Robotica (2024), 1–20
doi:10.1017/S0263574724000936

RESEARCH ARTICLE

MonoVisual3DFilter: 3D tomatoes' localisation with monocular cameras using histogram filters

Sandro Augusto Costa Magalhães[1,2], Filipe Neves dos Santos[2], António Paulo Moreira[1,2] and Jorge Manuel Miranda Dias[3,4]

[1]Faculty of Engineering, University of Porto, Porto 4200-465, Portugal
[2]INESC TEC – Instituto de Engenharia de Sistemas e Computadores, Tecnologia e Ciência, Porto 4200-465, Portugal
[3]Institute of Systems and Robotics, Department of Electrical Engineering and Computers, University of Coimbra, Coimbra, Portugal
[4]Khalifa University of Science, Technology, and Research, Abu Dhabi, United Emirates of Arabia (EUA)
Corresponding author: Sandro Augusto Costa Magalhães; Email: sandro.a.magalhaes@inesctec.pt

Received: 25 May 2023; **Revised:** 6 May 2024; **Accepted:** 13 May 2024

Keywords: 3D object detection; pose estimation; position estimation; Bayes filter; robotic manipulator arms; statistical localisation; active perception; active sensing

Abstract
Performing tasks in agriculture, such as fruit monitoring or harvesting, requires perceiving the objects' spatial position. RGB-D cameras are limited under open-field environments due to lightning interferences. So, in this study, we state to answer the research question: "How can we use and control monocular sensors to perceive objects' position in the 3D task space?" Towards this aim, we approached histogram filters (Bayesian discrete filters) to estimate the position of tomatoes in the tomato plant through the algorithm MonoVisual3DFilter. Two kernel filters were studied: the square kernel and the Gaussian kernel. The implemented algorithm was essayed in simulation, with and without Gaussian noise and random noise, and in a testbed at laboratory conditions. The algorithm reported a mean absolute error lower than 10 mm in simulation and 20 mm in the testbed at laboratory conditions with an assessing distance of about 0.5 m. So, the results are viable for real environments and should be improved at closer distances.

1. Introduction

Agriculture is a critical sector in the global economy. Farmers and the agro-food industry have been adapting to meet the demands of the worldwide population, which is increasing fast [1]. Several studies support that the population should keep increasing fast and reach about nine billion people by the year 2050 [2, 3]. Besides the increasing food demands to fulfil the global population [2], the area dedicated to agriculture can only increase marginally, requiring more optimised and precise strategies to improve production and cultivation ratios. These factors, associated with the labour shortage for agricultural tasks [4, 5], ally technology to farming and the agro-food industry.

Several scientific studies in the literature have been proving that robots can support farmers in agricultural tasks [6–8] and overcome the labour shortage. Mobile and intelligent robots can successfully perform tasks such as monitoring and harvesting. However, these robots require dedicated sensors to perceive fruits and other objects and estimate their localisation to the tools.

Recent literature reviews proved that most works for perceiving fruits use RGB-D sensors [9–13]. For instance, Sa *et al.* [14] performed a complete digitalisation of the scene to gather a digital twin of it and easily perceive the fruits and their three-dimension (3D) position. A support vector machine (SVM), based on colour and geometry features, performed a classification of the fruits. In another work, Jun *et al.* [15] used a YOLO v3 to detect the fruits in the scene, and, through an RGB-D camera, digitalised the fruit. Using this information, the authors built the Tool Centre Point algorithm to compute the centre of

Table I. *Comparision with literature review.*

Algorithm	**Deep Learning**	**Model dependency**	**<2 cm**	**Error (cm)**
SilhoNet [25]	√	√	97.5%	–
Nerf-Pose [26]	√	√	–	2.45
GDR-Net [27]	√	√	95.5%	–
[28]	✗	√	–	0.46–2.45
MORE [29]	√	√	93.94%	–
GhostPose [30]	√	√	93.9%	–
Imitrob [31]	√	✗	–	6.5

the fruit and the target point to harvest it. The point about these strategies is that they are being performed under controlled conditions at the laboratory or in controlled greenhouses. Therefore, most of these essays were performed under controlled lightening, ensuring the sensors' correct functioning [10]. RGB-D sensors tend to malfunction under open-field environments due to reflections or intense illumination [10, 16, 17]. So, using auxiliary algorithms and alternative technology is important to overcome the lightning effects.

The previous conceptualisation permits the establishment of a common problem in open-field contexts, which we aim to approach in this study: "How can we use and control monocular sensors to perceive the objects' positioning in the 3D task space?".

Concerning estimating the depth using monocular cameras, most of the more recent works focused in convolution neural networks (CNNs) to infer this relative depth to the sensor [18–23]. Mousavian *et al.* [21] used a CNN to estimate the 3D pose of an object and deployed a MultiBin loss function to optimise the model. Ma *et al.* [19, 20] deployed custom-made CNNs named MonoPointNet and PatchNet to generate 3D images from monocular images and detect objects. Recently, Ranft *et al.* [22] and Birkl *et al.* [18] released a family of CNNs based on MiDaS networks that aimed to estimate the relative depth to the cameras. The networks were trained and essayed on the set of the different datasets of RGB-D data in the literature. Also, Haq *et al.* [23] designed a new regional proposal network (RPN) with geometric constraints to detect 3D objects using monocular cameras. This architecture performed similarly to [19, 20]. Van and Do [24] used a chessboard background and a regression-based CNN to estimate the 3D pose of irregular objects using cuboids. The dependency on a chessboard background to predict the objects poses constraints on the model's applicability for unstructured environments.

In the literature, we can also find purposeful deep learning solutions to extract the pose of the objects directly. Table I reports some examples of algorithms in the state of the art. SilhoNet [25], Nerf-Pose [26], MORE [29], GhostPose [30], and GDR-Net [27] are some of these solutions. SilhoNet reports an overall translation error of about 2.45 cm using a complex state-of-the-art dataset containing multiple objects. Similarly, Nerf-Pose, MORE, GhostPose, and GDR-Net report an overall estimation error lower than 2 cm. Collet and Srinivasa [28] developed the introspective multiview algorithm that could estimate the pose of objects with estimation errors between 0.46 cm and 1.45 cm. These solutions illustrate remarkable results in the literature for objects' pose estimation. However, all of them are model dependent in successfully estimating the pose of an object. An exception to this factor is the Imitrob [31] that analyses the objects' configuration and structure and learns to estimate the pose of the objects from multiple perspectives.

Despite being largely explored in the literature, deep learning-based solutions are data exhaustive, requiring much and varied data with their features well identified and represented. Acquiring data to train deep learning models is expensive and difficult. For natural environments that requires going to the natural scenes and mapping the objects for the robot's context.

Other works used auxiliary sensors to create 3D scenes or depth estimation, such as light detection and rangings (LiDARs) [32]. However, high resolution and quality LiDARs are expensive and increase the effort over robotic manipulators to manoeuvre and pick objects and also constrained by their operation in open-field robotics.

There are still some researchers that opted for using monocular cameras and controlling the path to the objects using visual servoing strategies [33, 34]. Shen *et al.* [13] applied visual servoing using RGB-D cameras. A distinctive work is presented by Xu *et al.* [35], which used the brightness of the environment and the movement of the camera to estimate the depth and reconstruct the structure of the colon during endoscopies.

Probabilistic algorithms are also capable of estimating the objects' poses accurately. Algorithms such as Bayesian histogram filters were explored in the literature to identify and localise objects in the scenes using different kinds of sensors. Bayesian histogram filters are commonly used in robotics for localisation and navigation purposes [36–38], but their potentialities enable it for other aims such as identifying and localising other objects. Sarmento *et al.* [39] applied region histogram filters to identify people, animals and other obstacles in the ultrawideband scene to avoid collisions and to follow people. Also, Engin and Isler [40] used this algorithm to localise random objects. Márton *et al.* [41] used a histogram filter to complement the information provided by a state-of-the-art position estimator and accurately estimate the object's orientation.

The advantage associated with histogram filters is that they are only constrained by the detection capabilities of the algorithms behind the sensors. Therefore, using a well-defined model to perceive the fruits through monocular images, we can translate their 2D relation to the 3D and accurately estimate the objects' positions. Besides, the histogram filters propose a working philosophy similar to triangulation, which has their accuracy and functionality well proved in geospatial and cartography disciplines.

1.1. System and requirements

Greenhouses are a target-designed scene to optimise the production of fruits in environment-controlled conditions. They also have the advantage that the crops can be modelled to better fit the farm and objectives constraints. So, besides the complexity of agricultural scenes, the modelisation of the environment can be simplified in greenhouses, if robust perception algorithms are used. Besides, the greenhouse context actually has the purpose and the need to adapt to human and robotic systems [42, 43], becoming more effective and ergonomic for different operations.

Actual robotised greenhouses are usually operated by robots on trails [44]. However, these environments require the development of robot-specific environments and do not fit in the commonly operated greenhouses. Therefore, robots operating in the most common greenhouses should be composed of wheeled mobile robots. AgRob v16 from INESC TEC (Figure 1) is a wheeled robot based on the Clearpath Husky platform[1] and all-terrain wheels designed for operating under open-field and controlled agricultural environments. This robot is currently being essayed in Douro steep slope vineyards and tomato greenhouses. It has a perception and controlling head that gathers data and information from the environment for navigation and mapping. Additionally, the Robotis Manipulator-H was assembled at the backside to perform tasks in the cultivars, such as monitoring, pruning, or harvesting.

Perception for manipulation is usually performed through eye-in-hand techniques. The Robotis Manipulator-H can handle either monocular or RGB-D cameras. RGB-D cameras can perceive the 3D pose of objects, but they are bigger and more difficult to handle. Besides, RGB-D cameras have difficulty perceiving and estimating the objects' depth in open-field environments due to natural lightning interferences [10, 16, 17]. Therefore, using monocular cameras, the manipulator can perceive the position of the objects from multiple perspectives and estimate the objects' pose. Approaches based on machine learning or statistics are commonly explored in the literature [19–21, 23], albeit statistical approaches are more analytical solutions and with more predictable results.

[1]See Clearpath Robotics 2023. Husky – Unmanned Ground Vehicle. Online https://clearpathrobotics.com/husky-unmanned-ground-vehicle-robot/ [Last accessed on May 12th, 2023]

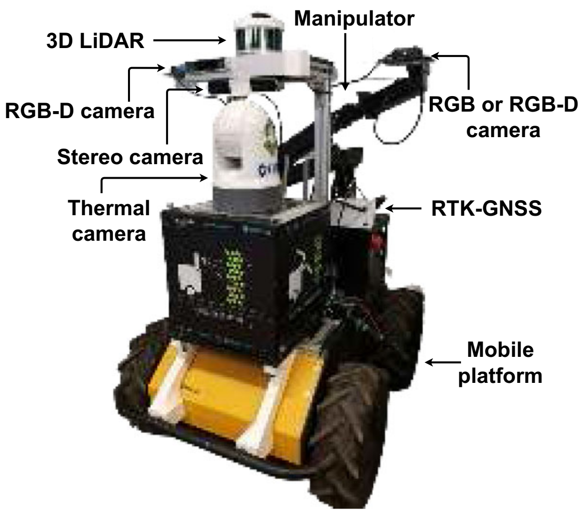

Figure 1. *Robot AgRob v16 from INESC TEC to operate under open-field and controlled agricultural environments.*

Therefore, as reviewed, Bayesian histogram filters are suitable for identifying the 3D position of objects and do not require the model's knowledge. Besides, the histogram filters are more predictable and explainable than deep learning-based solutions, which makes it also easier to readjust to new scenarios. So, for this work, we applied the histogram filter to identify the 3D position of tomatoes in a testbed using a monocular camera assembled in a robotic arm, in a solution called MonoVisual3DFilter. At the current stage, the arm used fixed multi-viewpoint to observe the tomatoes from multiple perspectives.

The current work benefits, in our knowledge, for the first essay in using Bayesian histogram filters for estimating the 3D position of objects in the range of a robotic manipulator using monocular cameras. The implemented algorithm was evaluated for fruit detection under simulation and testbed conditions in the laboratory. Therefore, the current work contributes by

- Introducing and essaying the MonoVisual3DFilter;
- First, apply histogram filters to detect the centre position of objects;
- Apply the algorithm to real-world problems, such as fruit detection in the plants' canopy; and
- Study the effect and advantages of different kernel types.

The following sections are structured: material and methods, results, discussion and conclusion. In section 2, we detail the conditions of the experiments and formalise the algorithm application. Section 3 introduces the different results for the different experiments, which are analysed and discussed, in section 4, and concluded in section 5.

2. Materials and methods

2.1. Real data and simulation

The development and the experiments with the algorithm MonoVisual3DFilter were done under two environments: simulation and a testbed in the laboratory (near real-world conditions).

A simplistic simulation environment was designed using the Ignition Gazebo Simulator.[2] The scene comprises six spheres to assess the algorithm's validity and test during implementation (Figure 2). The

[2]See Open Robotics, "Gazebo," accessed on May 12th, 2023. [Online]. Available: https://gazebosim.org/

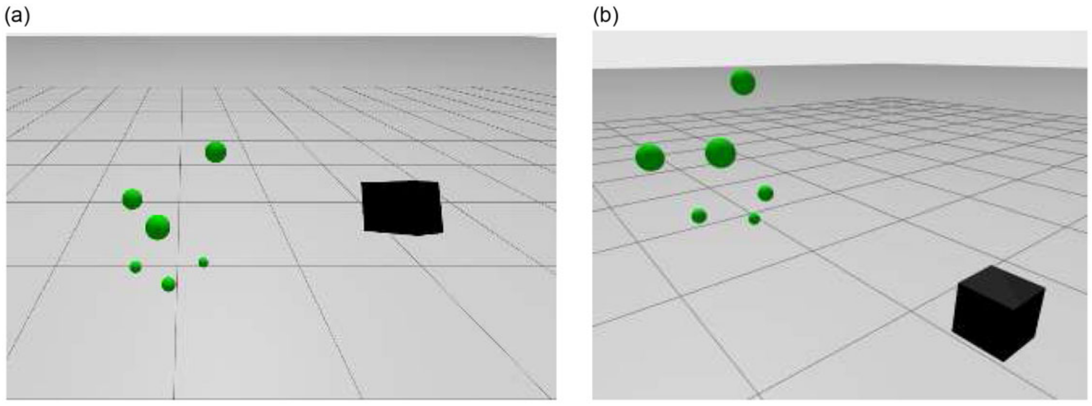

Figure 2. *Simulated environment to validate the histogram filter effectiveness. Green spheres are the objects being detected, representing the tomatoes, and the black box is the bounding box camera looking at the spheres.*

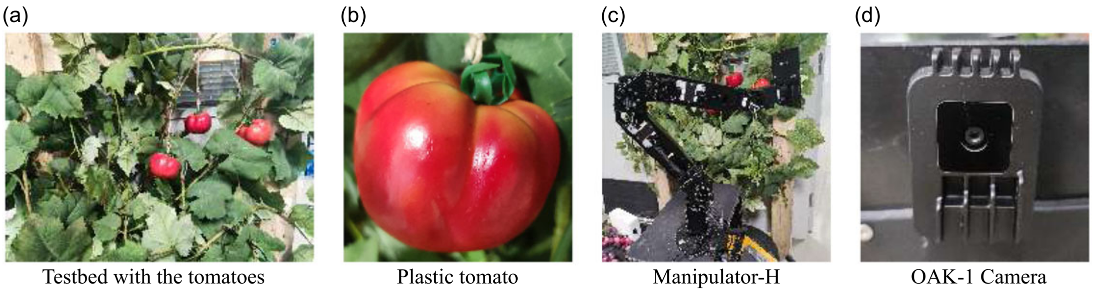

Figure 3. *Simulated testbed in the laboratory to essay the histogram filter algorithm.*

spheres have sizes of 5 cm and 10 cm. A bounding box camera was added to the scene to perceive the objects and support the position estimation algorithm. During the execution of the MonoVisual3DFilter (a histogram filter), the bounding box camera is moved to fixed viewpoints to enable and validate the estimator. The bounding box camera detects the objects through object detection algorithms using bounding boxes, detecting only their visible regions.

To essay the algorithm in the laboratory, near real-world greenhouse conditions, we designed a testbed with realistic leaves and plastic realistic tomatoes (Figure 3a and 3b). For perceiving the tomatoes, we assembled an OAK-1 camera[3] (Figure 3d), as a bounding box camera, to the 6 degrees of freedom (DoF) Robotis Manipulator-H (Figure 3c). The OAK-1 camera computes an object detection model algorithm trained to perceive tomatoes. We used the You Only Look Once (YOLO) v8 Tiny trained on the tomato dataset [45, 46] and some samples of the plastic tomato from multiple perspectives. However, any object detection or instance segmentation algorithm can be used to perceive the objects of interest in the scene, since they can effectively lead with the different environment perturbances, such as lighting variations or are robust to occlusion. The manipulator also moved to fixed viewpoints that ensured the tomatoes' visibility. The manipulator was assembled on the mobile platform AgRob v16 from INESC TEC (Figure 1), but 3D position of tomatoes was computed to the manipulator's base frame.

The OAK-1 is a fully integrated system for bounding box cameras. This camera was designed by the Luxonis Holding Corporation and has a single 12 MP RGB camera module. To allow the on-system

[3]See Luxonis Holding Corporation, "OAK-1," accessed on September 26th, 2023. [Online]. Available: https://docs.luxonis.com/projects/hardware/en/latest/pages/BW1093/

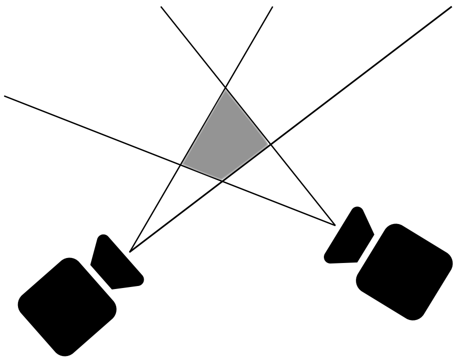

Figure 4. *Intersection between multiple viewpoints in 2D plane.*

object detection, the camera module is connected to the OAK-SoM.[4] The full camera connects to other devices through USB-C communication. The OAK-SoM is a system on module (SoM) designed to integrate into top-level and low-power systems and has the capability to process artificial neural networks (ANNs). This camera module was integrated into the AgRob v16 robot (Figure 1).

This robot is based on the Clearpath Husky[5] mobile platform and was designed to operate in harsh agricultural environments, such as the Douro's steep slope vineyards. At the front of the mobile platform, there is a controlling head, which is the unit with the computer and all the required devices to control the robot and establish communications. At the backside, the robot has the 6 DoF anthropomorphic manipulator Robotis Manipulator-H.[6]

2.2. Histogram filter

Histogram filters have been widely used in literature for self-localisation and navigation in mobile robots [36, 37]. However, for the current study, we intend to apply histogram filters to localise the 3D position of tomatoes concerning the manipulator's base frame, in a solution called MonoVisual3DFilter.

The histogram filter is a computationally intensive algorithm that can estimate the relative position of objects. The filter computes probabilities for multiple points in a discretised space. After, it intersects the chances of various views to estimate the localisation and the occupied area of the regions of interest (Figure 4).

Histogram filters can be set as an application of a discrete Bayes filter to the continuous state space. For this study, we applied the histogram filter as stated by Thrun [38].

The histogram filter decomposes the continuous state space in a finite number of regions. Eq. 1 describes a discretised state space. X_t is the random variable describing the state of the objects being detected at the time t. dom(X_t) denotes the state space, which is all the possible values that X_t might assume. The most straightforward discretisation of a continuous state space is through a multidimensional grid, where $x_{k,t}$ denotes each grid cell.

$$\operatorname{dom}(X_t) = x_{1,t} \cup x_{2,t} \cup \cdots \cup x_{K,t} \tag{1}$$

Only part of the state space must be discredited to limit computation efforts, mainly due to the manipulator's reachability. Therefore, as soon as the manipulator's camera detects an object of interest, in the first viewpoint, the space behind the camera is decomposed through a grid scheme. We considered the

[4]See Luxonis Holding Corporation, "OAK-SoM," accessed on September 26th, 2023. [Online]. Available: https://docs.luxonis.com/projects/hardware/en/latest/pages/BW1099/

[5]See Clearpath Robotics Inc., "Husky - Unmanned Ground Vehicle," accessed on September 26th, 2023. [Online]. Available: https://clearpathrobotics.com/husky-unmanned-ground-vehicle-robot/

[6]Robotis, "Robotis e-Manual – Manipulator-H," accessed on September 26th, 2023. [Online]. Available: https://emanual.robotis.com/docs/en/platform/manipulator_h/introduction/

probable space around the object as twice the manipulator's reaching limits. Twice the manipulator's reachability offers enough margin to identify and validate some fruits in the limits of the manipulator's reachability. The 3D decomposition is centred in the camera in $y\mathcal{O}z$, i.e. in $(0, 0)$ in the camera's frame and further distanced by the manipulator's reachability radius in $\mathcal{O}x$. Once decomposed, the discrete state space remains static and only $\mathrm{dom}(X_t)$ is updated.

The histogram filter demands moving the camera to multiple strategic viewpoints and updating the probabilities grid. So, at the space decomposition, an associated probabilities matrix is created with a probability to each cell initialised to one, i.e. $\mathrm{dom}(X_0) = [1 \ldots 1]$. This means that at the beginning, the object of interest is probable to be anywhere in the decomposed space.

For estimating the position of the objects, $\mathrm{dom}(X_t)$ is updated in each viewpoint. Each cell, $x_{i,t}$, from the decomposed state space is transformed from the manipulator's base frame to the image's frame. The probability of an object in a given cell, $x_{i,t}$, knowing the viewpoint, is given by (2). Finally, the updated probability of an object being in the cell $x_{i,t}$ is given by (3).

$$p(x_{i,t}|\mathrm{viewpoint}_k) = \frac{1}{N}\sum_{j}^{N} p(x_{i,t}|\mathrm{bbox}_j, \mathrm{viewpoint}_k) \tag{2}$$

$$p(x_{i,t}) = p(x_{i,t}|\mathrm{viewpoint}_k) \cdot p(x_{i,t-1}) \tag{3}$$

In eq. 2, to get $p(x_{i,t}|\mathrm{bbox}_j, \mathrm{viewpoint}_k)$, a kernel function was designed. Two kernel functions were essayed to localise the objects in the state space: the square and Gaussian functions. The square kernel, applied to each bounding box, states that if the transformed point is inside a bounding box of the image's frame, the probability is one; otherwise is zero (4). Using the square kernel function, we will have a binary mask stating that if a point is inside the bounding box, then we can have an object. Otherwise, we don't.

$$p(x_{i,t}|\mathrm{bbox}_j, \mathrm{viewpoint}_k) = \begin{cases} 1 & \text{if inside bounding box} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

As an alternative to the aggressive behaviour of the square function, we also essayed the bidimensional Gaussian function (5). This function delivers a smooth effect for the borders of the bounding box and some cells $x_{i,t}$ outside the bounding boxes. Therefore, a Gaussian function should better tolerate irregular objects and noise. In Eq. 5, (x_0, y_0) is the centre of bounding box j in the sensor's frame, and (x, y) is the position of each cell $x_{i,t}$ in the sensor's frame. The coordinates in the image's frame are projected by a projection model stated in section 2.3. The standard deviation values (σ_x, σ_y) correspond to half of the size of the bounding box j. All the values were obtained experimentally and had reasonable results. If we use the Gaussian kernel (5) to estimate the objects' position, Eq. 2 will correspond to a mixture of Gaussians, attending that the detection camera detects multiple objects. The mixture of Gaussians results in a function that smooths with increasing Gaussians in the mixture. To avoid this effect, a normalised version of the mixture of Gaussians is used to highlight the different detected objects (6). Besides, the updated state space $\dim(X_t)$ is also normalised at the end of each iteration of the histogram filter.

$$p(x_{i,t}|\mathrm{bbox}_j, \mathrm{viewpoint}_k) = \exp\left(-\frac{(x-x_0)^2}{2\sigma_x^2} - \frac{(y-y_0)^2}{2\sigma_y^2}\right) \tag{5}$$

$$p(x_{i,t}|\mathrm{viewpoint}_k) = \frac{p(x_{i,t}|\mathrm{viewpoint}_k)}{\max\left(p(x_{i,t}|\mathrm{viewpoint}_k)\right)} \tag{6}$$

The algorithm 1 summarises the procedures for updating the cell's weights during the histogram filtering.

Algorithm 1. Histogram filter – updating weights

```
Data: decomposition_grid, probabilities_matrix, bounding_boxes
Result: probabilities_matrix
for each viewpoint do
    for $x_{i,t}, p(x_{i,t})$ in decomposition_grid, probabilities_matrix do
        cell_camera ← transforms cell from the mainframe to camera's frame;
        cell_sensor ← cell_camera in the sensor's frame;
        (u,v) ← cell_sensor in the image's frame;
        $p(x_{i,t}|\text{viewpoint}_k) \leftarrow 0$ ;
        for bbox in bounding_boxes do
            $p(x_{i,t}|\text{viewpoint}_k) \leftarrow p(x_{i,t}|\text{viewpoint}_k) + \frac{1}{N} \times p(x_{i,t}|\text{bbox})$;
        end
    end
    $p(x_t|\text{viewpoint}_k) \leftarrow \text{normalise}(p(x_t|\text{viewpoint}_k))$;
    $p(x_t) \leftarrow p(x_t) \times p(x_t|\text{viewpoint}_k)$;
    $p(x_t) \leftarrow \text{normalise}(p(x_t))$;
end
```

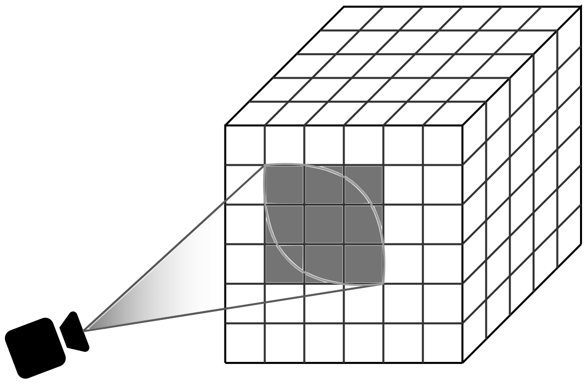

Figure 5. *Intersection of the camera around in the decomposed space.*

2.3. Camera projection model

While applying the histogram filter, we decomposed the state space in a finite state space grid. To effectively estimate the 3D position of the object, we moved the detection camera around the object and the decomposed state space to visualise the scene from multiple perspectives (Figure 5).

Detecting objects using the detection camera requires an effective projection model to estimate the object's position in the 3D space, transforming between the 3D space coordinates and the image's frame. For simplification, we applied the Pinhole model to transform the 3D space coordinates to the image's frame.

Acknowledging the points of the 3D space in the camera's frame, before we project them in the image's frame, we have to convert them to the sensor's frame. Both are placed in the same origin, but they have different orientations. The sensor uses the frame as illustrated in Figure 6. The illustrated rotation can be stated by Euler angles like Euler(YZX) $= (0^\circ, 90^\circ, -90^\circ)$ which reflects in the quaternion $q = (-0.5, 0.5, -0.5, 0.5)$.

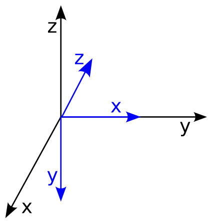

Figure 6. *Conversion between the camera's and sensor's frames (blue – sensor's frame; black – camera's frame).*

The transformation between the sensor's frame and the image's coordinates can be made by the intrinsics parameters matrix as stated in Eq. 7. This matrix depends on the image's width (w) and height (h), as well as on the camera's focal length (f). The focal length depends on the camera's horizontal field of view (HFOV) and the image's width and can be calculated by Eq. 8.

$$(u, v, 1) = \begin{bmatrix} f & 0 & \frac{w}{2} \\ 0 & f & \frac{h}{2} \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \frac{x}{z} \\ \frac{y}{z} \\ 1 \end{bmatrix} \tag{7}$$

$$f = \frac{0.5 \times w}{\tan\left(0.5 \times HFOV \times \frac{\pi}{180}\right)} \tag{8}$$

However, this model type is only valid behind ideal scenes, such as in the simulation. For the OAK-1 camera, an additional calibration step was required to estimate the intrinsic parameters. For calibration, we used the Kalibr software [47].

2.4. Objects positioning

At the end of the execution of the histogram filter for multiple viewpoints, the state space dom(X_t) should have different clusters of points. As we know the number of objects in the scene through the number of detected objects by the detection camera, we can use the k-means algorithm to aggregate the points and compute the centre of each cluster.

The k-means algorithm tries to cluster the different points of the discrete state space by minimising the geometric distance between points to the cluster's centre (9). In Eq. 9, we minimise the Euclidean distance between points to μ_j, the centre point of each cluster in C.

$$\sum_{i=0}^{n} \min_{\mu_j \in C} (||x_{i,t} - \mu_j||^2) \tag{9}$$

After clustering by the k-means, the state space should have as many point clouds as the number of objects detected by the detection camera. The computation of the centre of the clouds to get the position of the detected objects can be done in two ways:

1. computation of the geometric centre of the cloud; or
2. computation of the weighted centre of the cloud.

The geometric centre of each point cloud is the Euclidean centre μ_j minimised during the k-means algorithm for Eq. 9. Besides the geometric centre, the k-means also return the points, $x_{i,t}$, that belong

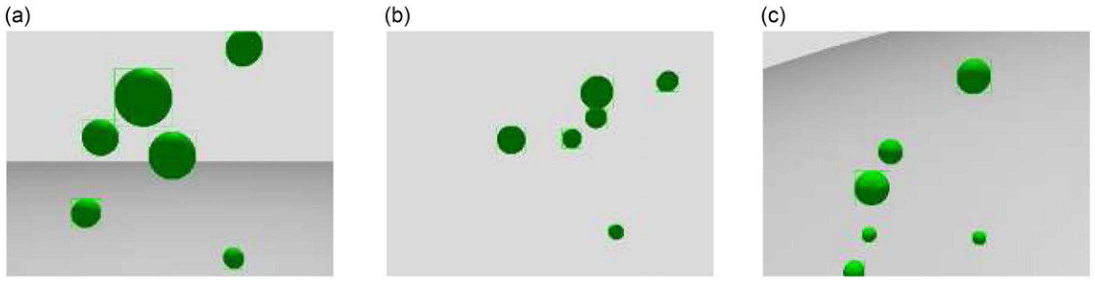

Figure 7. *View of the spheres by the bounding box camera at each fixed viewpoint. The green square boxes around the spheres are the bounding boxes of the detected spheres by the bounding box camera.*

to each cloud, S_j. Considering the state of each element of the state space $\mathrm{dom}(X_t)$ at the end of the histogram filter, each element $x_{i,t}$ should have a weight attributed, w_i. So the weighted centre is the weighted average (10) of the coordinates of $x_{i,t}$ that belong to the set S_j.

$$\mu_j = \left[\frac{\sum_i^N w_i \cdot x_{i,t_1}}{N} \quad \frac{\sum_i^N w_i \cdot x_{i,t_2}}{N} \quad \frac{\sum_i^N w_i \cdot x_{i,t_3}}{N} \right]^T \qquad \forall x_{i,t} \in S_j \tag{10}$$

2.5. Experiments

Three essays were performed in different environments to validate the effectiveness of the MonoVisual3DFilter.

Using the Gazebo simulator, we created a scene with multiple spheres to estimate their position in the scene (Figure 2). Once we used a bounding box camera without noise, this approach allowed validating the real performance of the filter without external artefacts or noise. Additionally, we performed an additional essay, introducing some Gaussian noise that randomly changes the position and size of the bounding box of the detected objects, as well as whether the object is successfully detected. Because we do not assemble any manipulator at the simulator, the bounding box camera has more freedom to state its pose. So, during the simulations, we set the camera's pose to ensure the spheres' visibility. Figure 7 illustrates the visible image of the camera at each pose. In the first pose, the camera looks straight towards the spheres (Figure 7a). After the camera moves down and left, looking upwards (Figure 7b), and finally, the camera moves up and right to the first pose, looking downwards (Figure 7c). This composition of the camera was kept for both experiments in the simulator.

In the third essay, a realistic testbed was deployed to experiment with the algorithm in near-real-world conditions at the laboratory (Figure 3). The testbed is composed of realistic artificial leaves and realistic plastic tomatoes. The tomatoes were hung in the testbed between the leaves. For baselining the tomatoes' position in the testbed, we relied on the manipulator's kinematics. For each tomato in the testbed, we moved the manipulator end-effector until the fruit and retrieved the end-effector's position. This will be the tomato position to the manipulator's base frame. Similarly to the essays in simulation, the bounding box camera at the simulator moved to three fixed poses that always assured the visibility of the tomatoes. A similar scheme to the one used before was set concerning the several limitations of the manipulator's manoeuvrability that made it difficult to set some poses to the camera. Unlike the simulation essays, several experiments were performed in the testbed. In the different experiments, we considered between one to three tomatoes being localised simultaneously, summing up to ten tomatoes and sixty measures. Figure 8 illustrates the tomatoes' visibility by the OAK camera at each selected pose, for the different experiments. The rows represent the different considered experiments and each image has the tomatoes being localised simultaneously.

To better assess the performance of the histogram filter to estimate the position of objects, we extracted some error metrics, namely, the mean absolute error (11), the mean square error (12), the

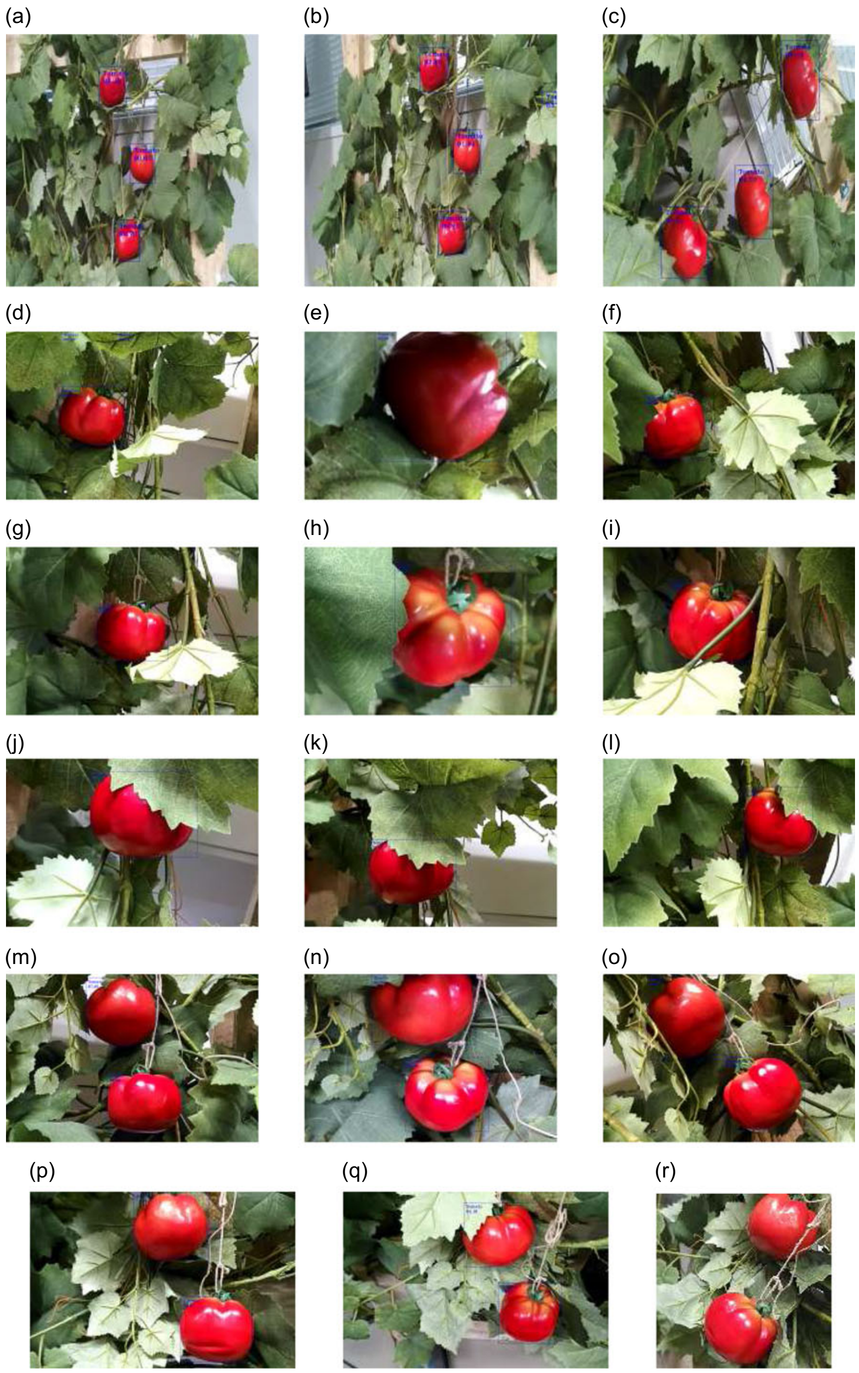

Figure 8. *View of the tomatoes in the testbed at each pose of the OAK-1 camera. The blue squares around the tomatoes are the detected tomatoes by the bounding box camera OAK-1 using a custom–trained YOLO v8 tiny detector. Inside each bounding box are the detected class (tomato) and the detection confidence. Each row is an experiment, in a total of six experiments, and each figure contains the number of tomatoes being detected.*

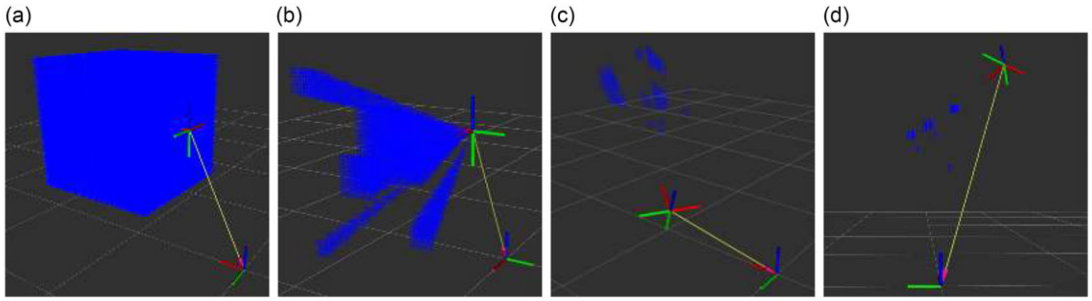

Figure 9. *Iteration of the histogram filter during simulation for detecting the six spheres, considering a square kernel. (a) Decomposition of the state space at the beginning of the algorithm; (b) detection at the end of the first viewpoint; (c) detection at the end of the second viewpoint; (d) detection at the end of the third viewpoint.*

root mean square error or standard deviation (13), and the mean absolute percentage error (14). In these equations, μ_j is the real centre of the object for the cluster S_j, and $\hat{\mu}_j$ is the estimated one using the previous methods, given the cluster j until the maximum of clusters M.

$$\text{MAE}\,(\mu_j, \hat{\mu}_j) = \frac{1}{N \cdot M} \sum_i^N \sum_j^M |\mu_{ij} - \hat{\mu}_{ij}| \qquad \forall j \in \mathbb{N} : \{1..M\} \tag{11}$$

$$\text{MSE}\,(\mu_j, \hat{\mu}_j) = \frac{1}{N \cdot M} \sum_i^N \sum_j^M (\mu_{ij} - \hat{\mu}_{ij})^2 \qquad \forall j \in \mathbb{N} : \{1..M\} \tag{12}$$

$$\text{RMSE}\,(\mu_j, \hat{\mu}_j) = \sqrt{\frac{1}{N \cdot M} \sum_i^N \sum_j^M (\mu_{ij} - \hat{\mu}_{ij})^2} \qquad \forall j \in \mathbb{N} : \{1..M\} \tag{13}$$

$$\text{MAPE}\,(\mu_j, \hat{\mu}_j) = \frac{1}{N \cdot M} \sum_i^N \sum_j^M \left| \frac{\mu_{ij} - \hat{\mu}_{ij}}{\mu_{ij}} \right| \times 100 \qquad \forall j \in \mathbb{N} : \{1..M\} \tag{14}$$

3. Results

As referred to in the previous section, we made three essays to validate the MonoVisual3DFilter's performance. Two essays happened in simulation, while the third was in a simulated testbed at the laboratory. To measure the performance, we recurred to different error metrics: (11), (12), (13) and (14). The manipulator and the bounding box camera were moved to fixed poses where all the fruits were always visible. All permutations between poses were considered for the essays, resulting in six estimations of each tomato for each experiment. The computed error results from the error of each estimated pose to ground truth pose of each corresponding tomato.

In the first essay, we used the simulation to estimate the 3D position of the green spheres (Figure 2) without any noise. During the execution of the histogram filter, the bounding box camera moved to different poses to intersect and be these positions. At each pose, the state space is operated to remove or smooth the existence of objects in each state according to the used kernel. Two kinds of kernels were considered, the square kernel (Figure 9) and the Gaussian kernel (Figure 10). Visually, both kernels performed similarly; however, the Gaussian kernel has two hyperparameters that deliver more freedom to set the kernel's size and estimate the objects' position, allowing the Gaussian filter to be more aggressive

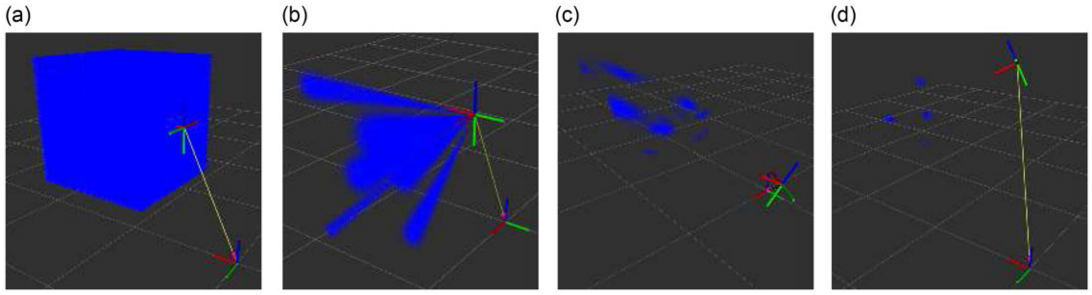

Figure 10. *Iteration of the histogram filter during simulation for detecting the six spheres, considering a Gaussian kernel,* $\mathcal{N}(0, 0.2)$*. (a) Decomposition of the state space at the beginning of the algorithm; (b) detection at the end of the first viewpoint; (c) detection at the end of the second viewpoint; (d) detection at the end of the third viewpoint.*

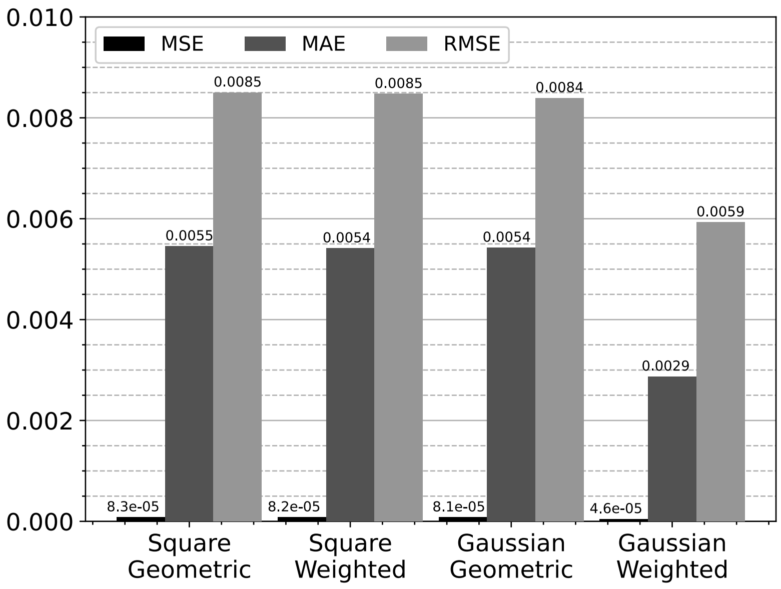

Figure 11. *Error in estimating the position of the spheres in simulation without noise.*

or smooth. We considered a two-dimensional Gaussian filter with $\mathcal{N}(0, \text{size}/2)$ for the current experiment, where size is the width or the height of the detected object. The plot of Figure 11 illustrates the error using a square or Gaussian kernel and the geometric centres from the k-means algorithm or the weighted centre resulting from applying the computed weights of the histogram filter. For this essay, using a Gaussian kernel with a weighted centre estimation was advantageous.

Adding some noise to the simulation by moving the bounding box's centre and size and deleting it randomly, we get the results of Figure 12. For moving the bounding box's centre and dimensions, we considered a Gaussian profile with $\mathcal{N}(0, 0.05)$. The bounding box will likely fail of 2‰. Once again, the Gaussian kernel with a weighted estimation of the spheres' centre performed better. Besides, in this experiment, the Gaussian kernel was more robust and accurate than the square kernel. The behaviour of the histogram filter in each pose is similar to the previous essay, but now, the algorithm has reported more noise. We set the Gaussian kernel such as $\mathcal{N}(0, size/3)$ to overcome this effect and have a more stable filter. This is the smallest value that assures that any fruit is lost and the point cloud is not too sparse, intersecting with the cloud of other fruits.

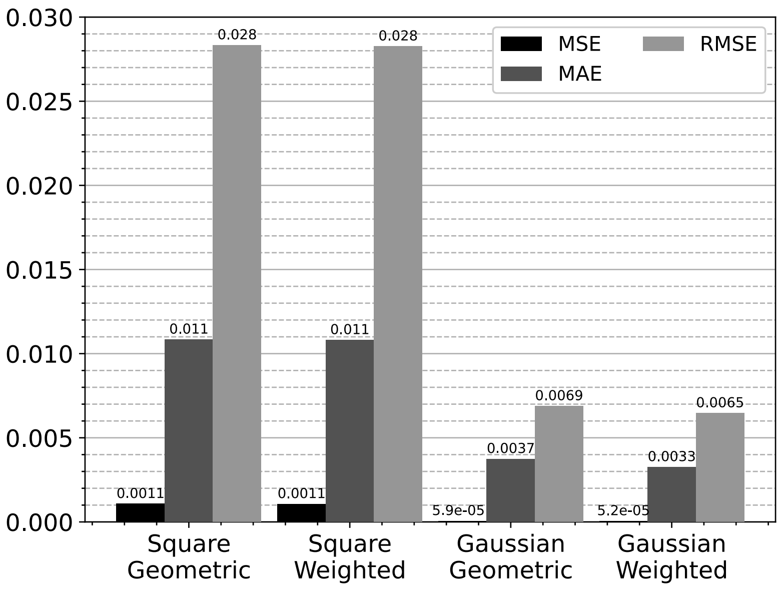

Figure 12. *Error in estimating the position of the spheres in simulation with noise.*

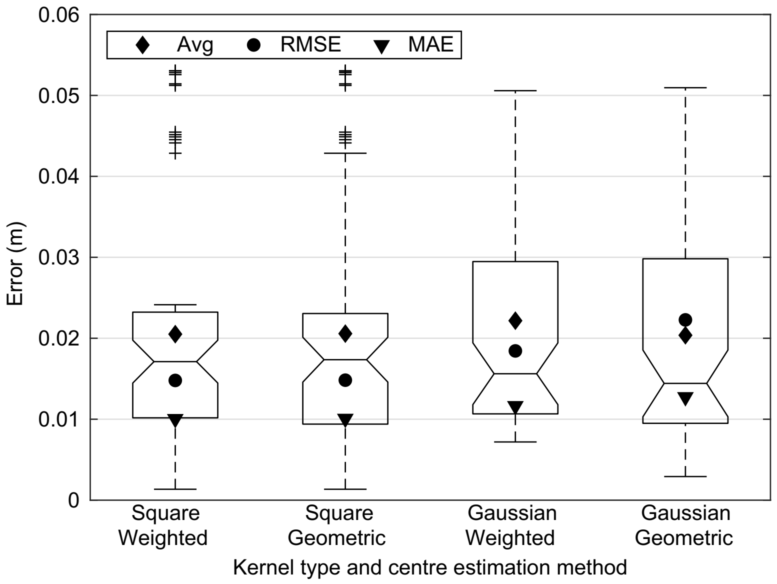

Figure 13. *Error in estimating the position of the tomatoes at the testbed.*

Given the overall success of the essays with a mean absolute error smaller than 1 cm, we essayed the algorithm in a testbed at the laboratory. Figure 13 illustrates the box plot error of the MonoVisual3DFilter in the testbed, using the Robotis Manipulator-H and the OAK-1 camera. A descriptive statement of the error is made in Table II. In this analysis, we also considered the average Euclidean error distance and the MAPE to better assess the feasibility of the MonoVisual3DFilter. In all the camera poses, all the tomatoes were always visible (Figure 8). For this essay, we considered six experiments that aimed to estimate the position between one to three tomatoes simultaneously, as stated in Figure 8, and summing up to sixty estimated measures for the position of the tomatoes. Against the expected, Gaussian kernels

Table II. *Error computations to the testbed experiments for the different kernels and centre estimation methods.*

	Square weighted	**Square geometric**	**Gaussian weighted**	**Gaussian geometric**
Euclidean Error (Avg)	0.0205 m	0.0206 m	0.0222 m	0.0204 m
MSE	0.2187×10^{-3}	0.2197×10^{-3}	0.3399×10^{-3}	0.4960×10^{-3}
RMSE	0.0148	0.0148	0.0184	0.0223
MAE	0.0100 m	0.0101 m	0.0116 m	0.0127 m
MAPE	63.52%	63.51%	57.35%	74.15%

performed worse than square kernels, and the results of weighted and geometric centres are identical, despite the weighted method tending to have a lower error for the same standard deviation.

4. Discussion

The overall use of MonoVisual3DFilter for estimating the 3D position of tomato fruits looks effective.

Under a simulated environment, the system always got a maximum error smaller than 10 mm. Increasing the resolution of the discretised state space could leverage better system accuracy, but increase the processing time and memory usage. As made in the second experiment, adding noise to the system makes the importance of using smooth kernels visible. The square kernel's aggressive binary behaviour rejects some state space positions and can never be recovered. On the other hand, the Gaussian filter has a smooth behaviour, and the positions are iteratively removed according to their distance from the filter's centre. So, smooth kernels can recover some state space points since they are never completely rejected. Besides, the Gaussian filter also was more failure-prone than the square kernel because the last one failed many times to detect the fruits, forcing us to repeat some experiments. Because the positions near the centre of the tomato have a bigger score, using weighted centre estimation procedures allows for a better fit of the estimated centre to the real centre of the tomato, reducing the effect and deviations of sparse clouds.

When moving the implemented algorithm to a real robot and camera in a testbed, we reported that, against expectations, the Gaussian kernels were not very effective and square kernels reached a higher accuracy. In this case, it is always irrelevant to consider geometric or weighted estimations of the centre. In the testbed, the system reported a mean absolute error around 20 mm, but the error can evolute to near 60 mm that can compromise the use of the algorithm for more demanding tasks. However, it is still important to better identify the source of the error, whether it comes from the MonoVisual3DFilter or the ground-truth's baseline, which was badly fitted to the tomato centre. Although the experiment was made with the viewpoints distanced by around 0.5 m of the fruits, the error should improve if a closer assessment of the tomato position is made. However, the reported error could not be critical whether using soft-grippers to aid harvesting tasks or using complementary algorithms. For tasks such as monitoring, this error could even be less relevant.

Additional experiments also allowed us to assess the importance of the selected viewpoints. Co-linear viewpoints do not allow for effective estimation of the position of the objects. However, normal viewpoints aim to better intersect the filter views and effectively estimate the position of the objects. This topic should be a concern under real world and testbed experiments because we frequently use manipulators with several positioning and orienting constraints.

Experiments as reported in Figs. 8j, 8k, 8l, and some others, could assess the feasibility of the MonoVisual3DFilter to partially occluded fruits. First, due to the limitations of detection algorithms to detect completely occluded objects, the MonoVisual3DFilter do not work for these cases. Concerning partial occlusion, we verified that the MonoVisual3DFilter could lead with the occlusion and effectively estimate the position of the tomato (Figure 14). Going further, we can even state that the

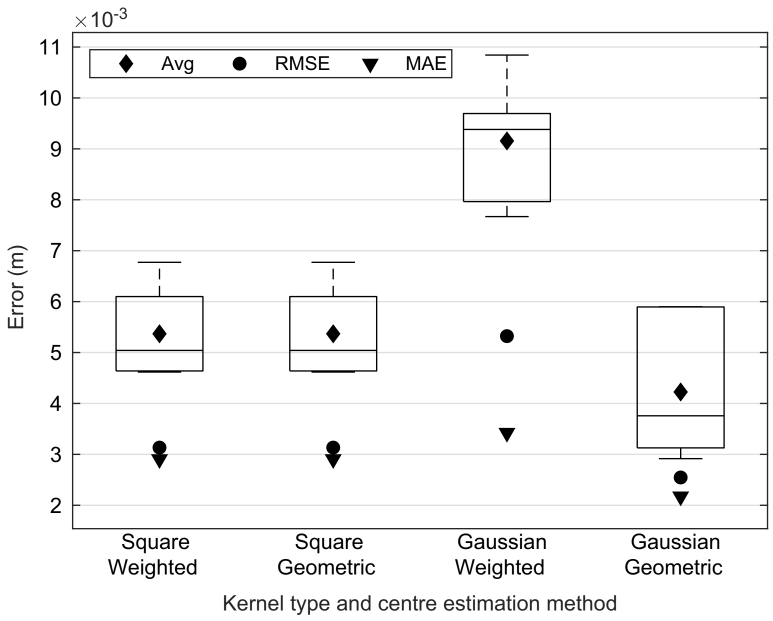

Figure 14. *Error for the case of partial occlusion of Figs. 8j, 8k, 8l,.*

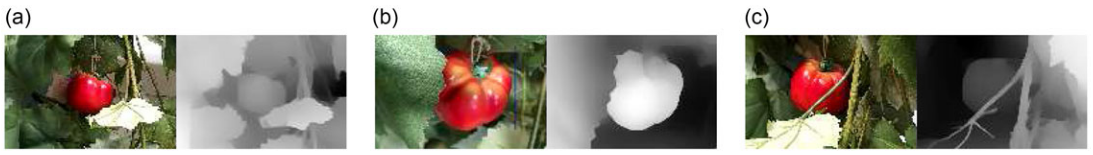

Figure 15. *RGB and depth images from the MiDaS v3.1 DPT SWIN2 Large 384 for estimating the tomatoes' distance to the camera's sensor.*

MonoVisual3DFilter is occlusion independent and estimates the position of occluded objects so good as good is the object detector.

Observing the literature (section 1), we can conclude that it tends to use deep learning solutions to infer the depth from monocular RGB cameras. A solution already available in the literature and that aims to effectively estimate the depth from images is the MiDaS network [18, 22]. For comparison with the MonoVisual3DFilter, we used the MiDaS v3.1 DPT SWIN2 Large 384, and applied it to the images of the experiment composed by the Figs. 8g, 8h, and 8i. Figure 15 reports the output of the MiDaS CNN for the proposed images. Once the network reports a relative pose, a calibration is required to estimate the real depth to the camera. According to [22], the absolute depth can be computed through a linear regression curve. So, a rough calibration was performed and reported the curve of Figure 16a. As can be visually concluded from Figure 15, the depth image reports a flat image, and it is difficult to understand the depth of the fruit. So, the network cannot effectively estimate the position of the fruit and reports errors up to 10 cm (Figure 16b). However, depth essays and calibration procedures should be made to purposefully conclude the noneffectiveness of MiDaS to estimate the depth of the image, i.e., MiDaS requires a complete RGB-D system to correctly calibrate the RGB sensor and network. Despite this error, deep learning-based solutions are much more computing demanding and less straightforward, difficulting to improve the results and track the origin of the errors. Besides, they also require much training and reliable data. On the other hand, MonoVisual3DFilter is data-independent, and its behaviour is more predictable.

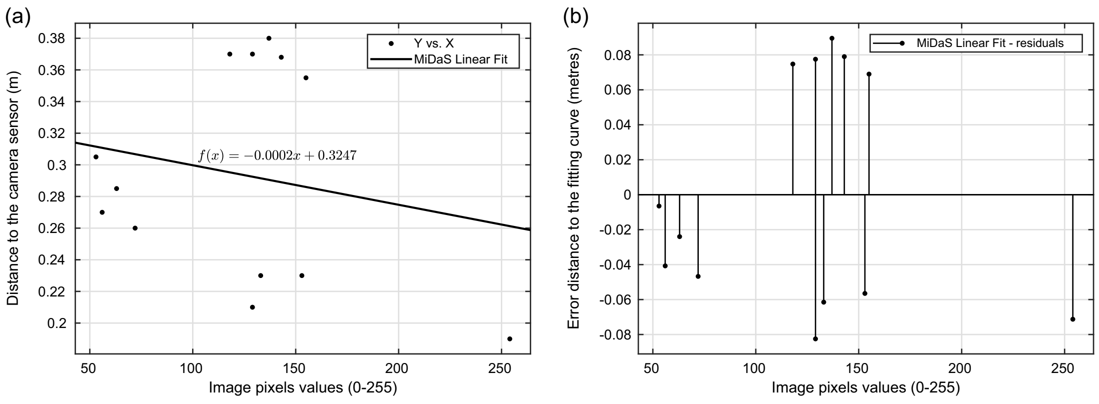

Figure 16. *Calibration curve to estimate the absolute depth in metres to the camera sensor for MiDaS CNN.*

Such as identified during the literature review in the introduction section are algorithms such as the SilhoNet, Nerf-Pose, or the GDR-Net. All of these algorithms have detection errors of about 2 cm. Near the MonoVisual3DFilter is the Imitrob model that can estimate the pose of the objects without their model, but reaches an estimation error of 6.5 cm on average. These solutions are competitive with the MonoVisual3DFilter, mainly if we consider complex datasets such as the one where they were essayed. Besides these algorithms can estimate the 6D pose of the target objects. Although these advantages are against the MonoVisual3DFilter, almost all of the solutions are model-dependent and computing-demanding. The MonoVisual3DFilter that we are here proposing is model-independent and only requires a mechanism capable of accurately detecting the target objects in a 2D scene. Besides, from these approaches, we can also conclude that our solution can be improved if we provide it with instance segmentation masks instead of rectangular bounding boxes that contain areas that do not belong to the objects.

The MonoVisual3DFilter can estimate objects' position in the 3D, mainly the circular ones. However, this algorithm is, currently, computationally demanding and requires many resources and time to compute a solution. In our case, the system took about one minute in each pose to compute the decomposed state-space using an Intel Core i7 with 8 GB of random-access memory (RAM). However, this is a highly parallelisable algorithm once all the positions are independent and can be computed simultaneously. So parallel implementation of the MonoVisual3DFilter at the computer processing unit (CPU), graphical processing unit (GPU), or mainly at the field programmable gate array (FPGA) can proportionally boost the speed of inference. Additionally, better optimisation of the implemented code can be done, mainly by using more efficient programming languages such as C or C++ instead of Python, which is less efficient and is interpreted during execution.

To better understand the advantages of parallelising the MonoVisual3DFilter, we studied the algorithm speedup through Amdahl's law [48] (15). In this equation, (15), $\sigma(n)$ is the inherently sequential computations, $\varphi(n)$ is the potentially parallel computations, and p is the number of processors (processes computing in parallel). It is important to mind that Amdahl's law ignores the communications between processes, so this law only computes the maximum speedup, $\Psi(n, p)$. Figure 17 illustrates the maximum reachable speedup by parallelising the histogram filter. The Gaussian filter reaches a higher speedup of about 17.5, but the square is a simpler kernel that operates faster, so the speedup is limited by the inherently sequential operations that cannot be optimised. During simulation, in a computer with an Intel Core i7 and 8 GB of RAM, the histogram filter took about 115 s/pose using the Gaussian Filter and 99s/pose using the square kernel, without parallelisation. The number of available cores also limits the

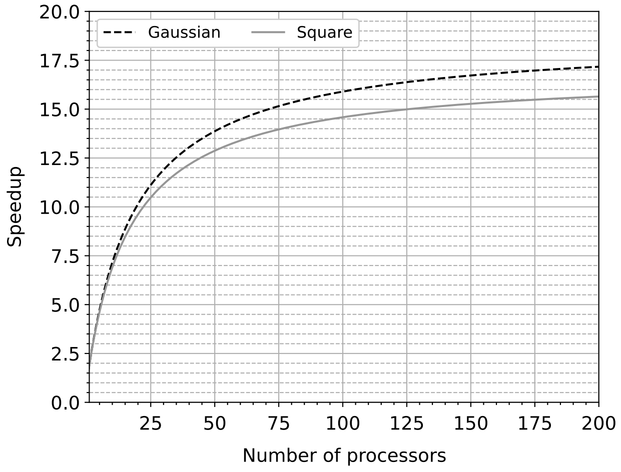

Figure 17. *Maximum speedup analysis for parallelisation according to Amdahl's law for the Gaussian and square kernels.*

speedup of the histogram filter. Once it is a very parallelisable algorithm, it benefits from using many cores, so the CPU is not so interesting such as GPU or FPGA because it is usually limited to 16 cores.

$$\Psi(n,p) \leqslant \frac{\sigma(n)+\varphi(n)}{\sigma(n)+\dfrac{\varphi(n)}{p}} \tag{15}$$

5. Conclusion

During this experiment, we designed a histogram filter-based algorithm, the MonoVisual3DFilter, to infer the 3D position of tomatoes in the tomato plant canopy using monocular cameras. The algorithm performed reasonably with an overall error of about 20 mm in laboratory-controlled conditions.

Despite the MonoVisual3DFilter being valid for estimating the tomatoes' position, some additional improvements are required. The next steps should focus on optimising the selection of the observation poses, making these poses adjustable and variable according to the fruit being analysed, and maximising observability through intelligent algorithms. The proposed algorithm can probably be improved if we feed it with instance segmentation masks instead of bounding boxes. Besides, improvements in execution time are still needed by optimising the developed code and implementing parallelisation strategies.

Other opportunities can also be explored while using the proposed algorithm. An example of that is using radar technology that allows one to perceive occluded objects behind the scene. Other similarly perceiving sensors can also be considered, acquiring the system with more robust sensors to perturbances in the scene.

Therefore, using histogram filters to estimate the position of objects, namely, the MonoVisual3DFilter, is viable and suitable for operating in the field under controlled scenarios. Further essays should be conducted on real scenarios and the implementation of active perception strategies.

Author contributions. Conceptualisation, S.C.M. and F.N.d.S.; funding acquisition, S.A.M and F.N.d.S.; investigation, S.C.M.; methodology, S.C.M.; software, S.C.M; formal analysis, S.C.M.; resources, F.N.d.S and A.P.M.; project administration, S.A.M, F.N.d.S., J.D. and A.P.M; supervision, F.N.d.S., J.D. and A.P.M.; validation, F.N.d.S., J.D. and A.P.M.; writing – original draft, S.C.M.; writing – review and editing, S.C.M., F.N.d.S., J.D. and A.P.M.

Financial support. This work is co-financed by Component 5 – Capitalization and Business Innovation, integrated in the Resilience Dimension of the Recovery and Resilience Plan within the scope of the Recovery and Resilience Mechanism (MRR) of the European Union (EU), framed in the Next Generation EU, for the period 2021–2026, within project PhenoBot-LA8.3, with reference PRR-C05-i03-I-000134-LA8.3.

Sandro Costa Magalhães is granted by the Portuguese Foundation for Science and Technology (FCT – Fundação para a Ciência e Tecnologia), through the European Social Fund (ESF) integrated into the Program NORTE2020, under scholarship agreement SFRH/BD/147117/2019 (DOI:10.54499/SFRH/BD/147117/2019).

Competing interests. The authors declare no conflicts of interest exist.

Ethical approval. Not applicable.

References

[1] J. Kitzes, M. Wackernagel, J. Loh, A. Peller, S. Goldfinger, D. Cheng and K. Tea, "Shrink and share: Humanity's present and future ecological footprint," *Phil Trans R Soc B: Biol Sci* **363**(1491), 467–475 (2007).

[2] FAOSTAT Statistical Database, Food and agriculture organization of the united states (2023). "Accessed on March 21st, 2023.

[3] M. Perry, "Science and innovation strategic policy plans for the, 2020s, (EU,AU,UK): Will they prepare us for the world in 2050?," *Appl Econ Finance* **2**(3), 76–84 (2015).

[4] M. Leshcheva and A. Ivolga, "Human Resources for Agricultural Organizations of Agro-Industrial Region, Areas for Improvement," **In:** *Sustainable Agriculture and Rural Development in Terms of the Republic of Serbia Strategic Goals Realization Within the Danube Region: Support Programs for the Improvement of Agricultural and Rural Development* (Institute of Agricultural Economics, 2017) pp. 386–400.

[5] R. L. V. Rica, G. G. Delan, E. V. Tan and I. A. Monte, "Status of agriculture, forestry, fisheries and natural resources human resource in Cebu and Bohol, central Philippines," *Trop Technol J* **19**, 11 (2015).

[6] euRobotics, Strategic agenda for robotics in Europe. resreport, euRobotics, (2014).

[7] A. McBratney, B. Whelan, T. Ancev and J. Bouma, "Future directions of precision agriculture," *Precis Agric* **6**, 7–23 (2005).

[8] A. Schmitz and C. B. Moss, "Mechanized agriculture : Machine adoption, farm size, and labor displacement," *AgBioForum* **18**(3), 278–296 (2015).

[9] G. Colucci, L. Tagliavini, A. Botta, L. Baglieri and G. Quaglia, "Decoupled motion planning of a mobile manipulator for precision agriculture," *Robotica* **41**(6), 1872–1887 (2023).

[10] M. S. Kumar and S. Mohan, "Selective fruit harvesting: Research, trends and developments towards fruit detection and localization — a review," *Proceed Insti Mech Eng Part C: J Mech Eng Sci* **237**(6), 1405–1444 (2022).

[11] S. A. Magalhães, A. P. Moreira, F. N. dos Santos and J. Dias, "Active perception fruit harvesting robots — a systematic review," *J Intell Robot Syst* **105**, 14 (2022).

[12] S. R. Nekoo, D. Feliu-Talegon, R. Tapia, A. C. Satue, J. R. M. de Dios and A. Ollero, "A 94.1 g scissors-type dual-arm cooperative manipulator for plant sampling by an ornithopter using a vision detection system," *Robotica* **41**(10), 3022–3039 (2023).

[13] J. Shen and N. Gans, "Robot-to-human feedback and automatic object grasping using an RGB-D camera–projector system," *Robotica* **36**(2), 241–260 (2017).

[14] I. Sa, C. Lehnert, A. English, C. McCool, F. Dayoub, B. Upcroft and T. Perez, "Peduncle detection of sweet pepper for autonomous crop harvesting – combined color and 3-D information," *IEEE Robot Autom Lett* **2**(2), 765–772 (2017).

[15] J. Jun, J. Kim, J. Seol, J. Kim and H. I. Son, "Towards an efficient tomato harvesting robot: 3D perception, manipulation, and end-effector," *IEEE Access* **9**, 17631–17640 (2021).

[16] J. Gené-Mola, J. Llorens, J. R. Rosell-Polo, E. Gregorio, J. Arnó, F. Solanelles, J. A. Martínez-Casasnovas and A. Escolà, "Assessing the performance of RGB-D sensors for 3D fruit crop canopy characterization under different operating and lighting conditions," *Sensors* **20**(24), 7072 (2020).

[17] O. Ringdahl, P. Kurtser and Y. Edan, "Performance of RGB-D Camera for Different Object Types in Greenhouse Conditions," **In:** *European Conference on Mobile Robots (ECMR) 2019*, (IEEE, 2019) pp. 1–6.

[18] R. Birkl, D. Wofk and M. Müller, MiDaS v3.1 – a model zoo for robust monocular relative depth estimation, (2023).

[19] X. Ma, S. Liu, Z. Xia, H. Zhang, X. Zeng and W. Ouyang, "Rethinking Pseudo-LiDAR Representation," **In:** *Computer Vision – ECCV*, of Lecture Notes in Computer Science, vol. **12358** (Online. Springer International Publishing, (2020) pp. 311–327.

[20] X. Ma, Z. Wang, H. Li, P. Zhang, W. Ouyang and X. Fan, "Accurate Monocular 3D Object Detection via Color-Embedded 3D Reconstruction for Autonomous Driving, Seoul, Korea (South)," **In:** *IEEE/CVF International Conference on Computer Vision (ICCV) 2019*, (IEEE, 2019) pp. 6850–6859.

[21] A. Mousavian, D. Anguelov, J. Flynn and J. Kosecka, "3D Bounding Box Estimation Using Deep Learning and Geometry," **In:** *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2017*, (IEEE, 2017) pp. 5632–5640.

[22] R. Ranftl, K. Lasinger, D. Hafner, K. Schindler and V. Koltun, "Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer," *IEEE Trans Pattern Anal* **44**(3), 1623–1637 (2022).

[23] Q. M. ul Haq, M. A. Haq, S.-J. Ruan, P.-J. Liang and D.-Q. Gao, "3D object detection based on proposal generation network utilizing monocular images," *IEEE Consum Electr Mag* **11**(5), 47–53 (2022).

[24] X. H. Van and N. Do, "An efficient regression method for 3D object localization in machine vision systems," *IAES Int J Robot Autom (IJRA)* **11**(2), 111–121 (2022).

[25] G. Billings and M. Johnson-Roberson, “SilhoNet: An RGB method for 6D object pose estimation,” *IEEE Robot Autom Lett* **4**(4), 3727–3734 (2019).
[26] F. Li, S. R. Vutukur, H. Yu, I. Shugurov, B. Busam, S. Yang and S. Ilic, “NeRF-Pose: A First-Reconstruct-Then-Regress Approach for Weakly-supervised 6D Object Pose Estimation,” **In:** *IEEE/CVF International Conference on Computer Vision Workshops (ICCVW) 2023*, (IEEE, 2023) pp. 2115–2125.
[27] G. Wang, F. Manhardt, F. Tombari and X. Ji, “GDR-Net: Geometry-Guided Direct Regression Network for Monocular 6D Object Pose Estimation,” **In:** *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) 2021*, (IEEE, 2021) pp. 16606–16616.
[28] A. Collet and S. S. Srinivasa, “Efficient Multi-View Object Recognition and Full Pose Estimation,” **In:** *2010 IEEE International Conference on Robotics and Automation*, (IEEE, 2010) pp. 2050–2055.
[29] T. Parisotto, S. Mukherjee and H. Kasaei, “MORE: Simultaneous multi-view 3D object recognition and pose estimation,” *Intel Serv Robot* **16**(4), 497–508 (2023).
[30] J. Chang, M. Kim, S. Kang, H. Han, S. Hong, K. Jang and S. Kang, “GhostPose: Multi-view Pose Estimation of Transparent Objects for Robot Hand Grasping,” **In:** *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2021*, (IEEE, 2021) pp. 5749–5755.
[31] J. Sedlar, K. Stepanova, R. Skoviera, J. K. Behrens, M. Tuna, G. Sejnova, J. Sivic and R. Babuska, “Imitrob: Imitation learning dataset for training and evaluating 6D object pose estimators,” *IEEE Robot Autom Lett* **8**(5), 2788–2795 (2023).
[32] G. Iyer, R. K. Ram, J. K. Murthy and K. M. Krishna, “CalibNet: Geometrically Supervised Extrinsic Calibration using 3D Spatial Transformer Networks,” **In:** *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2018*, (IEEE, 2018) pp. 1110–1117.
[33] H. Küçük, G. Parker and E. T. Baumgartner, “Robot positioning of flexible-link manipulator using vision,” *Robotica* **22**(3), 301–307 (2004).
[34] J. Qu, F. Zhang, Y. Fu and S. Guo, “Multi-cameras visual servoing for dual-arm coordinated manipulation,” *Robotica* **35**(11), 2218–2237 (2017).
[35] L. Xu, J. Li, Y. Hao, P. Zhang, G. Ciuti, P. Dario and Q. Huang, “Depth estimation for local colon structure in monocular capsule endoscopy based on brightness and camera motion,” *Robotica* **39**(2), 334–345 (2020).
[36] N. Boyko and Y. Hladun, “Histogram Filter for Robot Localization,” **In:** *IEEE 16th International Conference on Computer Sciences and Information Technologies (CSIT) 2021*, (IEEE, 2021) pp. 38–43.
[37] A. Moscowsky, “Subdefinite Computations for Reducing the Search Space in Mobile Robot Localization Task,” **In:** *Artificial Intelligence 2021*, Lecture Notes in Computer Science, vol. **12948** (Springer International Publishing, 2021) pp.180–196.
[38] S. Thrun, W. Burgard and D. Fox, “Probabilistic Robotics,” **In:** *Intelligent Robotics and Autonomous Agents Series*, (MIT Press, (2005).
[39] J. Sarmento, F. N. D. Santos, A. S. Aguiar, H. Sobreira, C. V. Regueiro and A. Valente, “FollowMe - A Pedestrian Following Algorithm for Agricultural Logistic Robots,” **In:** *IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC) 2022*, (IEEE, 2022) pp. 179–185.
[40] S. Engin and V. Isler, “Active Localization of Multiple Targets from Noisy Relative Measurements,” **In:** *Algorithmic Foundations of Robotics XIV. Springer Proceedings in Advanced Robotics*, (Springer International Publishing, 2021) pp. 398–413.
[41] Z. C. Márton, S. Türker, C. Rink, M. Brucker, S. Kriegel, T. Bodenmüller and S. Riedel, “Improving object orientation estimates by considering multiple viewpoints,” *Auton Robot* **42**(2), 423–442 (2017).
[42] C. W. Bac, E. J. van Henten, J. Hemming and Y. Edan, “Harvesting robots for high value crops: State of the art review and challenges ahead,” *J Field Robot* **31**(6), 888–911 (2014).
[43] L. van Herck, P. Kurtser, L. Wittemans and Y. Edan, “Crop design for improved robotic harvesting: A case study of sweet pepper harvesting,” *Biosyst Eng* **192**, 294–308 (2020).
[44] B. Arad, J. Balendonck, R. Barth, O. Ben-Shahar, Y. Edan, T. Hellström, J. Hemming, P. Kurtser, O. Ringdahl, T. Tielen and B. van Tuijl, “Development of a sweet pepper harvesting robot,” *J Field Robot* **37**(6), 1027–1039 (2020).
[45] S. A. Magalhães, L. Castro, G. Moreira, F. N. dos Santos, M. Cunha, J. Dias and A. P. Moreira, “Evaluating the single-shot multiBox detector and YOLO deep learning models for the detection of tomatoes in a greenhouse,” *Sensors* **21**(10), 3569 (2021).
[46] S. A. Magalhães, Dataset of tomato inside greenhouses for object detection in pascal VOC, (2020). Online, Last accessed on April 27th, 2023.
[47] J. Maye, P. Furgale and R. Siegwart, “Self-Supervised Calibration for Robotic Systems,” **In:** *IEEE Intelligent Vehicles Symposium (IV) 2013*, (IEEE, 2013) pp. 473–480.
[48] G. M. Amdahl, “Validity of the Single Processor Approach to Achieving Large Scale Computing Capabilities,” **In:** *Proceedings of the April 18-20, 1967, Spring Joint Computer Conference on - AFIPS ’67*, (ACM Press, 1967) pp. 483–485.

Cite this article: S. A. C. Magalhães, F. N. dos Santos, A. P. Moreira and J. M. M. Dias, “MonoVisual3DFilter: 3D tomatoes’ localisation with monocular cameras using histogram filters”, Robotica. https://doi.org/10.1017/S0263574724000936