agronomy

MDPI

Article

Tomato Disease Monitoring System Using Modular Extendable Mobile Robot for Greenhouses: Automatically Reporting Locations of Diseased Tomatoes

Chen Ouyang [1], Emiko Hatsugai [1,2] and Ikuko Shimizu [1,*]

[1] Graduate School of Engineering, Tokyo University of Agriculture and Technology, 2-24-16 Naka-cho, Koganei-shi, Tokyo 184-8588, Japan
[2] Yokogawa Electric Corporation, 2-9-32 Nakacho, Musashino-shi, Tokyo 180-8750, Japan
* Correspondence: ikuko@cc.tuat.ac.jp

Abstract: Based on the appearance of tomatoes, it is possible to determine whether they are diseased. Detecting diseases early can help the yield losses of tomatoes through timely treatment. However, human visual inspection is expensive in terms of the time and labor required. This paper presents an automatic tomato disease monitoring system using modular and extendable mobile robot we developed in a greenhouse. Our system automatically monitors whether tomatoes are diseased and conveys the specific locations of diseased tomatoes to users based on the location information of the image data collected by the robot, such that users can adopt timely treatment. This system consists of two main parts: a modular, extendable mobile robot that we developed and a server that runs a tomato disease detection program. Our robot is designed to be configured and extended according to the actual height of the tomato vines, thus ensuring that the monitoring range covers most tomatoes. It runs autonomously between two rows of tomato plants and collects the image data. In addition to storing the image data of tomatoes, the data server runs a program for detecting diseases. This program contains a two-level disease detection model: a detection network for detecting diseased tomatoes and a validation network for verifying the detection results. The validation network verifies the results of the detection network by classifying the outputs of the detection network, thus reducing the false positive rate of the proposed system. Experimentally, this work focuses on the blossom-end rot of tomatoes. In this paper, YOLOv5, YOLOv7, Faster R-CNN, and RetinaNet are trained and compared on datasets divided by different conditions. YOLOv5l showed the best results on the randomly divided dataset: the mAP@0.5 reached 90.4%, and the recall reached 85.2%. Through the trained YOLOv5l, a dataset was created for training the classification networks: ResNet, MobileNet, and DenseNet. MobileNetv2 achieved the best overall performance with a 96.7% accuracy and a size of 8.8 MB. The final deployment to the system included YOLOv5l and MobileNetv2. When the confidence threshold of YOLOv5l was set to 0.1, the two-level model's false positive and false negative rates were 13.3% and 15.2%, respectively. Compared to using YOLOv5l alone, the false positive rate decreased by 5.7% and the false negative rate increased by only 2.3%. The results of the actual operation of the proposed system reveal that the system can inform the user of the locations of diseased tomatoes with a low rate of false positives and false negatives, and that it is an effective and promotable approach.

Keywords: robotics; agricultural automation; tomato; disease monitoring

Citation: Ouyang, C.; Hatsugai, E.; Shimizu, I. Tomato Disease Monitoring System Using Modular Extendable Mobile Robot for Greenhouses: Automatically Reporting Locations of Diseased Tomatoes. *Agronomy* **2022**, *12*, 3160. https://doi.org/10.3390/agronomy12123160

Academic Editor: Egidijus Šarauskis, Vilma Naujokienė and Zita Kriauciuniene

Received: 28 October 2022
Accepted: 8 December 2022
Published: 13 December 2022

1. Introduction

Tomatoes are one of the most widely grown greenhouse vegetables in the world [1]. Although the temperature and humidity in greenhouses can be controlled, it is significantly difficult to prevent tomatoes from contracting diseases. Even some diseases are more likely to occur instead [2]. If these diseases are detected and treated at the early stages, losses can be effectively reduced. However, if the diseases are detected too late, the diseased tomatoes,

including their surrounding parts, may need to be abandoned, resulting in significant yield losses. Accordingly, tomatoes need to be monitored daily to detect diseases as early as possible. However, detecting diseases by human visual inspection for each plant is time-consuming and inefficient. In other words, monitoring all the tomatoes on a daily basis is impractical. Therefore, efficiently monitoring tomatoes to detect diseases in a timely manner has emerged as an important issue. With the increasing advancement in robotics and deep learning technologies, the use of robots for automatic tomato disease monitoring has become a viable solution. Specifically, there are two reasons for this:

First, the efficiency and accuracy of computer vision tasks have been significantly improved due to the advent of deep learning. Deep Neural Networks (DNNs) can extract representations with high abstraction levels and strong representational properties. It constructs a representation of all input data through a large amount of learning and has good generalization. Some networks have outperformed humans [3]. Some recent studies have shown that deep learning techniques can be used to detect tomatoes and their diseases. As a representative of one-stage detection models, a method called "You Only Look Once" (YOLO) [4] has been used for tomato disease detection in many methods, which is to solve object detection as a regression problem and outputs the object position and category based on an end-to-end network. For example, Wang et al. [5] developed an improved YOLOv3 model to detect tomato anomalies. Similar works using the YOLO series for tomato detection or tomato disease detection are [6,7]. In addition to the YOLO series, Faster R-CNN [8], a representative of two-stage object detection models, has higher accuracy than the one-stage object detection models of the same period, although the inference speed is slower. Unlike the one-stage network, it uses RPN to generate region suggestions in the first stage, while the location and class of the object are obtained in the second stage. Natarajan et al. [9] used the Faster R-CNN to detect four kinds of diseases in tomatoes. Ref. [10] also conducted a similar study. Ref. [11] reported using Faster R-CNN can detect ten kinds of diseases associated with tomato fruits. In addition to detecting tomatoes or tomato diseases from images, many studies directly classify images to discern the type of tomatoes or the diseases. These methods employ the classification networks which extract image features and predict which category the input image belongs to. For example, Zaki et al. [12] fine-tuned a network named MobileNetv2 to detect three tomato diseases. The MobileNet families [13–15] are lightweight networks developed specifically for mobile devices. Their feature is high classification accuracy with low resource consumption. There is also a famous network that must be mentioned, ResNet [16], which first proposed a residual block that allows extremely deep networks to be constructed. An improved ResNet created by Jiang et al. [17] can identify spot blight, late blight, and yellow leaf curl on tomato leaves. Furthermore, Lu et al. [18] classify the tomato species using DenseNet201. DenseNet [19] densely connects all the previous layers to the later ones to reuse the features, thus allowing extremely deep networks to be constructed. Similar studies using classification networks have been conducted in [20–22]. In other words, it is possible to automate the detection of diseases if the image data of the target tomatoes can be obtained.

Second, the collection of tomato images can be automated using robotics. Using mobile robots to collect image data is simpler and easier to maintain than retrofitting greenhouses. In contrast to the method that involves the installation of additional tracks to add cameras to monitor target crops proposed in [23], a robot does not require any changes to the existing environment. The greenhouse environment is also well-suited for robots. Crops are planted in rows in a regular pattern and are usually separated by straight aisles or hot water pipes. This implies that the robot can run automatically between the two rows of crops with a simple linear motion strategy and obtain images or video data of the crops through installed cameras. For tomato plants, in [24], Zu et al. developed a mobile robot equipped with a 4G wireless camera to collect image data of green tomatoes. Seo et al. [25] developed a tomato monitoring robot running on hot water pipes. The use of a mobile robot to photograph and count tomatoes has also been reported [26]. In addition,

Wspanialy et al. [27] developed a mobile platform loaded with a linear actuator device to photograph the leaves of tomatoes. This device allows the camera to be moved to different vertical positions to accommodate different plant heights. A similar design is also reflected in the robot used in [28]. The robot used by Fonteijn et al. [29] in their study can raise multiple cameras to cover most of the tomatoes.

This study aims to create an automatic tomato disease monitoring system working in a greenhouse to detect diseases early. The system uses modular, extendable mobile robot we developed to collect the image data of tomatoes automatically. Our robot is designed can be modularly configured and extended according to the height of tomato plants and allows free control of the monitor range and cost. Then, a two-level disease detection model is used to detect tomato diseases based on the collected image data. Ultimately, the user can dispose of the diseased tomatoes according to their locations indicated by the system to reduce yield losses.

It should be noted that the robot we developed has been described in [30]. In [30], the robot designs and mechanisms, and results of data collection tests are described. In this paper, this paper briefly introduces the robot, and key details are added that have not yet been disclosed. In addition, the function of the robot in the system is also described. It is our hope that these details will serve as a reference for researchers who aim to rapidly develop a prototype robot or system that can be operated in greenhouses.

2. Materials and Methods

2.1. Greenhouse Environment

The tomato disease monitoring system proposed in this paper applies to greenhouses. In greenhouses, rows of crops are usually separated by a flat walkway or a surface installed with two hot water pipes. As shown in Figure 1, our greenhouse belongs to the latter category. In addition to controlling the temperature, hot water pipes serve as driving tracks for some conventional equipment. Our robots also run mainly on hot water pipes. In addition, the tomato plants we want to monitor are guided to a height of about 3 m.

Figure 1. Our greenhouse environment. Hot water pipes are installed between two rows of tomato plants. The tomato plants are guided to different heights.

2.2. System Flow and Architecture

Our tomato disease monitoring system is divided into two main parts: modular, extendable mobile robot we developed for collecting image data of tomatoes automatically and a data server for receiving and saving image data of tomatoes and detecting regions in the images which contains diseased tomatoes. The monitoring flow of the system is shown in Figure 2. First, the user manually controls the robot to drive to the head of the hot water pipes. Then, the robot transforms to the extended state and automatically drives along the hot water pipes to collect image data of tomatoes. After completing the collection, the robot changes to the standard state and returns to the starting position. At the starting position, the collected image data are sent to the data server, and the robot is on standby for the next

instruction from the user. In accordance with the user's plans, the robot can be moved to the next hot water pipes and repeat the aforementioned process or be terminated. Finally, the disease detection program running in the data server detects the diseased tomatoes from the image data automatically and informs the user of their exact locations.

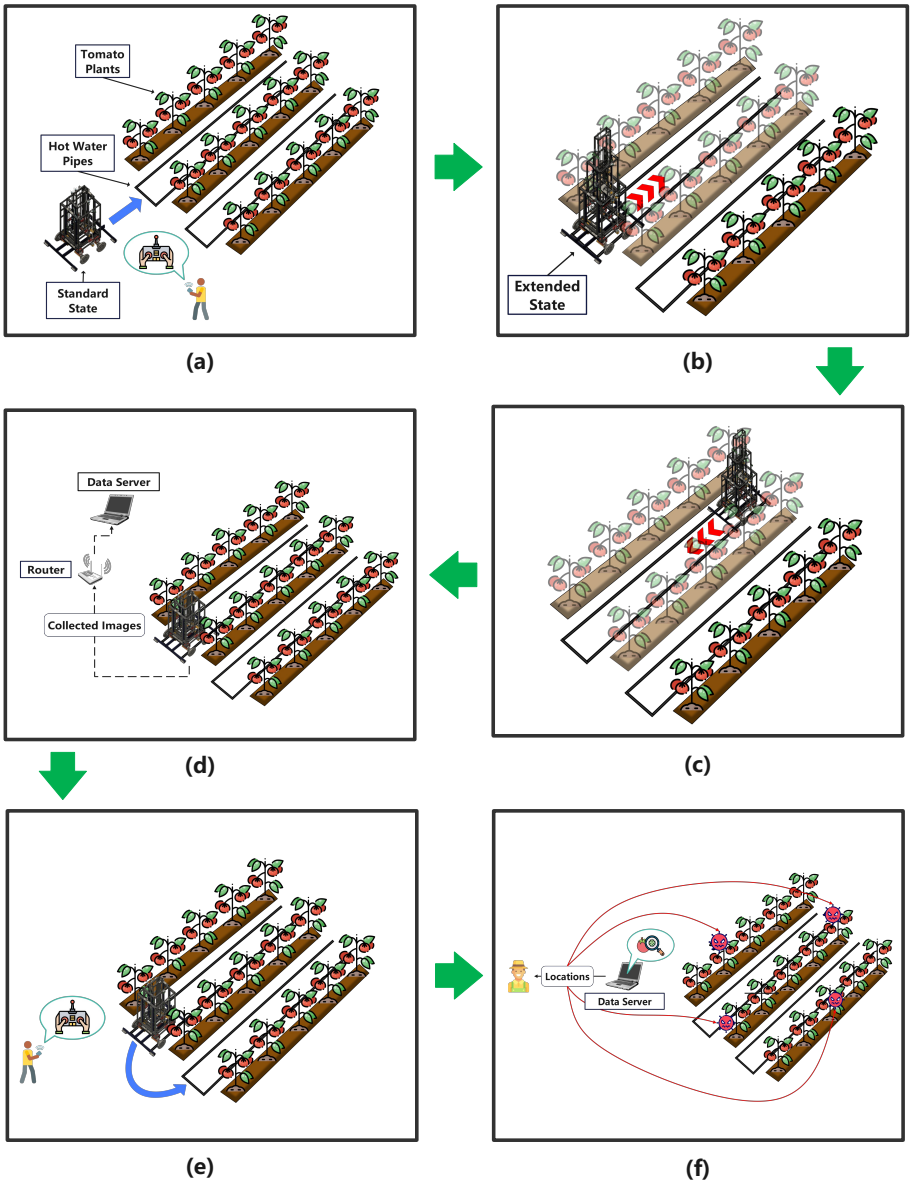


Figure 2. The workflow of the tomato disease monitoring system. (**a**) the user controls the robot to drive to the head of the hot water pipes using a remote controller; (**b**) the robot transforms to the extended state and automatically runs along the hot water pipes to collect image data of tomatoes; (**c**) when the robot reaches a predetermined position, it transforms back to the standard state and returns to the starting position; (**d**) the collected image data are sent to the server, and the robot waits for the user's instructions; (**e**) the user controls the robot to drive to the next pipes, then (**b**–**d**) are repeated; (**f**) finally, a disease detection program run by the data server informs the user of the exact locations of diseased tomatoes.

The overview of the proposed system is summarized in Figure 3. The images of tomatoes collected by the robot along with location information are sent to the data server via a local network. The data server runs a disease detection model consisting of a detection network and a validation network.

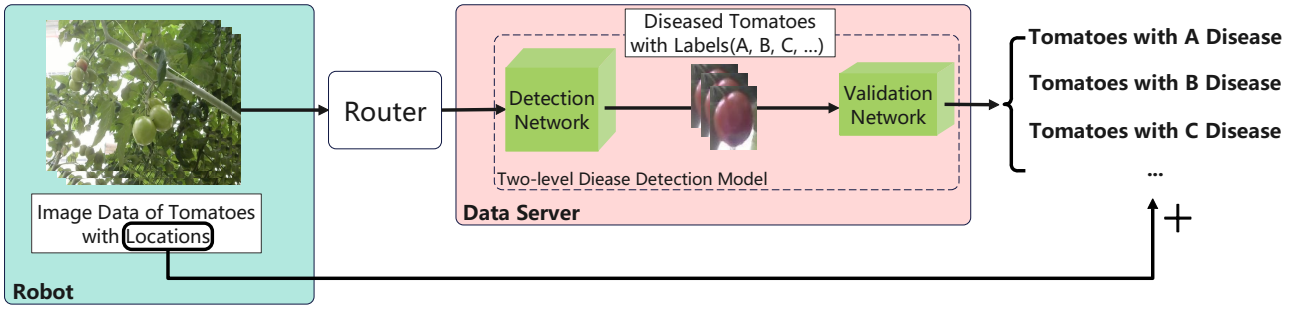


Figure 3. The overview of the tomato disease monitoring system.

The details are described below. Each image is tagged with location information based on the robot's location when it is collected. Images containing diseased tomatoes are directly fed into the detection network. The duty of the detection network is to detect the diseased tomatoes from the images, including the category labels of the diseases they belong to and their prediction boxes on the images. Based on the coordinates of the prediction boxes, the diseased tomatoes are clipped from the images.

The detection network's results are prediction values which can lead to a high false positive rate of the system by only trusting them. Thus, the clippings are input into the validation network for further verification to reduce false positives. The validation network is essentially a classification network. The classification network is chosen as the validation network because it generally achieves a high accuracy rate. The outputs of the validation network are set to the disease categories which needs to be detected. If the classification results of the validation network agree with the labels predicted by the detection network, the predicted results are considered to be accurate. If inconsistent, the clippings are put into the human visual inspection area. Since the human visual inspection area is not part of the automatic system, this paper does not discuss it. Finally, the system outputs the locations of the diseased tomatoes based on the original images' location information.

For example, suppose the detection network detects only one diseased tomato from an image, and the prediction result is 'A' disease. A clipping of the diseased tomato is obtained based on its coordinates and fed to the validation network. If the prediction of the validation network is also 'A' disease, the diseased tomato is considered 'A' disease. Then, the system outputs the location of the tomato with 'A' disease based on the location information of the original image. However, if the validation network classified the clipping as 'B' disease, in this case, the clipping is placed in the human visual inspection area for manual confirmation because the actual result is inconclusive. The two-level disease detection model is designed in this way to avoid too much wrong information and unnecessary work time.

2.3. Modular Extendable Mobile Robot

2.3.1. Introduction of the Robot We Developed

As mentioned in Section 1, the robot in this system has advantage over robots proposed in recent works. Based on the information from the papers, the details of the robots used or developed in recent works can be summarized in Table 1.

Combined with Figure 4 and Table 1, it is easy to see that, in a greenhouse with hot water pipes installed, the robots using special wheels allow them to move along the hot water pipes (Figure 4b,d–f). According to the papers' description of Figure 4a,b,d, the camera's tilt angle is adjustable to extend the shooting range. As shown in Figure 4d–f, the height of the camera can even be changed through a mechanism to increase the shooting range further. Combining these important references, our mobile robot is designed to be modular and extendable (Figure 5). The modular design allows the user to decide how many modules are needed, making the cost controllable. Extendable means that the cameras' position can be changed to extend the shooting range to cover most of the tomatoes. Detailed information about the robot we developed is described as follows.

Table 1. A comparison of robots used in recent works with our robot.

Reference	Navigating	Camera	Transformable/ Extendable	Modularization/ Cost Controllable	Shooting Range (At Least)
Figure 4a [24]	Electromagnetic Tracking	4G Wireless RGB Camera	No	No	~160 cm
Figure 4b [25]	Hot Water Pipes	ELP 4K USB Webcam	No	No	No Description
Figure 4c [26]	QR Codes	RGB Camera	No	No	No Description
Figure 4d [27]	Hot Water Pipes	SLR Camera	Yes	No	No Description
Figure 4e [28]	Hot Water Pipes	Depth Camera	Yes	No	No Description
Figure 4f [29]	Hot Water Pipes	Depth Camera	Yes	No	~300 cm
Figure 5 (Ours)	Hot Water Pipes	RGB Camera	Yes	Yes	~300 cm

(**a**) (**b**) (**c**)

(**d**) (**e**) (**f**)

Figure 4. Examples of robots working for tomato image collection in greenhouses. (**a**) A mobile robot with a 4G wireless camera [24]; (**b**) a mobile robot featuring a 4K camera [25]; (**c**) a mobile robot with multiple sensors, including cameras [26]; (**d**) a mobile platform equipped with a linear actuator. Adapted with permission from [27]. 2016, Elsevier; (**e**) a scouting robot fitted with a depth camera. Adapted with permission from [28]. 2021, IEEE Xplore; (**f**) a mobile robot fitted with four depth cameras [29]. (**a**–**c**) are fixed mechanical structures. (**d**,**e**) have some extension capability. (**f**) can cover most of the tomatoes by raising the cameras.

The robot proposed in this paper consists of a mobile chassis module and several nestable modules. As shown in Figure 6, the nestable modules are mounted on the mobile chassis module, and the smaller nestable module can be nested inside the larger nestable module. Therefore, the bottom nestable module has the largest size, and the top nestable module has the smallest size. The smaller nestable module can slide up and down within

the larger nestable module. Based on the situation, the user can decide how many nestable modules need to be used.

(**a**) (**b**)

Figure 5. Proposed modular, extendable robot. The user is free to configure the robot to cope with the height of the tomato plants and can control costs as needed. (**a**) standard state; (**b**) extended state.

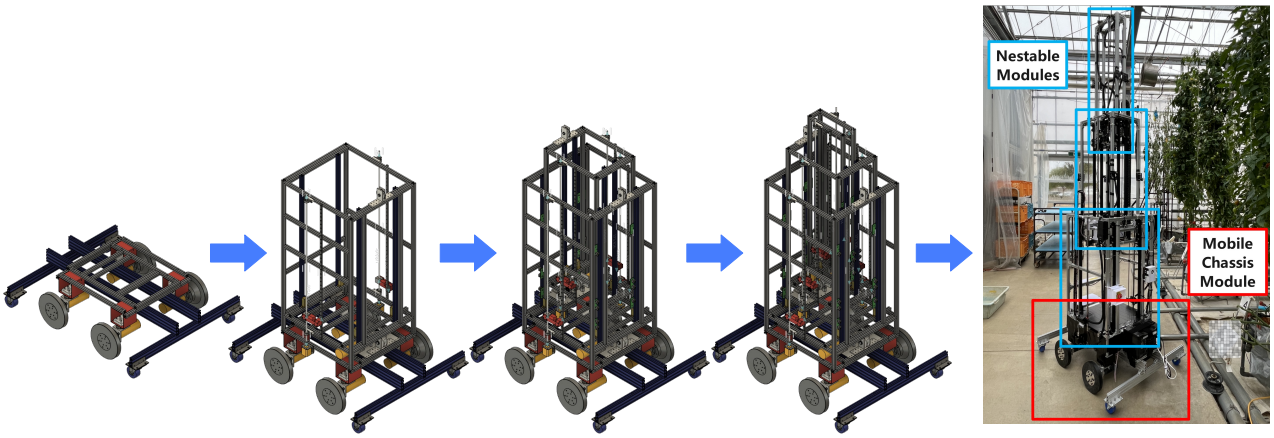


Figure 6. Configurations of the robot. In addition to the necessary mobile chassis module, users can decide how many nestable modules they need. In the case of using three nestable modules, the robot can transform to about 3 meters, which means it can photograph tomatoes guided to 3 m height.

As the core part for collecting image data, each side of the nestable module is fitted with a combination of a threaded rod, rail, camera mounting platform, and stepper motor. The camera mounting platform is 3D printed, which means it can be adjusted to adapt the camera. Fixed-focus webcams with automatic light correction are currently installed, which perform well in the greenhouse and are inexpensive (see Figure 7). The stepper motor's rotation allows the camera mounted on the mounting platform to be raised and lowered. This feature allows our robot to monitor almost all of the tomatoes.

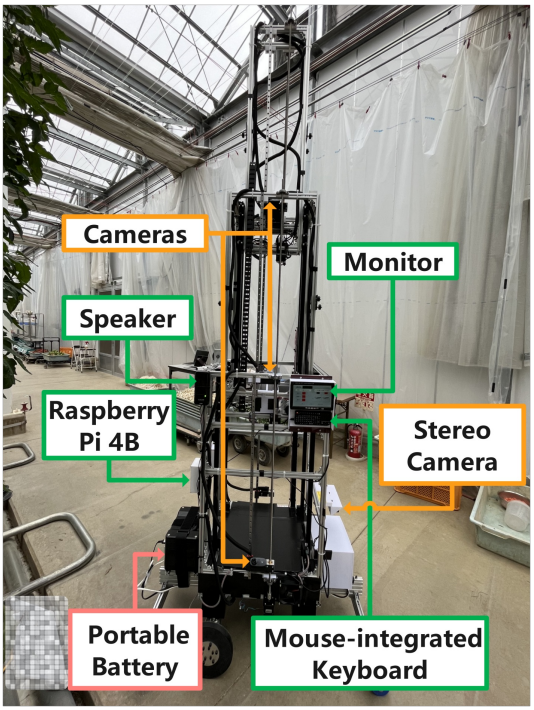


Figure 7. Devices that need to be highlighted.

It is important to note that the wheel is specially designed for driving in greenhouses. As shown in Figure 8, the road wheel with a diameter of 210 mm is used for driving on ordinary roads, and the rail wheel with a diameter of 100 mm is used for driving on hot water pipes. The hot water pipes are not placed directly flat on the floor but are supported by support devices. Due to this, the wheel is designed in a special shape: an inclination angle of about 130 degrees ensures that the robot does not derail while maintaining clearance from the pipe so it can pass the obstacle smoothly. The wall thickness is set at 2.4 mm, and the fill is set at 20% for printing to ensure the wheel's strength. Considering the wheels are 3D printed, four casters are added to share the robot's weight. In addition, a 100 mm diameter aluminum plate is added on each side for the actual installation to prevent denting.

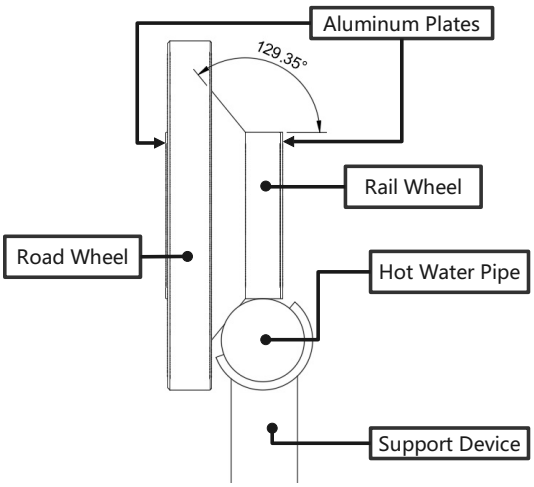


Figure 8. Cross-sectional view of the wheel and the hot water pipe. This design allows the wheel to maintain some clearance from the hot water pipe to avoid collision with the support device.

Hardware devices are deployed in every module. This paper focuses only on the devices that need to be emphasized. A stereo camera is used to estimate the robot's pose

and is mounted on the front of the robot. It is important to note that the stereo camera sometimes fails when exposed to direct sunlight, so each lens is equipped with a Neutral Density (ND) filter(1/8) to cut down the amount of light. It is very important for the user to have real-time access to the current status of a large robot during operation, which affects the user's safety. Therefore, human–machine interaction devices are essential. As shown in Figure 7, the Raspberry Pi 4B communicates with the host computer to obtain the robot's current status data and displays it through the monitor, which also displays the Graphical User Interface (GUI). Through a mouse-integrated keyboard, the user can give instructions to the robot by selecting the appropriate function. The speakers announce the current task that the robot is executing. They are mounted on the side of the bottom-most nestable module. In addition, a wireless joystick is used to control the robot as a remote controller.

The robot is powered by a 150 Ah portable battery rated at 500 W. It is strapped directly to the rear of the mobile chassis module and provides 110 V AC power outlets, as shown in Figure 7. There are several advantages of using the portable battery. First, the portability of the battery facilitates charging. Charging does not require the robot to move close to the charging socket because the battery can be removed to charge. Another advantage is that the battery can be quickly replaced, or a larger capacity battery can be used for the monitoring task, which facilitates flexible configuration. In addition, it has protection against overheating, which ensures that the robot does not risk spontaneous combustion when operating automatically.

Furthermore, since the robot runs on hot water pipes, its undercarriage is slightly warmer than the surrounding environment. It is especially true when exposed to direct sunlight. Therefore, adding a cooling fan to the side of the mobile chassis module is a must because the host computer that generates heat is also installed inside.

In the case of making and using three nestable modules, the cost of the robot is about 1 million Japanese yen. Mass production would make it cheaper. The mobile chassis module accounts for about half of the total cost since it contains critical equipment such as the main computer.

Excluding the time spent on the program and hardware adjustments, it ran in the greenhouse for 29 days. During this period, maintenance was performed every two weeks, including replacing the 3D-printed wheels, reinforcing screws, and cleaning dust. As a result, there were no failures during nearly 2 hours of daily operation, and the monitoring task was successfully completed.

2.3.2. Settings for Our Greenhouse

Depending on the greenhouse environment and the area that needs to be monitored, the settings of the robot are adjusted.

Because the tomatoes in our greenhouse are guided to a height of about 3 m, three nestable modules are used. The total weight is about 70 kg, and its height is about 3 m when extended.

Since there is currently only one row of tomato plants that need to be monitored in our greenhouse, the robot is set only to collect images from one side at a time. As shown in Figure 9, the axis perpendicular to the pipes is defined as the x-axis and the axis coinciding with pipes is defined as the y-axis. The maximum value of Y is set to 24, and the distance between the two values is set to 750 mm.

Section 2.2 describes how the robot automatically runs on the hot water pipes and collects image data. Combined with the above settings, the specific details of running are set as follows. When the robot runs at the (x_0, y_0) row, it automatically extends to the extended state at (x_0, y_0). The three cameras on the robot's right side rise simultaneously after the extension, and the interval of each rise is set to 80 mm. The total number of ascents is 9. After each rise, three images are collected. Therefore, together with the three images collected at the initial position, 30 images of tomatoes are obtained at (x_0, y_0). Thereafter, the camera returns to the initial position, and the robot automatically advances 750 mm to (x_0, y_1) and repeats the process of collecting the image data described above. Until the

robot finishes collecting images at (x_0, y_{24}), the robot shrinks to the standard state and automatically returns to (x_0, y_0). Finally, the robot sends the collected image data to the data server and enters the standby state to wait for the user's instructions. The difference in (x_1, y_0) row is that the image data are collected from the robot's left side.

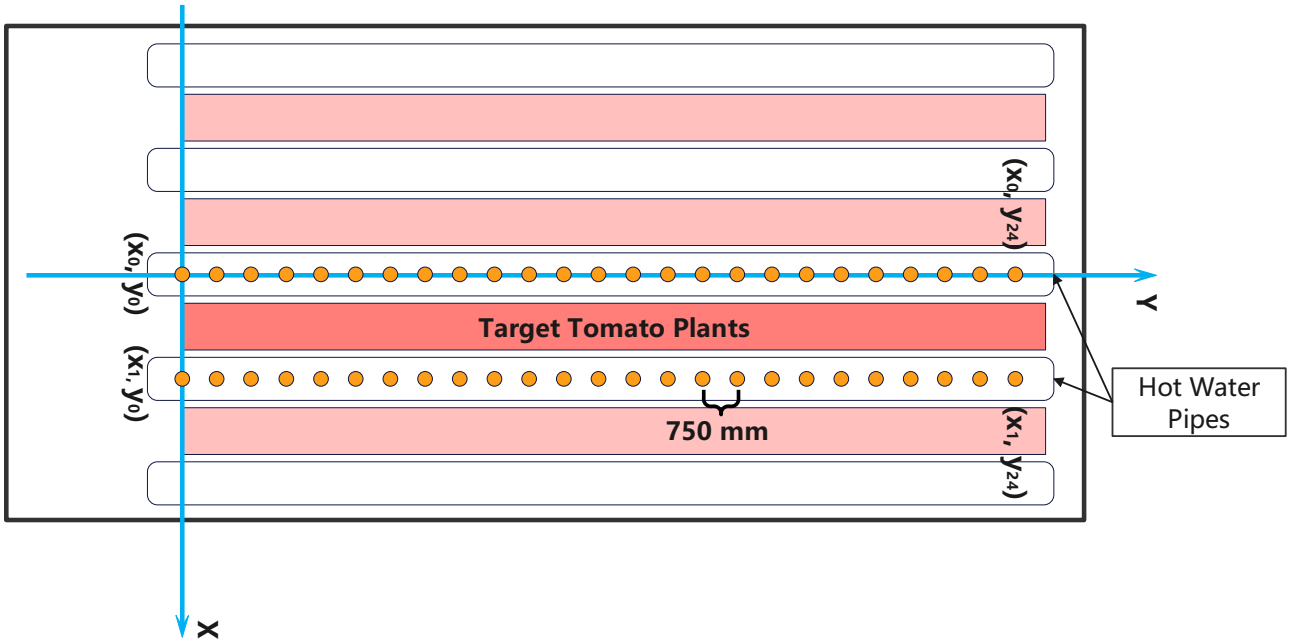


Figure 9. Schematic of our greenhouse. The robot is set to automatically stop at the orange point to collect image data of tomatoes.

Consequently, based on the values of x and y at the time of collection, which are tagged to images, the system can inform the user where the diseased tomatoes are so that the user can dispose of them in time. Furthermore, in order to make it clear to the user where the y-value corresponds, tapes are placed on the pipes to serve as a marker.

2.4. Two-Level Disease Detection Model

2.4.1. Datasets for Training

Experimentally, our system focuses on Blossom-End Rot (BER) in tomatoes, a common disease in our greenhouse. To focus on BER only, a suitable object detection network and a classification network need to be selected for the two-level disease detection model. In addition, a dataset is necessary for the selection.

There are many publicly available datasets of tomatoes on the internet, such as [31–33]. However, they can not be used in this work. There are two main reasons: first, the experimental varieties cultivated in our greenhouse are different in shape from regular tomatoes; second, the experiments in this paper focus on the BER, and the tail can not be observed in images of tomatoes taken from the front or from above. Therefore, only the image data collected by the robot are used as the dataset in this paper.

The robot ran in our greenhouse for 29 days from April 4 to June 20, according to the settings of Section 2.3.2. The cameras were set at an upward inclination of 45 degrees to obtain better images of the tail end of the tomatoes. The image data collected at a location are shown in Figure 10. The continuity of the images can be seen because the camera rises after each image is taken. Although the images appear to have some similarities, each one is taken from a different angle and has different content.

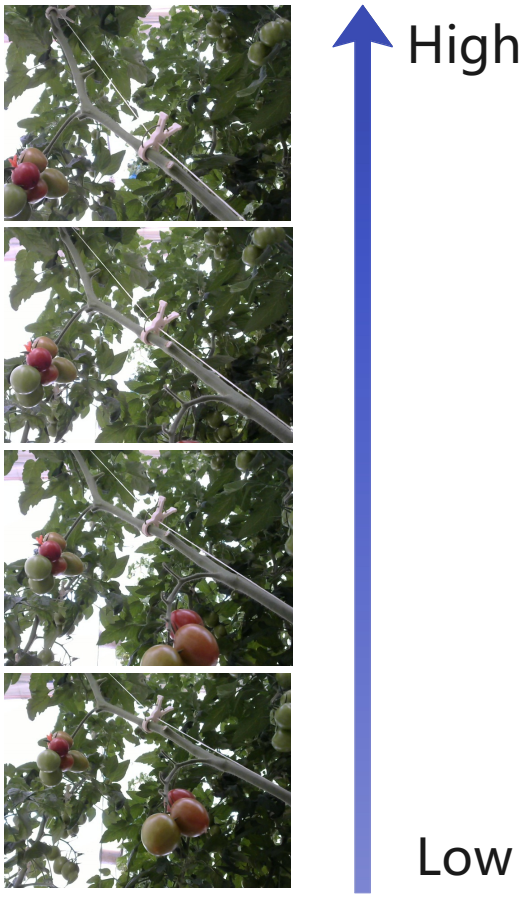


Figure 10. A part of the image data at a certain location. The order of image collection is from low to high, corresponding to low and high places in the real world.

The original size of the collected RGB images was 640x480. Makesense.ai [34] was used to label the diseased tomatoes included in the image, and the representative examples are shown in Figure 11. Then, the image size was reset to 640x640, and finally it was saved in YOLO and COCO format for training the networks. Since images with similar dates change very little, the images at certain date intervals were annotated. Thus far, the images from April 4, April 13, April 21, May 1, May 9, May 16, June 7, June 13, and June 20 were annotated. A total of 2559 images containing BER were obtained as a dataset.

Conventional training methods usually use a certain ratio to randomly divide the dataset into a training set, validation, and test set. The network is trained with the training set, observed the effect during training using the validation set, and tested with the test set to determine how well it performs after training. Unlike traditional datasets, our dataset is somewhat linked in time and location. To confirm the impact of our dataset on the model's training under different divisions and the best way to divide it, it was divided into three ways:

Dataset-1 Randomly divided the dataset into a training set, validation set, and test set in the ratio of 6:2:2.

Dataset-2 Division by dates. The data from April 4, April 13, May 1, May 9, June 7, and June 13 were used as a training set; the data from May 16 were used as a validation set; the data from April 21 and June 20 were used as a test set.

Dataset-3 Division by locations. As shown in Figure 12, the blue boxes were divided as the training set, the red boxes were divided as the validation set, and the green boxes were divided as the test set. Each location shown in the figure contains two sets of data. For example, '0' contains the images collected at (x_0, y_0) and (x_1, y_0).

Figure 11. Examples of labeled tomatoes with BER. Regardless of size or cover, all tomatoes with BER are labeled.

Specific information on the datasets produced in the three ways can be summarized in Table 2. For either dataset, 30 background-only images were added to it in the same proportions as the training, validation, and test set. In dataset-1, the training set had 1551 images, the validation set had 520 images, and the test set had 518 images. In dataset-2, the number of the training set, validation set, and test set were 1776, 436, and 377, respectively. The training set of dataset-3 was 1692, the validation set was 447, and the test set was 450.

Table 2. Details of the three differently divided datasets.

	Training Set	Validation Set	Test Set	Total
Dataset-1	1551	520	518	2589
Dataset-2	1776	436	377	2589
Dataset-3	1692	447	450	2589

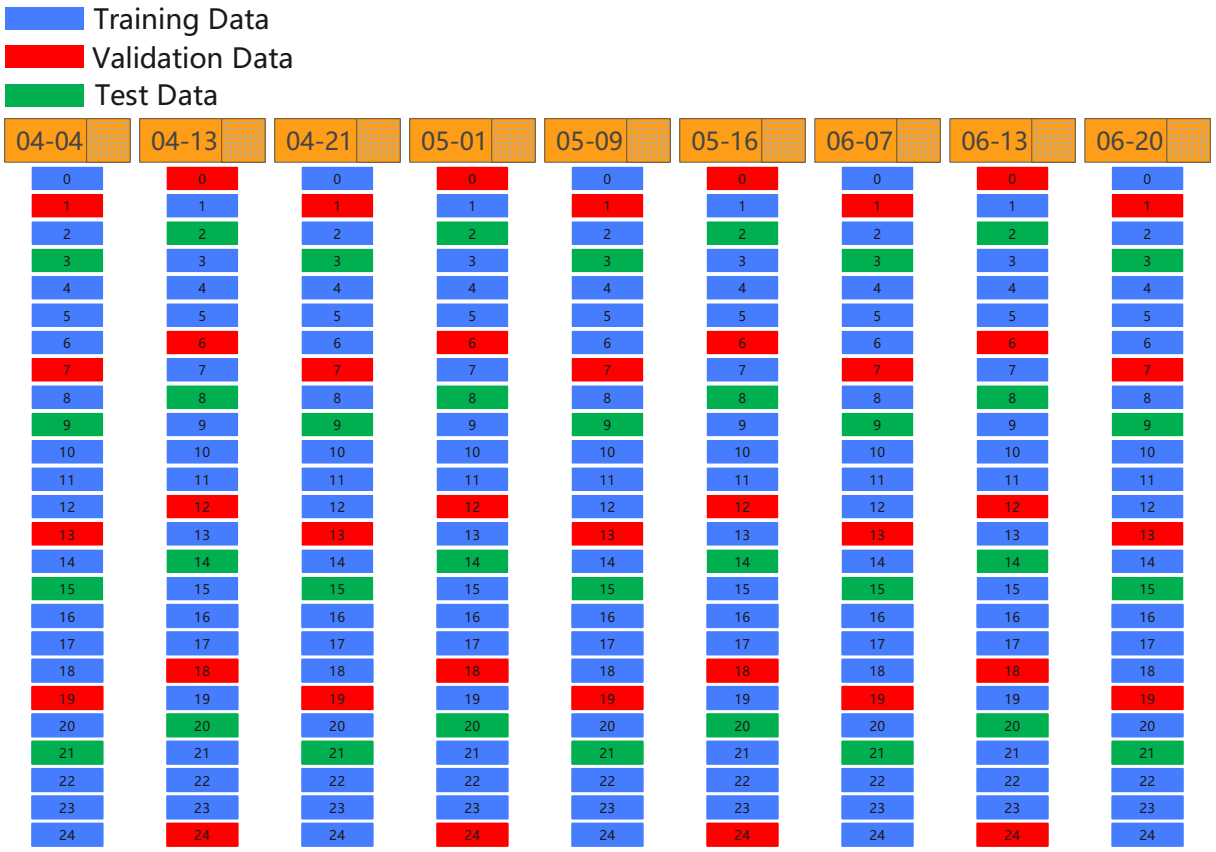


Figure 12. Division schematic diagram of dataset-3. It is divided based on location, and the locations used for the training, validation, and test set are staggered.

2.4.2. Selection of Object Detection Network

With the same philosophy as the development of our modular extendable mobile robot, the server should also be as cost-effective as possible for our proposed system to be applicable in agriculture. However, in practical deployments, the fast inference of large-size networks requires high-performance graphics cards, which can incur high costs. Therefore, considering the balance between the size and performance of object detection networks, in this system, part versions of YOLOv5 [35], YOLOv7 [36], Faster R-CNN, and RetinaNet [37] were selected and trained. For all networks, its final output was modified to one class, i.e., only tomatoes with BER were detected. All other training parameters were adopted from the official recommended default parameters.

The three datasets were used for training. Only three metrics that are practically and intuitively meaningful are listed: mean Average Precision(mAP), recall, and omission. The mAP@0.5 is an important indicator of model performance, with higher values indicating better performance. Recall and omission are opposite metrics, with recall reflecting the rate of correct objects detected from the image and omission representing the rate of false negatives. By default, the network gives the values of mAP@0.5 and recall, so the omission is calculated manually. The formula is as follows:

$$Omission = 1 - Recall \tag{1}$$

The training results are shown in Tables 3–5. Among them, the values of each metric of YOLOv5l reached the optimum when trained with dataset-1. In other words, YOLOv5l achieves the best performance on dataset-1 obtained by randomly dividing. Therefore, YOLOv5l is chosen as the first level of the two-level disease detection model.

Table 3. The training results based on dataset-1 (randomly divided in the ratio of 6 to 2 to 2).

Network	mAP@0.5 (%)	Recall (%)	Omission (%)
YOLOv5l	90.4	85.2	14.8
YOLOv5x	88.4	81.6	18.4
YOLOv7	88.3	83.8	16.2
YOLOv7-X	82.9	80.1	19.9
Faster R-CNN (R50-FPN)	80.9	74.0	26.0
Faster R-CNN (R101-FPN)	81.2	75.9	24.1
RetinaNet (R50)	78.4	72.6	27.4
RetinaNet (R101)	78.6	73.7	26.3

Table 4. The training results based on dataset-2 (division by dates).

Network	mAP@0.5 (%)	Recall (%)	Omission (%)
YOLOv5l	79.7	69.8	30.2
YOLOv5x	79.8	69.7	30.3
YOLOv7	77.7	72.4	27.6
YOLOv7-X	79.8	71.4	28.6
Faster R-CNN (R50-FPN)	74.0	73.1	26.9
Faster R-CNN (R101-FPN)	75.5	73.0	27.0
RetinaNet (R50)	73.6	72.9	27.1
RetinaNet (R101)	73.6	71.1	28.9

Table 5. The training results based on dataset-3 (division by locations).

Network	mAP@0.5 (%)	Recall (%)	Omission (%)
YOLOv5l	81.0	74.4	25.6
YOLOv5x	81.3	72.1	27.9
YOLOv7	82.4	73.7	26.3
YOLOv7-X	82.2	74.1	25.9
Faster R-CNN (R50-FPN)	81.3	72.0	26.9
Faster R-CNN (R101-FPN)	77.8	69.6	30.4
RetinaNet (R50)	77.4	71.3	28.7
RetinaNet (R101)	75.0	69.5	30.5

2.4.3. Selection of Classification Network

The best dataset and the object detection network YOLOv5l were discussed in Section 2.4.2. In order to produce the dataset used to train the classification networks, first, YOLOv5l's confidence threshold was set to 0.1, resulting in more false detections. Then, the training and validation set of dataset-1 were fed into the trained YOLOv5l, and 4473 clippings of the detected results were obtained. These clippings contain accurately detected tomatoes with BER and some incorrect detections such as backgrounds or healthy tomatoes. They were sorted into their respective folders for use as a dataset.

The number of incorrect detections was very sparse compared to the tomatoes with BER, so the number was increased through data augmentation. Then, all clippings reset

to 224 × 224 and divided into three sets: a training set, a validation set, and a test set, in the ratio 6:2:2. Finally, as shown in Table 6, the training set contained 5065 images, the validation set contained 1595 images, and the test set contained 1597 images.

Table 6. Details of the dataset for training classification networks. "BER" refers to tomatoes with BER, while "OTHER" contains images such as healthy tomatoes, backgrounds, and so on that do not belong to "BER".

	Training Set	Validation Set	Test Set
BER	2701	787	818
OTHER	2364	808	779
Total	5065	1595	1597

To select a classification network with better performance, part versions of ResNet, MobileNet, and DenseNet are trained and compared. The networks' outputs were set into two classes: "BER" and "OTHER". The training parameters used the official recommended default parameters. The results of training the networks with the created dataset are shown in Table 7. MobileNetv2 has the best overall performance. Therefore, the MobileNetv2 is chosen as the second level of the two-level disease detection model.

Table 7. The training results of each classification network.

Network	Accuracy (%)	Size (MB)
ResNet18	96.7	42.7
ResNet50	95.9	90.0
MobileNet v2	96.7	8.8
MobileNet v3 (large)	96.4	16.2
DenseNet121	97.6	27.1
DenseNet201	97.7	70.3

3. Results

3.1. Results of Two-Level Disease Detection Model

The test data used in this section are the test set of dataset-1. The test data were fed into the proposed two-level disease detection model to confirm its performance. As described in Section 2.2, the final outputs of the two-level disease detection model are the clippings of diseased tomatoes with labels. Therefore, the clippings can be reverted to prediction boxes with labels based on the labels and coordinates. Combined with the ground truth boxes of the test data, the IoU algorithm was used to estimate the number of correct detections. The formula is as follows:

$$IoU = \frac{|A \cap B|}{|A \cup B|} \tag{2}$$

where A represents the area of the ground truth box, and B represents the area of the prediction box. The IoU's value is between 0 and 1. The closer to 1, the higher the overlap between the prediction and the ground truth box. In Pascal VOC 2008 [38], the threshold value of IoU greater than 0.5 is considered a correct prediction. In our experiments, the threshold value of IoU was set to 0.65.

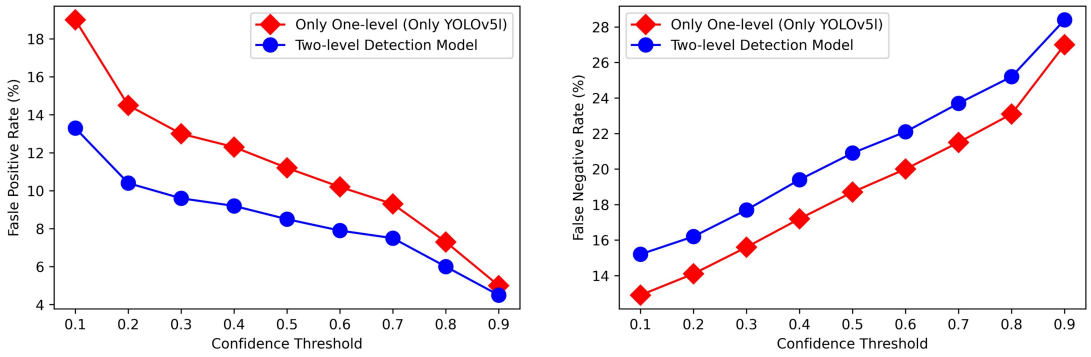
The number of false positives and false negatives can be obtained based on the number of correct predictions. Then, the rate of false positives and false negatives can be calculated. The calculations can be written as follows:

$$False\ Positive\ Rate = \frac{N_p - N_{cp>0.65}}{N_p} \tag{3}$$

$$False\ Negative\ Rate = \frac{N_{gt} - N_{cp>0.65}}{N_{gt}} \tag{4}$$

where N_p represents the total number of predictions, and $N_{cp>0.65}$ represents the total number of correct predictions when the threshold value of IoU is set greater than 0.65. N_{gt} represents the total number of ground truths.

Since the outputs of YOLOv5l of the two-level model differ at different confidence thresholds, the model was run every 0.1 from 0.1 to 0.9. Then, the false positive rate and the false negative rate were calculated. Since this paper focuses only on BER and the model is trained using images containing BER, only one value of the false positive rate and false negative rate is obtained at each confidence threshold. As an intuitive comparison, the results of only the first level of the two-level detection model were saved and calculated. The final results are shown in Figure 13.



(**a**) (**b**)

Figure 13. The results of running the two-level disease detection model with different confidence thresholds and only running the first level with different confidence thresholds. (**a**) Comparison of false positive rates; (**b**) comparison of the false negative rates.

As shown in Figure 13, whether only the first level of the model or the two-level model is used, their false positive rate decreases as the confidence threshold of the detection network increases, and the false negative rate increases as the confidence threshold increases. The false positive rate for the two-level model is 13.3% at a confidence threshold of 0.1, and the false negative rate is 15.2%.

The confidence threshold needs to be set to 0.1 for practical applications, as it allows the model to detect as many objects as possible. As shown in Figure 14, when the confidence threshold was set to 0.1, some false results were detected when using only YOLOv5l. As indicated by the arrows in Figure 14a,c, healthy tomatoes were incorrectly detected as tomatoes with BER. In addition, in Figure 14e, the background (a white clamp) was incorrectly detected as a tomato with BER. When using the two-level detection model, as shown in Figure 14b,d,f, the false detections were removed (moved to the human visual inspection area). However, not all errors were successfully classified, and the false detection indicated by the red arrow in Figure 14a was retained in Figure 14b.

In addition, the confusion matrix of the classification network in the two-level detection model when the confidence threshold was set to 0.1 is shown in Table 8. Based on the values, an accuracy of 96.4% can be calculated, which is close to the training result of MobileNetv2.

Table 8. This is a table caption. Tables should be placed in the main text near to the first time they are cited.

		Actual Values	
		BER	OTHER
Predicted Values	BER	909	16
	OTHER	21	80

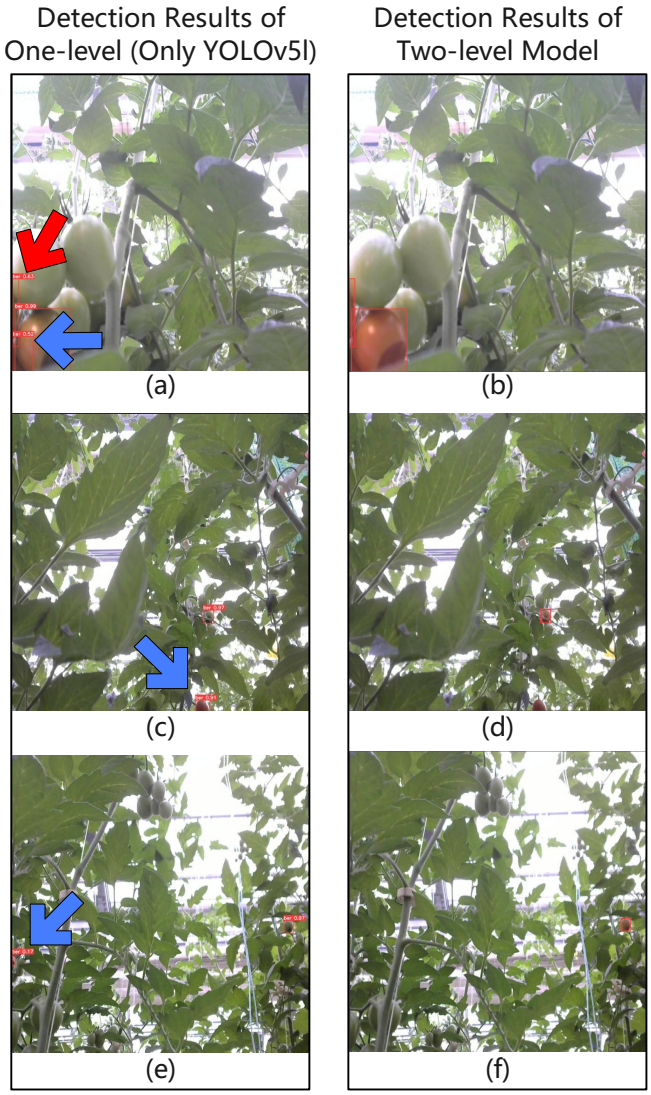
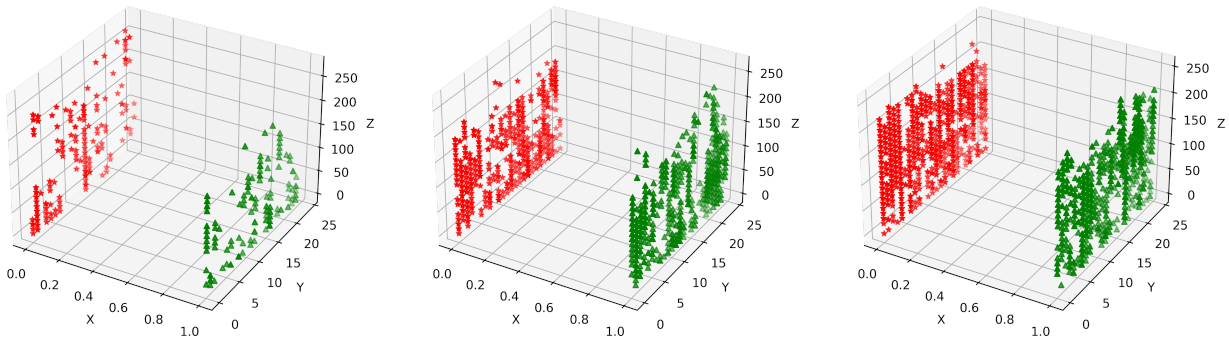


Figure 14. The detection results when the confidence threshold is set to 0.1. The two-level detection model successfully removes the false detection results pointed out by the blue arrow. The incorrect detection result pointed out by the red arrows is incorrectly retained. (**a**,**c**,**e**) Detection results of YOLOv5l; (**b**,**d**,**f**) detection results of the two-level model.

3.2. Results of System

Excluding data that had already been used, three days of data collected by the robot in the greenhouse were used to demonstrate the effectiveness of the proposed disease monitoring system. Data with concurrent dates imply that the results may be similar, so three days with large intervals were chosen: April 10, May 12, and June 10. Since there was no exact scale in the vertical direction, the z-axis coordinates were defined as follows: the images taken by the bottom-most nestable module to be 0 to 100, by the middle nestable module to be 100 to 200, and by the uppermost nestable module to be 200 to 300. The interval between each image was 10. For example, if the second image collected by the bottom-most module contains a tomato with BER, its z-axis coordinate was defined as 10. If there were multiple results in the same image, they were superimposed. The results are shown in Figure 15. A red or green dot represents where the system predicts a tomato with BER has occurred. The darker color indicates that multiple tomatoes are affected by BER at the location.



(**a**) (**b**) (**c**)

Figure 15. Distribution of tomatoes with blossom-end rot reported by the system. Red dots are the results of column x_0; green dots are the results of column x_1. Darker dots indicate multiple diseased tomatoes. (**a**) Results on April 10; (**b**) results on May 12; (**c**) results on June 10.

4. Discussion

4.1. Contribution to Automatic Disease Monitoring in Greenhouses

Tomato farming has a very large market size, but few studies have been conducted to automatically monitor tomato diseases in greenhouses. Moreover, since tomato cultivation in greenhouses is often guided to high places to make the best use of space, robots that can cover most of the tomatoes for monitoring are not only costly but also scarce. This paper introduces an automatic tomato disease monitoring system using a low-cost modular, nestable mobile robot we developed for greenhouses to overcome above-mentioned issues. The system automatically monitors tomatoes with minimal human assistance in the greenhouse for diseases and notifies users of the locations of diseased tomatoes on time. The nestable design of the robot used in the system allows it to be configured and extended according to the height of the tomato plants, such that users can choose to use several nestable modules to control costs according to their specific needs. In addition, the server used in the system does not require high performance, as the networks selected can run on lower-performance graphics cards, which further reduces costs.

The results of the proposed system conform to the actual situation. For example, there were few predicted results on April 10, shown in Figure 15 because our tomatoes started cultivation approximately at the end of March. It was higher on May 12 and June 10 because that was the time of high incidence period. In summary, this study gives a clear idea and basis to show that it is initially feasible to use a configurable mobile robot for automatically monitoring tomato diseases in a greenhouse.

4.2. Contribution to Reduction of False Positive Rate

False positives and false negatives exist in any object detection network. For false positives, in this study, we propose a two-level disease detection model that reduces false positives by further classifying the outputs of the object detection network. In an experimental comparison with different confidence thresholds, the average false positive rate of our proposed two-level disease detection model is 2.8% lower, and the average false negative rate is 2.1% higher than that using only the object detection network. It is an acceptable result because the reduced false positives can save many users' work time and improve productivity.

4.3. Comparison of Divisions and Networks

Three methods of dividing datasets were compared: random proportional division, division by dates, and division by locations. The experimental results in different object detection networks show that YOLOv5l has the best results in the first division.

None of the networks' mAP@0.5 values exceeded 80% on the data divided by dates. Our speculation is that the tomatoes with BER in the image data collected on different dates differed in the degree of lesions. This leads to poor generalization performance of the networks. Intuitively, the interlaced data should be sufficient for the networks to learn the features of diseased tomatoes from all locations. However, the networks performed much poorer on the dataset divided by locations than on the dataset divided randomly and proportionally. One possible reason is that the date interval is excessively long since users adjust or cultivate tomatoes every once in a while, which prevents the networks from learning the features of diseased tomatoes from all locations.

Additionally, in the case of the current data, the bigger, the better does not apply to classification networks. The selected MobileNetv2 has higher accuracy than ResNet50 (see Table 7) and is smaller in size.

4.4. Limits and Future Work

Notably, there are certain limitations to this study.

Firstly, since the current task only requires automatic running on the hot water pipes, our robot omits encoders and LiDAR sensors in order to develop a prototype robot quickly. However, this resulted in limiting the development of the robot into fully automatic, and it is also not convenient to control the robot remotely to move onto the hot water pipes. In addition, although a human visual inspection area is set, a system that does not require human intervention is desired.

Secondly, in this system, only tomatoes with BER are detected from the collected images, and no experiments are performed for other diseases. Hence, the system's efficiency in monitoring multiple diseases could not be confirmed.

Therefore, as future work, our plan is intended to add equipment to fully automate the tomato disease monitoring system and design it to be more efficient. For the two-level disease detection model, certain improvements to the network are optional in addition to increasing the training data. In addition to BER, there is a plan to monitor other tomato diseases to verify the system's generality.

5. Conclusions

This paper presents an automatic tomato disease monitoring system for greenhouses. This system uses a modular, extendable mobile robot to collect images of tomatoes in the greenhouse. The design of the robot we developed allows it to be configured and extended to match the height of the tomato plant. A server is used to receive image data and detect the diseased tomatoes from the images through a two-level disease detection model consisting of an object detection network and a classification network. Ultimately, the user can effectively locate the diseased tomatoes and treat them in time.

As experiments, the BER in tomatoes was focused on in this paper. For determining the optimal division of the dataset for training the object detection networks, the labeled image data were divided based on different conditions: random division by scale, division by date, and division by locations of the images. Several object detection networks were trained and compared, and the results showed that YOLOv5l was able to obtain the best performance on the randomly divided dataset. The dataset produced using YOLOv5l was used to train multiple classification networks, and the results showed that MoileNetv2 has the best balance of accuracy and size. Ultimately, YOLOv5l and MobileNetv2 were deployed to the two-level disease detection model. When the confidence threshold of YOLOv5l was set to 0.1, its false positive and false negative rates were 19.0% and 12.9%, respectively, compared to 13.3% and 15.2% for the two-level model. It means that the system using the two-level disease detection model is able to report to the user the locations of diseased tomatoes with a lower false positive rate, thus reducing the time wasted due to false positive rate and increasing the user's work efficiency.

Future research will focus on further achieving full automation. In addition to BER, monitoring other tomato diseases for the purpose of reducing yield losses is also planned.

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

Funding: This research has been made possible by funding from Yokogawa Electric Corporation.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: We are very thankful to Azuma Masashi for operating the robot in the greenhouse.

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

Abbreviations

The following abbreviations are used in this manuscript:

GUI	Graphical User Interface
ND	Neutral Density
YOLO	You Only Look Once
BER	Blossom-End Rot
LiDAR	Light Detection And Ranging
DNN	Deep Neural Network
RPN	Region Proposal Network

References

1. Viuda-Martos, M.; Sanchez-Zapata, E.; Sayas-Barberá, E.; Sendra, E.; Pérez-Álvarez, J.A.; Fernández-López, J. Tomato and Tomato Byproducts. Human Health Benefits of Lycopene and Its Application to Meat Products: A Review. *Crit. Rev. Food Sci. Nutr.* **2014**, *54*, 1032–1049. [CrossRef] [PubMed]
2. Gleason, M.L..; Edmunds, B.A. *Tomato Diseases and Disorders*; University Extension PM 1266; Iowa State University; Ames, IA, USA , 2006.
3. Alzubaidi, L.; Zhang, J.; Humaidi, A.J.; Al-Dujaili, A.; Duan, Y.; Al-Shamma, O.; Santamaría, J.; Fadhel, M.A.; Al-Amidie, M.; Farhan, L. Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *J. Big Data* **2021**, *8*, 53. [CrossRef] [PubMed]
4. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*; IEEE: Las Vegas, NV, USA, 2016; pp. 779–788. [CrossRef]
5. Wang, X.; Liu, J. Tomato Anomalies Detection in Greenhouse Scenarios Based on YOLO-Dense. *Front. Plant Sci.* **2021**, *12*, 634103. [CrossRef]
6. Lawal, M.O. Tomato detection based on modified YOLOv3 framework. *Sci. Rep.* **2021**, *11*, 1447. [CrossRef]
7. Liu, J.; Wang, X. Tomato Diseases and Pests Detection Based on Improved Yolo V3 Convolutional Neural Network. *Front. Plant Sci.* **2020**, *11*, 898. [CrossRef]
8. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef]
9. Dr. V. Anantha Natarajan, Ms. Macha Babitha, D.M.S.K. Detection of disease in tomato plant using Deep Learning Techniques. *Int. J. Mod. Agric.* **2020**, *9*, 525–540.
10. Zhang, Y.; Song, C.; Zhang, D. Deep Learning-Based Object Detection Improvement for Tomato Disease. *IEEE Access* **2020**, *8*, 56607–56614. [CrossRef]
11. Wang, Q.; Qi, F.; Sun, M.; Qu, J.; Xue, J. Identification of Tomato Disease Types and Detection of Infected Areas Based on Deep Convolutional Neural Networks and Object Detection Techniques. *Comput. Intell. Neurosci.* **2019**, *2019*, 1–15. [CrossRef]
12. Zaki, S.Z.M.; Asyraf Zulkifley, M.; Mohd Stofa, M.; Kamari, N.A.M.; Ayuni Mohamed, N. Classification of tomato leaf diseases using MobileNet v2. *IJ-AI* **2020**, *9*, 290. [CrossRef]
13. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv* **2017**, arXiv:1704.04861. https://doi.org/10.48550/ARXIV.1704.04861.
14. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520. [CrossRef]

15. Howard, A.; Sandler, M.; Chen, B.; Wang, W.; Chen, L.C.; Tan, M.; Chu, G.; Vasudevan, V.; Zhu, Y.; Pang, R.; et al. Searching for MobileNetV3. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1314–1324. [CrossRef]
16. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*; IEEE: Las Vegas, NV, USA, 2016; pp. 770–778. [CrossRef]
17. Jiang, D.; Li, F.; Yang, Y.; Yu, S. A Tomato Leaf Diseases Classification Method Based on Deep Learning. In *Proceedings of the 2020 Chinese Control And Decision Conference (CCDC)*; IEEE: Hefei, China, 2020; pp. 1446–1450. 2020.9164457. [CrossRef]
18. Lu, T.; Han, B.; Chen, L.; Yu, F.; Xue, C. A generic intelligent tomato classification system for practical applications using DenseNet-201 with transfer learning. *Sci. Rep.* **2021**, *11*, 15824. [CrossRef] [PubMed]
19. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely Connected Convolutional Networks. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 2261–2269. [CrossRef]
20. Tm, P.; Pranathi, A.; SaiAshritha, K.; Chittaragi, N.B.; Koolagudi, S.G. Tomato Leaf Disease Detection Using Convolutional Neural Networks. In *Proceedings of the 2018 Eleventh International Conference on Contemporary Computing (IC3)*; IEEE: Noida, India, 2018; pp. 1–5. [CrossRef]
21. Zhao, S.; Peng, Y.; Liu, J.; Wu, S. Tomato Leaf Disease Diagnosis Based on Improved Convolution Neural Network by Attention Module. *Agriculture* **2021**, *11*, 651. [CrossRef]
22. Abbas, A.; Jain, S.; Gour, M.; Vankudothu, S. Tomato plant disease detection using transfer learning with C-GAN synthetic images. *Comput. Electron. Agric.* **2021**, *187*, 106279. [CrossRef]
23. Ramaker, M.; Boode, A.H.; Heemskerk, C.; Fesselet, L. Accurate UAS Flight inside a Greenhouse A novel algorithm combining sparse block matching optical flow with UWB localization. In *Proceedings of the 2020 21st International Conference on Research and Education in Mechatronics (REM)*; IEEE: Cracow, Poland, 2020; pp. 1–6. [CrossRef]
24. Zu, L.; Zhao, Y.; Liu, J.; Su, F.; Zhang, Y.; Liu, P. Detection and Segmentation of Mature Green Tomatoes Based on Mask R-CNN with Automatic Image Acquisition Approach. *Sensors* **2021**, *21*, 7842. [CrossRef] [PubMed]
25. Seo, D.; Cho, B.H.; Kim, K.C. Development of Monitoring Robot System for Tomato Fruits in Hydroponic Greenhouses. *Agronomy* **2021**, *11*, 2211. [CrossRef]
26. Ge, Y.; Lin, S.; Zhang, Y.; Li, Z.; Cheng, H.; Dong, J.; Shao, S.; Zhang, J.; Qi, X.; Wu, Z. Tracking and Counting of Tomato at Different Growth Period Using an Improving YOLO-Deepsort Network for Inspection Robot. *Machines* **2022**, *10*, 489. [CrossRef]
27. Wspanialy, P.; Moussa, M. Early powdery mildew detection system for application in greenhouse automation. *Comput. Electron. Agric.* **2016**, *127*, 487–494. [CrossRef]
28. Zhou, X.; Wang, P.; Dai, G.; Yan, J.; Yang, Z. Tomato Fruit Maturity Detection Method Based on YOLOV4 and Statistical Color Model. In *Proceedings of the 2021 IEEE 11th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*; IEEE: Jiaxing, China, 2021; pp. 904–908. [CrossRef]
29. Fonteijn, H.; Afonso, M.; Lensink, D.; Mooij, M.; Faber, N.; Vroegop, A.; Polder, G.; Wehrens, R. Automatic Phenotyping of Tomatoes in Production Greenhouses Using Robotics and Computer Vision: From Theory to Practice. *Agronomy* **2021**, *11*, 1599. [CrossRef]
30. Ouyang, C.; Hatsugai, E.; Shimizu, I. A Novel Modular, Extendable Mobile Robot for Image Data Collection Task in a Greenhouse. In *Proceedings of the 2022 7th International Conference on Advanced Robotics and Mechatronics (ICARM)*; IEEE: Guilin, China, 2022. [CrossRef]
31. kaggle. Tomato Detection. Available online: https://www.kaggle.com/datasets/andrewmvd/tomato-detection (accessed on 10 November 2022).
32. Laboro. Laboro Tomato: Instance segmentation dataset. Available online: https://github.com/laboroai/LaboroTomato (accessed on 10 November 2022).
33. Math, R.M.; Dharwadkar, N.V.. *Real-World Tomato Image Dataset for Deep Learning and Computer Vision Applications Involving Precision Agriculture*; Mendeley Data, V1; Vachana Pitamaha DR PG Halakatti College of Engineering and Technology, Visvesvaraya Technological University; Vijaypur, India, 2020. [CrossRef]
34. Skalski, P. Make Sense. Available online: https://www.makesense.ai (accessed on 10 November 2022).
35. Jocher, G.; Chaurasia, A.; Stoken, A.; Borovec, J.; NanoCode012.; Kwon, Y.; TaoXie.; Michael, K.; Fang, J.; imyhxy.; et al. ultralytics/yolov5: v6.2 - YOLOv5 Classification Models, Apple M1, Reproducibility, ClearML and Deci.ai Integrations. 2022. Available online: https://zenodo.org/record/7002879#.Y5HrTnbMKUk (accessed on 10 November 2022) .
36. Wang, C.Y.; Bochkovskiy, A.; Liao, H.Y.M. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv* **2022**, arXiv:2207.02696. https://doi.org/10.48550/ARXIV.2207.02696.
37. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollar, P. Focal Loss for Dense Object Detection. In *Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV)*; IEEE: Venice, Italy, 2017; pp. 2999–3007. [CrossRef]
38. Hoiem, D.; Divvala, S.K.; Hays, J.H. Pascal VOC 2008 challenge. *World Lit. Today* **2009**, *24*.