Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 59-18 (2025) 253–258

Vision based robotic portion control for granular food

Alessandro Colombo * **Riccardo Busetto** ** **Leonardo Innocenti** * **Matteo Corno** * **Andrea Zanchettin** * **Sergio Matteo Savaresi** *

* *Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Milano, 20133, Italy*

** *IDSIA Dalle Molle Institute for Artificial Intelligence USI-SUPSI, Lugano-Viganello, Switzerland.*

Abstract: The food industry is increasingly adopting robotic automation for food preparation, driven by the need for precision and efficiency. Vision systems play a pivotal role in enabling robots to execute tasks effectively in these dynamic and complex environments. This work proposes two vision-based methods for real-time estimation of the mass of food collected by a robotic manipulator, allowing precise portion control for granular foods, such as rice. Specifically, we investigate the usage of 2D and 3D vision algorithms. The former estimates the quantity of food based on the area of the collection tool that is occluded by food, while the latter obtains the estimate from the volume of food collected. Experimental results demonstrate that the 2D algorithm excels at higher speed, while the 3D algorithm offers superior accuracy at lower speed, highlighting the strengths and limitations of each approach for different operating conditions.

Keywords: Food processing, Mechatronics, Parameter and state estimation, Robots manipulators, Robot Navigation Programming and Vision.

1. INTRODUCTION

The demand for efficient and streamlined operations in the food industry has sparked a growing interest in the automation of various processes. Activities such as mixing, processing, packaging, and sorting have already seen a great deal of investment in robotic solutions; operations that require a high level of human input are also likely to become fully automated in the future, despite their functional and computational complexity. The use of robots improves operational efficiency, monitoring, food safety, and consistency in order preparation.

One task that would benefit from robotic automation is preparing canteen trays. Most facilities rely on human workers placed alongside a conveyor belt, repetitively filling plates with food according to menu choices. This task requires both high flexibility, as the types of foods can be varied, and precise portioning, as dietary needs may be strict, especially in healthcare facilities.

In this paper, we develop vision algorithms that allow a robotic station to collect a specific amount of food. This approach uses visual feedback as a proxy for force feedback. By estimating food weight during collection, rather than afterward with a weight sensor, we can have faster execution time by removing the need for additional scooping actions to either increase or reduce the already collected quantity to meet requirements. Furthermore, the use of force sensors placed on the robot wrist (e.g. load cells) may require long calibration procedures and it introduces many disturbances in the measurements, due to inertial components during robot motion and potential elastic behaviors of the food-collecting tools. As such, a vision sensor was preferred.

Computer vision plays a key role in the deployment of robots for challenging tasks such as the proposed one. Algorithms that use data from cameras and other image sensors and provide visual insight allow robots to perform complex operations in dynamic environments and monitor the food's state in real-time Super Annotate Site (2023) Guthrie (2023).

In the literature, relevant features estimation from visual data has been applied in many sectors, from medicine (e.g. Al-Naji et al. (2022), Noel et al. (2024), Remus et al. (2024)) to agriculture and farming (e.g. Ballesteros et al. (2020), Franco et al. (2022), Wang et al. (2023)).

In the food industry, food identification and quality control has recently gained traction Zhao et al. (2025). For instance Jiang et al. (2020) identifies different types of food from single images, and estimates the associated nutrition values of the meal, assuming standard portions; Azarmdel et al. (2020) estimates the state of ripeness of white and red mulberries, by analyzing color, shape, and texture features with an Artificial Neural Network (ANN); Xu et al. (2022) uses 2D and 3D information to locate cherry tomatoes and their stem for automatic picking, by employing Region-based Convolutional Neural Networks (R-CNN).

These types of inferences, however, are not suitable for our task. In fact, they are performed "a posteriori", on still images and in static environments. Furthermore, they use complex models, which may prove to be too slow for real-time applications.

10.1016/j.ifacol.2025.10.229

Much akin to our task, Schenck et al. (2017) uses a robotic arm with a spoon to collect raw beans inside a container and dump them into another. The focus is however only on the volumetric arrangement of the beans: convolutional neural networks are trained to predict which action (from a parametrized set) can best achieve a desired 3D shape. The collected mass is estimated to improve some processing steps, but the quality of the estimation is not investigated. Furthermore, this methodology requires a large dataset to be collected, with over 10000 samples, as the predictor needs to learn the implicit dynamical model of the food.

As in our work we need to monitor in real-time the state of the food during collection, a visual servoing approach Hutchinson et al. (1996) is more appropriate: visual data is used directly in the low-level (robot motion) control loop. For instance, in Li et al. (2023), flour sacks are collected using a robotic arm: image moments and Gaussian Process Regression are used to align the robot and the sack. This approach however requires a specific shape to be tracked and assumes that this shape is static. In Park et al. (2024), cucumbers and their stems are identified by an automatic robot picker, and visual servoing is then used to guide the robot toward the stem. Here possible oscillations of the object of interest are taken into account and compensated, however, it still relies on the cucumber specific shape for segmentation. Cherubini et al. (2020) uses a full model-free approach to shape kinetic sand using a simple rectangular end-effector. Starting from human-demonstrated data, a robot learns how to appropriately rearrange sand using a control law built directly on the visual data (image-based visual servoing). Overall, visual servoing with a model-free approach is the most suitable approach for our task.

The main contributions of our work are: (*i*) we present two visual-based methods to solve the problem of real-time precise food quantity portioning of granular food with a robotic manipulator, (*ii*) we experimentally validate the algorithms, analyzing their performance in terms of variability of the collected food with respect of the robot speed during the collection process, comparing the approaches.

The paper is structured as follows. In Section 2, we formally present the problem statement. In Section 3, the methodology to solve the problem is illustrated, presenting both the 2D and 3D vision-based solutions. In Section 4, the experimental setup is described, while in Section 5 the obtained results are presented and a comparison of the two methods is performed. The paper ends with a summary and some concluding remarks in Section 6.

2. PROBLEM STATEMENT

The objective is the development of an algorithm that allows a robotic manipulator to collect a desired quantity $m_d^{(i)}$ (mass, [g]) of granular food from a given container during an experiment $i = 1, \dots, N$ of duration $T^{(i)} > 0$.

In detail, at each sampling instant $t = 0, \dots, T^{(i)}$, the robot has access to images $I_t^{(i)}$ —either 2D or 3D— from a wrist-mounted camera, used to provide a real-time estimate of the food quantity $m_t^{(i)}$ collected by a rigid-mounted tool (a skimmer, in this case).

Formally, the goal is to obtain a model f that minimizes

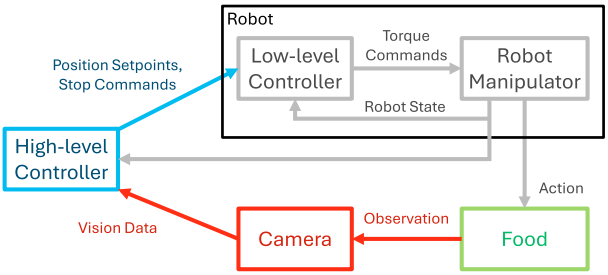


Fig. 1. Control scheme

$$\|m^{(i)} - \hat{m}_{\bar{t}}^{(i)}\|_2^2, \tag{1}$$

i.e., the difference between the measured food quantity $m^{(i)}$ at the end of the i-th experiment, and the estimated food quantity $\hat{m}_{\bar{t}}^{(i)} = f(I_{\bar{t}^{(i)}})$, where $\bar{t}^{(i)}$ is the instant where the robot receives the command to stop the collecting movement.

The obtained model f allows for real-time control of the quantity picked by the robot, by a simple heuristic. When condition

$$C = \hat{m}_t^{(i)} \geq m_d^{(i)}, \tag{2}$$

is satisfied, in other words when the estimated food quantity reaches the desired quantity for the experiment, a stop command is issued.

3. METHODOLOGY

The key elements of the system are a robotic arm equipped with a tool for food collection, a camera placed on the robot's wrist, and a container filled with a granular type of food (e.g. rice). The robot is commanded by a high-level controller via position references. The camera evaluates the food states and passes vision data to the high-level controller, which uses that data to compute the position references, closing the loop (see Fig. 1).

The overall procedure takes the following form:

(1) The robot moves along an open loop trajectory that inserts the tool in the container, putting progressively more food on the tool.
(2) The vision algorithm, operating in parallel, estimates the weight of the food that is currently on the tool.
(3) When a certain threshold is reached the high-level controller sends a stop command to the robot, pulls out the tool, and moves to pour the food inside a bowl

Given that the robot is only controlled via position references, this is not properly "visual servoing", however, while the robot moves along the open loop trajectory, the robot control is equivalent to having a control action

$$u(y) = \begin{cases} \bar{V} & \text{if } y \leq y_{th} \\ 0 & \text{else} \end{cases} \tag{3}$$

Where y is the value provided by the vision algorithm, y_{th} is the associated desired threshold, and $\bar{V}$ is the constant 6D speed of the robot during the open loop trajectory.

The open loop trajectory starting point p_0 is chosen randomly across the food container to avoid trying to collect always in the same location. Furthermore, to avoid collisions, the open loop trajectory ends before the tool

hits the container floor. The choice of an optimal point of ingress is outside of this paper's scope. The location of the plate into which the food is poured p_f is known and fixed.

Two algorithms were developed: the first uses 2D images and is detailed in Section 3.1, and the second uses 3D pointcloud data, and is described in Section 3.2.

3.1 2D algorithm

The first algorithm to estimate the collected food weight uses 2D data from the camera. The main idea behind this algorithm is that the area of the tool occluded by food is representative of the weight of the collected food. Given the granularity of the food, we can assume that a certain base area can support up to a certain amount of food before collapsing unto itself; furthermore, if we insert the tool in a location with enough food we can assume that enough volume is being collected. These two assumptions provide us with an upper and lower bound to the weight associated with a certain area seen by the camera.

The algorithm is composed of the following steps:

(1) Segment the image to focus only on the area where the food-handling tool head is.
(2) Compute the area (in number of pixels) of the tool that is covered in food.
(3) Compare the obtained area with the threshold one.

Given that the relative position between the tool and the eye-in-hand camera stays constant throughout the entire robot motion, step 1 can be achieved using a static mask, which highlights the skimmer head (fig 2). This mask is used to perform a bitwise AND operation with the raw image I_t to obtain $I'_t = \mathcal{F}_{\text{static}}(I_t)$. The mask being slightly larger then the tool head allows for small lateral displacements of the tool with respect to the camera.

I'_t is then converted into HSV space, and a filter is applied, highlighting the pixels whose hue (H), saturation (S), and brightness (V) are inside certain thresholds:

$$P_i = \begin{cases} 1 \text{ if} & \begin{aligned} H_{lb} \le H_i \le H_{ub} \wedge \\ S_{lb} \le S_i \le S_{ub} \wedge \\ V_{lb} \le V_i \le V_{ub} \end{aligned} \\ 0 \text{ otherwise} \end{cases} \tag{4}$$

Where P_i is a generic pixel, H_i, S_i, and V_i are its hue, saturation, and brightness respectively. H_{lb}, H_{ub}, S_{lb}, S_{ub}, V_{lb}, V_{ub} are empirical thresholds tuned to isolate the food-collecting tool color. Applying this filter we obtain the binary image $I''_t = \mathcal{F}_{\text{HSV}}(I'_t)$.

On I''_t, contours are found by using the algorithm proposed by Suzuki and Abe (1985). The largest contour is used to compute the current unobstructed area of the tool (in pixels). That contour is highlited in Fig. 3, both before (a) and during (b) collection. The number of pixels obtained is then subtracted from the one associated with the full tool area (known a priori and fixed) to obtain the food area $A_t = \mathcal{F}_{\text{Count}}(I''_t)$.

Assuming that the inquired relation $\hat{m}_t^{(i)} = f_{2D}(A_t^{(i)})$, where A_t is the computed food area, is strictly increasing

Algorithm 1 2D Algorithm

Require: m_d, p_0
  $A_d \leftarrow f_{2D}^{-1}(m_d)$
  $A_t \leftarrow 0$
  Move to p_0
  Set robot speed to $\bar{V}$
  while $A_t \le A_d$ **do**
    Capture image I_t
    $I'_t = \mathcal{F}_{\text{StaticMask}}(I_t)$
    $I''_t = \mathcal{F}_{\text{HSV}}(I'_t)$
    $A_t = \mathcal{F}_{\text{Count}}(I''_t)$
  Set robot speed to 0
  Move to p_f

(a) Initial image I_t (b) Image with the static mask I'_t

Fig. 2. Tool head segmentation

and as such invertible ($A_t^{(i)} = f_{2D}^{-1}(\hat{m}_t^{(i)})$), we modify Equation 2, and use as heuristic the condition

$$C_{2D} = A_t^{(i)} \ge A_d^{(i)}, \tag{5}$$

with $A_d^{(i)}$ as the area associated to the desired weight $m_d^{(i)}$.

The relationship between the food area and its weight can then be found empirically by setting different threshold values and weighing the collected food using a scale or a similar tool. By observing the variability in food weight given a certain threshold, we can validate the strength of the invertibility assumption.

The complete process of food collection using this algorithm can be seen in Algorithm 1.

(a) Clear skimmer head (b) Partially covered skimmer head

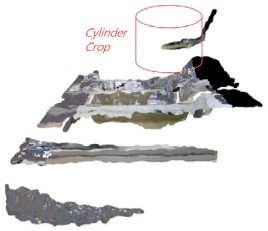
Fig. 3. Examples with the visible area of the skimmer highlighted

3.2 3D algorithm

The second algorithm we present uses pointcloud data from the camera. The main idea is to estimate the volume of the collected food, and from that the weight.

The algorithm is composed of the following steps:

(1) Segment the pointcloud by removing points outside a cylindrical area of interest around the skimmer head.

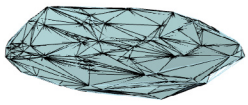


(a) Skimmer face isolation

(b) Initial pointcloud pcd'_{init}

(c) Captured pointcloud pcd'_t

(d) Combined pointcloud pcd''_t



(e) Mesh of the empty tool

(f) Mesh during food collection $mesh_t$

Fig. 4. 3D algorithm steps

(2) Decimate the pointcloud to reduce computational load.
(3) Combine the pointcloud with one captured when the tool was empty.
(4) Create a mesh from the resulting pointcloud and compute its volume.
(5) Compare the obtained volume with the threshold one.

When the skimmer is first grasped, a pointcloud (pcd_{init}) is captured and saved for the rest of the motion. The cylindrical area of interest is computed on this pointcloud (Fig. 4a). Once more, given that the relative position of the tool and the camera is constant throughout the movement, a static cylindrical mask is applied:

$$p_i \in P_{tool} \text{ if } \sqrt{(x_i - x_c)^2 + (y_i - y_c)^2} \leq r \\ \wedge \ |z_i - z_c| \leq h/2 \tag{6}$$

where $p_i = (x_i, y_i, z_i)$ is a generic point, (x_c, y_c, z_c) are the coordinates of the cylinder center, r its radius, and h is its height.

The resulting pointcloud pcd'_{init} contains only the points belonging to the empty tool. After a decimation process, in which the number of points is reduced to improve computational times, the pointcloud is saved for comparison.

While food is being collected, the same mask and decimation process is applied to the real-time pointcloud pcd_t; the obtained pointcloud $pcd'_t = \mathcal{F}_{\text{Crop}}(pcd_t)$ is then compared to pcd'_{init}. By overlaying the 2 pointclouds we can obtain a depiction of the collected food surface both on the side facing the camera and on the opposite one (see Fig. 4b, 4c, and 4d), $pcd''_t = \mathcal{F}_{\text{Sum}}(pcd'_t, pcd_{init})$.

Algorithm 2 3D Algorithm

Require: m_d, p_0
  $V_d \leftarrow f_{3D}^{-1}(m_d)$
  $V_t \leftarrow 0$
  Move to p_0
  Capture pointcloud pcd_{init}
  $pcd'_{init} = \mathcal{F}_{\text{Crop}}(pcd_{init})$
  Set robot speed to $\bar{V}$
  while $V_t \leq V_d$ **do**
    Capture pointcloud pcd_t
    $pcd'_t = \mathcal{F}_{\text{Crop}}(pcd_t)$
    $pcd''_t = \mathcal{F}_{\text{Sum}}(pcd'_t, pcd'_{init})$
    $mesh_t = \mathcal{F}_{\text{Delaunay}}(pcd''_t)$
    $V_t = \mathcal{F}_{\text{Volume}}(mesh_t)$
  Set robot speed to 0
  Move to p_f

(a) Full setup

(b) Camera and magnetic end-effector

Fig. 5. Experimental Setup

On the combined pointcloud, a mesh is obtained from through Delaunay Triangulation: $mesh_t = \mathcal{F}_{\text{Delaunay}}(pcd''_t)$, which allows us to compute the food volume $V_t = \mathcal{F}_{\text{Volume}}(mesh_t)$ (Fig. 4e and 4f).

As in the previous section, we assume that the inquired relation $\hat{m}_t^{(i)} = f_{3D}(V_t^{(i)})$, where V_t is the computed food volume, is strictly increasing and as such invertible. We then modify Equation 2, and use as a heuristic

$$C_{3D} = V_t^{(i)} \geq V_d^{(i)}, \tag{7}$$

Where $V_d^{(i)}$ represents the volume associated to the desired weight $m_d^{(i)}$.

Again, the relationship between the food volume and its weight can then be found empirically by setting different threshold values and weighing the collected food using a scale or a similar tool. By observing the variability in food weight given a certain threshold, we can validate the strength of the invertibility assumption.

4. EXPERIMENTAL SETUP

The experimental set-up, visible in Fig. 5, consists of a Doosan robotics collaborative robot arm (model M1013), a depth camera (Intel RealSense Depth Camera D435) mounted on the robot wrist, and a gastronorm (standard food container) filled with rice. The robot is equipped with a magnetic end-effector (Schunk emh-rp 045) used to grasp

Table 1. Experimental results, 2D algorithm

V [mm/s]	A_d [px]	4000	6000	8000	10000	12000
10	$m_{avg}^{(N_e)}$ [g]	35.5	56.8	72.1	110.4	140.4
	$\sigma_{\%}^{(N_e)}$	12.36	10.05	7.22	5.63	5.38
15	$m_{avg}^{(N_e)}$ [g]	33.6	57.4	82.7	112.8	135.0
	$\sigma_{\%(N_e)}$	10.66	9.83	6.99	7.57	5.99
30	$m_{avg}^{(N_e)}$ [g]	40.3	68.1	89.2	104.5	144.8
	$\sigma_{\%}^{(N_e)}$	14.26	11.17	11.04	5.28	5.36

food-collecting tools. A skimmer (visible in Fig. 3) was used to collect the rice; to improve the performance of the HSV filter described in Section 3.1, this tool was covered with food-graded black paint.

The camera placed on the robot outputs a 2D image with a resolution of 640×480p at a frame rate of 60FPS and a 3D pointcloud at 30FPS.

5. EXPERIMENTAL RESULTS

Both algorithms were tested using different pixel or volume values a threshold (4000, 6000, 8000, 10000, and 12000 pixels and 30, 60, 90, 120, and 150 cm^3 respectively), with the robot moving at 10, 15, and 30mm/s during the food collection motion. After each iteration, the collected food is placed inside a bowl and weighed. For each threshold i_t and each robot velocity i_v, we perform $N_e = 10$ experiments, and we then compute the average collected weight $m_{avg,i_t,i_v}^{(N_e)}$ and relative standard deviation $\sigma_{\%,i_t,i_v}^{(N_e)}$. Finally, the overall variability associated with a certain robot speed is computed as

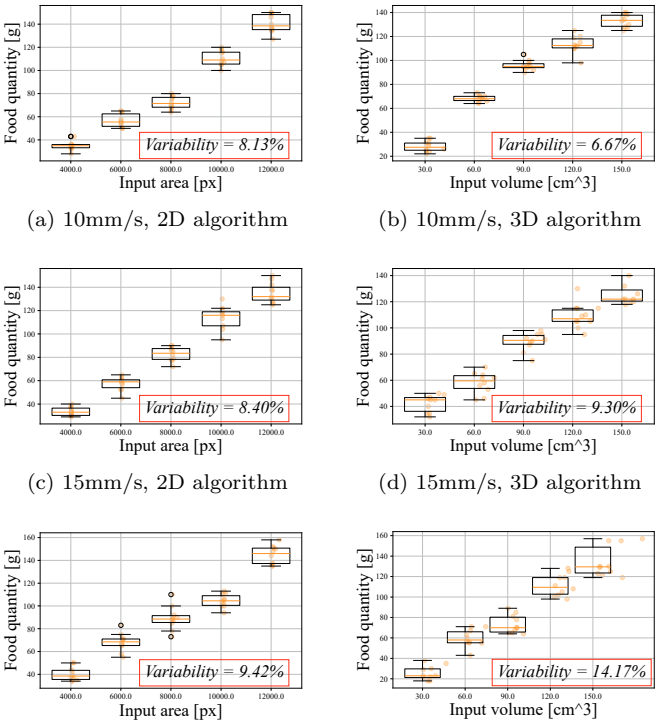
$$\bar{\sigma}_{\%,i_v}^{(N_e,N_t)} = \frac{\sum_{i_t} \sigma_{\%,i_t,i_v}^{(N_e)}}{N_t} \tag{8}$$

where N_t is the number of threshold values.

In Fig. 6, and in Table 1 and 2, the results of this experimental procedure are shown. For both algorithms, there is a clear increasing trend of the collected food weight with respect to the threshold size: our assumption of the intrinsic models f_{2D} and f_{3D} invertibility is then validated. Furthermore, it can be seen how the average variability increases with the tool speed, from 8.13% to 9.42% for the 2D algorithm and from 6.67% to 14.17% for the 3D algorithm. At higher speed, the actual volume of food collected for the 3D algorithm (indicated by the orange points in Fig. 6) tends to be quite higher than the threshold one. This can be attributed to the lower refresh rate of the 3D algorithm: in fact, while the 2D algorithm has an average refresh rate of 60Hz, the 3D one has an average refresh rate of 10Hz.

Comparing speed and variability, we can see in Fig. 7 how over about 12.6mm/s it is advisable to use the 2D algorithm for more accurate food weighting, while at lower speeds the 3D one is more performing.

Furthermore, it is worth pointing out how most of the aforementioned experiments were performed on an almost full gastronorm with a mostly level food surface. If the robot were to collect food in the proximity of "peaks" of food, the 2D algorithm would struggle, as it does not take



(a) 10mm/s, 2D algorithm (b) 10mm/s, 3D algorithm

(c) 15mm/s, 2D algorithm (d) 15mm/s, 3D algorithm

(e) 30mm/s, 2D algorithm (f) 30mm/s, 3D algorithm

Fig. 6. Experimental Results

Table 2. Experimental results, 3D algorithm

$\bar{V}$ [mm/s]	V_d [cm^3]	30	60	90	120	150
10	$m_{avg}^{(N_e)}$ [g]	28.3	68.2	95.8	113.4	133.2
	$\sigma_{\%}^{(N_e)}$	15.41	3.76	4.22	6.09	3.93
15	$m_{avg}^{(N_e)}$ [g]	42.3	58.0	89.3	107.4	125.2
	$\sigma_{\%}^{(N_e)}$	14.73	13.31	7.34	5.95	5.21
30	$m_{avg}^{(N_e)}$ [g]	25.7	59.7	73.5	111.5	134.4
	$\sigma_{\%}^{(N_e)}$	25.7	13.71	11.69	9.09	10.69

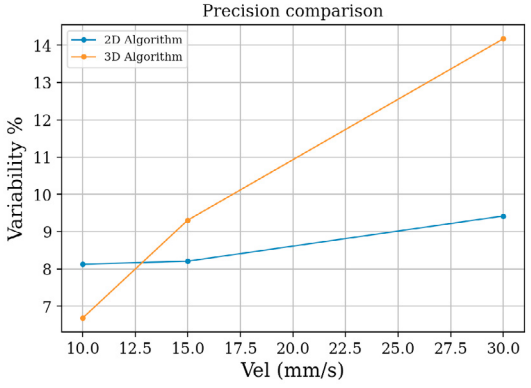


Fig. 7. Velocity versus output variability

into account the z component of the collected food. As such a tentative "mountain" experiment was performed, in which the robot collects food in the proximity of a peak in food distribution, about 7cm tall (see Fig. 8), at 15mm/s.

(a) Side view (b) Front view

Fig. 8. "Mountain" experiment

The 2D algorithm received as input 6000px, while the 3D one received 60cm^3, as both should output around 60g of rice, given the previous experiments. While the latter algorithm produced 75g, the former collected 150g of rice.

Finally, we can summarize the advantages and disadvantages of both algorithms:

The 2D algorithm has a high precision and is able to maintain a low variability even at higher speeds. Furthermore, the lower computational load allows the algorithm to issue the stop commands promptly so that the actual area of the collected food is close to the threshold one.

On the other hand, it relies on the color of the food to be different from the tool one, and in any case, color segmentation may not always be accurate. In addition to this, it does not take into account the z-axis distribution of food, so that in limit cases like the "mountain" experiment, it may fail to collect the desired amount of food.

The 3D algorithm can achieve higher performances at lower speeds, and it does not have to consider the food color. Furthermore, it can accurately detect the food z-axis distribution. However, this algorithm is computationally heavy, and becomes unreliable at higher speeds, causing delays in the stop commands.

6. CONCLUSIONS

In this paper, we presented two vision algorithms to estimate food weight from visual data, when collecting it autonomously with a robot.

The first algorithm uses bi-dimensional images, while the second uses pointcloud data. Both algorithms can accurately estimate the food weight, with a variability of 8.13% and 6.67% respectively for lower speed of operation.

The individual strengths and weaknesses of both algorithms were highlighted, showing how the first algorithm is more suitable for higher speeds, while the second is more accurate at lower speeds.

REFERENCES

Al-Naji, A.A., Fakhri, A., Mahmood, M., and Chahl, J. (2022). Contactless blood pressure estimation system using a computer vision system. *Inventions*, 7, 84.

Azarmdel, H., Jahanbakhshi, A., Mohtasebi, S.S., and Muñoz, A.R. (2020). Evaluation of image processing technique as an expert system in mulberry fruit grading based on ripeness level using artificial neural networks (anns) and support vector machine (svm). *Postharvest Biology and Technology*, 166, 111201.

Ballesteros, R., Intrigliolo, D., Ortega, J., Ramírez Cuesta, J., Buesa, I., and Moreno, M. (2020). Vineyard yield estimation by combining remote sensing, computer vision and artificial neural network techniques. *Precision Agriculture*, 21, 1–21.

Cherubini, A., Ortenzi, V., Cosgun, A., Lee, R., and Corke, P. (2020). Model-free vision-based shaping of deformable plastic materials. *The International Journal of Robotics Research*, 39(14), 1739–1759.

Franco, V., Hott, M., Andrade, R., and Goliatt, L. (2022). Hybrid machine learning methods combined with computer vision approaches to estimate biophysical parameters of pastures. *Evolutionary Intelligence*, 16.

Guthrie, R. (2023). Robotic vision improving food traceability: Sharper focus, safer food.

Hutchinson, S., Hager, G., and Corke, P. (1996). A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation*, 12(5), 651–670.

Jiang, L., Qiu, B., Liu, X., Huang, C., and Lin, K. (2020). Deepfood: Food image analysis and dietary assessment via deep model. *IEEE Access*, 8, 47477–47489.

Li, H., Zuo, Z., Chi, R., Du, Y., and Mao, E. (2023). Image moments-based visual servoing control of bagged agricultural materials handling robot. *International Journal of Agricultural and Biological Engineering*, 16.

Noel, L., Fat, S.C., Causey, J.L., Dong, W., Stubblefield, J., Szymanski, K., Chang, J.H., Wang, P.Z., Moore, J.H., Ray, E., and Huang, X. (2024). Sex classification of 3d skull images using deep neural networks. *Scientific Reports*, 14(1), 13707.

Park, Y., Kim, C., and Son, H.I. (2024). Fast and stable pedicel detection for robust visual servoing to harvest shaking fruits. *Computers and Electronics in Agriculture*, 220, 108863.

Remus, R., Sure, C., Selkmann, S., Uttich, E., and Bender, B. (2024). Soft tissue material properties based on human abdominal in vivo macro-indenter measurements. *Frontiers in Bioengineering and Biotechnology*, 12, 1384062.

Schenck, C., Tompson, J., Levine, S., and Fox, D. (2017). Learning robotic manipulation of granular media. In *Conference on Robot Learning*, 239–248. PMLR.

Super Annotate Site (2023). Computer vision applications in robotics.

Suzuki, S. and Abe, K. (1985). Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, 30(1), 32–46.

Wang, Y., Mücher, S., Wang, W., Guo, L., and Kooistra, L. (2023). A review of three-dimensional computer vision used in precision livestock farming for cattle growth management. *Computers and Electronics in Agriculture*, 206, 107687.

Xu, P., Fang, N., Liu, N., Lin, F., Yang, S., and Ning, J. (2022). Visual recognition of cherry tomatoes in plant factory based on improved deep instance segmentation. *Computers and Electronics in Agriculture*, 197, 106991.

Zhao, Z., Wang, R., Liu, M., Bai, L., and Sun, Y. (2025). Application of machine vision in food computing: A review. *Food Chemistry*, 463, 141238.