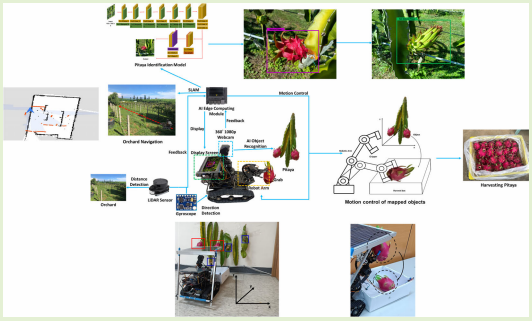
Design and Implementation of an Artificial Intelligence of Things-Based Autonomous Mobile Robot System for Pitaya Harvesting

Liang-Bi Chen, *Senior Member, IEEE*, Xiang-Rui Huang, *Student Member, IEEE*, and Wei-Han Chen

***Abstract*—Since aging in agriculture has currently become a problem in many advanced countries worldwide, in response to the growing shortage of agricultural labor, many of these countries have invested in the development of agricultural robots to solve the agricultural labor shortage problem. Automatic equipment to support agricultural harvesting can reduce the labor demand. As a result, this article proposes an artificial intelligence of things (AIoT)-based autonomous mobile robot (AMR) system for pitaya harvesting. The proposed system uses an artificial intelligence (AI) edge computing-based development board (NVIDIA Jetson Nano development board) and combines a 2-D simultaneous localization and mapping (SLAM) algorithm and an AI object recognition module. The SLAM algorithm is used for environmental detection in an unknown environment, and surrounding environment information can be detected by the sensor for map construction to facilitate robot navigation in pitaya orchards. In addition, this article describes an AI object recognition module used for pitaya recognition to facilitate pitaya harvesting. The accuracy of the proposed pitaya recognition model can reach 96.7% on the adopted NVIDIA Jetson Nano development board. A pitaya orchard is simulated in the experimental environment discussed in this article. In the simulated experimental environment, the proposed AMR system can achieve efficient pitaya harvesting and realize intelligent farming.**



***Index Terms*— Aging agricultural labor, artificial intelligence (AI) edge computing, AI object recognition, artificial intelligence of things (AIoT), autonomous mobile robot (AMR) system, farm intelligence, pitaya harvesting, simultaneous localization and mapping (SLAM).**

I. Introduction

CURRENTLY, due to the continuous increase in the global population, the demand for food will cause the world to face unprecedented pressure. According to estimates, the population will reach approximately 7.5–10.5 billion people by 2050 [1]. According to the Food and Agriculture Organization (FAO) of the United Nations (UN), it is predicted that by 2050, the global food demand will be 70% higher than that in 2006 to meet global population requirements. In contrast, due to the increasingly serious problem of extreme climate conditions attributed to the climate change trend, the world will face the problem of food supply shortages and price increases [2]. In addition, under the influence of rural population loss, aging, and the recent declining birth rate, the workforce engaged in agriculture has been drastically reduced, resulting in a considerable impact on agricultural productivity.

Today's agriculture faces the impacts of an extreme climate, aging population, lack of labor, high farming costs, and unstable yield rates. Traditional agriculture highly depends on labor and experience. These various phenomena indicate that the current agriculture industry must be urgently upgraded. The Internet of Things (IoT) system technology is a suitable opportunity for industrial upgrading; notably, farmers and

Manuscript received 25 February 2023; revised 29 March 2023; accepted 24 April 2023. Date of publication 1 May 2023; date of current version 14 June 2023. This work was supported in part by the Ministry of Science and Technology (MoST), Taiwan, under Grant MOST 110-2221-E-346-001; in part by the National Science and Technology Council (NSTC), Taiwan, under Grant NSTC 111-2622-E-346-001; and in part by the Higher Education Sprout Project, Ministry of Education (MoE), Taiwan, under Grant MOE-111G0009-1. The associate editor coordinating the review of this article and approving it for publication was Dr. Pedro Oliveira Conceição Junior. *(Corresponding author: Liang-Bi Chen.)*

The authors are with the Department of Computer Science and Information Engineering, National Penghu University of Science and Technology, Magong, Penghu 880011, Taiwan (e-mail: liangbi.chen@gmail.com; cren30475@gmail.com; e49556967@gmail.com).

Digital Object Identifier 10.1109/JSEN.2023.3270844

TABLE I
COMPARISON OF THE PROPOSED SYSTEM AND PREVIOUS WORKS ON THE TECHNICAL PROBLEMS OF ROBOTIC SYSTEMS FOR FRUIT HARVESTING

Related Works	Sensors Adoption	Robotic Arm	Hardware environment	Operating system	Recognition methods	Automatic movement methods	Model accuracy rate	Harvest success rate	Harvest
Kang *et al.* [14]	RGB-D camera (Intel Realsense D-435)	Industry 6-DoF robotic arm (UR5)	Nvidia-GTX 1080Ti or Jetson-X2	Ubuntu 16.04 LTS	Deep neural network Dasnet and 3D pose estimation algorithm	No	Above 0.8	Above 0.9	Apple
Jun *et al.* [30]	RGB-D camera (Intel Realsense D435)	6-DOF manipulator (UR3)	Nvidia Jetson TX2	Ubuntu 18.04 LTS	YOLOv3	No	Above 0.9	-	Tamato
SepúLveda *et al.* [31]	Prosilica GC2450C and Mesa SwissRanger SR4000	Two Kinova MICOTM endowed with the Kinova Gripper KG-3	Intel i7-4790	-	SVM	No	Above 0.88	91.67%	Eggplant
Yu *et al.* [11]	Laser sensor and USB camera	6-degrees-of-freedom (6DOF) arm	Intel CPU (R) with a core (TM) of i7-8700k and Nvidia 1080 GPU	-	R-YOLO	No	94.43%	84.35%	Strawberry
Li *et al.* [17]	Kinect V2	6-DOF manipulator equipped with a liftable walking platform	Intel Core i7-8700 CPU	Windows 10	Deeplabv3	No	Above 0.83	-	Litchi
Huang *et al.* [27]	Webcam	3-Axis Robotic arm	Nvidia Jetson Nano	Windows 10	YOLOv3-Tiny	No	95%	-	Pitaya
This work	Webcam, LiDAR sensor, Gyroscope sensor	3-Axis Robotic arm	Nvidia Jetson Nano	Ubuntu 18.04	YOLOv3-Tiny	**2D SLAM**	**96.7%**	**97%**	Pitaya

agricultural companies can improve agricultural efficiency by introducing the IoT system technology [2].

According to a Business Insider report, innovation in the application of IoT technology in agriculture is not new. Throughout history, agricultural revolutions have been initiated during different historical periods from handheld agricultural tools, the industrial revolution, and chemical fertilizer. Current IoT system technology can even lead agriculture to a new milestone. The IoT system technology entails a network structure that is extended and expanded based on the Internet architecture.

In addition, this technology is no longer limited to traditional computer equipment. Through information exchange and transmission with different devices, the functions and purposes of intelligent identification, tracking, monitoring, and management can be realized. Currently, IoT systems are widely used in fields, such as smart transportation, smart homes, smart health care, smart factories, smart cities, and smart campuses. In recent years, artificial intelligence (AI) technology has received global attention. Due to the increasing maturity of software and hardware technologies, the application of AI technology has achieved explosive growth. In recent years, AI has been further integrated with the IoT. Thus, the AI of things (AIoT) system technology has made the original IoT system more intelligent.

Aging in agriculture is already a problem encountered in many advanced countries worldwide. In response to growing agricultural labor force shortages, many of these countries globally have invested in the development of agricultural robots. In Taiwan, because of low birth rate, the agricultural labor force shortage due to aging is even more serious than that in other countries. In particular, agricultural population loss, community decline, community function weakening, and even aging across Taiwan's outlying islands are much more serious than those across Taiwan's main island.

Because the topsoil of Penghu in Taiwan's outlying islands is shallow with a poor water retention capacity, it provides an arid growth environment. In addition to droughts, wind and salt damage due to strong winds can occur. Cactaceae within the Hylocereus genus is resistant to drought, heat, salt, fog, and strong winds. In addition, this plant is associated with few pests and diseases and exhibits low pesticide consumption, easy care regimens, low production costs, and minimal soil and water requirements for growth. It is a crop with high development potential, so it is particularly suitable for cultivation and promotion in Penghu [3].

However, Penghu County is an island-type county and city, and due to the unique climatic conditions of Penghu, the salt and calcium contents in the soil are high. Therefore, compared to pitaya (also referred to as dragon fruit) produced on Taiwan's main island, pitaya cultivation is subject to market segmentation. Therefore, pitaya can become one of the representative fruits of Penghu County [3], and at the same time, it can promote the development of agricultural products in the Penghu region, which can promote the economic development of rural areas on Taiwan's outlying islands.

Due to a lack of labor in agriculture, there is no workforce that can manually harvest dragon fruit overproduction. The technical problem at the present stage is that there is no way to solve the problem of dragon fruit harvesting without a workforce. Thus, in this article, a robot system is developed for the autonomous harvesting of pitaya, which is a local agricultural product in Penghu, Taiwan. The leading technologies and main scientific contributions of the proposed system include pitaya object detection and the automatic movement (navigation) of a robot. Table I shows the comparison of the proposed system and previous works on the technical problems of robotic systems for fruit harvesting.

Compared to previous works, the hardware environment of this work uses a low-cost AI edge computing development module (NVIDIA Jetson Nano). Furthermore, 2-D simultaneous localization and mapping (2-D SLAM) technology is uniquely used in this work for the movement of fruit harvesting robots, which can move autonomously to perform harvesting tasks among fruit farmers. In addition, the innovation of the proposed system lies in the combination of pitaya recognition technology, robot system, and automatic movement (navigation) in NVIDIA Jetson Nano, and the accuracy rate of the model and the successful harvest rate are both above 90%.

Compared with other fruit harvesting robot research introduced in this article, the proposed system can perform the automatic movement function. In addition, it is superior to the pitaya fruit harvesting robot presented in [27] in terms of accuracy and efficiency of the dragon fruit model. Moreover, the proposed system outperforms the methods offered by Yu et al. [11], Kang et al. [14], and Sepúlveda et al. [31] in terms of harvest success rate.

Most commercial robots are mainly for harvesting fruits in greenhouses, such as strawberries and tomatoes. Furthermore, most commercial robots in greenhouses use slide rails to move. Outdoors, commercial robots are mainly controlled manually during the movement process. In addition, the pitaya orchard in Penghu, Taiwan, is a relatively harsh outdoor environment. It is easily affected by the solid northeast monsoon. The terrain changes easily, and the direction of plant growth is easily altered by external factors (terrain changes and wind). Based on environmental issues, today's commercial robots cannot address this. Therefore, this article uses the 2-D SLAM technology for the automatic movement (navigation) of the fruit harvesting robot to autonomously plan the harvesting route in an outdoor environment for navigation and obstacle avoidance, which is the main innovation of the proposed system in this article.

The main contributions and results of this article are arranged and summarized as follows.

1) This article realizes a dragon fruit recognition technology, a robot system, and automatic movement (navigation).
2) The proposed system architecture and method can provide high-efficiency harvesting for harvesting robots.
3) A data augmentation method is used for forward learning. The experimental results prove that the pitaya recognition model can be run on the NVIDIA Jetson Nano platform based on a Darknet architecture, achieving 96.7% accuracy and 4–13 frames/s.
4) The proposed system has an average harvest success rate of 97% in the experimental environment, and 3% of the error rate is caused by incorrect claw clamping when harvesting the model pitaya. In addition, experiments proved that the pitaya harvesting robot can complete the task within 1 h with 30 pitayas (including ripe and immature fruits), demonstrating execution efficiency.
5) In the experiments, we designed three routes to test the accuracy of the shortest path to verify the high accuracy of the path provided by the Dijkstra algorithm, which can accurately screen the shortest path and is suitable for the proposed system.

The remainder of this article is organized as follows. Section II reviews previous works. Section III describes the proposed system. Section IV demonstrates and analyzes the experimental results. Finally, we conclude this work in Section V.

II. Previous Works

In recent years, the concept of AIoT systems has been applied in the field of agricultural intelligence [4], [5], [6], [7], [8], [9], [10]. Elijah et al. [4] noted some of the advantages and challenges of the current agricultural IoT, introduced the way a networked ecosystem and the combination of the IoT and data analysis (DA) can achieve smart agriculture, and further proposed smart agriculture. These trends and opportunities can be divided into technological innovations, application scenarios, business opportunities, and marketability.

Grimblatt et al. [5] and Dholu and Ghodinde [6] proposed that crop growth depends on various factors, such as nutrients (NPK), soil characteristics, soil pH, soil moisture, temperature, weather, and light. To manage all the necessary information and complexity of crop growth, a system based on IoT technology that can measure, analyze, and take action is needed. The IoT-based system represents a solution for precision agriculture. In this regard, Grimblatt et al. [5] and Dholu and Ghodinde [6] proposed an IoT precision agriculture system that can process received information and transmit the most relevant information to the cloud for further statistical analysis. The system can measure the most important crop growth parameters through a set of sensors and adjust parameters through actuators when needed.

Namani and Gonen [7] proposed a smart unmanned aerial vehicle (UAV) model that can be used in farmland and is more effective in smart agriculture construction than for satellite technology. This article also examines the importance of public cloud systems in agriculture because they can promote resource sharing, cost savings, and data storage, which can be added to improve the functions of smart UAVs. Other functions include extending UAV functions by introducing a pluggable scheduling program, which functions as an intelligent analyzer in the UAV, thereby helping to reduce manual intervention in crop anomaly detection. Security and risk functions can be included. At the same time, UAVs must be equipped with recompilable software, which can help to transfer UAV base and crop data to the cloud, providing agriculture applications

and services for all users of this resource. Cloud computing-based IoT data centers should provide a more reliable virtualization platform, which can help establish more sustainable smart agriculture.

Veloo et al. [8] integrated IoT-based technologies to create an interactive farming sensing system to help farmers grow crops under optimal conditions and solve the problem of labor shortages. They created an environment in which data, such as temperature, humidity, and camera images, can be obtained and uploaded to a server. In addition, the system can incorporate different types of sensors, subsequently store the collected data in the server, and perform machine learning to estimate the condition of radish sprouts and rice more accurately.

Davcev et al. [9] proposed a power-saving and highly scalable IoT agricultural system based on the long-range wide-area network (LoRaWAN) for remote low-power data transmission from sensor nodes to cloud services. They reported that the cloud service system is highly scalable and can be used for DA.

To resolve the problem of fruit crop harvesting and lack of labor, many countries have further developed fruit-picking robot-related technologies [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], [29]. Leu et al. [10] proposed a prototype robotic harvester that can address the challenge of selective harvesting of green asparagus. The harvester can be driven along asparagus dams in the field, detect asparagus stalks, identify the stalks to be harvested, and harvest stalks without damage. The novelty of the proposed harvesting mechanism was the attained improvement in harvesting efficiency. The experimental results supported the applicability of the proposed robotic harvester.

Yu et al. [11] proposed a new type of harvesting robot for ridge cultivation and a rotational you only look once (R-YOLO) fruit position estimator, which considerably improved the positioning accuracy of picking points. The lightweight network Mobilenet-V1 [12] was used to replace the convolutional neural network (CNN) as the backbone network for feature extraction [11]. The simplified network structure greatly enhanced the operation speed. In addition, Yu et al. [11] used the rotation angle parameter to label the training set and establish the anchor point. The rotation of the bounding box of the target logic was predicted via logistic regression using rotation anchor points. The average recognition rate of this model reached 4.43%, the recall rate was 93.46%, and the harvesting success rate reached 84.35%. Kuznetsova et al. [13] and Kang et al. [14] developed an apple harvesting robot using deep learning technology combined with a deep-field camera. Kuznetsova et al. [13] employed the YOLO v3 model [15] as an object recognition module, and Kang et al. [14] used the Dasnet model [16]. These two works proposed a robot vision system that could perform fruit recognition, modeling, and environment modeling to achieve autonomous apple harvesting.

Chen et al. [16] detected and segmented fruits. Fruit modeling can locate the center and calculate the grasping posture of each fruit based on the Hough transform. Environmental modeling relied on the octal number system. Robot control obtained the path based on the calculated 3-D crop model and guided the manipulator during fruit picking. After experimental testing, the fruit recognition and modeling algorithm could accurately locate fruit and calculate the grasping posture under various conditions. The experimental results verified that compared to the method not calculating the fruit grasping posture, the proposed machine vision sensing and modeling method could effectively guide the robot arm in the separation process to improve the harvesting success rate.

Li et al. [17] developed a vision-based litchi harvesting robot system and developed a reliable RGB-D camera-based algorithm that can automatically and accurately detect and locate multiple lychees simultaneously in a large-scale environment, i.e., encompassing the fruit branches of two litchi clusters. In this study, the DeepLabv3 model, a semantic segmentation method, was used to segment RGB images into three categories: background, fruit, and twig segments. Moreover, with the use of the nonparametric density spatial clustering method for pixel determination in 3-D space, fruit branches of the same litchi cluster could be accurately detected. The experimental results indicated that the detection accuracy of litchi fruit branches was 83.33%, and the execution time of measuring a single litchi fruit branch reached 0.464 s.

Azhari et al. [18] used carbon nanotubes (CNTs) and polydimethylsiloxane (PDMS) materials for tomato harvesting. In this study, tomato hardness was measured, and tomato hardness measurement data were learned via the support vector machine (SVM) method. The experimental results demonstrated that the prediction rate reached 0.67, and immature tomatoes were clearly distinguished from mature tomatoes. Beegam et al. [19] used the RGB-D sensor of a harvesting robot to detect coffee cherries and applied the hybrid consensus and recovery block (HCRB) method for accurate coffee cherry harvesting. The experimental results indicated that the accuracy reached 93%, the precision was 97%, the recall rate reached 92%, and the $F1$-score was 93%.

Pak et al. [20] proposed a simultaneous localization and mapping (SLAM)-based algorithm, which was suitable for smart farms. This study described certain algorithms, including the Dijkstra algorithm, grid-based A* algorithm, sampling-based rapidly exploring random tree (RRT) algorithm, and sampling-based RRT* algorithm. The experimental results verified that the grid-based A* algorithm can be used in smart farms to reach the correct destination within the shortest time.

Gonzalez et al. [21] proposed a blueberry harvest management method that used the Mask R-CNN method for blueberry detection. The ResNet101, ResNet50, and MobileNetV1 architectures were experimentally tested for evaluation purposes. The ResNet50 architecture provided models with mAP values of 0.759 and 0.724, and the intersection over union (IoU) thresholds were 0.5 and 0.7.

Faisal et al. [22] proposed a ripening stage-based jujube fruit harvesting system that employed the CNN method to detect jujube fruits at seven ripening stages. The experimental results indicated that the accuracy reached 99.4%, the $F1$ score was 99.4%, the recall rate was 99.7%, and the precision reached 99.7%.

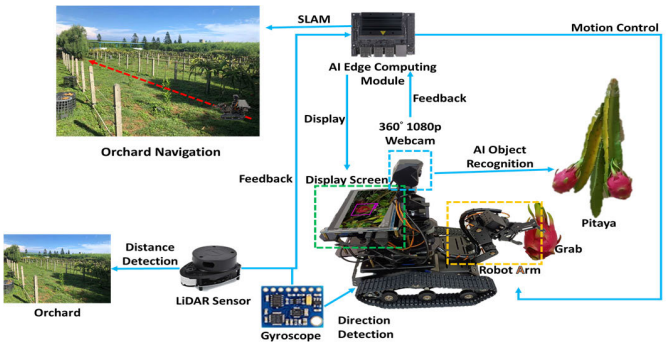


Fig. 1. System architecture of the proposed system.

The Farm Bot Taiwan User Group (FBTUG) open-source agricultural platform [23] combines the power of open source and uses technology to assist hardworking farmers, including using vision and correlation sensing to locate and distinguish individual fruits using a machine (arm) to pick fruit and a vehicle that can automatically pick fruit in a complete orchard.

In addition, the vehicle must contain a battery so that it does not need to be connected during operation. In the best case, it is possible to directly use a mobile phone to set and control the vehicle. The FBTUG combines talent and resources from various aspects and has achieved results in the collection of colored peppers [24]. Visual recognition technology can identify colored peppers and facilitate the release of associated software.

We address the above-related literature, focusing on the concept of an AIoT-based system that has been used in the field of agricultural intelligence to discuss related issues [4], [5], [6], [7], [8], [9], [10]. Many previous works have been carried out on fruit harvesting robotics to solve the problem of fruit crop harvesting and labor shortage [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], [29]. However, there is no complete automation of fruit harvesting robots at this stage. Most studies aim to improve the accuracy of fruit recognition technology or the efficiency of robot navigation functions. In a greenhouse or well-planned outdoor orchard environment, high-accuracy fruit recognition technology can achieve certain results for robots.

However, these previous studies cannot be used for complex outdoor environments because there are many external unpredictable factors in complex outdoor environments. Therefore, the robot's path cannot be planned in advance, and the accuracy of fruit recognition technology is also impacted by complex environments. Based on the problems that cannot be solved in the above literature, we design a robot that can automatically move and harvest fruits in complex outdoor environments. The realization of these two functions is the foothold of this work.

III. Proposed System

A. Hardware Configuration

Fig. 1 shows the system architecture of the proposed system, which is composed of webcam (image sensor), AI edge computing module (NVIDIA Jetson Nano), light detection and ranging (LiDAR) sensor, display screen, robot arm, and nine-axis gyroscope sensor. These components are installed and mounted on the autonomous mobile robot (AMR). The webcam inputs the dragon fruit image and sends it to the AI edge computing module through a USB communication serial port for AI object detection, with the harvesting image displayed on a small screen. Then, the fruit is grabbed or released with the claw. In addition, the LiDAR sensor detects the distance between the robot and obstacles, and the nine-axis gyroscope detects the robot's direction. The information from both is sent to the AI edge computing module for SLAM algorithm calculation to facilitate the automatic movement of the robot in the orchard function.

TABLE II
System Structure of the Proposed System

<table>
<tr><td colspan="2" rowspan="2">Layer</td><td colspan="4">Hardware Components</td></tr>
<tr><td>AI Edge Computing Module</td><td colspan="2">Robot</td><td>Gripper Based on a 3-Axis Robotic Arm</td></tr>
<tr><td rowspan="4">Software Layer</td><td>Operation Layer</td><td rowspan="3">Ubuntu 18.04</td><td colspan="2">Robot Moves</td><td>Robot Harvest</td></tr>
<tr><td>System Layer</td><td>SLAM</td><td>Location</td><td>Motion Control</td></tr>
<tr><td>Driver Layer</td><td colspan="3">ROS Module</td></tr>
<tr><td colspan="2">Image Processing</td><td>Sensor Data Preprocessing</td></tr>
<tr><td colspan="2">Transmission Layer</td><td colspan="2">USB Serial Ports</td><td>Peripherals Driver</td><td>Gyro Driver</td></tr>
<tr><td colspan="2">Sensing Layer</td><td>Webcam (Image Sensor)</td><td>LiDAR Sensor</td><td colspan="2">I/O Expansion
Gyroscope</td></tr>
</table>

Table II summarizes the system structure of the proposed system. This structure was designed and developed based on the following concepts: sensing layer, transmission layer, system layer, operation layer, and hardware layer. The sensing layer was equipped with three types of equipment, including a webcam (image sensor), a LiDAR sensor, and a nine-axis gyroscope. The transmission layer was divided into two parts. The first part involved the transmission of network camera and LiDAR sensor data to the driver layer through USB serial ports, and the second part entailed the transmission of data obtained by the nine-axis gyroscope to the driver layer through I/O expansion.

The driver layer was responsible for processing the returned information. The driver layer was divided into two parts: image processing and sensor data preprocessing. The robot operating system (ROS) module calculated the returned data of the two parts. The operation layer could perform SLAM, location, and motion control on the data calculated by the ROS module. The operation layer was divided into two parts. The first part involved robotic movement through SLAM and location processes, and the second part involved robot acquisition through motion control to perform robotic harvesting actions. The operating environment of the software layer was achieved with Ubuntu 18.04. The main hardware components were composed of an AI edge computing module (NVIDIA

TABLE III
DETAILED LIST OF ROBOT SPECIFICATIONS

Item	Description
Robotic arm	3-Axis Robotic arm
Robotic arm length	280 mm
Robot size	360 mm x 310 mm x 240 mm
Robot weight	3100 g
Claw opening and closing distance	50 mm
Gripper	2 degrees of freedom gripper
Maximum grab weight	50 g

Jetson Nano), AMR, and gripper based on a three-axis robotic arm. Table III shows the robot specifications described in this article. The claws that we use have two degrees of freedom, and claws with two degrees of freedom can be close to the folds of the pitaya skin so that the dragon fruit is stable in the claws.

To obtain the object depth information and center coordinate information, we used the web camera, LiDAR sensor, and nine-axis gyroscope. We used the LiDAR sensor to detect the distance information of the robot with regard to the obstacles and the nine-axis gyroscope to detect the direction information of the robot. In addition, we used webcams to detect objects in the images, and the detected images are inputted into a deep learning model to estimate the object coordinates.

Equations (1)–(4) illustrate the YOLO-based algorithm's estimation of b_x, b_y, b_w, and b_h of the bounding box. c_x and c_y are used to represent the grid width and height, respectively. $\sigma\ (t_x)$ and $\sigma\ (t_y)$ are used to represent the x and y offset values of the bounding box, respectively. p_w and p_h are used to indicate the bounding box width and height, respectively. b_x, b_y, b_w, and b_h of the image can be obtained by estimating the above parameters.

We can output b_x and b_y as the object's center coordinates $(x,\ y)$ and b_w and b_h as the length and width of the output object, respectively. Regarding the depth information in the image, we used a LiDAR sensor. The LiDAR sensor can detect the distance between the obstacle and the robot in real time. Therefore, compared with other related works, the depth distance of the object does not need to be measured in our work; thus, we read in the z parameter. The distance information of the LiDAR sensor is obtained when the pitaya object is read. Finally, we integrate the object coordinate information into $(z,\ x,\ y)$ values and input them into the ROS system

$$b_{\mathrm{x}} = \sigma\ (t_{\mathrm{x}}) + c_{\mathrm{x}} \tag{1}$$
$$b_y = \sigma\ (t_y) + c_y \tag{2}$$
$$b_w = p_{\mathrm{w}} e^{w} \tag{3}$$
$$b_h = p_h e^{h}. \tag{4}$$

B. Software System and Setup

As shown in Fig. 2, the software system of the robot is based on the embedded system ROS version 2 and the YOLO v3-Tiny model. The operating environment of ROS version 2 is Ubuntu version 18.04, and the software configuration is CUDA version 10.2, OpenCV 3.4, and cuDNN version 7. In addition,

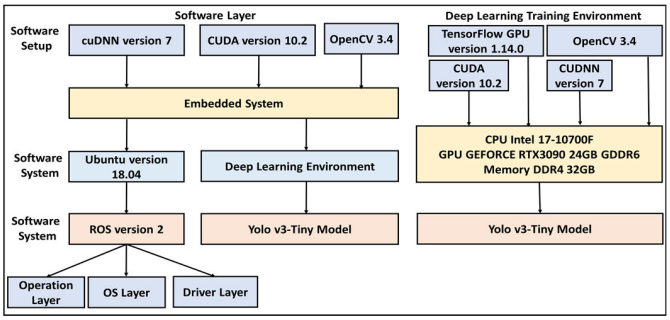


Fig. 2. Software system of the robot.

the software configuration of the deep learning training environment is the TensorFlow graphics processing unit (GPU) version 1.14.0, CUDA version 10.2, OpenCV 3.4, and cuDNN version 7. The deep learning training platform is CPU Intel I7-10700F, GPU GEFORCE RTX3090 24-GB GDDR6, and Memory DDR4 32 GB. The software system of the deep learning training environment is the YOLO v3-Tiny model. We mainly used Python for programming. CUDA version 10.2, OpenCV3.4, and cuDNN version 7 are mainly used in the environment of the YOLO v3-Tiny model. The deep learning training environment is also the same as the environment of the embedded system of the robot to facilitate training. The module is directly transplanted into the embedded system of the robot.

C. SLAM 2-D Mapping for Navigation

To verify the proposed system, as shown in Fig. 3, we built an indoor orchard with a space of 11 × 8 m to simulate the outdoor orchard environment. In the simulation environment, obstacles were set up to simulate the dragon fruit support frames in the orchard (see Fig. 4). In addition, as shown in Fig. 4, we established pitaya trees to simulate an outdoor pitaya orchard. In the process, the robot could navigate in this simulated environment and harvest pitaya. The moving route of the robot originates from dark blue as the starting route, and after harvest completion, the robot returns to the starting point from the green end route.

As shown in Fig. 5, when the robot is traveling, the SLAM algorithm is used for environment mapping and navigation. In the first step of the SLAM algorithm, the LiDAR sensor sends a LiDAR signal to the surrounding environment and reads the distance information of the LiDAR sensor. In the second step, the distance between the robot and surrounding objects is calculated using a front-end visual odometer. However, robot movement and environmental changes can inevitably cause errors. In the third step, backend optimization is performed on the previous steps to solve any errors. In the fourth step, a map is constructed based on the optimized environmental information obtained in the third step. Fig. 4 shows loop closure detection, which is intended to return to the error origin based on the environmental information obtained by the LiDAR sensor in conjunction with the third step of backend optimization. Finally, in the fifth step, the robot can perform point-to-point navigation based on the 2-D map.

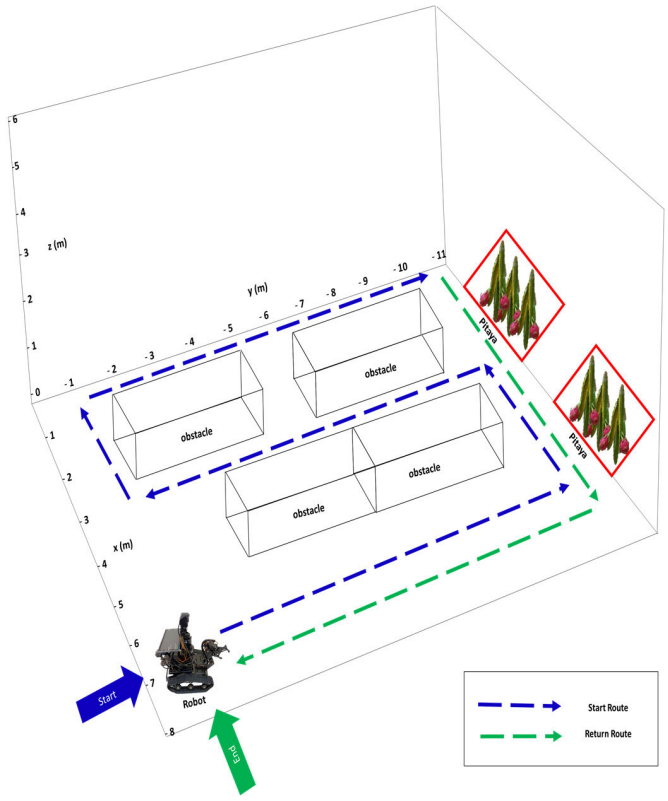


Fig. 3. Robot simulation of the execution trajectory in the orchard environment.

Fig. 4. Pitaya support stands in the orchard.

Point-to-point navigation is used for the robot's automatic movement. The overall concept is that the SLAM algorithm can use the obstacle distance detected by the LiDAR sensor to calculate the environmental information and build a map. As shown in Fig. 3, we can automatically move the robot in the environment by inputting the map's endpoint and starting point.

In the construction of the SLAM algorithm, movable/immovable obstacles can be detected and marked on the map, and we mark the movable obstacles as uncertain detection objects. Therefore, in the harvesting task, the SLAM

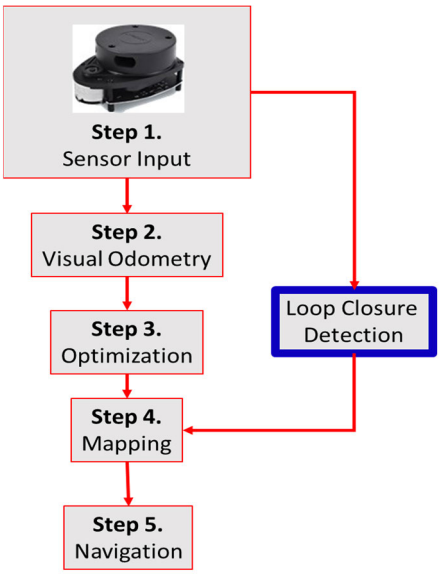


Fig. 5. Flowchart of the SLAM algorithm.

algorithm is used to establish map information in the environment, and the establishment of map information is very important for the robot's automatic movement (navigation).

During the robot's movement, the robot avoids these movable/immovable obstacles. In addition, we do not consider the problem of changing lanes because the actual outdoor pitaya orchard is fixed and well designed, and it is impossible to change the aisle in the orchard. As shown in Fig. 3, the robot built in this article is in the experimental stage, so the experimental test environment is designed first to carry out the robot harvesting task.

We use the Gmapping algorithm for autonomous robot movement (navigation) tasks. The Gmapping algorithm is an SLAM algorithm based on the Rao–Blackwellized particle filters (RBPFs) method of the 2-D LiDAR sensor to construct a 2-D grid map. Under the condition of the low scanning frequency of the LiDAR sensor, this algorithm can construct a high-precision environmental map in real time. In small scenes, the amount of calculation is small. However, in large scenes, the calculation cost is high because the calculation of building the map is large; thus, more memory is required.

The SLAM algorithm requires accurate robot positioning to match the map and uses a 2-D LiDAR sensor to return distance information and robot positioning information for map estimation. Moreover, we discuss when the robot is moving. We design the time parameter t to the sensor sensing data $s_{1:t}$, from the time parameter t to the control data $k_{1:t}$ for SLAM estimation, and the time parameter t to the robot trajectory movement data $w_{1:t}$ and map parameters m.

As shown in the joint probability distribution of (5), we designed three parameters to describe the SLAM algorithm. For the map construction of robot localization, we decomposed (5) using the RBPF method. As shown in (5), the RBPF method is used to decompose two kinds of problems, where $p\,(w_{1:t}, |k_{1:t}, s_{1:t})$ represents the robot movement trajectory data and $p\,(m|\,w_{1:t}, ,s_{1:t})$ represents the sensor and robot

movement trajectory data. Among the two problems, robot positioning is the primary problem. We used the particle filter method to estimate the robot positioning, used particles to correspond to the robot positioning and environmental information, and built a map

$$\begin{aligned} & p\left(w_{1:t}, m \mid k_{1:t}, s_{1:t}\right) \\ & \quad = p\left(w_{1:t}, \mid k_{1:t}, s_{1:t}\right) p\left(m \mid w_{1:t}, , k_{1:t}, s_{1:t}\right) \\ & \quad = p\left(w_{1:t}, \mid k_{1:t}, s_{1:t}\right) p\left(m \mid w_{1:t}, , s_{1:t}\right). \end{aligned} \tag{5}$$

We derived (6) through the Bayesian criterion. After the equation is simplified, as shown in (6) and (7), the trajectory of the robot is converted into an incremental estimate, and $p\left(w_{1:t}, \mid k_{1:t}, s_{1:t}\right)$ represents the previous moment. The robot's trajectory is also the particle swarm at the last moment.

Each particle swarm uses $p\left(w_t \mid w_{t-1}, k_t\right)$ for state propagation and predicts the trajectory corresponding to each particle, such as $p\left(s_t \mid w_t\right)$. Each predicted particle is weighted and normalized. Then, a map is constructed according to the estimated trajectory data. Each particle represents positioning information, weight information, and map position information. Then, state propagation is performed, and the positioning position information is predicted through (6) and (7). All particles store robot trajectory and map information. In addition, each particle contains a large amount of information. In the Gmapping algorithm, the method of reducing the particle swarm is used to reduce the memory. The range value controls the range sampling to reduce the number of particles that are used to predict the robot positioning distribution information

$$\begin{aligned} p\left(w_{1:t}, \mid k_{1:t}, s_{1:t}\right) &= \left(w_{1:t} \mid s_t, k_{1:t}, s_{1:t-1}\right) \\ &= np\left(\mathrm{s}_t, \mid w_{1:t}, k_{1:t}, s_{1:t-1}\right) \\ &\quad \times p\left(w_{1:t} \mid k_{1:t}, s_{1:t-1}\right) \\ &= np\left(s_t, \mid w_t\right) p\left(w_{1:t} \mid k_{1:t}, s_{1:t-1}\right) \\ &= np\left(s_t, \mid w_t\right) p\left(w_t \mid w_{1:t-1}, k_{1:t}, s_{1:t-1}\right) \end{aligned} \tag{6}$$

$$p\left(w_{t:t-1} \mid k_{1:t}, s_{1:t-1}\right) = np\left(s_t, \mid w_t\right) p\left(w_t \mid w_{t-1}, k_t\right). \tag{7}$$

As shown in (8), the range value uses the particle prediction distribution and map-matching speed to find the optimal parameters. As shown in (9), the sampling range is limited to a small area through the effective area of the nearest sensing data through the time parameter t. Regarding the particle resampling of the Gmapping algorithm, (10) is used. As the number of particle sampling increases, the repeatability of the particles will increase. Equation (10) shows the weight variance of the particles, and whether the particles are resampled is determined according to the degree of particle dispersion

$$\begin{aligned} w_t^i &= \operatorname{argmax}_{xt}\left\{p\left(s_t, \mid w_t, m\right) p\left(w_t \mid k_t, w_{t-1}^i\right)\right\} \\ &\sim p\left(w_t \mid k_t, w_{t-1}^i\right) \end{aligned} \tag{8}$$

$$\begin{aligned} p\left(w_t \mid w_{t-1}, k_t\right) &= p\left(w_t \mid w_{t-1}, k_t, s_t, m\right) \\ &= p\left(w_t \mid s_t, k_t, w_{t-1}, m\right) \\ &= np\left(s_t \mid w_t, k_t, w_{t-1}, m\right) p\left(w_t \mid w_{t-1}, k_t, m\right) \\ &= np\left(s_t \mid w_t, m\right) p\left(w_t \mid w_{t-1}, k_t\right) \\ &= np\left(s_t \mid w_t, m\right), w_t \epsilon Y^{(i)} \end{aligned} \tag{9}$$

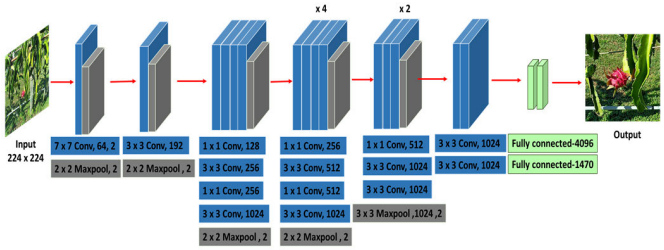


Fig. 6. YOLO architecture [26].

$$G_{\text{eff}} = \frac{1}{\sum\left(x^j\right)^2}. \tag{10}$$

We use the SLAM algorithm with regard to picking to generate maps and plan paths. Regarding path planning, the Dijkstra algorithm and the A* algorithm both use grids to distinguish between obstacles and nonobstacles and use grids to divide space and route finding, planning, and analysis. The characteristics of obstacles in the planned path will affect the performance of the algorithm. Therefore, we consider the long-term path planning of pitaya orchards as static obstacles. We use the Dijkstra algorithm to reach the target point the fastest.

Dynamic obstacles refer to unexpected dynamic obstacles. The A* algorithm has high accuracy with regard to path planning for dynamic obstacles, but the A* algorithm takes the longest time and is unsuitable for long-term path planning for static obstacles. However, Dijkstra's algorithm and the A* algorithm cannot effectively deal with negative weights. Negative weights represent the failure of route planning. Therefore, when designing the experimental environment, we should avoid clutter and complexity and conform to the real outdoor dragon fruit orchard for the simulation. We expect that, during agricultural picking, the robot can achieve a recovery rate of more than 90% in the designed experimental field, can quickly reach the target point, and can achieve a high success rate when the robot moves. This is because, during the harvesting process of the robot, it is possible that slippage or failure to harvest can occur.

D. AI Object Recognition: YOLO v3-Tiny Model

In recent years, with the advancement of AI object recognition technology, computing devices have gradually shifted from large-scale AI servers to small-scale edge computing devices. The fastest AI object recognition module is the YOLO model [25]. As shown in Fig. 6, the YOLO model is a CNN architecture. Compared to other CNN models, the YOLO model provides the following advantages: high accuracy and relatively fast model operation. To solve the linear regression problem of a single variable, the YOLO v3 model [15] uses the multiscale prediction method.

The YOLO v3 model uses the CNN architecture of Darknet-53 [15]. The CNN architecture is usually composed of a series of convolutional and pooling layers, while the Darknet-53 model is composed of 53 convolutional layers. The multilayer convolutional layer design results in complex parameter calculations. Therefore, as shown in Fig. 7, the YOLO

TABLE IV
COMPARISON OF ALL MODELS FOR PITAYA DETECTION ON JETSON NANO

Model	Backbone	Weight Size	Accuracy rate	Speed
SSD	MobileNet v2	54 MB	82.56%	180 ms
Mask-RCNN	Resnet-50	228 MB	86.43%	3 s
Fast RCNN	Resnet-50	82.4 MB	72.6%	360 ms
Faster RCNN	Resnet-50	76 MB	94.6%	230 ms
YOLO FastestV2	ShufflenetV2	10 MB	81.4%	**96 ms**
YOLO v4-tiny	CSPDarknet53-tiny	32 MB	92.1%	100 ms
The proposed method in this paper (YOLO v3-Tiny)	Darknet-50	72.6 MB	**96.7%**	120 ms

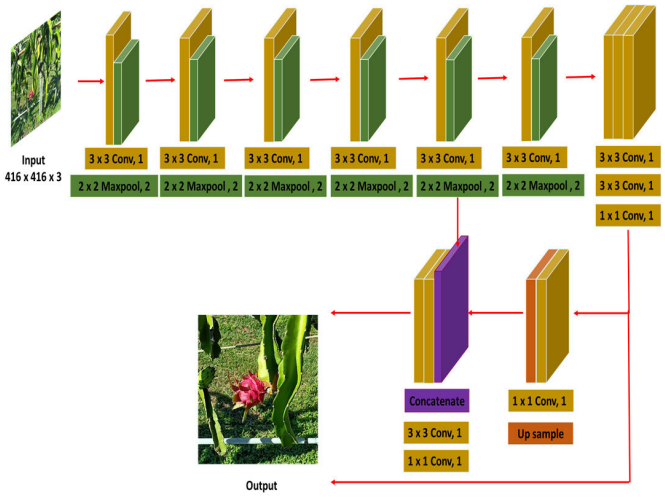


Fig. 7. YOLO v3-Tiny architecture [26].

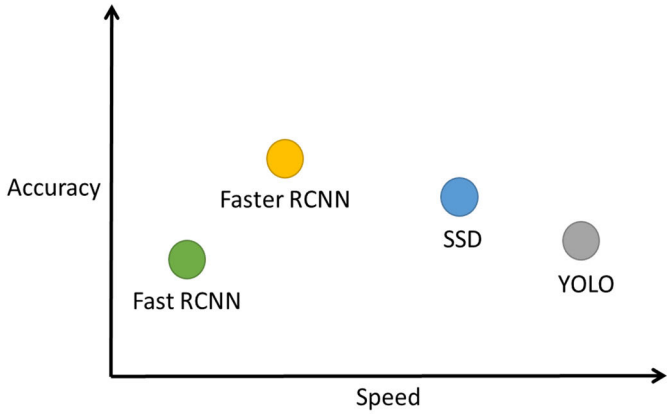


Fig. 8. Comparison with other neural network models [26].

v3-Tiny model [26] removes convolutional layers to reduce unnecessary feature extraction to accelerate model operation and uses two prediction branches, which follow the YOLO v3 version multiscale prediction method. The YOLO v3 tiny model implements end-to-end object detection. First, the input image is resized to 448 × 448; then, it is input into the CNN for processing; and finally, the prediction result is processed to obtain the detected target. The CNN uses the whole image as the input of $S \times S$ grid cells, and each grid is responsible for predicting the objects in the grid. Finally, the bounding box position and its category are output. Fig. 8 compares three classic NN models [26] in object recognition, including RCNN, YOLO, and SSD. Among them, the Faster RCNN model is accurate, and YOLO has the fastest running time. We consider that the speed of robot harvesting is the main priority, and the accuracy can be improved by model optimization. Therefore, we choose the YOLO model for development.

We need to verify that YOLO is faster and more accurate with regard to object detection. As shown in Table IV, for the problem of dragon fruit identification, we designed a comparison experiment of the model and compared it on the same platform (Jetson Nano). The platform we built is on the Jetson Nano AI edge computing development version, and the GPU is 128 Core Maxwell 472 GFLOPs (FP16). The test environment is JetPack 4.4.1, Cuda V10.0.166, OpenCV3.3, and cuDNN v7. The Darknet architecture under Jetson Nano is the most stable and lightweight version, which is more suitable for the low GPU conditions of the AI edge computing development version. Therefore, the higher end version is recommended to be built for higher GPU conditions, and the configuration version must also be compatible with Jetson Nano to run.

In Table IV, we tested the three models of the YOLO algorithm (YOLO FastestV2, YOLO v4-tiny, and YOLO v3-Tiny), which are superior to other models in terms of speed. Moreover, Faster RCNN is superior to other models in terms of accuracy. However, the method proposed in this article (using the YOLO v3-Tiny model) is superior to the model in terms of pitaya detection accuracies, and YOLO FastestV2 is superior to other models in terms of the model running speed. We chose YOLO v3-Tiny because it has the highest model accuracy and a speed of 120 ms. Although the speed of the YOLO v3-Tiny model is less than that of the YOLO FastestV2 model, it is still faster than most models. Finally, we chose the YOLOv3-Tiny version as the object recognition model in this article.

E. Dataset

In this article, a low-cost 1080P webcam (image sensor) was used as the image sensor. At this resolution, the webcam can generate 1920 × 1080 color images. In the experiment, we used a 1080P network camera for image acquisition in the Penghu pitaya orchard. A total of 1274 images were collected. The image samples were divided into mature and immature

TABLE V
SAMPLES FOR IMAGE ACQUISITION

Class	Image Sample	Percentage of Dataset	Description
Mature Pitaya		50%	To the naked eye, the color appears red.
Immature Pitaya		50%	To the naked eye, the color appears green.

TABLE VI
TRAINING AND TEST DATASETS

Class	Training dataset	Test dataset
Mature Pitaya	574	63
Immature Pitaya	574	63
Total	**1,148**	**126**
All Images	**1,274**	

fruit image samples, as shown in Table V; 50% of the data were collected in Penghu, Taiwan. During image collection, it should be noted that the range of the angle of view of the network camera covered the entire dragon fruit. The pitaya orchard is an outdoor orchard, and different daylight angles must be considered. Finally, after image collection, the dataset was divided into training and test sets at a ratio of 9:1, as summarized in Table VI.

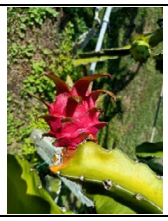
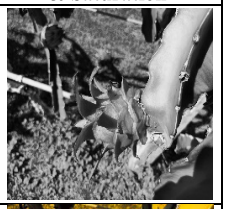
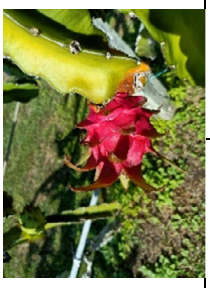
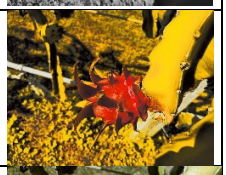
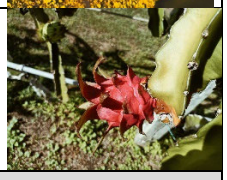
We used the manual labeling tool labelImg for Pitaya labeling, and the working time of manual labeling was 9 h. Corresponding to the YOLO v3-Tiny model, the label party was annotated in the YOLO format. The small number of 1274 images resulted in poor accuracy of the pitaya identification model. The subsequent work entailed model optimization via data enhancement and transfer learning to train the best pitaya identification model.

F. Data Augmentation and Transfer Learning

As indicated in Table VII, we used data augmentation to solve the issue of insufficient data. The original images of the training set were subjected to rotation and brightness and saturation processing operations, accounting for 4:6 of the total training set. After data enhancement, as indicated in Table VIII, the number of images increased from 1274 to 5762.

The increase in the amount of data helped to optimize the model accuracy to prevent the occurrence of overfitting. The pretraining model in this article used the CNN architecture: the Darknet-53 model. The Darknet-53 pretraining model already provided feature extraction of the basic objects. The module performed feature extraction based on the identification features of dragon fruit and finally trained the best model.

TABLE VII
EXAMPLES OF DATA AUGMENTATION

Original Image	Rotation	Brightness & Saturation
		
Original Image	**Rotation**	**Brightness & Saturation**
		
		
		
Percentage of the Training Dataset	**40%**	**60%**

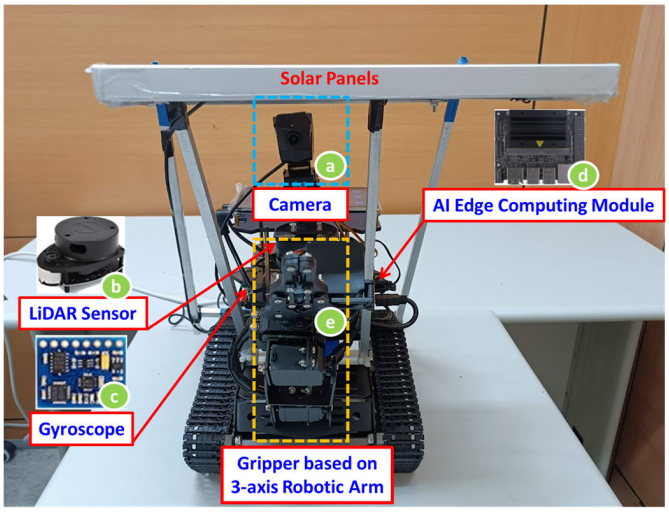


Fig. 9. Photograph of the proposed three-axis robotic arm robot. (a) Camera. (b) LiDAR sensor. (c) Gyroscope. (d) AI edge computing module. (e) Gripper based on three-axis robotic arm.

G. System Prototype

A photograph of the prototype of the proposed robot is also shown in Fig. 9. A solar panel was built above the robot

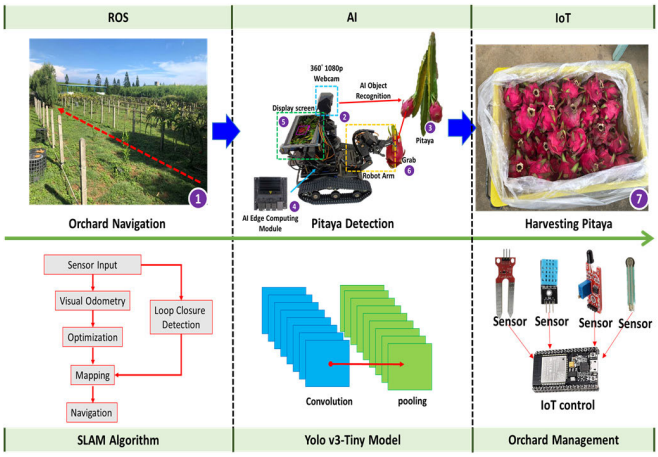


Fig. 10. Overall operation process of the proposed system.

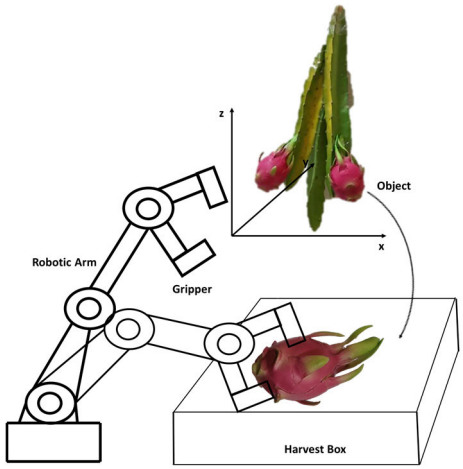


Fig. 11. Motion control of mapped objects.

Fig. 12. Pitaya location on the pitaya tree in reality.

to provide the power needed for the robot to move in the pitaya orchard. The overall hardware configuration was given as follows: the road camera [see Fig. 9(a)] was used as a sensor to detect image information for the determination of dragon fruit ripening. The LiDAR sensor [see Fig. 9(b)] was used to detect environmental information and collect information from the robot in the pitaya orchard environment. The nine-axis gyroscope [see Fig. 9(c)] was used to detect the direction of the robot in the pitaya orchard.

An AI edge computing module (Jetson Nano [see Fig. 9(d)]) with a high-performance GPU was used to perform hardware runtime computing tasks. We used an NVIDIA Jetson Nano AI edge computing development board as the main computer to control robot movement, including robotic arm movement and dragon fruit identification. The end-effector [see Fig. 9(e)] was driven by an Arduino L293D motor expansion module to control the motor signal, and through the control command sent by the NVIDIA Jetson Nano development board, the gripper could perform grasping or release actions.

The overall operation process of the robot is shown in Fig. 10. The LiDAR sensor can detect information of the surrounding environment of the robot, and the nine-axis gyroscope can detect the robot direction in the orchard navigation task [see Fig. 10(1)]. Then, the 1080P network camera [see Fig. 10(2)] transmits the image information of the pitaya in front of the robot [see Fig. 10(3)] to the NVIDIA Jetson Nano development board [see Fig. 10(4)] through the USB communication serial port, and the results can be obtained. A picture is displayed on a small screen [see Fig. 10(5)], and the fruit is then grasped or released by the claw [see Fig. 10(5)]. Finally, the harvest task is completed [see Fig. 10(7)].

Fig. 11 shows the motion control of the object corresponding to the end-effector in 3-D space. When the object is detected, the robot arm moves to the appropriate arm operating range, and the claw implements object release and grasping actions. As shown in Fig. 11, the manipulator only moves from top to bottom, and the claw is only opened and closed to perform a series of motion control actions. The positioning process of objects in 3-D space was relatively simple. We considered that the positions of pitaya fruits on the pitaya tree in reality were simplified and did not overlap (see Fig. 12). Objects can be released or grabbed by issuing a simple command.

Fig. 13 shows the flowchart of the proposed system. At the beginning of the process, object detection is started by image input, and the detected object images are input into the dragon fruit recognition model. If it is a pitaya fruit, the center coordinates (x, y, z) of the pitaya fruit object are output; if it is not a pitaya fruit object, we reperform object detection. The central coordinates of the output pitaya fruit object are calculated by the coordinate conversion tools provided by the ROS system, including axis-angles (axang), Euler angles (eul), quaternion (quat), a rotation matrix (rotm), homogeneous, transformation (tform), and translation vector (trvec). Finally, the calculated coordinate value provides the coordinates of the gripping and positioning of the gripper when the robot gripper is harvested.

IV. Experimental Results

In this article, pitaya branches and pitaya fruits were installed on an indoor wall to simulate the process of pitaya robot harvesting in an outdoor pitaya orchard. As shown in Fig. 14, the red box represents ripe pitaya fruit that can be harvested, and the blue box represents immature pitaya fruit.

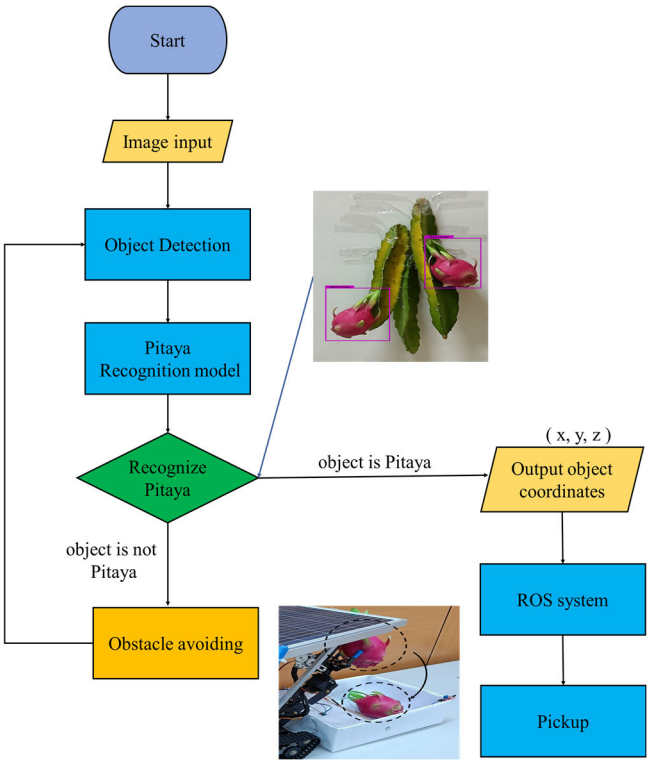


Fig. 13. Flowchart of the robot gripper harvesting a pitaya fruit.

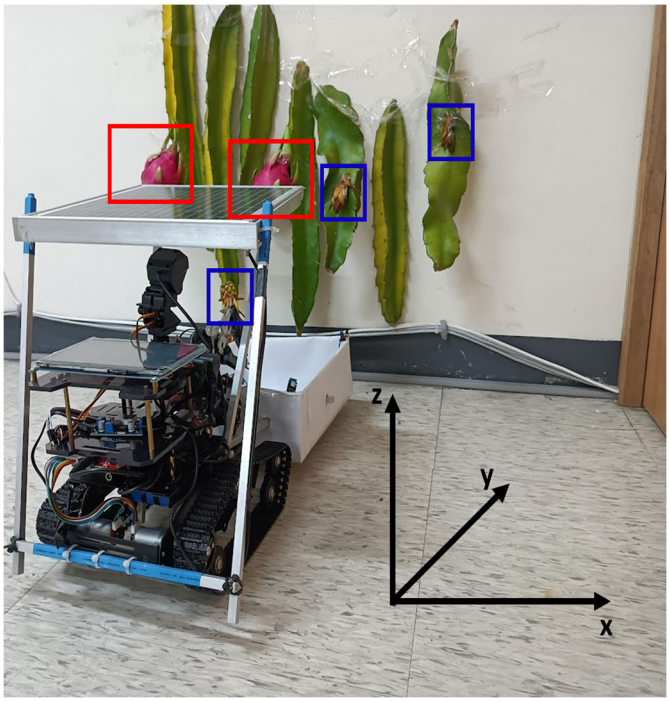
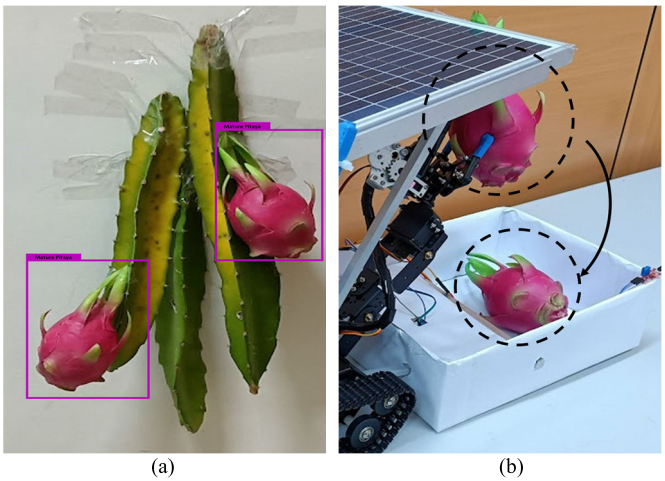


Fig. 14. Robot simulation of the conditions of pitaya harvesting.



(a) (b)

Fig. 15. Details of the robot simulation of the conditions of pitaya harvesting. (a) Identify. (b) Grab.

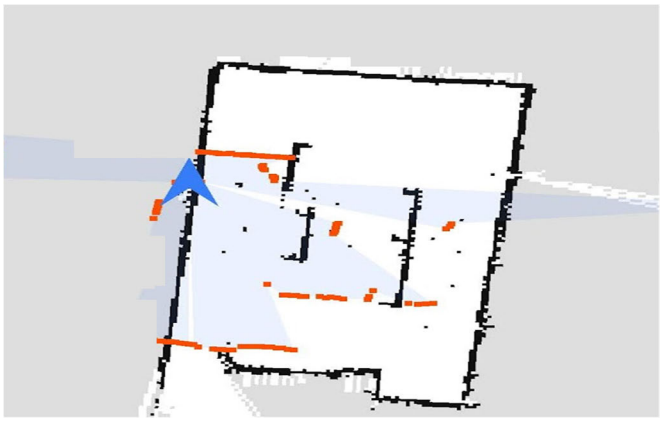


Fig. 16. SLAM 2-D mapping.

TABLE VIII
TRAINING AND TEST DATASETS

Class	Training dataset	Test dataset
Mature Pitaya	574	63
Immature Pitaya	574	63
Total	1,148	126
All Images	1,274	
Data augmentation		
Mature Pitaya	**2,593**	**288**
Immature Pitaya	**2,593**	**288**
Total	**5,186**	**576**
All Images	**5,762**	

The picked pitaya fruits were placed in the harvest box [see Fig. 15(b)].

Fig. 16 shows the 2-D SLAM construction of the robot in the indoor simulation experiment environment. The black dots are the obstacles, and the red dots are the uncertain objects of the robot. Fig. 17 shows the pitaya test images of the pitaya identification model. Fig. 17(a)–(e) shows the identification results of the pitaya identification model and Fig. 17(f)–(j) shows the original test images. The pink bounding box in the image denotes ripe pitaya fruit, and the green bounding box in the image denotes unripe dragon fruit.

To verify the accuracy of the pitaya identification model, the test image shown in Fig. 15 contains a single object image [see Fig. 17(a) and (b)] and a multiobject image [see Fig. 17(c) and (d)]. Therefore, Fig. 15 verifies that the dragon fruit identification model achieved a certain accuracy. In addition, as shown in Table IX, we evaluated the pitaya identification model to ensure the accuracy of all categories. Through verification of the confusion matrix, we calculated the accuracy of the pitaya identification model at 0.967 by the

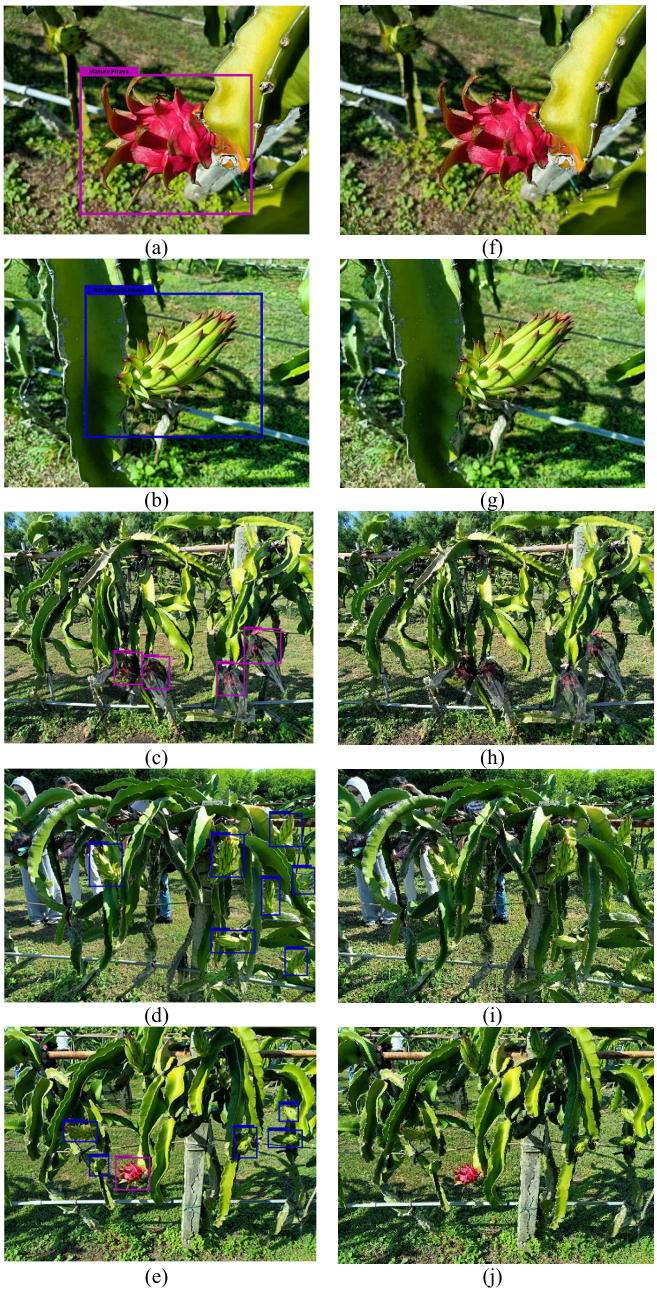


Fig. 17. Image samples for ripeness identification in pitaya images using the YOLO v3-Tiny model. (a) Identification results of the pitaya identification model for a single object (mature pitaya). (b) Identification results of the pitaya identification model for a single object (immature pitaya). (c) Identification results of the pitaya identification model for multiobject (mature pitaya). (d) Identification results of the pitaya identification model for multiobject (immature pitaya). (e) Identification results of the pitaya identification model for multiobject (mature/immature mixed pitaya). (f) Single object (mature pitaya) of original test images. (g) Single object (immature pitaya) of original test images. (h) Multiobject (mature pitaya) of original test images. (i) Multiobject (immature pitaya) of original test images. (j) Multiobject (mature/immature mixed pitaya) of original test images.

following equation:

$$\text{Accuracy} = \frac{\text{Correctly classified samples}}{\text{Total number of classified samples}}. \quad (11)$$

Huang et al. [27] proposed a dragon fruit harvesting robot. This study used a relatively small number of datasets. There were two types of dragon fruit, which were divided into mature and immature dragon fruit specimens. Based on the NVIDIA Jetson Nano development board, the model attained an accuracy of 95% and a frames/s value varying between 9 and 11. The method in this study can successfully divide mature and immature pitaya fruits.

The data augmentation method was used for transfer learning. The experimental results demonstrate that on the Darknet architecture, the pitaya identification model in this study deployed on the NVIDIA Jetson Nano development board can provide 96.7% accuracy, and the frames/s value varied between 4 and 13.

Therefore, the proposed system is superior to [27] in terms of dragon fruit identification. As shown in Table X, we planned experiments to examine the efficiency of robots performing tasks. Experiments show that the dragon fruit harvesting robot can complete the task within 1 h with 30 dragon fruits (including ripe and immature fruits), which shows the execution efficiency.

In this study, we verified whether our path planning, obstacle avoidance, and picking functions can achieve the expected goals and accuracy in the experimental field. As shown in Table X, we designed robotic harvesting in the experimental field (see Fig. 18). Table X shows that it takes 18 min for the robot to detect obstacles and avoid obstacles at the beginning of building the map, as shown in Fig. 19. After the map is created, the robot returns to its origin in 15 min, as shown in Fig. 20. Regarding the dragon fruit process, as shown in

TABLE IX
CONFUSION MATRIX OF THE PITAYA IDENTIFICATION MODEL BASED ON THE TEST DATASET

Actual		Predicted	
		Mature Pitaya	**Immature Pitaya**
	Mature Pitaya	278	10
	Immature Pitaya	9	279
Accuracy = **0.967**			

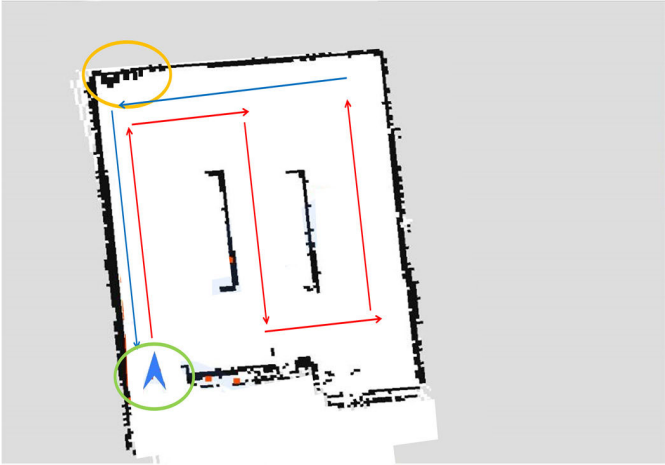


Fig. 18. Whole process of robot harvesting. The green circle is the start and end points, and the orange circles are the harvesting points.

TABLE X
EXECUTION EFFICIENCY EXPERIMENTS

Path	Time	Successful harvest	Failure harvest	Success rate	Error rate
Build map (Avoidance)	15~18 minutes	-	-	-	-
Robot return	13~15 minutes				
Harvesting process (30 pitayas)	52 minutes	27	3	90%	10%
The whole process of robot harvesting (30 pitayas)	1 hour 25 minutes				
Harvesting process (25 pitayas)	45 minutes	24	1	96%	4%
The whole process of robot harvesting (25 pitayas)	1 hour 18 minutes				
Harvesting process (20 pitayas)	36 minutes	20	0	100%	0
The whole process of robot harvesting (20 pitayas)	1 hour 9 minutes				
Harvesting process (15 pitayas)	28 minutes	10	0	100%	0
The whole process of robot harvesting (15 pitayas)	1 hour 1 minutes				
Harvesting process (10 pitayas)	17 minutes	15	0	100%	0
The whole process of robot harvesting (10 pitayas)	50 minutes				
Harvesting process (5 pitayas)	8 minutes	5	0	100%	0
The whole process of robot harvesting (5 pitayas)	41 minutes				

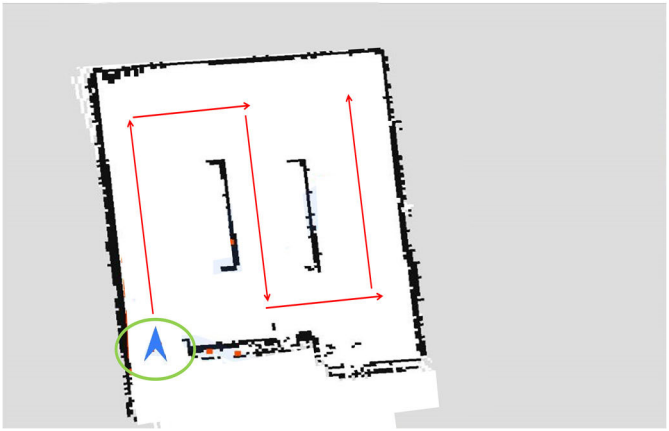


Fig. 19. Built map in the 2-D SLAM map. The green circle is the starting point.

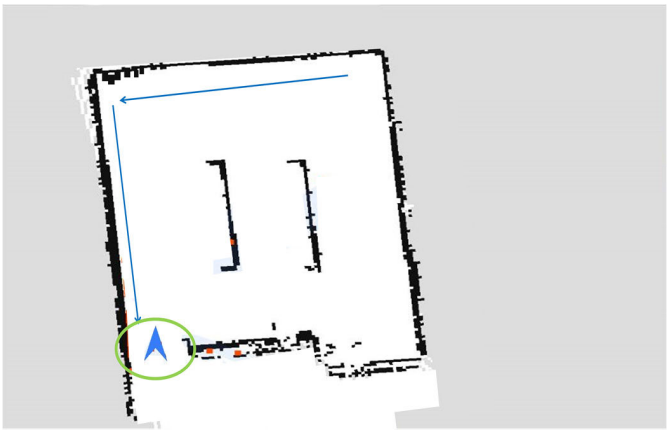


Fig. 20. Robot return in the 2-D SLAM map. The green circle is the starting point.

TABLE XI
COMPARISON OF THE TIME SPENT FOR ALL PATHS

Path	Time
a	15 minutes
b	12 minutes
c	16 minutes

Fig. 21, the harvest time for 30 dragon fruits is 52 min, and the overall time was 1 h and 25 min. The robot successfully harvested 27 dragon fruits and failed to harvest three dragon fruits, which indicates a success rate of 90%. It takes 45 min to harvest 25 dragon fruits, and the overall time takes 1 h and 18 min. Moreover, the robot successfully harvested 24 dragon fruits and failed to harvest one dragon fruit, which indicates a success rate of 96%. It takes 36 min to harvest 20 dragon fruits, and the overall time was 1 h and 9 min. The robot successfully harvested 20 dragon fruits and failed to harvest 0 dragon fruits, which indicates a success rate of 100%. It takes 28 min to harvest 15 dragon fruits, and the overall time was 1 h and 1 min. The robot successfully harvested fifteen dragon fruits and failed to harvest 0 dragon fruits. Thus, the success rate was 100%. It takes 17 min to harvest ten dragon fruits, and the total time spent was 50 min. The robot successfully harvested ten dragon fruits and failed to harvest 0 dragon fruits, which indicates a success rate of 100%. It takes 8 min to harvest five dragon fruits, and the overall time was 41 min. Finally, the robot successfully harvested five dragon fruits and unsuccessfully harvested 0 dragon fruits. Thus, the success rate was 100%.

In the experiments, the success rate of harvesting reaches more than 95%, and the time spent is longer as the number of pitayas increases. In addition, we want to verify the accuracy of choosing the shortest path of the robot harvesting dragon fruit process, as shown in Fig. 21. As shown in Fig. 22, we designed three paths to test how accurately the shortest path is chosen. In this figure, the starting point is a green circle, and the endpoint is a yellow circle. As shown in Table XI, the robot spent 12 min on Path b and less on the other routes. Path b is the shortest optimal path for the robot. As a result, the Dijkstra algorithm can accurately select the shortest path.

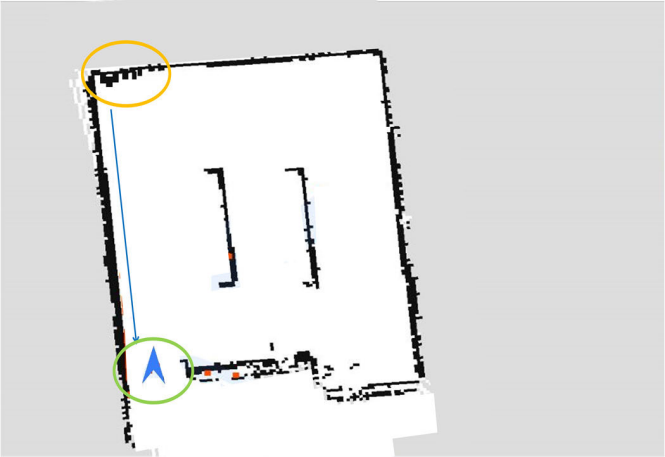


Fig. 21. Harvesting process in the 2-D SLAM map. The green circle is the start point and endpoint, and the orange circle is the harvesting point.

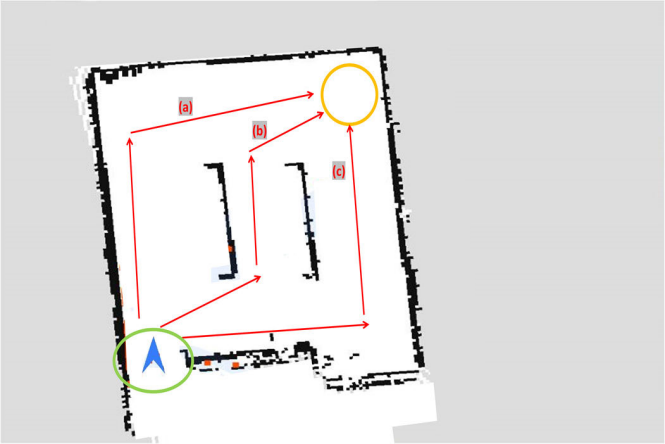


Fig. 22. Created path in the 2-D SLAM map. The green circle is the starting point, and the orange circle is the ending point. (a) Path a. (b) Path b. (c) Path c.

The proposed system has an average success rate of 97% in the experimental environment, and 3% of the error rate is caused by the incorrect clamping of the claws when harvesting the dragon fruit model. However, the high-accuracy rate is because we are using an experimental dragon fruit plastic prototype (model), and the experimental environment is a simulated outdoor dragon fruit orchard. In addition, we consider that the size and weight of the dragon fruit must be within the range of the torsion of the robot claw and develop it on the existing owned robot equipment. As a result, this experiment mainly proves that the proposed system is feasible and has high accuracy in the dragon fruit harvesting task.

V. Conclusion

In this article, an AIoT-based AMR system was proposed for pitaya picking by combining the SLAM algorithm and AI object recognition. The experimental results confirm that the harvesting robot system proposed in this article can successfully perform 2-D mapping in a quasi-experimental environment. AI object identification was achieved with a pitaya identification model based on the YOLO v3-Tiny model. The Darknet-53 model was adopted, and transfer learning was performed via a pretraining model. We used the data enhancement method to address the issue of insufficient data, including rotation, saturation, and brightness processing operations. The experimental results verify that the final dragon fruit identification model can provide 96.7% accuracy.

In addition, the main technical problem to be solved in this article is to accurately identify dragon fruit. We consider that outdoor dragon fruits may be easily affected by external sunlight. Therefore, we added rotation and saturation methods during training to improve the model's accuracy. Finally, using test images for testing, the model provides high-accuracy test results. The cost of the proposed system is approximately U.S. $2500, which cannot be applied to this equipment with the latest identification technology at this stage. In addition, the current literature on pitaya harvesting is located in the identification technology part, and we combine pitaya identification technology, robotic systems, and automatic movement (navigation) as the main scientific contribution of this article.

Therefore, the main innovation of this article lies in the dragon fruit harvesting robot that combines the high accuracy of dragon fruit identification technology with the automatic control of automatic movement (navigation). In addition, the proposed architecture and method can provide high-efficiency harvesting for harvesting robots.

In this article, a high-accuracy pitaya harvesting robot is proposed and is currently being tested in a simulated outdoor experimental field. In the experimental field, the average success rate of robot harvesting is 97%, which proves that the method proposed in this article is suitable for the dragon fruit harvesting task. For future work, with the software developed in this article, a robot with a larger size can be developed, and a mechanical claw with high torque can pick real dragon fruit and perform harvesting tasks in real dragon fruit orchards.

Acknowledgment

The authors wish to thank the organic pitaya orchard in Cimei Township, Penghu, Taiwan, for supporting the pitaya dataset, sharing orchard management experience, and providing dragon fruit for the simulation experiments.

References

[1] Council of Agriculture (CoA), Taiwan. *Smart Agriculture 4.0*. Accessed: Aug. 6, 2022. [Online]. Available: https://www.coa.gov.tw/ws.php?id=2505139

[2] *Smart Applications & Smart Cities*. Accessed: Aug. 15, 2022. [Online]. Available: https://www.digitimes.com.tw/iot/article.asp?cat=158&id=0000492590_mdt6dcup8x3lr61eutu4s

[3] *Profile of the Dragon Fruit Industry and Its Assistance Measures in Taiwan*. Accessed: Aug. 18, 2022. [Online]. Available: https://ap.fftc.org.tw/article/1294

[4] O. Elijah, T. A. Rahman, I. Orikumhi, C. Y. Leow, and M. N. Hindia, "An overview of Internet of Things (IoT) and data analytics in agriculture: Benefits and challenges," *IEEE Internet Things J.*, vol. 5, no. 5, pp. 3758–3773, Oct. 2018.

[5] V. Grimblatt, G. Ferre, F. Rivet, C. Jego, and N. Vergara, "Precision agriculture for small to medium size farmers—An IoT approach," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Sapporo, Japan, May 2019, pp. 1–5.

[6] M. Dholu and K. A. Ghodinde, "Internet of Things (IoT) for precision agriculture application," in *Proc. 2nd Int. Conf. Trends Electron. Informat. (ICOEI)*, Tirunelveli, India, May 2018, pp. 339–342.

[7] S. Namani and B. Gonen, "Smart agriculture based on IoT and cloud computing," in *Proc. 3rd Int. Conf. Inf. Comput. Technol. (ICICT)*, San Jose, CA, USA, Mar. 2020, pp. 553–556.
[8] K. Veloo, H. Kojima, S. Takata, M. Nakamura, and H. Nakajo, "Interactive cultivation system for the future IoT-based agriculture," in *Proc. 7th Int. Symp. Comput. Netw. Workshops (CANDARW)*, Nagasaki, Japan, Nov. 2019, pp. 298–304.
[9] D. Davcev, K. Mitreski, S. Trajkovic, V. Nikolovski, and N. Koteli, "IoT agriculture system based on LoRaWAN," in *Proc. 14th IEEE Int. Workshop Factory Commun. Syst. (WFCS)*, Imperia, Italy, Jun. 2018, pp. 1–4.
[10] A. Leu et al., "Robotic green asparagus selective harvesting," *IEEE/ASME Trans. Mechatronics*, vol. 22, no. 6, pp. 2401–2410, Dec. 2017.
[11] Y. Yu, K. Zhang, H. Liu, L. Yang, and D. Zhang, "Real-time visual localization of the picking points for a ridge-planting strawberry harvesting robot," *IEEE Access*, vol. 8, pp. 116556–116568, 2020.
[12] A. G. Howard et al., "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.
[13] A. Kuznetsova, T. Maleva, and V. Soloviev, "Using YOLOv3 algorithm with pre- and post-processing for apple detection in fruit-harvesting robot," *Agronomy*, vol. 10, no. 7, p. 1016, Jul. 2020.
[14] H. Kang, H. Zhou, and C. Chen, "Visual perception and modeling for autonomous apple harvesting," *IEEE Access*, vol. 8, pp. 62151–62163, 2020.
[15] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*.
[16] J. Chen et al., "DASNet: Dual attentive fully convolutional Siamese networks for change detection in high-resolution satellite images," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 1194–1206, 2021, doi: 10.1109/JSTARS.2020.3037893.
[17] J. Li, Y. Tang, X. Zou, G. Lin, and H. Wang, "Detection of fruit-bearing branches and localization of litchi clusters for vision-based harvesting robots," *IEEE Access*, vol. 8, pp. 117746–117758, 2020.
[18] S. Azhari et al., "Toward automated tomato harvesting system: Integration of haptic based piezoresistive nanocomposite and machine learning," *IEEE Sensors J.*, vol. 21, no. 24, pp. 27810–27817, Dec. 2021.
[19] K. S. Beegam, M. V. Shenoy, and N. Chaturvedi, "Hybrid consensus and recovery block-based detection of ripe coffee cherry bunches using RGB-D sensor," *IEEE Sensors J.*, vol. 22, no. 1, pp. 732–740, Jan. 2022.
[20] J. Pak, J. Kim, Y. Park, and H. I. Son, "Field evaluation of path-planning algorithms for autonomous mobile robot in smart farms," *IEEE Access*, vol. 10, pp. 60253–60266, 2022.
[21] S. Gonzalez, C. Arellano, and J. E. Tapia, "Deepblueberry: Quantification of blueberries in the wild using instance segmentation," *IEEE Access*, vol. 7, pp. 105776–105788, 2019.
[22] M. Faisal, M. Alsulaiman, M. Arafah, and M. A. Mekhtiche, "IHDS: Intelligent harvesting decision system for date fruit based on maturity stage using deep learning and computer vision," *IEEE Access*, vol. 8, pp. 167985–167997, 2020.
[23] *Solutions to Agricultural Population Shortages*. Accessed: Aug. 20, 2022. [Online]. Available: https://ctsphub.org.tw/news/info_more?id=add d7e8181cf466285dde93e35c95ac0
[24] *Machine Learning Applied to Fruit and Vegetable Identification*. Accessed: Aug. 20, 2022. [Online]. Available: https://www.slideshare.net/kobe38/ss-124269729
[25] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.
[26] P. Adarsh, P. Rathi, and M. Kumar, "YOLO v3-tiny: Object detection and recognition using one stage improved model," in *Proc. 6th Int. Conf. Adv. Comput. Commun. Syst. (ICACCS)*, Mar. 2020, pp. 687–694.
[27] X.-R. Huang, W.-H. Chen, W.-C. Hu, and L.-B. Chen, "An AI edge computing-based robotic arm automated guided vehicle system for harvesting pitaya," in *Proc. IEEE Int. Conf. Consum. Electron. (ICCE)*, Jan. 2022, pp. 1–2.
[28] C.-H. Huang, B.-W. Chen, Y.-J. Lin, and J.-X. Zheng, "Smart crop growth monitoring based on system adaptivity and edge AI," *IEEE Access*, vol. 10, pp. 64114–64125, 2022.
[29] *ThingSpeak*. Accessed: Aug. 21, 2022. [Online]. Available: https://thingspeak.com/
[30] J. Jun, J. Kim, J. Seol, J. Kim, and H. I. Son, "Towards an efficient tomato harvesting robot: 3D perception, manipulation, and end-effector," *IEEE Access*, vol. 9, pp. 17631–17640, 2021.
[31] D. Sepúlveda, R. Fernandez, E. Navas, M. Armada, and P. Gonzalez-De-Santos, "Robotic aubergine harvesting using dual-arm manipulation," *IEEE Access*, vol. 8, pp. 121889–121904, 2020.
[32] L.-B. Chen, X.-R. Huang, W.-H. Chen, W.-Y. Pai, G.-Z. Huang, and W.-C. Wang, "Design and implementation of an artificial intelligence of things-based autonomous mobile robot system for cleaning garbage," *IEEE Sensors J.*, vol. 23, no. 8, pp. 8909–8922, Apr. 2023, doi: 10.1109/JSEN.2023.3254902.

Liang-Bi Chen (Senior Member, IEEE) received the Ph.D. degree in electronic engineering from the Southern Taiwan University of Science and Technology, Tainan, Taiwan, in 2019.

He was a Visiting Researcher with the National University of Singapore, Singapore, in 2008; the University of California at Irvine, Irvine, CA, USA, from 2008 to 2009; and Waseda University, Tokyo, Japan, in 2010. In 2012, he joined BXB Electronics Company Ltd., Kaohsiung, Taiwan. In 2016, he joined the Southern Taiwan University of Science and Technology. In 2020, he joined the National Penghu University of Science and Technology, Magong, Penghu, Taiwan, as an Assistant Professor and the Director.

Dr. Chen has served as a TPC member, an IPC member, and a reviewer for many IEEE/ACM international conferences and journals. He received the 2021 Chester Sall Award (the Best Paper Award of IEEE TRANSACTIONS ON CONSUMER ELECTRONICS) from the IEEE Consumer Technology Society and the 2020–2021 Outstanding Associate Editor Award of IEEE ACCESS journal. Since 2022, he has been serving as the Chair for the IoT Technical Committee (TC), IEEE Consumer Technology Society. Since 2019, he has also been serving as an Associate Editor for IEEE ACCESS journal.

Xiang-Rui Huang (Student Member, IEEE) is currently pursuing the B.S. degree in computer science and information engineering with the National Penghu University of Science and Technology, Magong, Penghu, Taiwan.

His current research interests include deep learning, smart aquaculture, fish feeding systems design, artificial intelligence of things (AIoT) systems design, and computer vision for the IoT.

Wei-Han Chen is currently pursuing the B.S. degree in computer science and information engineering with the National Penghu University of Science and Technology, Magong, Penghu, Taiwan.

His current research interests include deep learning, smart aquaculture, and computer vision for big data analysis.