To Harvest or Not to Harvest: Mapping and Decision-Making for a Selective Table Grape Harvesting Robot

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

Ruben Beumer, Leonardo Saraceni, Daniele Nardi, Duarte Antunes, *Senior Member, IEEE*, René van de Molengraft, and Thomas Ciarfuglia, *Member, IEEE*

***Abstract*—This letter focuses on robotic harvesting of delicate crops such as table grapes, featuring selective harvesting based on individual product properties. The robot detects grape bunches and estimates their positions and quality attributes. However, sensor limitations and occlusions affect data completeness and accuracy, reducing the cost-effectiveness of automated harvesting systems. Determining in real-time the optimal harvesting order in the presence of uncertainty is therefore important for enhancing efficiency and grape quality for growers and consumers. This task is challenging not only due to data uncertainty, but also due to the need to consider factors such as obstructive low-quality bunches. Existing literature often resorts to sub-optimal approaches such as selecting the first available crop. In contrast, we propose (i) a mapping and tracking method based on multiple viewpoints to enhance bunch information quality and (ii) a decision-making algorithm in a decision-tree with a recursive structure based on a constructed reachability graph derived from the map to optimize harvested quality and execution time sequentially.**

***Index Terms*—Computer vision, decision-making, multiple object tracking, optimization, precision agriculture, semantic mapping.**

Fig. 1. The table grape harvesting robot equipped with an Intel RealSense D435i camera on the wrist.

I. Introduction

ROBOTICS in precision agriculture is advancing to mitigate labor shortages, reduce operational costs, and enhance crop quality to meet global food demand sustainably. Recent developments have focused on integrating advanced sensing, mapping, and decision-making capabilities to improve the performance of autonomous systems in challenging agricultural environments. This letter addresses one such challenge: decision-making for selective harvesting in complex scenarios characterized by occlusions and uncertainty. Building on prior advancements in detection [1] and tracking [2], we propose a novel framework that balances quality-driven objectives with operational constraints.

We validate our approach on a table grape harvesting robot (Fig. 1), employing incremental map-building in the field to enable informed, optimal decision-making for harvesting. We use the Quality Estimation Module (QEM) [3] which extracts four key attributes relevant for harvesting decisions described in Sec. III-A1 and a LiDAR-based EKF localization system, both developed in the CANOPIES project [4]. The localization is accurate enough for global mapping and integration of camera data into the world frame.

Information incompleteness and uncertainty due to sensor limitations and occlusions are some of the main challenges for selective harvesting robots (e.g., see [5]). In this work, we address key limitations in existing selective harvesting systems by proposing a method that integrates multi-view mapping and quality estimation to enhance spatial and semantic information. It tracks and maps grape bunches while combining positional and

Received 31 January 2025; accepted 7 July 2025. Date of publication 19 August 2025; date of current version 4 September 2025. This article was recommended for publication by Associate Editor T. Kiyokawa and Editor O. Stasse upon evaluation of the reviewers' comments. This work was supported in part by EU Horizon 2020 research and innovation programme under Grant Agreement number 101016906 – Project CANOPIES, in part by the Sapienza University of Rome as part of the work for project H&M: Hyperspectral and Multispectral Fruit Sugar Content Estimation for Robot Harvesting Operations in Difficult Environments, Del. SA n.36/2022, in part by Project AGRITECH Spoke 9 - Codice progetto MUR: AGRITECH "National Research Centre for Agricultural Technologies" - CUPunder Grant CN00000022, of the National Recovery and ResiliencePlan (PNRR) financed by the EU "Next Generation EU", in part by the Netherlands Organisation for Scientific Research (NWO) under Grant 17626, and in part by IMEC-One Planet and other private parties. *(Ruben Beumer and Leonardo Saraceni contributed equally to this work.) (Corresponding author: Ruben Beumer.)*

Ruben Beumer, Duarte Antunes, and René van de Molengraft are with the Department of Mechanical Engineering, Eindhoven University of Technology, 5600MB Eindhoven, The Netherlands (e-mail: r.m.beumer@tue.nl; d.antunes@tue.nl; m.j.g.v.d.molengraft@tue.nl).

Leonardo Saraceni and Daniele Nardi are with the Department of Computer, Control and Management Engineering (DIAG) "Antonio Ruberti", Sapienza University of Rome, 00185 Rome, Italy (e-mail: saraceni@diag.uniroma1.it; nardi@diag.uniroma1.it).

Thomas Ciarfuglia is with the San Raffaele University of Rome, 00166 Rome, Italy (e-mail: thomas.ciarfuglia@uniroma5.it).

Digital Object Identifier 10.1109/LRA.2025.3600147

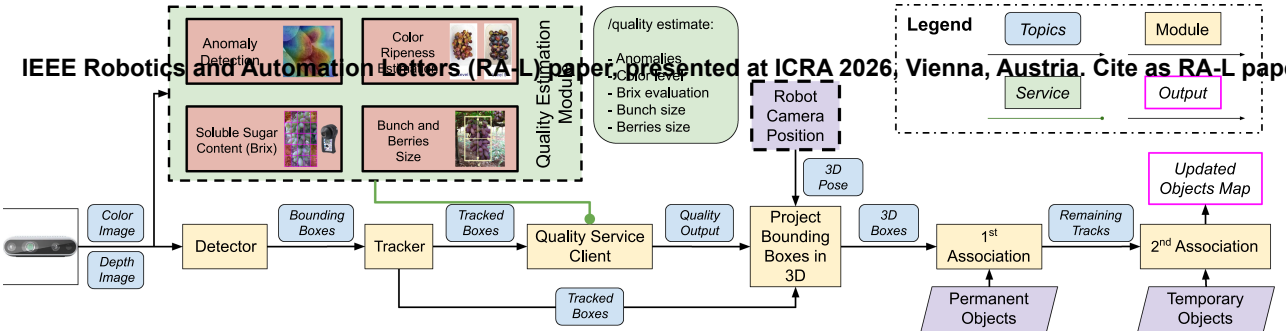


Fig. 2. Overview of the mapping pipeline. The pipeline integrates detection, tracking, and quality estimation to generate a 3D map of table grape bunches. The quality estimation module evaluates parameters such as anomalies, Brix degree, color, and size. Oriented bounding boxes (OBBs) are fitted to filtered point clouds, and the mapped bunches are updated through Kalman filter-based fusion across multiple viewpoints.

quality measurements, constructing a reachability graph to optimize the harvesting order. Unlike existing approaches, which often rely on heuristic decisions, our dynamic programming-based algorithm [6] optimizes both total harvested quality and motion execution time. Furthermore, we enable the implementation of quality prediction functions for multi-day planning.

Section II gives an overview of the work related to our research. In Section III we explain our method, followed by experimental results of applying it in Section IV. Concluding remarks and ideas for future research are given in Section V.

II. Related Work

A. Tracking and Mapping in Precision Agriculture

Robust tracking and mapping are critical for precision agriculture, where robots must localize and monitor multiple objects in dynamic, occluded environments. Recent approaches, such as AgriSORT [2], have demonstrated robustness in agricultural settings using lightweight tracking-by-detection frameworks to address challenges like occlusions and varying illumination. Similarly, NTrack [7] utilized optical flow and particle filtering to reliably track clustered objects, such as cotton bolls, achieving accurate detection and trajectory alignment. Other methods have incorporated multi-view 3D perception techniques to improve spatial reconstruction and tracking accuracy in unstructured agricultural scenarios [8]. Building on these advancements, we tackle the unique challenges of table grape harvesting, where dense clustering, varying occlusions, and the need for accurate quality estimation present significant complexities.

B. Decision-Making Algorithms and Applications

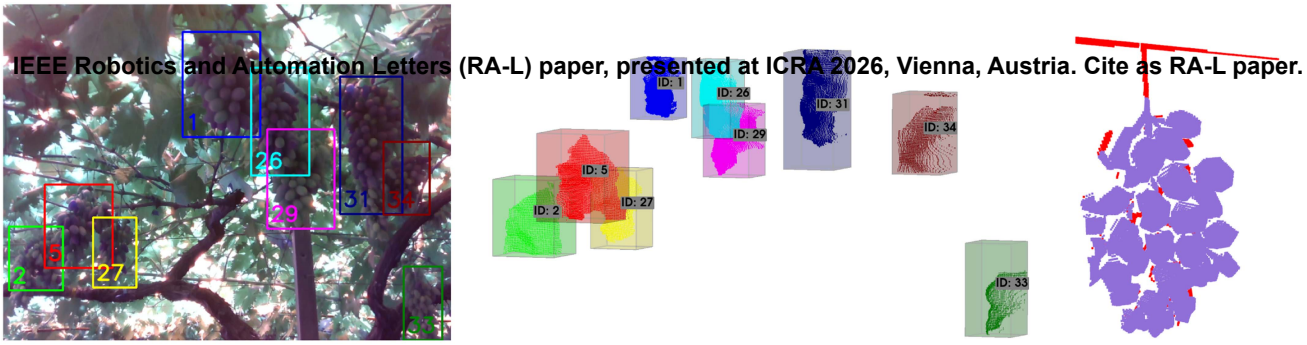
There are numerous decision-making applications for selective harvesting robots, primarily focusing on individual products or planning algorithms for the timing of specific operations [5], [9], [10]. Typically, product quality assessment informs independent one-at-a-time harvesting decisions. Based on these decisions, a path to harvest the selected bunches, e.g., by a robot arm, is planned and executed. Our aim is to determine the harvest order that maximizes the total harvested quality of the bunches and minimizes the execution time. This is a challenging problem when bunches occlude each other, leading to reachability dependencies: some bunches can only be reached after other bunches are removed. If the robot would not take this into account and try to harvest an unreachable bunch, it could damage the tree or other bunches. This type of information can be stored in a graph: a model of the world that allows for the organization, analysis, and visualization of geometric and semantic properties of objects [11]. To the best of our knowledge, researchers have not explored optimizing the product harvesting order while considering such reachability dependencies, which is particularly relevant for clustered products like table grapes or apples. This challenge is related to the well-known Travelling Salesman Problem (TSP) and has common features with TSP variants studied before.

The Selective TSP (also known as Orienteering Problem and closely related to the Tourist Trip Design Problem, e.g., see [12], [13]) considers location-dependent (but often time-independent) profits, which connects to the problem where the bunches' harvesting rewards depend on their quality. It usually considers a distance limit within which the total profit should be maximized by selecting a subset of the locations. An extended version of the Selective TSP with time-dependent profits is studied in [14]. Another related variant is the Prize Collecting TSP (PCTSP), which has no explicit distance limit but does favor faster paths.

Moreover, our case contains precedence constraints due to specific bunches limiting each other's reachability, a criticality faced by the TSP with Precedence Constraints (TSP-PC), also known as Sequential Ordering Problem, which is for example analyzed in [15], [16], [17], [18]. Our case is a combination of Prize Collecting TSP with precedence constraints and time-dependent profits, which pose a significant challenge. Nevertheless, by sequentially optimizing quality rewards and distance/time costs and leveraging the reachability graph, we propose a real-time solution for practical applications.

III. Methodology

The method we propose works as follows. While the robot navigates in the vineyard, it detects bunches and estimates their position and quality, summarizing these into a single reward value. The objective is to determine a harvesting order that maximizes reward while minimizing execution time. However, occlusions and limited sensor accuracy often result in incomplete and uncertain data. Since bunch quality affects harvesting and some bunches may block access to others, accurately estimating both quality and position is crucial. We therefore build a map by tracking bunches across multiple viewpoints. This enhances spatial completeness and improves accuracy by fusing measurements



(a) (b) (c)

Fig. 3. Visualization of tracking and 2D to 3D estimation results. (a) The camera image shows tracked grape bunches with unique IDs and 2D bounding boxes assigned by the tracker. (b) Corresponding 3D visualization of the mapped oriented bounding boxes (OBBs) with associated local point clouds. (c) Example of a point cloud filtered with DBSCAN. In red we represent the points discarded by DBSCAN, which do not belong to the grape bunch.

of the same bunches. This section first presents our mapping approach, followed by the decision-making process.

A. Mapping and Tracking

Our tracking and mapping framework integrates geometric and semantic properties, capturing the position and shape of each bunch and its quality attributes. The approach relies on the detection and tracking module developed by [2] and the Quality Estimation Module (QEM) [3]. A Kalman Filter (KF) estimates and updates the state of grape clusters by combining spatial measurements with quality indicators and uncertainties. The mapping pipeline is shown in Fig. 2.

1) State Representation: In this work, we represent each grape bunch as an oriented bounding box (OBB) in 3D, parametrized by its center $\boldsymbol{p}=\begin{bmatrix}p_x, p_y, p_z\end{bmatrix}^\top$, orientation $R \in SO(3)$, and lengths $\boldsymbol{l}=\begin{bmatrix}l_x, l_y, l_z\end{bmatrix}^\top$. The OBB $\mathcal{B}$ thus has the form:

$$\mathcal{B} = \left\{ \boldsymbol{r} \in \mathbb{R}^3 : \boldsymbol{r} = \boldsymbol{p} + R\boldsymbol{\ell}, \text{ with } \underset{i\in\{x,y,z\}}{\boldsymbol{\ell}_i} \in \left[-\frac{\boldsymbol{l}_i}{2}, \frac{\boldsymbol{l}_i}{2} \right] \right\}. \quad (1)$$

In addition to geometric properties, each bunch is characterized by quality attributes estimated by the QEM that we collect in a vector $\boldsymbol{q} \in \mathbb{R}^6$. These attributes are:

- **Anomalies** (a)**:** probability of presence of illness
- **Brix degree** (b)**:** floating-point measure of sugar content
- **Colour** (c)**:** ripeness on a scale from 0 to 5
- **Sizes** (l_x, l_y, l_z)**:** metric lengths of the OBB

Therefore, the state $\boldsymbol{x}$ of a bunch is modeled as

$$\boldsymbol{x} = \begin{bmatrix}\boldsymbol{p}^\top & \boldsymbol{\theta}^\top & \boldsymbol{l}^\top & a & b & c\end{bmatrix}^\top, \quad (2)$$

where $\boldsymbol{p}$ is the center, $\boldsymbol{\theta}=\begin{bmatrix}\theta_x, \theta_y, \theta_z\end{bmatrix}^\top$ the parametric orientation using Euler angles, $\boldsymbol{l}$ the OBB lengths, and a, b and c the quality parameters.

2) Multi-View 3D Estimation From 2D Measurements: The robot is equipped with an RGB-D camera that enables 3D point cloud generation from image data. We extract 2D detections of grape bunches (Fig. 3(a)) for each image along with their quality estimates from the QEM. Given camera intrinsics (f_x, f_y, c_z, c_y) and pixel coordinates (u_i, v_i) with corresponding depth d_i, the point $\boldsymbol{p}_i \in \mathbb{R}^3$ is obtained by:

$$\boldsymbol{p}_i = \begin{bmatrix}x_i \\ y_i \\ z_i\end{bmatrix} = \begin{bmatrix}(u_i + c_x)d_i/f_x \\ (v_i + c_y)d_i/f_y \\ d_i\end{bmatrix}, \quad (3)$$

in which x_i, y_i and z_i are the coordinates of the projected point in 3D. To construct an OBB for each detection, we first project all pixels belonging to the corresponding 2D bounding box into 3D space. The resulting 3D points may contain noise due to inaccuracies in the depth map or occlusions from leaves and branches. To mitigate this, we apply the DBSCAN [19] clustering algorithm as a pre-processing step on the point cloud to remove outlier points (Fig. 3(c)). We then fit an enclosing OBB to the remaining points in the filtered point cloud (Fig. 3(b)). We transform the OBB points from the local camera ($\boldsymbol{p}_c$) to the global world ($\boldsymbol{p}_w$) frame using the transformation matrix T_c^w derived from the robot's pose in the localization system:

$$\begin{bmatrix}\boldsymbol{p}_w \\ 1\end{bmatrix} = T_c^w \begin{bmatrix}\boldsymbol{p}_c \\ 1\end{bmatrix}. \quad (4)$$

3) Kalman Filter-Based Fusion: We track each object using a Kalman Filter (KF), which recursively fuses new observations. Given a prior state estimate $\hat{\boldsymbol{x}}_{k|k-1}$ and a measurement $\boldsymbol{z}_k$, the KF updates the state as $\hat{\boldsymbol{x}}_{k|k} = \hat{\boldsymbol{x}}_{k|k-1} + K_k\,(\boldsymbol{z}_k - H\,\hat{\boldsymbol{x}}_{k|k-1})$, where K_k is the Kalman gain and H is the measurement model that maps state variables to the observed OBB and quality parameters. The KF incorporates multiple viewpoints over time, reducing uncertainty and refining geometry and quality estimates.

4) Two-Phase Association and Object Confirmation: We employ a two-phase association process to handle data association, distinguishing between Temporary (o) and Permanent ($\mathcal{O}$) Objects. We classify all newly detected grape bunches as Temporary Objects to ensure that only consistently observed bunches are promoted to Permanent Objects, leading to a more stable and accurate map.

1) Permanent Objects: We first match detections to permanent (fully confirmed) objects by solving an assignment problem. We construct a cost matrix A using an IoU-based metric. The Hungarian algorithm is applied to find the optimal assignment. Let $\mathcal{D}$ be the set of detections and $\mathcal{O}$ the set of permanent

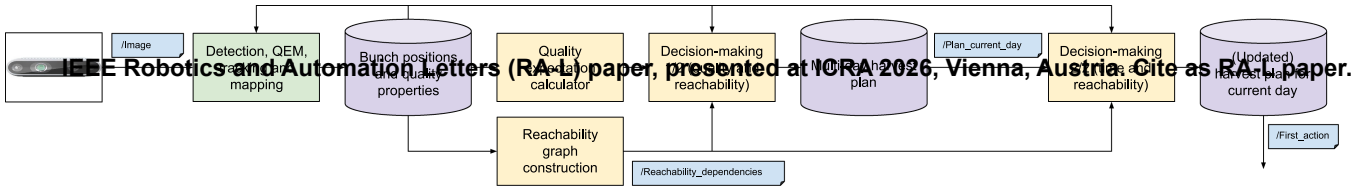


Fig. 4. Overview of the decision-making pipeline. The block 'Detection, QEM, tracking and mapping' is elaborated in Section III-A and Fig. 2. Based on the bunches' quality and position, we calculate the quality expectation and create the reachability graph. In the quality optimization step of the decision-making process, a multi-day harvest plan is made, indicating which bunches to harvest on which day. The harvesting order in the current day's plan is then optimized for execution time, still taking into account the reachability dependencies. The procedure is repeated after execution of the first action.

objects. The assignment π^* is a minimizer (of possibly multiple minimizers) of

$$\pi^* \in \arg\min_{\pi} \sum_{d\in\mathcal{D}} \sum_{o\in\mathcal{O}} A_{d,o}\pi_{d,o}, \tag{5}$$

subject to the constraints that each detection is assigned to at most one object and vice versa, with binary assignments $\pi_{d,o}\in\{0,1\}$. We try to assign every detection, but only if $A_{d,o} < \mathrm{IoU}_{\mathrm{lim}}$. We update assigned objects, while we predict unassigned permanent objects forward in time.

2) Temporary Objects: We repeat the association with the remaining detections for the temporary objects. We classify all the newly detected grape bunches as temporary. Assignments promote temporary objects to permanent once they have been matched for μ consecutive frames. If a temporary object remains unmatched for τ consecutive frames, it is discarded. Let μ_o be the count of consecutive matches of object o and τ_o the count of consecutive misses:

$$\text{if} \quad \mu_o > \mu \quad \Rightarrow \quad o \in \mathcal{O}, \tag{6a}$$

$$\text{if} \quad \tau_o > \tau \quad \Rightarrow \quad o \text{ discarded}. \tag{6b}$$

B. Decision-Making

We model the system to describe how actions (decisions) cause the state, based on the map with information about the bunches and robot, to change. A reward function scores action sequences and the proposed algorithm finds the best sequence. A schematic overview is shown in Fig. 4.

1) Model:

a) State and Bunch Information: The state $\boldsymbol{x}_k$ at step k is represented by the harvest day d_k, the robot arm's position information $\boldsymbol{p}_k$, and $\boldsymbol{h}_k$ with elements $h_{k,i}\in\{0,1\}$ indicating for any bunch $i\in\{1,\dots,n\}$ within the robot's reach whether it has already been harvested ($h_{k,i}=1$) or not ($h_{k,i}=0$):

$$\boldsymbol{x}_k = \begin{bmatrix} d_k & \boldsymbol{p}_k^\top & \boldsymbol{h}_k^\top \end{bmatrix}^\top. \tag{7}$$

Note that this is different from the individual bunch state in (2). Furthermore, we define $\boldsymbol{p}_i\in\mathbb{R}^3$ as the 3D position of the center of bunch i, $\boldsymbol{q}_{k,i}$ as its (expected) quality properties (see Section III-A1), and $\mathcal{R}_i$ as a set (with an a priori unknown size) of other bunches due to which it is not reachable. We define reachability of a bunch as the robot's ability to harvest it from its current position by only moving its arm, which depends on the robot and end-effector design. For the sake of simplicity, in this letter we represent the position information of the robot arm $\boldsymbol{p}_k$ by the 3D position of the end effector, but this could easily be extended. We use the sets $\mathcal{R}_i$ for $i\in\{1,\dots,n\}$ to create a graph representation for reachability dependencies (Fig. 5), which provides a core structure for the decision-making.

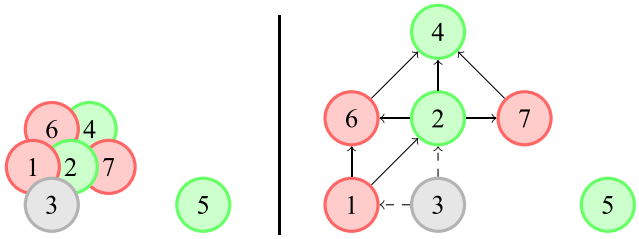


Fig. 5. Example of reachability graph representation. Green and red bunches indicate high and low quality respectively, and grey bunches have already been harvested. The arrows indicate reachability dependencies between them. In this example, $\mathcal{R}_6=\{1,2\}$ means the robot can reach bunch 6 only after picking bunches 1 and 2.

b) Actions: The action $\boldsymbol{u}_k$ at step k is given by $\boldsymbol{u}_k = \begin{bmatrix} u_{k,1} & u_{k,2} \end{bmatrix}^\top$ in which $u_{k,1}$ can take the following values: the index of the bunch to harvest; $u_s := -1$ if the robot should stop harvesting; or $u_w := -2$ if the robot should wait until the next day. Moreover, $u_{k,2}$ indicates whether to keep the collected bunch ($u_{k,2}=1$) or discard it ($u_{k,2}=0$). The set of possible actions at step k is given by

$$\mathcal{U}_k = \begin{cases} \left\{\begin{bmatrix} u_s & 0 \end{bmatrix}^\top\right\} \cup \mathcal{H}_k \cup \left\{\begin{bmatrix} u_w & 0 \end{bmatrix}^\top\right\} & \text{if } d_k < d_{\max} \\ \left\{\begin{bmatrix} u_s & 0 \end{bmatrix}^\top\right\} \cup \mathcal{H}_k & \text{if } d_k = d_{\max} \\ \left\{\begin{bmatrix} u_s & 0 \end{bmatrix}^\top\right\} & \text{otherwise,} \end{cases} \tag{8a}$$

in which $d_{\max}$ is the maximum number of days in the harvest plan and $\mathcal{H}_k$ is the set of actions to harvest the reachable bunches, defined by

$$\mathcal{H}_k = \left\{ \begin{bmatrix} i \\ 0 \end{bmatrix}, \begin{bmatrix} i \\ 1 \end{bmatrix} \mid h_{k,i} = 0 \wedge h_{k,j} = 1 \, \forall \, j \in \mathcal{R}_i \right\}. \tag{8b}$$

Hence, the set $\mathcal{U}_k$ contains the following actions: stop harvesting, harvest (and keep or discard) any reachable unharvested bunch, or wait until the next day (if there are more days left).

c) Update: The actions cause the state to change over time, according to

$$x_{k+1} = f(\boldsymbol{x}_k, \boldsymbol{u}_k) = \begin{cases} f_s(\boldsymbol{x}_k, \boldsymbol{u}_k) & \text{if } u_{k,1} = u_s \\ f_w(\boldsymbol{x}_k, \boldsymbol{u}_k) & \text{if } u_{k,1} = u_w \\ f_h(\boldsymbol{x}_k, \boldsymbol{u}_k) & \text{otherwise,} \end{cases} \tag{9a}$$

with

$$f_s(\boldsymbol{x}_k, \boldsymbol{u}_k) = \begin{bmatrix} k_{\max}+1 & \boldsymbol{p}_{\text{def}}^\top & \boldsymbol{h}_k^\top \end{bmatrix}^\top \tag{9b}$$

$$f_w(\boldsymbol{x}_k, \boldsymbol{u}_k) = \begin{bmatrix} d_k+1 & \boldsymbol{p}_{\text{def}}^\top & \boldsymbol{h}_k^\top \end{bmatrix}^\top \tag{9c}$$

$$f_h(\boldsymbol{x}_k, \boldsymbol{u}_k) = \begin{bmatrix} d_k & \boldsymbol{p}_{\boldsymbol{u}_k}^\top & \boldsymbol{h}_h^\top \end{bmatrix}^\top$$
$$\text{with } \boldsymbol{h}_h = \begin{bmatrix} h_{k,1}, \dots, h_{k,u_{k,1}-1}, 1, h_{k,u_{k,1}+1}, \dots, h_{k,n} \end{bmatrix}^\top \tag{9d}$$

in which $\boldsymbol{p}_{\text{def}}$ represents the robot arm's default position and $\boldsymbol{p}_{\boldsymbol{u}_k}$ the updated robot arm's position after harvesting bunch $u_{k,1}$, which can depend on whether the bunch is kept or discarded ($u_{k,2}$). This formulation shows that either the day d_k changes, or the boolean indicating that bunch $u_{k,1}$ has been harvested ($h_{k,u_{k,1}}$) is updated to 1. In this work, we assume that harvesting always succeeds as we only select reachable bunches; robotic handling is researched in parallel [4]. The reachability dependencies are also updated over time according to the (expected) development of bunches' sizes, which are calculated using (13) as explained below.

Note that by construction, the last action is always to stop harvesting. If this action is chosen before the horizon $k_{\max}$ is reached, the day is set to $d_{\max}+1$ according to (9b), so that the only possible next action will also be to stop harvesting (the latter case in (8a)).

2) Reward Function: The reward function $g(\boldsymbol{x}_k, \boldsymbol{u}_k)$ assigns a score to each state-action pair to compare different action sequences based on state $\boldsymbol{x}_k$ and action $\boldsymbol{u}_k$. For the reward, we consider both the estimated agronomic quality of the grape bunches and the cost of the execution time. Therefore, the reward function of a single action is:

$$g(\boldsymbol{x}_k, \boldsymbol{u}_k) = g_q(\boldsymbol{x}_k, \boldsymbol{u}_k) + g_t(\boldsymbol{x}_k, \boldsymbol{u}_k) \tag{10}$$

in which g_q is the agronomic quality score and g_t is related to the cost of the operation time.

a) Quality Reward: We model the quality reward function for a given state-action pair as:

$$g_q(\boldsymbol{x}_k, \boldsymbol{u}_k) = \begin{cases} 0 & \text{if } u_{k,1} \in \{u_s, u_w\} \\ -\epsilon & \text{if } u_{k,1} \geq 0 \wedge u_{k,2} = 0 \\ g_q^*(\boldsymbol{q}_{k,u_{k,1}}) & \text{otherwise,} \end{cases} \tag{11}$$

indicating that the reward is equal to the quality score of the harvested bunch ($g_q^*(\boldsymbol{q}_{k,u_{k,1}})$, explained below) if it is kept, $-\epsilon$ if it is discarded and 0 otherwise. The infinitesimally small ϵ is used to ensure that actions to stop or wait are considered slightly better than harvesting a bunch and discarding it without causing any advantage for the next steps. The quality score function $g_q^*(\boldsymbol{q})$ for a bunch depends on its quality properties $\boldsymbol{q} = \begin{bmatrix} a, b, c, l_x, l_y, l_z \end{bmatrix}^\top$ as mentioned in Section III-A1. Some properties, such as color, require a certain threshold to be considered sufficient. The size, on the other hand, can directly correspond to a continuous quality score scaling, since fruits are usually sold per weight unit. For the Brix value, we use a combination of both; there is a threshold, but higher values result in higher scores. Any detected or expected anomaly penalizes the quality score. In this work, we apply the following quality score function:

$$g_q^*(\boldsymbol{q}) = l_x l_y l_z z \frac{b}{b_t}(1-a)$$

$$\text{with } z = \begin{cases} -1 & \text{if } b < b_t \vee c < c_t \\ 1 & \text{otherwise} \end{cases} \tag{12}$$

in which l_x, l_y, and l_z are the width, height, and depth of the grape bunch (with $l_z = l_x$ assumed due to grape morphology), b the Brix value, b_t the Brix threshold, c the color level, c_t the color threshold and a the penalty for anomalies between 0 (if no anomalies) and 1. The auxiliary variable z is used to ensure that only bunches that meet the Brix degree and color level thresholds are awarded a positive quality score. The most important property is that it is desirable to harvest this bunch if and only if $g_q^*(\boldsymbol{q}) > 0$ if there would be no other costs, bunches, opportunities to harvest, etc.

Furthermore, the recursive function f_q represents the expected quality development of a bunch over time, given by:

$$f_q(\boldsymbol{q}_{k,i}, d) = \begin{cases} f_q^*(f_q(\boldsymbol{q}_{k,i}, d-1)) & \text{if } d > 1 \\ f_q^*(\boldsymbol{q}_{k,i}) & \text{if } d = 1 \end{cases} \tag{13}$$

in which d is the number of days to wait and $f_q^*(\boldsymbol{q}_{k,i})$ represents the expected development of the quality properties when waiting one day.

b) Time Reward: To calculate the time reward, we use the position information of the robot arm $\boldsymbol{p}_k$ in the state (see (7)) to compute the duration of the robot movement to complete an action. The cost function associated with this time is implemented as a negative reward, given by:

$$g_t(\boldsymbol{x}_k, \boldsymbol{u}_k) = -\alpha_t t(\boldsymbol{x}_k, \boldsymbol{u}_k) \tag{14}$$

in which α_t is a constant that assigns a cost per second, and $t(\boldsymbol{x}_k, \boldsymbol{u}_k)$ is the execution time that we model as:

$$t(\boldsymbol{x}_k, \boldsymbol{u}_k) = \begin{cases} t_m(\boldsymbol{x}_k, \boldsymbol{u}_k) + t_h + t_r(\boldsymbol{x}_k, \boldsymbol{u}_k) & \text{if } u_{k,1} \geq 0 \\ t_r(\boldsymbol{x}_k, \boldsymbol{u}_k) & \text{otherwise,} \end{cases} \tag{15}$$

in which the different components are the required times to:

- move the arm from the current position to the position to harvest the bunch ($t_m(\boldsymbol{x}_k, \boldsymbol{u}_k)$),
- cut the peduncle (t_h),
- move the arm to place the bunch in the box or discard it, or to return to the default position when stopping or waiting until the next day ($t_r(\boldsymbol{x}_k, \boldsymbol{u}_k)$).

3) Action Sequence Selection Algorithm: For every possible state, a limited number of actions is available, and by assuming a finite time horizon, actions are also limited over time. The main factors determining the number of options are the number of bunches and time steps considered (days or weeks). We use a branched tree structure to describe all possible action sequences and apply a depth-first search (DFS, for reasons explained below) to find the sequence that leads to the optimal result. For every decision point, we consider all options by recursively finding the best solution to the tail of the problem. The total reward function, which also shows the recursive structure, is defined by:

$$s_k(\boldsymbol{x}_k) = \max_{\boldsymbol{u}_k \in \mathcal{U}_k} g(\boldsymbol{x}_k, \boldsymbol{u}_k) + s_{k+1}(f(\boldsymbol{x}_k, \boldsymbol{u}_k)) \tag{16}$$

with $s_{k_{\max}}(\boldsymbol{x}_{k_{\max}}) = 0$ in which $k_{\max} = n + d_{\max}$ is the horizon. There is no (theoretical) maximum for $d_{\max}$, although making it longer than the usual harvest time window is unnecessary. It can even be beneficial to limit $d_{\max}$ to an even shorter period, as explained below.

a) Managing the computational complexity: To enable real-time applicability, the two key factors are computation time and storage requirements. For speed and reusability, each branch can be harvested only once, with the option to either keep or discard it, resulting in n actions with two choices each. The harvest plan spans up to $d_{\max}$ days, allowing for $d_{\max}-1$ identical 'wait' actions. Hence, the theoretical maximum number of action sequences, even excluding early termination, is $2^n \times \frac{(n+d_{\max}-1)!}{(d_{\max}-1)!}$, which quickly makes this problem intractable. The problem resembles a special variant of the TSP, which is NP-hard. Therefore, we propose simplifying assumptions tailored to robotic selective harvesting, splitting the problem into smaller subproblems solvable in real-time with a DFS algorithm. DFS is memory-efficient, storing only the currently being explored action sequence, but computation time remains a concern. We already limit the amount of options as we only consider bunches that are reachable either immediately or subsequently, preserving performance while limiting n. Besides, the reachability dependencies limit the number of possible actions by construction. In addition, we limit the complexity as follows:

i) The factor 2^n can be eliminated by basing the decision to keep or discard a bunch solely on its expected quality (and not on a time reward) at harvest. This is achieved by replacing (8b) with:

$$\mathcal{H}_k = \left\{ \begin{bmatrix} i \\ u \end{bmatrix} \mid h_{k,i} = 0 \wedge h_{k,j} = 1 \, \forall \, j \in \mathcal{R}_i \right\} \tag{17}$$

with $u = 1$ if $g_q^*(\boldsymbol{q}_{k,i}) > 0$ and $u = 0$ otherwise.

ii) Sequential optimization of quality rewards and time costs. First, without (further) loss of optimality, quality optimization can then be performed per cluster of interconnected bunches in the reachability graph (e.g, $\mathcal{S}_1^q = \{1,2,3,4,6,7\}$ and $\mathcal{S}_2^q = \{5\}$ in Fig. 5). To determine the set $\mathcal{H}_k$, we still use (17), but only for $i \in \mathcal{S}_l^q$ when considering subgroup l, and respectively we apply (16) with horizon $k_{\max} = n_{\mathcal{S}_l^q} + d_{\max}$ in which $n_{\mathcal{S}_l^q} = |\mathcal{S}_l^q|$. Furthermore, as we only focus on quality and not on time in this optimization, we replace (10) by $g(\boldsymbol{x}_k, \boldsymbol{u}_k) = g_q(\boldsymbol{x}_k, \boldsymbol{u}_k)$. Secondly, we consider the bunches to be harvested per day for time-optimization. We consider subgroups that are based on the union of the action plans per day from the different subgroups from the quality-optimization. E.g., the subgroup considering the bunches to be harvested on day 2 could be $\mathcal{S}_2^t = \{1,2,5\}$, in which bunches 1 and 2 are from $\mathcal{S}_1^q$ and bunch 5 is from $\mathcal{S}_2^q$. In this step, we can replace (8a) by $\mathcal{U}_k = \mathcal{H}_k$ and to determine the set $\mathcal{H}_k$, we still use (17), but only for $i \in \mathcal{S}_l^t$ when considering subgroup l. Again, we apply (16) but now with horizon $k_{\max} = n_{\mathcal{S}_l^t} = |\mathcal{S}_l^t|$. Furthermore, here we replace (10) by $g(\boldsymbol{x}_k, \boldsymbol{u}_k) = g_t(\boldsymbol{x}_k, \boldsymbol{u}_k)$. It should be noted that, because of optimizing sequentially, the value of α_t no longer influences the decision (as long as $\alpha_t > 0$).

iii) We can limit the maximum number of days ($d_{\max}$) in the harvest plan using a receding horizon approach. Accurate prediction of several weeks is difficult as uncertainty increases with $d_{\max}$. Besides, bunch quality follows a concave curve [20], [21], and hence their summed quality curve is also concave, even if their peaks do not align in time. Thus, if delaying harvest for two weeks is optimal, a shorter delay (e.g., three days) is likely also beneficial, which means that considering a shorter horizon results in the same decision.

b) Single action execution: The optimal action at time k is determined by maximizing (16). Due to the problem's recursive structure, the entire sequence can be stored. However, in practice, it is often preferable to apply only the first action and recompute the solution at each subsequent time step. This approach accounts for a shifting horizon and accommodates using newly available measurements that may be more accurate than the initial data due to removed occlusions after harvesting and differences between the modeled and actual development of the bunches.

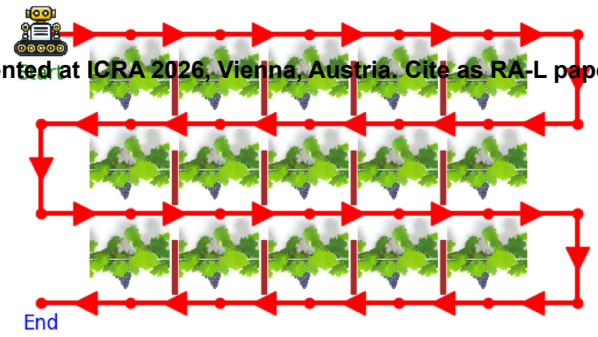


Fig. 6. S-shaped path employed to test the mapping algorithm.

IV. Experiments

In this section, we apply our algorithm to an example based on data from real-life experiments. This scenario demonstrates that mapping and tracking using multiple viewpoints, while accounting for reachability dependencies in the harvesting plan, can enhance the harvesting robot's performance in terms of total harvested quality.

A. Experimental Setup

We tested the system on a real robot platform composed of a tracked Alitrak DCT-300P base, a modified PAL Robotics Tiago++ manipulator, and a RealSense D435i RGB-D camera (Fig. 1). For quantitative mapping validation, however, we used a customizable Unity-based simulator [22] that replicates the robot and vineyard structure, providing access to ground truth to evaluate grapes position accuracy. We tested the decision-making component on structured real-world data collected from experiments in the vineyard, preserving realistic geometry and occlusion patterns while enabling repeatable evaluation. The robot navigates through the field with the camera oriented towards the vines, following a predefined trajectory (Fig. 6) designed to simulate standard monitoring conditions. The robot acquires RGB-D frames at 1 Hz while moving along the path resulting in uniformly sampled viewpoints. Localization is provided by a LiDAR-based algorithm specifically developed for vineyards [4].

B. Mapping Validation

We validated our mapping approach (Section III-A) using a controlled simulation environment that reproduced realistic vineyard scenarios. By varying field size and vine configurations, we systematically assessed mapping performance under different numbers of bunches and levels of occlusion.

We focus our evaluation on the following metrics: Mean Absolute Error (MAE) for position estimates, maximum and

TABLE I
MAPPING EXPERIMENTS RESULTS WITH THE VARYING OF THE SIZE OF THE FIELD CONFIGURABLE IN THE SIMULATOR AS NUMBER OF ROWS AND COLUMNS. MAE, MAXIMUM ERROR, MINIMUM ERROR, AND STD ARE EXPRESSED IN METERS

Size	MAE	Max. Err.	Min. Err.	STD	FP	# Objs.
2×2	0.12	0.53	0.01	0.10	1	38
3×3	0.10	0.53	0.00	0.11	6	134
4×4	0.10	0.76	0.01	0.11	13	287
5×5	0.10	0.93	0.01	0.11	19	395
6×6	0.13	0.67	0.00	0.09	25	505

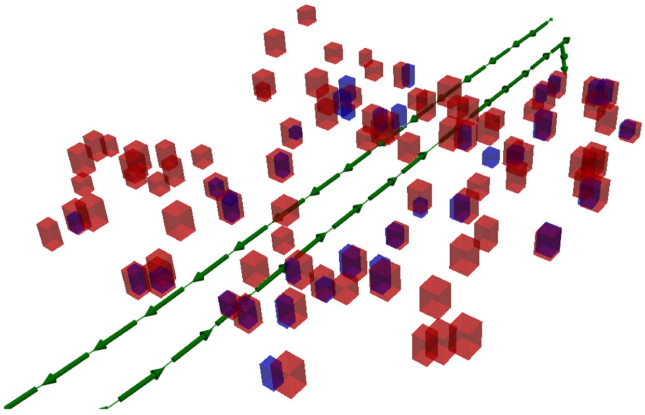


Fig. 7. Visualization of the observed data (blue boxes) and ground truth (red boxes) with the robot's path in green. Most objects along the path are successfully mapped, while missed detections primarily occur at greater distances or above the robot, where they remain outside its field of view.

minimum observed errors, standard deviation (STD) of errors, and the number of false positives counted when the position error exceeds 0.2 m. We also tracked the total number of detected objects to assess scalability.

In our evaluation, we apply thresholds $\text{IoU}_{\text{lim}} = 0.3$, $\mu = 4$, and $\tau = 4$ for object permanence and discarding unmatched objects described in Section III-A4. In the KF, we model the initial state uncertainty with covariance matrix $P = \text{diag}(10^{-2})$. Process noise covariance $Q = \text{diag}(10^{-4})$ and measurement noise covariance $R = \text{diag}(10^{-2})$ were selected to balance noise suppression with the confidence in the model.

As shown in Table I, the MAE remained consistently low, confirming robust position estimates. The small STDs reflect stable tracking performance, and although some scenarios produced slightly higher maximum errors due to occlusions, these remained limited to a small number of instances. Notably, the absolute number of false positives increased with field size, because the number of transient artifacts grows with the navigation time. Despite this, the two-phase association method remained effective in filtering out most of them, maintaining a reasonable level of false positives relative to the total number of detected objects. We provide a visualization of an example result of the mapping in Fig. 7.

C. Decision-Making Results

To arrive at the results described in this section using our proposed algorithm for decision-making, bunches are considered to be within reach by the robot arm when they are within 1.5 m from the robot's camera. For this simple example, they are assumed

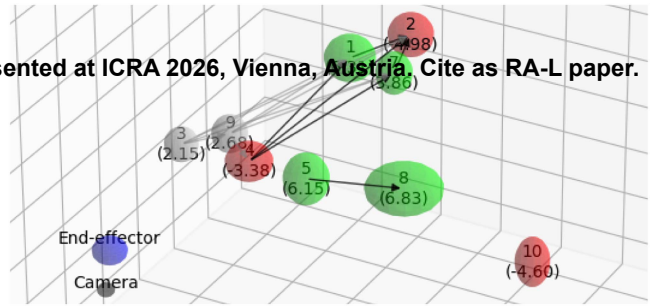


Fig. 8. Visualization of the reachability graph for the example used to test the decision-making algorithm. The ellipsoids represent the bunches with their IDs and quality scores. Green, red and grey respectively represent high quality, low quality and already harvested bunches. The arrows indicate reachability dependencies between bunches.

to limit each other's reachability when the straight line from the camera to the bunch's center intersects another bunch. If actual physical harvesting is considered, this will be extended to take into account the design of the robot, its end-effector and path planning algorithm. The parameters used in the quality score function described in (12) are $c_t = 3.5$ (as only the color levels 0–3 indicate that the bunch is ripe) and $b_t = 15$. The quality prediction function $f_q^*(\boldsymbol{q}_{k,i})$ in (13) used in the tests is:

$$f_q^*(\boldsymbol{q}_{k,i}) = \begin{bmatrix} \min(a_{k,i} + a_{\text{inc}}, 1) \\ b_{k,i} + b_{\text{inc}} \\ \max(c_{k,i} + c_{\text{inc}}, 0) \\ l_{x,k,i} + l_{x,\text{inc}} c_{k,i} \\ l_{y,k,i} + l_{y,\text{inc}} c_{k,i} \end{bmatrix} \tag{18}$$

with incremental values $a_{\text{inc}} = 0.1$ if $c_{k,i} > 0$ and $a_{\text{inc}} = 0$ otherwise, $b_{\text{inc}} = 1/7$, $c_{\text{inc}} = -0.3$, and $l_{x,\text{inc}} = l_{y,\text{inc}} = 1$ mm. This description of the quality expectation function has not been validated with real data, as it is beyond the scope of the study. Therefore, in our experiments we assume that predicted and actual developments are identical. In practice, this assumption may hold approximately if the function is fitted to real data, with discrepancies addressed by executing only the first action and recalculating the plan using updated information. Lastly, we used $\alpha_t = 0.01$ in (14), $t_h = 1.0$ s and motion execution times are calculated as if they are proportional to the 3D Euclidean distance with an end-effector speed of 25 cm/s for t_m and t_r in (15).

In Fig. 8, the local map (the part within reach of the current position of the robot) of the bunches in the real field with their quality scores and reachability dependencies (indicated by the arrows) is shown. The numerical data of their estimated properties are also given in Table II. We let the algorithm compute the harvest sequence for harvest plans of a different number of days. For $d_{\max} = 1$, the algorithm outputs the harvesting sequence $[3, 9, 5, 8, 4, 7, 1, u_s]$. All bunches in this plan are kept except for bunch 4, which is discarded to enable access to high-quality bunches 1 and 7 located behind it. Bunches 2 and 10 are not harvested, and bunch 6 lies outside the local map. This strategy achieves a total quality score of 27.97, and after subtracting the time costs a total reward of 27.46. In comparison, a baseline approach that harvests only the directly reachable high-quality bunches (3 and 5) yields a quality score of 8.30. Including the additional subsequent

TABLE II
BUNCHES AND THEIR ESTIMATED QUALITY PARAMETERS AND REACHABILITY DEPENDENCIES FOR THE EXAMPLE USED TO TEST THE DECISION-MAKING ALGORITHM

ID	Brix degree	Color	Size (mm)	Quality	Obstructed by
1	19.15	2	181×150	6.31	3, 4, 9
2	18.28	4	174×135	-4.98	1, 3, 4, 7, 9
3	18.31	3	135×97	2.15	
4	18.25	4	143×135	-3.38	9
5	18.35	3	198×128	6.15	
7	18.30	3	171×108	3.86	3, 4, 9
8	18.22	3	161×218	6.83	5
9	18.31	3	146×103	2.68	3
10	18.35	4	197×97	-4.60	

reachable bunches (9 and 8) in this baseline approach raises the quality score to 17.81, which remains significantly lower than the proposed method. With $d_{\max} = 10$, the computed harvest sequence is $[u_w,u_w,u_w,u_w,u_w,u_w,u_w,3,9,4,1,u_w,7,u_w,5,2,8,10,u_s]$, while with $d_{\max} = 14$, it is $[u_w,u_w,u_w,u_w,u_w,u_w,u_w,3,9,4,1,u_w,7,u_w,u_w,u_w,5,8,u_w,2,u_w,10,u_s]$. In these plans, there is enough time to wait for all bunches to ripen, so they are all kept and the total harvested quality score is respectively 60.15 and 61.81. After subtracting time costs, they result in total rewards of respectively 59.35 and 61.02. In comparison, for a robot arm or human that could harvest any bunch at any time (i.e., without reachability constraints), the maximum total quality score with $d_{\max} = 14$ would be 63.67.

V. CONCLUSION AND RECOMMENDATIONS FOR FUTURE WORK

This letter presents a mapping and decision-making algorithm for a table grape harvesting robot, addressing the challenges of uncertain and incomplete information. The method uses multiple viewpoints and a graph containing reachability dependencies between bunches to optimize the harvesting order, maximizing the total quality score and outperforming single-image-based approaches. Our approach is flexible enough to be applied to different types of crops given a working detection algorithm and an appropriate quality expectation function tailored to the specific crop. Future work includes active perception with occlusion-aware graphs, active viewpoints planning strategies to ensure higher coverage, stochastic modeling of crop development, heuristic-based search methods to find optimal harvesting sequences, and human-robot collaboration for inaccessible bunches to further improve efficiency and performance.

ACKNOWLEDGMENT

The Synergia project is organized and led by Wageningen University and Research in close cooperation with Next Food Collective as well as the Universities of Delft, Twente, Eindhoven, and Nijmegen. The authors have declared that no competing interests exist in the writing of this publication.

REFERENCES

[1] T. A. Ciarfuglia, I. M. Motoi, L. Saraceni, M. Fawakherji, A. Sanfeliu, and D. Nardi, "Weakly and semi-supervised detection, segmentation and tracking of table grapes with limited and noisy data," *Comput. Electron. Agriculture*, vol. 205, 2023, Art. no. 107624.

[2] L. Saraceni, I. M. Motoi, D. Nardi, and T. A. Ciarfuglia, "AgriSORT: A simple online real-time tracking-by-detection framework for robotics in precision agriculture," in *Proc. 2024 IEEE Int. Conf. Robot. Automat.*, 2024, pp. 2675–2682.

[3] T. A. Ciarfuglia, I. M. Motoi, L. Saraceni, and D. Nardi, "Can robots 'taste' grapes? Estimating SSC with simple RGB sensors," 2024, *arXiv:2412.20521*.

[4] "EU canopies project: A collaborative paradigm for human workers and multi-robot teams in precision agriculture systems," 2021. [Online]. Available: https://canopies.inf.uniroma3.it/

[5] G. Kootstra, X. Wang, P. M. Blok, J. Hemming, and E. van Henten, "Selective harvesting robotics: Current research, trends, and future directions," *Current Robot. Rep.*, vol. 2, no. 1, pp. 95–104, 2021, doi: 10.1007/s43154-020-00034-1.

[6] R. E. Bellman, *Dynamic Programming*. Princeton, NJ, USA: Princeton Univ. Press, 2010.

[7] M. A. Al Muzaddid and W. J. Beksi, "NTrack: A multiple-object tracker and dataset for infield cotton boll counting," *IEEE Trans. Automat. Sci. Eng.*, vol. 21, no. 4, pp. 7452–7464, Oct. 2024.

[8] N. Hu et al., "Segmentation and tracking of vegetable plants by exploiting vegetable shape feature for precision spray of agricultural robots," *J. Field Robot.*, vol. 41, no. 3, pp. 570–586, 2024.

[9] B. Harel, Y. Edan, and Y. Perlman, "Optimization model for selective harvest planning performed by humans and robots," *Appl. Sci.*, vol. 12, no. 5, 2022, Art. no. 2507.

[10] V. Rajendran et al., "Towards autonomous selective harvesting: A review of robot perception, robot design, motion planning and control," *J. Field Robot.*, vol. 41, no. 7, pp. 2247–2279, 2024.

[11] J. Senden et al., "Multi-hypothesis tracking in a graph-based world model for knowledge-driven active perception," *IEEE Robot. Automat. Lett.*, vol. 8, no. 9, pp. 5934–5941, Sep. 2023.

[12] D. Gavalas, C. Konstantopoulos, K. Mastakas, and G. Pantziou, "A survey on algorithmic approaches for solving tourist trip design problems," *J. Heuristics*, vol. 20, no. 3, pp. 91–328, Mar. 2014, doi: 10.1007/s10732-014-9242-5.

[13] G. Laporte and S. Martello, "The selective travelling salesman problem," *Discrete Appl. Math.*, vol. 26, no. 2, pp. 193–207, 1990.

[14] "Analysis of the selective traveling salesman problem with time-dependent profits," vol. 31.

[15] Z. Ahmed and S. Pandit, "The travelling salesman problem with precedence constraints," *Opsearch*, vol. 3, pp. 299–318, 2001.

[16] N. Ascheuer, M. Jünger, and G. Reinelt, "A branch & cut algorithm for the asymmetric traveling salesman problem with precedence constraints," *Comput. Optim. Appl.*, vol. 17, no. 1, pp. 61–84, 2000.

[17] F. C. J. Lokin, "Procedures for travelling salesman problems with additional constraints," *Eur. J. Oper. Res.*, vol. 3, no. 2, pp. 135–141, 1979.

[18] Y. Salii, "Revisiting dynamic programming for precedence-constrained traveling salesman problem and its time-dependent generalization," *Eur. J. Oper. Res.*, vol. 272, no. 1, pp. 32–42, 2019, doi: 10.1016/j.ejor.2018.06.003. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0377221718305265

[19] K. Khan, S. U. Rehman, K. Aziz, S. Fong, and S. Sarasvady, "DBSCAN: Past, present and future," in *Proc. 5th Int. Conf. Appl. Digit. Inf. web Technol.*, 2014, pp. 232–238.

[20] E. L. Avanzini, A. F. M. Cawley, J. R. Vera, and S. Maturana, "Comparing an expected value with a multistage stochastic optimization approach for the case of wine grape harvesting operations with quality degradation," *Int. Trans. Oper. Res.*, vol. 30, no. 4, pp. 1843–1873.

[21] J.-C. Ferrer, A. M. Cawley, S. Maturana, S. Toloza, and J. Vera, "An optimization approach for scheduling wine grape harvest operations," *Int. J. Prod. Econ.*, vol. 112, no. 2, pp. 985–999, 2008.

[22] PaleBlue AS, "VR farming environment specification," The CANOPIES Project, 2014. [Online]. Available: https://www.iri.upc.edu/research/webprojects/canopies/D2.4_VR%20Farming%20Environment%20Specification.pdf