Automated harvesting by a dual-arm fruit harvesting robot

Takeshi Yoshida (yoshidat934@naro.affrc.go.jp)
National Agriculture and Food Research Organization https://orcid.org/0000-0003-2638-8739
Yuki Onishi
Ritsumeikan University: Ritsumeikan Daigaku
Takuya Kawahara
Ritsumeikan University: Ritsumeikan Daigaku
Takanori Fukao
University of Tokyo: Tokyo Daigaku

Research Article

Keywords: Harvesting robot, Robot manipulation, Deep learning

Posted Date: March 21st, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1455100/v1

RESEARCH

Automated harvesting by a dual-arm fruit harvesting robot

Takeshi Yoshida[1*], Yuki Onishi[2], Takuya Kawahara[2] and Takanori Fukao[3]

Abstract

In this study, we propose a method to automate fruit harvesting with a fruit harvesting robot equipped with robotic arms. Given the future growth of the world population, food shortages are expected to accelerate. Since much of Japan's agriculture is dependent on imports, it is expected to be greatly affected by this upcoming food shortage. In recent years, the number of agricultural workers in Japan has been decreasing and the population is aging. As a result, there is a need to automate and reduce labor in agricultural work using agricultural machinery. In particular, fruit cultivation requires a lot of manual labor due to the variety of orchard conditions and tree shapes, causing mechanization and automation to lag behind. In this study, a dual-armed fruit harvesting robot was designed and fabricated to reach most of the fruits on joint V-shaped trellis that was cultivated and adjusted for the robot. For the fruit harvesting robot to harvest the fruit, it uses sensors and computer vision to detect and estimate the position of the fruit, and then inserts end-effectors into the lower part of the fruit. During this process, there is a possibility of collision within the robot itself or with other fruits depending on the position of the fruit to be harvested. In this study, inverse kinematics and a fast path planning method using random sampling is used to harvest fruits with robot arms. This method makes it possible to control the robot arms without interfering with the fruit or the other robot arm by considering them as obstacles. Through experiments, this study showed that these methods can be used to detect pears and apples outdoors and automatically harvest them using the robot arms.

Keywords: Harvesting robot; Robot manipulation; Deep learning

Introduction

In recent years, various food-related issues have arisen around the world. According to statistics from the United Nations, the world's population reached 7.7 billion in mid-2019. The global population is expected to grow to around 8.5 billion in 2030, 9.7 billion in 2050, and 10.9 billion in 2100[1]. According to another statistic from the Food and Agriculture Organization of the United Nations, there are more than 800 million undernourished people in the world, and food shortages are expected to accelerate as the population grows[2]. Looking at Japan as an example, the calorie-based food self-sufficiency rate in 2020 is 37%, and Japan relies on imports for most of its food[3]. Therefore, the country will be greatly affected by the food shortages that are expected to occur in the future. The number of agricultural workers in Japan has decreased by 394,000 from 1,557,000 to 1,363,000 over the five years from 2015 to 2020. Furthermore, the percentage of people aged 65 and over has increased from 64.8% to 69.5%, indicating that the number of agricultural workers is decreasing and the aging of the population is becoming more serious[4].

Fruit cultivation requires more labor compared to other crops. This is because many tasks such as pollination, fruit picking, fruit set management, and harvesting are done manually. In addition, the fact that orchards are located on a wide variety of terrain, from flat to sloping, and that each orchard and species of tree has a different shape, is one of the reasons why mechanization/automation has been very slow. In order to solve this problem, it is essential to use agricultural machinery and robots that can handle fruit management and harvesting. In particular, pear and apple cultivation require more labor time than other fruits, and account for a large portion of fruit cultivation in Japan. Therefore, this study aims to automatically harvest fruits (pears and apples) in an orchard using a harvest robot.

There are two major challenges in the automatic harvesting of fruits by robots: detection and localization

*Correspondence: yoshidat934@naro.affrc.go.jp
[1]Research Center for Agricultural Robotics, National Agriculture and Food Research Organization, Tsukuba, Japan
Full list of author information is available at the end of the article

of fruits using sensors and harvesting of the detected fruits by the robot. To detect fruits outdoors, we use an object detection method based on deep learning for RGB images to detect the location of fruits in the image. By using deep learning, we aim to stably detect fruits in the shadow of leaves or other fruits, or in an environment with changing light intensity. In addition, since it is difficult to identify the exact location of the fruit using only RGB images, we combine depth images to identify the fruits more accurately.

When harvesting fruits with robot arms, the robotic arm may collide with the robot itself or other fruits depending on the position of the fruit to be harvested. In this study, inverse kinematics and a fast path planning method using random sampling is used to harvest fruits using robot arms. This method makes it possible to control the robot arms without interfering with the fruit or the robot arm by considering them as obstacles. The fruit is harvested by grasping the fruit with a fruit harvesting end-effector attached to the end of the robot arm and twisting the fruit.

Harvesting robot

Outline of harvesting robot

Fig. 1 shows the harvesting robot used in this study. This harvest robot consists of four RGB-D cameras for detecting and locating fruits, two robot arms with end-effectors for harvesting, and a computer for controlling them. In this study, we used UR3 and UR5 robot arms manufactured by Universal Robots. The harvest robot is equipped with two robot arms to increase work efficiency. The upper robot arm (UR5) harvests the fruit on the upper side of the tree, while the lower robot arm (UR3) harvests the fruit on the lower side. It is also designed to approach many of the target fruits, considering the robot arm's operating range and fruit tree standards. In this study, we used Intel's Real sense D435 as the RGB-D camera. As Fig. 1 shows, four RGB-D cameras were set up on the robot: two to look up at the fruit tree from directly below, one to look diagonally upward, and one to look directly to the side. By installing cameras in such a way as to view the fruit tree from many directions, we tried to reduce the number of fruits in the blind spots hidden behind leaves and branches as much as possible. As shown in Fig. 2, the end-effector grasps the fruit when it is close enough and automatically harvests the fruit by twisting it with the rotation of the end-effector.

Target tree and fruit characteristics

In this study, pears (Hosui) and apples (Fuji) are to be harvested. A variety of cultivation methods are used in fruit cultivation, and the degree of branch protrusion and the position of fruits differ depending on each cultivation method. The purpose of this study is to automate harvesting with a fruit harvesting robot for joint V-shaped trellis, which is a type of lined dense planting cultivation[5]. By making the position where the fruit grows flat, lined dense planting cultivation makes it possible for workers or robots to work with high efficiency. We have been working on a harvesting robot which has a single 6-DOF arm for Joint V-shaped trellis [6]. Fig. 3 shows an example of the joint V-shaped trellis.

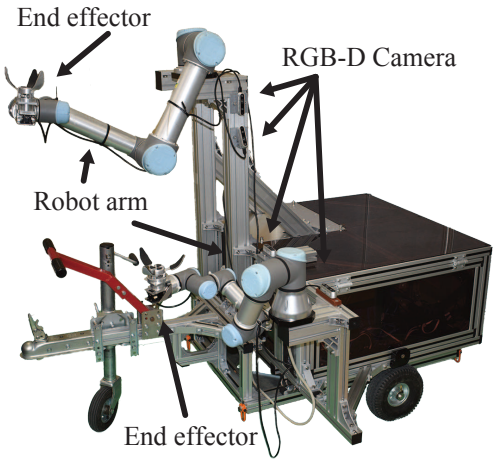


Figure 1: Components of harvesting robot

Figure 2: Harvesting with an end-effector

Related works

Various studies have been done with a focus on developing harvesting robots. These studies can be classified into two categories.

The first category mainly focuses on detection and localization of fruits using sensors. Lin et al. proposed a vision sensing algorithm that can detect guava fruits on trees and obtain promising 3D pose information with an RGB-D sensor [7]. They applied Euclidean clustering to obtain all the individual fruits from the

Figure 3: Joint V-shaped trellis

fruit point cloud corresponding to segmented fruits on the image and estimated the position of the fruit relative to its mother branch. Yu et al. proposed a localization algorithm to detect the picking point on strawberry stems with Rotational You Only Look Once (R-YOLO), which predicts the rotation of the bounding box of the fruit target [8]. Their harvesting robot measures the distance to the target fruit with a pair of laser beam sensors attached to the head of the fingers of the robot instead of detecting the depth of the target fruit. Yoshida et al. proposed a method for detecting cutting points on tomato peduncles using an RGB-D camera mounted on a harvesting robot [9] [10]. In their approach, several types of region growings were used to construct a directed acyclic graph. Subsequently, using the Mahalanobis distance defined based on statistical information, they detected appropriate cutting points on the peduncles.

The second category deals primarily with robotic systems that perform everything from recognition to harvesting. Irie et al. proposed an asparagus harvesting robot that measured whether the asparagus was tall enough to harvest using a 3D sensor [11, 12]. They also proposed a robotic arm mechanism and an end effector to grasp and cut asparagus. Hayashi et al. proposed a strawberry-harvesting robot consisting of a cylindrical manipulator, end effector, machine vision unit, storage unit, and traveling unit [13]. The end-effector of their robot was composed of a gripper for simultaneously grasping and cutting the peduncle of the fruit, a suction device for holding the fruit to avoid damage. Lili et al. proposed a tomato harvesting robot consisting of a four-wheel independent steering system, 5-DOF harvesting system, laser navigation system, and binocular stereo vision system [14]. The harvesting robot was designed for harvesting tomatoes in a greenhouse. Yaguchi et al. proposed a tomato fruit recognition method for the harvesting robot [15]. First, color-based point cloud extraction was applied to a 3D point cloud from a stereo camera. Second, distance-based clustering was applied to separate the candidate point cloud into tomato clusters. Thereafter, the harvesting robot inserts its end-effector into the fruit position, which is decided with sphere fitting using RANSAC. Silwal et al. presented the design and field testing of a robotic system to harvest apples [16]. Their robotic system integrated a global camera setup, 7-DOF manipulator, and three-fingered grasping end-effector to execute fruit picking with open-loop control. From the Results of field studies, they showed that horticultural practices play a critical role in the performance of robotic fruit harvesting systems. Arad et al. proposed a robot for harvesting sweet pepper fruits in greenhouses [17] [18]. They proposed a Flash-No-Flash controlled illumination acquisition protocol to stabilize the effects of illumination for color-based detection algorithms. Their sweet pepper harvesting robot applies a visual servo that keeps the detected center of the fruit in a predetermined position in the camera image to lower the requirements for camera calibration and 3D coordinates.

Focusing on a harvesting robot that has dual arms, Ling et al. proposed a dual-arm harvesting robot for harvesting tomato in a greenhouse [19]. Their robot had 2 mirrored 3-DOF arms, a right arm for grasping and a left arm for detaching. Their proposed framework detected ripe tomatoes by an algorithm combining an AdaBoost classifier and color analysis using the RGB image. Then, the 3D position of a tomato object is obtained according to the relationship between 2D image pixel coordinate and 3D point cloud coordinate acquired from the stereo camera. Based on the 3D position, the vacuum cup-type end-effector of the robot grasped the target fruit and another end-effector cut the stem to harvest the fruit. Their dual-arm approach avoided the tomato shaking of the tomato when cutting the stem.

Sepúlveda et al. proposed a dual-arm aubergine harvesting robot [20]. The robot consisted of two robot arms configured like humans to optimize the dual workspace. They applied the image segmentation algorithm based on a Support Vector Machine pixel classifier, a watershed transform and a point cloud registration for detecting and localizing aubergines. Depending on the workspace and the locations of the fruits, the planning algorithm determined the movement which involved either the simultaneous harvesting of two pieces of fruit or harvesting a single fruit with a single arm. In addition, the planning algorithm determined a collaborative behavior between the arms if an aubergine was occluded by leaves.

These robots have robot arms attached to each of their shoulders across their torso like humans, and the possibility of collisions at the joints along the way is extremely low. On the other hand, the robot used in this study has dual robot arms attached to the same side, which increases the possibility of collision between the arms, and this is an issue that needs to be addressed. This study proposes a method to automate fruit harvesting with a fruit harvesting robot equipped with robotic arms without colliding with the target fruit or the robot. In addition to the harvesting method, we also propose a method for locating and integrating fruits in an outdoor environments. Thus, this study belongs to the above second category.

Automatic fruit harvesting method

Fig. 4 shows a flowchart of the sequence of automatic harvesting by the harvesting robot. Automatic fruit harvesting by a robot consists of five steps: (1) fruit detection, (2) fruit localization, (3) integration of information from each camera, (4) inverse kinematics, and (5) path planning.

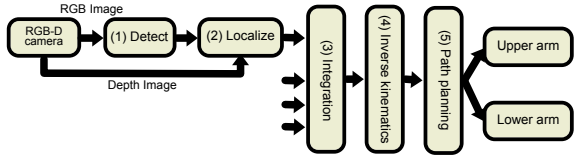


Figure 4: Flowchart of automatic fruit harvesting

First, RGB-D cameras that can simultaneously obtain RGB and depth information are used to detect and locate the fruits in the images. Deep learning is performed on the RGB image acquired from RGB-D camera to detect the position of fruits in the image. Next, the 3D positions of the fruits are identified by combining the positions of the fruits in the RGB image and the depth image.

In order to reduce the number of fruits that cannot be seen by the cameras due to occlusion by leaves or other fruits, the robot has multiple cameras. The next step is to integrate the information obtained from these multiple cameras. The order of the integrated information is random, so it is replaced by an order suitable for harvesting. Here, the fruit information we have obtained so far is the 3D position. However, in the case of an articulated robot arm, the information required for commands is the angle of each joint. Therefore, each joint angle is calculated from the 3D position and posture of the end-effector using inverse kinematics. Next, the path is planned to reach the joint angles calculated by inverse kinematics, and the harvesting motion is performed.

Fruit detection

First, the RGB images are acquired from the RGB-D cameras mounted on the robot, the fruits in the images are detected. It is necessary to combine information such as color and texture in order to achieve sufficient accuracy. In this study, we apply Single Shot Multibox Detector (SSD), which is one of the object detection algorithms, to detect fruits in images [21]. SSD is a method for detecting objects in images using a single neural network, and was proposed by Liu et al. Other methods for object detection include Faster R-CNN[22], You Only Look Once[23]. In this study, SSD is used because speed and accuracy are important. All the information about the detected bounding box $\boldsymbol{D}$ is obtained from the results of the detection of the fruit in the image by SSD.

The bounding box information $\boldsymbol{D}$ is shown in Eq. (1). One bounding box information σ shown in Eq. (2) consists of the pixel coordinates $(x_{min},\ y_{min})$ of the upper left corner of the box and the pixel coordinates $(x_{max},\ y_{max})$ of the lower right corner of the box.

$$\boldsymbol{D} = \begin{bmatrix} \sigma_1 & \cdots & \sigma_n \end{bmatrix} \tag{1}$$

$$\sigma = \begin{bmatrix} x_{min} & x_{max} & y_{min} & y_{max} \end{bmatrix}^T \tag{2}$$

Fruit localization

In the next step, the fruit is considered as a sphere, and the coordinates and radius of the sphere are estimated from the bounding box information σ obtained in the previous section, RGB image, and depth image. Considering that the spherical shape of the fruit is projected onto the 2D image, the circular shape of the fruit is detected from the bounding box detected by SSD. In this study, we use the Hough transform for circle detection to detect fruit circles from RGB images [24].

The relationship between the point $(X_{camX},\ Y_{camX},\ Z_{camX})$ in 3D space and the point $(x,\ y)$ in 2D image can be expressed by Eq. (3), where $f_x,\ f_y$ are the focal lengths of the camera, $c_x,\ c_y$ are the image centers of the camera, and d is the depth information at the point $(x,\ y)$ obtained from the depth image.

$$d \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_{camX} \\ Y_{camX} \\ Z_{camX} \end{bmatrix} \tag{3}$$

The coordinates $(X_{camX},\ Y_{camX},\ Z_{camX})$ of the sphere and the radius R of the sphere can be obtained by the least squares method using the equation of the sphere.

The next step is to perform a coordinate transformation of the fruit. The coordinates of the fruit obtained from Eq. (3) are in the camera coordinate sys-

tem. However, the position in the robot arm coordinate system is required in order to give commands to the robot arm. Therefore the rotation and translation matrices $\boldsymbol{T}$ between each robot arm and the camera coordinate system, which were obtained in calibration beforehand, are used to perform coordinate transforms to the robot arm coordinates as shown in Eq. (4).

$$\begin{bmatrix} X_{arm} \\ Y_{arm} \\ Z_{arm} \\ 1 \end{bmatrix} = \boldsymbol{T} \begin{bmatrix} X_{camX} \\ Y_{camX} \\ Z_{camX} \\ 1 \end{bmatrix} \tag{4}$$

Finally, we use Eq. (5) to find the reciprocal of the pixel distance from the center of the circle detected to the center of the image, where w is the width of the image and h is the height of the image. Fruits at the edge of the image may cause large deviations in sphere detection due to insufficient RGB and depth information. Therefore, the coordinates of each camera near the center of the image can be used preferentially by using this index when integrating the information of each camera.

$$s = \frac{1}{(w/2 - c_x)^2 + (h/2 - c_y)^2} \tag{5}$$

Thus, the information $\boldsymbol{L}$ for all fruits can be obtained by using Eq. (4) and Eq. (5). Equation (6) shows the information $\boldsymbol{L}$ for all fruits.

The information σ of one fruit consists of the position (X_{arm}, Y_{arm}, Z_{arm}) of the fruit in the robot arm coordinate system, the radius R of the fruit, and the score s

$$\boldsymbol{L} = \begin{bmatrix} \sigma_1 & \cdots & \sigma_n \end{bmatrix} \tag{6}$$

$$\sigma = \begin{bmatrix} X_{arm} & Y_{arm} & Z_{arm} & R & s \end{bmatrix}^T \tag{7}$$

Integration of fruit information

In this section, we integrate the fruit information $\boldsymbol{L_X}$ obtained from each camera. We can find $\boldsymbol{L_{All}}$ that matches the actual number of fruits without duplication of identical fruits by using Algorithm 1 for all $\boldsymbol{L_X}$. First, algorithm 1 extracts the element σ_{detect} of $\boldsymbol{L_X}$ and the element σ_{target} of $\boldsymbol{L_{All}}$. Next, in the seventh line, if the distance between each element σ is less than 0.05 [m], they are considered to be the same object. We set the threshold here to less than 0.05 [m] since the radius of the fruit is roughly 0.05 [m]. If the score of σ_{detect} is greater than the score of σ_{target} in line 9, set σ_{target} of $\boldsymbol{L_{All}}$ to σ_{detect} and end the iteration process. This operation makes it possible to prioritize the use of coordinates close to the center of the image for the same fruit. If only the condition in line 7 is satisfied, the iteration will be terminated without any processing. If all the elements in $\boldsymbol{L_{All}}$ do not satisfy the condition in line 7, add σ_{detect} to $\boldsymbol{L_{All}}$ as a new harvest target. This process is repeated for all elements of $\boldsymbol{L_{All}}$ and $\boldsymbol{L_X}$ of all cameras to obtain $\boldsymbol{L_{All}}$ of all the detected harvest targets.

Algorithm 1 Synchronize Target Info

```
1: the empty L : L_All
2: the cameraX L : L_X
3: for X in 1, 2, 3, ··· do
4:     for σ_detect in L_X do
5:         AddFlag = True
6:         for σ_target in L_All do
7:             if Distance(σ_detect, σ_target) ≤ 0.05[m] then
8:                 AddFlag = False
9:                 if σ_detect score ≥ σ_target score then
10:                    ConvartData(L_All, σ_target, σ_detect)
11:                end if
12:                Break
13:            end if
14:        end for
15:        if AddFlag then
16:            AddNewTarget(L_All, σ_detect)
17:        end if
18:    end for
19: end for
```

Next, out proposed method rearranges $\boldsymbol{L_{All}}$ in the order in which they will be harvested to avoid robot collisions. The robot arm approaches from the direction where the X coordinate becomes negative due to the configuration of the robot. Therefore, we prioritize the harvesting of fruits in order from the negative X-coordinate, so that fruits not targeted for harvesting do not become obstacles during path planning. First, $\boldsymbol{L_{All}}$ is sorted in descending order based on the value of X in $\boldsymbol{L_{All}}$. If the distance between the X coordinates of different fruits σ_i and σ_{i+1} on the same branch is less than a threshold, the lower fruit will be harvested first. This means that the sorting is done in descending order based on the Z-axis direction only between σ_i and σ_{i+1}. Let $\boldsymbol{L}$ obtained after these permutations be $\boldsymbol{L_{AllSorted}}$. In this study, quick sort, which is practical and fast, was used because fast processing leads to shorter harvesting time.

Inverse kinematics

In this step, each joint angle of the robot arm at an arbitrary position and posture is obtained using inverse kinematics[25] for each $\boldsymbol{L_{AllSorted}}$ obtained in the previous section. In order to harvest fruit with an end-effector attached to the end of the robot arm, the end-effector needs to be moved in its position $\boldsymbol{p}$ and posture $\boldsymbol{R}$ ($\boldsymbol{P} = (\boldsymbol{p}, \boldsymbol{R})$) as specified. In the case of the articulated robot arm used in this study, the position and posture $\boldsymbol{P}$ of the end-effector are determined by the angle $\boldsymbol{q}$ of each joint. Therefore, it is necessary

to establish the relationship between the joint coordinate system representing the joint angles of the robot arm and the end-effector coordinate system representing the position and posture of the end-effector. The problem of determining the angle $\boldsymbol{q}$ of each joint from the position and posture $\boldsymbol{P}$ of the end-effector is called an inverse kinematics problem, and its solution is expressed using a nonlinear function $\boldsymbol{f}^{-1}$ as follows.

$$\boldsymbol{q} = \boldsymbol{f}^{-1}(\boldsymbol{P}) \tag{8}$$

The inverse kinematics problem is more difficult to solve than the forward kinematics problem because there may be several solutions $\boldsymbol{q}$ for a certain $\boldsymbol{P}$, or there may be no solution $\boldsymbol{q}$. The solution of the inverse kinematics problem can be roughly divided into the following two types.

1 It is found numerically using iterative algorithms.
2 It is obtained geometrically or algebraically using the features of the mechanism.

In this study, we use the latter geometric solution method considering real time use.

The Denavit-Hartenberg notation is used to set up a coordinate system for each joint in order to obtain the robot's end-effector position and posture.

As shown in Fig. 5, d_i is the distance between the links, θ_i is the angle between the links, a_i is the length of the links, and α_i is the torsion angle of the links. The homogeneous transformation matrix ${}^{i-1}\boldsymbol{T}_i$ from coordinate system Σ_{i-1} to coordinate system Σ_i is expressed as in Eq. (9).

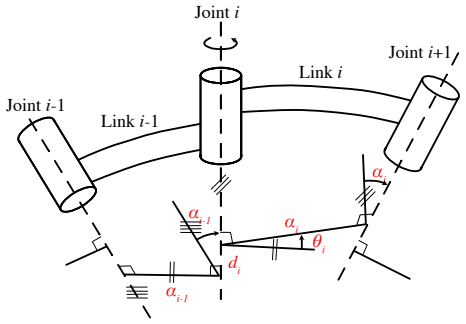


Figure 5: Denavit-Hartenberg parameters

$${}^{i-1}\boldsymbol{T}_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{9}$$

The computations are performed in turn until a solution is obtained for $\boldsymbol{L}_{\boldsymbol{All\ sorted}}$. When a solution is obtained, the information is used in the next step.

Inverse kinematics model for UR arms

Table 1 shows Denavit-Hartenberg parameters for robot arms (UR arms) used in this study.

Table 1: Denavit-Hartenberg parameters for UR arms

link	a_i[m]	α_i[rad]	d_i[m]	$\theta_i[rad]$
1	0	$\frac{\pi}{2}$	d_1	θ_1
2	a_2	0	0	θ_2
3	a_3	0	0	θ_3
4	0	$\frac{\pi}{2}$	d_4	θ_4
5	0	$-\frac{\pi}{2}$	d_5	θ_5
6	0	0	d_6	θ_6

Eq. (10) is a position of the end-effector, and Eq. (11) is a posture of the end-effector, where ϕ is the rotation of the end-effector around the Z-axis (roll), θ is the rotation of the end-effector around the Y-axis (pitch), and ψ is the rotation of the end-effector around the X-axis (yaw).

$$\boldsymbol{p} = \begin{bmatrix} p_x & p_y & p_z \end{bmatrix} = \begin{bmatrix} X & Y & Z \end{bmatrix} \tag{10}$$

$$\begin{aligned} \boldsymbol{R} &= \begin{bmatrix} R_{11} & R_{21} & R_{31} \\ R_{12} & R_{22} & R_{32} \\ R_{13} & R_{23} & R_{33} \end{bmatrix} \\ &= \begin{bmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \\ &\quad \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\psi & -\sin\psi \\ 0 & \sin\psi & \cos\psi \end{bmatrix} \end{aligned} \tag{11}$$

The angles $\theta_i (i = 1, 2, \ldots, 6)$ of each joint in Eq. (10) and Eq. (11) of UR arms are as follows, where a_i, d_i are the parameters for the Denavit-Hartenberg notation.

$$\begin{aligned} \theta_1 = &\arctan\left(\frac{p_y - d_6 R_{23}}{p_x - d_6 R_{13}}\right) \\ &\pm \arccos\left(\frac{d_4}{\sqrt{{p_y - d_6 R_{23}}^2 + {p_x - d_6 R_{13}}^2}}\right) + \frac{\pi}{2} \end{aligned} \tag{12}$$

$$\theta_5 = \pm \arccos\left(\frac{\sqrt{p_x \sin\theta_1 - p_y \cos\theta_1 - d_4}}{d_6}\right) \tag{13}$$

$$\theta_6 = \arctan\left(\frac{R_{22}\cos\theta_1 - R_{12}\sin\theta_1}{R_{11}\sin\theta_1 - R_{21}\cos\theta_1}\right) \tag{14}$$

$$\begin{aligned} x_{04x} = &-(R_{13}\cos\theta_1 + R_{23}\sin\theta_1)\sin\theta_5 \\ &-((R_{12}\cos\theta_1 + R_{22}\sin\theta_1)\sin\theta_6 \\ &-(R_{11}\cos\theta_1 + R_{21}\sin\theta_1)\cos\theta_6)\cos\theta_5 \end{aligned} \tag{15}$$

$$x_{04y} = (R_{21}\cos\theta_6 - R_{32}\sin\theta_6)\cos\theta_5 - R_{33}\sin\theta_5 \quad (16)$$

$$\begin{aligned} p_{13x} =& d_5((R_{11}\cos\theta_1 + R_{21}\sin\theta_1)\sin\theta_6 \\ &+ (R_{12}\cos\theta_1 + R_{22}\sin\theta_1)\cos\theta_6) \\ &- d_6(R_{13}\cos\theta_1 + R_{23}\sin\theta_1) \\ &+ p_x\cos\theta_1 + p_y\sin\theta_1 \end{aligned} \quad (17)$$

$$p_{13y} = p_z - d_1 - d_6 R_{33} + d_5(R_{32}\cos\theta_6 + R_{31}\sin\theta_6) \quad (18)$$

$$\theta_3 = \arccos\left(\frac{{p_{13x}}^2 + {p_{13y}}^2 - {a_2}^2 - {a_3}^2}{2a_2a_3}\right) \quad (19)$$

$$\theta_2 = \arctan\left(\frac{(a_2 + a_3\cos\theta_3)p_{13y} \pm (a_3\sin\theta_3)p_{13x}}{(a_2 + a_3\cos\theta_3)p_{13y} \mp (a_3\sin\theta_3)p_{13x}}\right) \quad (20)$$

$$\theta_4 = \arctan\left(\frac{x_{04y}\cos(\theta_2+\theta_3) - x_{04x}\sin(\theta_2+\theta_3)}{x_{04x}\cos(\theta_2+\theta_3) + x_{04y}\sin(\theta_2+\theta_3)}\right) \quad (21)$$

Path planning

Our harvesting robot has dual robot arms attached in the same direction from the robot's body, which increases the possibility of collision between the arms, and this is an issue that needs to be addressed. In this section, in order to avoid damaging the fruits and to prevent the robot arm from colliding with the robot itself or another arm, we plan a path to move to the joint angle determined in the previous section by avoiding obstacles such as the fruit and the robot itself. Fig. 6 shows the robot's perception of its own state and the location of the fruits, which is a prerequisite for path planning.

Various methods of path planning have been proposed in the past. They can be broadly classified into two categories: methods based on given nodes in the target space, and methods based on a continuous space. In situations where there are obstacles or other people in the operating area, or where the environment is dynamically changing, it is difficult to provide specified nodes. Therefore, when considering path planning for autonomous vehicles and articulated robots, the latter method is applied. These methods include the potential method, PRM(probabilistic road map)[26] and RRT(rapidly-exploring random tree)[27].

PRM and RRT use random sampling to speed up the computation. Specifically, RRT is applicable to search in high-dimensional spaces because it does not require pre-processing and has a high ability to avoid local solutions. Therefore, several modification methods have been proposed to extend RRT to be applicable to path planning under various conditions. Among them, T-RRT (Transition-based RRT), which introduces an evaluation function that can be designed according to the situation into the routing procedure, has high versatility[28]. Therefore, in this study, T-RRT is used for automatic harvesting by robotic arms.

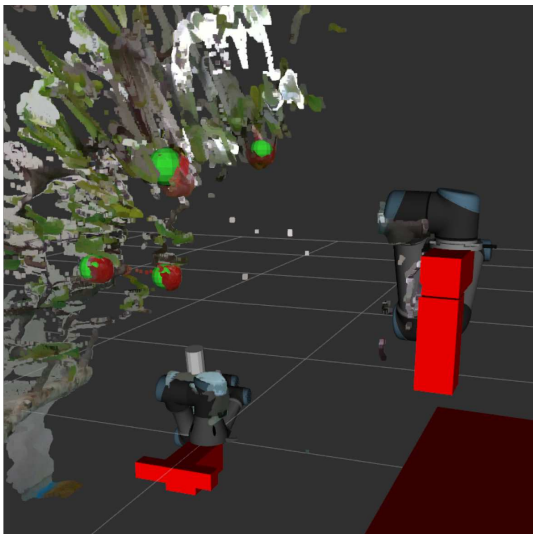


Figure 6: Example of robot and fruits arrangement

Algorithm 2 and Fig. 7 show the process of T-RRT. The first step is to create the surrounding environment CS for path planning. The next step is to set the evaluation function $c()$ of path planning and a start point q_{init} and a goal point q_{goal}. The fourth line initializes the search tree τ with the starting point q_{init}. The fifth line loops until the search tree τ reaches the goal point q_{goal}. The sixth line sets q_{rand}, which does not contact any obstacle in the search area. The seventh line searches for the node q_{near} that is closest to q_{rand} in the search tree τ. The eighth line judges whether it is possible to connect q_{near} to q_{new} in the search tree τ. If no connection can be made, the algorithm returns to the beginning of the loop. The 11th line evaluates q_{new} and q_{near} by the evaluation function and finds the shortest connection and makes a decision to add q_{new} to the search tree τ.

Algorithm 3 shows a function of transition test of T-RRT. The first step is to filter by the maximum cost c_{max}. The next step is to compare the cost c_j of the new node with the cost c_i of the parent node and the result is True if the cost of the new node is lower. If the cost of the new node is higher, the algorithm makes

Algorithm 2 Transition-based RRT

the configuration space CS
the cost function c
the root q_{init} and the goal q_{goal}
$\tau \leftarrow$ InitTree(q_{init})
while not StopCondition(τ, q_{goal}) **do**
  $q_{rand} \leftarrow$ SampleConf(CS)
  $q_{near} \leftarrow$ BestNeighbor(q_{rand}, τ)
  if not Extend(τ, q_{rand}, q_{near}, q_{new}) **then**
    Continue
  end if
  if TransitionTest($c(q_{near})$, $c(q_{new})$, $d_{near-new}$) **and** MinExpandControl(τ, q_{near}, q_{rand}) **then**
    AddNewNode(τ, q_{new})
    AddNewEdge(τ, q_{near}, q_{new})
  end if
end while

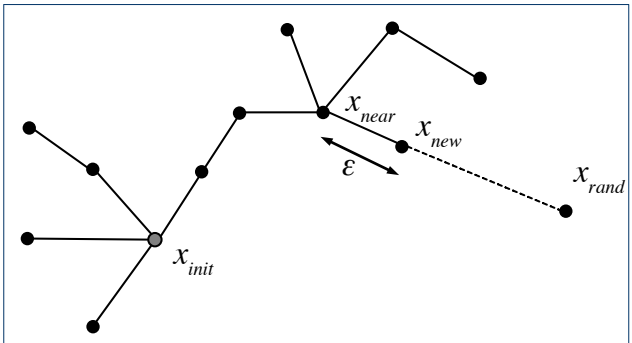


Figure 7: Transition-based RRT

a True or False decision using the probability p. The number of costly nodes added decreases by gradually decreasing the probability p.

Experiments

Fruit detection

In this study, we used images of pears and apples. We collected images of fruit trees viewed from multiple directions as viewed by the harvesting robot. We used a lot of the lower images because they have more occlusion and are more affected by sunlight. We also added images taken at other times of the day to the training set in order to accommodate various sunlight conditions. Next, we evaluate how well the SSD learning model can perform detection on untrained images. The evaluation method is based on the correct response data by visual inspection, and the test is based on how well the system can detect 50 untrained images in front-lit and back-lit conditions. Table 2 shows learning parameters and results of detection. As a result, it was possible to detect more than 95% of fruits in both targets.

Fig. 8 and Fig. 9 show some of the resulting images. We were able to detect some fruits that were occluded by other fruits and leaves, but we could not detect

Algorithm 3 TransitionTest

if $c_j > c_{max}$ **then**
  return False
end if
if $c_j < c_i$ **then**
  return True
end if
$p = \exp \frac{-(c_j - c_i)/d_{ij}}{K \cdot T}$
if $Rand(0,1) < p$ **then**
  $T = T/\alpha$
  $nFail = 0$
  return True
else
  if $nFail > nFail_{max}$ **then**
    $T = T \cdot \alpha$
    $nFail = 0$
  else
    $nFail = nFail + 1$
  end if
  return False
end if

Table 2: Learning parameters and results of detection

Target	Pear (Housui)		Apple (Fuji)	
Environment	front-lit	back-lit	front-lit	back-lit
Architecture	TensorFlow			
Net	Mobile Net			
Fine Tuning	ssd mobilenet v2 coco			
Base learning rate	0.0001			
batch size	32			
Learning times	56197		60000	
Detected	383	414	405	431
Not Detected	15	15	12	14
False Detected	17	12	7	9
Precision	95.8%	95.5%	97.1%	96.9%
Recall	95.2%	97.8%	98.3%	98.0%
F-measure	0.955	0.968	0.977	0.974

fruits that were almost hidden. This problem can be solved by supplementing the fruit with images from cameras installed in multiple directions.

Comparison of path planning methods

In order to compare the various path planning methods, we simulated the harvesting operation. The simulations were performed for the same fruit from the same position for the simulated fruit. Table 3 shows the characteristics of each path planning method and the time required for the harvesting operation. In some cases, the optimization of the route search did not converge. Therefore, when more than 10 seconds had elapsed since the start of the route search, the calculation was terminated, and the operation was performed on the route planned up to that point. PathPlan shows the time taken for the path plan. Harvest shows the time taken for harvesting. Sum shows the total time for all of them.

This result shows that T-RRT can perform very fast path planning compared to RRT*, PRM, and PRM*. On the other hand, RRT is faster than T-RRT because

Table 3: Comparison of path planning methods

Target		RRT [27]	T-RRT [28]	RRT* [29]	PRM [26]	PRM* [29]
1	PathPlan[sec]	0.097	0.236	10.115	10.083	10.111
	Harvest[sec]	9.793	9.744	9.785	9.907	9.739
	Sum [sec]	9.89	9.98	19.9	19.99	19.85
2	PathPlan[sec]	0.497	0.497	10.114	10.109	10.132
	Harvest[sec]	9.843	11.833	11.996	11.951	11.898
	Sum[sec]	10.34	12.33	22.11	22.06	22.03
3	PathPlan[sec]	0.432	0.268	10.078	10.179	10.104
	Harvest[sec]	13.088	11.962	11.842	13.041	11.986
	Sum[sec]	13.52	12.23	21.92	23.22	22.09
4	PathPlan[sec]	0.269	0.24	10.137	10.106	10.083
	Harvest[sec]	8.511	8.86	8.313	8.344	8.047
	Sum[sec]	8.78	9.10	18.45	18.45	18.13

(a) Frontlighting (b) Backlighting

Figure 8: Result of pear detection

(a) Frontlighting (b) Backlighting

Figure 9: Result of apple detection

(a) Pear (b) Apple

Figure 10: Experimental fields

the optimization method is not introduced in the algorithm itself. However, as shown in the result of Target 3, it was observed that the harvesting operation took a long time due to the inclusion of unnecessary movements. Based on these results, T-RRT, which can perform stable and fast path planning and harvesting operations, is used in this study.

Experiments for autonomous harvesting

Experimental environment

Automated harvesting experiments were conducted at three locations. Fig. 10a shows pear field at Kanagawa Agricultural Technology Center. Fig. 10b shows apple field at Miyagi Prefectural Institute of Agriculture and Horticulture.

Results and Discussion

Fig. 11 to Fig. 13 show the results of the automatic pear harvesting experiment. In this experiment, only the lower arm was used, and only one camera was used for harvesting. Fig. 12 shows that the location identification also coincided with the fruit point cloud and the estimated red spheres. Fig. 13a to Fig. 13c show that the automatic harvesting could be done without colliding with the target fruit or the robot. On the other hand, in the fruit detection, the overlapping fruits could not be detected as shown in Fig. 11. This shows that a single camera is not sufficient for a fruit harvesting robot and that it is necessary to integrate multiple cameras.

Fig. 14 to Fig. 16 show the results of the automatic apple harvesting experiment. In this experiment, we used all the arms and cameras shown in Fig. 1. Fig. 15 shows that the fruit point cloud and the estimated red spheres are consistent for the fruit localization. Fig. 16a to Fig. 16c show that the automatic harvesting could be done without colliding with the target fruit or the robot. Fig. 14 shows that the detection

Figure 11: Result of pear detection

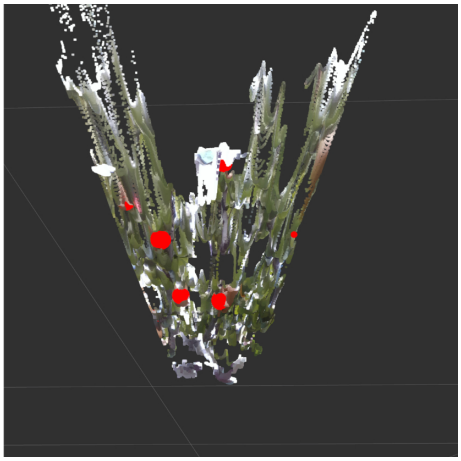
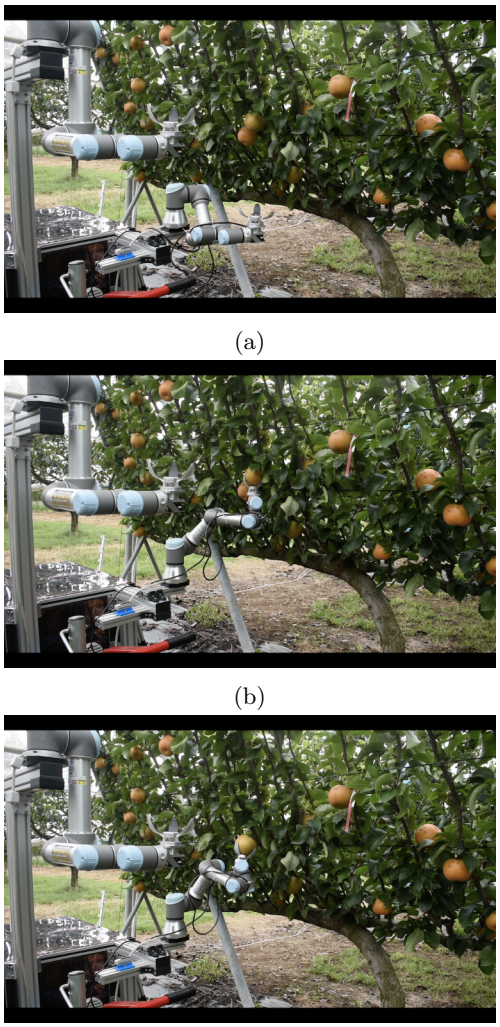


Figure 12: Result of pear localization

that could not be done by one camera can be done by another camera, indicating that they are sufficiently complementary. In addition, although the illumination conditions varied among the cameras, each camera was able to detect the image sufficiently.

We showed that the proposed method can be used to automatically harvest pears and apples with a robot arm in about 20 seconds per harvest. As shown in these experiments, fruits that were more than a certain distance from branches and occluded by other fruits were able to be harvested with sufficient success. On the other hand, the success rate was very low for harvesting fruits that were very close to branches. This can be caused by the robot arm or hand getting caught on a branch or colliding with it. If the entire fruit tree is included in the collision detection, the leaves also become obstacles and the workspace of the robot arm becomes very narrow. In order to use only branches for collision detection, it is necessary to detect only branches, but this is difficult because there are so many occlusions with leaves. In order to be able to perform automatic harvesting even in such a very close environment to the branches, the harvesting method, behavior, and detection methods need to be improved significantly.



(a)

(b)

(c)

Figure 13: Pear harvesting motion

Conclusion

In this study, we proposed a method for locating fruits in an outdoor environment and a method for automated harvesting of fruits with robot arms.

By using SSD, we have shown that fruit detection can be performed with an accuracy of more than 95% outdoors, even in back-lit conditions. The fruit detection system developed in this study can detect even different varieties of pears and apples by re-learning the target fruit. However, as shown in the automatic pear harvesting experiment using a single camera, the accuracy of detecting a fruit hidden by leaves or other fruits was reduced. Therefore, in this study, the number of occluded fruits was reduced as much as possible by installing cameras in multiple directions.

In order to avoid damaging the fruits and to prevent the robot arm from colliding with the robot it-

Figure 14: Result of apple detection

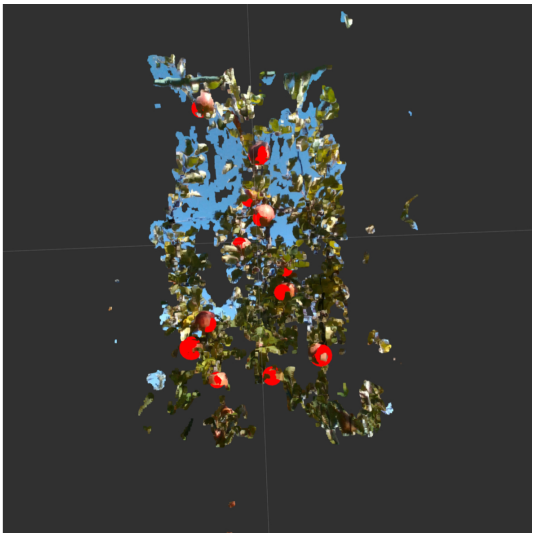


Figure 15: Result of apple localization

(a)

(b)

(c)

Figure 16: Apple harvesting motion

self or another arm, we showed that path planning to the harvesting target can be done relatively quickly, in less than 0.5 seconds using inverse kinematics and T-RRT. In addition, when integrating images taken from multiple directions, the proposed method is set up to harvest them in a safer order to avoid collisions.

Through our experiments, we showed that the fruits can be harvested in about 20 seconds each time. This means that a single fruit can be harvested in a maximum of 10 seconds by moving two robot arms simultaneously, which is equivalent to harvesting by a human. As with detection, it is likely that it is possible to harvest different varieties of pears and apples.

Acknowledgements

We would like to express our gratitude to Kanagawa Agricultural Technology Center, Miyagi Prefectural Institute of Agriculture and Horticulture and other related parties for their cooperation in cultivating fruits.

Funding

This research was supported by grants from the Project of the Bio-oriented Technology Research Advancement Institution, NARO (the research project for the future agricultural production utilizing artificial intelligence).

Abbreviations

RGB-D: Red, Green, Blue and Depth
DOF: Degree of Freedom
R-CNN: Region-based Convolutional Neural Network
R-YOLO: Rotational You Only Look Once
RANSAC: RANdom SAmple Consensus
SSD: Single Shot MultiBox Detector
RRT: Rapidly-exploring Random Tree

T-RRT: Transition-based Rapidly-exploring Random Tree

PRM: Probabilistic Road Map

Availability of data and materials

Not applicable

Competing interests

The authors declare that they have no competing interests.

Authors' contributions

TY and YO conducted all research and experiments. TK and TF conducted a research concept, participated in design adjustment, and drafted a paper draft assistant. All authors read and approved the final manuscript.

Author details

[1]Research Center for Agricultural Robotics, National Agriculture and Food Research Organization, Tsukuba, Japan. [2]Graduate School of Science and Engineering, Ritsumeikan University, Kusatsu, Japan. [3]Graduate School of Information Science and Technology, University of Tokyo, Bunkyo, Japan.

References

1. Nations, U.: World Population Prospects 2019: Highlights (2019). https://population.un.org/wpp/Publications/Files/WPP2019_Highlights.pdf Accessed Accessed 26 Feb 2022
2. Food, of the United Nations, A.O.: The future of food and agriculture - Alternative pathways to 2050 (2018). https://www.fao.org/3/CA1553EN/ca1553en.pdf Accessed Accessed 26 Feb 2022
3. Ministry of Agriculture, F., Fisheries: Food Balance Sheet. (In Japanese) (2021). https://www.maff.go.jp/j/tokei/kouhyou/zyukyu/attach/pdf/index-1.pdf Accessed Accessed 26 Feb 2022
4. Ministry of Agriculture, F., Fisheries: Overview of the 2020 Census of Agriculture and Forestry. (In Japanese) (2021). https://www.maff.go.jp/j/press/tokei/census/attach/pdf/210427-3.pdf Accessed Accessed 26 Feb 2022
5. Kusaba, S.: Integration of the tree form and machinery). FARMING MECHANIZATION (3189), 5–9 (2017). (In Japanese)
6. Onishi, Y., Yoshida, T., Kurita, H., Fukao, T., Arihara, H., Iwai, A.: An automated fruit harvesting robot by using deep learning. ROBOMECH Journal **6**(13) (2019)
7. Lin, G., Tang, Y., Zou, X., Xiong, J., Li, J.: Guava detection and pose estimation using a low-cost rgb-d sensor in the field. Sensors **19**(2) (2019)
8. Yu, Y., Zhang, K., Liu, H., Yang, L., Zhang, D.: Real-time visual localization of the picking points for a ridge-planting strawberry harvesting robot. IEEE Access **8**, 116556–116568 (2020)
9. Yoshida, T., Fukao, T., Hasegawa, T.: Fast detection of tomato peduncle using point cloud with a harvesting robot. Journal of Robotics and Mechatronics **30**(2), 180–186 (2018)
10. Yoshida, T., Fukao, T., Hasegawa, T.: Cutting point detection using a robot with point clouds for tomato harvesting. Journal of Robotics and Mechatronics **32**(2), 437–444 (2020)
11. Irie, N., Taguchi, N., Horie, T., Ishimatsu, T.: Development of Asparagus Harvester Coordinated with 3-d Vision Sensor. J. Robot. Mechatron. **21**(5), 583–589 (2009)
12. Irie, N., Taguchi, N.: Asparagus Harvesting Robot. J. Robot. Mechatron. **26**(2), 267–268 (2014)
13. Hayashi, S., Shigematsu, K., Yamamoto, S., Kobayashi, K., Kohno, Y., Kamata, J., Kurita, M.: Evaluation of a strawberry-harvesting robot in a field test. Biosystems Engineering **105**(2), 160–171 (2010)
14. Lili, W., Bo, Z., Jinwei, F., Xiaoan, H., Shu, W., Yashuo, L., Qiangbing, Z., Chongfeng, W.: Development of a tomato harvesting robot used in greenhouse. International Journal of Agricultural and Biological Engineering **10**(4), 140–149 (2017)
15. Yaguchi, H., Nagahama, K., Hasegawa, T., Inaba, M.: Development of an autonomous tomato harvesting robot with rotational plucking gripper. In: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 652–657 (2016)
16. Silwal, A., Davidson, J.R., Karkee, M., Mo, C., Zhang, Q., Lewis, K.: Design, integration, and field evaluation of a robotic apple harvester. Journal of FIELD ROBOTICS **34**(6), 1140–1159 (2017)
17. Arad, B., Kurtser, P., Barnea, E., Harel, B., Edan, Y., Ben-Shahar, O.: Controlled lighting and illumination-independent target detection for real-time cost-efficient applications. the case study of sweet pepper robotic harvesting. Sensors **19**(6) (2019)
18. Arad, B., Balendonck, J., Barth, R., Ben-Shahar, O., Edan, Y., Hellström, T., Hemming, J., Kurtser, P., Ringdahl, O., Tielen, T., van Tuijl, B.: Development of a sweet pepper harvesting robot. Journal of Field Robotics **37**(6), 1027–1039 (2020)
19. Ling, X., Zhao, Y., Gong, L., Liu, C., Wang, T.: Dual-arm cooperation and implementing for robotic harvesting tomato using binocular vision. Robotics and Autonomous Systems **114**, 134–143 (2019)
20. SepúLveda, D., Fernández, R., Navas, E., Armada, M., González-De-Santos, P.: Robotic aubergine harvesting using dual-arm manipulation. IEEE Access **8**, 121889–121904 (2020)
21. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., Berg, A.C.: Ssd: Single shot multibox detector. In: European Conference on Computer Vision, pp. 21–37 (2016). Springer
22. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems, pp. 91–99 (2015)
23. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 779–788 (2016)
24. Atherton, T.J., Kerbyson, D.J.: Size invariant circle detection. Image and Vision Computing **17**(11), 795–803 (1999)
25. Slotine, J.-J.E., Asada, H.: Robot Analysis and Control, 1st edn. John Wiley & Sons, Inc., New York, NY, USA (1992)
26. Kavraki, L.E., Svestka, P., Latombe, J.-C., Overmars, M.H.: Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE transactions on Robotics and Automation **12**(4), 566–580 (1996)
27. LaValle, S.M.: Rapidly-exploring random trees: A new tool for path planning (1998)
28. Jaillet, L., Cortés, J., Siméon, T.: Transition-based rrt for path planning in continuous cost spaces. In: 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2145–2150 (2008). IEEE
29. Karaman, S., Frazzoli, E.: Sampling-based algorithms for optimal motion planning. The international journal of robotics research **30**(7), 846–894 (2011)