INF-82000

Multi-robot approach for a visual sensing task in greenhouse horticulture

João Otávio Araujo da Silva
Information Technology
September 2023

WAGENINGEN
UNIVERSITY & RESEARCH

“Multi-robot approach for a visual sensing task in greenhouse horticulture”

João Otávio Araujo da Silva

Registration number: 951220017020

Supervisor

dr. JR (João) Pereira Valente

A thesis submitted in partial fulfilment of the degree of Master of Science at Wageningen University and Research Centre, The Netherlands.

September 2023

Wageningen, The Netherlands

Thesis code number: *INF-82000*

Study program: *MSc Biosystems Engineering*

Educational institute: *Wageningen University and Research Centre*

Chair group: *Information Technology Group*

Disclaimer

This report is written by a student of Wageningen University as part of a masters programme and is executed under supervision of the Information Technology Group. This report is not an official publication of Wageningen University or Wageningen UR. The content of this report is not the opinion of Wageningen University or Wageningen UR.

Use of information from this report is for own risk and it is advised to check this independently before the information is used.

Wageningen University is never liable for the consequences that result from use of information from this report.

Information Technology Group

Wageningen University, De Leeuwenborch, building 201, 6th floor

Hollandseweg 1, 6706 KN Wageningen, The Netherlands

T: +31 (0) 317 489876 / +31 (0) 317 481403

Abstract

Unmanned Aerial Vehicles (UAVs) have been progressively more used in agricultural tasks. The technology of these drones, although evolving, still faces issues, such as battery autonomy, that halter their applicability in more situations. A recent field of study investigates the potential of drone swarms to compensate for these limitations of UAVs through task sharing and efficient allocation of resources. However, because it is a new line of research, many applications are still lacking investigation for design choices. This study aims at exploring the opportunities and challenges of using drone swarms for counting fruits in a tomato greenhouse, comparing design choices and swarm size. Three exploration approaches, namely Lawnmower Strategy (*LMS*), Completely Random Walk (*CRW*) and Reinforced Random Walk (*RRW*), will be compared along three levels of swarm size with 1, 2 and 3 agents each. The experiment was carried out in a virtual simulation environment created with Unity3D. Comparisons were done based on the metrics of Root Mean Squared Error (RMSE), simulation time and the progression of the discovery ratio (DR) of tomatoes. Results indicated that a *LMS* with 3 agents was the most efficient strategy for fruit counting, based on the time needed to complete the task: 63 seconds. In conclusion, although promising at reducing mission time, further research is needed to better explore the potential of drone swarms to count tomatoes in a complex and confined space of greenhouses.

Keywords: Unmanned Aerial Vehicle (UAV), drone swarm, reinforced random walk, fruit counting.

Contents

1 Introduction

Yield estimation is an important component in the planning and management of crop cycles. It can contribute to the long-term economic planning by predicting the gross income on the moment of selling, as well as it can improve the daily management of inputs (e.g., water, nutrients) by estimating more precisely what is needed by the crops. Prediction of yield may also improve logistics by preparing the grower for the harvest (Nuske et al., 2014).

Although relevant, yield estimation can be labour-intensive, expensive, and unprecise if not done correctly. Most traditional methods consist of combining historical information, such as weather data, with manual sampling to calculate a crop's yield. The sampling methods are usually destructive and biased, since a small portion of the crop must be collected and calculations are then extrapolated to the whole area. Visual yield estimates are capable of increasing the sampling area and decrease the time needed, while being a non-destructive method (Nuske et al., 2014).

In order to achieve higher accuracy, visual yield estimation needs large amount of data. For that matter, automated data collection in moveable platforms has gained popularity in the agricultural sector. Modern Unmanned Aerial Vehicles (UAV's) are highly mobile and relatively cheap when compared to other means of remote data gathering, such as airplanes or satellites (Goudarzi et al., 2019; Karion et al., 2013).

The combination of UAV's and visual sensing has been extensively researched to perform plant phenotyping tasks, such as yield estimation. Some of the challenges of this technique include achieving high communication and perception levels in complex environments to avoid obstacles and ensure effective navigation (Arafat et al., 2023). Hence, research has focused on developing autonomous navigation systems. Regarding data management, the large amount of information captured over various perspectives and time may pose a challenge for data processing. The choice of analysis technique and interpretation impacts the accuracy and usefulness of UAV data acquisition (Liang et al., 2023).

Other constraint, in particular, of smaller, low-altitude UAV's, is their low battery and computing capacity (Arafat et al., 2023). Commercial drones, which have a flight autonomy of around 20 – 30 min, cause limitations to the volume of information acquired over a single flight (Biczyski et al., 2020). More recently, strategies that consist in using multiple drones simultaneously have been receiving attention from the scientific community because of their potential to enable longer flight missions by subdividing a large target measuring space over different agents.

These multi-drone approaches, otherwise called swarm drone robotics, are guided by swarm intelligence principles which target the design of systems that are fault tolerant, scalable, and flexible (Dorigo et al., 2014). Moreover, multi-robot approaches aim to tackle the need for sophisticated drones of single-robot missions by substituting them for multiple simpler robots (Nemitz et al., 2018). This is especially relevant in confined spaces where large-scale UAV's with higher battery capacity are not fit for the task.

Automated methods of data collection using UAV's, which can navigate in every direction, mostly depend on the construction of a 3D model of the environment. The exploration strategy

determines the efficiency with which this 3D map can be reconstructed (Palazzolo & Stachniss, 2018). Next-best-view planning algorithms focus on creating an exploration path in real-time by generating velocity commands to position the sensor on the next best viewpoint by considering the already explored space of all agents (UAV's) and their current field of view, minimizing the uncertainty of the succeeding measurements (Palazzolo & Stachniss, 2018). In a vertically grown crop, such as tomato, where the observation of a point of interest (e.g., the fruits) might be partially occluded by part of the plant (e.g., stem or leaves), information-gain might benefit the path making by taking measurements of the object of interest from different angles or sides of the crop row. In addition, such exploration strategies need to account for other hard constraints, such as avoiding collisions, or soft constraints, such as time or battery saving, to optimize the pathing.

2 Objective and research questions

2.1 Problem statement

The combination of drone swarms and visual sensing for environment exploration has been attempted from a nadir (top-down) point of view in open agricultural fields (Albani et al., 2018; Albani, IJsselmuiden, et al., 2017; Albani, Nardi, et al., 2017; Carbone et al., 2018, 2022; Trianni & López-Ibáñez, 2015). A common approach for navigation in such tasks is to use coverage path planning (CPP) algorithms, which are mostly combined with UAV's. Although CPP has been applied to ground robots, applications in agricultural context are still limited (Monica et al., 2019; Naazare et al., 2022; Yang et al., 2023). Implementing a similar technique for vertically grown crops in commercial greenhouses involves new challenges such as dealing with spatial constraints, close range observations, high incidence of obstructions and a horizontal point of view of the target area.

2.2 Objective

The present work aims at addressing a fruit counting task to a drone swarm in a tomato greenhouse environment. The objective is to investigate the performance that different swarm sizes and exploration approaches have in tackling this task.

2.3 Research questions

Main question: What are the opportunities and challenges of using drone swarms and computer vision techniques for a tomato detection task?

Sub-questions:

A. How can a base map for vertically grown crops be defined to be representative of the crop and easily navigable by the UAV's?
B. What adjustments are needed for the state-of-the-art exploration algorithms to be applicable for vertically grown crops?
C. Is there a performance improvement of swarm robotics over single-robot approaches for tomato counting based on visual sensing?
D. Is there a performance improvement of an exploration algorithm over coverage path planning strategies?

2.4 Research outline

To fulfil the objective of this project, an experimental set-up must be designed by combining drone swarm, visual sensing, and exploration algorithm. The first step will be to define the map where agents will navigate. It should be representative of the tomato row, capable of holding information, coordinating drone movement, and simple implementation. For that, a literature research will be done over the maps being used in similar studies.

After a map is defined, literature research will continue to identify what exploration algorithms are suitable for drone swarms. An exploration strategy that allows implementation in the chosen map must be selected and adapted with features found in literature research.

When sub-questions A and B are answered, an experiment can be designed and implemented in a virtual environment to generate results that will answer sub-questions C and D. The methodology of those experiments is thoroughly explained in chapter 4.

With all sub-questions answered, a discussion will be elaborated comparing the results obtained in the work with those of others. Explanations about the outcomes of the present work will be done based on literature research.

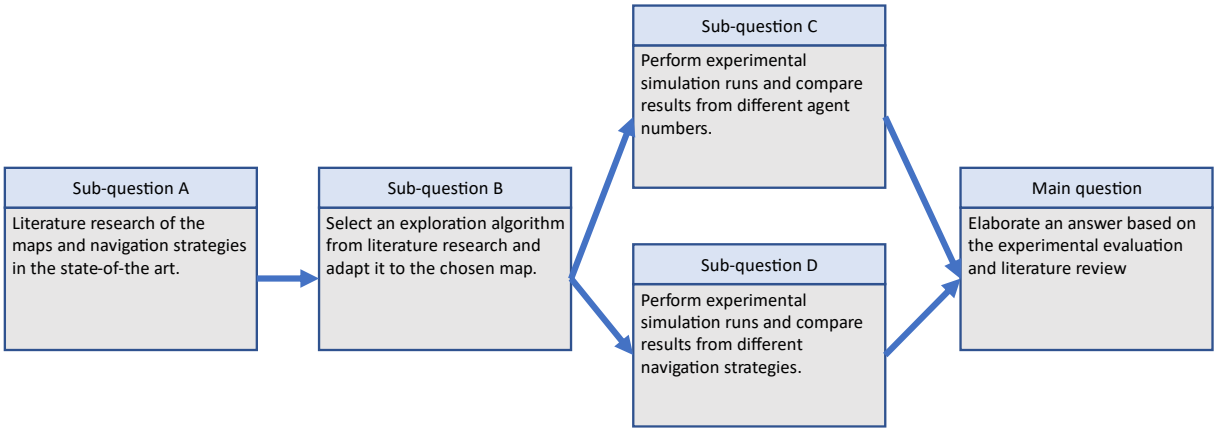


Figure 1: Flow of the research outline.

3 Literature review: state-of-the-art

3.1 Computer vision in fruit counting

Machine vision or computer vision refers to the use of algorithms and image processing techniques to automatically interpret and extract information from digital images. In the context of agriculture, computer vision can be employed on tasks such as disease recognition, plant phenotyping and fruit detection (Mavridou et al., 2019).

In tomato farming, fruit counting is essential for yield estimation. Manual counting is a laborious and costly task that can be inaccurate in a tomato greenhouse, a complex environment where leaf obstructions and double counting often occur (Egi et al., 2022). Automating this task with computer vision can improve its efficiency while decreasing the labour demand (Egi et al., 2022).

Since the advance of computer vision research, many algorithms were design with the purpose of tackling different challenges. Mask R-CNN, for instance, can detect objects as well as the pixels that compose them, enabling more precise estimation of their shape and dimensions. This model has been used for detecting and estimating the size of low-density crops, such as potatoes and lettuce (Machefer et al., 2020). For tomatoes, Mask R-CNN has been used for counting and classifying fruits based on their growth stage (Fawzia Rahim & Mineno, 2022). Studies showed that the algorithm is also capable of exclusively counting tomatoes placed in the foreground of a tomato row, by implicitly learning object depth (Afonso et al., 2020).

YOLO is another object detection model commonly used for tomato counting. It was designed to be used for real-time detection, which makes it appropriate for video footage. Its most outstanding characteristic is the speed of detection (Du, 2018). Liu et al. (2020) used a modified version of YOLO, denominated YOLO-Tomato, where the traditional rectangular bounding boxes were substituted for circular bounding boxes to better adapt to the tomatoes shape. Ge et al. (2022) used a YOLO based network to detect and track tomatoes and tomato flowers. Lawal (2021) also used YOLO-Tomato to detect tomatoes in a complex environment, where obstructions and illumination variation were common.

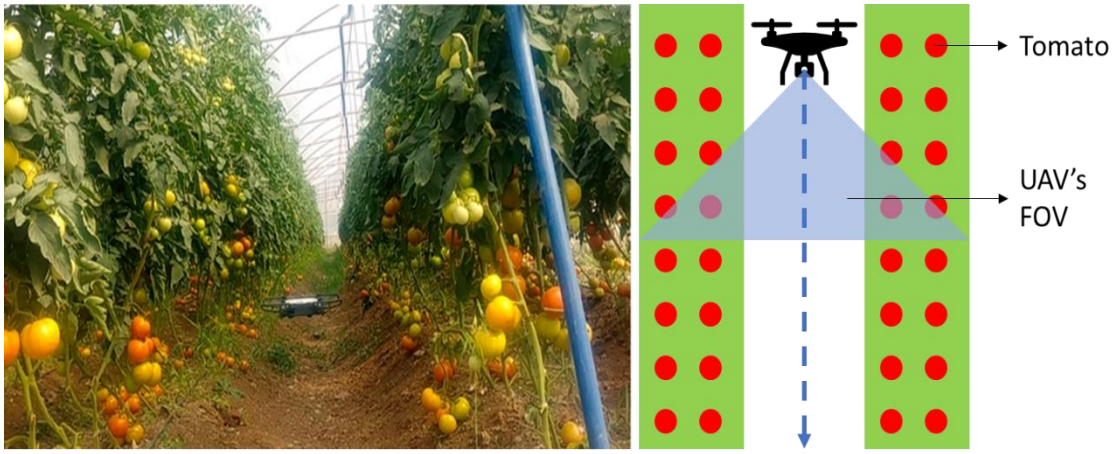


Figure 2: Set-up of an experiment using YOLO for tomato detection. In this work, the agent path was a straight line between two tomato rows, allowing image capturing of both rows simultaneously (Egi et al., 2022).

Other popular choice of object detection algorithm for agricultural related tasks is Faster R-CNN, which combines a Region Proposal Network (RPN), responsible for proposing bounding boxes to be examined, and a deep convolutional network from its predecessor Fast R-CNN. This allows the network to detect and classify objects quickly and precisely (Ren et al., 2015). The network was used to count and classify tomatoes according to their maturity stage (Widiyanto et al., 2021). In other agricultural research, Gao et al. (2020) and Fu et al. (2020) made use of this network for detecting and classifying apples in an orchard, an environment that occasionally presented obstruction of fruits. Carbone et al. (2022) combined Faster R-CNN with drone swarms for a weed detecting task.

3.2 Swarm robotics

UAVs allow fast and cost-efficient monitoring of large areas, making them useful tools for the agriculture industry, which is one of their biggest adoption leaders (Skobelev et al., 2018; Valente et al., 2020). While most solutions involving the use of UAVs for agricultural tasks are designed for single drones, multi-drone approaches enable efficient use of resources (e.g., time, battery) that result in reduced mission time by reallocating and sharing tasks between the agents (Skobelev et al., 2018). These multi-agent approaches are frequently referred to as swarms, fleets or flocks.

Ankit et al. (2021) utilized a UAV swarm controlled by Robot Operating System (ROS) to estimate yield and detect drought stress in a 3D simulated farm. Their approach consisted of using visual measurements of the farm to solve a routing problem for allocating the agents.

Albani, IJsselmuiden, et al. (2017) simulated weed detection with a drone swarm and concluded that their approach was viable for mapping and monitoring. By performing multiple visits from different agents, they were able to obtain robust measurements.

Carbone et al. (2022) also used swarm robotics to detect weeds in a simulated environment. Similarly to Albani, IJsselmuiden, et al. (2017), the agents navigated through the simulation using a map divided in squared cells, while keeping the same height during the entire mission. The navigation of agents was determined by selecting the next-best-viewpoint based on the highest uncertainty reduction that a cell would get from having more measurements being performed in it.

In another study about drone swarm in weed detection, the path planning consisted of a reinforced random walk where agents would be directed to points of interest based on the placement of beacons that attract agents (Albani, Nardi, et al., 2017). This concept, similar to many concepts used in swarm robotics, is inspired by collective behaviours in nature, such as bees and birds (Carbone et al., 2018).

Saffre Fabrice and Hildmann (2022) simulated swarm robotics over a map with hexagonal cells. In this work, the authors recommended that, in future works, the path planning should account for priority locations to be set based on data acquired during the mission. The paths were planned deterministically according to information in the surroundings from where measurements were taken, but a stochastic approach, which accounts for probabilities of different events, should be considered in the future.

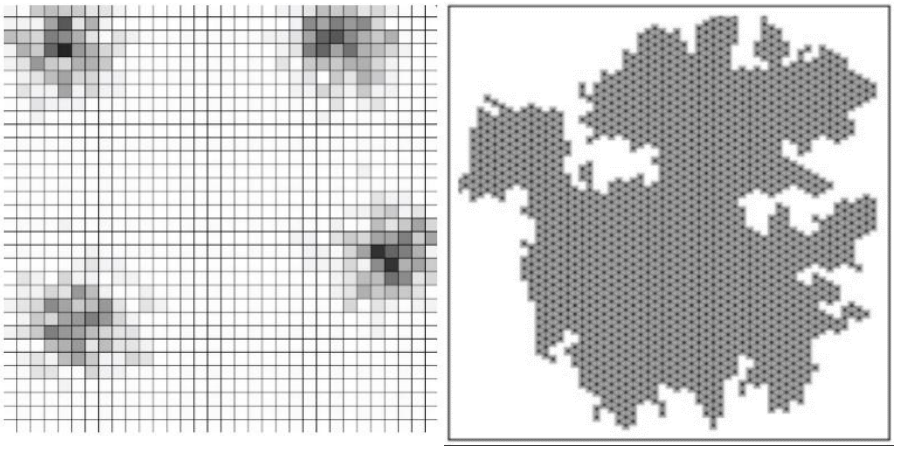


Figure 3: Squared cell map (left) and hexagonal cell map (right) used in drone swarm missions (Albani, IJsselmuiden, et al., 2017; Saffre Fabrice and Hildmann, 2022).

Dutta et al. (2021) investigated the state-of-the-art of current multi-robot approaches for precision agriculture and highlighted the three main challenges being path planning, security and integrity of information, and energy efficiency. The authors also state that the use of swarms for information gathering is still at a prototype stage.

3.3 Path Planning

The planning required to cover an unknown area is called coverage path planning (CPP). CPP can be expressed as a mathematical function where coverage must be maximized, while the travelled length or time to achieve it are minimized. The two main methods to solve such problem are frontier-based methods and sampling-based methods (X. Zhou et al., 2020).

Yamauchi (1997) introduced frontier-based method as an approach for exploring space based on selecting boundaries of unexplored space as targets for the next step in the trajectory. This facilitates the mapping of large spaces with separate areas, at the expense of longer times for full exploration (Selin et al., 2019).

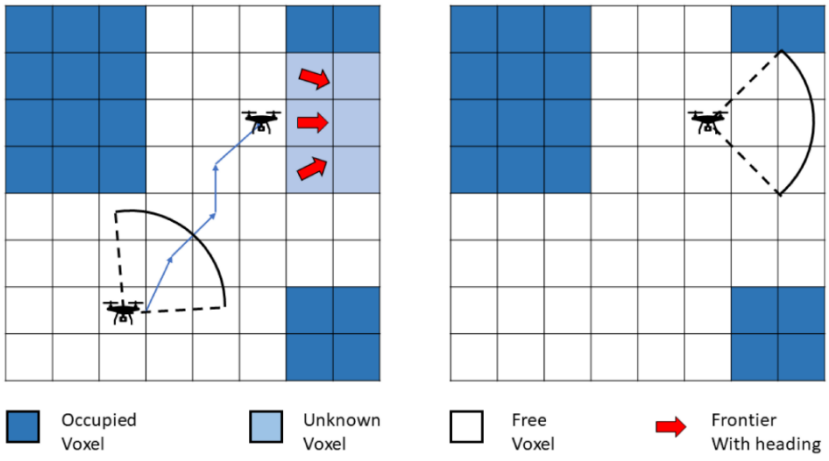


Figure 4: Exemplification of frontier-based exploration approach (Lu et al., 2020).

Sampling-based methods calculate the information gain from performing multiple feasible trajectories, selecting the one which will maximize the gain, while lowering the costs of a cost-utility function (Kingston et al., 2018; Y. Wang et al., 2023). Next-Best-View (NBV) is one of the most popular sample-based exploration methods (Y. Wang et al., 2023; X. Zhou et al., 2020). Initially developed for 3D reconstruction, NBV creates a scene from the viewpoints captured, then proceeds to select the next viewpoints form the current scene model in an iterative process (Lauri et al., 2020). This method may, however, find problems with local minima and get stuck when mapping large areas (X. Zhou et al., 2020).

In swarm-robotic approaches, path planning methods can be improved by leveraging the ability of parallel-monitoring from multiple agents (Carbone et al., 2022). The authors proposed a decentralized strategy, flexible to the number of UAVs, based on reinforced random walks (RRW), which prioritizes regions with relevant information. In this study, RRW was combined with Information Gain (IG) to measure the uncertainty reduction gained from exploring all possible target regions. IG was calculated based on an uncertainty model from detection errors of a Convolutional Neural Network (CNN). This approach was used in previous works (Albani et al., 2019; Albani, Nardi, et al., 2017) for weed mapping where the input data were NADIR images. However, approaches that use decentralized path planning methods, such as RRW, based on image detection from horizontally acquired images still lack research.

4 Methodology

To enable answering the research questions, a workflow was designed for agents (UAVs) to perform a visual counting task of a target (tomatoes) based on a coverage path planning. To enable that, the architecture consisted of three main blocks: (i) exploration approach, (ii) object detection algorithm and (iii) simulation environment.

The simulation environment is where all the visual part of the experiment would be handled. The tomato plants, the environment and the agents must be rendered in this environment. It should allow movement of the agents, randomization of the environment and extraction of information through capture of images to be used as input for object detection. Unity3D, a game development software which is also suitable for modelling virtual reality (S. Wang et al., 2010), was selected to render this environment.

The object detection algorithm would perform the task of extracting information from the environment in the form of predicted bounding boxes of virtual tomato models. This information was input for the coverage path planning and for the Root Mean Squared Error (RMSE) metric. Faster R-CNN was used as the baseline model for object detection.

Finally, the exploration approach would dictate how the agents should move. It involved designing the map with the boundaries to where agents could navigate, the logic behind the movements and the coverage path planning that would decide to where agents should move every step. For that, Reinforced Random Walk (*RRW*) was selected to be compared with other baseline coverage path planning approaches such as the Lawnmower and a Complete Random Walk strategies.

4.1 Exploration approach

4.1.1 Map and UAV navigation

A map was used to guide the navigation of the UAV agents and store information of what was explored. To allow simple 2D movement of the agents, the map was designed as a 2D plane divided in cells, which is a popular choice for visual detection tasks with drone swarms in agricultural environments (Albani et al., 2018; Albani, Nardi, et al., 2017; Carbone et al., 2022; De Rango et al., 2017). It is comprised of *Nc* x *Nr* cells of dimensions *Cs* x *Cs* and was positioned on the *xy* plane alongside the tomato row.

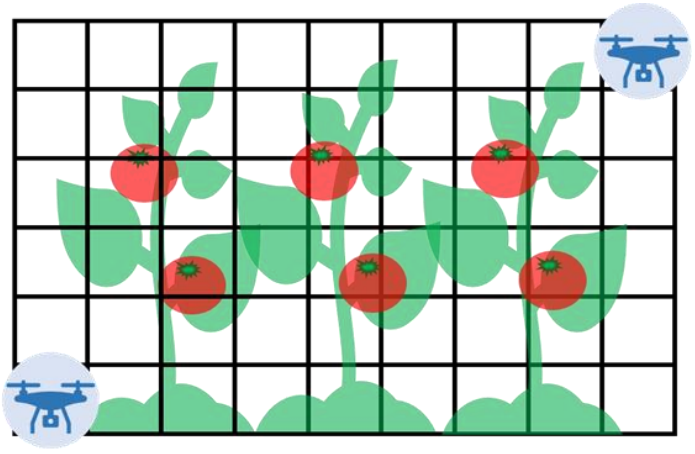


Figure 5: Representation of the map used to navigate agents across the simulated tomato row.

The distance assumed between agents and the plant row (d_{row}) was 0.7 m, because it would be within the boundaries of the row spacing of 1.2 m from commercial greenhouse tomatoes (Amundson et al., 2012), while being distant enough from the row to allow a clear visualization of the fruits. It was also assumed that the field-of-view angle (FOV_{angle}) of the agents' camera would be 82.6° across the diagonal, same as a DJI Tello UAV. Based on d_{row} and FOV_{angle}, the point-of-view (*POV*) that the agents would have of the plant row is a squared region with side length (POV_{side}) approximately equals to 0.869 m, based on the following equation:

$$POV_{side} = \sin 45° * 2\left(d_{row} * \tan\left(\frac{FOV_{angle}}{2}\right)\right) \quad (1)$$

Because agents move from cell to cell and the size of the field-of-view is low due to the close-range, a 3 x 3 grid of cells would allow longer displacement at each simulation step, while also having a central cell from which the agent could perform an orthogonal measurement. It was, therefore, determined that the area observed by the UAV camera would be a grid of 3 x 3 cells with side (*Cs*) equals to a third of POV_{side}, which is approximately 0.290 m.

The number of columns (*Nc*) and rows (*Nr*) were enough to allow visualization of all the plants in the row, without extrapolation. For 15 plants with approximately 2 m height and 0.5 m spacing between plants (Amundson et al., 2012), *Nc* = 26 and *Nr* = 11, resulting in 286 total cells.

Each cell was capable of holding essential information for navigation and counting, such as its coordinates, whether it was already visited by any agent and the measurements performed in the cell. For organization, a class called Measurement was created to store the number of tomatoes counted and the confidence score of the detection.

Table 1: Variables stored by cells.

Name	**Type**	**Information**
WasMeasured	bool	True if it was visualized and measured by any agent
WasVisited	bool	True if it was visited by any agent
IsTargeted	bool	True if targeted by any agent to be visited next
PotentialTomatoes	bool	True if cell potentially presents tomatoes
CoordID	int, int	Integer coordinates of the cell in relation to their position in the x-axis [0, Nc] and the y-axis [0, Nr]
CoordLength	float, float	Simulation world coordinates, in meters, of the centre of the cell
ActiveFlag	bool	True if a flag was placed in the cell
MeasurementList	List<Measurement>	List with all the measurements made in the cell
BestMeasurement	Measurement	Measurement with the best score

Agents' movements are based on cells, moving from the center of a cell to the center of another. They are, however, always at a distance d_{row} from the entire map, thus from the cells.

Which means that they are positioned orthogonally to the center of the cells, 0.7 m distant from it.

Agents may move vertically, horizontally and diagonally towards cells that are adjacent to the cell where they are currently positioned. These directly adjacent cells are in a region called *vicinity 1* (*V1*). If none of the cells on *V1* are accessible, agents will move to a cell in *V2* and if none of the cells on *V2* are accessible, agents will move to a cell in *V3*, continuing the loop until an available cell is found.

The agents' point-of-view (*POV*) is always a 3 x 3 cell grid, because the distance between agents and tomato row never changes. This means that the *POV* comprehends the cell where the agent is positioned (visiting) plus the neighboring cells (Figure 6).

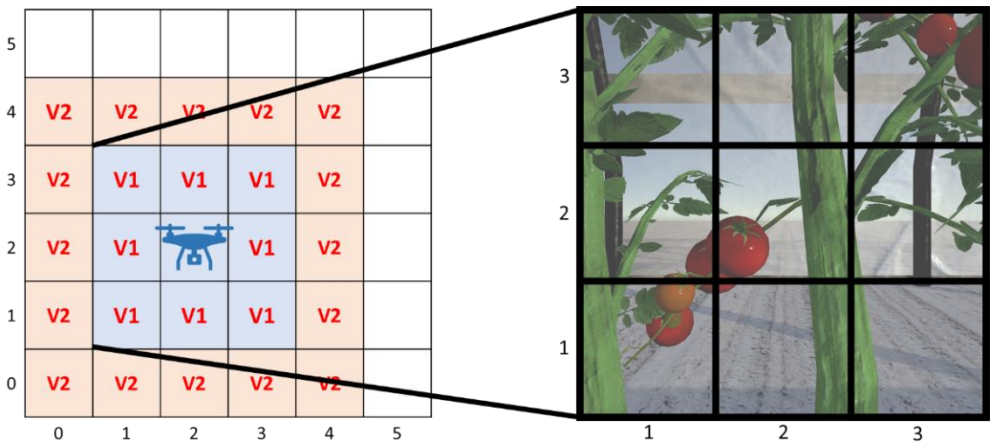


Figure 6: Cells on the first vicinity (V1) of the agent, to where the agent will attempt to move primarily, and on the second vicinity (V2), to where the agent will attempt to move if V1 cells are unavailable (left). The agent's POV (right).

When agents take a snapshot to perform counting, all cells within their *POV* are marked as *measured* (*M*), including the cell where the agent is currently positioned, which is marked as *visited* (*V*).

To perform a measurement, agents first take a snapshot of all 9 cells within their *POV* (i.e., the cell where it is positioned plus the 8 cells surrounding that). The snapshot image is used as input for the object detection model, which outputs a list of bounding boxes for each tomato in the *POV*. Using the coordinates of the bounding boxes, the number of tomatoes located in each cell is estimated, because the coordinates of each cell are known. The count of tomatoes in each cell is saved in the map.

It is important to highlight the difference between the status *measured* and *visited* for cells. Visited cells are those which had been visited by an agent, meaning that an agent was positioned orthogonally to its central point. Measured cells are those which had been caught in the agents *POV*, thus being attributed a measurement according to the output of the object detection algorithm.

Visited cells cannot be visited again, for any exploration approach. However, any cell might be measured up to 9 times, from all the possible viewpoints where it is visible in the

drone's *POV* (Figure 8). This design choice was taken based on the assumption that when a drone visits a cell and takes a snapshot, any other snapshot taken from the same position would be redundant, as it would be the same from any other one taken from that position, thus incentivizing the agents to visit other cells.

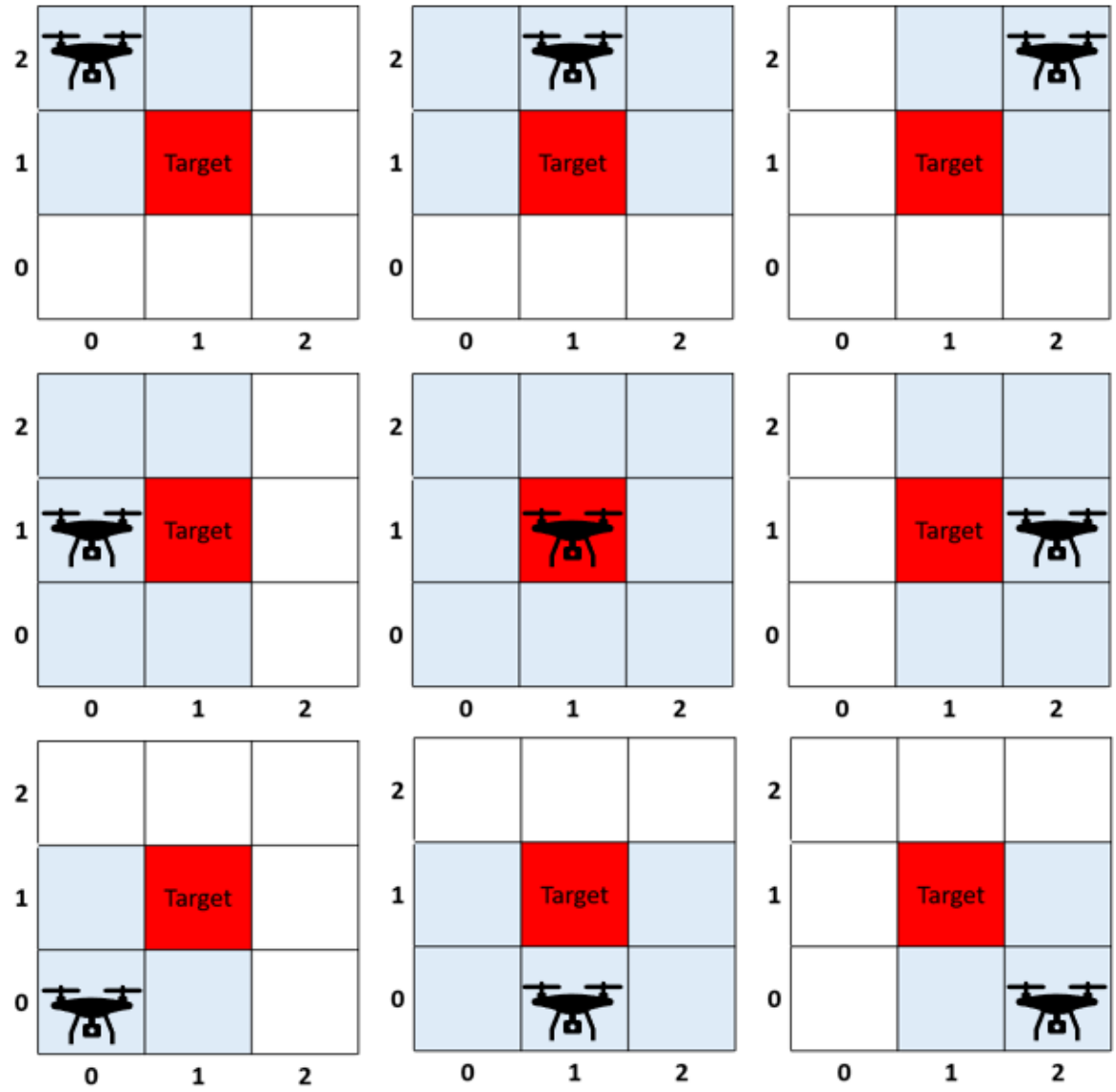


Figure 7: Schematic representation of how an UAV can measure the same target (red) cell from 9 different viewpoints due to its 3 x 3 cell POV (blue). The UAV is only visiting the target cell when it is standing in the position [1, 1].

If a measurement is done multiple times in a cell and the number of tomatoes counted differs, a flag will be placed in that cell. Flags attract agents to make more measurements of the same cell from different viewpoints. This attraction increases the bias of an agent to move towards the attraction points, by adding more weight to the vector that will dictate the agent's next movement. The assumption is that different tomato counting results of a certain cell from measurements performed from different viewpoints indicate that there might be tomatoes hidden by other tomatoes or plant structures.

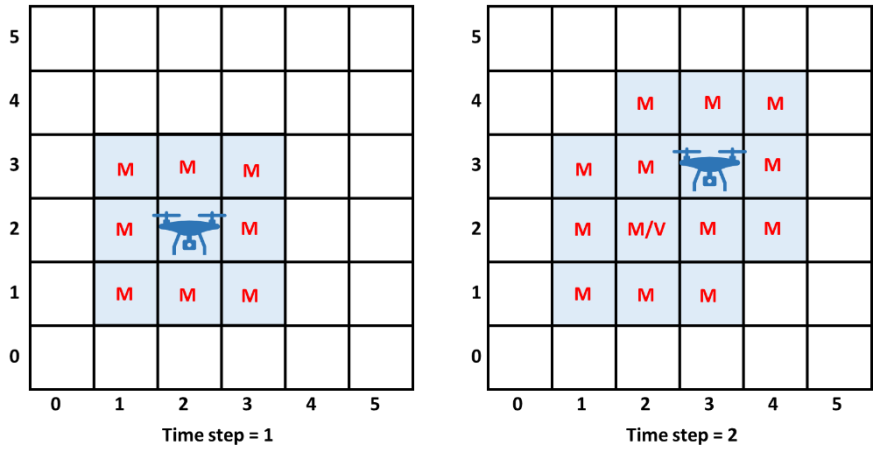


Figure 8: An example of an agent that performs a measurement, moves to cell [3, 3] and finally performs another measurement. Cells [2, 3], [2, 2], [3, 3] and [3, 2] were measured twice.

4.1.2 Baseline algorithms

Three strategies were tested to control the movement of agents across the map, which are Lawnmower Strategy (*LMS*), Completely Random Walk (*CRW*) and Reinforced Random Walk (*RRW*). For this work, *RRW* is the target of research, while *LMS* and *CRW* are the baseline algorithms that will be used for comparison.

Two requirements were settled for every navigation strategy, being: (i) every cell must be measured at least once; (ii) none of the cells are allowed to be visited twice. The second requirement ensures that there would be no unnecessary measurements. As the snapshots are made in a simulation without noise affecting the image quality, multiple snapshots from the same position yield the same input image.

LMS consists of placing agents on fixed start positions and having them move sideways, visiting one cell at a time, until reaching the border of the map, then switching to the adjacent row. Because of that, every different run of *LMS* results in the same path, exploration time and number of steps. Lawnmower navigation is a popular choice for agricultural environment exploration using UAVs, even in multiple-robot approaches (Aydoğan, 2018; Duan et al., 2017; P. Hu et al., 2018; Sousselier et al., 2015).

In *CRW*, agents start in the same position, but the next target location for every agent is completely randomized, until all cells are measured at least once.

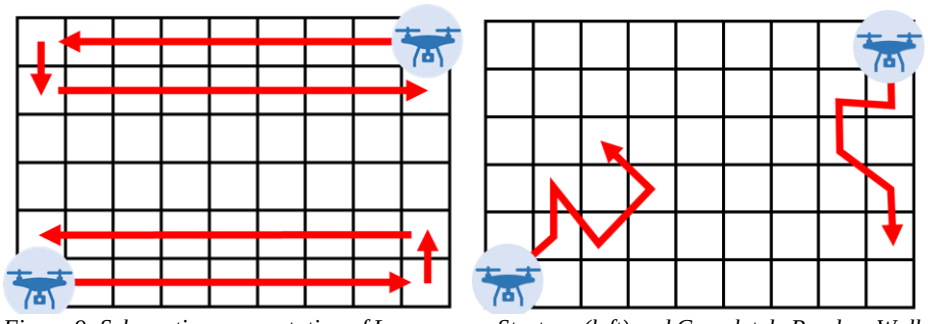


Figure 9: Schematic representation of Lawnmower Strategy (left) and Completely Random Walk (right) for UAV navigation.

4.1.3 Reinforced Random Walk algorithm (RRW)

In *RRW*, whenever an agent moves, the next target location is selected based on a weighted random selection. A bias vector dictates which cells had higher weight and thus what cells have higher chance of being selected.

There are attraction factors that direct the bias vector towards them, and repulsion factors that direct the bias vector against them. This strategy was based on the work of Albani et al., (2017), which used the concepts of marking points of interest on the map that would function as pheromones to attract other agents, while repulsing agents from each other for better map coverage. The same approach was used by other authors from the field of drone swarms (Carbone et al., 2022; Cimino et al., 2016).

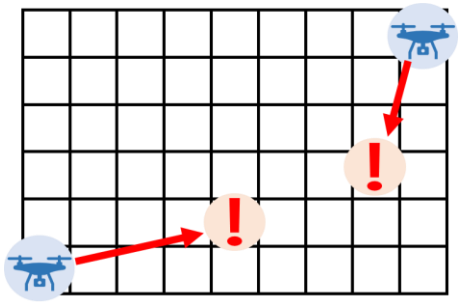


Figure 10: Schematic representation of Reinforced Random Walk strategy for UAV navigation. The red exclamation mark icons represent points of attraction that influence the bias of the direction to where UAVs will move.

In the experimental simulations, every run (i.e., repetition) followed a routine which consisted of a first phase that would be executed once and a second phase, specific for each navigation strategy, with steps that would be repeated until either all cells were measured, or all planned steps were done for *LMS*.

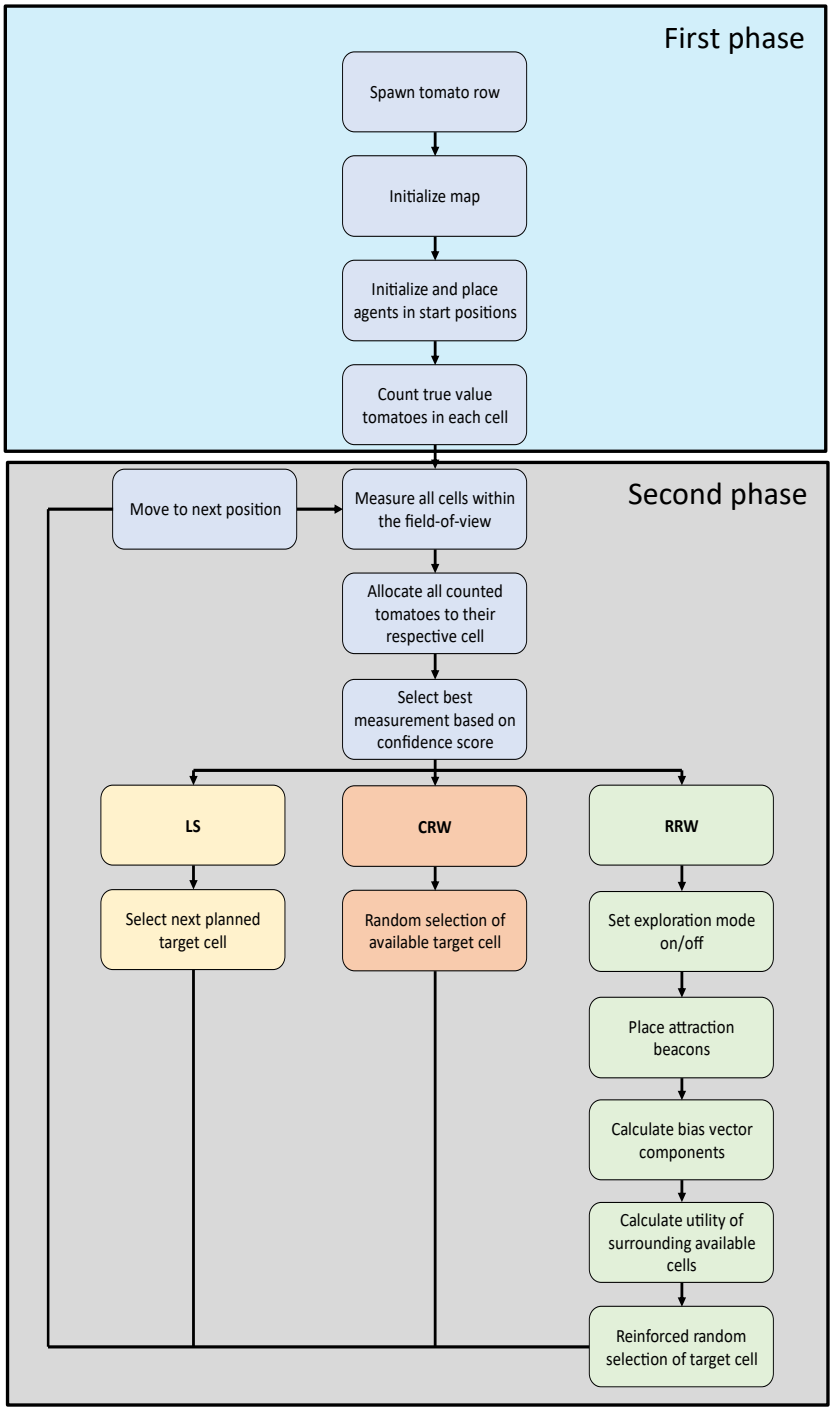


Figure 11: Flowchart of the steps taken by each simulation run, for all strategies.

4.1.3.1 Bias vector

In *RRW*, the next movement of agents is dependent on a bias vector (v), which is a resultant of three components: (i) momentum m_h, which points to the direction where the agent is currently moving with a magnitude based on the 80 g mass of a DJI Tello and 0.4 m s^{-1} cruise speed (Carbone et al., 2022); (ii) attraction vector a_h, that points towards cells that were marked with an attraction beacon $b \in B$ for potentially having tomatoes needing extra measurements; (iii) repulsion vector r_h, that points in the opposite direction of other agents.

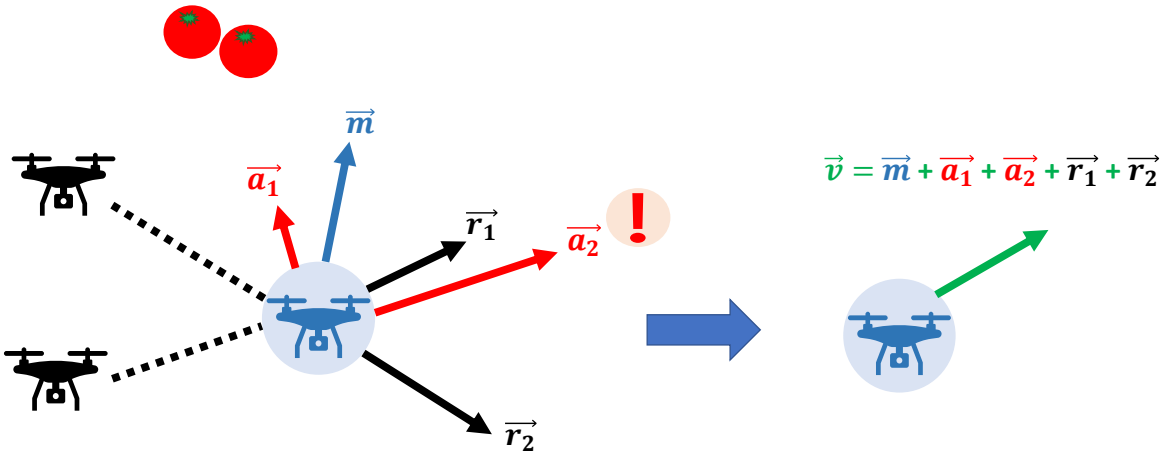


Figure 12: Schematic representation of how the bias vector is calculated. It is a resultant vector which points towards attraction points, such as cells that potentially have tomatoes, and against other UAVs.

Both attraction and repulsion vectors are calculated according to the Euclidean distance *v* between the position of attraction beacons x_b and the agent position x_h or between the position of other agents x_u and x_h:

$$r_h = \sum_{u \neq h} S(x_h - x_u, \sigma_a), a_h = \sum_{b \in B} S(x_b - x_h, \sigma_b), S(v, \sigma) = 2e^{i \angle v} e^{-\frac{|v|}{2\sigma^2}} \quad (2)$$

which results in a vector with a Gaussian length with spread σ and direction *v*. The magnitude of the resulting vector is directly related to the distance between the agent and the source of attraction or repulsion (Albani, Nardi, et al., 2017; Carbone et al., 2022).

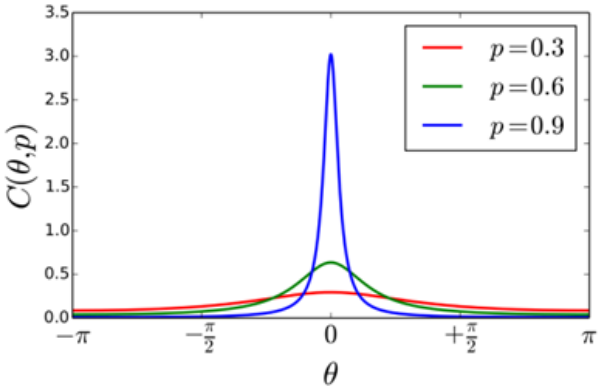
The next location towards where an agent will move is decided based on its bias vector $v_h = m_h + a_h + r_h$ and a weighted random selection from the cells in *V1*, if there are any available. Each cell $c \in V1$ has a utility u_c computed from the angular difference between v_h and the centre of *c*. The probability of selecting *c* as the next target is $P_c = u_c / \sum_{i \in V1} u_i$, with increasing probabilities for lower angular differences (Albani, Nardi, et al., 2017; Carbone et al., 2022). Utility is calculated as follows:

$$u_c(\theta_c, p) = \frac{1}{2\pi} \frac{1-p^2}{1+p^2-2p\cos\theta_c} \quad (3)$$

where θ_c corresponds to the angular difference between x_c and = v_h and *p* is the persistence parameter of the wrapped Cauchy density function $u_c(\theta_c, p)$. Persistence is computed according to the following equation:

$$p = 1 - e^{\frac{|v_h|}{2}} \quad (4)$$

where a higher magnitude of v_h results in a higher persistence, thus increasing the bias for the agent to move towards cells in the same direction of the bias vector.



After every u_c is computed, agents will first attempt to move towards cells with centre point located within an angular distance of 90° from the bias vector. If there are no available cells that fulfil this requirement, the others become eligible.

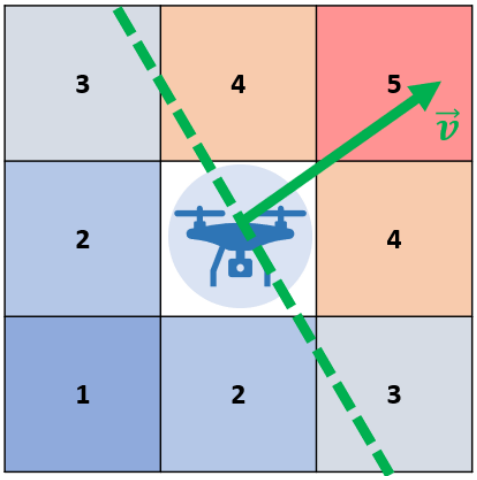


Figure 13: Representative utility values for all cels in V1 of the agent. Cells with center at a lower angular difference from the bias vector (in green) will have higher utility values.

4.1.3.2 Attraction factors

There are two types of factors that attract agents: (i) potential tomato cells, which indicate cells that may present tomatoes based on the surrounding cells; (ii) flags, which indicate cells that were measured more than once, but have unmatching number of tomatoes from the different measurements.

Every time an agent encounters tomatoes in any cell, the surrounding unmeasured cells will be marked as potentially presenting tomatoes. If any of these marked cells is measured, the mark is removed, regardless of the measurement result. Because tomatoes are spawned as groups in the plant branches and the field-of-view of the agents is usually not wide enough to visualize entire groups, the detection of tomatoes in a cell indicates high probability of other existing tomatoes in the adjacent cells.

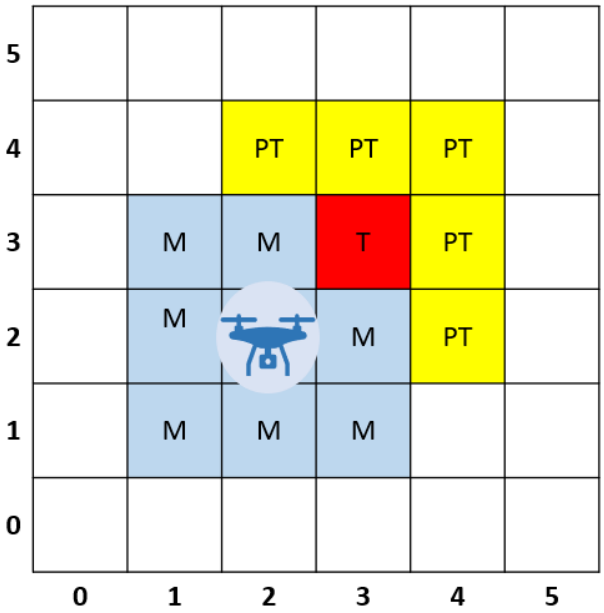


Figure 14: Schematic representation of how cells that potentially contain tomatoes (PT) are marked, according to the newly measured cell with tomato (T).

Flags are placed in every occurrence of unmatching measurements performed of a single cell. After marked, a cell will remain flagged until at least 50% of the surrounding cells are visited, resulting in more measurements of the same cell from different viewpoints.

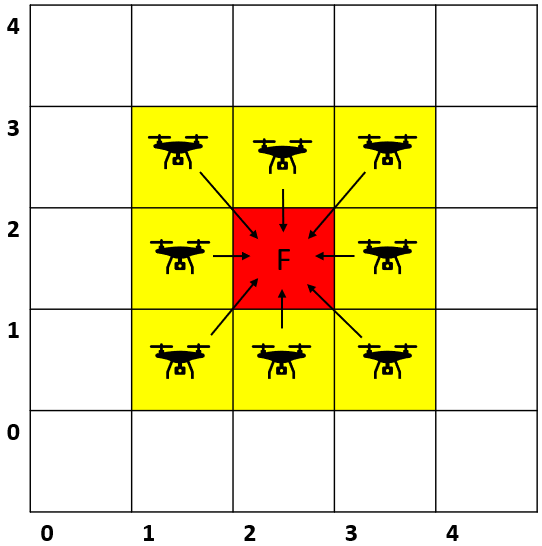


Figure 15: Schematic representation of a cell with a flag (F) and the surrounding cells (in yellow). At least four of the surrounding cells must be visited for the flag to be removed. Black arrows indicate the different angles from which the agents can observe cell F.

4.1.3.3 Exploration mode

To optimize agent navigation, by reducing the number of steps necessary to finish the simulation and find all unmeasured cells faster, an exploration mode was designed. Whenever an agent performed a measurement and detected no tomatoes, it was assumed that this agent was positioned in a part of the map where only plant stems and leaves were visible. In *RRW*, when the beforementioned situation occurred, the agent would enter exploration mode.

When in exploration mode, agents used cells in *V2* for their next target location, instead of the default *V1*. If no cells in *V2* were available, agents used *V3* cells and so forth. If the agent evaluated the availability of all vicinity cells until the limits of the map were reached and none were eligible, it meant that the only available cells were in *V1*, thus resetting the next target location to *V1*. Another characteristic of this mode is to prioritize unmeasured cells during the weighted election phase.

4.2 Object detection: Faster R-CNN

The task of counting the tomatoes from images was performed by the object detection model Faster R-CNN. This fully convolutional neural network was developed by a group of Microsoft researchers to quickly and accurately predict the location bounds and confidence scores of objects (Wu et al., 2019).

Faster R-CNN is one of the most popular models for fruit detection in field conditions (Gao et al., 2020) and has been used for a variety of plant-organ detection tasks (Velumani et al., 2021; P. Wang et al., 2022; Z. Zhang et al., 2016). For tomatoes, it is most used for phytopathology research (Alruwaili et al., 2022; Q. Wang & Qi, 2019; Y. Zhang et al., 2020), compared with its usage for tomato fruit counting (C. Hu et al., 2019).

This neural network was created by merging the Fast Region-based Convolutional Network (Fast R-CNN) with a Region Proposal Network (RPN). The RPN is responsible for generating bounding box proposals for where the object detector Fast R-CNN should focus. Because both RPN and Fast R-CNN utilize convolutional layers, the computations are shared across these components, resulting in a faster model (Ren et al., 2015).

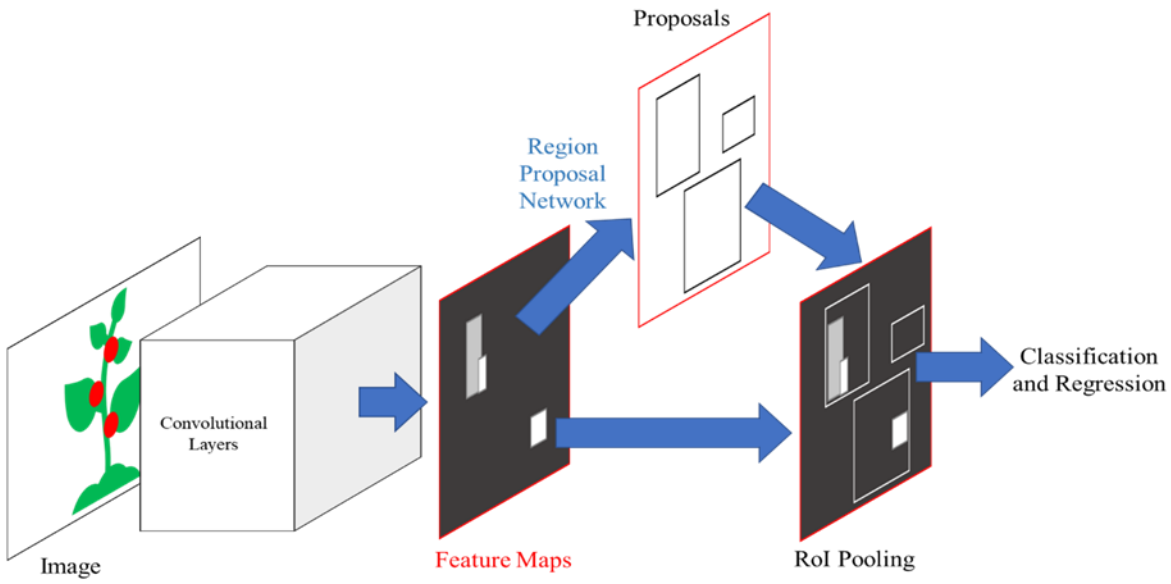


Figure 16: Summarized representation of the Faster-RCNN model (Ren et al., 2015).

First, the input image is fed to a backbone Convolutional Neural Network (CNN), resulting in feature maps that are used by the RPN to generate bounding box proposals. These bounding boxes are used to pool features from the feature maps, which is done by a Region of Interest (RoI) layer. The RoI layer takes each proposal, divides it in sub-windows and performs max-pooling over those. Finally, the result is passed through two sibling branches, one for bounding box regression (i.e., refining its boundaries) and another for classification (Ren et al., 2015). For this work, one "tomato" class was used.

4.2.1 Dataset generation

Image generation for the training and validation dataset was performed in Unity3D. For each image in the dataset, a new virtual scenario was generated and a snapshot was taken from a random viewpoint. Snapshots were automatically saved and labeled together with its annotations with ground truth bounding boxes of the tomato fruits.

Automation of the dataset generation was done in Unity3D through C# coding. Every image was acquired from a repeating routine that performed the following steps:

1. Place 15 equally spaced tomato plant spawners.
2. For each spawner, place a tomato plant without fruits at a random rotation (0° to 360°), with a random increment in size (-10% to +10%) in relation to the baseline scale.
3. For each tomato plant, place a random number of fruit bearing branches (0 to 3) at each of the fruit branch spawn locations.
4. For each tomato branch spawned, place a random number of tomatoes (0 to 5) with a random increment in size in relation to the baseline scale (-10% to +10%) and with a random color (red or reddish orange).
5. Randomize light angle and intensity.

6. Place a UAV agent facing the tomato row at a fixed orthogonal distance and at a random vertical and horizontal position within the boundaries of the generated row.
7. Take a snapshot and save it as a 1024 x 1024 PNG image with a unique label.
8. Create bounding boxes for all tomatoes observed in the snapshot according to their coordinates in the virtual world.
9. Create and save annotation files in JSON with the image label and every tomato's bounding box.

Steps 1 to 4 were also used at the start of every experimental run and will henceforth be referred to as spawn tomato row.

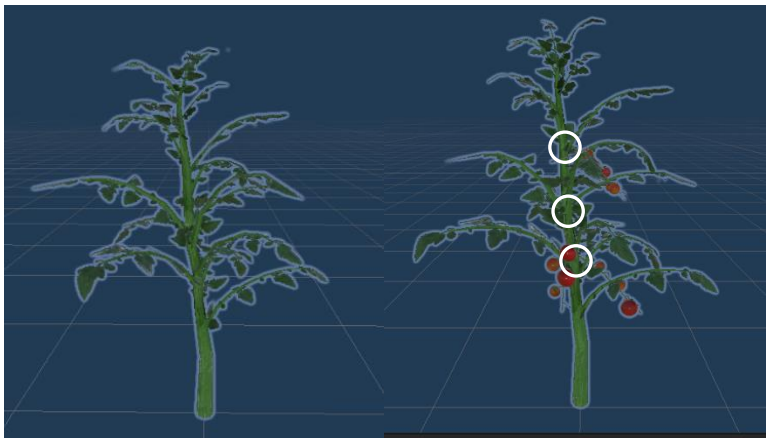


Figure 17: Tomato plant prefab model without fruits (left) and the same model with tomato branches spawned in the pre-determined spawn locations, marked with white circles (right).

4.3 Simulation environment: Unity3D

Experiments were set up in a virtual environment consisted of the following components: (i) Unity3D, which rendered the graphics of the simulation environment and controlled actions inside of it through C# scripts; (ii) image folder where snapshots taken by the simulated UAV agents are saved; (iii) a server coded in Python that reads specifically requested images and use them as input for the object detection algorithm, returning the bounding boxes of the predicted tomato locations.

Navigator was the script that controlled the actions of the agents, spawned new plants through the Plant Spawner script, tracked the detection of tomatoes and saved the results of each run.

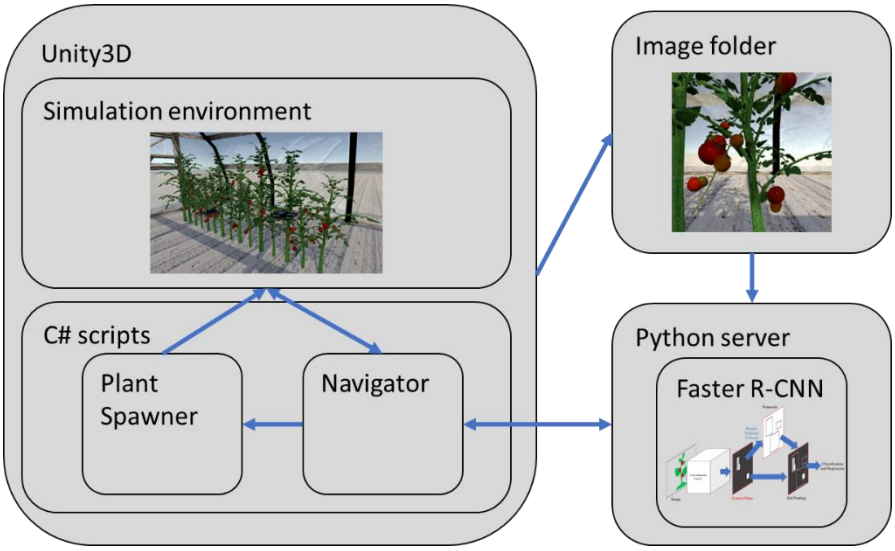


Figure 18: Schematic representation of how the simulation environment interacts with the code to spawn the scenery, coordinate UAV navigation and run the object detection algorithm.

The nature of this work requires visual sensing of target objects (i.e., tomatoes) that are positioned in a 3-dimensional structure (i.e., tomato row). In real-life situations, the targets could be frequently obstructed by plant stems, leaves and even other tomatoes. Having these obstructions in an experiment that makes use of multiple UAVs would be crucial to test the swarm capability of observing the same location from multiple angles. The assumption is that some of these angles would allow the detection of targets that are otherwise obstructed when observed from a different angle.

To optimize productivity, crops are kept relatively close to each other in a greenhouse. For tomatoes, the distance can be from 75 to 120 cm between rows (Ali et al., 2017; Bechar et al., 2007). A UAV performing sideways maneuvering along a crop row would need to be confined to its boundaries, which means that the agents would be positioned at a short distance of the plants while acquiring images. At that distance, image texture and details become important to ensure that the object detection algorithm is robust and applicable to a real-life scenario. Thus, another requirement for this work is that of realistic scenarios.

Because of the beforementioned reasons, Unity3D, a videogame development software that allows the creation of realistic 3D virtual environments, was chosen to develop the simulation environment. Unity3D is a popular tool for generating virtual environments for agricultural research that require detailed visual aspects, including for tomato research (H. Li et al., 2017) and with multi-robot approach (Roldán et al., 2016).

Unity3D is coded in C# and possesses an Asset Store where users can purchase and sell packages of 3D scenarios and objects, including agricultural related structures such as plants and structures.

4.3.1 3D models

The baseline 3D models used for the virtual environment, including tomato plants, tomato fruits and the greenhouse were obtained through the Asset Store, from the package "Greenhouse - Gardening Tools", by the author "PropDrop". Other assets that constituted the virtual environment were the UAV models and a floor texture that imitated an agricultural field, both obtained from Carbone et al. (2022).

Tomato branches from the baseline models were edited to resemble realistic tomato clusters. Plants, which here are defined as the combination of stem and leaves, were set to a height of 2 m (Fatnassi et al., 2009) and the dimensions of the tomato clusters, defined as the combination of fruit bearing branches and fruits, were obtained from real-life measurements of a supermarket bought tomato vine (*trostomaten*, in Dutch language). The UAV model dimensions were based on the dimensions of a DJI Tello (i.e., 98 x 92.5 x 41 mm).

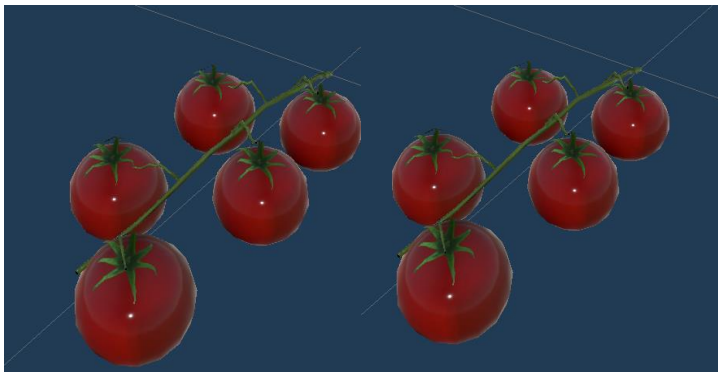


Figure 19: Example of the original prefab 3D model from tomato branches loaded with fruits (left) and the improved prefab model used in the simulations (right).

The virtual simulation environment consisted of a greenhouse structure positioned on top of a generic agricultural soil. Inside the greenhouse, a row of 15 tomato plants was placed. The agents were 3D objects that resembled the AR Parrot.

Figure 20: Example of a randomly generated simulation environment, where the UAV model is highlighted.

4.4 Experimental design

To answer research question C, three levels of number of agents were used. Initially, the maximum number of agents was four, however due to computational limitations it was changed to three. To evaluate the performance of the exploration algorithm *RRW*, two other navigation strategies were used for comparison. Two parameters of the *RRW* were also analysed at two levels: *low = 2*, which was the lowest value experimented by Albani, Nardi, et al. (2017) and *high = 8*, which was the value that resulted in the lowest mapping time by the same authors. Each of the 16 treatments had ten repetitions, resulting in 160 simulation runs.

Table 2: Experimental design displaying the parameters of each treatment.

Exploration approach	Number of agents	σ_a	σ_r	Repetitions
LS	1	-	-	10
	2	-	-	10
	3	-	-	10
CRW	1	-	-	10
	2	-	-	10
	3	-	-	10
RRW	1	2	-	10
		8	-	10
	2	2	2	10
		2	8	10
		8	2	10
		8	8	10
	3	2	2	10
		2	8	10
		8	2	10
		8	8	10

4.5 Metrics

To evaluate the performance of the simulations, three metrics were used: Root Mean Squared Error (RMSE), Simulation Time (t_s) and Discovery Ratio (DR).

RMSE is a common metric for evaluating the robustness of object detection algorithms in fruit counting tasks (Bhattarai & Karkee, 2022; Rahnemoonfar & Sheppard, 2017; Underwood et al., 2016; Z. Wang et al., 2019). It is a measurement of accuracy that computes the quadratic mean of the difference between the ground-truth value (i.e., real number of

tomatoes) and the predicted value from the CNN output. This metric was calculated for each cell, then summed and averaged out by the total number of cells in the map, as described by the following equation:

$$RMSE = \sum_{i,j=0}^{r,c} \frac{\sqrt{(\hat{y}_{i,j} - y_{i,j})}}{rc} \quad (5)$$

where r and c are the row and column number, respectively, used to reference each cell, $\hat{y}$ is the number of predicted tomatoes from the object detection model output and y is the real number of tomatoes inside cell i,j.

In order to speed up the experimental process, simulations were not run in real-life time. Meaning that simulation time had to be calculated and stored for each simulation step. Whenever an agent needed to move from point A to point B, the time required for such movement was calculated based on the agent's speed and the distance that it must cover, according to the following equation:

$$t_{s,sim\_step} = \frac{c_{sim\_step} - c_{sim\_step-1}}{0.4} \quad (6)$$

where the simulation time t_s at simulation step *sim_step* is the distance between the centre of the cell where the agent was positioned in previously and the cell where it is currently standing, divided by the agent speed of 0.4 m s^{-1}. When two or more agents had to move simultaneously, the simulation time for that simulation step was considered to be the highest time between all agents.

Discovery ratio is a measurement of the percentage of tomato cells discovered (i.e., measured) by agents from the total number of cells with tomatoes. At the start of each run, every cell that contained tomatoes was mapped and their coordinates were stored, this number is referred as Total Tomato Cells (*TotalTomCell*). In every simulation step, the new cells measured by drones that contained tomatoes were accounted and updated the number of Discovered Tomato Cell (*DiscTomCell*). For every simulation step (*sim_step*), the discovery ratio was calculated as:

$$DR_{sim\_step} = \frac{DiscTomCell_{sim\_step}}{TotalTomCell} \quad (7)$$

5 Results

5.1 Object Detection: Training and Validation

A dataset of 350 images generated artificially was used for the training of the neural network and 150 images were used for validation, which was done with Detectron2. It is an open-source platform developed by a group of Facebook researchers and implemented in PyTorch. It includes implementations of several object detection algorithms, including Faster R-CNN, which was developed by the same company (Pham et al., 2020).

To initialize training, the model *faster_rcnn_R_101_FPN_3x* from the detectron2 model zoo, pre-trained on the COCO dataset was selected. This model is both high scored for object detection and uses less memory than similar models (Wu et al., 2019). Training took 24 minutes and 10 seconds, resulting in a total loss of 0.521.

For validation, a score testing threshold of 0.7 was used, which means that only objects detected with a confidence score of at least 70% were accounted. The Average Precision (AP) of validation was 52.8, which is not high, but still superior to the 36.2 AP of the same model on the COCO dataset (Lin et al., 2017).

Figure 21: Result from the object detection model in one of the virtually generated images.

Another validation was performed with a dataset of 123 real-life images of greenhouse tomatoes. This dataset and annotations were provided by (Afonso et al., 2020). The purpose of this second validation was to observe if a model trained with virtually generated images would be able to perform in a dataset of real-life images.

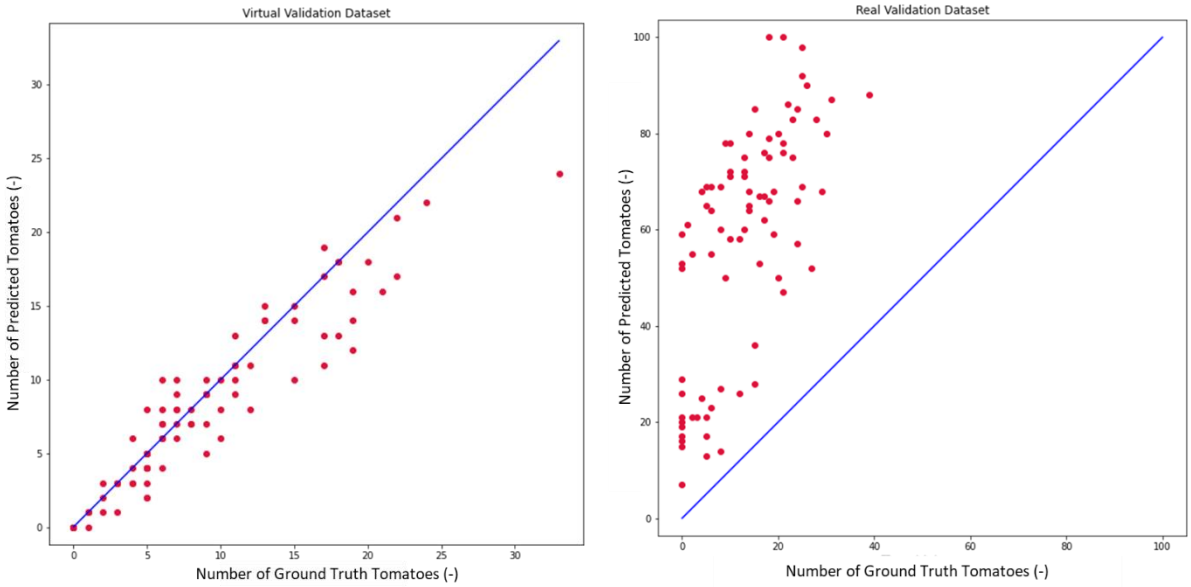


Figure 22: Number of tomatoes predicted by the object detection model versus the ground truth number while using the virtual dataset (left) and the real-life dataset (right). Blue line indicates where predicted number of tomatoes is equal to the ground truth number.

Although the trained model seemed to correctly detect real-life tomatoes, the results show an overestimation of the total real amount. The explanation is that the annotations of the real-life dataset accounted only for tomatoes directly in front of the plant row, while for the current work, the model was trained to also detect tomatoes behind the row.

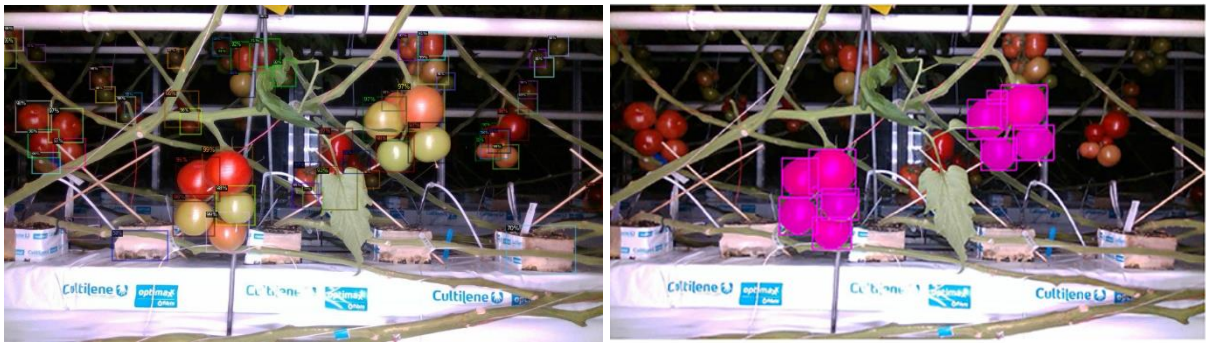


Figure 23: Results from the object detection model using real-life tomato pictures (left) and the ground-truth bounding box and masks from the annotations (right) (Afonso et al., 2020).

After training, the model was saved as a PyTorch model file. Because Unity3D runs on C# and the model runs on Python, a socket communication was used to enable interaction between the code that controls actions inside Unity3D and the Python code that ran the object detection model.

5.2 Root Mean Squared Error (RMSE)

By the end of each simulation run, the RMSE was computed. There were 10 repetitions for each treatment, which means that there are 10 RMSEs calculated for every treatment used, which were averaged and displayed in the boxplot below (Figure 26).

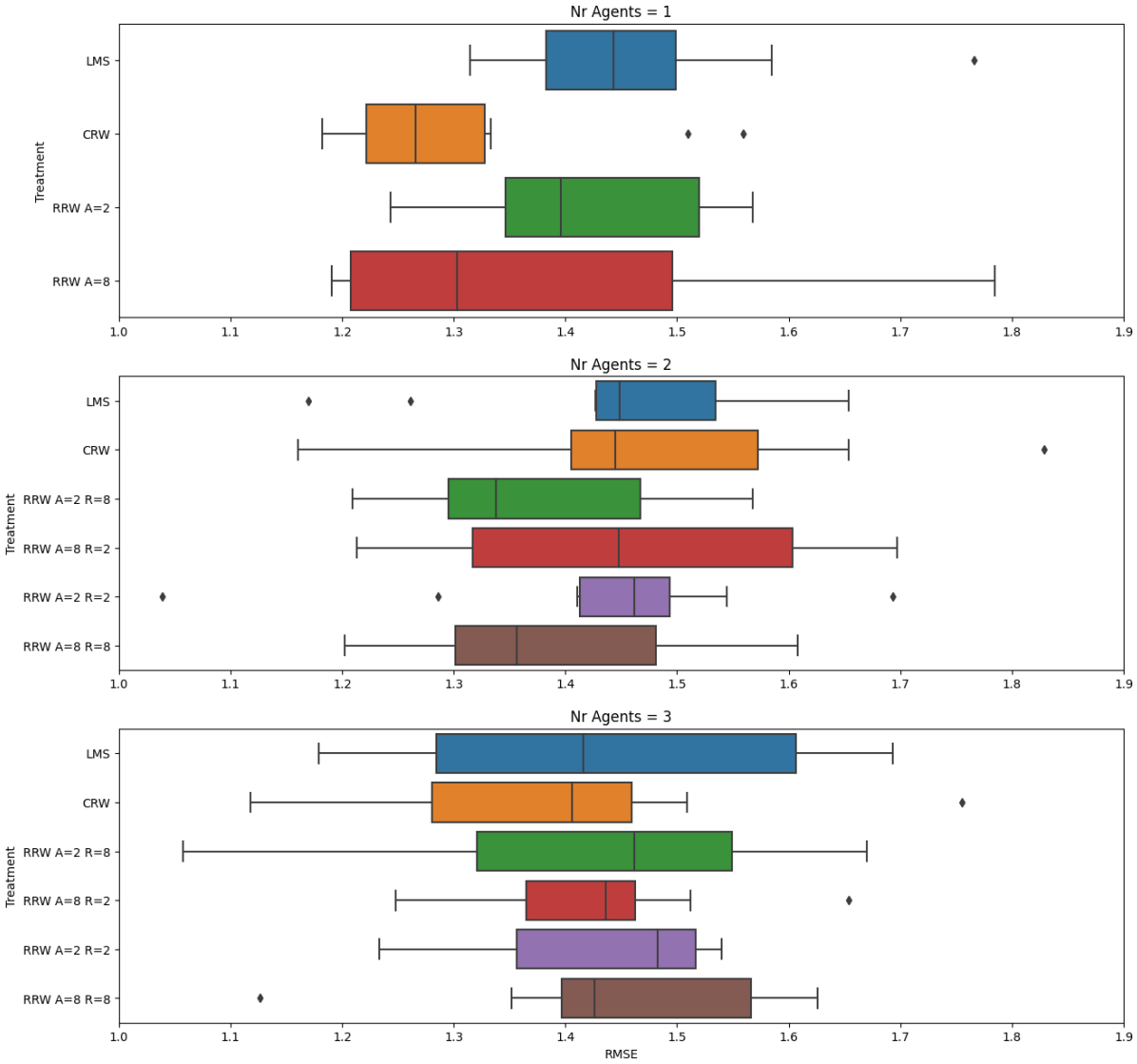


Figure 24: RMSE of all treatments used in the simulations.

The lowest average RMSE obtained was 1.308 for exploration strategy RS with 1 agent, while the highest was 1.469 also for exploration strategy RS, but with 2 agents.

Performing Levene's Test to check for homoscedasticity, it was obtained a p-value of 0.643, thus rejecting the null hypothesis. This means that there is no sufficient evidence that the variance of MSE from all treatments differs significantly. A histogram of the normalized RMSE also showed that the distribution of the data was normal. Both of these indications are assumptions for an ANOVA test.

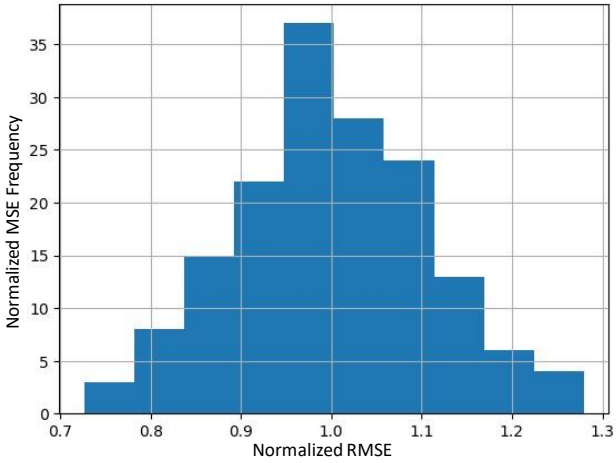


Figure 25: Histogram of normalized RMSE indicating normal distribution.

Statistically, the RMSE showed no significant difference for different number of agents or exploration strategies, with a p-values equal to 0.320 and 0.348 respectively, from a two-way ANOVA with significance of 1%. It also resulted in no significant difference for varying levels of attraction and repulsion parameters withing the *RRW* strategy, with a p-value of 0.905, 0.542 and 0.788 for the factors of attraction parameter, repulsion parameter and number of agents, respectively.

Overall, the RMSE of all treatments are comparable to research with similar approach, such as (Rahnemoonfar & Sheppard, 2017), which achieved a RMSE of 1.16 for a task of tomato fruit counting and even lower then (Oliveira et al., 2022), which achieved a RMSE of 1.54 for the task of counting orange fruits in an orchard.

5.2.1 Two-sided RRW

To further investigate the possibility to achieve a significant reduction in RMSE, a new approach was tested. It consisted of using *RRW* with 2 agents but placing each of the agents in opposite sides of the tomato row. The hypothesis is that tomatoes that were being occluded from the agent's *POV* from one side of the row might be visible to the other agent's *POV* from the opposite side of the row, as represented in the scheme in Figure 28.

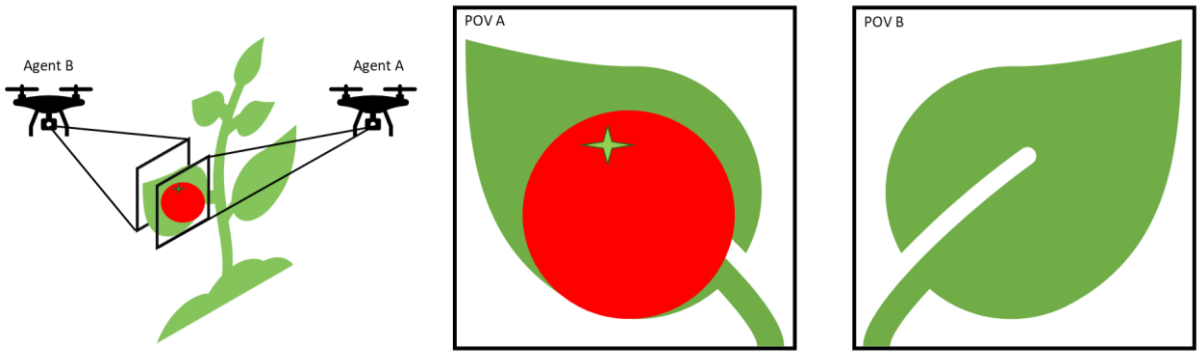


Figure 26: Schematic representation of the type of situation where observing the plant row from both sides could prevent errors due to obstruction. Here, Agent A can observe a tomato that Agent B cannot.

Some adaptations had to be done for the two-sided *RRW*. Starting with the rule on visiting cells, agents were now allowed to visit the same cell, because the image taken from the same cell would be done from opposite sides of the plant row. Secondly, the bounding boxes yielded from object detection had to be adjusted to be coherent when added to the map, because the map was created based on XY-coordinates that used only one side of the plant row as reference.

Two-sided RRW did not manage to reduce the RMSE considerably when compared to the other *RRW* treatments with Number of Agents equals to 2, as shown by an ANOVA test with significance of 5%, and p-value equals to 0.921.

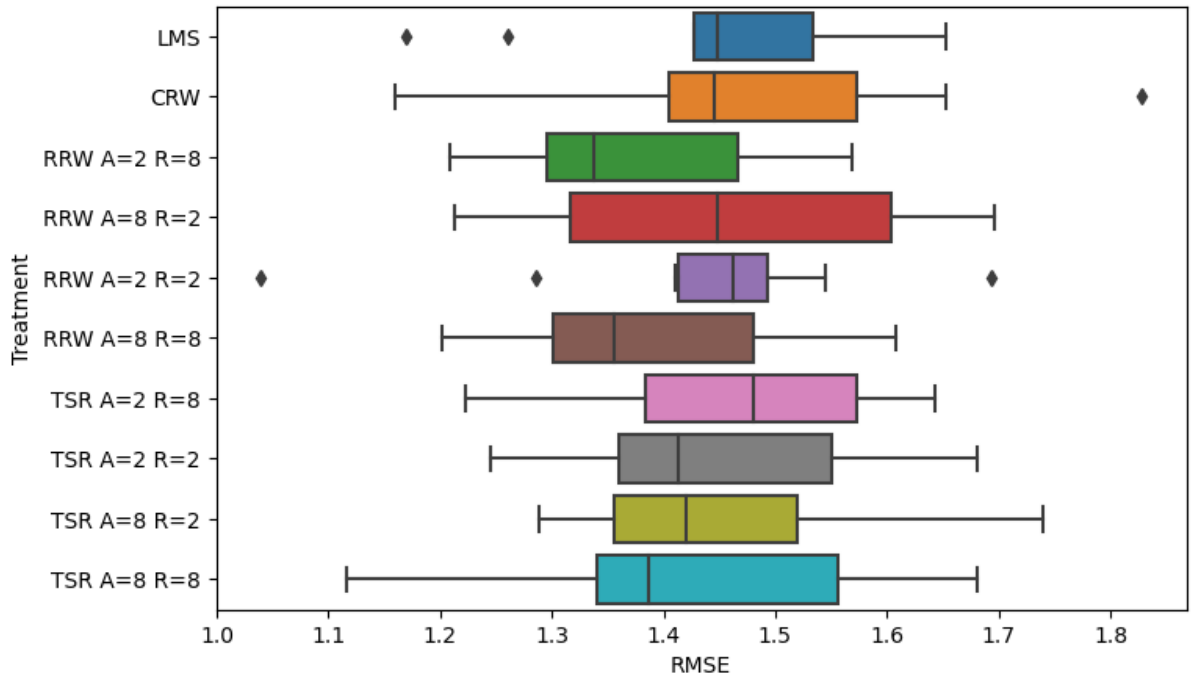


Figure 27: RMSE of TSR compared with the other treatments used with two agents.

The already low magnitude of the RMSE might influence the difficulty to lower it even more. There are some hypotheses regarding the reasons for these low RMSE results: (i) they demonstrate the robustness of the object detection algorithm, (ii) there lacked enough noise to the visual sensing component (e.g., adding more obstructions, adding distortion or quality loss to the image acquisition in order to better simulated real-life scenarios).

5.3 Simulation time (t_s)

Simulation time required to finish the task was also computed for 10 repetitions of each treatment and shown in the boxplot below (Figure 30). The results for the Lawnmower Strategy appears as a single stripe because all the resulting simulation times are equal, due to the exploration approach having the same pre-determined steps for every repetition.

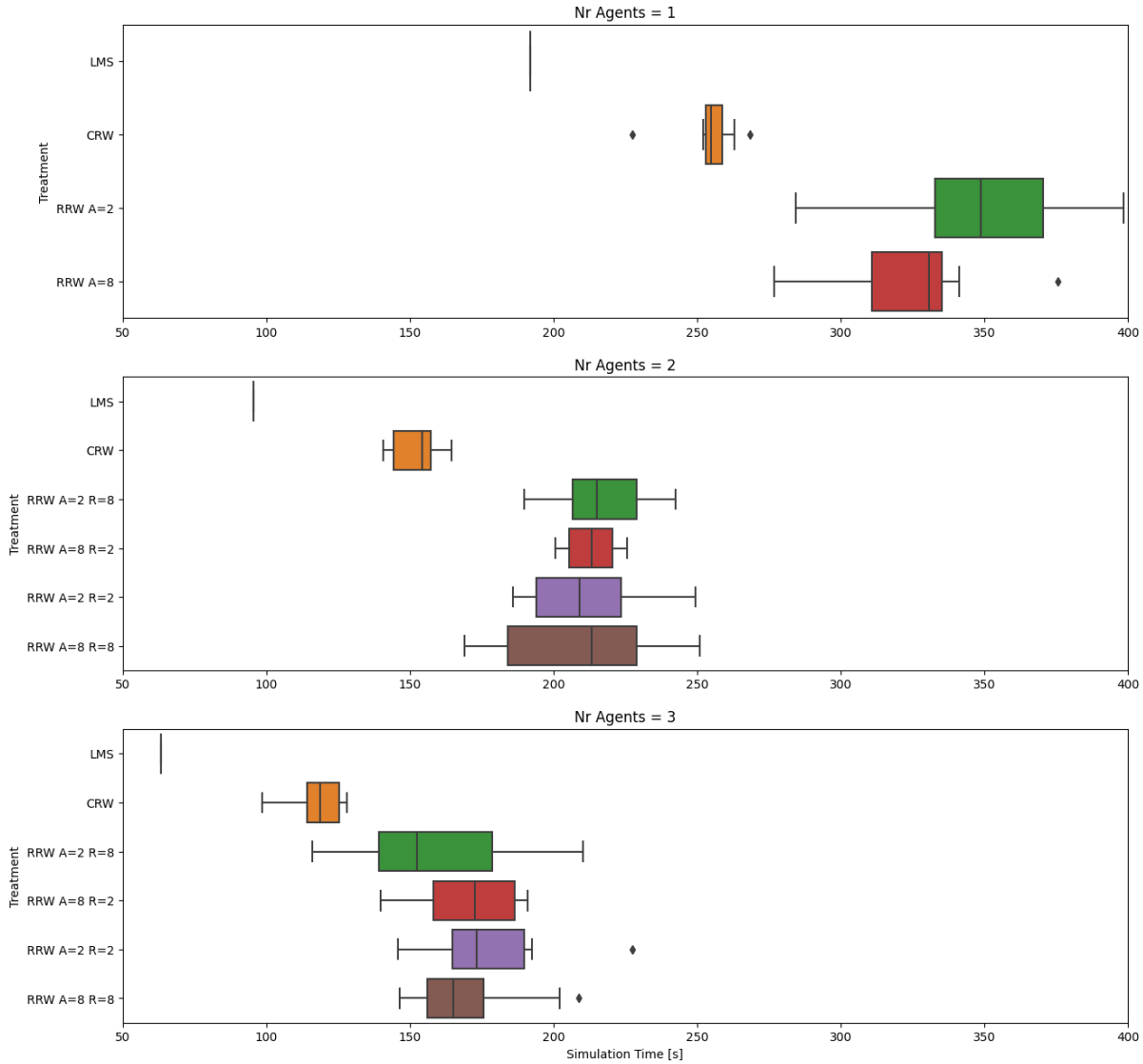


Figure 28: Simulation time in seconds of all treatments used in the simulations.

There was a significant difference between the means of simulation time across the different number of agents as well as exploration strategies, as demonstrated by the two-way ANOVA with significance of 1% and p-value lower than 0.001.

The lowest time taken to finish the task were, as expected, achieved by the higher number of agents. Within each number of agents level, the lowest t_s was obtained from the lawnmower strategy, which contradicted the expectations. Initially, it was expected that *LMS* would result in a higher t_s than *RRW* due to the premise that, in *LMS*, agents must visit, every single cell in the map for the run to end, whereas in RRW, every cell must only me measured at least once. Since an agent is able to measure 9 cells at every step, it was believed that, in *RRW*, the agents would take less steps to finish measuring every cell in the map, compared to *LMS*.

5.3.1 Number of simulation steps

The assumption of using *RRW* with the addition of an *exploration mode* was to reduce the number of steps necessary to finish the simulation. This assumption is based on the constraint to end the simulation in *LMS* versus in *RRW*. In *LMS*, all cells in the map must be

visited, whereas, in *RRW*, all cells must be only at least measured. *RRW* in combination with *exploration mode* would allow agents to move to cells in *V2* when there would be no tomatoes visible from their current *POV*, covering a wider area in a single step and needing fewer total steps to map the entire area.

The number of steps needed by simulations with *RRW* should also be lower than those needed by simulations with *CRW*. The assumption is that in *RRW*, agents would be directed towards cells that potentially contain tomatoes, while *CRW* had no specific priority. This would cause agents in *RRW* to enter *exploration mode* soon after measuring all cells with tomato and cover the rest of the map in fewer steps than *CRW*.

In order to assess these assumptions, it was observed the total number of steps in each simulation, which are demonstrated in the following table.

Table 3: Mean and minimum number of steps taken to complete the task, for all treatments used.

Nr of Agents	Treatment	Nr of steps (mean)	Nr of steps (min)
1	LMS	285	285
	CRW	276	263
	RRW A=2	280	266
	RRW A=8	277	270
2	LMS	142	142
	CRW	140	138
	RRW A=2 R=8	141	139
	RRW A=8 R=2	140	137
	RRW A=2 R=2	140	135
	RRW A=8 R=8	138	129
3	LMS	94	94
	CRW	93	86
	RRW A=2 R=8	92	86
	RRW A=8 R=2	94	91
	RRW A=2 R=2	94	90
	RRW A=8 R=8	93	89

The average total number of steps needed to complete the simulations using *RRW* are slightly lower than the amount needed by *LMS*. Still, the total simulation time with *RRW* is higher than *LMS*. To investigate this, the average simulation time was plotted against each step (Figures 28, 29 and 30).

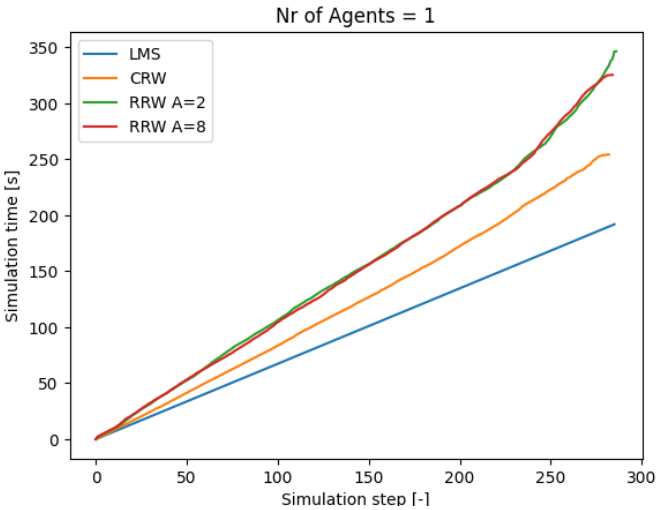


Figure 29: Simulation time against step for simulations with 1 agent.

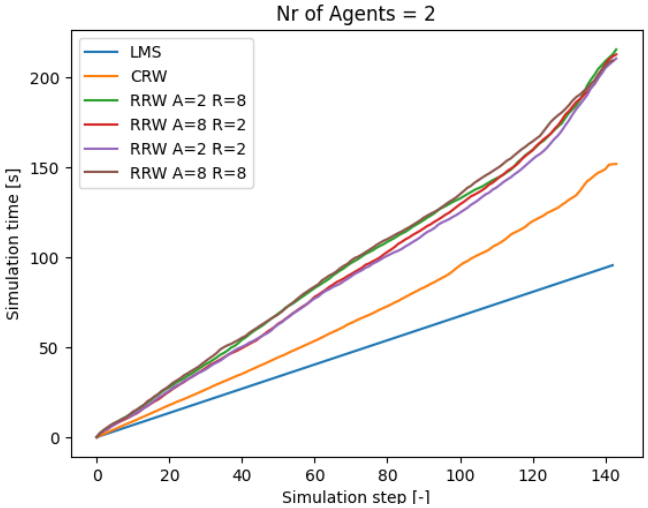


Figure 30: Simulation time against step for simulations with 2 agents.

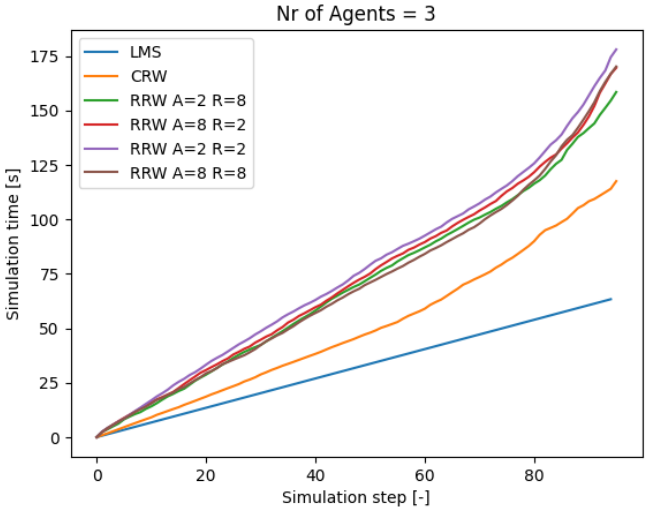


Figure 31: Simulation time against step for simulations with 3 agents.

The duration of each step in *LMS*, naturally, stays constant, because every step is done to an adjacent cell. An average higher duration for each step was already expected for *RRW*, because when in *exploration mode*, agents are programmed to move with a stride of 2 cells (cells in *V2*), causing them to cover more distance with the same speed in a single step.

However, the time taken for each step in *RRW* increases more than expected as simulation time proceeds. It is suspected that the reason can be local minima. When the agents are surrounded by cells marked as *visited* they are programmed to enter a loop to seek the next available, finding them distant from their surrounding vicinity (*V1*). This must happen with higher frequency by the end of the simulations with *RRW*.

5.4 Discovery Ratio (DR)

Discovery Ratio of cells with tomatoes is demonstrated in the line plots below (Figures 31, 32 and 33), where each line represents the average from the 10 repetitions for each treatment.

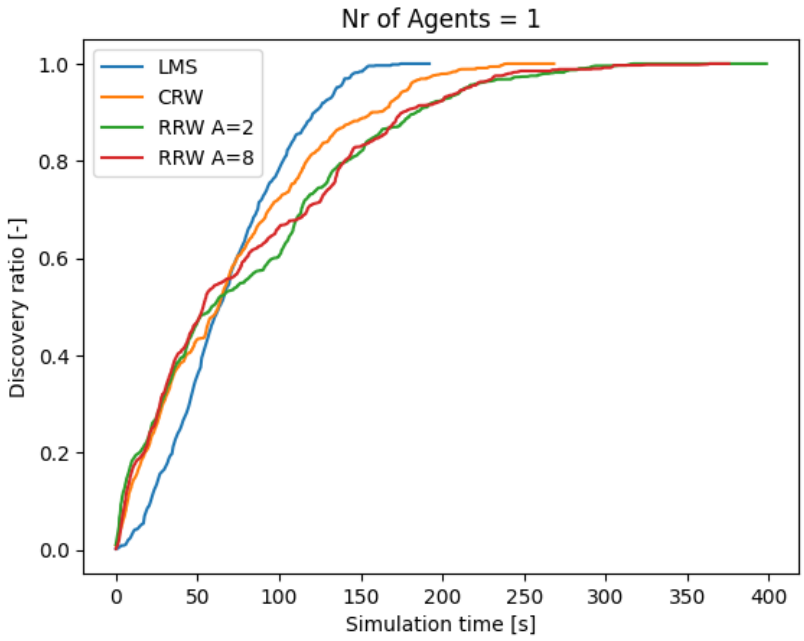


Figure 32: Discovery ratio of cells with tomatoes when simulating with 1 agent.

As seen from Figure 34, with 1 agent, *RRW* demonstrates the highest Discovery Ratio for the first half of cells with tomatoes discovered, which means that it is faster than other strategies to find cells with tomatoes at first, but decreasing in speed as simulation time increases.

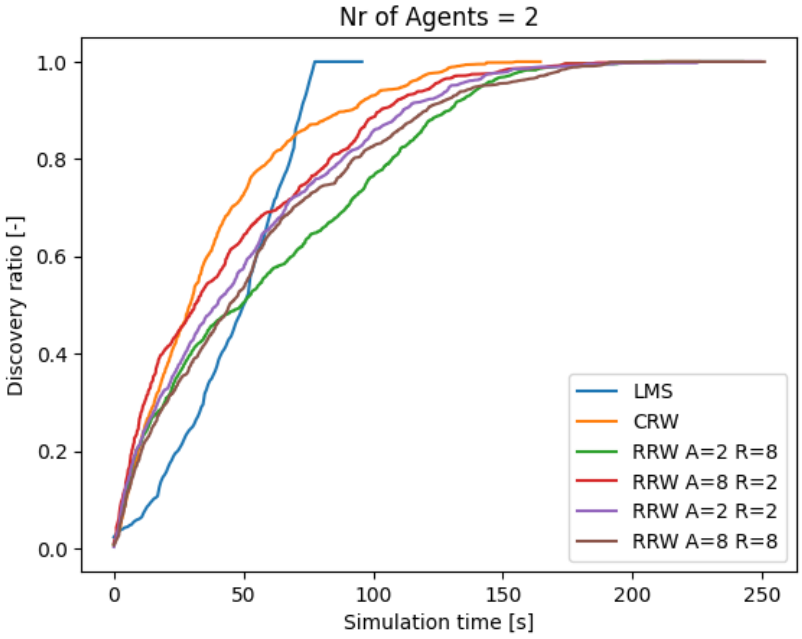


Figure 33:Discovery ratio of cells with tomatoes when simulating with 2 agents.

With 2 agents, *RRW* still enables a faster discovery of cells with tomatoes than *LMS*. Although, this time *CRW* performs even better than *RRW*, when compared with 1 agent. Similar to the results with 1 agent, the performance of *RRW* decreases and *LMS* is able to detect all tomato cells faster. There are also no clear differences in the Discovery Ratio between the different attraction and repulsion parameters tested with *RRW*.

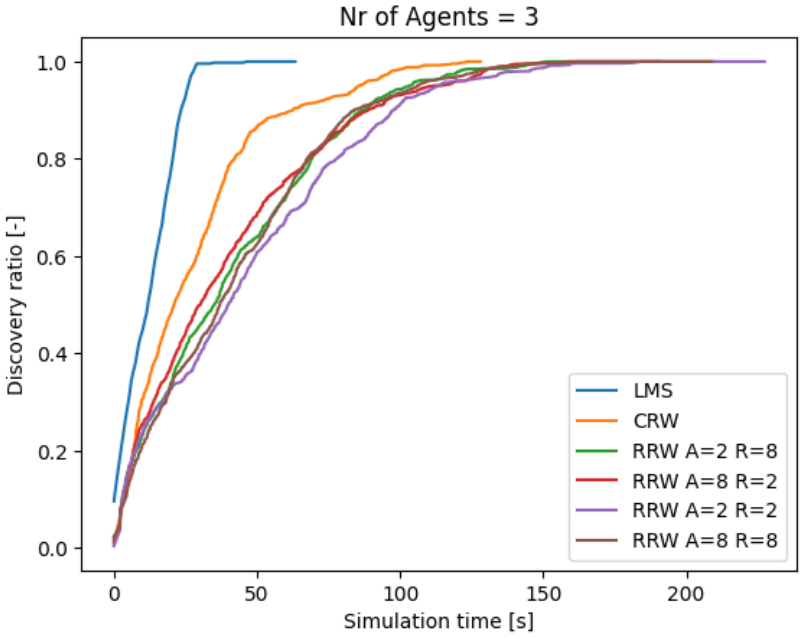


Figure 34: Discovery ratio of cells with tomatoes when simulating with 3 agents.

Differently than in previous results, with 3 agents *LMS* demonstrated the lowest time for the agents to measure all cells that contained tomatoes, while *RRW* needed more time for the same task.

6 Discussion

6.1 Alternative strategy (*ALT*)

Due to the unexpected low performance of the *RRW* in comparison with *LMS* and *CRW*, which were baseline strategies, an alternative strategy was designed to try to tackle flaws from the original *RRW*. The new design, denotated *ALT*, would consist of a *RRW* strategy for the selection of the agents' next target location, but with the following modifications:

1. Removal of *exploration mode*, as it increased the time to perform each step without bringing any benefits in the reduction of total simulation time.
2. A new requirement to end the simulation which would be based on a total number of steps equals to 120 instead of it being based on map coverage.
3. Flags, which indicated cells that should have more measurements due to uncertain fruit counting, were removed based on the assumption that RMSE was already low and that flags were contributing to the incidence of local minima.
4. The method to mark cells as potentially containing tomatoes changed from marking all unmeasured cells around a newly measured cell with tomatoes, to only marking unmeasured cells that bordered the bounding box of a detected tomato, as demonstrated in the schematic example in Figure 37.

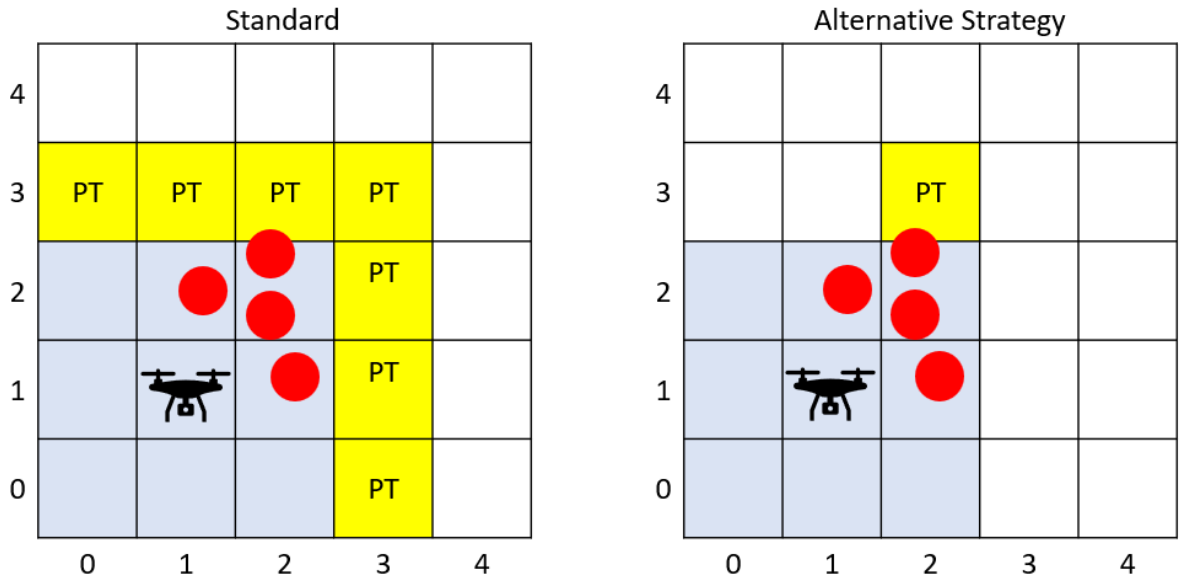


Figure 35: Schematic representation of the standard strategy to mark cells that potentially contain tomatoes (PT), versus the alternative strategy).

In the example above, the drone positioned in cell [1, 1] performs measurement in all cells that directly surround it, marked in blue. Tomatoes were detected in cells [1, 2], [2, 1] and [2, 2]. In the standard strategy, for each detected tomato cell, all unmeasured cells that directly surround it will be marked as PT, while for the alternative strategy, only cells that are directly in contact with the bounding box of detected tomatoes will be marked as PT. This is meant to reduce the number of unnecessary attraction points and direct the drone towards cells with higher probability of containing other tomatoes.

This alternative strategy was only simulated for 2 agents, to avoid technical issues frequently encountered when running simulations with 3 agents, such as software and system

crashes. The values for the attraction and repulsion parameters selected for this simulation were 8 and 2, respectively, due to the slightly better performance of this combination on Discovery Ratio, when compared with the other combinations for *RRW* with 2 agents.

The results for Discovery Ratio (Figure 38) showed that there was a visible enhancement in the performance of the alternative strategy (*ALT*) versus the original approach (*RRW*). The time needed to achieve a Discovery Ratio of 90% was on average 83.6 seconds with the *ALT* strategy and 104.3 seconds for *RRW*. Also, *ALT* achieved a Discovery Ratio of 40% in 18.0 seconds, while *CRW* needed 24.8 seconds to achieve the same result. This shows that the newly designed strategy *ALT* was able to inherit from *RRW* the speed to discover the first 40% of the tomatoes, while discovering the last tomatoes earlier, just as *CRW*.

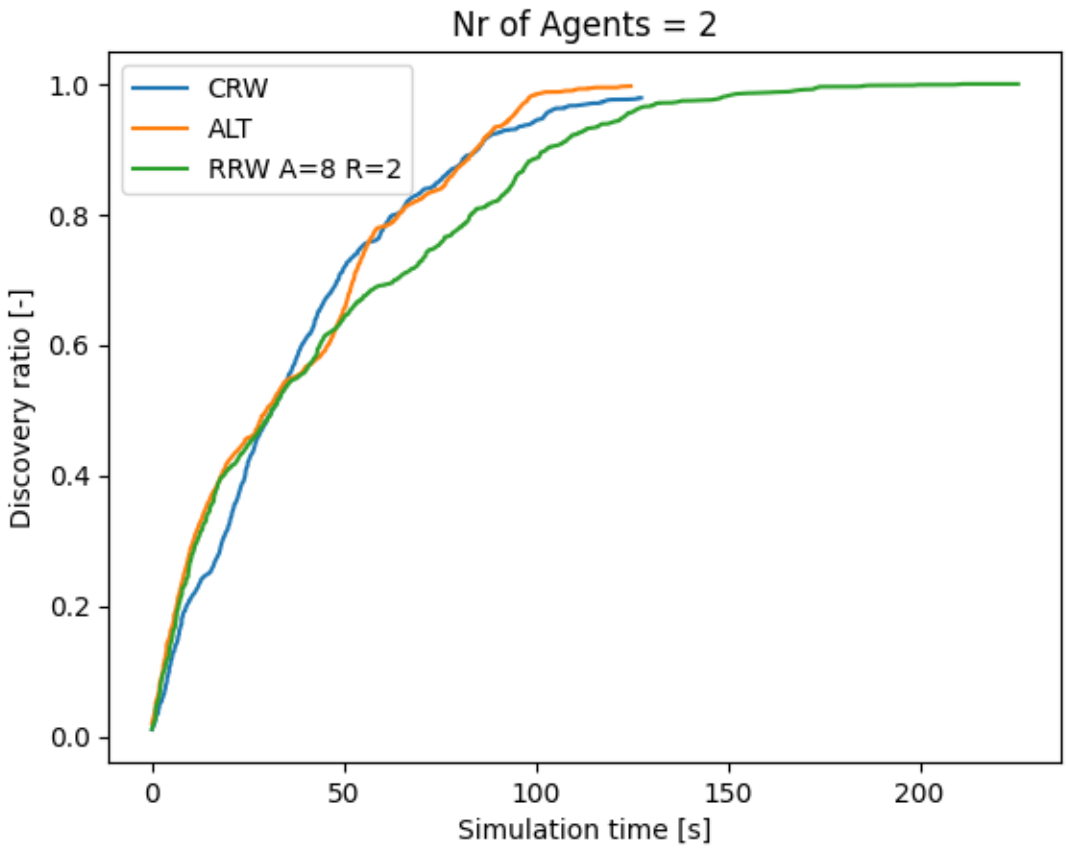


Figure 36: Discovery ratio average over simulation time. Simulations were performed with maximum 120 steps, repeated 10 times for each strategy.

6.2 Sub-question C: Number of agents

There was a clear improvement in the required time to complete the fruit counting task when using a higher number of UAVs. Comparing with simulations with 1 agent, simulations with 2 agents had their total simulation time lowered by 50, 60 and 65 % for the *LMS*, *CRW* and *RRW* respectively. The higher reduction for *CRW* and RRW is due to the higher total time to complete the mission for those strategies. Adding a third agent, reduced the simulation time by 67, 92 and 108 %, relative to 1 agent simulations for the *LMS*, *CRW* and *RRW* respectively (Table 4).

Table 4: Average total simulation time and relative time reduction for the three main strategies compared.

Exploration approach	Nr of Agents	Simulation time [s]	Relative reduced time [%]
LMS	1	192	-
	2	96	50
	3	63	67
CRS	1	254	-
	2	152	60
	3	118	92
RRW	1	352	-
	2	230	65
	3	191	108

As beforementioned in Sections 1 and 3.2 battery autonomy is one of the main constraints of using low-altitude UAVs due to the reduced time available for their flights (Arafat et al., 2023). Using drone swarms could be an answer to such problems by reducing the necessary time through task sharing between agents (Skobelev et al., 2018). To understand how the number of agents would impact in the efficiency of the tomato counting task, a real-life scenario was proposed. The following characteristics were assumed to be representative of a standard tomato greenhouse:

1. Plant row height of 2 m (Amundson et al., 2012)
2. Plant spacing ($plant_{space}$) of 0.5 m (Amundson et al., 2012)
3. Row spacing (row_{space}) of 0.8 m (Testa et al., 2014)
4. Greenhouse dimensions of 9.14 (gh_{width}) x 29.26 (gh_{length}) m (Athearn et al., 2018)
5. Walking space ($walk_{space}$) of 1 m along each wall of the greenhouse

With the assumed greenhouse characteristics and the equations presented below, it was estimated that such greenhouse would present 9 rows of tomato plants (n_{rows}) with 55 plants in each row (row_{plants}), resulting in 495 total plants.

$$n_{rows} = \frac{gh_{width} - 2walk_{space}}{row_{space}} \tag{8}$$

$$row_{plants} = \frac{gh_{length} - 2walk_{space}}{plant_{space}} \tag{9}$$

To calculate what the appropriate sample size is for estimating a population parameter, such as fruit number per plant, the following equation was used (Walsh et al., 2022):

$$n = \frac{\sigma^2 \left(t_{\alpha/2,n-1}\right)^2}{e^2} \tag{10}$$

where n is the required sample size, σ is the standard deviation of the population parameter (i.e. number of tomatoes per plant), $t_{\alpha/2,n-1}$ is the t-score and e is the margin of error, which was set to 1. This resulted in n = 128, which means that the minimum sample size to confidently estimate the average number of tomatoes per plant in a population of 495 plants is 128 sample plants. The standard deviation σ was obtained from simulating 495 plants in the same virtual environment where the simulations were performed (Figure 39).

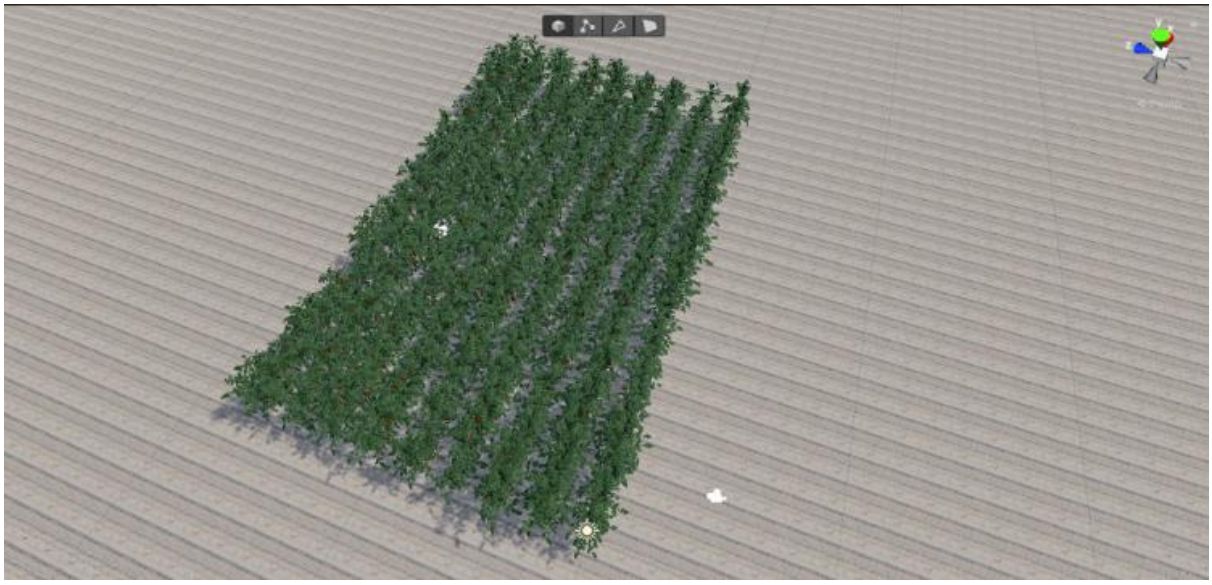


Figure 37: Simulation of 9 rows of tomatoes, with 55 plants per row, performed to obtain the standard deviation of tomato counts per plant.

Considering the *LMS* algorithm, since it was the fastest, the time taken to complete the fruit counting task on 15 plants was 192 and 63 seconds, for 1 and 3 agents respectively, meaning that $LMS_{NrAgents=1}$ (i.e., *LMS* strategy with 1 agent) needed 12.8 seconds per plant, while $LMS_{NrAgents=3}$ needed 4.2 seconds per plant. Assuming a battery autonomy of 13 minutes, according to the DJI Tello User Manual, $LMS_{NrAgents=1}$ would need 2.1 cycles of the battery to complete the task, whereas $LMS_{NrAgents=3}$ would need only 0.69. In conclusion, starting with a fully charged battery, using 1 agent would require the battery to be recharged 2 times, in contrast with 3 agents that would allow the task to be completed without pause. The battery cycles required for all the strategies can be found in Table 5.

RMSE did not differ significantly when using a larger number of agents within the same strategy, as presented in Section 5.2, however increasing the number of agents significantly lowered the time to conclude the task.

Table 5: Battery cycles needed to fully perform a fruit counting task on the real-life assumption example.

Exploration approach	Nr of agents	Battery cycles to complete task [-]
LMS	1	2.10
	2	1.05
	3	0.69
CRS	1	2.78
	2	1.66
	3	1.29
RRW	1	3.85
	2	2.52
	3	2.09

6.3 Sub-question D: Exploration algorithms

As mentioned in Section 5.2, the RMSE did not differ significantly between explorations algorithms, meaning that none of the strategies contributed to an increase in the precision of the tomato counting.

The initial hypothesis would be that *LMS* algorithm would result in the lowest RMSE, since an image is captured for every single cell before the simulation ends, meaning that there are images captured from all possible viewpoints. *CRW* should result in the highest RMSE, since the simulation would end when all cells were measured, but not necessarily from all possible viewpoints. *RRW* should result in a lower RMSE than *CRW* because of the attraction factors (Section 4.1.3.2) that were designed to direct the agents towards regions with higher uncertainty of measurements.

It was not observed a trade-off between simulated time and RMSE, as longer flights did not result in significantly lower RMSE. Consequently, the metric used to determine the most efficient strategy was simulation time, which was lowest in *LMS*.

Simulation times were higher in *CRS* and *RRW* due to the agents being trapped in local minima. This is shown by the mean and maximum time steps (Table 6). In *LMS*, because every movement is pre-determined, agents will move exactly one cell horizontally or vertically. With *CRW* and *RRW*, when agents are surrounded by cells that are inaccessible (Section 4.1.1), they will move to the next nearest available cell, which will be located at a distance of two or more cells from the agent's position, increasing the duration of the step.

Table 6: Number of steps, mean and maximum time to perform a step in LMS, CRS and RRW.

Exploration approach	Nr of Agents	Nr of Steps	Mean step time	Maximum step time [s]
LMS	1	285	0,67	0,67
	2	142	0,67	0,67
	3	94	0,67	0,67
CRS	1	276	0,92	6,17
	2	140	1,08	6,74
	3	93	1,25	8,16
RRW	1	281	1,20	11,68
	2	142	1,51	11,85
	3	95	1,79	13,87

The reason for local minima to occur especially in *RRW* is due to a design flaw of the combination between the attraction factors and the fact that agents are not allowed to revisit cells. The two attraction factors (i) potential tomato cells and (ii) flags were designed to increase the speed at which tomatoes are discovered and increase the measurement accuracy, respectively. These attraction factors would achieve their goal by (i) directing agents towards cells that are adjacent to where other tomatoes were found and (ii) by directing drones to perform more measurements around cells which had dubious tomato count results. The placement of these attraction factors caused the agents to move in circles or semi-circles (Figure 16 and Figure 17). However, because agents were not allowed to visit any cell twice, after visiting cells marked with attraction factors, agents would isolate themselves from the rest of the map.

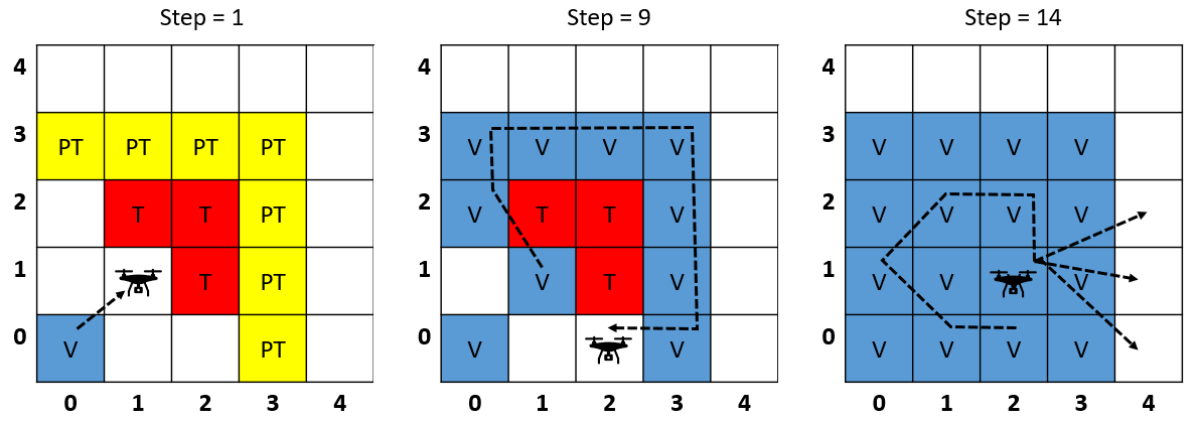


Figure 38: Schematic representation of the local minima problem. V = Visited cells, T = cells where tomatoes were detected, PT = cells that potentially contain tomatoes.

This problematic is represented in Figure 40, where an agent spawned in cell [0, 0] at step = 0 moves to cell [1, 1] in step = 1 and measures tomatoes in cells [1, 2], [2, 2] and [2, 1], resulting in the unmeasured cells on the vicinity of the tomato cells (T) to be marked as potentially containing tomatoes (PT). Each PT cell will attract the agent until they are measured, causing the agent to move towards them. After visiting each one of them in step = 9, if the agent happens to move to cell [2, 0] instead of cell [4, 0], it will be trapped in a local minimum, where it will visit every unvisited cell before moving out of this zone (i.e., even cells where tomatoes were detected, but were not visited by the agent, are subjected to be visited). Also, after visiting all cells inside this local minimum, the next cell visited by the agent (e.g., step = 14) will be a cell with coordinates [4, y] or [x, 4], which are at a distance larger than one cell from any cell inside this region, causing the step time to increase.

One option to solve the problem of agents being trapped inside local minima would be to allow the agents to revisit cells, thus breaking the barrier formed by visited cells. Other alternatives include using virtual waypoints, instead of marking multiple cells with attraction factors. Virtual waypoints were used in other studies with drone path planning which were facing issues with local minima (Hao et al., 2023; W. Yang et al., 2021). Instead of having to deal with multiple targets at the same time, which could cause local minima, the agents would be presented with one virtual target at a time to guarantee that they would move towards the direction of interest. Other studies, suggest adding a time-based coefficient to the exploration approach (Huang et al., 2020). In this case, there could be a repulsion factor that repels agents from a certain region if they are surveying the area for a long period without acquiring new information, or there could be a time-based coefficient that increases the bias towards attraction factors which were not surveyed for a long period of time.

Discovery ratio (DR) was the only metric where *LMS* was outperformed by the other exploration approaches when the number of agents was equal to 1 and 2. With these levels of number of agents, *LMS* had a later start at discovering tomatoes in the simulation environment. This is attributed to the fact that with these number of agents in *LMS*, the starting point of agents were at the extremities of the map, regions where the simulated tomato plants usually

do not bear fruits (Figure 19). With 3 agents in *LMS*, one of the agents was set to initiate the mission in the middle of the map to allow for an equal division of it into three parts (Figure 41), which provided an initial advantage on DR.

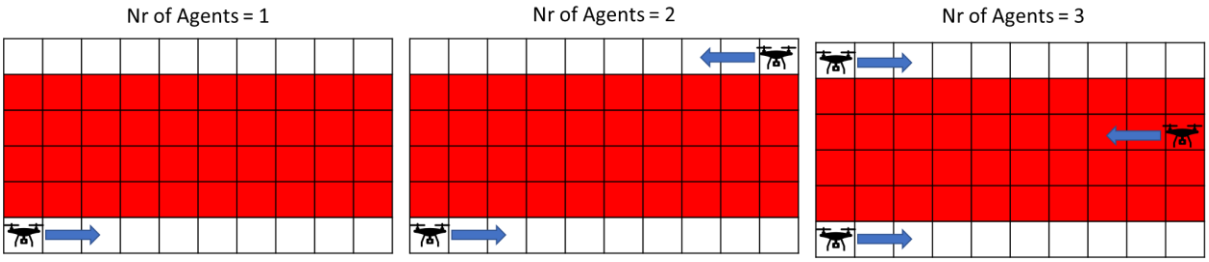


Figure 39: Starting points from agents in LMS. Red cells represent the region where tomatoes were more likely to spawn. The blue arrow indicates the initial movement direction for each agent.

This indicates that the starting position also influences the performance of the exploration approaches. Setting the starting point of agents in *LMS* to the central rows of the map would even increase the speed of the DR for this exploration strategy.

6.4 Main question: opportunities and challenges of drone swarms in greenhouse horticulture

The usage of drone swarms for fruit counting in a greenhouse horticulture setting has shown to improve the speed of the task when compared to a scenario where one drone is used. According to the simulations performed in this study, a swarm composed of three agents would be enough to perform counting of a representative sample in a greenhouse with standard commercial dimensions, demonstrating the opportunity of this approach in a real-life situation.

However, there are still challenges on defining a more efficient exploration approach for this task. The results of this study indicate that a lawnmower navigation with 3 agents is the most efficient strategy, with the lowest required time, fastest discovery ratio progression and equal RMSE when compared with any other strategy.

Other than allocating different shares of area of the exploration environment between the different agents, which intuitively reduces the required time, *LMS* does not make usage of other benefits of drone swarms. The exploration approach of *RRW* was designed as a collaborative strategy where agents would identify regions of interest on the map, so that agents that were currently exploring regions without targets could shift their attention to the regions where reinforce was needed. Even so, this approach did not result in a good performance with the settings that were tested in this study.

Challenges, such as local minima and design choices, impaired the performance of the *RRW* approach, which was supposed to take advantage of drone swarm perks, such as allocating tasks and decision making. Nonetheless, with some adjustments, the Alternative strategy (*ALT*) was able to have an improvement in performance compared to *RRW*.

There are still opportunities to improve the performance of using drone swarms in a greenhouse horticulture context by refining the design choices to make a better use of the potential of multiple agents. One of these opportunities would be to implement task assignment for the UAVs (Qian et al., 2023), by having part of the agents perform a preliminary count of

tomatoes in a lawnmower style navigation, while identifying regions with higher uncertainty and assigning other agents to verify the count in these regions. This verification could be performed by varying the pitch and yaw angle of the UAVs, allowing snapshots from different viewpoint angles, which could reduce the RMSE. In this study, the agents were designed to always face the plant row at the same orthogonal angle, leading to no significant difference in counting accuracy between the strategies. However, other studies indicate the potential of using multiple viewpoint angles from the agents in a drone swarm to reduce the uncertainty of information acquisition (Aburasain et al., 2020; Tahir et al., 2019; Zheng et al., 2023; Zualkernan et al., 2023).

6.5 Reality gaps

In the present study, simulations were performed in a virtual environment due to the practicality of repeating multiple runs, while avoiding the logistical constraints of arranging real-life equipment and crops. Because of that, the experiments conduced in this study are not yet suitable to be repeated in a real-life scenario. There are still factors that need to be considered before this transition from a virtual to a real environment, also known as reality-gaps.

6.5.1 Positioning and tracking of UAVs

To navigate across the environment, it is necessary to have information about the real-time position of each UAV in relation to the map. For outdoor UAVs, the most common method to track their positioning is through the global navigation satellite system (GNSS) in combination with real-time kinematic (RTK) equipment for correction of their coordinates, allowing centimetre-level precision. In three-dimensional complex environments, however, signal blockages may pose a challenge to GNSS positioning (J. Zhou et al., 2023).

For confined spaces, technologies such as Light Detecting And Ranging (LiDAR) sensors can be used to estimate the position of drones. LiDAR sensors consist of an equipment that emits infrared light towards multiple directions and estimates the three-dimensional position of the aircraft based on the time taken by the light to be reflected to the sensor (N. Li et al., 2022). It is a precise and robust technology that is capable of functioning in any light condition and recent advances have turned this equipment lighter and affordable (Aldao et al., 2022).

Other methods used in confined space include the integration of WiFi and inertial measurement unit (IMU), which are present in most commercial drones, for UAV positioning. This method, although cheaper than LiDAR, showed a mean error of 1.37 m (Z. Li & Zhang, 2022), which is unsuitable for the type of mission simulated in this study, where the viewpoints are distant 0.29 m from each other.

6.5.2 UAV communication

Coordinating multiple UAVs in an autonomous way requires a communication system capable of assigning actions to each of the agents of the swarm. This system should be able to receive the information captured by the agents (i.e., snapshots), process the information (i.e., object detection neural network) and return commands (i.e., coordinates for the next move) (Campion et al., 2019).

The most commonly used architecture for drone swarm communication consists of a ground control software (GCS) that receives and sends telemetry data in the form of velocity

vectors through a transceiver. This type of communication system, where a central software controls each UAV individually is called infrastructure-based swarm architecture and it usually utilizes unlicensed radio frequency bands (e.g., 900 MHz). This architecture may be prone to failure when UAVs are out of the communication range of the GCS but is unlikely to happen in close-range missions such as the object of the present study (Campion et al., 2019).

6.5.3 Collision avoidance

Another aspect of real-life multi-agent systems which must be considered in the exploration approach is collision avoidance. When UAVs are moving in a three-dimensional space, their path must consider the possible collisions with static objects (e.g., crops, structures) and dynamic objects (e.g., other UAVs). This is especially relevant in confined spaces with many agents (Arul et al., 2019).

In the present study, it is assumed that all agents are positioned between two plant rows, at a safe distance from any vegetable material. However, it is still necessary to prevent collisions between UAVs.

One of the most consolidated collision avoidance algorithms used in drone swarms is the optimal reciprocal collision avoidance (ORCA). This algorithm uses the principle of velocity space to avoid collisions (James et al., 2020). In practice, if two or more agents are in a route of collision, their velocities will be adjusted to prevent it.

Another method of collision avoidance in UAV swarms is the Force-based Motion Planning (FMP). This algorithm is based on the concept of force fields, where each agent has a repulsion force towards other agents, similar to the mechanism of repulsion vectors used in *RRW*. The difference between *RRW*'s repulsion vector and FMP is that in FMP the repulsion force has a radius of action and will only influence other agents when these are within the range of this radius (Semnani et al., 2020).

6.5.4 Sensor noise

Sensor noise is an inherent issue of any real-life sensor (He et al., 2006). In a scenario where UAVs are used to capture images in a greenhouse crop production, noise may derive from the characteristics of the camera, the motion and vibrations of the UAV, light conditions or even from the humidity of the environment.

Noise impacts on the quality of the images captured and thus in the performance of the detection of tomatoes, having a direct effect on the RMSE of the measurement. To enable the replication of the exploration designs discussed in the present work in a real-life scenario, it would be necessary to simulate artificial noise and evaluate its effect in the performance of the designs. One alternative is to use white Gaussian noise to mimic the effect of random natural processes on the snapshots taken during the simulations (F. Li et al., 2023).

6.6 Future research

Ideally, the next steps of this research will involve the implementation of the reality-gaps and the analysis of their implications in the mission performance. The addition of artificial sensor noise may cause a decrease in counting accuracy, thus elevating the importance to design an exploration approach that minimizes RMSE. There would also be the necessity to evaluate the trade-offs between increasing speed for a faster mission completion and the impacts that it would cause in RMSE due to increasing noise from the sensor's motion. The addition of collision avoidance algorithm may also impact in the performance of the

exploration approaches where agents are susceptible to crossing paths (i.e., *CRW* and *RRW*) due to the extra time needed to avoid collisions.

New exploration approaches should also be experimented with to fully explore the potential of drone swarms in greenhouse fruit counting, which the present study failed to do. Strategies such as information gain algorithms could further help to decrease RMSE significantly, since it presented positive results in weed counting (Carbone et al., 2022).

Furthermore, one of the promising results obtained from this study was the use of artificially generated images used to train a fruit detection model capable of detecting fruits from real-life images. The possibility to create a large dataset with readily annotated images automatically has great potential in the field of machine vision in agricultural research as a fast and cheap solution for model training data acquisition.

7 Conclusion

The use of drone swarms for fruit counting in a simulated tomato greenhouse demonstrated a performance enhancement over single-drone approaches by reducing the total mission time in 67% and 108% for the *LMS* and *RRW* exploration approaches, respectively. When comparing exploration approaches, the baseline *LMS* was superior to *RRW* in all levels of swarm size, achieving a total simulation time of 192, 96 and 63 seconds for 1, 2 and 3 agents, respectively. The combination of *LMS* with 3 agents was considered to be the most efficient strategy in terms of mission time.

In terms of RMSE, the other metric used, there were no significant differences between any of the strategies due to design flaws that did not explore the full potential of drone swarms, such as utilizing different snapshot angles.

In conclusion, drone swarms present opportunities to be used in greenhouse fruit counting, due to its time saving potential, but further investigation should be performed on accuracy improvement. The challenges that this technique may face are mostly related to the confined space of navigation, which may limit the viewpoint angles and increase the susceptibility of collisions.

8 References

Aburasain, R. Y., Edirisinghe, E. A., & Albatay, A. (2020). Palm tree detection in drone images using deep convolutional neural networks: Investigating the effective use of YOLO v3. *Conference on Multimedia, Interaction, Design and Innovation*, 21–36.

Afonso, M., Fonteijn, H., Fiorentin, F. S., Lensink, D., Mooij, M., Faber, N., Polder, G., & Wehrens, R. (2020). Tomato Fruit Detection and Counting in Greenhouses Using Deep Learning. *Frontiers in Plant Science*, *11*. https://doi.org/10.3389/fpls.2020.571299

Albani, D., IJsselmuiden, J., Haken, R., & Trianni, V. (2017). Monitoring and mapping with robot swarms for agricultural applications. *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 1–6. https://doi.org/10.1109/AVSS.2017.8078478

Albani, D., Manoni, T., Arik, A., Nardi, D., & Trianni, V. (2019). Field coverage for weed mapping: toward experiments with a UAV swarm. *Bio-Inspired Information and Communication Technologies: 11th EAI International Conference, BICT 2019, Pittsburgh, PA, USA, March 13–14, 2019, Proceedings 11*, 132–146.

Albani, D., Manoni, T., Nardi, D., & Trianni, V. (2018). Dynamic UAV swarm deployment for non-uniform coverage. *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems*, 523–531.

Albani, D., Nardi, D., & Trianni, V. (2017). Field coverage and weed mapping by UAV swarms. *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 4319–4325.

Aldao, E., González-de Santos, L. M., & González-Jorge, H. (2022). LiDAR Based Detect and Avoid System for UAV Navigation in UAM Corridors. *Drones*, *6*(8). https://doi.org/10.3390/drones6080185

Ali, Q., Erkan, M., & Jan, I. (2017). Morphological and agronomic characterization of tomato under field conditions. *Pure and Applied Biology (PAB)*, *6*(3), 1021–1029. https://www.thepab.org/index.php/journal/article/view/203

Alruwaili, M., Siddiqi, M. H., Khan, A., Azad, M., Khan, A., & Alanazi, S. (2022). RTF-RCNN: An Architecture for Real-Time Tomato Plant Leaf Diseases Detection in Video Streaming Using Faster-RCNN. *Bioengineering*, *9*(10). https://doi.org/10.3390/bioengineering9100565

Amundson, S., Deyton, D. E., Kopsell, D. A., Hitch, W., Moore, A., & Sams, C. E. (2012). Optimizing Plant Density and Production Systems to Maximize Yield of Greenhouse-grown 'Trust' Tomatoes. *HortTechnology Hortte*, *22*(1), 44–48. https://doi.org/10.21273/HORTTECH.22.1.44

Ankit, K., Kolathaya, S. N. Y., & Ghose, D. (2021). Multi-Agent Collaborative Framework for Automated Agriculture. *2021 15th International Conference on Advanced Computing and Applications (ACOMP)*, 30–37. https://doi.org/10.1109/ACOMP53746.2021.00011

Arafat, M. Y., Alam, M. M., & Moh, S. (2023). Vision-Based Navigation Techniques for Unmanned Aerial Vehicles: Review and Challenges. *Drones 2023, Vol. 7, Page 89*, *7*(2), 89. https://doi.org/10.3390/DRONES7020089

Arul, S. H., Sathyamoorthy, A. J., Patel, S., Otte, M., Xu, H., Lin, M. C., & Manocha, D. (2019). LSwarm: Efficient collision avoidance for large swarms with coverage constraints in complex urban scenes. *IEEE Robotics and Automation Letters*, *4*(4), 3940–3947.

Athearn, K. R., Hochmuth, R. C., Laughlin, W. L., Clark, J. L., & others. (2018). Economic analysis of small-scale greenhouse tomato production in Florida. *Proceedings of the Florida State Horticultural Society*, *131*, 99–102.

Aydoğan, Y. (2018). Drone technology in agricultural mechanization. *Mechanization in Agriculture & Conserving of the Resources*, *64*(2), 36–39.

Bechar, A., Yosef, S., Netanyahu, S., & Edan, Y. (2007). Improvement of work methods in tomato greenhouses using simulation. *Transactions of the ASABE*, *50*(2), 331–338.

Bhattarai, U., & Karkee, M. (2022). A weakly-supervised approach for flower/fruit counting in apple orchards. *Computers in Industry*, *138*, 103635. https://doi.org/https://doi.org/10.1016/j.compind.2022.103635

Biczyski, M., Sehab, R., Whidborne, J. F., Krebs, G., & Luk, P. (2020). Multirotor Sizing Methodology with Flight Time Estimation. *Journal of Advanced Transportation*, *2020*. https://doi.org/10.1155/2020/9689604

Campion, M., Ranganathan, P., & Faruque, S. (2019). UAV swarm communication and control architectures: a review. *Journal of Unmanned Vehicle Systems*, *7*(2), 93–106. https://doi.org/10.1139/juvs-2018-0009

Carbone, C., Albani, D., Magistri, F., Ognibene, D., Stachniss, C., Kootstra, G., Nardi, D., & Trianni, V. (2022). Monitoring and Mapping of Crop Fields with UAV Swarms Based on Information Gain. In F. Matsuno, S. Azuma, & M. Yamamoto (Eds.), *Distributed Autonomous Robotic Systems - 15th International Symposium, 2022* (pp. 306–319). Springer Nature. https://doi.org/10.1007/978-3-030-92790-5_24

Carbone, C., Garibaldi, O., & Kurt, Z. (2018). Swarm Robotics as a Solution to Crops Inspection for Precision Agriculture. *KnE Engineering*, *3*, 552. https://doi.org/10.18502/keg.v3i1.1459

Cimino, M. G. C. A., Lazzeri, A., & Vaglini, G. (2016). Using differential evolution to improve pheromone-based coordination of swarms of drones for collaborative target detection. *ICPRAM*, 605–610.

De Rango, F., Palmieri, N., Tropea, M., & Potrino, G. (2017). UAVs Team and Its Application in Agriculture: A Simulation Environment. *Simultech*, *2017*, 374–379.

Dorigo, M., Birattari, M., & Brambilla, M. (2014). Swarm robotics. *Scholarpedia*, *9*(1), 1463. https://doi.org/10.4249/scholarpedia.1463

Du, J. (2018). Understanding of Object Detection Based on CNN Family and YOLO. *Journal of Physics: Conference Series*, *1004*(1), 12029. https://doi.org/10.1088/1742-6596/1004/1/012029

Duan, T., Chapman, S. C., Guo, Y., & Zheng, B. (2017). Dynamic monitoring of NDVI in wheat agronomy and breeding trials using an unmanned aerial vehicle. *Field Crops Research*, *210*, 71–80. https://doi.org/https://doi.org/10.1016/j.fcr.2017.05.025

Dutta, A., Roy, S., Kreidl, O. P., & Bölöni, L. (2021). Multi-Robot Information Gathering for Precision Agriculture: Current State, Scope, and Challenges. *IEEE Access*, *9*, 161416–161430. https://doi.org/10.1109/ACCESS.2021.3130900

Egi, Y., Hajyzadeh, M., & Eyceyurt, E. (2022). Drone-Computer Communication Based Tomato Generative Organ Counting Model Using YOLO V5 and Deep-Sort. *Agriculture*, *12*(9), 1290.

Fatnassi, H., Leyronas, C., Boulard, T., Bardin, M., & Nicot, P. (2009). Dependence of greenhouse tunnel ventilation on wind direction and crop height. *Biosystems Engineering*, *103*(3), 338–343. https://doi.org/https://doi.org/10.1016/j.biosystemseng.2009.03.005

Fawzia Rahim, U., & Mineno, H. (2022). Highly Accurate Tomato Maturity Recognition: Combining Deep Instance Segmentation, Data Synthesis and Color Analysis. *2021 4th Artificial Intelligence and Cloud Computing Conference*, 16–23. https://doi.org/10.1145/3508259.3508262

Fu, L., Majeed, Y., Zhang, X., Karkee, M., & Zhang, Q. (2020). Faster R–CNN–based apple detection in dense-foliage fruiting-wall trees using RGB and depth features for robotic harvesting. *Biosystems Engineering*, *197*, 245–256. https://doi.org/https://doi.org/10.1016/j.biosystemseng.2020.07.007

Gao, F., Fu, L., Zhang, X., Majeed, Y., Li, R., Karkee, M., & Zhang, Q. (2020). Multi-class fruit-on-plant detection for apple in SNAP system using Faster R-CNN. *Computers and Electronics in Agriculture*, *176*, 105634. https://doi.org/https://doi.org/10.1016/j.compag.2020.105634

Ge, Y., Lin, S., Zhang, Y., Li, Z., Cheng, H., Dong, J., Shao, S., Zhang, J., Qi, X., & Wu, Z. (2022). Tracking and Counting of Tomato at Different Growth Period Using an Improving YOLO-Deepsort Network for Inspection Robot. *Machines*, *10*(6). https://doi.org/10.3390/machines10060489

Goudarzi, S., Kama, N., Anisi, M. H., Zeadally, S., & Mumtaz, S. (2019). Data collection using unmanned aerial vehicles for Internet of Things platforms. *Computers and Electrical Engineering*, *75*, 1–15. https://doi.org/10.1016/j.compeleceng.2019.01.028

Hao, G., Lv, Q., Huang, Z., Zhao, H., & Chen, W. (2023). UAV Path Planning Based on Improved Artificial Potential Field Method. *Aerospace*, *10*(6). https://doi.org/10.3390/aerospace10060562

He, Z., Iyer, R. V, & Chandler, P. R. (2006). Vision-based UAV flight control and obstacle avoidance. *2006 American Control Conference*, 5 pp.-. https://doi.org/10.1109/ACC.2006.1656540

Hu, C., Liu, X., Pan, Z., & Li, P. (2019). Automatic Detection of Single Ripe Tomato on Plant Combining Faster R-CNN and Intuitionistic Fuzzy Set. *IEEE Access*, *7*, 154683–154696. https://doi.org/10.1109/ACCESS.2019.2949343

Hu, P., Chapman, S. C., Wang, X., Potgieter, A., Duan, T., Jordan, D., Guo, Y., & Zheng, B. (2018). Estimation of plant height using a high throughput phenotyping platform based on unmanned aerial vehicle and self-calibration: Example for sorghum breeding. *European Journal of Agronomy*, *95*, 24–32. https://doi.org/https://doi.org/10.1016/j.eja.2018.02.004

Huang, H., Savkin, A. V, & Li, X. (2020). Reactive Autonomous Navigation of UAVs for Dynamic Sensing Coverage of Mobile Ground Targets. *Sensors*, *20*(13). https://doi.org/10.3390/s20133720

James, S., Raheb, R., & Hudak, A. (2020). Impact of packet loss to the motion of autonomous UAV swarms. *2020 AIAA/IEEE 39th Digital Avionics Systems Conference (DASC)*, 1–9.

Karion, A., Sweeney, C., Pétron, G., Frost, G., Michael Hardesty, R., Kofler, J., Miller, B. R., Newberger, T., Wolter, S., Banta, R., Brewer, A., Dlugokencky, E., Lang, P., Montzka, S. A., Schnell, R., Tans, P., Trainer, M., Zamora, R., & Conley, S. (2013). Methane emissions estimate from airborne measurements over a western United States natural gas field. *Geophysical Research Letters*, *40*(16), 4393–4397. https://doi.org/10.1002/grl.50811

Kingston, Z., Moll, M., & Kavraki, L. E. (2018). Sampling-Based Methods for Motion Planning with Constraints. *Annual Review of Control, Robotics, and Autonomous Systems*, *1*(1), 159–185. https://doi.org/10.1146/annurev-control-060117-105226

Lauri, M., Pajarinen, J., Peters, J., & Frintrop, S. (2020). Multi-Sensor Next-Best-View Planning as Matroid-Constrained Submodular Maximization. *IEEE Robotics and Automation Letters*, *5*(4), 5323–5330. https://doi.org/10.1109/lra.2020.3007445

Lawal, M. O. (2021). Tomato detection based on modified YOLOv3 framework. *Scientific Reports*, *11*(1), 1447. https://doi.org/10.1038/s41598-021-81216-5

Li, F., Fang, F., Li, Z., & Zeng, T. (2023). Single image noise level estimation by artificial noise. *Signal Processing*, *213*, 109215. https://doi.org/https://doi.org/10.1016/j.sigpro.2023.109215

Li, H., Zhang, X., Meng, W., & Ge, L. (2017). Visualization of Tomato Growth Based on Dry Matter Flow. *International Journal of Computer Games Technology*, *2017*, 2302731. https://doi.org/10.1155/2017/2302731

Li, N., Ho, C. P., Xue, J., Lim, L. W., Chen, G., Fu, Y. H., & Lee, L. Y. T. (2022). A progress review on solid-state LiDAR and nanophotonics-based LiDAR sensors. *Laser & Photonics Reviews*, *16*(11), 2100511.

Li, Z., & Zhang, Y. (2022). Constrained ESKF for UAV Positioning in Indoor Corridor Environment Based on IMU and WiFi. *Sensors*, *22*(1). https://doi.org/10.3390/s22010391

Liang, H., Lee, S.-C., Bae, W., Kim, J., & Seo, S. (2023). Towards UAVs in Construction: Advancements, Challenges, and Future Directions for Monitoring and Inspection. *Drones 2023, Vol. 7, Page 202*, *7*(3), 202. https://doi.org/10.3390/DRONES7030202

Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. *Proceedings of the IEEE International Conference on Computer Vision*, 2980–2988.

Liu, G., Nouaze, J. C., Touko Mbouembe, P. L., & Kim, J. H. (2020). YOLO-Tomato: A Robust Algorithm for Tomato Detection Based on YOLOv3. *Sensors*, *20*(7). https://doi.org/10.3390/s20072145

Machefer, M., Lemarchand, F., Bonnefond, V., Hitchins, A., & Sidiropoulos, P. (2020). Mask R-CNN Refitting Strategy for Plant Counting and Sizing in UAV Imagery. *Remote Sensing*, *12*(18). https://doi.org/10.3390/rs12183015

Mavridou, E., Vrochidou, E., Papakostas, G. A., Pachidis, T., & Kaburlasos, V. G. (2019). Machine Vision Systems in Precision Agriculture for Crop Farming. *Journal of Imaging*, *5*(12). https://doi.org/10.3390/jimaging5120089

Monica, R., Aleotti, J., & Piccinini, D. (2019). Humanoid Robot Next Best View Planning under Occlusions Using Body Movement Primitives. *IEEE International Conference on Intelligent Robots and Systems*, 2493–2500. https://doi.org/10.1109/IROS40897.2019.8968239

Naazare, M., Rosas, F. G., & Schulz, D. (2022). Online Next-Best-View Planner for 3D-Exploration and Inspection with a Mobile Manipulator Robot. *IEEE Robotics and Automation Letters*, *7*(2), 3779–3786. https://doi.org/10.1109/LRA.2022.3146558

Nemitz, M. P., Marcotte, R. J., Sayed, M. E., Ferrer, G., Hero, A. O., Olson, E., & Stokes, A. A. (2018). Multi-Functional Sensing for Swarm Robots Using Time Sequence Classification: HoverBot, an Example. *Frontiers in Robotics and AI*, *5*(MAY), 17. https://doi.org/10.3389/frobt.2018.00055

Nuske, S., Wilshusen, K., Achar, S., Yoder, L., & Singh, S. (2014). Automated visual yield estimation in vineyards. *Journal of Field Robotics*, *31*(5), 837–860. https://doi.org/10.1002/rob.21541

Oliveira, W. F. de, Santos, S. R. dos, Struiving, T. B., & Silva, L. A. da. (2022). Citrus orchards under formation evaluated by UAV-Based RGB Imagery. *Scientia Agricola*, *79*(5), e20210052. https://doi.org/10.1590/1678-992X-2021-0052

Palazzolo, E., & Stachniss, C. (2018). Effective Exploration for MAVs Based on the Expected Information Gain. *Drones*, *2*(1), 9. https://doi.org/10.3390/drones2010009

Pham, V., Pham, C., & Dang, T. (2020). Road Damage Detection and Classification with Detectron2 and Faster R-CNN. *2020 IEEE International Conference on Big Data (Big Data)*, 5592–5601. https://doi.org/10.1109/BigData50022.2020.9378027

Qian, F., Su, K., Liang, X., & Zhang, K. (2023). Task Assignment for UAV Swarm Saturation Attack: A Deep Reinforcement Learning Approach. *Electronics*, *12*(6). https://doi.org/10.3390/electronics12061292

Rahnemoonfar, M., & Sheppard, C. (2017). Deep Count: Fruit Counting Based on Deep Simulated Learning. *Sensors*, *17*(4). https://doi.org/10.3390/s17040905

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in Neural Information Processing Systems*, *28*.

Roldán, J. J., Garcia-Aunon, P., Garzón, M., De León, J., Del Cerro, J., & Barrientos, A. (2016). Heterogeneous Multi-Robot System for Mapping Environmental Variables of Greenhouses. *Sensors*, *16*(7). https://doi.org/10.3390/s16071018

Saffre Fabrice and Hildmann, H. and K. H. and L. T. (2022). Self-Swarming for Multi-Robot Systems Deployed for Situational Awareness. In P. and N. N. Lipping Tarmo and Linna (Ed.), *New Developments and Environmental Applications of Drones* (pp. 51–72). Springer International Publishing.

Selin, M., Tiger, M., Duberg, D., Heintz, F., & Jensfelt, P. (2019). Efficient Autonomous Exploration Planning of Large-Scale 3-D Environments. *IEEE Robotics and Automation Letters*, *4*(2), 1699–1706. https://doi.org/10.1109/LRA.2019.2897343

Semnani, S. H., Liu, H., Everett, M., de Ruiter, A., & How, J. P. (2020). Multi-Agent Motion Planning for Dense and Dynamic Environments via Deep Reinforcement Learning. *IEEE Robotics and Automation Letters*, *5*(2), 3221–3226. https://doi.org/10.1109/LRA.2020.2974695

Skobelev, P., Budaev, D., Gusev, N., & Voschuk, G. (2018). Designing Multi-agent Swarm of UAV for Precise Agriculture. In J. Bajo, J. M. Corchado, E. M. Navarro Martínez, E. Osaba Icedo, P. Mathieu, P. Hoffa-Dąbrowska, E. del Val, S. Giroux, A. J. M. Castro, N. Sánchez-Pi, V. Julián, R. A. Silveira, A. Fernández, R. Unland, & R. Fuentes-Fernández (Eds.), *Highlights of Practical Applications of Agents, Multi-Agent Systems, and Complexity: The PAAMS Collection* (pp. 47–59). Springer International Publishing.

Sousselier, T., Dreo, J., & Sevaux, M. (2015). Line formation algorithm in a swarm of reactive robots constrained by underwater environment. *Expert Systems with Applications*, *42*(12), 5117–5127. https://doi.org/https://doi.org/10.1016/j.eswa.2015.02.040

Tahir, A., Böling, J., Haghbayan, M.-H., Toivonen, H. T., & Plosila, J. (2019). Swarms of Unmanned Aerial Vehicles — A Survey. *Journal of Industrial Information Integration*, *16*, 100106. https://doi.org/https://doi.org/10.1016/j.jii.2019.100106

Testa, R., Trapani, A. M. di, Sgroi, F., & Tudisca, S. (2014). Economic Sustainability of Italian Greenhouse Cherry Tomato. *Sustainability*, *6*(11), 7967–7981. https://doi.org/10.3390/su6117967

Trianni, V., & López-Ibáñez, M. (2015). Advantages of Task-Specific Multi-Objective Optimisation in Evolutionary Robotics. *PLOS ONE*, *10*(8), e0136406. https://doi.org/10.1371/journal.pone.0136406

Underwood, J. P., Hung, C., Whelan, B., & Sukkarieh, S. (2016). Mapping almond orchard canopy volume, flowers, fruit and yield using lidar and vision sensors. *Computers and Electronics in Agriculture*, *130*, 83–96. https://doi.org/https://doi.org/10.1016/j.compag.2016.09.014

Valente, J., Sari, B., Kooistra, L., Kramer, H., & Mücher, S. (2020). Automated crop plant counting from very high-resolution aerial imagery. *Precision Agriculture*, *21*(6), 1366–1384. https://doi.org/10.1007/s11119-020-09725-3

Velumani, K., Lopez-Lozano, R., Madec, S., Guo, W., Gillet, J., Comar, A., & Baret, F. (2021). Estimates of maize plant density from UAV RGB images using Faster-RCNN detection model: impact of the spatial resolution. *Plant Phenomics*, *2021*.

Walsh, K., McGlone, V. A., & Wohlers, M. (2022). *Sampling and statistics in assessment of fresh produce*.

Wang, P., Meng, F., Donaldson, P., Horan, S., Panchy, N. L., Vischulis, E., Winship, E., Conner, J. K., Krysan, P. J., Shiu, S.-H., & Lehti-Shiu, M. D. (2022). High-throughput measurement of plant fitness traits with an object detection method using Faster R-CNN. *New Phytologist*, *234*(4), 1521–1533. https://doi.org/https://doi.org/10.1111/nph.18056

Wang, Q., & Qi, F. (2019). Tomato Diseases Recognition Based on Faster RCNN. *2019 10th International Conference on Information Technology in Medicine and Education (ITME)*, 772–776. https://doi.org/10.1109/ITME.2019.00176

Wang, S., Mao, Z., Zeng, C., Gong, H., Li, S., & Chen, B. (2010). A new method of virtual reality based on Unity3D. *2010 18th International Conference on Geoinformatics*, 1–5. https://doi.org/10.1109/GEOINFORMATICS.2010.5567608

Wang, Y., Li, X., Zhuang, X., Li, F., & Liang, Y. (2023). A Sampling-Based Distributed Exploration Method for UAV Cluster in Unknown Environments. *Drones*, *7*(4). https://doi.org/10.3390/drones7040246

Wang, Z., Walsh, K., & Koirala, A. (2019). Mango Fruit Load Estimation Using a Video Based MangoYOLO-Kalman Filter-Hungarian Algorithm Method. *Sensors (Basel)*, *19*(12). https://doi.org/10.3390/s19122742

Widiyanto, S., Wardani, D. T., & Wisnu Pranata, S. (2021). Image-Based Tomato Maturity Classification and Detection Using Faster R-CNN Method. *2021 5th International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*, 130–134. https://doi.org/10.1109/ISMSIT52890.2021.9604534

Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y., & Girshick, R. (2019). *Detectron2*. https://github.com/facebookresearch/detectron2

Yamauchi, B. (1997). Frontier-based approach for autonomous exploration. *Proceedings of IEEE International Symposium on Computational Intelligence in Robotics and Automation, CIRA*, 146–151. https://doi.org/10.1109/CIRA.1997.613851

Yang, J., Rebello, J., & Waslander, S. L. (2023). *Next-Best-View Selection for Robot Eye-in-Hand Calibration*. https://arxiv.org/abs/2303.06766v1

Yang, W., Wu, P., Zhou, X., Lv, H., Liu, X., Zhang, G., Hou, Z., & Wang, W. (2021). Improved Artificial Potential Field and Dynamic Window Method for Amphibious Robot Fish Path Planning. *Applied Sciences*, *11*(5). https://doi.org/10.3390/app11052114

Zhang, Y., Song, C., & Zhang, D. (2020). Deep Learning-Based Object Detection Improvement for Tomato Disease. *IEEE Access*, *8*, 56607–56614. https://doi.org/10.1109/ACCESS.2020.2982456

Zhang, Z., Heinemann, P. H., Liu, J., Baugher, T. A., & Schupp, J. R. (2016). The development of mechanical apple harvesting technology: A review. *Transactions of the ASABE*, *59*(5), 1165–1180.

Zheng, Z., Wang, S., Gong, B., Su, E., & Huang, H. (2023). Study on angle-only relative navigation for unmanned aerial vehicle formation in GPS-denied environment. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, *237*(10), 2252–2265. https://doi.org/10.1177/09544100221149235

Zhou, J., He, L., & Luo, H. (2023). Real-Time Positioning Method for UAVs in Complex Structural Health Monitoring Scenarios. *Drones*, *7*(3). https://doi.org/10.3390/drones7030212

Zhou, X., Yi, Z., Liu, Y., Huang, K., & Huang, H. (2020). Survey on path and view planning for UAVs. *Virtual Reality & Intelligent Hardware*, *2*(1), 56–69.

Zualkernan, I., Abuhani, D. A., Hussain, M. H., Khan, J., & ElMohandes, M. (2023). Machine Learning for Precision Agriculture Using Imagery from Unmanned Aerial Vehicles (UAVs): A Survey. *Drones*, *7*(6). https://doi.org/10.3390/drones7060382