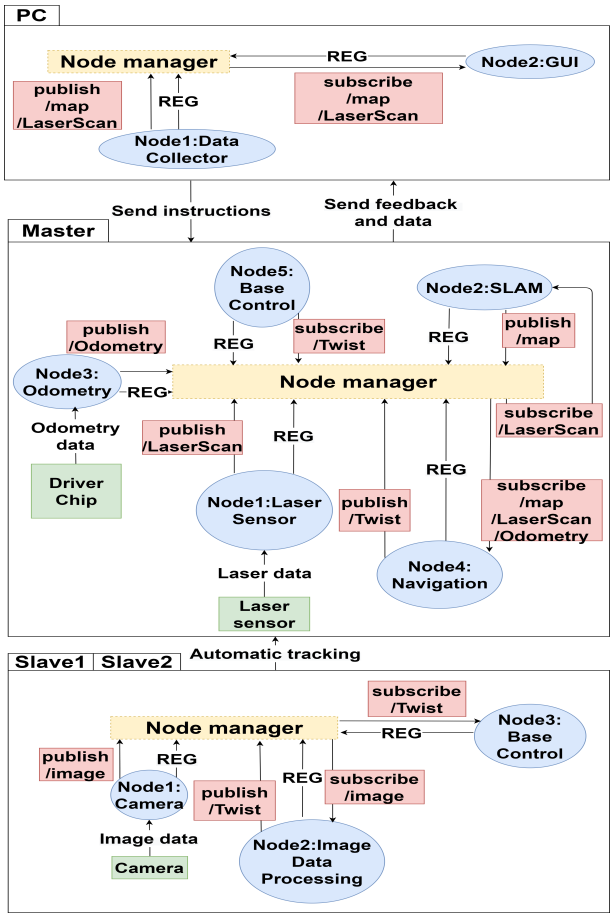

Figure 6.4: The node structure in this system

In Master part, there are five registered nodes in node manager which are called Node1: Laser sensor, Node2: SLAM, Node3: Odometry, Node4: Navigation and Node 5: Base Control. First, the laser sensor can collect the environment's information and transmit them to laser data. Node1: laser sensor will then publish /LaserScan to node manager which will be subscribed by Node2: SLAM. Node2: SLAM will publish /map after /LaserScan had been processed. While the master is moving, its driver chip will collect odometry data to Node3: Odometry for publishing /Odometry. When /map, /LaserScan and /Odometry have been collected, these messages will be subscribed by Node4: Navigation and publish

/Twist. The /Twist message will be subscribed by Node5: Base control to realize the navigation.

Slave1 and Slave2 are two independent devices in the system. Both of them are only automatically tracking AprilTag installed in the master. Therefore, their structure is the same as each other. There are three registered nodes, Node 1: Camera, Node 2: Image Data Processing and Node 3: Base Control. After the camera collects images data, Node 1: Camera will publish /image to the node manager. Node 2: data processing is responsible for subscribing /image and publishing /Twist. The /Twist message containing the velocity information will finally be subscribed by Node 3: Base control to driver the vehicle.

In this project, there are totally three AGV prototypes used in this multi-AGV system, one is the master vehicle and the other two are the slave vehicles. Regarding the master AGV, it has two chips, two sensors, one camera, one battery and two motors. In terms of the salve AGV, it has two chips, one camera, one battery and two motors.

The components used in both the master and the slave are two chips, one battery and two motors. Firstly, the chip called Raspberry Pi 4 Model B is the most significant component for all vehicles, providing critical functionalities such as the installation of the software platform, the connection of sensors, and the collection of data received from sensors. This project selects Raspberry Pi 4B with 4GB Random Access Memory (RAM) as the main chip for its extraordinary performance. The Raspberry Pi 4B has a Broadcom BCM2711 System on Chip (SOC), quad-core Cortex-A72 (ARM v8) 64-bit Central Processing Unit (CPU), 500MHz Video Core VI Graphics Processing Unit (GPU), along with Gigabit Ethernet, 2.4 GHz and 5.0 GHz IEEE 802.11ac onboard wireless networking, and Bluetooth 5.0 [170]. Moreover, there are totally 9 ports on this chip, including two USB 3.0 ports, two USB 2.0 ports, two micro-HDMI ports, one 2-lane MIPI DSI display port, one 2-lane MIPI CSI camera port, and one 4-pole stereo audio and composite video port [170]. In addition, a minimum 3A and 5V DC power supply must be provided to drive the Raspberry Pi 4B. Secondly, the additional driven chip acts as a protection and a chassis driver for AGV. It has the STM32 Microprogrammed Control Unit (MCU) based on the Arm Cortex-M3 processor, a power processing unit for voltage stabilization, and a gyroscope as the Inertial Measurement Unit (IMU) sensor. This chip, as a protection unit,

connects between batteries and Raspberry Pi 4B, with a switch to control this connection being on or off. Since two motors need high voltage to drive the wheels, the 12V 8400mAh Li battery used in this project can provide 12.6V DC voltage at maximum. However, the rated supply voltage of Raspberry Pi 4B is only 5V. If Raspberry Pi 4B is directly connected to batteries, there exist security problems that the high voltage provided by batteries may harm or even ruin Raspberry Pi 4B. Therefore, the additional driven chip is used as a buffer to transfer the high voltage from the battery into the lower one, and then provides it to Raspberry Pi 4B. Furthermore, the additional driven chip is also for driving motors, and obtaining the data by the gyroscope in IMU to calculate the posture of AGV. However, the sensors used in the master and slave AGVs are different. The master vehicle has two sensors, namely a IMU sensor and a laser radar, which can perform functions of navigation and collision avoidance. Whereas, the slave vehicle only has a camera to detect AprilTag and track the master AGV.

As mentioned above, the master AGV has two chips (Raspberry Pi 4B and the additional driven chip), three sensors (a IMU sensor, a laser radar, and a camera), one battery and two motors in total. Figure 6.5 shows the picture of the master AGV.

Figure 6.5: Master AGV

The slave AGV has two chips (Raspberry Pi 4B and the additional driven chip), one camera, one battery and two motors in total. Figure 6.6 shows the picture of the slave AGV.

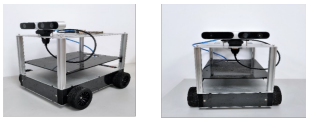

Figure 6.6: Slave AGV

There are two communication mechanisms in the ROS system, one is asynchronous and another is synchronous. For the asynchronous communication called the publish/subscribe

model, there are four essential elements, namely the publisher, the subscriber, the topic, and the message. In this model, nodes communicate with each other by publishing and subscribing messages to topics, where the topic is the bus over which nodes exchange messages and the message is a simple data structure [171, 172]. The publisher is the node that generates data and publishes it to the relevant topic, and the subscriber is the node that subscribes the data from the relevant topic. Moreover, there can be multiple publishers and subscribers to a single topic [171]. For the synchronous communication called the request/reply model, there are three essential elements, namely the server, the client, and the service. In this model, one-to-one communication is done via the service, which is defined by a pair of messages: one for the request and one for the reply [173]. The server is a providing ROS node that offers a service under a string name, and the client is a receiving ROS node that calls the service by sending the request message and awaiting the reply [173].

This project selects the publish/subscribe model as the communication mode. According to the ROS documentation, topics should be used for continuous data streams, while services should be used for remote procedure calls that terminate quickly and should never for longer running processes [174]. Since all functionalities in this project are executed continuously, the publish/subscribe model is more suitable than the later. A more detailed comparison between these two models is illustrated in Table 6.2.

Table 6.2: The comparison between two communication models

	publish/subscribe	request/reply
synchronicity	Asynchronous	Synchronous
protocol	TCP / UDP	
feedback	no	yes
real-time	weak	strong
node relationship	many publishers many subscriber	one server many clients

In 2019, AprilTag 3 was proposed by M. Krogius's team [175]. In comparison with traditional square tags, this system allows flexible tag layouts, in order to increase the

Table 6.3: Detected data of AprilTag

Sample	Pose_X	Pose_Y	Pose_Z	distance	Angel_yaw	Angel_pitch	Angel_roll	ProcessingTime
1	0.0460288	-0.000812352	0.596836	0.598608	4.40465	-6.88372	-8.20532	1.026467638
2	0.0458277	-0.000841411	0.594822	0.596585	4.22069	-9.16448	-9.6069	0.299247974
3	0.0458426	-0.000838805	0.594611	0.596376	4.27719	-8.22805	-10.1105	0.284649161
4	0.0459801	-0.000806649	0.596003	0.597775	4.41576	-5.71747	-9.29448	0.284084104
5	0.0458639	-0.000828704	0.594941	0.596707	4.3211	-7.73174	-9.97052	0.292468245
6	0.0460182	-0.000793029	0.597126	0.598897	4.3303	-7.25399	-7.37816	0.296832262
7	0.0459545	-0.000829471	0.595924	0.597694	4.38787	-6.28899	-9.5437	0.291803743
8	0.0459251	-0.000811722	0.595752	0.59752	4.29081	-7.99599	-9.05479	0.289348919
9	0.046014	-0.000783591	0.597028	0.598799	4.37338	-6.56247	-7.51251	0.290211552
10	0.0460224	-0.000807865	0.596584	0.598357	4.37359	-6.14989	-8.78231	0.281133592
11	0.0460661	-0.000798347	0.597053	0.598828	4.339904	-5.61404	-8.11978	1.075676208
12	0.0460459	-0.000781463	0.597205	0.598978	4.38537	-5.54577	-7.46402	0.287729709
13	0.0459971	-0.00078924	0.597106	0.598875	4.28272	-8.14498	-7.01015	0.296391389
14	0.0459351	-0.000806709	0.595997	0.597765	4.35444	-6.52675	-8.96954	0.295068514
15	0.0460364	-0.000790448	0.596841	0.598615	4.46154	-4.67199	-8.12324	0.290447035
16	0.0459261	-0.000804632	0.595762	0.59753	4.34102	-7.64638	-9.14672	0.287047762
17	0.0459153	-0.000819007	0.595558	0.597326	4.35839	-7.21496	-9.5576	0.291283260
18	0.0460632	-0.000787145	0.597242	0.599016	4.45682	-5.00217	-7.73684	0.292008632
19	0.0460629	-0.000786151	0.597227	0.599001	4.49231	-4.32794	-7.57731	0.297153096
20	0.0459239	-0.000832701	0.595953	0.597721	4.34747	-7.34497	-8.78239	0.347739264

data density of standard square shaped tags [175]. Furthermore, the detector in this system is faster and has higher recall than the AprilTag 2 detector while maintaining precision. Therefore, this project chooses it as the tracking system used in slave vehicles. In 2020, J. Kallwies's team presented the results of an extensive comparison of four freely available libraries for detecting AprilTags, namely AprilTag 3, AprilTags C++, ArUco, and OpenCV [176]. Not only the localization accuracy but also the processing time was taken into consideration for analyzing the algorithm efficiency. Overall, the recommendation is to use ArUco and OpenCV for a required short computing time, to use AprilTags C++ when considering the high requirement of accuracy, and to use AprilTag 3 when a very high detection rate even for small tags is important [176]. Generally, the testing results demonstrate that AprilTag 3 provides the best overall tradeoff between computation time and localization accuracy, which is the reason for choosing AprilTag 3 in this project.

6.2.2 Experiments and analysis

AprilTag 3 is chosen as the detector to identify the Apriltag family Tag36h11. A quadrilateral boundary and a circle represent the border and the center of the tag respectively.

Figure 6.7: The experimental environment

In order to analysis the detection program in details, there is an experiment conducted to record not only the distance information, but also the processing time for one-time detection. Figure 6.7 shows the environment for this experiment, where an AGV faces towards a tag belonging to Tag36h11 family to perform the detection. The measured distance between the AGV and the tag is displayed in Figure 6.8.

Table 6.3 records 20 samples of a static tag detection with the 6DOF information, distance, and processing time. The unit of Pose_X, Pose_Y, Pose_Z, and distance is meter, the unit of Angle_yaw, Angle_pitch, Angle_roll is degree, and the unit of processing time is second. It should be noted that the camera coordination is different from the robot coordination.

Figure 6.8: Distance between an AGV and Apriltag

The measured distance can be obtained as 60 cm. From Table in appendices, the detected distance varies from [0.596376, 0.599016]. Therefore, it can be concluded that the detection has high accuracy since the difference between maximum and minimum values is only 0.00264 m. Although the detection has high accuracy, the detection efficiency is not satisfactory. The recorded processing time varies from [0.281133592, 0.347739264],

except 1.026467638 s in the first detection and 1.075676208 s in Sample 11. For the first detection, there is a longer time for initializing the detector by setting options. However, the exception in Sample 11 may be caused by the sudden network delay, the blocking in CPU of Raspberry Pi, or even the poor connection between the camera and Raspberry Pi. Eliminating these accidental data, the average processing time is 0.289976 s, which is a bit high. Thus, this system may be not able to achieve real-time tracking function due to the high detection delay.

On account of that AprilTag detection is not real-time and there also exist time delays in the communication not only between nodes but also between the chassis and the base control node, this project tests four groups of speeds (0.07 m/s, 0.075 m/s, 0.08 m/s, 0.085 m/s), so as to find the most suitable velocity for tracking by considering both the safe distance to avoid collision and the processing time of running the program.

Table 6.4: Testing results for 0.07 m/s

Processing Time	distance	z	x	velocity
0.002306999	0.582765	0.582753	0.00282	0.07
/	0.584066	0.584055	0.00283	0.07
/	0.582602	0.58259	0.00286	0.07
/	0.582388	0.582376	0.00307	0.07
/	0.580470	0.580459	0.00296	0.07
/	0.583423	0.583411	0.00311	0.07
/	0.580613	0.580601	0.00306	0.07
/	0.583408	0.583397	0.00271	0.07
/	0.584174	0.584162	0.00325	0.07
/	0.566681	0.566601	0.00644	0.07
/	0.533076	0.532785	-0.00731	0
5.993638121	0.498778	0.498015	-0.000859	0
6.593781839	0.467892	0.466519	-0.001652	0
6.993517009	0.435848	0.433447	-0.002286	0

Table 6.5: Testing results for 0.075 m/s

Processing Time	distance	z	x	velocity
0.002121451	0.597263	0.597144	0.011911	0.075
/	0.594705	0.594585	0.011895	0.075
/	0.597187	0.597068	0.011899	0.075
/	0.598384	0.598265	0.011907	0.075
/	0.597892	0.597773	0.011889	0.075
/	0.598012	0.597893	0.011907	0.075
/	0.597555	0.597436	0.011905	0.075
/	0.598811	0.598692	0.011915	0.075
/	0.594431	0.594312	0.11907	0.75
/	0.572032	0.571879	0.01144	0.075
/	0.541822	0.541463	0.011192	0.075
/	0.503291	0.502481	0.010785	0
6.490954316	0.469614	0.468110	0.010346	0
6.990804035	0.432967	0.430352	0.009109	0
7.491044129	0.398950	0.394902	0.007925	0

From Table 6.4-6.7, it can be concluded that 0.07 m/s is the most appropriate velocity. When all the speeds including angular velocity and linear velocity become 0, the program will print the data of processing time. Therefore, the processing time will be recorded at the beginning and the velocity becomes 0. Since AGV will continue to move even after the velocity is set to 0, it is necessary to preserve the enough distance for AGV to stop. In comparison to 0.335m for 0.075 m/s, 0.318m for 0.08 m/s, and 0.22m for 0.085 m/s, 0.369m/s for 0.07 m/s is more applicable to avoid collision for emergencies. In addition,

information about the measured stop distance for these four groups of speeds respectively is also measured, which coincides with the distance given by the detector. Significantly, the detected distance is 0 when the speed is 0.085 m/s, which means that the tag is out of the camera view and AGV is too close to the tag. The speeds which are less than 0.07 m/s or around 0.07m/s are also tested. Referring to processing time, the figure for 0.07 m/s has the shortest time (5.99s). Therefore, 0.07 m/s is chosen.

Table 6.6: Testing results for 0.07 m/s

Processing Time	distance	z	x	velocity
0.002863891	0.59615	0.593255	0.020356	0.08
/	0.593282	0.592922	0.020348	0.08
/	0.592793	0.592432	0.020352	0.08
/	0.593479	0.593119	0.020358	0.08
/	0.592877	0.592517	0.020534	0.08
/	0.591422	0.591062	0.020316	0.08
/	0.593542	0.593181	0.020371	0.08
/	0.593541	0.593181	0.020363	0.08
/	0.593526	0.593165	0.020374	0.08
/	0.593672	0.593312	0.021070	0.08
/	0.587917	0.587519	0.021070	0.08
/	0.550388	0.549793	0.020214	0.08
/	0.518142	0.517170	0.018774	0
6.491357224	0.480052	0.478407	0.017876	0
6.991308976	0.440502	0.437808	0.016912	0
7.491303172	0.405400	0.401228	0.016912	0

Table 6.7: Testing results for 0.075 m/s

Processing Time	distance	z	x	velocity
0.00268864	0.592582	0.592496	0.00973	0.085
/	0.592923	0.592837	0.009704	0.085
/	0.592445	0.592446	0.009727	0.085
/	0.593843	0.593757	0.009726	0.085
/	0.593025	0.592939	0.009722	0.085
/	0.593896	0.593810	0.009728	0.085
/	0.593159	0.593072	0.009759	0.085
/	0.593453	0.593367	0.009721	0.085
/	0.593432	0.593345	0.009722	0.085
/	0.563150	0.562971	0.009251	0.085
/	0.501611	0.500749	0.008612	0
/	0.424879	0.421947	0.007304	0
7.486471176	0.369860	0.363917	0.005990	0
7.986708737	0.331119	0.321903	0.004867	0
8.486590321	0	0	0	0

6.2.3 Collision avoidance

This system selects Dijkstra algorithm for optimal path planning in global planner and Dynamic Window Approaches (DWA) to avoid collision and obstacles in local planner. The Dijkstra algorithm is greedy, implying that it only considers the distance from the current node to the next node, and uses traversal search to find the shortest path [177]. Thus, the obtained path has high reliability and good robustness, but the complexity is high and it may fall into local optimum. However, this project is conducted in a relatively simple and small indoor office, where local optimum is much preferred by choosing Dijkstra algorithm. DWA mainly samples multiple sets of velocities and then simulates the trajectory of the robot at these speeds [178]. After testing and evaluating sets of trajectories, the speed corresponding to the optimal trajectories is selected to drive the robot motion. Dynamic

window in this technology intends to control the speed sampling space within a feasible dynamic range according to the current acceleration and deceleration states of AGV [178].

This system can fulfill the function of avoiding collision of the static barrier in good performance. Setting a target point indicated by the green array in Figure 6.9, red line and green line represent for the global path and the local path respectively. However, only the position of the master AGV that performs the navigation can be displayed on the map. The position of the salve AGVs can be detected by the radar on master, and be viewed as an obstacle behind the master.

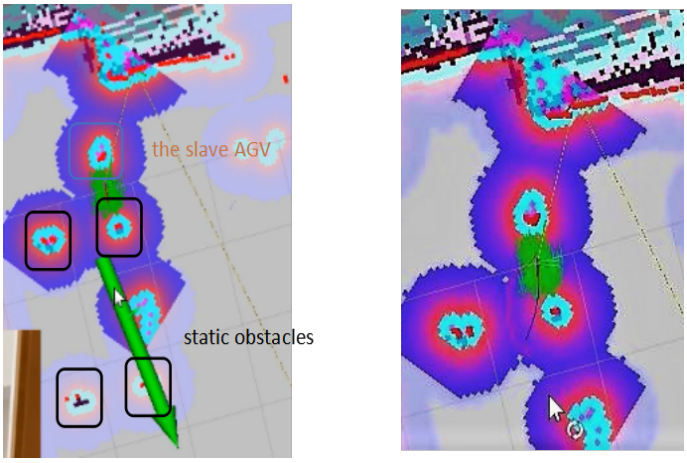

Figure 6.9: Collision avoidance for static obstacle

When a direction is set, a path is generated for AGV to achieve the set point with avoiding the dynamic obstacle. For example, the path represented in the green line bypasses the red obstacle bounded by the blue rectangle in Figure 6.10, instead of selecting the shortest path that directly towards the destination. Although the dynamic collision avoidance can be realized by master, it is hard for the whole system to accomplish this function. Due to the fast pre-defined velocity in ROS navigate package, the relatively-low speed of slaves (0.07 m/s) makes it difficult to tracking the master on time. In another words, lots of effort and time should be paid for finding how to adjust the speed in a complicated package. Nevertheless, the time for finishing this project is limited, so the dynamic collision avoidance function is not be realized by far.

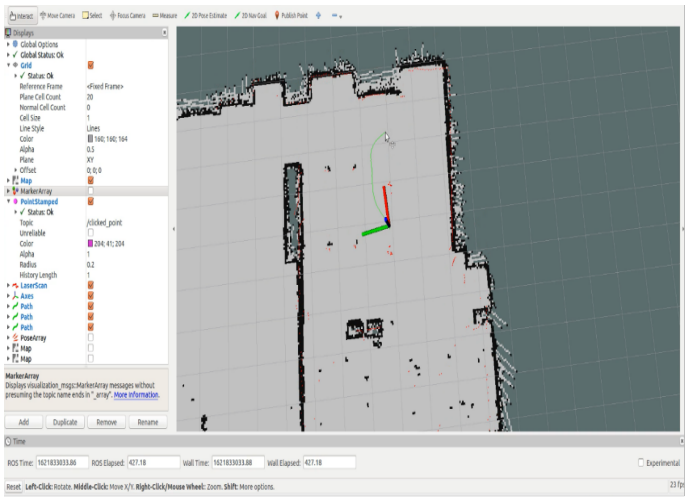

Figure 6.10: Collision avoidance for dynamic obstacle

Coordination

There are two formations for coordination of this system, one is line and another is triangle. Due to the master-slave mode, no communication exists between two slaves and all velocity controls are based on the tag which is attached on the previous vehicle.

Line In line formation, all three robots form in a line with the master at the head of the line. However, slave 2 will not be able to detect the tag attached the master due to the existing of slave 1. In order to solve this occlusion problem, there is also a tag attached on slave 1.

Rectangle In rectangle formation, it is similar to the line formation. Masters are in the front of each line with several slaves in the back. Because of the number of AGV prototypes is only three. This formation is only simulated without application.

Triangle In triangle formation, two slaves are on both sides of the back of the master. On account of the limitation existing in the camera view, the distance between two slaves is relatively small, so as to make sure the tag on the master is in the camera view during the movement. Although the cluster in this formation can move forward successfully, it is difficult to make a turn. For instance, if this system is required to make a left turn, the tag on the master will only exist in the view of slave 1. In this case, slave 2 will lost the object

to follow and finally be completely out of the system. This problem is mainly caused by the tracking function is totally relied on AprilTag detection, which is inflexible.

In summary, the designed system has completed three functionalities, including collision avoidance based on integrated navigation, tracking of AprilTag, and coordination in two forms. Before testing other functions, it completed basic movement and calibration of the master AGV. Basic movement by keyboard control is elemental for moving AGV to build the map, and the calibration for linear and angular speed is another important component for precise measurement to eliminate other environmental influence. The map of the experimental environment has already been built by SLAM technology with Hector algorithm, which was carried out in this study using AGV with a laser radar that can rotate with 360 degrees. The IMU sensor which determines the AGV position and posture in the indoor environment is used as an assistance to ensure the orientation of the master AGV prototype. As for integrated navigation using the laser radar and the IMU sensor, Dijkstra algorithm is selected for optimal path planning in global planner and DWA is applied to avoid collision and obstacles in local planner. Tracking function using AprilTag Tag36h11 family as the visual reference is the first step to complete coordination. Afterwards, the other two slave AGV prototypes were added in the system to accomplish coordination. According to two designed patterns, three AGV prototypes can form a triangle and a straight line to move and turn.

Chapter 7

Conclusions and Future work

In summary, this thesis is focused on establishing a scheduling optimization and coordination with a target tracking algorithm under a heterogeneous network in an AGV system. In the literature review, three major research aspects are outlined: a heterogeneous network environment, scheduling and optimization methods, and visual tracking methods. I have reviewed the state-of-the-art topics in the three research aspects. More importantly, I have made my contributions in each of the aspects.

7.1 Contributions and Findings

In a heterogeneous network environment area, I made a network selection optimization algorithm based on Analytical Hierarchy Process(AHP) and Technique for Order Preference by Similarity to an Ideal Solution(TOPSIS) in a heterogeneous network environment is presented. Meanwhile, an improved method for preventing CNN immune from backdoor attacks to ensure a multi-AGV system's communication environment is presented. Furthermore, a transmission framework for a multi-AGV system is presented. The system takes the message and big files apart, transmits the message through the cloud platform, and transmits the big file through the socket by the edge computers. Adding encryption and decryption of big data to the transmission can ensure the real-time nature of the message and the privacy of the big data at the same time. The testing results show the designed

framework can realize the message and object model message transmission between AGV and the cloud platform through MQTT to achieve real-time monitoring. It can also enable communication between devices in different network environments and cloud platforms. At the same time, it can realize the transfer of large files and the encryption and decryption of public and private keys from multiple devices to the same device The framework can run smoothly in the simulation environment and has good performance. This framework can be used in diversity environment.

In the aspect of scheduling and optimization methods, a multi-robot planning algorithm with quad tree map division for obstacles of irregular shape is presented. The innovation of the system design lies in the shortest path algorithm with combination of the waiting mode and motion coordination, as well as the new motion coordination method based on quad tree division. The designed algorithm not only has low time delay but also decreases the possibility of collision in the multi-robot system. Compared with general graph like Voronoi graph, it reduces computation complexity and is flexible for a multi-robot system. In addition, a scheduling optimization platform is presented. Related experiments and testings have been done. The superiority of the proposed scheduling optimization method has been proved. Meanwhile, parameters that can influence the systems performance have been tested. The handover mode is better than basic mode. Most parameters are correlated with the system's time cost except the number of AGVs. There exists the best number of AGVs in a multi-AGV system using handover mode. Therefore, the handover method should be paid more attention to be applied in multi-AGV systems' scheduling optimization area.

In the aspect of visual tracking method,an improved Camshift Algorithm is presented and applied to AGV prototypes. The experimental results demonstrated reduced tracking time delay of 12%. The stability and robustness of target tracking has a performance gain of 51.74%. The system architecture provides a solution to balance limited computation power of individual AGVs and the requirement of computation-intensive image processing for AGV tracking. The experimental results also revealed the challenges encountered in the current AGV tracking system design including speed limit and complexity in the color/shape of the target, which worth further investigation. Besides, mobile robot tracking

with three deep learning models under the specific environments is presented. I construct three recent object detection models, which are Faster RCNN, SSD and YOLOv5. Comprehensive experiments are conducted to evaluate the detection performance with our data sets. Subsequently, I design a Random Erasing method to create new testing data set with various partial occlusions to verify the system performance. I also use image views from different angles, distances and surrounding environments to test the model performance. Finally, I analyze numerically the adaptivity of the models against different degrees of occlusions through experiments. Experimental results showed that the SSD model has the best performance and is most promising for the addressed application.Subsequently, a novel method for real-time visual tracking and distance measurement based on SSD is proposed. This algorithm can measure the distance of the tracking object with high accuracy in real-time. The error can be controlled at a centimeter-level.

Finally, A tracking and collision avoidance system is applied in a multi-AGV system. The system has completed three functionalities, including collision avoidance based on integrated navigation, tracking of AprilTag, and coordination in two forms.

7.2 Future Work

The presented algorithms, methods, and systems are able to improve a multi-AGV system. This research could be further developed in the following aspects:

(1)In the future work, the designed network selection optimization algorithm on various terminals under heterogeneous network environments should be improved and run, so as to test the switching efficiency of the algorithm in practical application scenarios.

(2)The backdoor attack-resistant neural network model proposed in this thesis still has shortcomings. When the model has good anti-attack performance against a single trigger, but when there is more than one trigger in the poisoned dataset, using a single Softmax Classifier trained in the second model to identify the poisoned data is no longer sufficient to cope with the threat of data poisoning, and in reality, it is more effective to poison the model using multiple triggers. In future work, we may focus on how to make the model effectively determine the number and type of triggers used by attackers before learning to

recognize them and assign the corresponding classifiers.

(3)In the future, we plan to improve and apply the design framework to a real multi-AGV system to help the system to own a better communication environment.

(4)The future work is to expand our designed scheduling optimization method. We plan to optimize the current time cost calculator so that it can deal with the n-path situation. Meanwhile, we plan to add a parameter handover time to the time cost calculator with other specific parameters to better test what will influence a multi-AGV system and how can they influence the system. We plan to optimize the current simulation platform so that it can simulate and schedule robots with different capabilities, apply different path planning and obstacle avoidance algorithms, and add more specific conditions to make the simulation environment close to the real scene. Thus, the deployed robot system can be tested with more applicability.

(5)In future work, we will improve our deep learning visual tracking models' data sets. The size of the current data sets is small and the diversity of the backgrounds should be expanded. The performance of the current deep learning algorithms is not satisfactory in the detection of small objects. To address the issue, the deeper layers and shallower layers can be concatenated to enrich the semantic information of the shallower layers. In addition, some deep learning models have the structure of a single stage. They can be developed into a multi-stage solution that can increase the precision of detection and localization of small objects.

(6)In the future, SSD algorithms can be improved by adding anti-convolution layers to optimize the performance to detect small objects and increase their semantic comprehension ability. The distance measurement algorithms can be replaced by the monocular deep estimation algorithms and the experimental environment will not be limited to flat ground.

(7)Through numerical analysis, we identified the limitations and drawbacks of this system. In collision avoidance, the cluster of AGVs is only able to avoid obstacles that already exist on the map. For dynamic obstacles, only the master can avoid them since the time delay in tracking and relatively low speed in slaves prohibit them from following the master timely. In tracking, time delay not only in the detection but also in the data

transmission among nodes is the most serious problem leading to the relatively low speed in slaves and non-real-time detection. In coordination, the triangle formation needs more intelligent adjustment controlled by the master, in order to make a turn. Since the camera view is limited, coordination totally depending on AprilTag is not suitable to provide sufficient information on the positions of the two slaves. Currently, there are three created prototypes. The experiment environment is a small indoor environment. In the future, it is possible to use Gazebo simulator to simulate more prototypes and larger environments based on the current results to try to realize a larger AGV system, where Gazebo is a 3D dynamic simulator that accurately and effectively simulates crowds of robots in complex indoor and outdoor environments. Furthermore, the current tracking algorithm still has limitations and should be optimized. This is also a research point that should be paid attention to. The Apriltag used in the system is a relatively mature method. However, it is possible to improve the method to increase the detecting frequency of Apriltag to decrease time delay. Moreover, a communication method should be established between the master and slaves in the future. It is found that while AGV prototypes are forming a team or doing tasks as a team when accidents happen, the system could not solve them automatically because slaves totally depend on AprilTag and may be out of control. It makes the system more effective and easier to be established, but decreases the system's tolerate error.

Reference

[1] I. Draganjac, D. Mikli, Z. Kovai, G. Vasiljevi and S. Bogdan, "Decentralized Control of Multi-robot Systems in Autonomous Warehousing Applications," IEEE Transactions on Automation Science and Engineering, vol. 13, no. 4, pp. 1433-1447, Oct. 2016, doi: 10.1109/TASE.2016.2603781.

[2] H. Andreasson et al., "Autonomous transport vehicles: Where we are and what is missing",IEEE Robot. Autom. Mag., vol. 22, no. 1, pp. 64-75, Mar. 2015.

[3] T. T. Mac, C. Copot, D. T. Tran and R. De Keyser, "Heuristic approaches in robot path planning: A survey",Robot. Auto. Syst., vol. 86, pp. 13-28, Dec. 2016.

[4] S. Manca, A. Fagiolini and L. Pallottino, "Decentralized coordination system for multiple robots in a structured environment",IFAC Proc., vol. 44, no. 1, pp. 6005-6010, 2011.

[5] K. Zheng, D. Tang, W. Gu and M. Dai, "Distributed control of multi-robot system based on regional control model",Prod. Eng., vol. 7, no. 4, pp. 433-441, Jul. 2013.

[6] J. J Huang, T .Gwohshiung, Multiple attribute decision making[J]. Lecture Notes in Economics & Mathematical Systems, 1994, 404(4):287-288.

[7] V. Digani, L. Sabattini, C. Secchi and C. Fantuzzi, "Hierarchical traffic control for partially decentralized coordination of multi robot systems in industrial environments",Proc. IEEE Int. Conf. Robot. Autom. (ICRA), pp. 6144-6149, May/Jun. 2014.

[8] L. Cancemi, A. Fagiolini and L. Pallottino, "Distributed multi-level motion planning for autonomous vehicles in large scale industrial environments",Proc. IEEE 18th Conf. Emerg. Technol. Factory Autom. (ETFA), pp. 1-8, Sep. 2013.

[9] W. Malopolski, "A sustainable and conflict-free operation of robots in a square topology",Comput. Ind. Eng., vol. 126, pp. 472-481, Dec. 2018.

[10] J. Hershberger and S. Suri. An optimal algorithm for Euclidean shortest paths in the plane. SIAM Journal on Computing, 28(6):2215C2256, 1999.

[11] F. Bari and V. Leung, "Application of ELECTRE to Network Selection in A Hetereogeneous Wireless Network Environment", In the Proceedings of the 2007 IEEE Wireless Communications and Networking Conference (WCNC '07), pp. 3810-3815, 2007.

[12] D. Maroua, O. Mohammed and A. Driss, "VIKOR for multi-criteria network selection in heterogeneous wireless networks," 2016 International Conference on Wireless Networks and Mobile Communications (WINCOM), 2016, pp. 82-86, doi: 10.1109/WINCOM.2016.7777195.

[13] M. Cirillo, F. Pecora, H. Andreasson, T. Uras and S. Koenig, "Integrated motion planning and coordination for industrial vehicles",Proc. 24th Int. Conf. Autom. Planning Scheduling, 2014.

[14] A. Jayant, P. Gupta, S. K. Garg and M. Khan, "TOPSIS-AHP Based Approach for Selection of Reverse Logistics Service Provider: A Case Study of Mobile Phone Industry", Procedia Engineering, vol. 97, pp. 2147-2156, 2014.

[15] C. Prakash and M. K. Barua, "Integration of AHP-TOPSIS method for prioritizing the solutions of reverse logistics adoption to overcome its barriers under fuzzy environment", Journal of Manufacturing Systems, vol. 37, pp. 599-615, 2015.

[16] Y. S. Bagi, S. Suyono and M. F. Tomatala, "Decision Support System for High Achieving Students Selection Using AHP and TOPSIS," 2020 2nd International

Conference on Cybernetics and Intelligent System (ICORIS), 2020, pp. 1-5, doi: 10.1109/ICORIS50180.2020.9320823.

[17] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In CVPR, pages 770C778, 2016.

[18] Y. Zhang and B. Wallace, "A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification", arXiv.org, 2014. [Online]. Available: https://arxiv.org/abs/1510.03820

[19] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. Deep speech 2: End-to-end speech recognition in english and mandarin. In ICML, pages 173C182, 2016.

[20] Dsdsd B. Biggio, B. Nelson and P. Laskov, "Poisoning attacks against support vector machines", Proceedings of the 29th International Conference on International Conference on Machine Learning, pp. 1467-1474, 2012

[21] L. Yingqi, M. Shiqing, A. Yousra, L. Wen-Chuan, Z. Juan, W. Weihang, et al., "Trojaning attack on neural networks", Proceedings of Network and Distributed System Security Symposium (NDSS), 2018.

[22] X. Chen, C. Liu, B. Li, K. Lu and D. Song, "Targeted Backdoor Attacks on Deep Learning Systems Using Data Poisoning", arXiv.org, 2017. [Online]. Available: https://arxiv.org/abs/1712.05526v1.

[23] S. Cheng and M. Tseng, "Analysis of invisible data poisoning backdoor attacks against malware classifiers," in Proc. HITCON 2021, Taipei, Taiwan, Nov. 2021.

[24] T. Brown, D. Man, A. Roy, M. Abadi and J. Gilmer, "Adversarial Patch", arXiv.org, 2022. [Online]. Available: https://arxiv.org/abs/1712.09665. [Accessed: 17- Aug- 2022].

[25] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus. Intriguing properties of neural networks. ICLR 2014. arXiv:1312.6199

[26] Y. Gao, B. Doan, Z. Zhang, S. Ma, J. Zhang, A. Fu, S. Nepal and H. Kim, "Backdoor Attacks and Countermeasures on Deep Learning: A Comprehensive Review", arXiv.org, 2020. [Online]. Available: https://arxiv.org/abs/2007.10760.

[27] M. Jagielski, G. Severi, N. Harger and A. Oprea, "Subpopulation Data Poisoning Attacks", arXiv.org, 2020. [Online]. Available: https://arxiv.org/abs/2006.14026.

[28] T. Gu, B. Dolan-Gavitt, and S. Garg, Badnets: Identifying vulnerabilities in the machine learning model supply chain, in Proc. of Machine Learning and Computer Security Workshop, 2017

[29] Y. Li, Y. Jiang, Z. Li and S. Xia, "Backdoor Learning: A Survey", arXiv.org, 2020. [Online]. Available: https://arxiv.org/abs/2007.08745.

[30] A. Turner, D. Tsipras and A. Madry, "Label-Consistent Backdoor Attacks", arXiv.org, 2019. [Online]. Available: https://arxiv.org/abs/1912.02771.

[31] J. Zhu, R. Kaplan, J. Johnson and L. Fei-Fei, "HiDDeN: Hiding Data With Deep Networks", arXiv.org, 2018. [Online]. Available: https://arxiv.org/abs/1807.09937.

[32] K. Liu, B. Dolan-Gavitt and S. Garg, "Fine-Pruning: Defending Against Backdooring Attacks on Deep Neural Networks", 2018.

[33] Y. Liu, Y. Xie and A. Srivastava, "Neural Trojans," 2017 IEEE International Conference on Computer Design (ICCD), 2017, pp. 45-48, doi: 10.1109/ICCD.2017.16.

[34] Y. Li, T. Zhai, Y. Jiang, Z. Li and S. Xia, "Backdoor Attack in the Physical World", arXiv.org, 2021. [Online]. Available: https://arxiv.org/abs/2104.02361.

[35] B. Wang et al., "Neural Cleanse: Identifying and Mitigating Backdoor Attacks in Neural Networks," 2019 IEEE Symposium on Security and Privacy (SP), 2019, pp. 707-723, doi: 10.1109/SP.2019.00031.B. Chen et al., "Detecting Backdoor Attacks on Deep Neural Networks by Activation Clustering," ed, 2018.

[36] K. Grosse, T. Lee, B. Biggio, Y. Park, M. Backes and I. Molloy, "Backdoor smoothing: Demystifying backdoor attacks on deep neural networks", Computers & Security, vol. 120, p. 102814, 2022.

[37] J. Cohen, E. Rosenfeld and J. Kolter, "Certified Adversarial Robustness via Randomized Smoothing", arXiv.org, 2022. [Online]. Available: https://arxiv.org/abs/1902.02918v2. [Accessed: 17- Aug- 2022].

[38] B. Wang, X. Cao, J. jia and N. Gong, "On Certifying Robustness against Backdoor Attacks via Randomized Smoothing", arXiv.org, 2020. [Online]. Available: https://arxiv.org/abs/2002.11750.

[39] K. M. Ahmed, A. Imteaj and M. H. Amini, "Federated Deep Learning for Heterogeneous Edge Computing," 2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA), 2021, pp. 1146-1152, doi: 10.1109/ICMLA52953.2021.00187.

[40] D. Boneh and G. Durfee, "Cryptanalysis of RSA with private key d less than N/sup 0.292/," in IEEE Transactions on Information Theory, vol. 46, no. 4, pp. 1339-1349, July 2000, doi: 10.1109/18.850673.

[41] A Imteaj ,M H Amini . FedAR: Activity and Resource-Aware Federated Learning Model for Distributed Mobile Robots[J]. 2021.

[42] W Liu ,L Chen ,Y Chen, et al. Accelerating Federated Learning via Momentum Gradient Descent[J]. 2019.

[43] M. Gharibi and P. Rao, "RefinedFed: A Refining Algorithm for Federated Learning," 2020 IEEE Applied Imagery Pattern Recognition Workshop (AIPR), 2020, pp. 1-5, doi: 10.1109/AIPR50011.2020.9425094.

[44] S. B. Azmy, A. Abutuleb, S. Sorour, N. Zorba and H. S. Hassanein, "Optimal Transport for UAV D2D Distributed Learning: Example using Federated Learning," ICC 2021 - IEEE International Conference on Communications, 2021, pp. 1-6, doi: 10.1109/ICC42927.2021.9500304.

[45] M. A. Ferrag, O. Friha, L. Maglaras, H. Janicke and L. Shu, "Federated Deep Learning for Cyber Security in the Internet of Things: Concepts, Applications, and Experimental Analysis," in IEEE Access, vol. 9, pp. 138509-138542, 2021, doi: 10.1109/ACCESS.2021.3118642.

[46] Z. Xin and T. Xiaofei "Research and implementation of RSA algorithm for encryption and decryption," Proceedings of 2011 6th International Forum on Strategic Technology, 2011, pp. 1118-1121, doi: 10.1109/IFOST.2011.6021216.

[47] D., Dorothy E . Digital Signatures with RSA and Other Public-Key Cryptosystems[J]. Communications of the Acm, 1984, 27(4):388-392.

[48] M D Shieh , J H Chen ,H H Wu , et al. A New Modular Exponentiation Architecture for Efficient Design of RSA Cryptosystem[J]. IEEE Transactions on Very Large Scale Integration Systems, 2008, 16(9):1151-1161.

[49] F. Mo, Y. -C. Hsu, H. -H. Chang, S. -C. Pan, J. -J. Yan and T. -L. Liao, "Design of an Improved RSA Cryptosystem Based on Synchronization of Discrete Chaotic Systems," 2016 International Conference on Information System and Artificial Intelligence (ISAI), 2016, pp. 9-13, doi: 10.1109/ISAI.2016.0012.

[50] R. Imam, Q. M. Areeb, A. Alturki and F. Anwer, "Systematic and Critical Review of RSA Based Public Key Cryptographic Schemes: Past and Present Status," in IEEE Access, vol. 9, pp. 155949-155976, 2021, doi: 10.1109/ACCESS.2021.3129224.

[51] E. Gustafsson and A. Jonsson, "Always best connected", IEEE Wireless Communications, vol. 10, no. 1, pp. 49-55, 2003.

[52] A. Sgora, D. D. Vergados and P. Chatzimisios, "An access network selection algorithm for heterogeneous wireless environments," The IEEE symposium on Computers and Communications, 2010, pp. 890-892, doi: 10.1109/ISCC.2010.5546556.

[53] K. Piamrat, A. Ksentini, J.-M. Bonnin and C. Viho, "Radio resource management in emerging heterogeneous wireless networks", Comput. Commun, vol. 34, pp. 1066-1076, 2011.

[54] L. Chen and H. Li, "An MDP-based vertical handoff decision algorithm for heterogeneous wireless networks," 2016 IEEE Wireless Communications and Networking Conference, 2016, pp. 1-6, doi: 10.1109/WCNC.2016.7564804.

[55] S. Dhar Roy and S. R. Vamshidhar Reddy, "Signal Strength Ratio Based Vertical Handoff Decision Algorithms in Integrated Heterogeneous Networks[J]", Wireless Personal Communications, vol. 77, no. 4, pp. 2565-2585, 2014.

[56] S. Xin, L. Wenmin, Z. Minglei and L. Feng , "A network selection algorithm based on FAHP/GRA in heterogeneous wireless networks," 2016 2nd IEEE International Conference on Computer and Communications (ICCC), 2016, pp. 1445-1449, doi: 10.1109/CompComm.2016.7924942.

[57] K. Savitha and C. Chandrasekar, "Vertical Handover decision schemes using SAW and WPM for Network selection in Heterogeneous Wireless Network", Global Journal of Computer Scienceand Technology, vol. 11, no. 9, pp. 1-7, May 2011.

[58] C. L. Hwang and K. Yoon, Multiple attribute decision making: Methods and applications A State of the Art Survey, New York:Springer-Verlag, 1981, ISBN 3540105581.

[59] S Zhao, W Shi and S Fan, "A GRA-based network selection mechanism in heterogeneous wireless networks", Computer Mechatronics Control and Electronic Engineering (CMCE) 2010 International Conference on, pp. 215-218, 2010

[60] J. Zajc and W. Maopolski, "Structural on-line control policy for collision and deadlock resolution in multi-AGV systems," Journal of Manufacturing Systems, Article vol. 60, pp. 80-92, 07/01/July 2021 2021, doi: 10.1016/j.jmsy.2021.05.002.

[61] M. De Ryck, M. Versteyhe, and F. Debrouwere, "Automated guided vehicle systems, state-of-the-art control algorithms and techniques," Journal of Manufacturing Systems, Review Article vol. 54, pp. 152-173, 01/01/January 2020 2020, doi: 10.1016/j.jmsy.2019.12.002.

[62] M.-h. Yuan, Y.-d. Li, F.-q. Pei, and W.-b. Gu, "Dual-resource integrated scheduling method of AGV and machine in intelligent manufacturing job shop," Integrated

scheduling problem of AGV and machine dual resources in intelligent manufacturing workshop, Original Paper vol. 28, no. 8, p. 2423, 2021, doi: 10.1007/s11771-021-4777-8.

[63] M. Li et al., "Decentralized Multi-AGV Task Allocation based on Multi-Agent Reinforcement Learning with Information Potential Field Rewards," ed, 2021.

[64] M. De Ryck, D. Pissoort, T. Holvoet, and E. Demeester, "Decentral task allocation for industrial AGV-systems with resource constraints," Journal of Manufacturing Systems, Article vol. 59, pp. 310-319, 04/01/April 2021 2021, doi: 10.1016/j.jmsy.2021.03.008.

[65] G. Li, X. Li, L. Gao, and B. Zeng, "Tasks assigning and sequencing of multiple AGVs based on an improved harmony search algorithm," Journal of Ambient Intelligence and Humanized Computing, Original Paper vol. 10, no. 11, p. 4533, 2019, doi: 10.1007/s12652-018-1137-0.

[66] Y. Bao, G. Jiao, and M. Huang, "Cooperative optimization of pod repositioning and AGV task allocation in Robotic Mobile Fulfillment Systems," ed: IEEE, 2021, pp. 2597-2601.

[67] N. Fazal, M. T. Khan, S. Anwar, J. Iqbal, and S. Khan, "Task allocation in multi-robot system using resource sharing with dynamic threshold approach," PLoS ONE, Article vol. 17, no. 5, pp. 1-22, 2022, doi: 10.1371/journal.pone.0267982.

[68] Y. Lian, L. Zhang, W. Xie, and K. Wang, "An Improved Heuristic Path Planning Algorithm for Minimizing Energy Consumption in Distributed Multi-AGV Systems," ed: IEEE, 2020, pp. 70-75.

[69] M. Majdi, H. S. Anvar, R. Barzamini and S. Soleimanpour, "Multi AGV path planning in unknown environment using fuzzy inference systems," 2008 3rd International Symposium on Communications, Control and Signal Processing, 2008, pp. 172-177, doi: 10.1109/ISCCSP.2008.4537214.

[70] D. Yu, X. Hu, K. Liang, and J. Ying, "A parallel algorithm for multi-AGV systems," Journal of Ambient Intelligence and Humanized Computing, Original Paper pp. 1-15, 06/24/ 2021, doi: 10.1007/s12652-021-02987-3.

[71] V. Digani, L. Sabattini, C. Secchi and C. Fantuzzi, "Ensemble Coordination Approach in Multi-AGV Systems Applied to Industrial Warehouses," in IEEE Transactions on Automation Science and Engineering, vol. 12, no. 3, pp. 922-934, July 2015, doi: 10.1109/TASE.2015.2446614.

[72] C. Wang et al., "Path planning of automated guided vehicles based on improved A-Star algorithm", 2015 IEEE

[73] G. Qing, Z. Zheng and X. Yue, "Path-planning of automated guided vehicle based on improved Dijkstra algorithm", 2017 29th Chinese Control And Decision Conference (CCDC), pp. 7138-7143, 2017.

[74] J Li, T Dong, Y Li et al., "Study on Robot Path Collision Avoidance Planning Based on the Improved Ant Colony Algorithm[C]//", International Conference on Intelligent Human-Machine Systems and Cybernetics, 2016.

[75] X. Liyun, W. Ning, and L. Xufeng, "Study on Conflict-free AGVs Path Planning Strategy for Workshop Material Distribution Systems," Procedia CIRP, vol. 104, pp. 1071-1076, 05// 2021, doi: 10.1016/j.procir.2021.11.180.

[76] J. Luo, Y. Wan, W. Wu, and Z. Li, "Optimal Petri-Net Controller for Avoiding Collisions in a Class of Automated Guided Vehicle Systems," IEEE Transactions on Intelligent Transportation Systems, Intelligent Transportation Systems, IEEE Transactions on, IEEE Trans. Intell. Transport. Syst., Periodical vol. 21, no. 11, pp. 4526-4537, 11/01/ 2020, doi: 10.1109/TITS.2019.2937058.

[77] KZL M , Zbayrak M , Papadopoulou T C . Evaluation of dispatching rules for cellular manufacturing[J]. The International Journal of Advanced Manufacturing Technology, 2006, 28(9-10):985-992.

[78] Song S , Li A , Xu L . AGV Dispatching Strategy Based on Theory of Constraints[C]// IEEE Conference on Robotics. IEEE, 2008:922-925

[79] Egbelu, P. J., & Tanchoco, J. M. A. (1984). Characterization of automatic guided vehicle dispatching rules. International Journal of Production Research, 22(3), 359-374. https://doi.org/10.1080/00207548408942459]

[80] Egbelu P J . Pull versus push strategy for automated guided vehicle load movement in a batch manufacturing system[J]. Journal of Manufacturing Systems, 1987, 6(3):209-221.

[81] Co C G , Tanchoco J . A Review of Research on AGVS Vehicle Management[J]. Engineering Costs & Production Economics, 1991, 21(1):35-42.

[82] Koo, Pyung-Hoi & Jang, Jaejin & Suh, Jungdae. (2005). Vehicle dispatching for highly loaded semiconductor production considering bottleneck machines first. International Journal of Flexible Manufacturing Systems. 17. 23-38. 10.1007/s10696-005-5992-6.

[83] Marvizadeh, S. & Choobineh, Fred. (2014). Entropy-based dispatching for automatic guided vehicles. International Journal of Production Research. 52. 3303-3316. 10.1080/00207543.2013.871590.

[84] Mahnaz, J. D. ; Rahebe, N. A.. Moving object tracking based on mean shift algorithm and features fusion. International Symposium on Artificial Intelligence and Signal Processing (AISP). 2011. Page(s): 48 C 53. DOI: 10.1109/AISP.2011.5960981.

[85] Xiu, C.; Su, X.; Pan, X.. Improved target tracking algorithm based on Camshift. Chinese Control and Decision Conference (CCDC). Shenyang. China. 9-11 June 2018.

[86] Mahnaz, J. D.; Rahebe, N. A.. Moving object tracking based on mean shift algorithm and features fusion. International Symposium on Artificial Intelligence and Signal Processing (AISP). Tehran. Iran. 15-16 June 2011.

[87] Guan, W.; Liu, Z.; Wen, S.; Xie, H.; Zhang, X.. Visible Light Dynamic Positioning Method Using Improved Camshift-Kalman Algorithm. IEEE Photonics Journal. 2019. Volume: 11, Issue: 6. DOI: 10.1109/JPHOT.2019.2944080.

[88] Mark, V.. Bright future for agvs. Engineering & Technology. 2008. Volume: 3 , Issue: 11. Page(s): 48 C 50. DOI: 10.1049/et:20081105.

[89] Deilamani, M.J. ; Asli, R.N., Moving object tracking based on mean shift algorithm and features fusion, Artificial Intelligence and Signal Processing, Tehran,Germany. 15-16 June 2011.

[90] Guo, B.;Feng, X.. Color target tracking strategy in Binocular stereo vision based on Camshift algorithm. 2012 24th Chinese Control and Decision Conference (CCDC). Taiyuan,China. 23-25 May 2012.

[91] Ultrasonic Ranging Module HC - SR04. Available online: https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf (accessed on 1 May 2020)

[92] Guan, W.; Liu, Z.; Wen, S.; Xie, H.; Zhang, X.. Visible Light Dynamic Positioning Method Using Improved Camshift-Kalman Algorithm. 2019. IEEE Photonics Journal. DOI:10.1109/JPHOT.2019.2944080.

[93] Mahnaz, J. D.; Rahebe, N. A.. Moving object tracking based on mean shift algorithm and features fusion. 2011 International Symposium on Artificial Intelligence and Signal Processing (AISP). Tehran, Iran. 15-16 June 2011.

[94] Hu, K.; Fan, E.; Ye, J.; Pi, J.; Zhao, L.; Shen, S.. Element-Weighted Neutrosophic Correlation Coefficient and Its Application in Improving CAMShift Tracker in RGBD Video. Information (MDPI). 2018, Volume: 9, Page 126. DOI10.3390/info9050126.

[95] Huang, M.; Guan, W.; Fan, Z.; Chen, Z.; Li, J.; Chen, B.. Improved Target Signal Source Tracking and Extraction Method Based on Outdoor Visible Light Communication Using a Cam-Shift Algorithm and Kalman Filter. Sensors (MDPI). 2018. 28 November. DOI10.3390/s18124173.

[96] KlanAr G , Kristan M , Karba R . Wide-angle camera distortions and non-uniform illumination in mobile robot tracking. Robotics and Autonomous Systems, 2004, 46(2):125-133.

[97] Jaulin K, Dominique W . Guaranteed mobile robot tracking using interval analysis. MISC99 workshop on application of interval analysis to system & control, 1999.

[98] Hautop H, Esther L, Cuenca V, et al. A simple real-time mobile robot tracking system. 1996. Available online: http://citeseerx.ist.psu.edu/viewdoc/versionsdoi=10.1.1.47.9920.

[99] Zhong Z, Zheng L, Kang G, et al. Random erasing data augmentation. Proceedings of the AAAI Conference on Artificial Intelligence, 2017, 34(7).

[100] Wang H.,Peng J.,Zheng X. and Yue S., "A Robust Visual System for Small Target Motion Detection Against Cluttered Moving Backgrounds," in IEEE Transactions on Neural Networks and Learning Systems, vol. 31, no. 3, pp. 839-853, March 2020, doi: 10.1109/TNNLS.2019.2910418.

[101] Li J.,Pan Z., Zhang Z. and Zhang H., "Dynamic ARMA-Based Background Subtraction for Moving Objects Detection," in IEEE Access, vol. 7, pp. 128659-128668, 2019, doi: 10.1109/ACCESS.2019.2939672.

[102] Yuan X., Guo J. , Hao X. and Chen H., Traffic sign detection via graph-based ranking and segmentation algorithms, IEEE Transactions on Systems, Man, and Cybernetics: Systems. vol. 45, no. 12, pp. 1509-1521, Dec. 2015, DOI: 10.1109/TSMC.2015.2427771.

[103] Cui Y, You H E, Tang T, et al. A new target tracking filter based on deep learning. Chiese Journal of Aeronautics, vol. 35, no. 5, pp. 11-24, 2022.

[104] Cheng S, Sun J X, Cao Y G, et al. Target tracking based on incremental deep learning. Optics and Precision Engineering, 2015, 23(4):1161-1170.

[105] Kwan C, Chou B, Kwan L. A comparative study of conventional and deep learning target tracking algorithms for low quality videos, International Symposium on Neural Networks. 2018.

[106] Wang P, Yuille AL (2016) DOC: deep occlusion estimation from a single image. ECCV (1), Springer, Lecture Notes in Computer Science, vol 9905, pp 545C561

[107] Ren X, Fowlkes C C, Malik J . Figure/Ground assignment in natural images, Computer Vision - ECCV 2006, 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006, Proceedings, Part II. Springer, Berlin, Heidelberg, 2006.

[108] Hoiem D. ,Stein A. N. ,Efros A. A. and Hebert M., Recovering occlusion boundaries from a single image, 2007 IEEE 11th International Conference on Computer Vision, 2007, pp. 1-8, DOI: 10.1109/ICCV.2007.4408985.

[109] Kotsia I , Buciu I , Pitas I . An analysis of facial expression recognition under partial facial image occlusion. Image & Vision Computing, 2008, 26(7):1052-1067.

[110] W. Zhang, Y. Zheng, Q. Gao and Z. Mi, "Part-Aware Region Proposal for Vehicle Detection in High Occlusion Environment," in IEEE Access, vol. 7, pp. 100383-100393, 2019, doi: 10.1109/ACCESS.2019.2929432.

[111] Yuille, Alan L. and Chenxi Liu, arXiv, Deep Nets: What have they ever done for vision,2018, Submitted on 10 May 2018 (v1), last revised 25 Nov 2020 (this version, v4), Available online: https://arxiv.org/abs/1805.04025

[112] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2017. ImageNet classification with deep convolutional neural networks. Commun. ACM 60, 6 (June 2017), 84C90. https://doi.org/10.1145/3065386

[113] Girshick R., Faster RCNN, in Proc. IEEE Int. Conf. Comput. Vis., 2015, pp. 1440C1448.

[114] Liu W, Anguelov D, Erhan D, et al. SSD: Single Shot MultiBox Detector// European Conference on Computer Vision. Springer, Cham, 2016.

[115] Joseph Nelson, Jacob Solawetz, JUN 10, 2020, YOLOv5 is Here: State-of-the-Art Object Detection at 140 FPS, Available online: https://blog.roboflow.com/yolov5-is-here/.

[116] Ross G., Dectinc C., Max E., 23 Jan 2018, 2022 GitHub, Inc. Available online: https://github.com/rbgirshick/py-Faster RCNN.

[117] Yang F.,Chen H., Li J.,Li F.,Wang L. and Yan X., Single Shot Multibox Detector with Kalman Filter for online pedestrian detection in video, IEEE Access, vol. 7, pp. 15478-15488, 2019, DOI: 10.1109/ACCESS.2019.2895376.

[118] Ma W.,Wang X. and Yu J., A lightweight feature fusion Single Shot Multibox Detector for Garbage Detection, IEEE Access, vol. 8, pp. 188577-188586, 2020, DOI: 10.1109/ACCESS.2020.3031990.

[119] Redmon J., Divvala S., Girshick R. and Farhadi A., You Only Look Once: Unified, Real-Time Object Detection, 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779-788, DOI: 10.1109/CVPR.2016.91.

[120] Joseph Nelson, Jacob Solawetz, JUN 10, 2020, YOLOv5 is Here: State-of-the-Art Object Detection at 140 FPS, Available online: https://blog.roboflow.com/yolov5-is-here/.

[121] Joseph Nelson, Jacob Solawetz, JUN 10, 2020, YOLOv5 is Here: State-of-the-Art Object Detection at 140 FPS, Available online: https://blog.roboflow.com/yolov5-is-here/.

[122] Glenn Jocher, Ultralytics,12 Oct 2021, 2022 GitHub, Inc. Available online: https://github.com/ultralytics/yolov5.

[123] Mai X., Zhang H., Jia X. and Meng M. Q., Faster RCNN with classifier fusion for automatic detection of small fruits, IEEE Transactions on Automation Science and Engineering, vol. 17, no. 3, pp. 1555-1569, July 2020, DOI: 10.1109/TASE.2020.2964289.

[124] Nsaif A K, Ali S, Jassim K N, et al. FRCNN-GNB: Cascade Faster RCNN with Gabor Filters and Nave Bayes for enhanced eye detection. IEEE Access, 2021, 9:15708-15719.

[125] Kim J and Cho J. RGDiNet: Efficient onboard object detection with Faster RCNN for air-to-ground surveillance. Sensors, 2021, 21(5):1677.

[126] Wang Y., Wang C. and Zhang H., Combining single shot multibox detector with transfer learning for ship detection using Sentinel-1 images, 2017 SAR in Big Data Era: Models, Methods and Applications (BIGSARDATA), 2017, pp. 1-4, DOI: 10.1109/BIGSARDATA.2017.8124924.

[127] Jubayer M F, Soeb M, Paul M K, et al. Mold detection on food surfaces using YOLOv5 . 2021.

[128] Lema D G, Pedrayes O D, Usamentiaga R, et al. Cost-Performance evaluation of a recognition service of livestock activity using aerial images. Remote Sensing, 2021, 13(12):2318.

[129] Shetty A. K. , Saha I., Sanghvi R. M.,Save S. A. and Patel Y. J., "A Review: Object Detection Models," 2021 6th International Conference for Convergence in Technology (I2CT), 2021, pp. 1-8, DOI: 10.1109/I2CT51068.2021.9417895.

[130] R.Wudhikarn, P. Charoenkwan and K. Malang, "Deep Learning in Barcode Recognition: A Systematic Literature Review," in IEEE Access, vol. 10, pp. 8049-8072, 2022, doi: 10.1109/ACCESS.2022.3143033.

[131] H. Kato and K. T. Tan, "First read rate analysis of 2D-barcodes for camera phone applications as a ubiquitous computing tool.," TENCON 2007 - 2007 IEEE Region 10 Conference, 2007, pp. 1-4, doi: 10.1109/TENCON.2007.4428778.

[132] T. Yuan, Y. Wang, K. Xu, R. R. Martin and S. -M. Hu, "Two-Layer QR Codes," in IEEE Transactions on Image Processing, vol. 28, no. 9, pp. 4413-4428, Sept. 2019, doi: 10.1109/TIP.2019.2908490.

[133] ISO/IEC 18004:2000. Information technonlogy-Automatic identification and data capture techniques-Bar code Symbology-QR Code,2000.

[134] Yue Liu, Ju Yang and Mingjun Liu, "Recognition of QR Code with mobile phones," 2008 Chinese Control and Decision Conference, 2008, pp. 203-206, doi: 10.1109/C-CDC.2008.4597299.

[135] F. Bergamasco, A. Albarelli, E. Rodol and A. Torsello, "RUNE-Tag: A high accuracy fiducial marker with strong occlusion resilience," CVPR 2011, 2011, pp. 113-120, doi: 10.1109/CVPR.2011.5995544.

[136] M. Fiala, "Designing Highly Reliable Fiducial Markers," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 32, no. 7, pp. 1317-1324, July 2010, doi: 10.1109/TPAMI.2009.146.

[137] Olson E . AprilTag: A robust and flexible visual fiducial system[C]// Robotics and Automation (ICRA), 2011 IEEE International Conference on. IEEE, 2011.

[138] M. Fiala, "Designing Highly Reliable Fiducial Markers," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 32, no. 7, pp. 1317-1324, July 2010, doi: 10.1109/TPAMI.2009.146.

[139] M. Fiala, "Designing Highly Reliable Fiducial Markers," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 32, no. 7, pp. 1317-1324, July 2010, doi: 10.1109/TPAMI.2009.146.

[140] Fiala M . ARTag, a fiducial marker system using digital techniques[C]// 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). IEEE, 2005.

[141] J. Chen, Y. Gao and S. Li, "Real-time Apriltag Inertial Fusion Localization for Large Indoor Navigation," 2020 Chinese Automation Congress (CAC), 2020, pp. 6912-6916, doi: 10.1109/CAC51589.2020.9326501.

[142] H. Duan, Y. Zhang and W. Sheng, "Image Digital Zoom Based Single Target Apriltag Recognition Algorithm in Large Scale Changes on the Distance," 2019 1st International Conference on Industrial Artificial Intelligence (IAI), 2019, pp. 1-6, doi: 10.1109/ICIAI.2019.8850822.

[143] G. Yu, Y. Hu and J. Dai, "TopoTag: A Robust and Scalable Topological Fiducial Marker System," in IEEE Transactions on Visualization and Computer Graphics, vol. 27, no. 9, pp. 3769-3780, 1 Sept. 2021, doi: 10.1109/TVCG.2020.2988466.

[144] Hernandez-De-Menendez M , Morales-Menendez R , Escobar C A , et al. Competencies for Industry 4.0[J]. International Journal for Interactive Design and Manufacturing (IJIDeM), 2020, 14:1511-1524.A. Sgora, D. D. Vergados and P. Chatzimisios, "An access network selection algorithm for heterogeneous wireless environments," The IEEE symposium on Computers and Communications, 2010, pp. 890-892, doi: 10.1109/ISCC.2010.5546556.

[145]]Babenko B , Yang M H , Belongie S . Visual tracking with online Multiple Instance Learning[C]// Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on. IEEE, 2009.

[146] J.F. Henriques, R. Caseiro, P. Martins and J. Batista, "High-Speed Tracking with Kernelized Correlation Filters," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 37, no. 3, pp. 583-596, 1 March 2015, doi: 10.1109/TPAMI.2014.2345390.

[147] Z. Kalal, K. Mikolajczyk and J. Matas, "Tracking-Learning-Detection," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 34, no. 7, pp. 1409-1422, July 2012, doi: 10.1109/TPAMI.2011.239.

[148] D. S. Bolme, J. R. Beveridge, B. A. Draper and Y. M. Lui, "Visual object tracking using adaptive correlation filters," 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2010, pp. 2544-2550, doi: 10.1109/CVPR.2010.5539960.

[149] Ross B. Girshick,Dectinc Chen, LoneStar, Max Ehrlich,23 Jan 2018, 2022 GitHub, Inc. Available online: https://github.com/rbgirshick/py-faster-rcnn.

[150] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779-788, doi: 10.1109/CVPR.2016.91.

[151] Liu W , Anguelov D , Erhan D , et al. SSD: Single Shot MultiBox Detector[C]// European Conference on Computer Vision. Springer, Cham, 2016.

[152] Ren S , He K , Girshick R , et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2017, 39(6):1137-1149.

[153] Joseph Nelson, Jacob Solawetz ,JUN 10, 2020, YOLOv5 is Here: State-of-the-Art Object Detection at 140 FPS, Available online: https://blog.roboflow.com/yolov5-is-here/.

[154] Glenn Jocher, ultralytics,12 Oct 2021, 2022 GitHub, Inc. Available online: https://github.com/ultralytics/yolov5.

[155] Jubayer M F , Soeb M , Paul M K , et al. Mold Detection on Food Surfaces Using YOLOv5. 2021.

[156] B. Chen et al., "Detecting Backdoor Attacks on Deep Neural Networks by Activation Clustering," ed, 2018.

[157] L. Bottou et al., "Comparison of classifier methods: a case study in handwritten digit recognition," Proceedings of the 12th IAPR International Conference on Pattern Recognition, Vol. 3 - Conference C: Signal Processing (Cat. No.94CH3440-5), 1994, pp. 77-82 vol.2, doi: 10.1109/ICPR.1994.576879.

[158] X. Ou, H. Ling, L. Yan and M. Liu, "Convolutional neural codes for image retrieval," Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2014 Asia-Pacific, 2014, pp. 1-10, doi: 10.1109/APSIPA.2014.7041557.

[159] Y. Wang, Y. Wang, H. Li, Z. Cai, X. Tang and Y. Yang, "CNN Hyperparameter Optimization Based on CNN Visualization and Perception Hash Algorithm," 2020 19th International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES), 2020, pp. 78-82, doi: 10.1109/DCABES50732.2020.00029.

[160] X. Ding, Y. Guo, G. Ding and J. Han, "ACNet: Strengthening the Kernel Skeletons for Powerful CNN via Asymmetric Convolution Blocks," 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 1911-1920, doi: 10.1109/ICCV.2019.00200.

[161] B. Mishra and A. Kertesz, "The Use of MQTT in M2M and IoT Systems: A Survey," in IEEE Access, vol. 8, pp. 201071-201086, 2020, doi: 10.1109/ACCESS.2020.3035849.

[162] James T. Lin and Meng-Wei Hsyu, A Support Vector Machine Approach for AGV Dispatching, IAM 2018 : International Conference on Innovation and Management (IAM2018 Summer). 2018.

[163] Valtorta M . A result on the computational complexity of heuristic estimates for the A* algorithm[J]. Artificial Intelligence, 1992, 34(1):47-59.

[164] Woods, J.; Radewan, C.. Kalman filtering in two dimensions. IEEE Transactions on Information Theory. 1977. Volume: 23 , Issue: 4. Page(s): 473 C 482. DOI: 10.1109/TIT.1977.1055750.

[165] Groves. P.D(2008)Principles of GNSS, Inertial and Multi-Sensor Integrated Navigation Systems, Artech House, Chapter 3, p. 55-96.

[166] Welch, G.; Bishop, G.. An Introduction to the Kalman Filter,[Online]Available at: https://www.cs.unc.edu/ welch/media/pdf/kalman_intro.pdf (Accessed: 1 May 2020)

[167] Muhammed A S , Ucuz D . Comparison of the IoT Platform Vendors, Microsoft Azure, Amazon Web Services, and Google Cloud, from Users' Perspectives[C]// 2020

8th International Symposium on Digital Forensics and Security (ISDFS). 2020. Economics & Mathematical Systems, 1994, 404(4):287-288.

[168] Mishra B , Kertesz A . The Use of MQTT in M2M and IoT Systems: A Survey[J]. IEEE Access, 2020, 8:201071-201086.

[169] Hunkeler U , Hong L T , Stanford-Clark A . MQTT-S A publish/subscribe protocol for Wireless Sensor Networks[C]// Communication Systems Software and Middleware and Workshops, 2008. COMSWARE 2008. 3rd International Conference on. IEEE, 2008.

[170] RASPBERRY PI FOUNDATION. Raspberry Pi 4 Tech Specs. Retrieved May 15, 2021 from https://www.raspberrypi.org/products/raspberry-pi-4-model-b/specifications/

[171] Omoniyi Ajoke Gbadamosi and Dayo Reuben Aremu. 2020. Design of a Modified Dijkstras Algorithm for finding alternate routes for shortest-path problems with huge costs.. In 2020 International Conference in Mathematics, Computer Engineering and Computer Science (ICMCECS). 1C6. https://doi.org/10.1109/ICMCECS47690.2020.240873

[172] Jan Kallwies, Bianca Forkel, and Hans-Joachim Wuensche. 2020. Determining and Improving the Localization Accuracy of AprilTag Detection. In 2020 IEEE International Conference on Robotics and Automation (ICRA). 8288C8294. https://doi.org/10.1109/ICRA40945.2020.9197427

[173] Kamiccolo. ROS/Patterns/Communication. Retrieved May 16, 2021 from http://wiki.ros.org/ROS/Patterns/Communication

[174] Maximilian Krogius, Acshi Haggenmiller, and Edwin Olson. 2019. Flexible Layouts for Fiducial Tags. In 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 1898C1903. https://doi.org/10.1109/IROS40897.2019.8967787

[175] TullyFoote. Topic. Retrieved May 16, 2021 from https://wiki.ros.org/Topics

[176] Rundong Yan, Sarah Dunnett, and Lisa Jackson. 2019. Maintenance Modelling of Complex Automated Guided Vehicle Systems. In 2019 Annual Reliability and Maintainability Symposium (RAMS). 1C6. https://doi.org/10.1109/RAMS.2019.8769020

[177] Z. Nie and H. Zhao, "Research on Robot Path Planning Based on Dijkstra and Ant Colony Optimization," 2019 International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS), Shanghai, China, 2019, pp. 222-226.

[178] Y. Zhang, Z. Xiao, X. Yuan, S. Li and S. Liang, "Obstacle Avoidance of Two-Wheeled Mobile Robot based on DWA Algorithm," 2019 Chinese Automation Congress (CAC), Hangzhou, China, 2019, pp. 5701-5706.