Автономная некоммерческая образовательная организация высшего образования «Сколковский институт науки и технологий»

На правах рукописи

Осокин Илья Витальевич

Робот и система компьютерного зрения для поиска заболеваний и оценки объема урожая при мониторинге теплиц

Специальность 2.3.1. Системный анализ, управление и обработка информации, статистика

Диссертация на соискание ученой степени
кандидата технических наук

Научный руководитель:
доктор физико-математических наук, доцент
Осиненко Павел Валерьевич

Москва — 2025

Autonomous Non-Profit Organization for Higher Education "Skolkovo Institute of Science and Technology"

As a manuscript

Osokin Ilya Vital'evich

A robot and a computer vision system for disease detection and crop yield estimation in greenhouse monitoring

Speciality **2.3.1.** System Analysis and Information Processing

Dissertation for the Academic Degree of
Doctor of Philosophy in Engineering

Supervisor:
Doctor of Science in Physics and Mathematics, Associate Professor
Osinenko Pavel Valer'evich

Moscow — 2025

Аннотация

Современному сельскому хозяйству свойственны промышленные масштабы и высокая эффективность. Автономное обнаружение заболеваний и оценка урожая критически важны для автоматизации, предотвращения потерь продукции и дальнейшего повышения эффективности производства. В процессе автоматизации этих процессов встают некоторые технические задачи, которым и посвящена эта диссертация. В этой работе представлено комплексное решение, включающее в себя робота на всенаправленном колесном шасси, систему камер для мониторинга растений в условиях современного тепличного хозяйства, а также набор алгоритмов, решающих задачи детектирования мучнистой росы и оценки объема эллипсоидальных фруктов и овощей. Была предложена модификация алгоритма построения эллипсоида по зашумленному облаку точек, а именно процесс оценки гипотез был сведен к матричному умножению с целью ускорения. Все предложенные алгоритмы были широко протестированы на реальных данных и были оценены по метрикам, релевантным для сельского хозяйства. Результаты показывают, что предложенные методы могут быть внедрены в практику в современных тепличных хозяйствах.

Abstract

Modern agriculture is characterized by scale and efficiency. Autonomous disease detection as well as autonomous yield estimation are crucial to the automatization, loss prevention and further efficiency improvement. As some of these labour-intensive tasks are being automated, certain technical problems should be solved. In this work a complex solution is presented, including a robot with omnidirectional shassis and a camera setup, and a set of algorithms that solve the problems of powdery mildew identification and yield volume estimation for ellipsoidal fruit and vegetables. An improvement was proposed to the algorithm of ellipsoidal model fitting to the point cloud data. Specifically, the process of checking the hypothesis was sped up. All the proposed algorithms were extensively tested on real-world data and evaluated in terms of the metrics relevant to the industry. The results suggest the applicability of the proposed approaches in the real-world scenarious.

Content

Introduction

The volume and structure of the work. The dissertation consists of an abstract, introduction, 6 chapter, conclusions, lists of abbreviations, figures, tables and 0 appendix. The full thesis' volume is 106 pages, including 41 figure and 15 table. The list of references consists of 66 sources.

0.1 Relevance

Modern farming is characterized by both scale and efficiency. Both precise yield estimation and disease detection are its integral parts. Tangerines and tomatoes are one of the most important fruit in the agricultural produce worldwide [1].

More specifically, knowing the mass of the tomatoes that are ready to be picked is crucial for proper logistics planning [2]. And volume is one of the most important physical attributes of the fruit yield in agricultural production. If performed manually, it is time-consuming, expensive and prone to error due to the loss of focus under the monotonous conditions of such labour. In order to automate this process, computer vision-based systems are often used. Because of the scale of modern farms and greenhouses, covering them entirely with stationary cameras is infeasible. Thus, the majority of the solutions are based on autonomous robots, i.e. flying or wheeled.

This work addresses the problem of estimating position, size and orientation of tomatoes and tangerines under the conditions close to the real greenhouse. This problem is addressed with two different approaches, one relying on classical algorythmical surface fitting, and another one being an end-to-end Neural Network-based approach.

Secondly, disease detection is crucial in loss prevention and taking timely measures to optimize the production. There are multiple diseases that manifest in a visible way, with powdery mildew being one of them. In this work the the problem of powdery mildew identification is addressed. In order to solve it, a robot was developed, as well as a multi-camera setup for data collection.

0.2 Dissertation Goals

The goal of this dissertation is to investigate the performance of volume estimation algorithms that process 3D point cloud data in application to ellipsoidal objects in agricultural setting. The second goal it to evaluate the performance of the disease identification algorithms in application to the tomato plants.

In order to achieve these goals, the following problems are solved.

Thie first is to develop an implementation of the ellipsoid fitting algorithm with point cloud input. The second is to evaluate its performance in multiple settings, including various data collection conditions and different objects. The third is to solve the problem of volume estimation with an end-to-end approach, i.e. Neural Network. The fourth is to perform data collection in real greenhouse conditions, as well as markup, data coherency evaluation and model training for the problem of disease detection. The fifth is to develop a robot capable of moving in the greenhouse on two types of surface, while carrying a computer and a camera setup, with sufficiently long battery lifetime.

0.3 Research methodology

The methodology incorporates theoretical ideas from the field of pattern recognition. The design of the main algorithms is based on random sample consensus-based approaches, as well as the properties of the second-order curves and surfaces. At the same time, a lot of the specific design solutions both in terms of the software and the hardware were made in accordance with the real-world demands from the engineering standpoint. It provided the means to consider the prospective usage in terms of vibration reliability, autonomous operation time, and locomotion abilities of the robot. The following is a list of main software tools and frameworks used to conduct numerical experiments:

- OpenCV
- Open3D
- ROS2
- Matlab

– PyTorch

0.4 Validity of the obtained results

The obtained results were verified by an array of numerical experiments on both real-world and synthetic data, see the respective sections in the relevant chapters.

0.5 Approbation

The results were presented on 4 conferences: MIPT conference, Zavalishinsky readings, and two papers at EDM conference.

0.6 Personal contribution of the author

The author actively contributed to several research activities.

First, the autor formulated the research hypothesis, proposed and validated experimental setups, as well as participated in experiments, including ones in the real greenhouse environment.

Second, the author participated in the algorithms development and numerical experiments.

Third, the author took part in the designing, manufacturing, and overall working on the omnidirectional robot, including milling, assembly, painting, components purchase, cable management, soldering.

Fourth, the author participated in writing all 6 papers, that are the foundation of the thesis presented.

Fifth, the author made significant contribution to the legal part of the project, in particular grant-related activities, reports, patenting, and TRL.

Sixth, the author contributed to the organization of the project, including personnel search, trips, and procurement.

0.7 Propositions for Defense

1. The problem of model evaluation for the second order surface fitting was reduced to a matrix multiplication. The experimental results suggest that the performance of the proposed method on a single modern laptop allows for the real-time greenhouse monitoring.
2. The influence of the noise on the quality of ellipsoid fitting was numerically evaluated. The practical outcomes include results for the second-order curves as a simpler object.
3. A unique dataset of tomato leaves infected by powdery mildew was collected, marked and patented. The defining feature is that the images were taken with a user-grade camera from a moving robot in the real greenhouse. The publicly available datasets are all filmed in lab conditions with artificial lighting, steady objects and expensive cameras.
4. A robot was developed and its crucial feature (the wheel) was patented. The robot was specifically designed to serve as a hardware support for the whole project, and it encompasses the camera setup, the computing capabilities onboard, and a prolongued lifetime on a single charge.

0.8 Engineering Novelty and Practical Outcomes

In terms of the practical results, the following main results should be highlighted.

First, a problem of tomato and tangerine volume estimation was solved on the real point cloud data. Second, a problem of powdery mildew identification on tomato leaves was solved on real RGB data from user-grade cameras. Third, PointNet was adapted to the regression problem and used to estimate the volume of tangerines in the end-to-end fashion

The robot that was developed during the research was certified to be TRL-5 (Technology Readiness Level) in terms of its ability to move on both flat surface and rails. The robot is also capable of using its kinematic model to follow any curve on a two-dimensional surface. Practically it means that the platform is ready to carry scientific or industrial load, if controlled in the environment properly.

Three patents were issued to Skoltech during the research.

The first is for a dataset of tomato leaves infected by powdery mildew. The unique feature is that the images were filmed with a mid-price RGB camera from a moving robot in the real greenhouse, which is in great contrast with the publicly available datasets, filmed in the lab, with artificial lighting, steady scene and expensive camera setups.

The second is for a computer program capable of classifying images into infected by powdery mildew and not infected.

The third is for a wheel, corresponding of a Mecanum wheel, rail wheel, mounted coaxially, integrated motor, and a suspension system.

Finally, a company AIDA Robotics is in process of being established.

Chapter 1. Background

1.1 Modern Agriculture

1.1.1 Modern Greenhouses

The target environment for the monitoring robot can be described as follows. Modern greenhouses are highly structured, consisting of repeating sections, where the plants are grown. A considerable portion of tomato farming facilities are equipped with rails, that are regularly layed perpendicular to the central alley. The monitoring by a human worker is normally performed either by walking or by riding a specialized vehicle with an elevation mechanism, allowing them to inspect the whole height of the plant. An image of the line with tomato plants is presented in the Figure 3.14.

1.1.2 Imaging Hardware in Monitoring Automation

The physical examination of the fruit and measuring the volume with the underwater submersion can be used, as well as the measurement of the physical dimensions of the fruit and using a geometrical model to obtain the volume. However, this approach is excessively labour-intensive and requires manual manipulation of the fruit. Thus, a number of contactless approaches were proposed, mainly relying on monocular or stereoscopic vision.

In the recent years computer vision-based monitoring methods gain traction. They mainly rely on RGB sensors, and in certain cases depth cameras. While RGB-only sensors are much cheaper, depth information about the scene allows for more precise volume estimation. One of the key factors in the vision-based volume estimation is the noise that is an inherent property of the sensor data in the real greenhouse facility. In order to accommodate for these noises, robust volume estimation techniques should be applied.

Let us briefly cover the main advantages and disadvantages of these approaches. On the one hand, monocular setups are often cheaper than stereoscopic. On the other hand, the performance of the monocular vision is limited by the inability of such methods to perceive true depth in the scene. Stereoscopic approaches capture the true volume better, but their usage is complicated by the setups being more expensive, bulky and difficult to work with.

Disease detection, object detection for calculation and a variety of other problems can be perfectly solved with monocular vision. But precise volume estimation requires at least some awareness about the true depth in the scene [3]. Acquiring quality depth map could be a challenge, given the varying lighting conditions in certain challenging environments like greenhouses.

Stereoscopic cameras with active infrared lighting, such as Intel RealSense D435i [4], are a standard solution here. They not only emit monochromatic light in a pattern so that the depth could be recovered, but also perform all the calculation onboard, providing the user with a virtual camera with the depth.

Going further, the most notable research belongs to the group of methods that rely on the point clouds. This data modality proves itself to be convenient for precise position, volume and rotation estimation. Point cloud-based data is sparse, thus compact, in contrast to voxels.

1.1.3 Volume Estimation and Ellipsoidal Model

Tomatoes, tangerines and other fruit can be approximated as ellipsoids. An ellipsoid is described by 9 parameters: three coordinates, three semi-axis, and orientation. In this work episode model was used for the tangerines and tomatoes and the results show that this approximation allows for precise enough volume estimation.

The ellipsoid identification problem widely arises as a part of modern computer vision in a variety of applications, including agricultural [5] [6], medical [7], robotics [8], and environmental reconstruction [9].

The real-world setting is often associated with noise in the sensor output, which naturally leads to the demand for robust algorithms for obtaining ellipsoids.

While classical model-based approaches are already well-developed, in the recent years neural network-based end-to-end methods gain traction. One of the

papers comprising this work is devoted to the case study of applying a PointNet++-like neural network to the problem of tangerine volume estimation with the point cloud data. The output quality was assessed via comparing the prediction of the method with the real volume of the fruit. The average volume estimation error slightly exceeds 10%.

Regarding general volume estimation, in the monocular case the most common method relies on the dividing of the object into slices with evaluating their volume based on the geometric features, and adding the obtained volumes. Such is the work [10], where the volume of Thai apple ber is evaluated.

In the case of irregularly shaped objects, similar approach can be applied [11]. The volume of the sweet potato is evaluated with adding all the volumes of the slices of the same height.

Another family of approaches can be used with regularly shaped objects only, such as tomatoes, tangerines and alike. A model is chosen that corresponds well to the object, and its parameters are fitted to the observed data [12].

While multiview setup is used, both approaches remain feasible, but fitting model parameters is prevalent.

In the series of papers [13],[14],[15] authors propose the combination of the aforementioned approaches by combining slicing with circular slice parameters estimation from two cameras.

However, slicing approach has a number of drawbacks, one of the most significant of them being the loss of precision while working with sparse point clouds. An examplary case of the ladder is [16]. This work is devoted to the estimation of the volume of pile-like objects, like stored sand or grain. The proposed method relies on the sampling of a voxel grid in accordance with the input point cloud.

Finally, there is a group of shape estimation techniques that rely on Random Sample Consensus (RANSAC) [17]. The presented work belongs to this class of methods.

A number of the objects can be approximated as ellipsoids, including pineapples [18], human head [19] and jujubes [20]. There is also a branch of research that relies on spherical models [21], but its applicability is limited for elongated objects.

Moreover, fitting a surface directly to the point cloud data instead of using proxy metrics like projections [22], [22] is more straightforward and systematic way of approaching the problem.

However, to our knowledge there are no works that combine NN-based detection with robust volume estimation.

The proposed method was compared with three modern approaches on both synthetic and real data from the agricultural context. The results are presented in the Tables 6, 7, 8 and 9.

The first one is [23]. In this work a Bayesian approach to ellipsoid fitting was taken, that generalizes well even to higher dimensions. This algorithm maximizes the probability of the output model given the data, thus being robust to noise and out-of-distribution points.

The second one is the implementation of Matlab Ellipsoid Fit [24]. In this work the problem of ellipsoid obtainment is approached with LLSQ, followed by bringing the ellipsoid to the form of a quadric surface.

Finally, the proposed approach was compared with [25]. The authors noted that in a lot of RANSAC implementations either the geometrical or algebraic distance was used in order to evaluate the quality of the model. An alternative approach was proposed, relying on the combination of the axial and Sampson distance, which gave high robustness against outliers and competitive processing time.

1.2 Literature Review

1.2.1 Evolution of Pattern Recognition Methods

Curve recognition and shape recognition is an important part of many modern industrial applications. In particular, ellipses are used to approximate the section of a pipe, human head, and other objects that appear as ellipses [20]. Real-world 3D circles appear as ellipses to a tilted camera, and its parameters provide information about the position, distance and the inclination angle of the said circle.

There is a number of approaches to the shape recognition problem. The most widely used include least squares method, Hough transform and RANdom SAmple Consensus(RANSAC) [17]. Let us briefly cover the main features of these approaches.

The input of all these algorithms is a number of two-dimensional points, and the output of the algorithm for ellipses are five parameters, namely two coordinates, two semiaxes, and the rotation angle. The standard way of using ellipse recognition approaches is the following. First, an image is taken with the robot's onboard camera or other imaging device. Second, the image is segmented. Third, the part of the segmentation mask that corresponds to the ellipse is processed in order to decrease the number of pixels in. Finally, all the points are fed into the recognition algorithm.

Least Square Method

First of all, least squares method is simple, straightforward, fast, easy to interpret. It relies on the quadratic error function that is differentiated with respect to the model parameters. It results in a set of equations that could be solved for the values of the parameters that minimize the quadratic error. While this method is widely used, it has certain drawbacks. The main one is that (in its vanilla form) this method lacks robustness to noise. If presented with a data with a small portion of strong outliers, classical least squares method will incorporate those outliers in the calculation of the optimal model parameters, which significantly degraded the quality of the output model. Consequently, least squares methods cannot be used if significant noise is present.

There are generalisations and improvements on top of LSQ, such as weighted LSQ, but they still present more limitations than the method that was used in the experiments.

Hough Transform

In the 1960s, when modern computers started to appear in research institutions, a novel approach to data analysis was invented. It was widely adopted, previously being unusable by human workers because of heavy computational burden that should be carried in order for this method to work. This method goes by the name of Hough transform. In contrast to the previously mentioned least squares

method, this approach relies on the explicit formation of the discretized parameter space for all the possible models. For such objects as lines the dimensionality of the parameter space is two if these lines exist on the two-dimensional plane. For such objects as circles or parabolas the dimensionality of the parameter space will be equal to three. For such objects as ellipses the dimensionality of the parameter space will be equal to five.

Hough transform algorithm works as follows. First, a so called accumulator is created. The dimensionality of this array is equal to the number of independent parameters in the model. It is usually set to be an array with the integer elements. For instance, for circles, it will be a three-dimensional array. Each of the axis will represent x coordinate, y coordinate and the radii of the circle, respectively.

After the accumulator is formed, the main loop begins. It consists of iterative going through all the input data points. For each of them all the possible models are considered that go through that point. A discrete step corresponding to the discretization of the accumulator array is used. For two-dimensional lines a number of the possible models is proportional to Pi over the discretization step. For circles there are two different discretization steps, resulting in the quadratic number of possible models. It could be noted that with the growth of the dimensionality of the model, the size of the accumulator and the number of iterations grow rapidly. Consequently, for five-dimensional surfaces and curves applying classical Hough transform becomes infeasible.

The second method that is oftem used for pattern and shape recognition is Hough transform [26]. In contrast to the previously mentioned method, this approach relies on the explicitly considering the discretized parameter space for all the possible models. For lines on the plane the dimensionality of the parameter space is two, for circles it is three, for ellipses it is five, which (being implemented in the vanilla straightforward way) is already demanding in terms of the memory requirements.

In the presented work the dimensionality of the objects under consideration is nine, and the data is noisy, making both LSM and Hough transform not applicable.

RANdom SAmple Consensus

Random sample consensus is a de-facto standard for a number of computer vision problems, including perspective transform evaluation, 3D reconstruction [27], and pose estimation. A number of works were devoted to the evaluation of the influence of the noise on the output quality [28]. However, RANSAC is not a single algorithm, it is rather a family of algorithms with their performance, speed, complexity, and convergence, depending solely on the class of the models considered [29], [3].

In this work, a systematic approach is taken towards examining the performance of the random sample consensus approach in application to a specific problem of second order curse fitting.

Compared to Least Squares, RANSAC demonstrates superior robustness to outliers. RANSAC-based approaches are often used under challenging circumstances, see [30]. With these methods it is possible to solve problems where parts of the data directly contradict the resultant hypothesis, which is often the case with imperfect feature correspondence in vision-related problems. Despite these difficulties RANSAC allows for precise and robust two-view image correspondence [31] [32] and pose estimation [33].

Random Sample Consensus (RANSAC) is an iterative method of fitting the model to the data. It is capable of producing quality results even in the presence of high out-of-distribution noise.

In contrast to the Hough transform, RANSAC relies on a relatively small pool of models, lifting the memory consumption burden. Moreover, RANSAC does not take into account all the input points, thus being capable of disregarding outliers. On each step a small subset of data is randomly chosen. After that, a single model is obtained. The size of this subset allows one to specify a single unique model for the object considered. After the model is obtained, a number of data points that are represented well by this model is calculated.

The exact metric for the quality of the fitting can vary. The most straightforward way of evaluating that is measuring the distance from this point to the model. However, it is not always possible to find an analytical expression for that distance, leading to the lengthy iterative evaluation process. There is an alternative, relying on the algebraic distance. Algebraic distance is the value of the polynomial, that describes the object under consideration, be it curve or surface. There are other

approaches to the measurement of the distance from the point to the model, in particular, a mixture of the geometrical distance and the distance measured along the semiaxes of the ellipsoid, as proposed in the paper [25].

Overall, random sample consensus is a standard way of solving a number of computer vision problems, including perspective transform evaluation, 3D reconstruction[27], and pose estimation.

Regarding the drawbacks of RANSAC, it is computationally demanding, especially for complex objects. Thus, a balance should be found between the quality of the output and the performance requirements.

To our knowledge, there are no works on the application of RANSAC to the tangerine volume estimation. This paper aims to address this research gap.

Limitations of RANSAC

However, there are certain shortcomings to this approach. First, it is inherently stochastic, lacking reproducibility. Second, looping through the whole set of input data multiple times can be time-consuming. Third, increasing the number of iterations gives diminishing returns in terms of the output quality. Fourth, there is a number of hyperparameters in RANSAC, particularly the threshold value for inlier counting (see Subsection 2.1 for details) and the assumed fraction of inliers. Both of them have to be tuned manually, and if chosen improperly, can lead to degradation in quality. Fifth, increase in the noise level leads to exponential growth in the number of necessary iterations [34]. Sixth, as the complexity of the parametric model grows, the number of iterations necessary also increases. For line extraction [35] it is enough to sample two points from the data and solve small linear system, but for more complex objects like quadric surfaces the number of the equations in the system grows to 9 [25].

Extensions, Generalizations and Improvements of RANSAC algorithm

A lot of work was done in order to address these issues since the publication of the original work [36]. There are numerous works that are aimed at extending the applicability of RANSAC [30]. One of the possible approaches relies on sampling a set of points with the number that exceeds the minimal necessary [37] in contrast to the baseline RANSAC.

Another extension of the method is Differentiable RANSAC [38]. It relies on the shift towards gradient-based optimization instead of grid search in standard RANSAC. In order to optimize the number of iterations, Variable Sample Consensus (VARSAC) was proposed [29].

In the last years a number of works were concerned with adding adaptiveness to RANSAC, for instance by iteratively updating the output hypothesis considering residuals from the previous iterations [39].

Influence of Noise on RANSAC Results

There are works devoted to the evaluation of the influence of the noise on the RANSAC output for certain problems, such as plane estimation [40]. However, none can be found that focus specifically on the quadric surfaces, that is a specific case in the domain of all the RANSAC applications. First, the number of the parameters needed to describe a unique ellipsoid is 9, which is signifinactly more than for a line or a circle, thus leading to a small probability of sampling a subset of points that do not contain outliers. Second, a conventional way of measuing the distance from the surface to the data point is not geometric, but algebraic, meaning the value of the polynomial that defines the surface. In summary, this work is a case study of the application of the classical baseline RANSAC to a specific problem, that is supposed to serve as an estimate of what could be expected of RANSAC-based algorithms in such applications.

Combining RANSAC with other Image Processing Methods

Shape and pattern recognition were developed since the dawn of the modern computers. The first methods like Hough transform [26] were applied in 1960s to the problem of recognizing tracks in the bubble chamber to analyze the behavior of the particles in the accelerator. The automation of the recognition significantly reduced the burden of data processing for scientists and engineers, making it possible for them to focus more on the substance of their experiments instead of tedious measurements and calculations performed by hand. A number of robust methods were developed, capable of extracting the meaningful information from the noisy data. Suddenly it has become possible to analyze large quantities of data in short time by the means of computers.

Several decades later shape and pattern recognition methods started to be applied in other areas of human activity. In particular, they are nowadays used in document recognition, medicine, monitoring, and security. Regarding the agricultural applications, pattern and shape recognition has a variety of applications. They include activity monitoring of human workers, safety monitoring, disease detection, crop loss prevention and yield estimation.

In order for the supply chain to function properly it is necessary to estimate the yield that can be harvested at a certain facility. This includes the number of fruit and their total mass, which is often simply proportional to the volume. If the yield estimation is performed by human workers, it could be slow and prone to error. Moreover, manual assessment is a boring and tedious task, requiring facility inspection during prolonged periods of time. Automated methods of yield estimation are already introduced into the market, but they are not adopted everywhere yet.

Among all the methods of automated monitoring of the agricultural facilities, computer vision-based methods are applied more frequently than the other. It can be attributed to the advantages of the vision channel of perception. Computer vision-based methods allow for contactless monitoring. Digital cameras are already well-developed and cheap enough for them to be widely adopted. Moreover, there already exists a wide range of methods that could be straightforwardly applied to the problems of classification, detection, segmentation and regression in the agricultural context.

Some of these problems are more straightforward to solve than the other. In particular, the development of a system to detect the object in the greenhouse is streamlined to the collection of the dataset and training and already existing model of YOLO family [41], which often gives a solution as good as the data allows. The classification problem can be solved in the similar manner, considering a case of RGB monocular images. Overall, monocular image processing in agricultural context could be considered to be in almost solved problem. There are a number of works with the classical computer vision approaches being applied to volume estimation, see [27], [3].

When it comes to the stereoscopic setups or point cloud processing, the number of widely available tools becomes smaller. There are two major families of approaches in this field. First of them relies on the classical shape recognition techniques like least squares, Hough transform or Random Sample Consensus (RANSAC) [17]. While the first approach is sensitive to the noise, and the second is heavy on memory consumption, RANSAC is capable of solving the problem of fitting complex objects with a lot of parameters, such as ellipsoids.

1.2.2 End-to-end Volume Estimation

Let us briefly recap the widely adopted approach to the volume estimation with this method, applied to the ellipsoid-like objects like tomatoes and tangerines. First, the point cloud data is obtained by a camera that is aligned with an RGB sensor. After that, all the objects of interest are detected. The point clouds that correspond to this object are extracted. Finally, RANSAC is applied to fit a single model to the object. This method allows one to estimate the volume, since the model includes three semiaxes of the object, and if it could be approximated as an episode, the volume of the fitted model will be close to the volume of the object of interest.

RANSAC relies on multiple samplings of a small subset of the input data. For each subset, a single model is fitted. After that, it is measured against all the input data points and it is evaluated how good or bad does this model describe the entirety of the input. The output of the method is the model that corresponds the best to the input points, excluding the out-of-the-distribution noise, which is inevitable in real data in a greenhouse facility. This method is widely used and developed, still

receiving attention from the community, and it is modified in certain details, such as the method of measuring the distance from the model to the point [25].

With all the advantages of this method, there are a number of drawbacks. Due to the iterative manner of model generation, RANSAC can require significant computational power. Constructing the model of a high-dimensional object is heavy on computations as well. The development of a method to obtain a model from a set of points requires manual engineering for each new type of objects under consideration. Moreover, RANSAC relies on a number of assumptions and hyperparameters that should be manually tuned.

It could be noted that in order to evaluate the volume of the object it is not necessary to construct its full model. A representation of the object should be constructed, but it is not required to include the coordinates and the orientation. Recent advancements in neural networks make it possible to develop an end-to-end method of volume estimation, bypassing the construction of the full model.

The nature of point cloud data requires a specific approach for it to be processed by a neural network. While some of the most widely used types of neuron networks, like fully-connected or convolutional, are capable of approximating complex functions, learning dependencies, and producing state-of-the-art results in a number of applications, they cannot be directly applied to the raw point cloud data. Point clouds are sets of three-dimensional points, in certain cases with color. The model that will process this data should ideally be invariant to the permutations in the data. It was shown to be difficult to enforce such a property to a standard neural network. They could be applied to the data if the point cloud is transformed with the methods like voxelization, but this approach leads to significant growth of the computational complexity. Point clouds are inherently sparse, and this property should be taken into account while the method is developed.

In 2016 a novel approach PointNet [42] was proposed. It relies on the processing of all the data points individually and then extracting the necessary features from the point cloud of variable size, which is not possible with the other, standard approaches. Another novelty of PointNet is the application of two smaller networks, that are used to generate rotation matrices that are transforming the data before it is fed into the main network. Further advancements in the development of the PointNet family include joining them with the generalization of convolution [43], that was developed specifically to fit the demands of the point cloud data. With

these generalized convolutions the local features of the point cloud could be taken into account, which is beneficial for the end result.

In this work a PointNet++-like model was used.

1.2.3 Disease detection

There is a variety of great works in the field, let us briefly cover some of them. One of the closely related ones is [27]. In this work the mass and volume estimation relies on the depth images of the cherry tomatoes on a conveyor belt. As a result, a relation between the tomato mass and volume was established. While this result is applicable in certain cases, non-invasive or contactless volume estimation requires a solution that will work without a conveyor belt, while the tomato is hanging in the air. There is a number of works that rely on conveyors and contrastive background, including [44], [11] and [12].

Another related paper is [18]. A sequence of classical algorithms is used, including Harris corner detection, SIFT keypoint descriptor and SVM. The method is applied in the problem of pineapple 3D reconstruction. After obtaining 3D points of the pineapple surface, the center, orientation, and radii of the fruit are estimated with the help of the Least Squares. Judging by the metrics, this approach works well on the given type of data. However, such fitting methods as Least Squares are prone to error while presented with the noisy data. And the point clouds in the greenhouse are very noisy, which is evident from the Figure 3.15.

There is a number of reasons behind this. First, the tomatoes themselves are quite small, while compared to the pineapples, thus limiting the resolution of the data. Second, there are leaves and other vegetation around, partially obstructing the view. Third, there could be errors in distinguishing between one tomato and another, because they often hang in direct contact. Fourth, in order to assure sufficient monitoring speed, the inspection has to be performed as the robot moves, which leads to various types of noises and disturbances in the data, including motion blur and shaking of the camera sensor. Thus, a robust method should be proposed, that allows for the effective outlier exclusion [29].

A very closely related paper by Han, Kan and Wang is [20] which is an adaptive algorithm on top of RANSAC. A two-ellipsoid method with one of them being

fitted inside the jujube and another outside is proposed. However, in this work a color-based point cloud segmentation method is used, which could be inconvenient in real-world scenarios. In the family of RANSAC-based methods also [5] should be noted.

Chapter 2. RANSAC for Quadric Curves and Surfaces

2.1 Vanilla RANdom SAmple Consensus Algorithm

Let us describe the classical RANSAC algorithm, as per paper by Fischler and Bolles [17]. It was previously mentioned, that least squares suffer from sensitivity to outliers, and Hough transform from the rapid growth of the accumulator size. This is where random sample consensus-based algorithms could be used. In contrast to the Hough transform, where all the parameter space is considered explicitly, random sample consensus relies on the chosen models from all the possible ones from the perimeter space, thus saving memory. At the same time, in contrast to the least squares methods, random sample consensus is not obliged to take into account all the points of the input data. It relies on the sampling of a number of subset of data. For each subset, a separate model is created. The number of the points in this subset has to be such that it determines a single unique model for lines. It will be two points for a line, and for ellipse it will be five points. After the model is obtained, it is calculated how many of the input data points are described by the model.

This notion of how the data points are described by the model is in itself a variety of different approaches. The first way to tell if the data point is described by a model is to measure the distance from this point to this model, and the second way of doing this relies on the so-called algebra distance. Considering a case of an ellipse, the equation describing it is a second-order equation with two independent variables. If the left side of this equation is considered as a polynomial, the value of this polynomial is zero on the ellipse and not zero inside and outside. The value of the polynomial on the data point could be considered as a distance from the point to the model. There are also alternative approaches to the measurement of the distance from the point to the model. In particular, there is a mixture of the geometrical distance, and the distance measured along the semiaxes of the ellipsoid in the paper [25].

There are also ways to count the fitness of the model for the dataset. After the distances are obtained with one of the methods described above, a single scalar metric should be calculated. It could be a sum of indicator functions. If the distance from the point to the model is below a certain threshold, the point is considered

to be fit for the model. The best model among the same ones is the one that has the biggest number of inliers. However, there are alternative approaches to counting the fitness of the model for the data. For instance, monotonous decreasing function could be used as a weight function instead of the threshold function, such as the exponential of minus distance from the point to the model.

Random sample consensus algorithm is inherently stochastic. It relies on the random generator, and also if part of the input data is outliers, it will produce a reasonable output only with a certain probability. Let us briefly cover the question of the numerical evaluation of this probability. Let n be the number of the input points and α be the share of the non-noise data points, k be the number of points that describe a unique model, and P be the desired probability of obtaining an output model that corresponds to the real object. The probability of sampling a single point that is an inlier is α. The probability of sampling k points that are all inliers is α^k. The probability of sampling at least one outlier point is equal to $1 - \alpha^k$. Sampling even a single outlier point will result in the incorrect model. If m subsets of points are sampled, the probability of all of them containing at least one outlier is equal to $(1 - \alpha^k)^m$. This formula could be equated to $1 - P$, which is the acceptable probability of not producing a feasible model: $(1 - \alpha^k)^m = 1 - P$. Taking logarithm from both sides yields the following result for the requred number of sampling iterations: $m = \frac{\ln(1-P)}{\ln(1-\alpha^k)}$

It is evident that as the desired probability grows, the number of the necessary samplings increases.

Let us first briefly describe the RANSAC algorithm according to [36].

RANSAC is a well-known algorithm for obtaining parametric objects in the presence of noise. Unlike methods such as least squares that fit a single model to all the data, including outliers, RANSAC generates multiple models based on small random subsets of data. This increases the probability of selecting points and models belonging to the real object.

RANSAC algorithm for ellipsoids consists of the following steps:

1. **Sampling a random subset of data points.** The size of the subset is chosen in such a way that a unique model can be precisely fitted to it. For ellipsoids 9 points are required.
2. **Fitting the model to the selected subset.** An ellipsoid can be represented as a set of points satisfying the equation in the form of

$p_1x^2 + p_2y^2 + \cdots + p_9z = 1$. With the aforementioned 9 points a system of linear equations is formed, that is solved with standard linear algebra.

3. **Checking the model.** Since there are multiple types of quadric surfaces other than real ellipsoid, a number of additional conditions are checked, specifically the correct signs of certain invariants [45].
4. **Assessment of the model quality.** The fitness of the model is the numerical metric of how well it represents the entire point cloud. This assessment is performed by counting the number of points that lie close to the surface of the ellipsoid model. To determine this proximity, a threshold-based point counting is performed based on the algebraic distance between the points and the predicted surface. Here the simplest approach to evaluation is used, that was proposed in the classical work of Fischler and Bolles [36].
5. **Iterative search.** Steps 1-3 are repeated a number of times, and the best model is chosen by the criteria of the number of well-represented points.

The detailed pseudocode is given in Algorithm 1.

2.2 Quadric Curves

Representing the parameters of the ellipse equation allows to rewrite the equation with an arbitrary scale parameter F set to 1. Five such equations for a system of equations with a unique solution for the ellipse parameters if the points are in non-degenerate configuration. Finally, the center, semiaxes and the rotation angle are obtained, following [46].

A number of experiments were conducted. The points of the ellipse were corrupted by additive normal noise with increasing variance. The results of the first series of experiments are presented in the Figure 1(a). As the variance of the noise becomes comparable to the size of the ellipse, the intersection over union becomes unacceptably small for the real-world scenarios.

The second result is presented in the Figure 1(b). A comparison is made between the performance of the algorithm on the same data, but with different number of iterations. It could be noted that while the overall trend of the quality decrease is evident in all the curves, the orange and green curves (that correspond

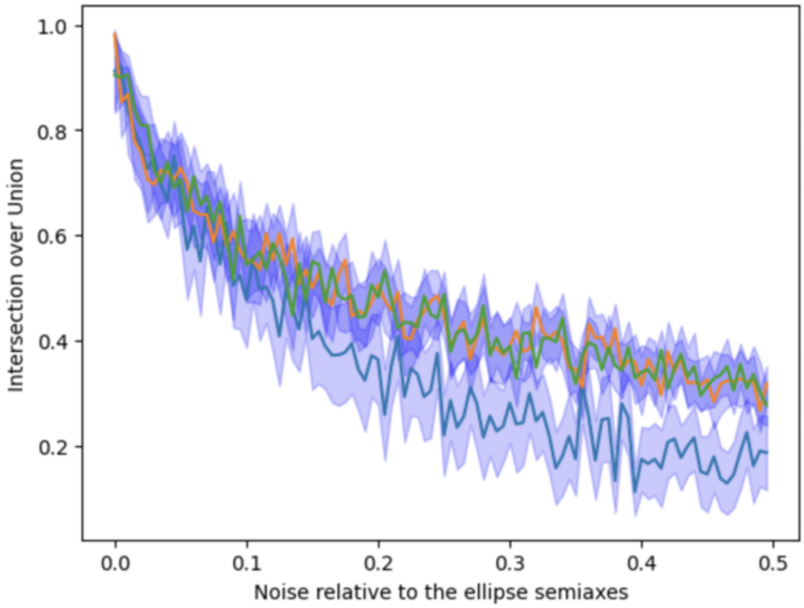


Figure 2.1 — For each noise level, a number of random sets of data were generated, and the variances were calculated. A clear trend with the decrease in the Intersection over Union can be seen as the ratio between the noise variance and the major semiaxes of the ellipse grows.

to 50 and 500 iterations) have lower decrease rate. Moreover, the difference between the performance with 50 and 500 iterations is not significant.

After the coefficients of the polynomial that describes the ellipse as a second-order curve were obtined, the center coordinates, the semiaxes and the angle of the ellipse are expressed as follows:

$$a, b = \frac{-\sqrt{2\big(AE^2 + CD^2 - BDE + (B^2 - 4AC)F\big)\big((A + C) \pm \sqrt{(A - C)^2 + B^2}\big)}}{B^2 - 4AC}$$

$$x = \frac{2CD - BE}{B^2 - 4AC}$$

$$y = \frac{2AE - BD}{B^2 - 4AC}$$

$$\theta = \arctan\left(\frac{-B}{C - A}\right)$$

2.3 Quadric Surfaces

Now let us describe the ellipsoid model extraction, that is performed at each step of RANSAC, generally following[47].

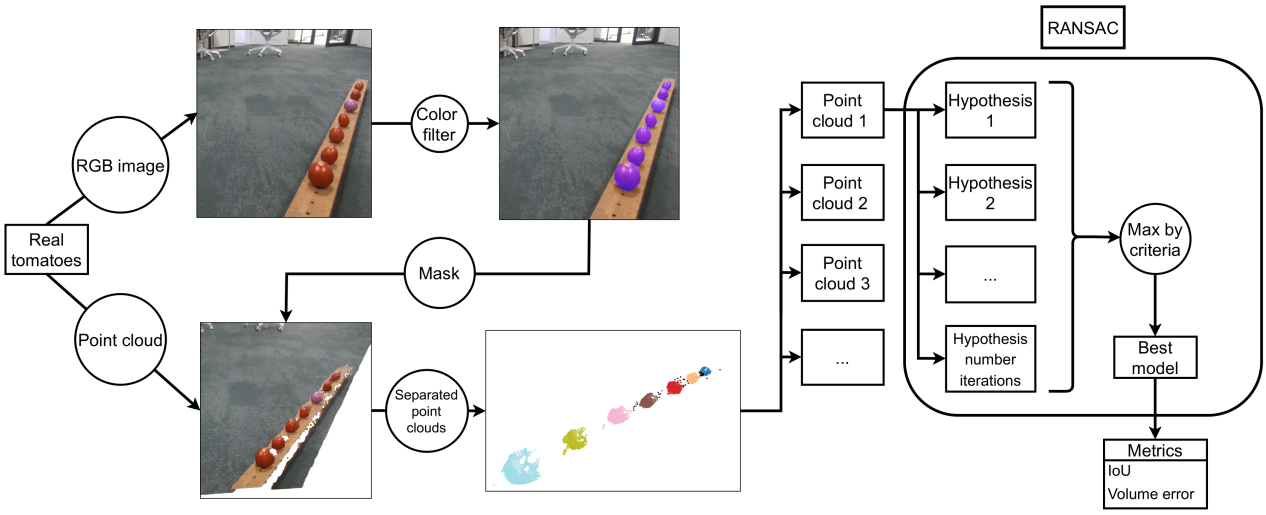


Figure 2.2 — The scheme of the experimental setup. First, an RGB and depth-images are taken. Second, a color-based filter is applied to the RGB image and the corresponding parts of the point cloud are cut. Finally, RANSAC is applied to each of the objects and the results are evaluated against the markup.

The proof will begin with an ellipse in the XY-plane and then be extended to the ellipsoid.

First, consider a set of points on the unit circle, P_c:

$$P_c = \{r_c \in \mathbb{R}^2 \mid r_c^T r_c = 1\}. \tag{1}$$

Let us introduce

$$\begin{aligned} &t \in \mathbb{R}^2 \ - \ \text{translation vector}, \\ &s \in \mathbb{R}^2 \ - \ \text{scale vector}, \\ &\alpha \in [0, 2\pi) \ - \ \text{rotation angle}. \end{aligned} \tag{2}$$

With the matrix operations it is possible to transform unit circle into an arbitrary ellipse.

Let us unwrap the full form of these matrices.

$$S = \begin{pmatrix} s_1 & 0 \\ 0 & s_2 \end{pmatrix} \quad - \text{ scale matrix, where } s_1 \text{ and } s_2 \text{ are parameters of scale vector } \mathbf{s},$$
$$R = \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{pmatrix} \quad - \text{ rotation matrix.} \tag{3}$$

Let P_e be the points of the ellipse. Scaling, rotation and translation of the unit circle yields

$$P_e = \{RSr_c + t \mid r_c \in P_c\}. \tag{4}$$

From (1) a matrix equation of the ellipse can be obtained. For $r \in P_e$ the equation of the ellipse takes form of

$$(r-t)^T[R^{-1}]^T[S^{-1}]^T S^{-1} R^{-1}(r-t) = 1. \tag{5}$$

Since R is an orthogonal matrix and S is a diagonal matrix, $R^{-1} = R^T$ and $S^T = S$.

$$(r^T - t^T)RS^{-2}R^T(r-t) = 1. \tag{6}$$

Let us introduce $V = RS^{-2}R^T$ for convenience. Inserting this into equation (6) yields:

$$r^T V r - 2t^T V r + t^T V t = 1. \tag{7}$$

Let us also introduce vector $v^T = t^T V$ and scalar $v_0 = t^T V t$.

$$r^T \left[\frac{V}{v_0 - 1}\right] r - 2\left[\frac{v^T}{v_0 - 1}\right] r + 1 = 0. \tag{8}$$

Finally, let us define a matrix $M = \frac{V}{v_0 - 1}$ and a vector $m^T = \frac{v^T}{v_0 - 1}$. Now it is possible to formulate the general matrix equation of an ellipse as follows:

$$r^T M r - 2m^T r + 1 = 0. \tag{9}$$

From this equation all the parameters needed to describe an arbitrary n-dimensional ellipsoid can be extracted, specifically translation, rotation and scaling.

Let us evaluate M^{-1} first.

$$M^{-1} = \left[\frac{V}{v_0 - 1}\right]^{-1} = (v_0 - 1)V^{-1} = (v_0 - 1)RS^2R^T. \tag{10}$$

It could be observed that

$$t = M^{-1}m. \tag{11}$$

Considering the eigendecomposition of M yields

$$\begin{aligned} M = \frac{1}{v_0 - 1}V = \frac{1}{v_0 - 1}RS^{-2}R^T = \\ R\left[\frac{1}{v_0 - 1}S^{-2}\right]R^T = Q\Lambda Q^{-1}. \end{aligned} \tag{12}$$

From (12) it could be seen that the eigenvectors matrix of M is exactly the rotation matrix R.

$$R = Q. \tag{13}$$

Moreover, the matrix of the eigenvalues M and S are diagonal matrices:

$$\Lambda = \frac{1}{v_0 - 1}S^{-2}. \tag{14}$$

Thus, it becomes possible to extract the scale matrix S from the eigenvalues matrix Λ.

Scale matrix S consists of squeezes and stretches along orthogonal axes of the ellipse, or just radii. Each eigenvalue can be presented as

$$\lambda_i = \frac{1}{v_0 - 1}s_i^{-2}. \tag{15}$$

In order to extract the radii, it is necessary to find v_0 first. Since

$$v^T V^{-1} v = v_0. \tag{16}$$

And

$$m^T M^{-1} m = \frac{1}{v_0 - 1} v_0. \tag{17}$$

Finally, an expression for s_i can be found.

$$s_i = \sqrt{\frac{m^T M^{-1} m - 1}{\lambda_i}}. \tag{18}$$

The solution holds with respect to the permutations of the axes of the ellipse. First, let us establish the notation conventions.

- T_k is the markup model for each of the K tomatoes. For each tomato there are 146 point clouds PC_k^i.
- m_j denotes an ellipsoid model number j.
- $IoU(T_k, m_j)$ stands for 3D Intersection over Union of the tomato markup T_k and ellipsoid model m_j.
- $d(m_j, x_i)$ is the algebraic distance between the surface of the ellipsoid m_j and data point x_i.
- $TotRelVolErr$ for a full scene with K tomatoes denotes

$$\frac{|\sum_{k=1}^{K} V_{predicted} - \sum_{k=1}^{K} V_{true}|}{\sum_{k=1}^{K} V_{true}}$$

2.4 Speeding Up Model Evaluation

According to the algorithm described above, all the data points of a point cloud that corresponds to a single tomato have to be combined with all the model

hypothesis for this point cloud. It could be done in a simple *for* loop. However, there is a way to speed up the computations.

The ellipsoid model obtainment algorithm follows the classical approach [47], while filtering out quadrics of improper type [45]. Since the value of the ellipsoid polynomial is equal to

$$p_1x^2 + p_2y^2 + \cdots + p_9z - 1 = 0,$$

it could be represented in the form of $(\vec{P}, \vec{X})$ for each pair of the point x_i and model m_j, where

$$\vec{P} = \begin{pmatrix} p_1 \\ \vdots \\ p_9 \\ -1 \end{pmatrix}, \quad \vec{X} = \begin{pmatrix} x^2 \\ y^2 \\ \vdots \\ z \\ 1 \end{pmatrix}$$

Stacking all the models in the form of P_i along the vertical axis into a matrix $\mathbf{P}$ and all the points in the form of X_j along the horizontal axis into a matrix $\mathbf{X}$ reduces the calculation of all the distances to a single matrix multiplication: $\mathbf{P}\ \mathbf{X}$.

ij-th element of this matrix will be equal to the value of the polynomial of the j-th ellipsoid model on the i-th data point.

Because of the matrix $\mathbf{P}$ having size of ransac_iterations_num $\times$ 9 and matrix $\mathbf{X}$ having size of 9 $\times$ data_points_num, the asymptotic complexity of the fitness calculation for all the models is

$$\Theta(\text{ransac\_iterations\_num} \times \text{data\_points\_num})$$

Table 1 presents the comparison of the performance of the algorithm under different noise and with changing number of iterations. It could be noted that first, the quality decreases as the noise becomes heavier, and second, the quality improves with the increase in the number of iterations. An important finding is that the difference between 5 iterations and 50 is much more significant than the difference between 50 and 500 iterations.

In this work the influence of noise on the quality of the output of RANSAC method was examined. This method was applied to the problem of second-order curve recognition, specifically an ellipse. RANSAC for ellipses was implemented

in Python. It relies on the representation of the input data as zero, first and second-order monomials. It follows the standard RANSAC pipeline with inliers being counted by a binary threshold. A set of synthetical noisy data was generated. A number of experiments on was conducted in order to evaluate the influence of that noise on the quality of the output. The results suggest that the increase of the number of iterations leads to diminishing returns in quality. Consequently, it becomes less and less reasonable to allocate more computational resources as the convergence is being reached. Furthermore, the dependence of the output metric on the noise intensity was obtained in a series of tests with increasing noise. It suggests that the quality in terms of IoU drops significantly as the noise variance approaches the order of the size of the ellipse's semiaxes.

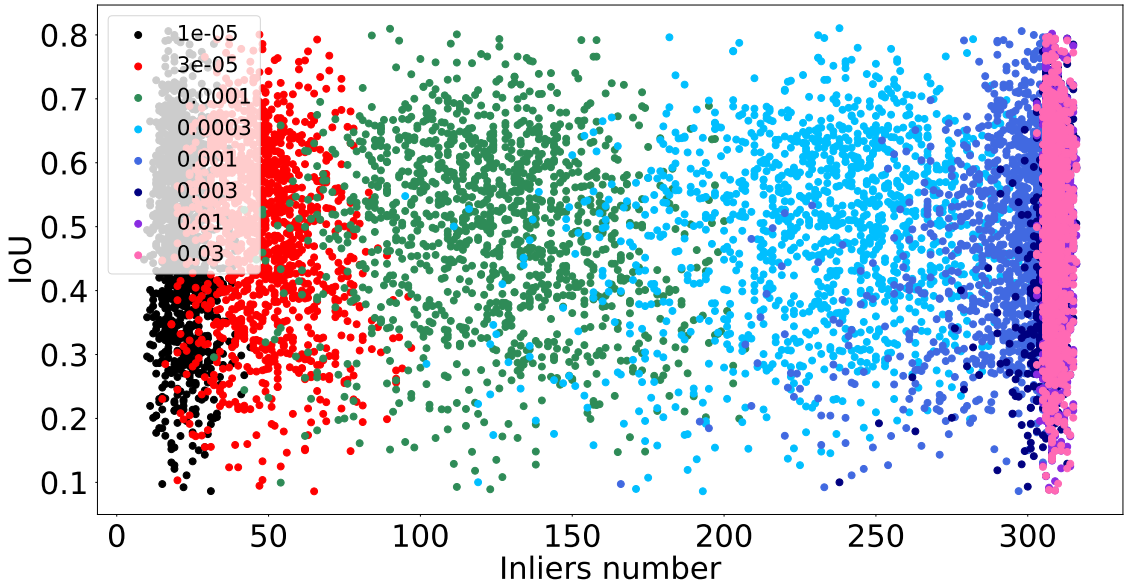


Figure 2.3 — Distribution of model IoU versus inlier number for different thresholds. No obvious trend could be observed here. However, the results in the Figure 3.18 suggest the presence of IoU improvement over the number of iterations.

2.5 Choosing optimal hyperparameters

RANSAC has a number of hyperparameters, one of them being the threshold for the inliers if inlier counting method is used in order to assess the quality of the model. Manual hyperparameter picking cannot be expected to maximise the

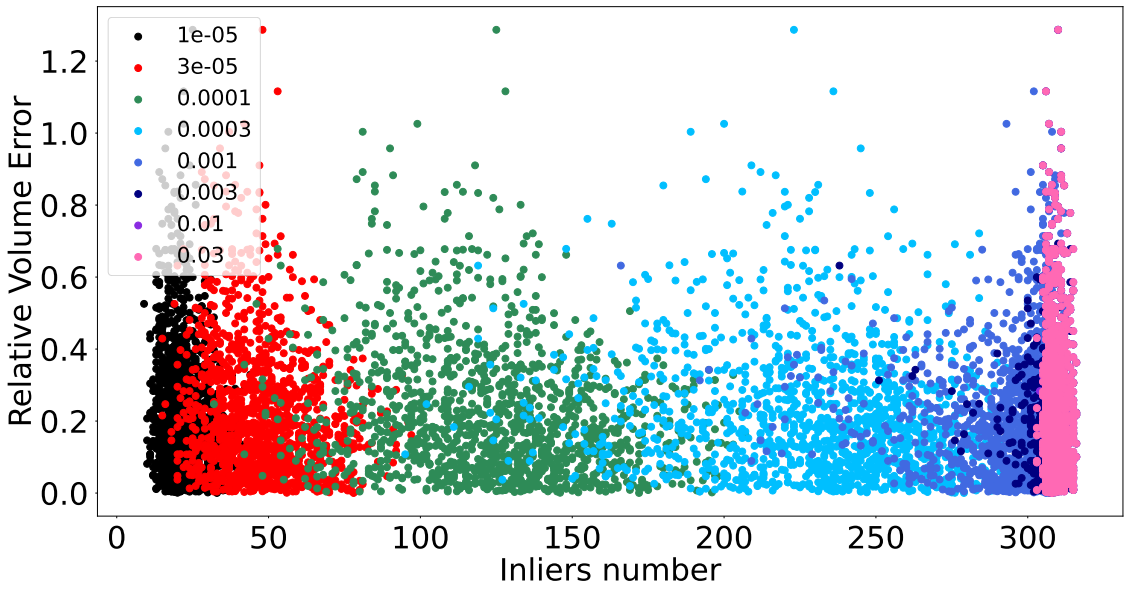


Figure 2.4 — Distribution of model volume error versus inlier number for different thresholds. A vague pattern could be observed, that the models with higher inlier number correspond to lower volume error.

Figure 2.5 — Distribution of (a) iou and (b) error against inlier number in RANSAC.

performance on a novel set of data. This issue was addressed with semi-automated hyperparameter search, which is a search over the threshold values in a series of numerical experiments.

The target metric that was maximized during this process is IoU between the predicted tomato and the real one. The best fitting tomato model among M models constructed from a n-point cloud with threshold value t is the one that maximises the number of inliers:

$$\hat{T}_k = \arg\max_{j\in[1,\cdots,M]} \sum_{i=1}^{n} \mathbb{I}(d(m_j, x_i) \leqslant t)$$

The optimal threshold value among p candidates reads

$$t^* = \arg\max_{t\in[t_1,\ldots,t_p]} \frac{1}{K} \sum_{k=1}^{K} IoU(T_k, \arg\max_{j\in[1,\cdots,M]} \sum_{i=1}^{n} \mathbb{I}(d(m_j, x_i) \leqslant t))$$

In the similar manner other criteria can be used, in particular average volume, which is the metric that the end user is mostly interested in.

Algorithm 1: RANSAC algorithm description

Input: 3D points, k (number of points that uniquely define a model), filtering criteria (restrictions on acceptable ellipsoids), iterations number, inlier threshold

Output: ellipsoid model

Result: quadric polynomial, center coordinates, radii, orientation

1. Points sampling;

points_samples = [];

for $i \leftarrow 0$ **to** *iterations number* **by** 1 **do**

  Randomly select a subset of k points. Save the subset into an array *points_samples*;

end

2. Estimation of the parameters of the models;

hypotheses = [];

for $i \leftarrow 0$ **to** *iterations number* **by** 1 **do**

  For each subset of points from *points_samples* form a $k \times k$ system and solve it; Calculate invariants

  if *invariants have proper signs* **then**

    Save the corresponding polynomial P, matrices and parameters M_i into an array of *hypotheses*;

  else

    Continue;

  end

end

3. Model checking and validation;

$Models$ = [];

m = size(*hypotheses*) **for** $i \leftarrow 0$ **to** m **by** 1 **do**

  if M_i *semi-axes from hypotheses satisfy the conditions on size and semiaxis ratio* **then**

    Save $Model_i$ to array $Models$

  else

    Continue;

  end

end

4. Inliers analysis;

Loop through the models and find i such that the model $Model_i$ maximises $\sum_{j=1}^{m} \mathbb{I}(|P_i(x_j)| < \text{threshold})$, where $P_i(x_j)$ is the value of the polynomial number i on the data point number j.

Return this model

Iterations number	Noise 0.3	Noise 0.5	Noise 0.8
5	0.403	0.219	0.174
50	0.499	0.450	0.314
500	0.550	0.487	0.339

Table 1 — The values of the Intersection over Union between the real ellipse and the predicted one for the noise of different severity (from 0.3 of maximal amplitude to 0.8).

Chapter 3. Volume estimation

The experiments with the real data were conducted in accordance with the scheme in the Figure 2.2: RANSAC for ellipsoids is applied after cutting a part of the point cloud that consists of the object only.

The metrics that were used to assess the method's performance are the following. First, it is the Intersection-over-Union between the real ellipsoid and the output of the algorithm. Second, it is the ratio between the predicted and real volume of the ellipsoid. Finally, it is the relative error in volume for the whole set of ellipsoids:

$$\frac{\sum V_{predicted} - \sum V_{real}}{\sum V_{real}}. \tag{22}$$

The hyperparameters used are the following:

- Threshold for the algebraic distance is 0.001
- Maximal iterations number is 10000. The results for the other iteration numbers are presented in the tables below
- Maximal acceptable ratio between the biggest and the smallest semiaxis is 2 (used for filtering)
- Maximal ratio between the maximal semiaxis of the ellipsoid and the maximal distance between the data points is 5 (used for filtering)

Figure 3.2 presents mutual distributions of a hypothesis IoU versus number of inliers for that model for different noise levels. For very strong noise, the constellation of dots in the plot appears as an almost vertical blob. It is bounded from the right, meaning that no model has lots of inliers. Consequently, the influence of the stochastic nature of the algorithm can lead to unsatisfactory results.

On the other hand, with light noise a clear trend can be observed: with the growth of the number of inliers the quality in terms of IoU grows as well.

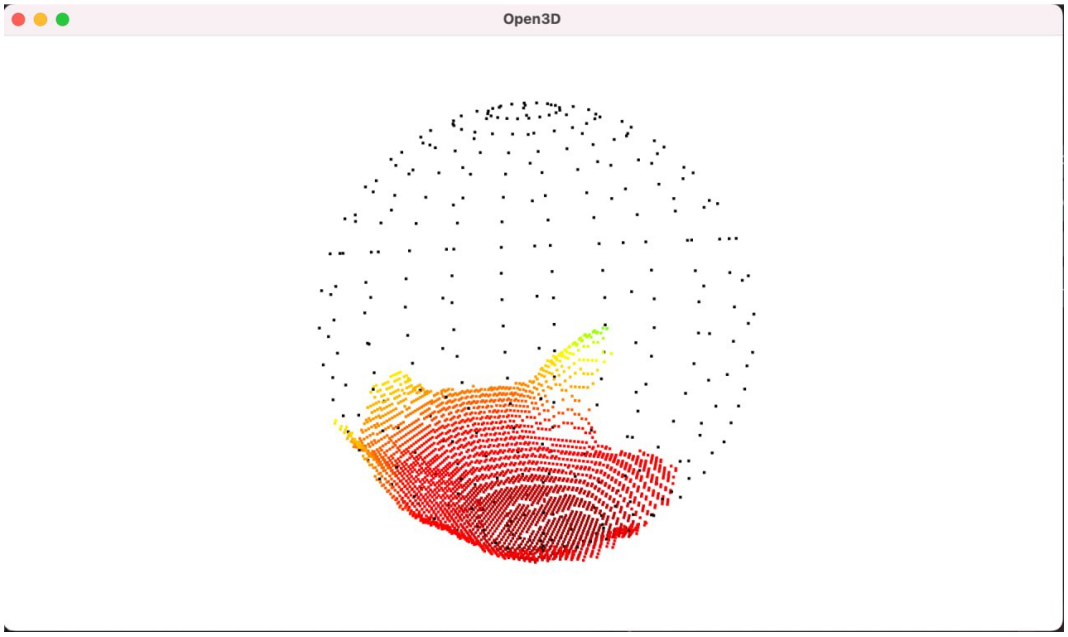


Figure 3.1 — Output of RANSAC on a point cloud from the real data. Notice that despite the complexity of the data, i.e. small section of the ellipsoid covered, non-uniform density, surface distortion — the output is reasonably good. This exact sample was cherry picked for demonstration purposes.

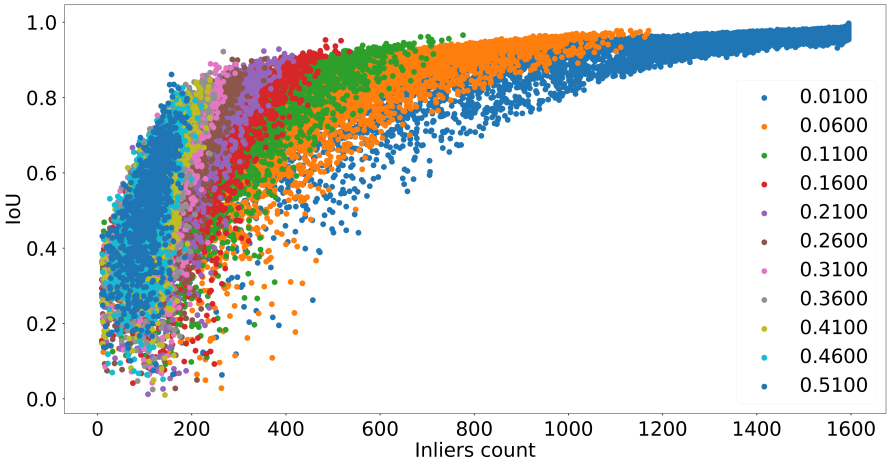


Figure 3.2 — Mutual distribution of IoU and number of inliers for full ellipsoids with different noise amplitudes. A trend can be observed that generally the more inliers there are for the model, the higher the IoU with the true ellipsoid is.

Iterations	Full ellipsoids with noise				Ellipsoid segments with noise			
	0.0	0.01	0.26	0.41	0.0	0.01	0.03	0.05
10	1.0	0.9518	0.5428	0.5238	0.9978	0.2403	0.1582	0.1022
100	1.0	0.9523	0.5714	0.5569	0.9989	0.3079	0.1555	0.0969
1000	1.0	0.9427	0.5497	0.5219	0.9995	0.3885	0.2086	0.1333

Table 2 — Average IoU with different number of iteration (10, 100 and 1000) on different data: full noised ellipsoids and noised ellipsoid segments, generated with ray tracing

Iterations	Full ellipsoids with noise				Ellipsoid segments with noise			
	0.0	0.01	0.26	0.31	0.0	0.01	0.03	0.05
10	0.0	0.0071	0.5214	0.7591	0.0008	0.2173	0.6498	0.6690
5000	0.0	0.0267	0.5527	0.1307	0.0	0.1394	1.7770	0.5463
10000	0.0	0.0091	0.0742	0.0336	0.0	0.1394	0.3276	0.4036

Table 3 — Average Relative Volume Error with different number of iteration (10, 100 and 1000) on different data: full noised ellipsoids and noised ellipsoid segments, generated with ray tracing

Iteration number	**Precision**	**Recall**
1250	0.5346	0.8564
2500	0.5611	0.8445
5000	0.5697	0.8601
7500	0.5705	0.8623
10000	0.5798	0.8856

Table 4 — Precision and Recall values for inliers for different number of iterations.

3.1 Experimental Results

It could be noted that in the transition from the full ellipsoids to the segments, that are closer to the real data, the mutual distribution is blurred significantly, see Figure 3.3.

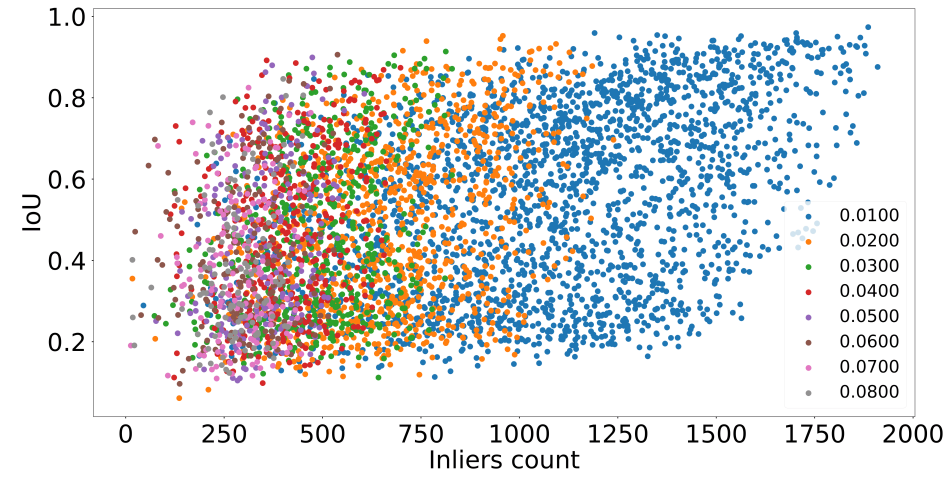


Figure 3.3 — Mutual distribution of IoU and number of inliers for segments of ellipsoids with different noise amplitudes. In comparison with the Figure 3.2 the trend is much less prominent.

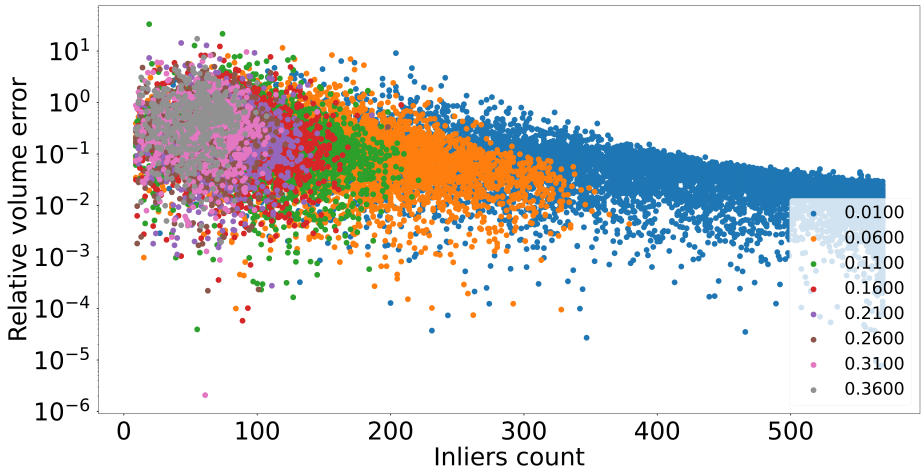


Figure 3.4 — Mutual distribution of Relative Volume Error (RVE) and number of inliers for full ellipsoids with different noise amplitudes. With the growth of the number of inliers the average error declines.

Similar trend, but with the declining relative volume error can be observed on the Figure 3.4. For strong noise the number of inliers is small and the overall volume error is high, and for the light noise it is the exact opposite.

It could be observed that there it a noisy, but distinct trend of the increase in the model quality with the increase of the inlier number. However, due to the stochastic nature of the algorithm, it is not guaranteed that the best in terms of the IoU model will be the same that has the biggest inlier number.

Table 4 presents precision and recall for the inliers for the best fitted models. An improvement in the quality can be observed as the number of iterations grows. However, it is associated with a growing compuptional burden, so a balance should be found for real applications between the quality of the output and the computational demands of the algorithm.

Now let us observe the dependence of the quality of the best model as the number of algorithm iterations increase.

The results for the different number of iterations are presented in the Table 2.

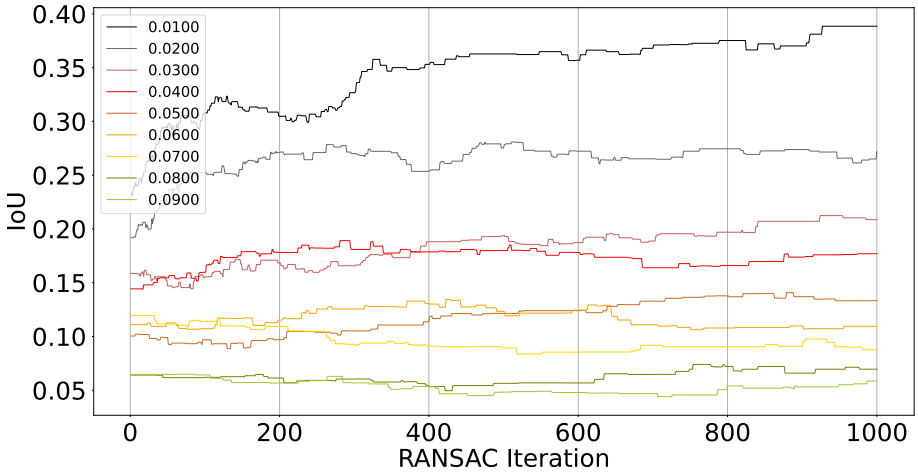


Figure 3.5 — Dependence of IoU of the best model from the number of iterations for synthetic segments for different levels of noise. It is worth noting that overall with the growth of the iteration number there is a slight, but noticeable improvement in IoU.

The best depth perception for Intel RealSense D435i is declared to belong to the range between 0.6 and 6 meters. However, Figure 3.7 as well as Figure 3.8 suggest that the variance in the distribution in the relative volume error grows significantly beyond approximately 1.5 meters. It can be explained by the decline in the infrared pattern density as the distance from the emitter increases.

The real applications are bounded by even more strict constraints in terms of the distance between the camera and the object. Since the distance between the rows of the tomatoes is limited by approximately 2 m, and the vegetation should be taken into account, the distance between the camera and the plant in the real environment is supposed to be from 0.5 to 1.5 m.

It could be seen from the Figure 3.12 that despite high variance in the relative volume error the average measured volume matches the real one very precisely.

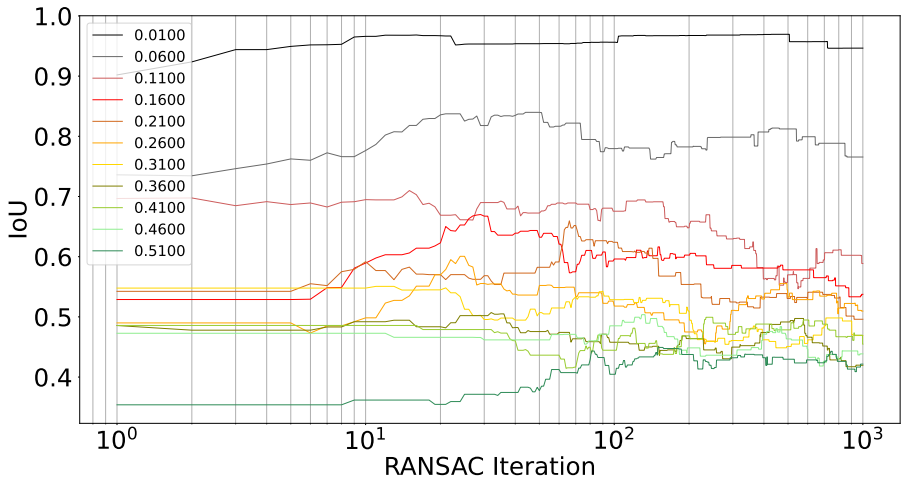


Figure 3.6 — Dependence of IoU of the best model from the number of iterations for full synthetic ellipsoids for different levels of noise. x axis is logarithmic for better view. For light noise the IoU grows almost to 1.0 very rapidly (black line). For heavy noise the growth is much slower (green line).

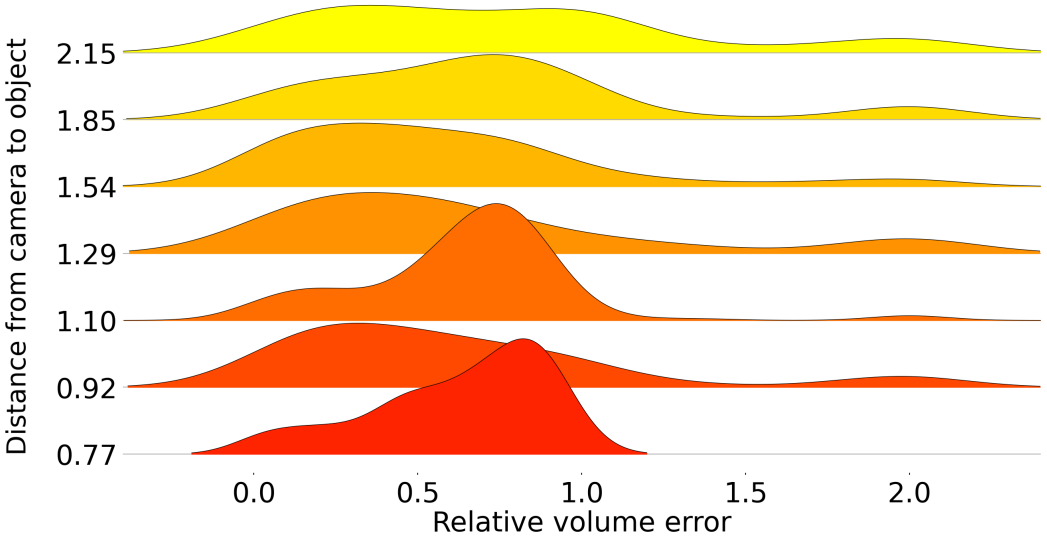


Figure 3.7 — Distributions of Relative Volume Error for different distances from the camera for the point clouds filmed with 55 degrees between the camera direction and line of tomatoes

This result suggests that the method considered is applicable even under fairly noisy circumstances.

Figure 3.10 presents the results of averaging the estimated volumes among all the point clouds for each tomato. The real volume is marked with a black dot. Despite high variance, the average volume is estimated with acceptable precision.

In this work an empirical study is carried out, aimed at examining the influence of noise on the performance of RANSAC method on quadric surfaces. A method

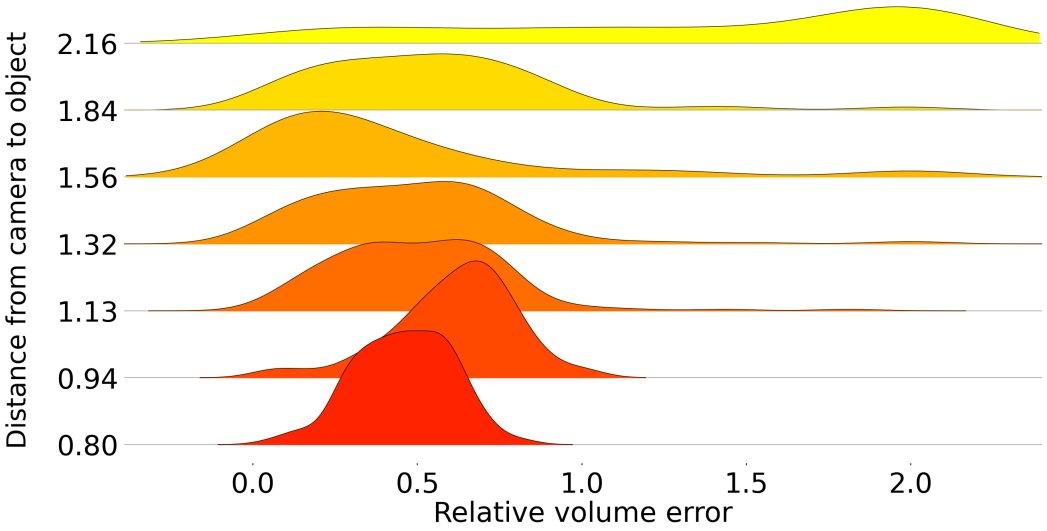


Figure 3.8 — Distributions of Relative Volume Error for different distances from the camera for the point clouds filmed with 55 degrees between the camera direction and line of tomatoes

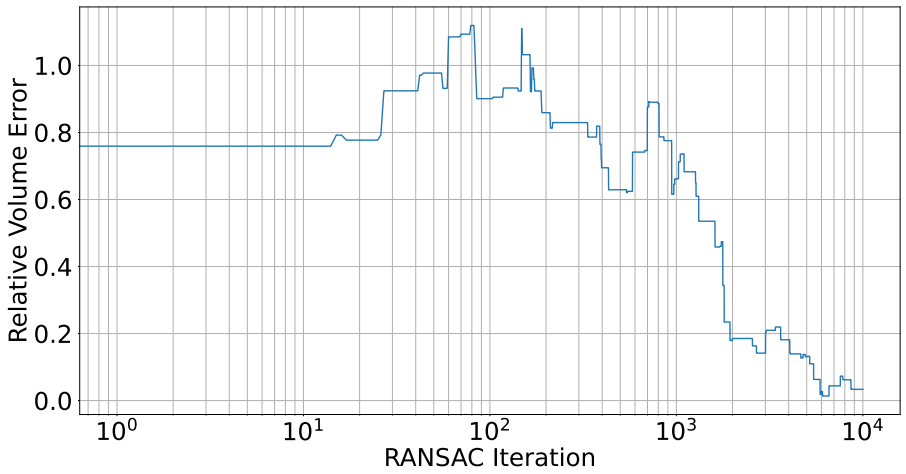


Figure 3.9 — Dependence of the Total Relative Volume Error (sum over all the ellipsoids) on the number of iterations of RANSAC (logarithmic scale) for full noised synthetic ellipsoids. All the ellipsoids in the set have volume in the same order of magnitude. The noise level is 0.41.

for synthetic data generation with ray tracing was developed. The dependence of the error on the noise was examined on the synthetic point clouds. Experiments on the real point clouds were carried out in the agricultural setting. RANSAC for quadric surfaces was implemented in Python.

As one may expect, the results suggest that noisy data is more challenging for the algorithm. The numerical evaluation of this phenomenon is given in the Results section 3.2.2. All in all, the qualitative findings are the following:

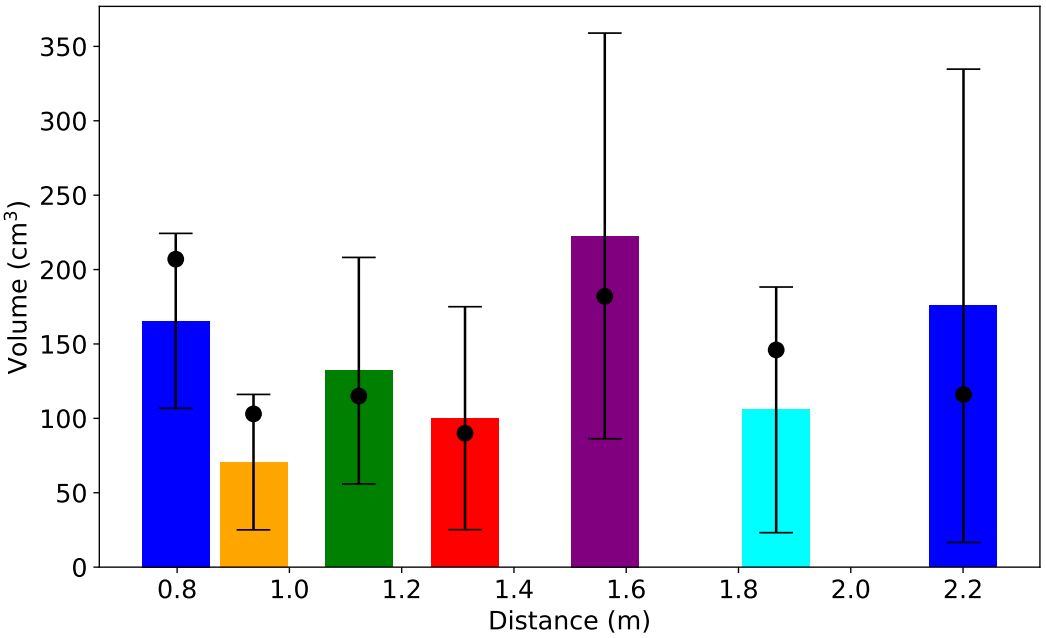


Figure 3.10 — For each of the objects from the real dataset the volume was estimated for each of the point clouds. The average volumes with the error bars are listed along the axis represenating the distance from the camera to the object. The true volume for each of the tomatoes is marked with a thick black dot.

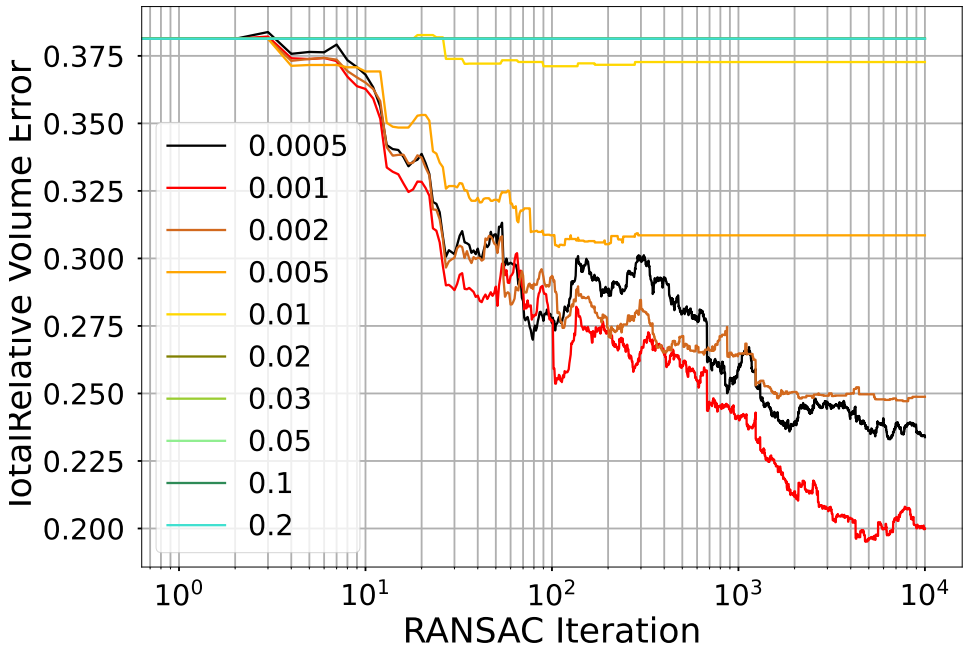


Figure 3.11 — Dependence of the Total Relative Volume Error (sum over all the ellipsoids) on the number of iterations of RANSAC (logarithmic scale) for real data for different threshold values. The error clearly declines with the increase of the number of iterations.

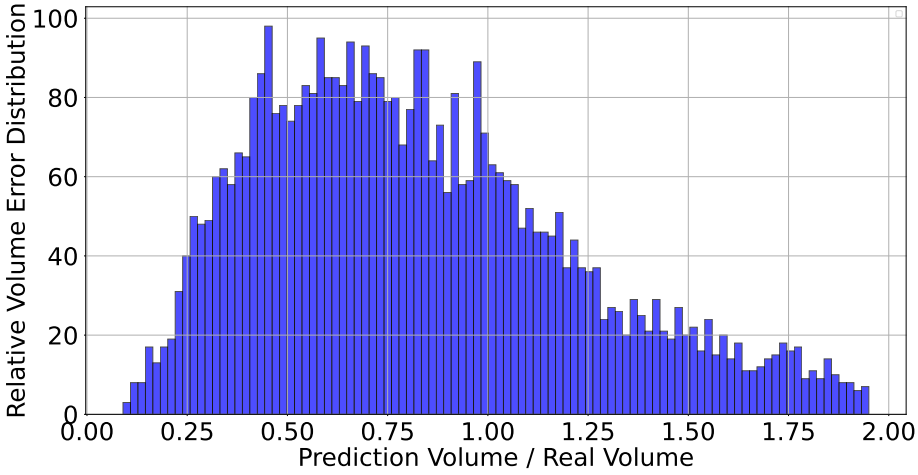


Figure 3.12 — The distribution of the relative volume error for all the tomatoes from the dataset. The mean value is 1.0020. No additional scaling was applied.

- The number of inliers for the model is correlated with its quality in terms of the output metric. This correlation becomes less prominent with the growth of the noise.
- On the challenging data the number of iterations plays crucial role in the quality improvement, see Figure 3.9 and 3.5 for results on synthetic data.
- IoU cannot be used as a single criteria for quality assessment, see Figure 3.6 and Figure 3.5 for a comparison of the plots for different noise levels.
- The quality of the output model declines with the growth of the distance from the sensor to the object. For the specific type of the camera that was used the cutoff distance that bounds the zone of reliable performance is nearly 1.5 meters.
- It was thus shown, that for the real data, which is the most challenging, RANSAC gives a high variance in the volume, but very small average error, which makes it applicable to the problem of yield estimation in the agricultural setting, see Figure 3.11. The example of the algorithm's output is presented in the Figure 3.1.

A method of ellipsoid volume estimation by noisy point cloud data is proposed. It relies on the combination of RGB and depth data. Specifically, it consists of two parts: detecting the objects in the RGB image, that is registered to the point cloud from a depth camera, and fitting ellipsoids to the corresponding subsets of that point cloud. The proposed method was tested on synthetic data and real data in agricultural context. Its main feature is that it is suitable for the real-world environ-

ments. For industrial applications, such as modern tomato greenhouses, it means that the tomatoes are not supposed to be manually picked. In contrast to many modern approaches, the proposed one does not rely on the contrastive background or an expensive setup with multispectral cameras. For the real experiments Intel RealSense D435i camera was used. The method's performance was evaluated on both synthetic and real data in terms of the Intersection over Union (IoU) between the predicted ellipsoid and the real one, volume error and processing time. The dependence of the output quality on the parameters, such as number of RANSAC iterations and inlier threshold, is presented. The method is compared with three other modern algorithms, showing superior performance on the complex real-world data. It is lightweight enough to be used on an autonomous agricultural robot with a user-grade laptop on board.

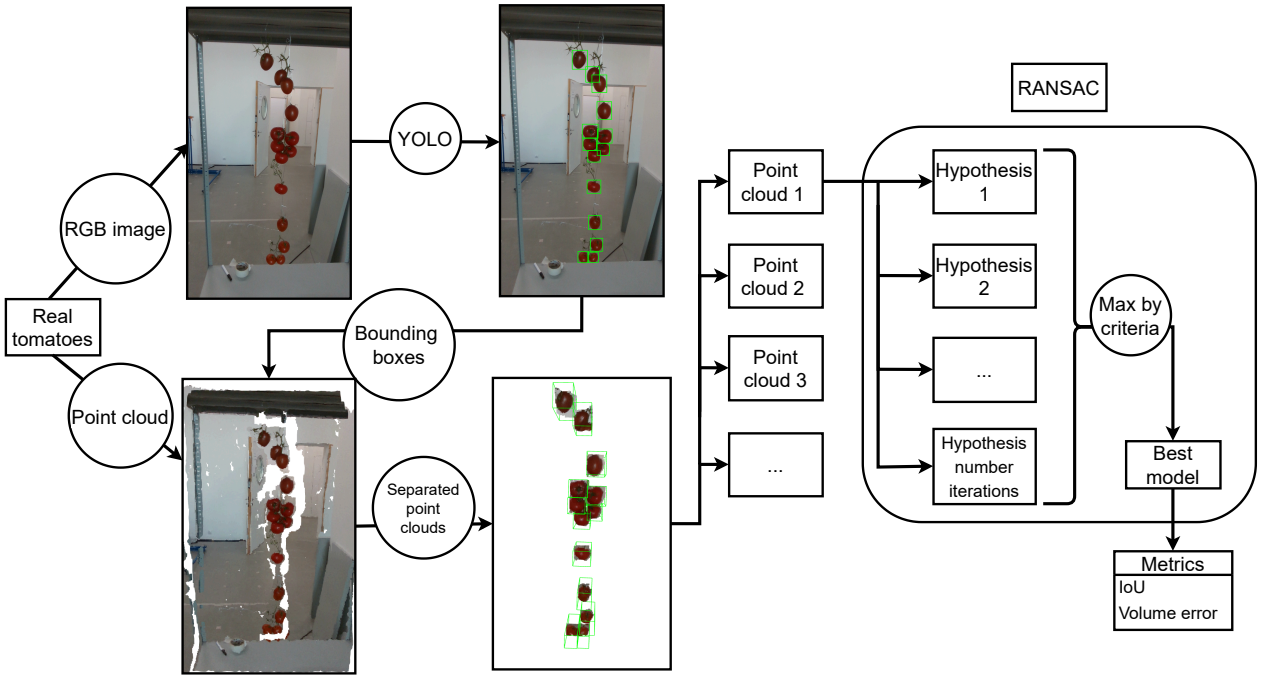


Figure 3.13 — Proposed algorithm. First, RGB and point cloud data are obtained with distinct, but registered sensors. Second, NN-based detection is performed and point clouds corresponding to individual objects are extracted. Finally, RANSAC algorithm is used to fit the proper ellipsoid models to them, consisting of repetitive subsampling and model evaluation.

Figure 3.14 — Tomato plants viewed from the side. The image was taken using the robot's onboard camera.

3.2 Combining YOLO with RANSAC for quadric surfaces

The tomato position, orientation and volume estimation consists of the following steps. First, the tomato detection via YOLOv8n is performed. Second, the corresponding point clouds are cropped from the original point cloud. It is worth noting, that in order to do that properly, the RGB and depth sensors have to be precisely mutually calibrated in order to exclude any offset during the cropping process.

Finally, RANSAC for quadric surfaces is applied in order to extract the parametric description of the tomato as an ellipsoid. The detailed scheme of the proposed method is given in the Figure 3.13.

Figure 3.15 — Point cloud with two tomatoes. It could be noted that the surface appears noisy and uneven.

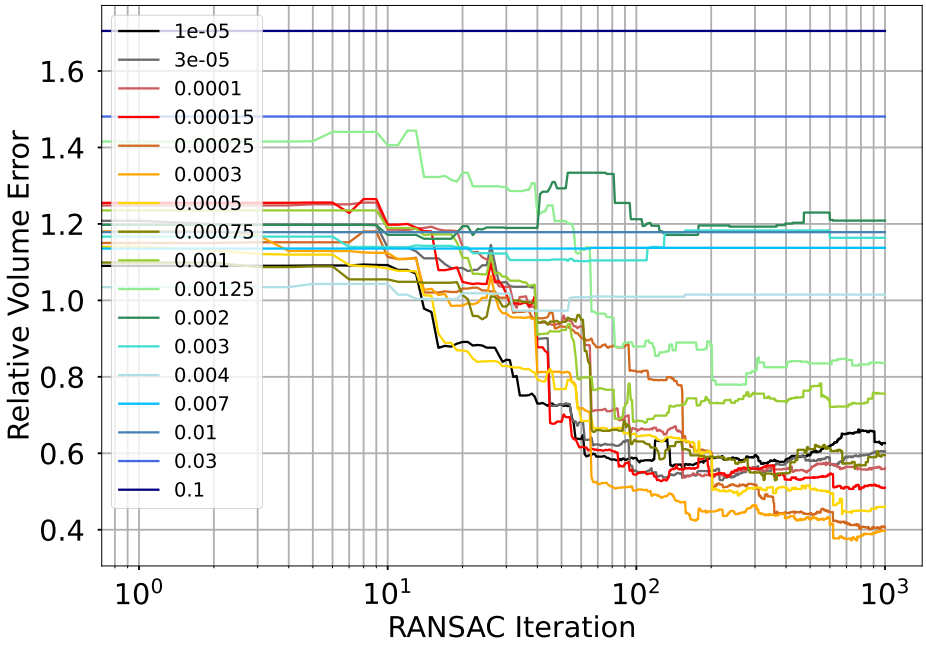


Figure 3.16 — Relative volume error for a single tomato for a variety of thresholds. It could be noted that the ones in the order of 0.001 produce the best results

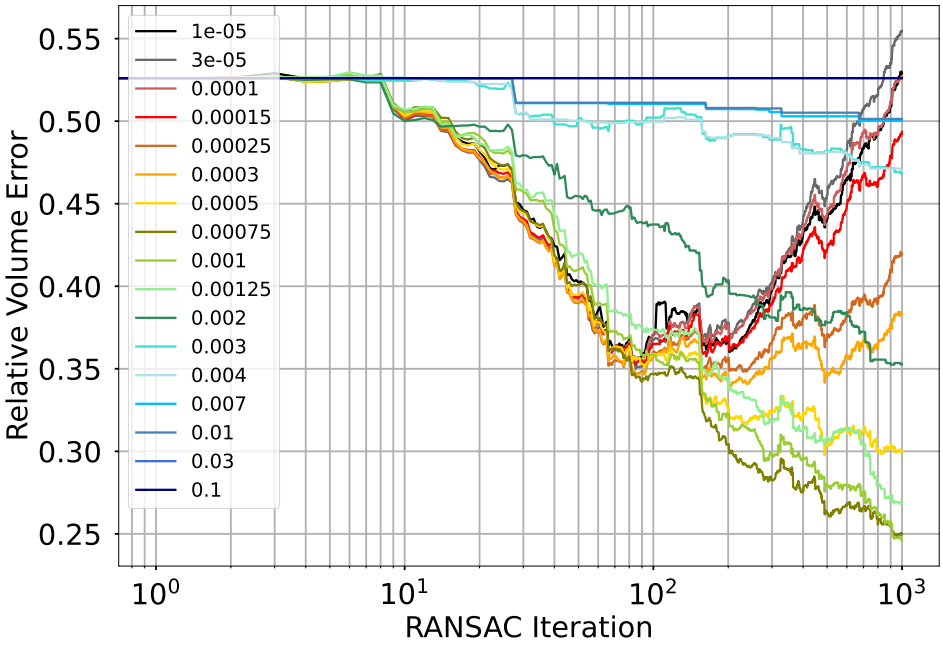


Figure 3.17 — Relative volume error for the whole real dataset by the number of RANSAC iterations. With certain thresholds the overall quality of the output increases as the number of iterations grows

3.2.1 Computer

Testing was carried out on the target device - the robot's onboard laptop. Its characteristics are the following: Intel Core i5-11400H 2.70 GHz, GeForce RTX 3050Ti laptop, 16 Gb of RAM. Machine operated under Ubuntu 20.04.

Iterations	IoU	Volume Error
10	0.2125	0.5084
100	0.2145	0.3587
1000	0.2164	0.2459

Table 5 — IoU and Volume Error for different number of RANSAC iterations on real data

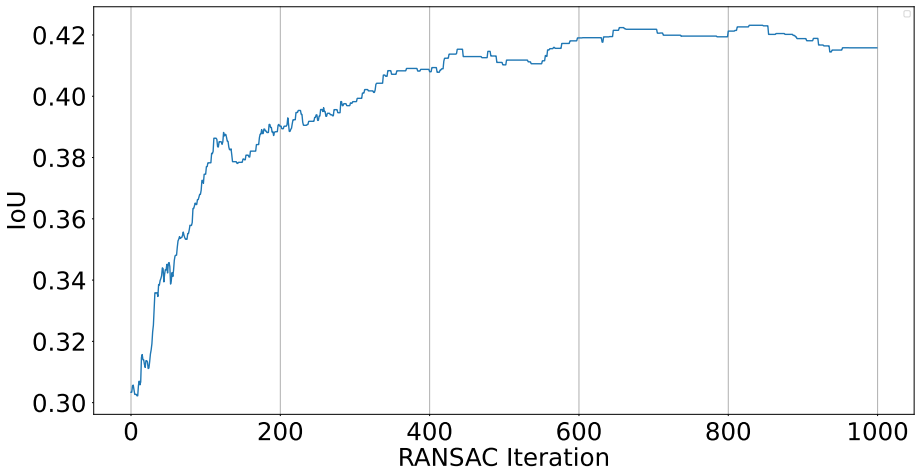


Figure 3.18 — IoU by the iterations number for a single tomato with the threshold of 0.001, averaged over all the point clouds with this tomato. It could be observed that the quality in terms of IoU reaches its limit faster than in terms of the volume error.

3.2.2 Experiments and results

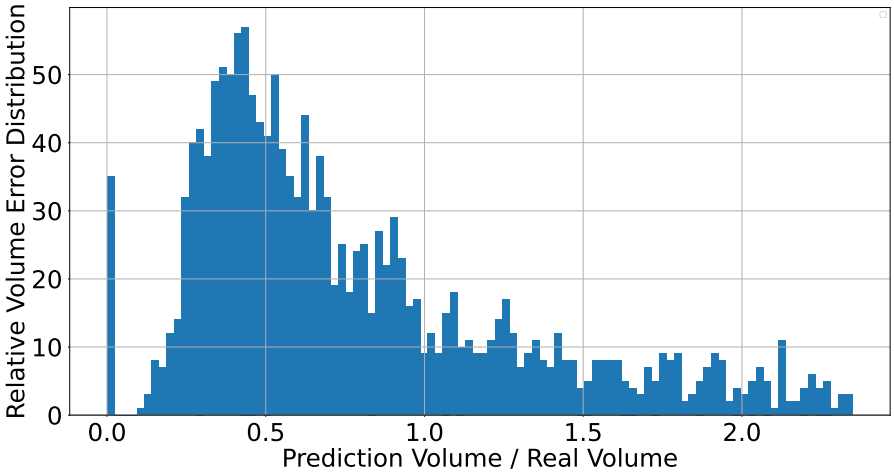


Figure 3.19 — Distribution of the predicted volume divided by the real volume for all the tomatoes in the real data. Despite the variance being considerable, the mean value is nearly 1.05, which means that the main metric that the potential user will be interested in, which is the volume estimation precision, is high.

	Clean Models			Normal Noise		
	IoU	Volume Error	Time	IoU	Volume Error	Time
Bayfit	0.0078	0.1164	1.5849	0.0104	2.88e+25	0.3709
CAS	1.0	0.0	0.0037	0.7202	0.164	0.0174
Matlab Ellipsoid Fit	1.0	0.0	0.0003	0.8024	0.0925	0.0003
RANSAC (ours)	1.0	0.0	0.0297	0.5101	1.17	0.0297

Table 6 — The comparison of the algorithms on clean and noisy full ellipsoids

	Segment Clean Models			Segment Normal Noise		
	IoU	Volume Error	Time	IoU	Volume Error	Time
Bayfit	0.0005	0.3985	1.1149	N/A	N/A	N/A
CAS	0.9899	0.01	0.0027	0.2054	4.0796	0.0029
Matlab Ellipsoid Fit	0.8447	20.8465	0.0006	0.1387	0.426	0.0005
RANSAC (ours)	0.9697	0.0192	0.0287	0.1985	2.1242	0.0298

Table 7 — The comparison of the algorithms on Segment Clean data and Segment Normal Noise. N/A means that the algorithm did not provide a result

	Real Data		
	IoU	Volume Error	Time
Bayfit	0.0185	0.9629	0.0657
CAS	0.1501	18.3553	0.0037
Matlab Ellipsoid Fit	0.0003	7.1066	0.0010
RANSAC (ours)	**0.2237**	**0.7617**	0.0254

Table 8 — Real Data Results and Iterations

	Paper	Year	Lang	Clean	Noisy	Clean Segm	Noisy Segm	Real Data
Bayfit	link	2024	Matlab	Yes	Yes	Yes	No	Yes
EllipsoidFit	-	2018	Matlab/Python	Yes	Yes	Yes	Yes	Yes
CAS	link	2022	Matlab	Yes	Yes	Yes	Yes	Yes
Detection + RANSAC(ours)	-	2024	Python	Yes	Yes	Yes	Yes	Yes

Table 9 — Summary of the methods. Yes and No for different data types denote if the method is working with them or not

First, let us observe the distribution of IoU versus the inlier number, see Figure 2.3. The trend here is considerably weaker than in the synthetic experiments. It could be connected with the distortions in the data that are not modeled properly by normal noise only. Each curve in the plot represents a single threshold. There is a variety of values that were chosen basing on the physical properties of the Intel RealSense camera.

Second, let us observe the distribution of the volume error versus the inlier number, see Figure 2.4. It could be noted that the volume error tends to decrease as the inlier number is increasing.

Figure 3.16 demonstrates the change in the relative volume error as the RANSAC iterations number grows. For certain threshold values the error is driven to nearly 0.4 at 1000 iterations.

Figure 3.17 presents the same data, but for all the real tomatoes. One of the thresholds drives the error down to nearly 0.25 with 1000 iterations.

It could be noted that IoU converges to its final value much faster, than the total volume error is decaying. This discrepancy could be interpreted as follows. While the IoU growth is inevitably limited by the quality of the manual markup, volume error on the other hand is decreasing as the models for the tomatoes are being adjusted.

The metrics for different number of iterations for the best threshold are presented in the table 5.

Finally, let us compare the performance of the proposed method with the other algorithms, see Tables 6, 7, 8 and 9.

On some of the synthetic data the proposed method performs worse than the other methods, see Normal Noise IoU, where Matlab Ellipsoid Fit gives higher IoU.

However, on the most challenging sort of data, which is the real ellipsoids, the proposed method shows superior performance in terms of both IoU and volume error. Moreover, it could be noted that the average volume for a group of tomatoes closely matches real values, despite high variance in individual outputs 3.19.

This work is focused on the development of a method of tangerine volume estimation based on the point cloud data. While monocular vision-based methods are often easier to implement and deploy in real world both in terms of algorithms and the sensors required, depth data is necessary to obtain precise volume estimate. In this work tangerines are approximated by ellipsoids, that are fitted to the point clouds in 3D. A dataset of real tangerines was collected and marked. For each tangerine the mass and volume were measured. RANdom SAmple Consensus (RANSAC)

was used for the ellipsoid identification. A number of experiments were conducted with varying algorithm hyperparameters, including iteration number and the inlier threshold value. The output quality was assessed via comparing the prediction of the method with the real volume of the fruit. Experimental results suggest that the good enough performance could be achieved in real time with a portable computer.

It was shown that good enough performance could be achieved with the means of a user-grade active stereo camera and a modern laptop.

3.2.3 Algorithm Description

The input of the method is a number of three-dimensional points. The output of the algorithm is the volume of the tangerine.

Before the algorithm is applied, the data is collected and processed. The full volume estimation pipeline is as follows.

- An RGB image and a point cloud are taken with an Intel RealSense D453i camera.
- The tangerines are detected and segmented on the RGB image by the means of the color-based filtering.
- For each of the tangerines a corresponding point cloud is cropped.
- The obtained points are fed into the ellipsoid recognition algorithm.
- The volume of the tangerine is evaluated using the semiaxes of the ellipsoid.

3.2.4 Results

The experiment results are presented below. Since the main quality metric is precision of volume estimation, the average volume error was evaluated.

Fig. 3.20 presents the values of the total volume error that was measured for all the tangerines, meaning that the total volume was compared with the ground true total volume, depending on the number or RANSAC iterations. A number of thresholds was considered. If the threshold value is too small, the output quality degrades as the number is the iterations grow, since the algorithm converges to

fitting noisy artifacts present in the data. With more reasonable threshold values the average volume error drops to nearly 0.1 over almost 200 iterations, making it possible to rapidly evaluate the volume of the tangerines.

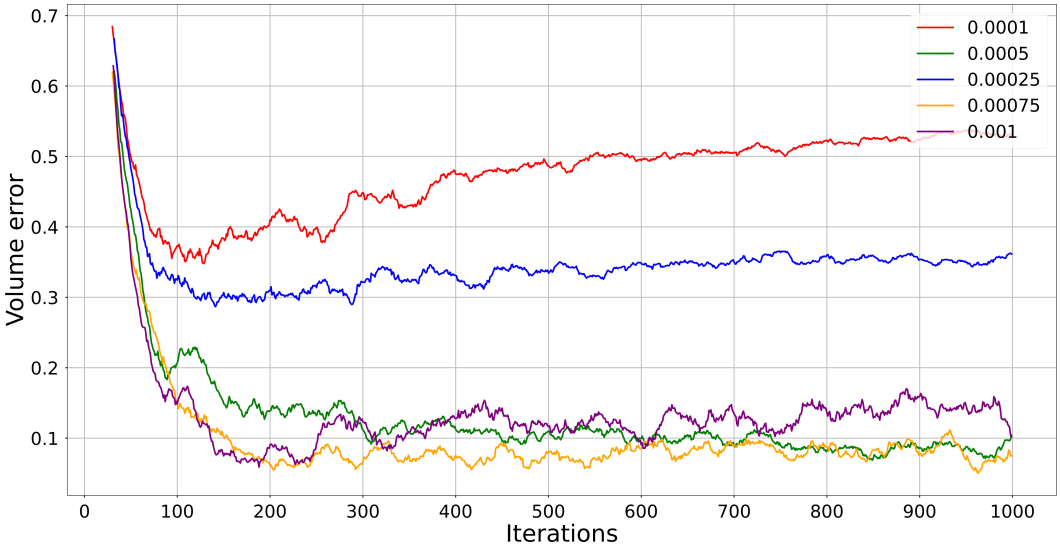


Figure 3.20 — The dependence of the total volume error on the number of RANSAC algorithm iterations with different inlier threshold values. The best performance is achieved with the threshold values 0.001 and 0.00075.

Fig. 3.21 presents the values of the average volume error tangerine-wise, depending on the number or RANSAC iterations.

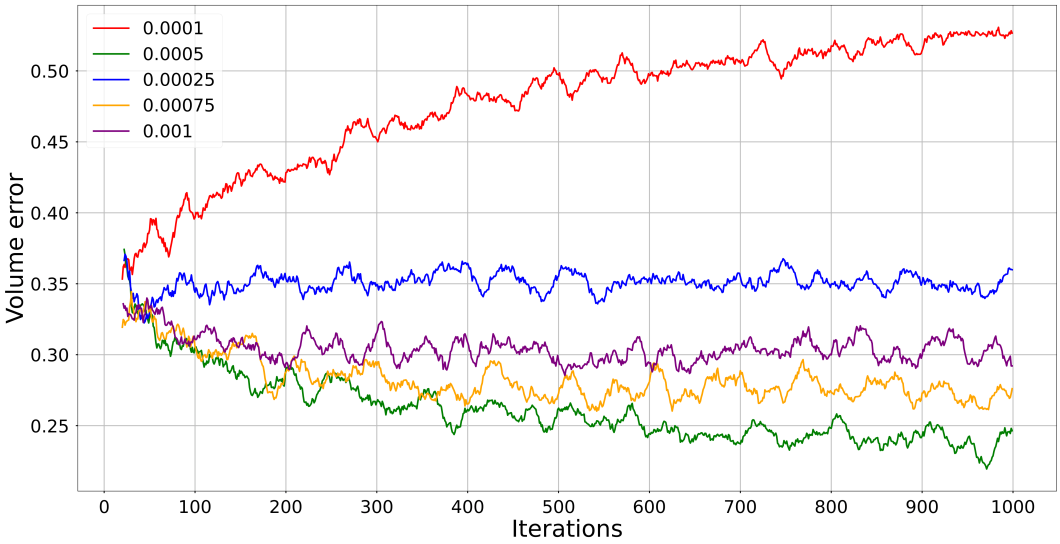


Figure 3.21 — The dependence of the average volume error on the number of RANSAC algorithm iterations with different inlier threshold values.

The numerical results across different iteration numbers and threshold values are presented in the tables 10 for total error and 11 for average error.

Table 10 — Total Volume Error for different iterations number across a number of thresholds.

Iterations	**Threshold**				
	0.0001	**0.0005**	**0.00025**	**0.00075**	**0.001**
10	0.3543	0.4482	0.5301	0.3227	0.4382
100	0.4136	0.2908	0.2942	0.2860	0.2968
1000	0.5078	0.2371	0.3265	0.2967	0.3328

Table 11 — Average Volume Error for different iteration number across a number of thresholds.

Iterations	**Threshold**				
	0.0001	**0.0005**	**0.00025**	**0.00075**	**0.001**
10	0.7013	0.8693	0.9357	0.6617	0.8053
100	0.4009	0.2301	0.3252	0.0939	0.1649
1000	0.5169	0.1644	0.3262	0.1688	0.1468

3.3 Neural Network-based Volume Estimation

The contributions of this paper are as follows.

- A dataset of point clouds with tangerines was collected in the environment mimicking the conveyor belt.
- Dataset markup was performed, meaning the measurement of the volume of all the tangerines.
- Numerical experiments were conducted with a PointNet++-like neural network on ellipsoids.
- The results of the experiments were evaluated in terms of the volume error.

Fig. 6.3 presents a point cloud from the dataset, captured by Intel RealSense D435i.

3.3.1 Algorithm Description

The input of the method is a number of three-dimensional points. The output of the algorithm is the volume of the tangerine. Before the algorithm is applied, the data is collected and processed. The full volume estimation pipeline is as follows.

- An RGB image and a point cloud are taken with an Intel RealSense D453i camera.
- The tangerines are detected and segmented on the RGB image with the color-based filtering.
- For each of the tangerines a corresponding point cloud is cropped.
- The obtained points are fed into the neural network.

3.3.2 Neural Network Architecture and Training

The neural network encompasses 393025 parameters. All the point clouds in the dataset containing 30 tangerines were split into 7980 for training and 3990 for test.

Fig. 3.22 presents the architecture of the neural network, closely following PointNet++ [42]. It consists of the following parts. First, a number of spatial processing sections are appiled. The spatial processing section is a crucial part of this architecture. It subsamples a number of points from the neighborhood of each point, conserving local distibution of the points. After that, the points are grouped and a vector representation for them is obtained via multilayer perceptron, followed by max pooling. After three spatial processing sections global max pooling is performed. Finally, a series of fully connected layers are used.

3.3.3 Results

The training took 89.5 minutes on a computer with processor Intel Core i5-13400 (13th Gen, 10C/16T, 2.5 GHz), 32 Gb of RAM DDR5 and graphics card

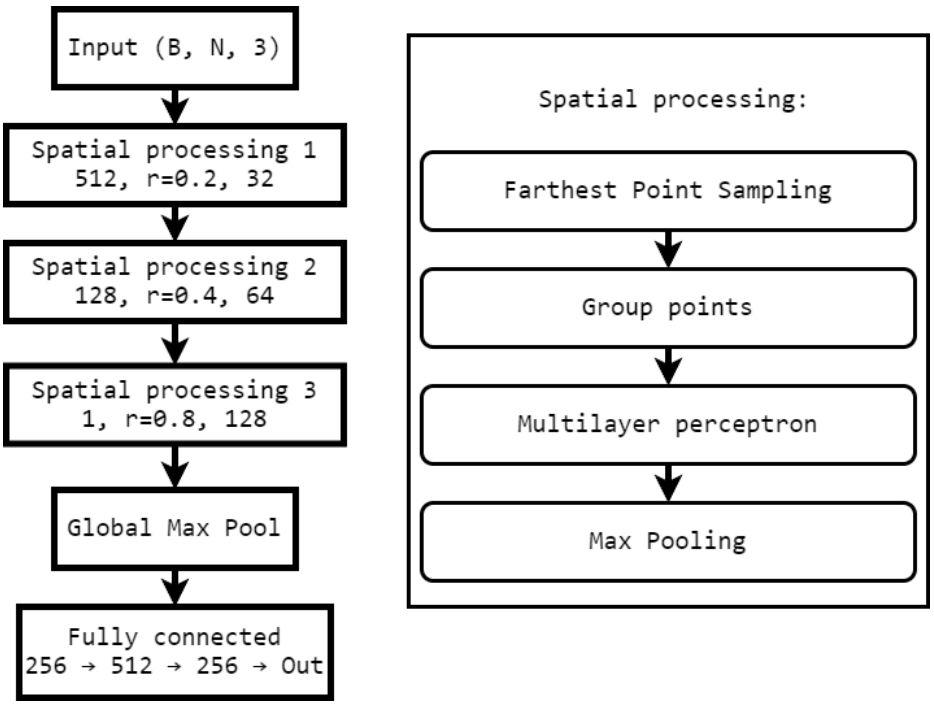


Figure 3.22 — The architecture of the neural network used in the work.

NVIDIA GeForce RTX 4070 (12 GB GDDR6X). The inference time on a single point cloud is 290 milliseconds.

The results are presented below. Since the main quality metric is precision of volume estimation, the average volume error was evaluated.

Fig 3.23 presents the values of the loss function as the training progresses. It could be noted that after epoch 30 the decline stagnated, meaning that the generalization capabilities of the model and the quality of the data do not allow for the further improvements. Fig. 3.24 presents the dependence of the average volume of the number of epochs of training.

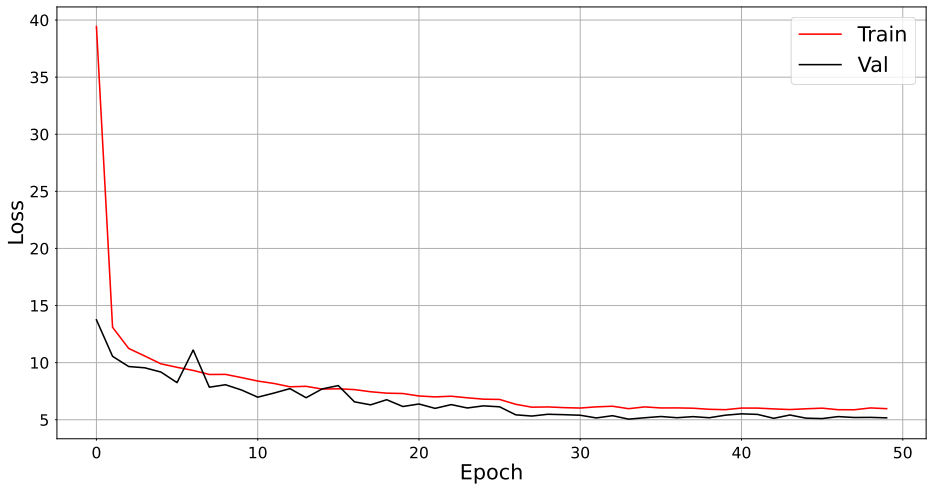


Figure 3.23 — Loss function during the training.

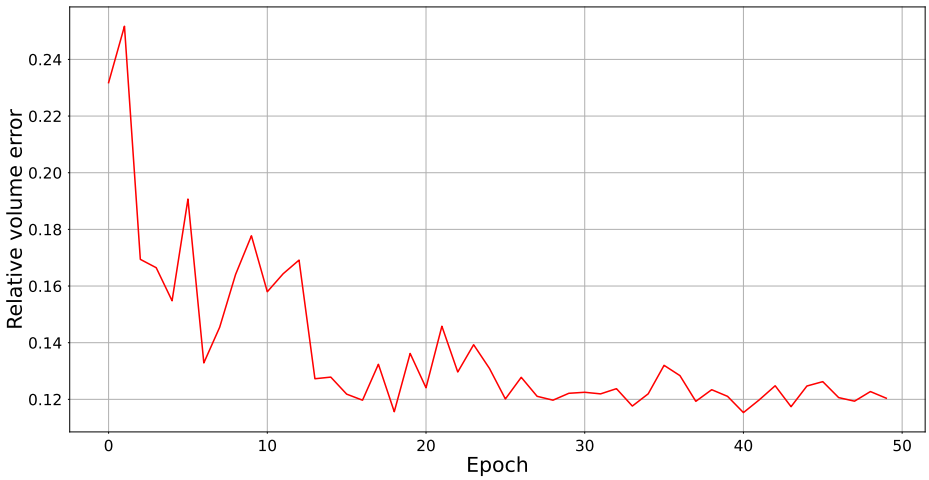


Figure 3.24 — Relative Volume Error during the training.

The main result of the presented work is the following. After the training the Mean Relative Volume Error on the real point cloud data with tangerines from the test set reached 0.1070. Overall, the performance meets the demands of the market in terms of the quality of volume estimation and inference time on a modern laptop. The sensor used is a user-grade active stereo camera.

Chapter 4. Disease detection

Modern agriculture poses several challenges in terms of the requirements for monitoring speed and precision. Certain greenhouses of the major agricultural companies are so huge, that it is virtually impossible to timely examine the whole facility with a reasonable human workforce. While it is possible to introduce more and more human workers, a sustainable long-term solution is to use autonomous robots to automate such repetitive tasks.

While robot-assisted agriculture has been developing for a long time already, the integration of a robot into an industrial-scale production often demands substantial effort. It requires the robot to have a high degree of autonomy while keeping the modifications that the greenhouse facility should go through to the minimum.

4.0.1 Target environment

The target greenhouses for this project are rectangular 120000 m^2 (12 hectares) buildings with glass rooftops. During the night and in cloudy weather red-blue LEDs are used to provide sufficient lighting. The temperature and humidity are kept stable at all times. Several types of tomatoes are grown there with the help of hydroponics. One tomato plant grows up to 35 meters in length during 10 months of cultivation. The majority of the length of the plant is kept horizontal, and the top of the plant is at a height of approximately 4 meters.

4.0.2 Powdery mildew

The density of the biomass (see Figure 5.1) in these greenhouses allows for cost savings because of the scale of the production. On the other hand, it leads to the rapid spread of various kinds of bacterial infections, fungi, viruses, and insects. Thus, timely monitoring in such greenhouses is crucial. Powdery mildew is known to infect the lower leaves first and then spread up the plant. It develops best at temperatures

slightly below 30°C and high humidity, which are precisely the conditions that the tomatoes are grown in for the great portion of the production cycle. Thus, the development of the powdery mildew is rapid, taking 4 to 7 days to progress from early to medium stage [48].

With the aforementioned requirements for monitoring speed in mind, an autonomous robot on an omnidirectional platform was developed and manufactured. It is equipped with cameras, covering the whole height of the tomato plant, and an onboard computer to process the data in real-time.

This paper presents a work on the data acquisition with this robot, as well as coordinating the markup procedure by human experts and a comparison of the performance of several neural networks.

4.0.3 Datasets available and related work

The majority of the open datasets are either small or taken in laboratory conditions, meaning professional cameras, artificial lighting, separation of the leaves from the plant, or manual preprocessing. By far the most popular dataset in the field [49] consists of images of leaves on a contrastive background. Collecting such a dataset is expensive, and its usage is limited by the differences between the laboratory and real conditions.

While there are several publicly available datasets out there, relying on them could be risky due to the covariance shift [50]. The main difficulties for obtaining qualitative and consistent markup are[51]:

- Symptom variations.
- The potential presence of different disorders with similar symptoms.
- Ill-defined edge cases, leading to the markup ambiguity.

Taking into account all the aforementioned factors, a dataset for classification, consisting of images from the target greenhouse in normal conditions was collected and marked, see section 6.2. It is worth noting that accuracy is by far the most used evaluation metric in the field of automatic plant disease detection [52].

4.0.4 Related work

There are multiple solutions to the same problem that were stated in different environments. They can rely on hyperspectral imaging [53], [54], [55], [56], on recurrent NNs [57], a custom convolutional architecture [58]. Hyperspectral cameras are usually expensive, which limits their availability. In this work, relatively cheap user-grade RGB cameras were used.

There are several works that are aimed at comparing the performance of already widely adopted models on specific tasks, such as [59], [60], and [61]. The aim of this work is to produce a solution that fits the specific needs of the environment from an engineering standpoint, see section 5.2.

4.0.5 Contribution

The contributions in the field of disease detection are as follows:

- Roboticised data acquisition was performed. Namely, a robot was designed and manufactured, and a dataset of powdery mildew-infected tomato leaves was gathered. The data collection was performed in a real environment under varying lighting conditions.
- Marking of the 6817 images into positive and negative classes was coordinated. The markup was performed by human experts with formal education in agriculture. The consistency of the experts was assessed by providing the same samples to different experts.
- Using this data, an experiment was conducted: YOLOv8 (n, s, m), EfficientNet (V2l, B3, B4), ResNet (34 and 152), MobileNetV3 (small and large) were compared in terms of their accuracy and the inference time on the target computer.
- It was thus demonstrated that good enough performance could be achieved with the means of user-grade cameras and images that were taken from the moving robot. The mutual expert consistencies were matched by the NN model in terms of the classification accuracy, see section 4.2.

The chapter is structured as follows. First, the mechanical platform is briefly described and an overview of the solution to the disease detection problem is given. Second, the data acquisition and markup procedures are described. Third, the training in terms of the models, parameters, and experiments is described. Finally, a comparison of the performance of different NN models is given.

4.1 Training

4.1.1 Hardware

A two-tier computing strategy for training and testing the neural networks was used. The training was performed on a stationary server: Intel Core i9-9900 3.10 GHz, GeForce RTX 3080, 32 Gb. Testing was carried out on the target device - the robot's onboard computer. Its performance is much more limited: Intel Core i5-11400H 2.70 GHz, GeForce RTX 3050Ti laptop, 16 Gb. Both machines operated under Ubuntu 20.04.

4.1.2 Models

A comparison of 10 top-performing classification models provided by Roboflow [62] was carried out. These models encompass:

1. Three versions of YOLOv8[63]: YOLOv8n, YOLOv8s, and YOLOv8m.
2. Three versions of EfficientNet[64]: EfficientNetV2l, EfficientNetB3 and EfficientNetB4.
3. Two versions of ResNet[65]: ResNet34 and ResNet152.
4. Two versions of MobileNet[66]: MobileNetV3small and MobileNetV3large.

4.1.3 Data sampling

Three distinct datasets were created, each addressing the class imbalance challenge within the original dataset. Initially, the data comprised 5684 instances of the negative class and 1133 instances of the positive class. To tackle the imbalance, three different strategies were pursued:

- For the first set 1133 images of the negative class were randomly selected, equalizing the number of photos in both classes.
- For the second set all the photos from the positive class were duplicated to increase the number of original images of the negative class. The number of training photos per class grew from 850 to 1700.
- For the third set, all the labeled data available was used. To achieve this, extensive and diverse augmentation techniques were applied. The positive class images were duplicated six times with augmentation, and similar augmentation transforms were applied to $\frac{6}{7}$ of the negative class images. This training set consists of 5361 instances of the positive class and 5316 instances of the negative class.

In all the experiments 75-15-10 % train-val-test split was used.

4.1.4 Data Preprocessing

Resizing: All the images underwent cropping to a square shape and a resizing process with the resultant dimensions of 1056 $\times$ 1056 pixels. The rationale behind adopting this specific size is the following. First, some of NN models that were planned to be used could work only with the image sizes that are multiples of 32. Second, the downscaling of the images was not feasible, because powdery mildew appears with quite small features, and reducing the image size would entail a substantial loss of critical information. Third, resizing rectangular images into a square will again lead to the loss of data.

Augmentation: Across all three datasets, a standardized suite of transformations was applied. These transformations encompassed horizontal and vertical flipping, shifting, rotation, and HSV color transformation. However, it is worth not-

ing that the third dataset underwent a more expansive augmentation strategy. For this subset, the following augmentations were applied:

- *Predominantly:* Defocus, stochastic mist, noise injection, RandomBrightnessContrast, and HSV color transformation featuring augmented values.
- *To a smaller extent:* ColorJitter, rgbshift, ChannelShuffle, ChannelDropout, image inversion, and sepia.

The selection of these augmentation techniques aimed to diversify the dataset as much as possible to enhance the model's generalization capabilities.

4.1.5 Experimental setup

The setup in terms of the training is the following:

- Training the majority of the models for 40 epochs before choosing the best checkpoint. The exact number for each model is given in the table 12.
- SGD optimizer was used.
- The default values for the learning rate of 0.01 for YOLO models and 0.001 for other models were used.

The choice on the number of epochs was made following preliminary training runs with the learning curves stagnating after such a number of epochs. The batch size for all models was determined out of the memory constraints.

4.2 Results

Model	Params	Accuracy [%]	Server inference [ms]	Onboard PC inference [ms]	Training time	Epochs
Resnet34	21.8M	84.40	34.7	38.23	169m 40s	40
Resnet152	60.2M	85.40	34.00	-	96m 58s	40
YOLOv8n	3.2M	**85.80**	5.9	**6.6**	25m 54s	30
YOLOv8s	11.2M	85.00	10.6	12.5	27m 37s	30
YOLOv8m	25.9M	84.50	30.7	33.0	29m 40s	30
EfficientNetV2l	118.5M	84.07	49.3	-	124m 19s	40
EfficientNetB3	12.2M	80.53	14.86	-	66m 45s	40
EfficientNetB4	19.3M	79.20	19.7	-	74m 29s	40
MobilenetV3s	2.5M	83.63	4.15	6.64	54m 23s	40
MobilenetV3	5.5M	84.51	4.81	19.21	52m 23s	40

Table 12 — Performance of all models on the first dataset.

Model	Accuracy	Inference server (ms)	Inference (ms)	Training time	Epochs	Batch
YOLOv8n	85.4	5.9	6.6	83m 52s	40	16
MobilenetV3s	84.07	4.2	6.61	182m 59s	40	32
MobilenetV3l	82.74	4.87	19.41	357m 19s	40	32

Table 13 — Performance of selected models on the second dataset.

Model	Accuracy	Inference server (ms)	Inference (ms)	Training time	Epochs	Batch
YOLOv8n	77.5	3.3	7.3	472m 32s	40	16
MobilenetV3s	77.45	4.08	6.65	293m 1s	40	16
MobilenetV3l	77.45	4.95	19.16	321m 32s	40	16

Table 14 — Performance of selected models on the third dataset.

Overall, a dataset of powdery mildew-infected leaves was collected with the user-grade RGB cameras. A number of neural networks were trained with the best of them performing on par with the human experts. The inference time on the onboard computer is well below the maximal acceptable time, thus leaving room for the introduction of other models or increasing the number of cameras.

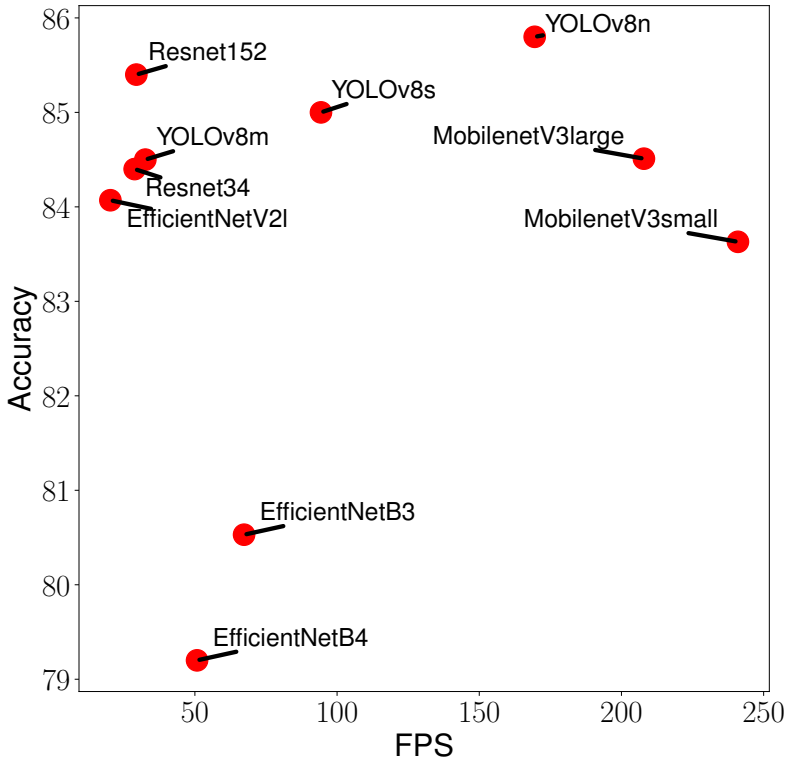


Figure 4.1 — Performance and FPS of the models trained on the first dataset with inference on the server.

Table 12 presents the results of the training of all the models considered on the first version of the dataset, see Section 4.1.3. It was not possible to run certain models on the onboard laptop, taking into account the target resolution of 1056 × 1056. These models are marked with a dash sign in the *Onboard PC inference* column. These models were excluded from further experiments because they were too demanding in terms of computational resources.

All the models were examined from the standpoint of the combination of accuracy and inference time, see Figure 4.1 for the results on the server. It turns out, that three of them are Pareto-optimal, meaning that no other model is faster and more precise at the same time. Those are YOLOv8n, MobilenetV3small and MobilenetV3large. Figure 4.2 gives their results on the target machine.

Another series of experiments was carried out with three aforementioned models and the second version of the dataset, see Table 13. The results did not subvert

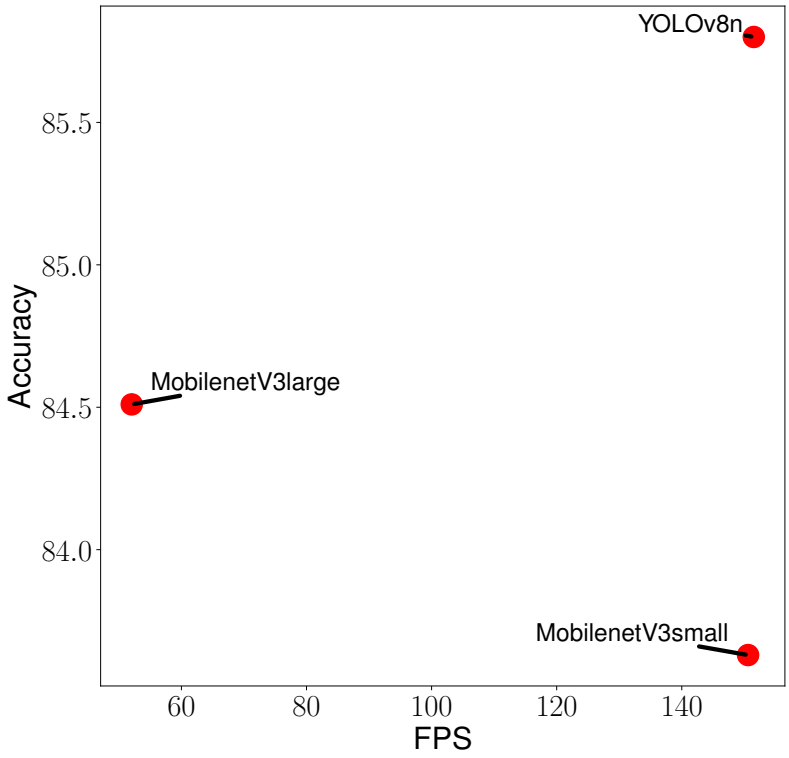


Figure 4.2 — Performance and FPS of the chosen best models trained on the first dataset with inference on the onboard laptop.

the expectations with accuracy slightly increasing for MobilenetV3small and slightly dropping for other models.

With the third version of the dataset the accuracy values have become significantly worse, see Table 14. The values of the metrics for different models are relatively close to each other, matching the consistency of the experts in terms of accuracy. Thus, the reasonable qualitative explanation could be that the performance is limited more or less by the data quality, not the models' generalization capabilities. The best performance by both accuracy and time was shown by YOLOv8n neural network. The False Negative rate for it is 6 %.

In order to assess the model performance on the high-quality data, a set of 100 images was labeled by a consensus of 4 experts. The consensus here means that the image-wise labels were agreed upon by all the experts. The output of the model matched the expert labeling for 96 samples of 100.

The overall performance in terms of speed is well above the minimal requirements, meaning that the monitoring speed is limited by the robot's motion (because of safety concerns), not by the inference time of the neural networks.

It is worth noting that this excessive performance could be beneficial for solving other related problems. For instance, to detect certain tiny insects it will be

necessary to install much more cameras with small FoV. Their number could rise to dozens, requiring very low inference time for a single image. These cases could be addressed without any substantial modification to the vision pipeline.

Chapter 5. Robot

Let us describe the main design decisions behind the proposed autonomous platform. First of all, the target environment should be considered. It is a greenhouse 300 by 400 m where several hundred lines of tomatoes are growing. In the middle of the greenhouse there is a concrete road elevated above the main floor. The tomatoes are growing in lines perpendicular to this center alley. In order for the robot to be truly autonomous it is supposed to have two modes of locomotion, particularly on the concrete floor and on the rails. While the movement on the rails can be implemented in a straightforward manner, the design decision for the concrete is not as simple.

On the one hand, the number of actuators necessary should be kept at minimum. Moreover, any kind of complicated rail system will be prone to the elements from the surrounding. The environment in the greenhouse is humid, and there could be grains of sand, dust, or even small puddles of water. On the other hand, the width of the concrete road is approximately 3 m. It allows the personnel to walk and manually move the machinery. And the control algorithms in combination with the mechanical platform have to be capable of executing sharp turns and obstacle avoidance maneuvers.

After rigorous consideration and literature review Mecanum wheels-based system was designed. It is much more mechanically complicated than the regular car-like platform with only the front wheels steering, but it allows for the omnidirectional motion, as well as turning on the spot. Four wheels were placed at the corners of the robot. An important factor is this smoothness, or the uniformity of the concrete surface that cannot be guaranteed in the unknown environment. Thus, an individual suspension module for each of the wheels was designed. Modern greenhouses are standardized, meaning that they have the rails at a specified distance from each other. So the geometrical configuration of the robot was derived from that. The first version of the robot was designed and assembled, which was followed by the real test in a greenhouse during the tomato production.

During the testing, it has become clear that despite the specifications, unknown and unpredictable circumstances can appear. One of the main ones was that despite the robot properly feeding in between the lines of the tomatoes, it was at times touching the tomato plant parts that were hanging lower than they should be. The integration of an agricultural robot should be performed as smooth as pos-

sible, meaning the non-invasive manner and keeping the greenhouse modifications to the bare minimum, as well as the workflow there. Thus, a decision was made to reduce the width of the robot by integration of the actuators inside the wheel itself. With such a modification the robot has become even more compact, minimizing the probability of the contact between it and the plant.

The road wheels and the rail wheels are mounted coaxially, keeping the number of the actuators necessary down to only four. The motors that are used in the last (third) version of the robot are normally used in electric bikes. They are protected from the humidity and dust and are capable of functioning in a wide range of external conditions, including high temperature. Moreover, with the current mass of the robot, the perspective velocity and the limited acceleration, they are used under very moderate load, assuring their longevity and reducing the need for maintenance. During the design of the last version of the wheel modules, a model of mechanical wheels that was found convenient, was reproduced using more rigid type of steel. The wheels from the second version of the robot broke during the testing, making it necessary to develop a more sturdy version. All in all, the last version of the wheel modules can be used in a number of agricultural applications. They could be mounted to any object or machine with a few modifications, meaning only the mounting surfaces and the necessary wiring. The control boards of the wheel motors were chosen so that they can carry much heavier loads in terms of the current then the ones that typically appear in the robot.

Regarding the main hull of the robot, it is manufactured from the aluminum tubes with square cross-section and planar aluminum elements. In the first version of the robot composite materials were used to manufacture the planar elements. Later they were changed to aluminum. It is heavier, but it provides rigidity to the structure of the robot. Moreover, it lifts the burden of active cooling of the interior of the robot by having a surface area of approximately 2 m^2.

Regarding the electrical supply of the robot, lithium batteries were chosen with high energy density, and the voltage that the motors can work at without a converter. Six batteries approximately 5 kg each were placed in the exterior module of the robot. The internal space can be used to accommodate the computer and other equipment. The perspective power consumption of 500 W can be sustained for 12 hours. In order to assure smooth integration of all the equipment to the robot a DC-AC converter was installed on board providing 220 V for all the devices, allowing them to be powered by their default power units. While this solution

is excessive in terms of the power consumption, it makes the integration of novel modules easier. The devices on board include the main computing module, which is a laptop with a discrete GPU, a Wi-Fi router, a set of cameras, a power converter and the charger. The charger was chosen such that it can charge all the batteries in three hours and can be controlled with RS-485 protocol. This makes it possible not only to perform the monitoring autonomously, but also charge autonomously. Wi-Fi router is installed on board in order for the robot to be capable of providing its own wireless network for the convenience of the user. While the greenhouses are often highly automated, they are also often located in remote places that lack sufficient coverage of the Internet providers. Moreover, it could be expensive to provide a wireless network of low latency and high bandwidth for all the greenhouse. This was one of the reasons why all the computations are performed on board. The second reason is that with onboard computations the result could be obtained instantly. Alternatively, an external server could be installed, but such a solution will require a high bandwidth wireless network, capable of streaming up to six FullHD videos simultaneously. With the current developments in the field of mobile computing devices, it is possible to perform the inference of modern neural networks and other algorithms to process data on the board in real time. With the speed of the robot that is acceptable from the safety standpoint and the computational capabilities of the discrete GPU the problems of disease identification and tomato volume estimation can be solved in real time as the robot drives along the line of tomatoes. The software on robot is pretty standard in terms of the framework and tools. Ubuntu 20 was used, as well as OpenCV, PyTorch, Open3D, Python, C++ and ROS2. The motor drivers were programmed in C.

5.1 Platform

With a prospective weight of the robot of around 100 kilograms, an aluminum frame made of a profile with a square cross-section was designed. The design of the chassis was performed with the intent to minimize the number of individually actuated elements. The result is an omnidirectional platform with 4 identical wheel pairs.

The power supply and consumption of the robot could be summarized as follows:

Figure 5.1 — Image taken in a greenhouse at the top of the plants at the height near 4 meters above ground level.

а) б)

Figure 5.2 — (a) Sketched Fields of View of the cameras. (b) Physical assembly during the data acquisition.

– 6 × 24 Ah 48 V LiNCA batteries
– 144 Ah (approximately 7 kWh)
– Nearly 14 hours of uptime on one charge with the power consumption of 500 W

One of the crucial decisions that had to be made was the positioning of the host computer for the NNs inference. The final design of the robot included a relatively powerful computer on board, see section 4.1. It would be reasonable to do this in another way – by dividing the system into a robot and an external server. This would bring the problem of providing coverage of the entire greenhouse with a wireless network capable of broadcasting a number of (at least) FullHD videos in real-time.

5.2 Overview of the solution

For the problem of tomato greenhouse monitoring the main requirements for the vision system are the following.

First, the combination of the Fields of View (FoV) of the cameras should cover the entire height of the plants, from the tomatoes to the tips of the vegetation. Diseases, parasites, and damage can occur anywhere on the plant. This requirement is addressed by installing a number of cameras, their proper positioning, and their parameters. Three web cameras were utilized. The sketch of this setup is given in Figure 5.2a, and the real assembly is given in Figure 5.2б. At this point, only one side is covered by combined FoVs, but it is planned to replicate that and cover the other side as well.

Second, the image processing should be performed in real-time. Ideally, the robot is expected to spend all the time in the field moving. With real-time processing, it is possible for the workers to rapidly examine specific rows of tomato plants. The robot should be capable of observing the same part of the plant from different locations, meaning that the data processing should happen in real-time with a certain overlap in the frames.

During the design of the vision subsystem, the following assumptions were adopted. In the initial testing, the robot was moving at a speed of 0.26 m/s, which is a safe value for indoor applications. Considering potential acceleration for the sake of faster monitoring, an upper-speed limit could be set to 0.5 m/s. The horizontal FoV for the chosen optics covers 1.07 m. Thus, moving an object through it from left to right takes nearly 2 s. To assure the 5-fold overlap, each of the cameras has to capture an image at least once every 0.4 s. That results in the requirement for the

processing speed of 8 frames of 1056 $\times$ 1056 pixels per second. This requirement is met with a huge margin.

Chapter 6. Data collection

6.1 Data

Three sets of data were used in the numerical experiments. The first are the synthetic ellipsoids generated with a naive straightforward method, see subsection 6.1.2. The second are the data generated with the ray emulation approach, see subsection 6.1.1. Finally, the third is the real data recorded with Intel RealSense D435i camera, see subsection 6.1.3. Let us describe these sets of data.

6.1.1 Naive synthetic ellipsoids generation

The first dataset contains points evenly distributed on the surface of ideal ellipsoids. The ellipsoid is described by its center, semiaxes, and rotation. After specifying those, a point cloud is created, consisting of points on the surface of the ellipsoid. These data were used as the simplest possible to evaluate the algorithm's performance. As one might expect, the error on clear data was precisely zero, so data with additive normal noise was considered as well, see Table 2 and Table 3.

The point clouds that are generated this way are different from the real ones due to the non-uniform surface coverage in the real data and only partial visibility to the real camera.

6.1.2 Ray emulation-based synthetic data generation

Thus, an alternative approach was taken with a proper geometrical model of the camera. Stereoscopic vision, either active or passive, relies on the difference in the appearance of the same objects from two distinct viewpoints. Their correspondence allows for the reconstruction of the depth in the scene. It should be noted that with such an approach the ellipsoid is covered with points not uniformly, see

Figure 6.8. An explicit geometrical model was used to obtain data that closely resembles the real point clouds.

According to the used model, a tomato can be represented as an ellipsoid. Hence, the point cloud obtained from a real tomato can be approximated as the point cloud of an ideal ellipsoid. In this section, the process of image formation is considered.

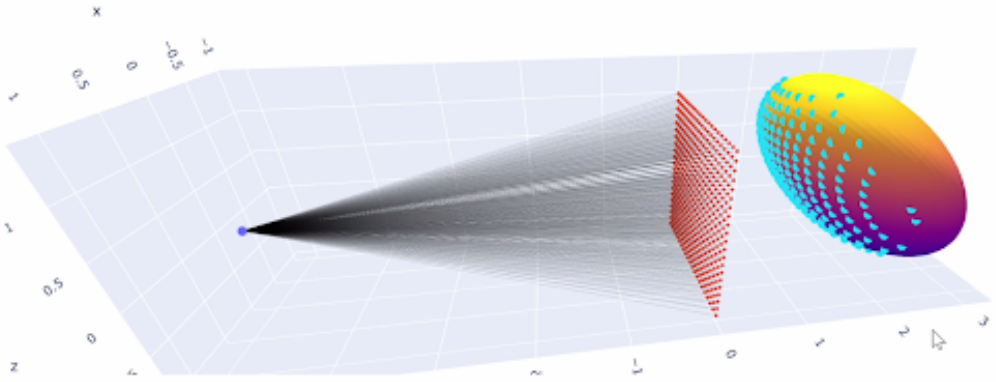


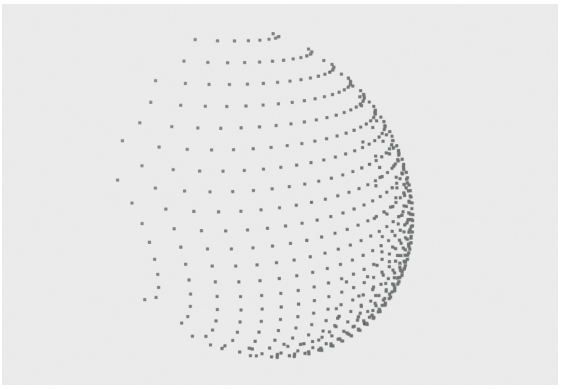
Figure 6.1 — Scheme of the ray tracing-based approach to point cloud generation. On the left the focal point of the camera is presented, the red grid in the middle is the camera matrix, and the rays casted from through it intersect with the surface of the ellipsoid in the points marked with cyan triangles.

The leftmost violet point in Figure 6.8 represents the focal center of the camera, while the red points correspond to the light-sensitive pixels of the camera matrix. A line between the blue and red points visualizes a beam of light reflected from the ellipsoid that intersects with the camera matrix. Extending this line to intersect with the ellipsoid yields a point in the point cloud (cyan triangles). Thus, the problem of finding the intersection of a line formed by two points with the ellipsoid needs to be addressed.

Let us consider an ellipsoid defined by S (scale matrix), R (rotation matrix), and t (translation vector), along with a line formed by points A and B.

First, let us transform the ellipsoid so that it becomes a unit sphere at the origin in order to simplify the problem. The points A and B will be transformed as follows:

$$\begin{aligned} A' &= S^{-1}R^T(A - t), \\ B' &= S^{-1}R^T(B - t). \end{aligned} \tag{19}$$



а) Example of a point cloud of an ellipsoid segment without noise

б) Example of a point cloud of an ellipsoid segment with noise

Figure 6.2 — Samples from a part of the synthetic dataset, generated with the ray tracing: (a) clear (b) with additive normal noise.

The direction vector defining the light beam line in the new coordinates can be found as:

$$a = \frac{A' - B'}{\|A' - B'\|}. \tag{20}$$

The solution for the line-sphere intersection yields two points P_1 and P_2, that can be found according to the subsequent computational steps derived from solving a quadratic equation, if $\Delta \geqslant 0$.

$$\begin{aligned} \Delta &= (a^T A')^2 - (A')^T A + 1, \\ C' &= A' - a^T A' a, \\ P_1' &= C' - a\sqrt{\Delta}, \\ P_2' &= C' + a\sqrt{\Delta}, \\ P_1 &= RS(P_1' + t), \\ P_2 &= RS(P_2' + t). \end{aligned} \tag{21}$$

By repeating this algorithm for each red point representing pixels, an array of points that lie on the ellipsoid is formed, see Figure 6.2a. After that the noise is added into the data, see Figure 6.2б. The amplitude of noise was chosen in accordance with the real data.

6.1.3 Real data

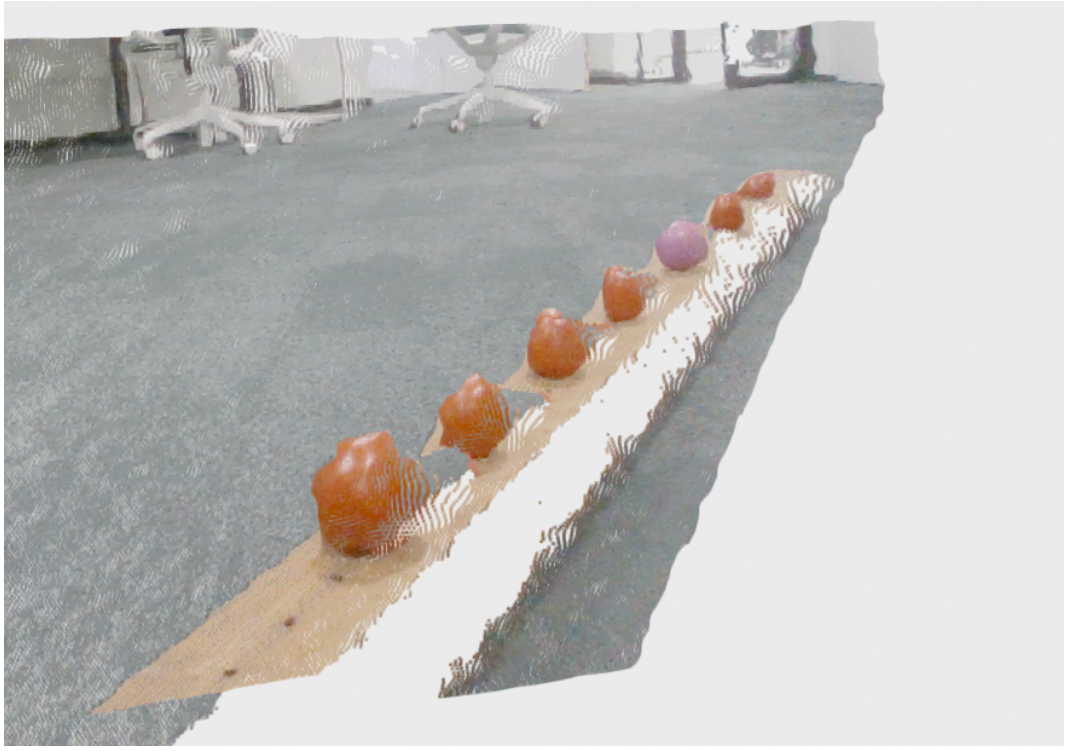


Figure 6.3 — Sample point cloud from the real part of the dataset. Tomatoes and a rubber ball are placed on a wooden support.

The point cloud in the Figure 6.3 is from the real part of the dataset. These point clouds are a challenging piece of data. Tomatoes are described by an ellipsoid model only in a certain approximation, varying from fruit to fruit and from strain to strain. It means that even the best possible fit will result in a certain non-zero error.

Moreover, the real noise cannot be approximated by the normal noise model. In the Figure 6.4 the distribution of the algebraic distance from the ellipsoid surface is presented for the synthetic data. In the Figure 6.5 the same is given for the real data. It could be seen that the distribution for the real data is not unimodal and skewed for all three tomatoes considered, meaning that the real data does not fully comply with the used model.

The last considerable obstacle are the distortions that occur on the surface of the tomatoes. The exact reasons behind this effect are not clear, but such a distorted appearance was noticed a number of times.

The ways to deal with these complications include the following. In order to account for the tomatoes being not precisely ellipsoid a more general model of the fruit can be used, such as hyperellipsoids or even the most general case, such as a

combination of shperical harmonics. The noise with the complex distributions can be dealt with by standard RANCAS, as numerical experiments on the real data suggest. Finally, end-to-end algorithms, such as neural networks, can be used.

To test the algorithm on real data, a series of experiments was conducted with tomatoes located on a horizontal wooden board. The experiment involved six tomatoes and one rubber ball, that were placed at a distance of between 10 and 15 cm from each other, see Figure 6.3. The camera captured scenes at various angles of the board: 55, 60, 70, 80 and 90 degrees. This made it possible to study the effect of the viewing angle on the perception of the shape of objects by the depth camera. For each angle, 146 point clouds were taken. Each point cloud contains approximately 111,000 points, where nearly 2,200 belong to the tomatoes. For a single object this number varies from 831 points for the nearest one to 130 for the farthest. The volume of the tomatoes ranges from 90 to 207 cubic cm. For each tomato the ball their actual size and volume were measured, and the exact location relative to the camera was determined. In this experiment, tomatoes were modeled as ideal spheres, where the radius of the sphere was determined by the average value of the semi-axes of the ellipsoid. The tomatoes used were reasonably close to a sphere, which made it possible to make this assumption fair. As a result, a dataset was created containing the centers and radii of objects, that were compared with the output of the algorithm.

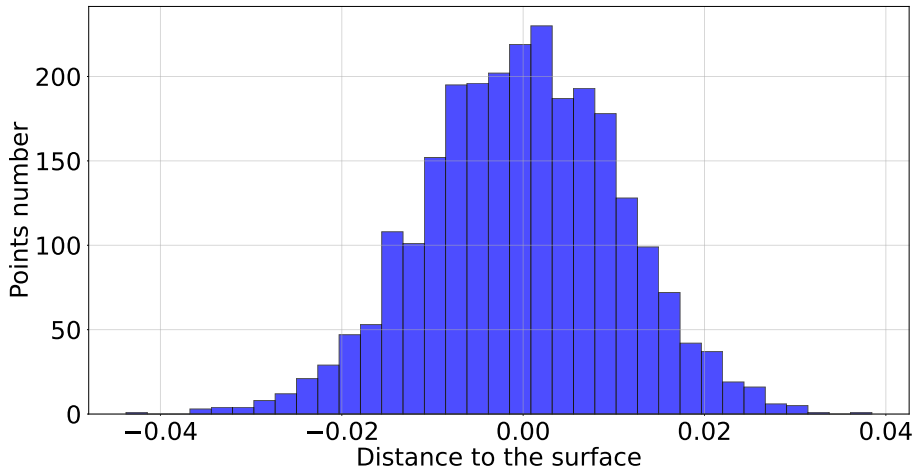


Figure 6.4 — Distribution of the algebraic distance between the points and the surface of the best fitted ellipsoid model for synthetic data. A clear unimodal distribution can be seen.

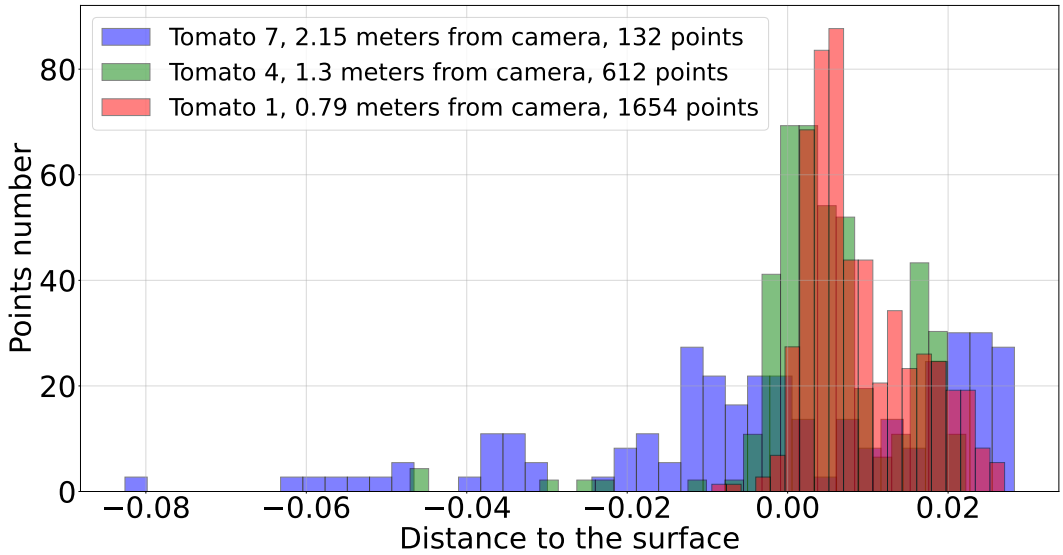


Figure 6.5 — Distribution of the algebraic distance between the points and the surface of the best fitted ellipsoid model for three tomatoes from the real data. For all three of them the distribution is rather noisy and clearly not unimodal, suggesting that the real data is significantly more challenging than synthetic.

The real data was captured with Intel RealSense D435i. The exposure and white balance were set to auto. The scene was captured under artificial lighting.

6.1.4 Data Collection

The data was collected as follows. Thirty tangerines were arranged on a flat table surface in three rows of ten, see Fig. 6.6. They were numbered sequentially from 1 to 30, moving from left to right. An Intel RealSense D435i camera was mounted above the table to capture images from a top-down perspective. After the images were taken, each tangerine was measured along its axes, and its volume was determined. The resulting point clouds were segmented into separate tangerine point clouds and a plane (table surface) using color filtering.

The characteristics of the tangerines captured are as follows.

- Minimum tangerine volume: 41.5 cm^3
- Maximum tangerine volume: 121.1 cm^3
- Average tangerine volume: 76.3 cm^3
- Smallest point cloud: 184 points
- Largest point cloud: 352 points

– Average point cloud size: 270 points

In total, 8 different camera positions were considered with the number of tangerines in the scene varying from 8 to 30. 205 frames were recorded for each scene, including an RGB image, a depth image, and a point cloud.

Figure 6.6 — RGB frame from the dataset. It contains 30 tangerines. The images and the point clouds were captured with Intel RealSense D435i camera.

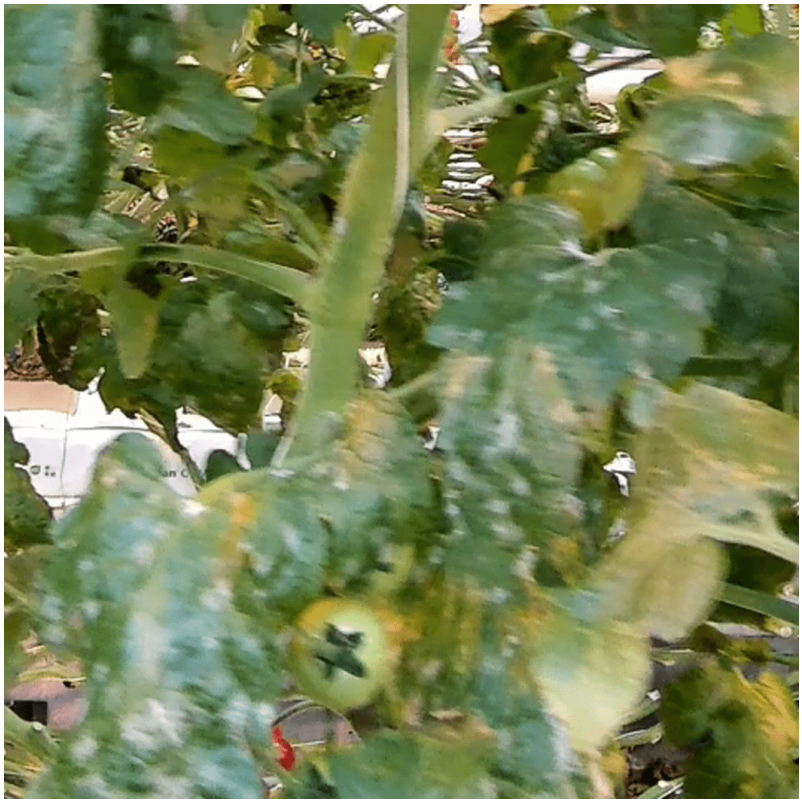
6.2 Data markup and quality evaluation

A set of approximately 300000 images was collected with some of them exhibiting manifestations of the powdery mildew, see Figures 6.7a, 6.7б. In order to obtain a dataset of a reasonable size, every 30-th frame was chosen, resulting in approximately 10000 frames that are uniformly distributed across the entire original dataset.

The process of the data labeling was organized as follows. At each of the 4 iterations, each of the experts was provided with 550 images of the tomato leaves, which had to be separated into two classes: negative and positive. Among these 550 images, there were 500 unique for each expert and 50 images shared by all the experts in order to assess their consistency. The mutual consistencies of the experts in the subsets are given in Table 15.

It is worth noting, that the major part of inconsistencies occur as a consequence of the frame quality. It is possible to overcome this issue by reducing motion blur

a) Image of powdery mildew-infected leaves.



б) A scattering of white patches is a clear visual sign of the presence of the disease.

Figure 6.7 — (a) Image from the bottom camera of the robot. (b) Zoomed in.

with the help of the cameras with a global shutter, however, this work shows that the target problem can be solved with cameras with a rolling shutter.

6.3 Data

6.3.1 Experimental setup

The view of the stand is given in Figure 3.13.

The dataset was collected by simulating the static scenario of the robotic platform aquiring the data while being positioned on the pipe rails. The RealSense D435i camera was mounted vertically downward at a height of 0.9 m from the base of the rails. To capture the lower row of plants, a branch was hang with tomatoes 0.80 m away from the camera, ensuring all fruits were in sight. This setup positioned the tomatoes between 0.92 m and 1.27 m from the camera lens, with a horizontal spread of approximately 0.2 m.

The data was recorded data using both RGB and Depth channels, synchronized via standard RealSense SDK methods. Each tomato was physically marked for identification during post-processing. The lighting conditions were consistent, as the room was equipped with a window that blocks infrared light, and the overcast weather prevented direct sunlight.

By default, auto-exposure and auto-white balance settings were used, which were stable enough for short captures, as confirmed by minimal hyperparameter changes in the frame metadata. The resolution was set to 1920×1080 for RGB and 1280×720 for depth at 30 FPS.

The Fields of View of the cameras are the following.

– Depth FOV: $87° \times 58°$

– RGB FOV: $69° \times 42°$

Given the platform's maximum speed of 1.5 m/s, an evaluation was carried out in order to assure that the rate of 30 FPS was sufficient to avoid motion blur.

6.3.2 Data description and storage

All the data consists of two parts: synthetic and real. Regarding the synthetic data, full ellipsoids were considered, segments of ellipsoids, noisy full ellipsoids and noisy ellipsoid segments.

Regarding the real data, after the point clouds are cropped by YOLO, all the cropped raw points alongside the ground truth data are stored in numpy *.npz* format. The number of point clouds is the following:

- 146 point clouds with 14 tomatoes each for the real data
- 200 point clouds for clean data
- 200 for clean segment
- 900 for noised full ellipsoids
- 1100 for segments of noised segments

In order to obtain the ellipsoid model, and .npz file is loaded, and 1000 for synthetics and 1000 for polygon sets of 9 points are sampled from the point cloud with permutations and without replacement to ensure no duplicate sets exist using torch library to speed up the process using GPU. For each 10000 iteration a system of equations is formed and solved. The results are stored in their entirety using numpy .npz format.

For each combination of hyperparameters (43 for polygon and 67 for synthetics), a table with 1000 rows and 52 columns, including the predicted characteristics and IoU, is produced. It is stored as a columnar data in parquet files for the sake of low storage capacity to overcome and reduce the potential capacity requirement down to almost 100 GB overall.

Analysing all data at once would become difficult considering 100 GB of data and only 16 GB of RAM capacity. Thus, DuckDB database system supporting parquet format and SQL queries is used. In order to ease the analysis, each time a filter predicate is defined and a SQL query executed through DuckDB to benefit from both the high read/write speed and low memory usage. The data received from DuckDB, then is stored in a polars dataframe as a tool for analysis.

Markers	A	B	C	D
A	1.0	0.72	0.88	0.82
B		1.00	**0.60**	0.90
C			1.00	**0.70**
D				1.00

1 iteration

Markers	A	B	C	D
A	1.0	0.88	**0.68**	0.96
B		1.00	0.80	0.92
C			1.00	0.72
D				1.00

2 iteration

Markers	A	B	C	D
A	1.0	0.92	0.86	0.90
B		1.00	0.94	0.98
C			1.00	0.96
D				1.00

3 iteration

Markers	A	B	C	D
A	1.0	0.94	0.92	0.92
B		1.00	0.89	0.86
C			1.00	0.92
D				1.00

4 iteration

Table 15 — Tables of mutual consistencies of the human experts.
Each table represents one of 4 iterations of the data markup. The consistencies of 0.7 and below are highlighted in bold. A, B, C, and D in the tables represent individual human experts. The number in ij-th element of the table is the fraction of the matching labels (positive or negative) for the corresponding experts on a subset of 50 images that they shared during that iteration. The mean consistency (excluding the consistency of the experts with themselves) is 0.8579, which closely matches the accuracy of the best model.

Conclusion

Overall, a work was done in the field of automated greenhouse monitoring. The main contribution lies in the intersection of theoretically supported developments in the sufrace recognition and engineering problems that were solved during this work.

The main results of this dissertation are:

1. A method of ellipsoid volume estimation was developed and tested in the real environment. Synthetic data was generated with gradual increase in complexity. A real dataset of point clouds and corresponding RGB images was gathered and marked. The dependence of the quality of the method's outout on the iterations number was evaluated in numerical experiments, showing minimal average relative volume estimation error of 0.25.

   The comparison of the method with three alternative approaches was made, showing superior performance on the real data both in terms of IoU and volume error. It was thus shown that the proposed method can be applied to the real-world ellipsoid volume estimation.
2. A method of robust volume estimation was applied to tangerines. A dataset of point clouds and RGB images was gathered, and the tangerines were measured in terms of mass and volume. Numerical experiments were conducted with varying hyperparameters. The results suggest that Random Sample Consensus can be successfully applied in the problem of volume estimation. The proposed approach has a number of applications in agriculture. First, it can be applied to the problem of the instant yield estimation. In order to assess the volume of the yield, a robot can be used that will monitor the agricultural facility and provide the estimated volume. Second, the measurements of the volume across different time periods can be helpful for the development of a prognostic tool for the prospective yield.

   Future work can include the generalization of the method to other types of fruit, including non-elliptical ones. In particular, they can have a shape of a curved ellipsoid, like a banana, or a superellipsoid, like sweet pepper.

   The proposed method can run on a normal computer with resonably computational load. In future RANCAS for ellipsoids can be adapted to the CUDA-based inference in order to speed up the computations.

3. A neural network-based end-to-end tangerine volume estimation method was proposed. It relies on the PointNet++ architecture. The training on the custom dataset takes nearly 1.5 hours, and the error in the volume estimation lightly exseeds 10%.

   The inference time of 290 milliseconds allows for the real-time onboard inference with limited computational resources. The proposed approach can be used on mobile robots for the agricultural facilities monitoring.
4. An autonomous agricultural robot on an omnidirectional platform was designed, prototyped, and tested in a real environment. The camera's positioning allows for the examination of the whole tomato plant, and the onboard computer is capable of processing the data in real-time.

   A dataset of powdery mildew-infected tomato leaves was collected, labeled and assessed in terms of consistency. It contains a sufficient amount of data for training modern neural networks of reasonable size, which makes it useful for further industrial applications.

   Several modern neural networks were trained for classification and compared on two test sets with one of them being marked by a consensus of experts. The performance of the models is sufficient and matches with the mutual consistencies of the human experts.

   It was demonstrated that real-time disease recognition could be performed on the robot while moving with the means of the user-grade cameras. The main limitation of the chosen approach is that RGB cameras require sufficient lighting to function properly. However, in the given circumstances this requirement is naturally met.

Overall, the work advances the ongoing endevour of automating the greenhouse monitoring. There are still problems to be addressed, in particular they are connected to the integration of the proposed technologies into the on-site practice in the real greenhouses.

The next steps in this project are planned to be the following.

First, high-level control should be implemented in the system for the robot to be capable of localizing itself and autonomously following the trajectory set by the user.

Second, a Graphical User Interface should be implemented for the end user to control the robot without substantial UNIX knowledge.

Figure 6.8 — The third version of the robot with the camera masts installed. The height of the robot is approximately 3.5 meters, allowing it to inspect the whole tomato plant in one go.

Third, more disease types and crop types should be introduced into the vision subsystem.

Acknowledgements

I am deeply grateful to my colleagues that I had a pleasure to work with on this project: Ilya Ryakin, Sina Moghimi, Mikhail Patrikeev, Ilya Barsky, Sergei Davidenko, and Vladimir Guneavoy.

I thank my research supervisor Pavel Osinenko for his guidance and insites about academic perseverance. I express my gratitude to Grigory Yaremenko for introducing me to a number of fundamental concepts in modern control theory and for his neverending curio-sceptical rethinking of science and knowledge.

I am grateful to my friends and family, who supported me during this long journey. I thank Marina for the patience and support. I thank Roman Mikhailov for his cinematography and advice about PhD. I thank my parents for giving me an opportunity to study math and physics.

References

1. MetaFruit meets foundation models: Leveraging a comprehensive multi-fruit dataset for advancing agricultural foundation models / J. Li [et al.] // Computers and Electronics in Agriculture. — 2025. — Vol. 231. — P. 109908.
2. *Tahir, G. A.* A comprehensive survey of image-based food recognition and volume estimation methods for dietary assessment / G. A. Tahir, C. K. Loo // Healthcare. Vol. 9. — MDPI. 2021. — P. 1676.
3. Direct and accurate feature extraction from 3D point clouds of plants using RANSAC / M. Ghahremani [et al.] // Computers and Electronics in Agriculture. — 2021. — Aug. — Vol. 187. — P. 106240.
4. Intel RealSense Stereoscopic Depth Cameras / L. Keselman [et al.]. — 2017. — URL: https://arxiv.org/abs/1705.05548.
5. Direct and accurate feature extraction from 3D point clouds of plants using RANSAC / M. Ghahremani [et al.] // Computers and Electronics in Agriculture. — 2021. — Vol. 187. — P. 106240.
6. An Improved Adaptive Threshold RANSAC Method for Medium Tillage Crop Rows Detection / Y. Xie [et al.] // 2021 6th International Conference on Intelligent Computing and Signal Processing (ICSP). — IEEE. 2021. — P. 1282—1286.
7. *Hossein-Nejad, Z.* A-RANSAC: Adaptive random sample consensus method in multimodal retinal image registration / Z. Hossein-Nejad, M. Nasri // Biomedical Signal Processing and Control. — 2018. — Vol. 45. — P. 325—338. — URL: https://www.sciencedirect.com/science/article/pii/S174680941830154X.
8. Ransac for robotic applications: A survey / J. M. Martínez-Otzeta [et al.] // Sensors. — 2022. — Vol. 23, no. 1. — P. 327.
9. An improved RANSAC for 3D point cloud plane segmentation based on normal distribution transformation cells / L. Li [et al.] // Remote Sensing. — 2017. — Vol. 9, no. 5. — P. 433.
10. Computer vision model for estimating the mass and volume of freshly harvested Thai apple ber (*Ziziphus mauritiana L.*) and its variation with storage days / S. M. Mansuri [et al.] // Scientia Horticulturae. — 2022. — Vol. 305. — P. 111436.

11. *Huynh, T. T. M.* A vision-based method to estimate volume and mass of fruit/vegetable: Case study of sweet potato / T. T. M. Huynh, L. TonThat, S. V. T. Dao // International Journal of Food Properties. — 2022. — Vol. 25, no. 1. — P. 717—732.

12. *Jana, S.* A De novo approach for automatic volume and mass estimation of fruits and vegetables / S. Jana, R. Parekh, B. Sarkar // Optik. — 2020. — Vol. 200. — P. 163443.

13. *Khojastehnazhand, M.* Determination of tangerine volume using image processing / M. Khojastehnazhand, M. Omid, A. Tabatabaeefar // Tarım Makinaları Bilimi Dergisi. — 2008. — Vol. 4, no. 4. — P. 407—412.

14. *Khojastehnazhand, M.* Determination of tangerine volume using image processing methods / M. Khojastehnazhand, M. Omid, A. Tabatabaeefar // International Journal of Food Properties. — 2010. — Vol. 13, no. 4. — P. 760—770.

15. *Omid, M.* Estimating volume and mass of citrus fruits by image processing technique / M. Omid, M. Khojastehnazhand, A. Tabatabaeefar // Journal of Food Engineering. — 2010. — Vol. 100, no. 2. — P. 315—321.

16. DIVESPOT: Depth Integrated Volume Estimation of Pile of Things Based on Point Cloud / Y. Ling [et al.] // arXiv preprint arXiv:2407.05415. — 2024.

17. *Fischler, M. A.* Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography / M. A. Fischler, R. C. Bolles // Communications of the ACM. — 1981.

18. *Chaivivatrakul, S.* Towards Automated Crop Yield Estimation / S. Chaivivatrakul, J. Moonrinta, M. N. Dailey. — 2010.

19. Combined Framework for Real-time Head Pose Estimation using Facial Landmark Detection and Salient Feature Tracking. / J. M. D. Barros [et al.] // VISIGRAPP (5: VISAPP). — 2018. — P. 123—133.

20. *Han, M.* Ellipsoid fitting using variable sample consensus and two-ellipsoid-bounding-counting for locating Lingwu long Jujubes in a natural environment / M. Han, J. Kan, Y. Wang // IEEE Access. — 2019. — Vol. 7. — P. 164374—164385.

21. Field-grown tomato yield estimation using point cloud segmentation with 3D shaping and RGB pictures from a field robot and digital single lens reflex cameras / B. Ambrus [et al.] // Heliyon. — 2024. — Vol. 10, no. 20.

22. *Chopin, J.* A new method for accurate, high-throughput volume estimation from three 2D projective images / J. Chopin, H. Laga, S. J. Miklavcic // International Journal of Food Properties. — 2017. — Vol. 20, no. 10. — P. 2344—2357.

23. A Bayesian Approach Toward Robust Multidimensional Ellipsoid-Specific Fitting / M. Zhao [et al.] // IEEE Transactions on Pattern Analysis and Machine Intelligence. — 2024.

24. *Yury.* Ellipsoid fit / Yury. — n.d. — Accessed: 2025-03-02. https://www.mathworks.com/matlabcentral/fileexchange/24693-ellipsoid-fit.

25. Robust ellipsoid fitting using combination of axial and Sampson distances / M. Han [et al.] // IEEE Transactions on Instrumentation and Measurement. — 2023. — Vol. 72. — P. 1—14.

26. *Hough, P. V. C.* Method and means for recognizing complex patterns / P. V. C. Hough. — 1962.

27. Tomato volume and mass estimation using computer vision and machine learning algorithms: Cherry tomato model / I. Nyalala [et al.] // Journal of Food Engineering. — 2019. — Vol. 263. — P. 288—298.

28. Prostate volume estimation using the ellipsoid formula consistently underestimates actual gland size / E. Rodriguez [et al.] // The Journal of Urology. — 2008. — Vol. 179, no. 2. — P. 501—503.

29. Outlier elimination for robust ellipse and ellipsoid fitting / J. Yu [et al.] // 2009 3rd IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP). — IEEE. 2009. — P. 33—36.

30. *Raguram, R.* A comparative analysis of RANSAC techniques leading to adaptive real-time random sample consensus / R. Raguram, J.-M. Frahm, M. Pollefeys // Computer Vision–ECCV 2008: 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part II 10. — Springer. 2008. — P. 500—513.

31. *Torr, P. H.* MLESAC: A new robust estimator with application to estimating image geometry / P. H. Torr, A. Zisserman // Computer vision and image understanding. — 2000. — Vol. 78, no. 1. — P. 138—156.

32. *Hossein-nejad, Z.* Image registration based on SIFT features and adaptive RANSAC transform / Z. Hossein-nejad, M. Nasri // 2016 International Conference on Communication and Signal Processing (ICCSP). — IEEE. 2016. — P. 1087—1091.

33. 1-point RANSAC-based method for ground object pose estimation / J.-K. Lee [et al.] // arXiv preprint arXiv:2008.03718. — 2020.

34. RANSAC Back to SOTA: A Two-Stage Consensus Filtering for Real-Time 3D Registration / P. Shi [et al.] // IEEE Robotics and Automation Letters. — 2024.

35. Vision-based Navigation Line Extraction by Combining Crop Row Detection and RANSAC Algorithm / X. Li [et al.] // 2022 IEEE International Conference on Mechatronics and Automation (ICMA). — 2022. — P. 1097—1102.

36. *Fischler, M. A.* Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography / M. A. Fischler, R. C. Bolles // Communications of the ACM. — 1981. — Vol. 24, no. 6. — P. 381—395.

37. *Rosten, E.* Improved RANSAC performance using simple, iterative minimal-set solvers / E. Rosten, G. Reitmayr, T. Drummond // arXiv preprint arXiv:1007.1432. — 2010.

38. Generalized differentiable RANSAC / T. Wei [et al.] // Proceedings of the IEEE/CVF International Conference on Computer Vision. — 2023. — P. 17649—17660.

39. Consensus-adaptive ransac / L. Cavalli [et al.] // arXiv preprint arXiv:2307.14030. — 2023.

40. 1-Point RANSAC-Based Method for Ground Object Pose Estimation / J.-K. Lee [et al.] // ArXiv. — 2020. — Vol. abs/2008.03718. — URL: https://api.semanticscholar.org/CorpusID:221090323.

41. You only look once: Unified, real-time object detection / J. Redmon [et al.] // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. — 2016. — P. 779—788.

42. PointNet: Deep learning on point sets for 3D classification and segmentation / C. R. Qi [et al.] // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. — 2017. — P. 652—660.

43. KPConv: Flexible and deformable convolution for point clouds / H. Thomas [et al.] // Proceedings of the IEEE/CVF International Conference on Computer Vision. — 2019. — P. 6411—6420.

44. A novel tomato volume measurement method based on machine vision / H. Li [et al.] // Tehnički vjesnik. — 2021. — Vol. 28, no. 5. — P. 1674—1680.

45. *Andrews, J.* Type-constrained direct fitting of quadric surfaces / J. Andrews, C. H. Séquin // Computer-Aided Design and Applications. — 2014. — Vol. 11, no. 1. — P. 107—119.

46. *Abbott, P.* On the Perimeter of an Ellipse / P. Abbott // Mathematica. — 2009. — P. 172—185.

47. *Groshong, B.* Fitting a quadratic surface to three dimensional data : tech. rep. / B. Groshong, G. L. Bilbro, W. E. Snyder ; North Carolina State University. Center for Communications ; Signal Processing. — 1989.

48. *Wspanialy, P.* Early powdery mildew detection system for application in greenhouse automation / P. Wspanialy, M. Moussa // Computers and Electronics in Agriculture. — 2016. — Vol. 127. — P. 487—494. — URL: https://www.sciencedirect.com/science/article/pii/S0168169916304318.

49. An open access repository of images on plant health to enable the development of mobile disease diagnostics / D. Hughes, M. Salathé, [et al.] // arXiv preprint arXiv:1511.08060. — 2015.

50. Direct importance estimation for covariate shift adaptation / M. Sugiyama [et al.] // Annals of the Institute of Statistical Mathematics. — 2008. — Vol. 60. — P. 699—746.

51. *Barbedo, J. G. A.* A review on the main challenges in automatic plant disease identification based on visible range images / J. G. A. Barbedo // Biosystems engineering. — 2016. — Vol. 144. — P. 52—60.

52. *Teeffelen, D. van.* Plant disease detection with machine and deep learning : PhD thesis / van Teeffelen Dide. — KN Wageningen, 2023.

53. Detection Method for Tomato Leaf Mildew Based on Hyperspectral Fusion Terahertz Technology / X. Zhang [et al.] // Foods. — 2023. — Vol. 12, no. 3. — P. 535.

54. Cucumber powdery mildew detection using hyperspectral data / C. I. Fernández [et al.] // Canadian Journal of Plant Science. — 2021. — Vol. 102, no. 1. — P. 20—32.

55. Hyperspectral Monitoring of Powdery Mildew Disease Severity in Wheat Based on Machine Learning / F. Ziheng [et al.] // Frontiers in Plant Science. — 2022. — Mar. — Vol. 13. — P. 828454.

56. Detecting powdery mildew disease in squash at different stages using UAV-based hyperspectral imaging and artificial intelligence / J. Abdulridha [et al.] // Biosystems engineering. — 2020. — Vol. 197. — P. 135—148.

57. *Varshney, T.* Deep Learning Models for Prediction of Tomato Powdery Mildew Disease / T. Varshney, A. Chug, A. P. Singh // 2021 8th International Conference on Signal Processing and Integrated Networks (SPIN). — IEEE. 2021. — P. 1036—1041.

58. Deep learning-based segmentation and quantification of cucumber powdery mildew using convolutional neural network / K. Lin [et al.] // Frontiers in plant science. — 2019. — Vol. 10. — P. 155.

59. *BOYAR, T.* Powdery Mildew Detection in Hazelnut with Deep Learning / T. BOYAR, K. Yildiz // Hittite Journal of Science and Engineering. — 2022. — Sept. — Vol. 9. — P. 159—166.

60. *Varshney, T.* Deep Learning Models for Prediction of Tomato Powdery Mildew Disease / T. Varshney, A. Chug, A. P. Singh // 2021 8th International Conference on Signal Processing and Integrated Networks (SPIN). — 2021. — P. 1036—1041.

61. Effect of directional augmentation using supervised machine learning technologies: A case study of strawberry powdery mildew detection / J. Shin [et al.] // Biosystems Engineering. — 2020. — Mar. — Vol. 194.

62. RoboFlow: a data-centric workflow management system for developing AI-enhanced Robots / Q. Lin [et al.] // Conference on Robot Learning. — PMLR. 2022. — P. 1789—1794.

63. ultralytics/yolov5: v7. 0-yolov5 sota realtime instance segmentation / G. Jocher [et al.] // Zenodo. — 2022.

64. *Tan, M.* Efficientnet: Rethinking model scaling for convolutional neural networks / M. Tan, Q. Le // International conference on machine learning. — PMLR. 2019. — P. 6105—6114.

65. Deep residual learning for image recognition / K. He [et al.] // Proceedings of the IEEE conference on computer vision and pattern recognition. — 2016. — P. 770—778.

66. Mobilenets: Efficient convolutional neural networks for mobile vision applications / A. G. Howard [et al.] // arXiv preprint arXiv:1704.04861. — 2017.

List of figures

List of tables