MUSHNOMICS

Unlocking data-driven innovation for improving productivity and data sharing in mushroom value chain

D2.2 - Report on MUSHNOMICS AI-algorithms

Document	
Deliverable title	Report on MUSHNOMICS AI-algorithms
Related Work package	WP2
Responsible editor	Dimitrios Argyropoulos (UCD)
Contributors	Christos Charisis, Dimitrios Argyropoulos (UCD)
Delivery date	M29 (31/07/2023)

Version history			
Author	Comment	Version	Date
Christos Charisis	Draft	0.1	
Sari Nuwayhid	Draft	0.3	
Dimitrios Argyropoulos	Final	1.0	

Disclaimer

The MUSHNOMICS Consortium is the following:

No	Beneficiary Name	Short Name	Country
1	SC HOLISUN SRL	HS	Romania
2	Department of Plant and Environmental Sciences, University of Copenhagen	UCPH	Denmark
3	Pilze-Nagy Ltd	PILZE	Hungary
4	University College Dublin	UCD	Ireland

Table of contents

List of Figures

List of Tables

1. Introduction

Accurate mushroom detection is a prerequisite for automated growth tracking in real production environments and represents a significant challenge. Recent developments in the field of sensors and computer vision have promoted the application of crop detection systems, leading to their increasing use in agricultural task automation. Computer-vision models have been widely used and undergo certain improvements in the field of crop detection and localization in order to assist crop growth monitoring. Therefore, precision agriculture needs to take advantage of the latest cutting-edge technologies in computer science in order to perform sustainable and efficient agronomic management, thus increasing production at a low environmental and economic cost.

This document represents the deliverable 2.2 and details the design and development of the AI algorithms for the detection and segmentation of mushrooms in the MUSHNOMICS project. The primary novelty of the MUSHNOMICS AI approach lies in the integration of deep learning (DL) technology with instance segmentation for the rapid identification of mushrooms in a farm environment. This integration offers a visually intuitive and contextually informative platform, bridging the gap between digital predictions and real-world scenarios.

Specifically, an image-based system that captures images of mushrooms in a farm environment and displays the predicted growth instances by utilizing cutting-edge deep learning technology was designed and tested for the rapid detection and segmentation of mushrooms at different growth stages. The algorithms were developed using a comprehensive image dataset of mushrooms grown on state-of-the-art substrate blocks and validated using images acquired under several production trials with the MUSHNOMICS buckets (D2.3). Finally, the DL-based framework for the real-time detection of mushroom instances integrated into MUSHNOMICs platform (D4.2) and demonstrated as a practical course in (D4.3).

This report provides an overview of the algorithm design, testing and implementation processes also giving an insight into the key considerations and functions for the planning, installation of equipment, execution of experiments and production trials in the mushroom farm (PILZE, Hungary). Based on the outcomes reported here, the MUSHNOMICS AI algorithms were developed, tested and validated using a series of image datasets acquired from multiple production units at PILZE premises.

In summary, the novel MUSHNOMICS end-to-end DL-based instance segmentation pipeline includes six different steps and described in detail in the following chapters:

- Acquisition of image datasets using RGB cameras in a farm environment (Pilze, Hungary);
- Development of a semi-automatic annotation procedure for mushrooms in the images;
- Application of pre-processing techniques to counter possible data defects;
- Selection and testing of suitable deep learning instance segmentation architectures;
- Evaluation of model performance through performance metrics;
- Validation of the MUSHNOMICS algorithms using new image datasets (D2.3).

2. Description of the DL pipeline for static and dynamic datasets

Nowadays, deep learning is one of the most prominent techniques for digital image processing and analysis with important applications across a multitude of industry sectors including agriculture. Existing deep convolutional neural networks (CNNs), the most utilized DL neural network type, have achieved impressive performance in object detection and semantic image segmentation. Instance segmentation models, in turn, are based largely on CNNs. Such models are capable of learning complex patterns and features in the images while extracting fine-gained information in an automated way.

While commercially cultivated oyster mushrooms are primarily grown in controlled environments, the spatial variability of clusters within the farm introduces complexities that challenge the adaptivity of computer vision systems to real-world farming conditions. Digital images represent important data sources that could provide precise, region-specific information and address a variety of agricultural problems. However, image datasets for oyster mushrooms do not exist. An efficient localization and accurate mapping of complex morphology agricultural products such as the oyster mushrooms can help farmers to tackle this variability and mitigate the labor-intensive cost associated with harvesting.

The aim of this task (T2.2), therefore, was to design, test and validate a robust DL-based system to support detection, localization and tracking of mushrooms in a farm environment. Towards this objective, a comprehensive image dataset from a commercial oyster mushroom farm under real production conditions was developed. The proposed solution was based on the segmentation of single mushrooms in a cluster and as whole mushroom cluster from RGB images using deep convolutional neural networks (Mask R-CNN model).

An end-to-end DL-based instance segmentation pipeline consists of four different steps, namely (i) data acquisition, (ii) data annotation, (iii) data preprocessing, and (iv) instance segmentation. After the pipeline is established, the last step is evaluating its performance based on domain-specific performance metrics (v). The steps of the complete pipeline are shown in the schematic of Figure 1.

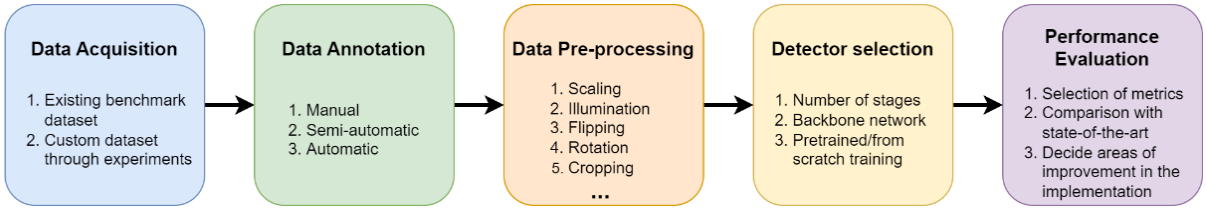

Figure 1. End-to-end deep learning-based instance segmentation pipeline includes image dataset acquisition, annotation, preprocessing, instance segmentation detector, and performance evaluation.

Data acquisition can be conducted in two different approaches. The first path is to explore and find existing benchmark datasets, similar to the aims of the project. The availability of such datasets is usually limited and, in some cases, non-existent (e.g., in our case, oyster mushroom image datasets are not available). The second path is to collect the necessary data through various

experiments utilizing various sensors inside the farm installations. The MUSHNOMICS project followed the second path, scheduling and executing a series of experiments and data collection procedures in order to create a comprehensive dataset capable of training DL models.

The first step after data collection is to annotate the images in the dataset, making them compatible with supervised learning architectures. This is a crucial step as supervised learning requires that the data provided to the model are labeled, guiding the model to learn from the examples provided to it. If the dataset is already constructed in a benchmark format, usually it is annotated by the creators. However, in the case of manually collected data, the annotation process is unavoidable. As the MUSHNOMICS project collected its own dataset, the annotation process was manually conducted for the preparation of the dataset.

Depending on the quality of the data or the needed input for the DL architectures, data pre-processing techniques can be employed. Some of the most common techniques are implemented for data augmentation when the size of the dataset is small, illumination correction for issues in the lighting conditions, and size reduction by scaling, resizing, cropping, etc. In the MUSHNOMICS project, the collected data were in no need of extended pre-processing.. The only pre-processing applied was the default techniques of the DL library for the instance segmentation architectures, which included resizing, random flipping, and normalization to be compatible with the pre-trained weights of the model used through the transfer learning approach.

When all the previous steps for dataset preparation are completed, the next stage is the selection of the DL instance segmentation detector. A great number of different detector architectures are published in the research community, with many different configurations and pre-trained components. The three main CNN-based detectors are:

- One-stage: In this category, the detectors calculate bounding boxes, class labels, and instance masks in a single step inside the model. This family of detectors usually shows higher processing speed and lower accuracy, as their architectures are characterized by simplicity in their components. This kind of detector is widely preferred in applications where real-time results are crucial, such as robotic applications for detecting and harvesting target crops.
- Two-stage: The detectors of this category divide the detection task into two distinct stages. The first stage comprises a region proposal component, which is tasked with locating areas of interest inside the image. These areas are then forwarded to the next stage which examines the area deciding if an object is present, along with the bounding box, class label, and instance mask. Typically, a detector of this category can achieve higher accuracy than a One-stage counterpart, but with lower processing speed. If the application does not require real-time speed performance, these detectors are the most popular option.
- Muli-stage: This category is an expansion of the Two-stage, as more than one stage is present in the architecture before the final proposal of a region of interest. The existence of more stages usually contributes to even higher accuracy at the expense of more

processing time and resources. Their use is encouraged when a quick response is not required but the need for higher accuracy is imperative.

In the following paragraphs, each of the aforementioned parts of the pipeline will be discussed in detail with respect to the models developed in MUSHNOMICS .

3. Experimental setup and image acquisition

This task involved the creation of a comprehensive image dataset of oyster mushroom clusters (*Pleurotus ostreatus* (Jacq. Ex Fr.) P. Kumm) grown on wheat substrate blocks. The dataset was constructed by collecting several images from commercial mushroom farms. The set of images was acquired in June 2022 at Pilze-Nagy Ltd, located in Budapest, Hungary. Figure 2 shows the installation and setting up of a timelapse camera in the farm for acquiring images of mushrooms at the different growth stages throughout the entire production cycle.

Figure 2. Mushroom farm (a) with artificial lighting, configuration, and installation of a timelapse camera (b, c).

The data collection aim was to create two kinds of datasets. One for detecting single mushrooms inside a cluster (static) and one for capturing the growth of a cluster until it reaches the harvesting point (dynamic). For the former, unharvested mushroom clusters were meticulously captured from multiple angles using the dual 12MP camera system of an iPhone 12 smartphone, which offers a resolution of 4032x3024 pixels. Field photography primarily took place during daylight hours, utilizing artificial illumination within the controlled environment capturing in detail the individual mushrooms within a cluster.

A collection of 200 RGB images was obtained, depicting raw oyster mushrooms in clusters with varying sizes, shapes, and distribution densities. Figure 3 presents four representative sample images from this dataset. Particular emphasis was placed on acquiring the clusters from different points of view in a systematic way (front, side, top and bottom views)

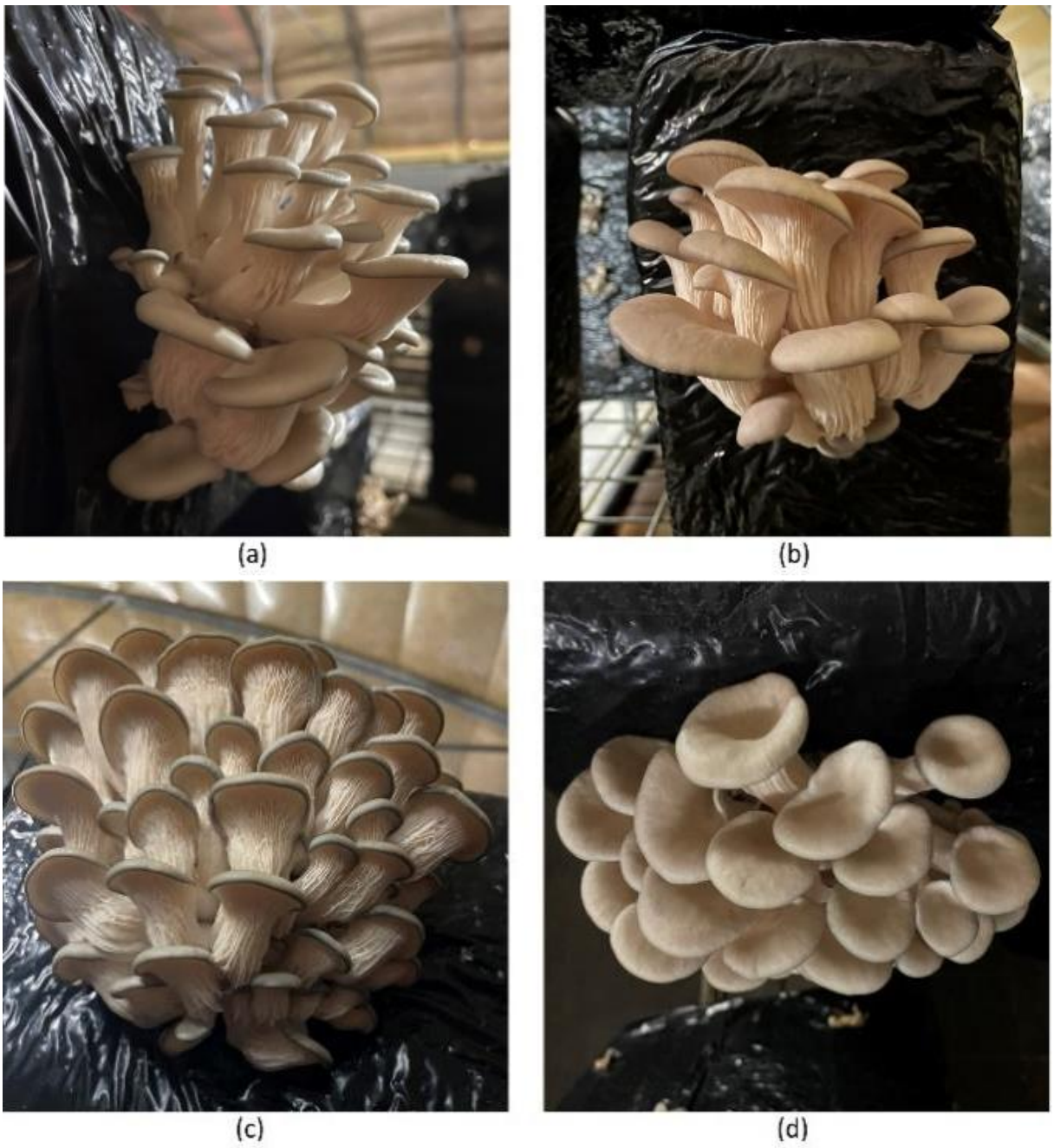

Figure 3. Examples of mushroom cluster images from the collected dataset at different angles: side (a), front (b), bottom (c), and top (d) views obtained from a real production environment.

Furthermore, the dataset contains clusters including different numbers of individual mushrooms, with some clusters consisting of fewer than 10 single mushrooms and others consisting of more than 30 mushrooms. The presence of semi-occluded instances may be discerned from the images. An important challenge to overcome by the instance segmentation task is the detection of individual hidden mushrooms due to their very small visible fruiting body part.

A timelapse camera was set to capture images of the mushroom clusters growing on a substrate block every 15 minutes. The Wingscapes TimelapseCam Pro camera [1] captures very high-resolution images of 6080x3420 pixels, with demonstration samples that can be seen in Figure 4 collected over a full production cycle. The illumination was artificial from the existing lights inside the farm, assisted by the flash of the camera, resulting in high-quality images where the foreground was adequately illuminated while the background remained relatively dark. In total 4 such time-lapse dynamic image datasets were collected from the mushroom farm, each containing approximately 4000 images.

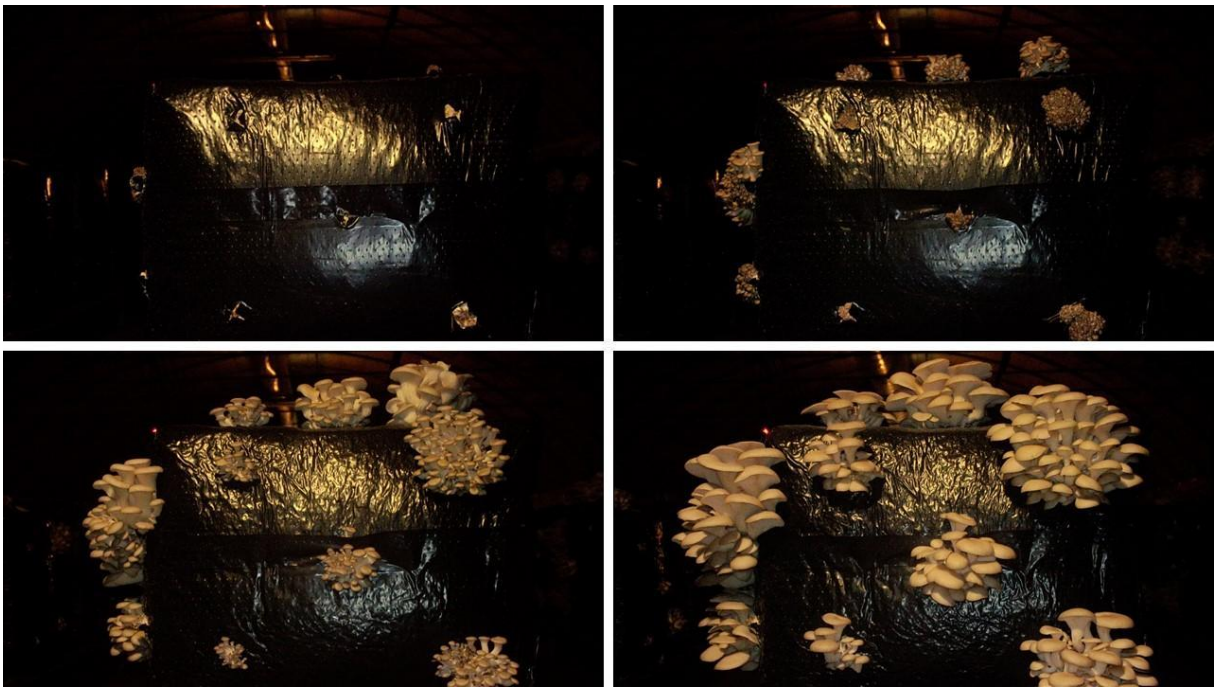

Figure 4. Sample images of the dynamic dataset collected by the timelapse camera installed in the mushroom farm. Different stages of mushroom cluster growth are exemplarily illustrated.

4. Image dataset and annotation process

The raw datasets of oyster mushrooms were obtained and subsequently transformed into formats suitable for training and evaluating deep neural network architectures. The models were trained using the supervised learning approach, which required the construction of target label-masks for the collected images. The process of annotating the images involved accurately defining the boundaries of each mushroom within the cluster depicted in each image (static dataset), and the boundaries of each mushroom cluster grown from the substrate (dynamic dataset).

The annotation process involved manual and semi-manual detection and labeling of the distinct mushroom regions inside the oyster clusters. This was accomplished using an open-source tool named CVAT (Computer Vision Annotation Tool) [2], which facilitated the annotation of images and videos. The resulting annotations served as the necessary labels for training, validation, and testing of the models. The graphical user interface (GUI) of CVAT is depicted in Figure 5.

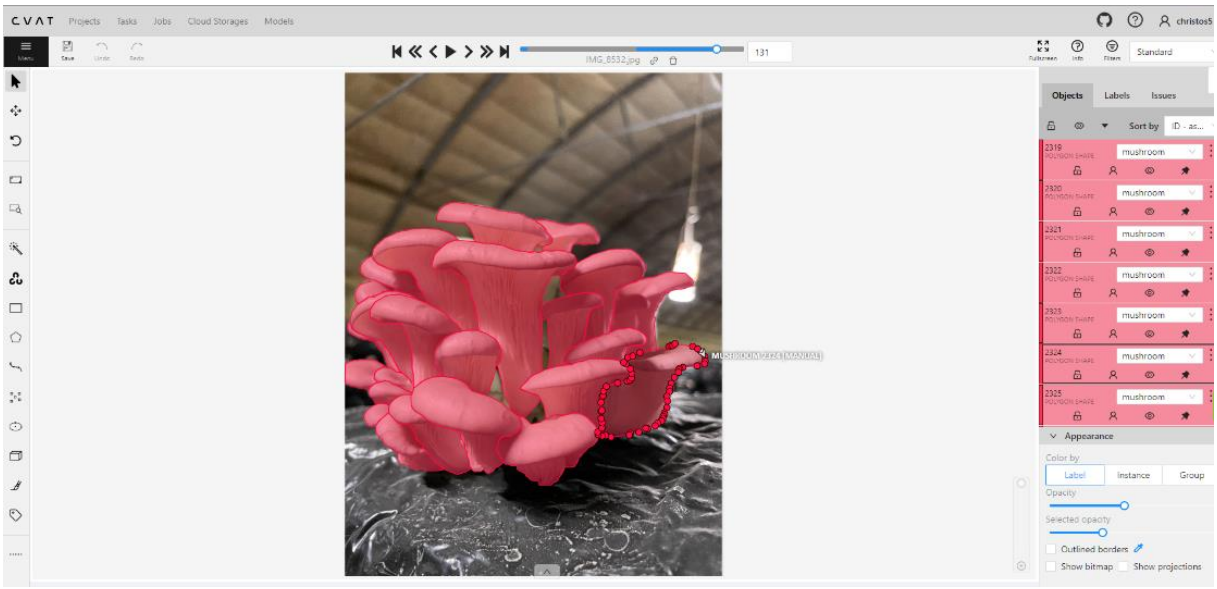

Figure 5. Annotation of mushroom instances using CVAT (Computer Vision Annotation Tool) tool interface.

The region of a mushroom was manually delineated by inserting red dots, as shown in Figure 5. The annotation approach was found to be time-consuming due to the number of mushrooms within the clusters, resulting in an average annotation rate of 1.5 images per hour. Figure 6 presents more examples of annotated single mushroom instances in the static dataset.

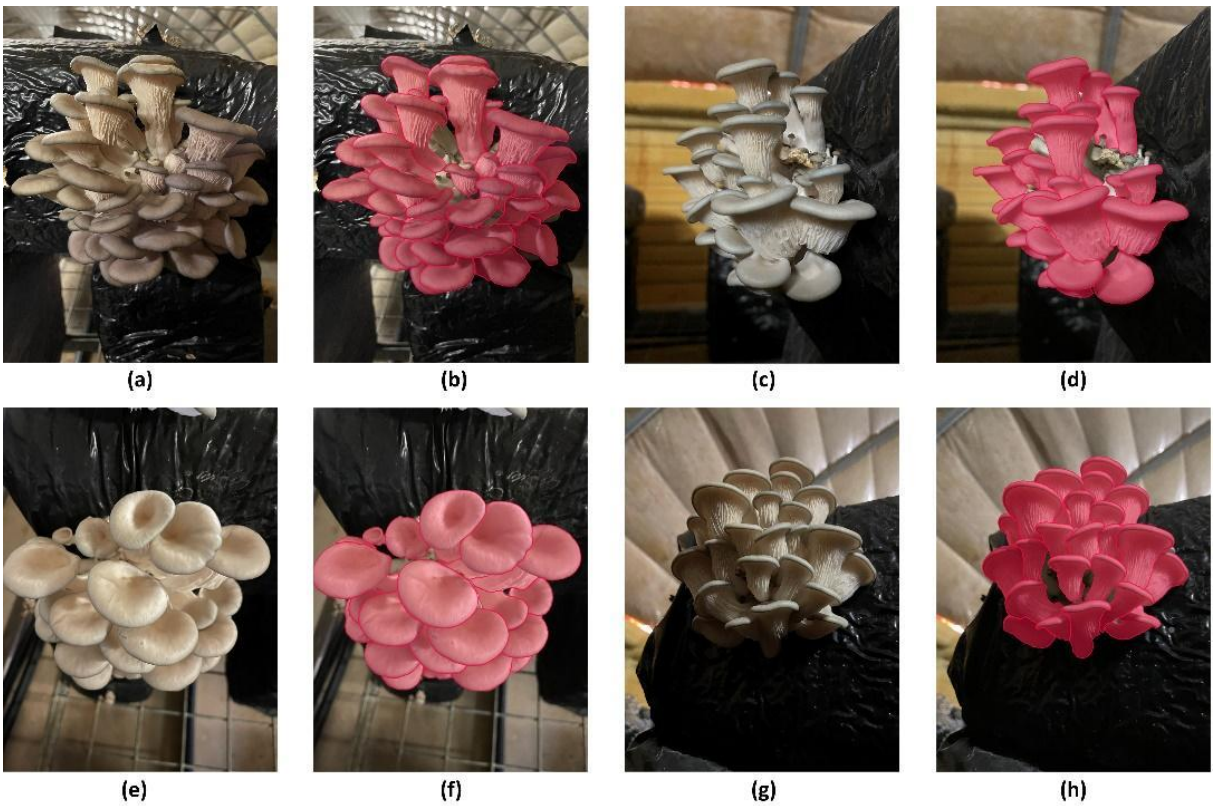

Figure 6. Mushroom clusters captured from front (a), side (c), top (e), and bottom (g) views and their respective annotated masks (b, d, f, h)

Out of the entire set of 200 images, a total of 3,944 instances of mushrooms were accurately labeled. This indicates an average of approximately 20 instances of mushrooms per image. As the purpose of this work is the identification of individual mushrooms inside a cluster, the 3,944 instances can be regarded as the training inputs rather than the initial 200 images.

In a similar way, the mushroom clusters of the dynamic dataset were annotated using the same annotation tool. In Figure 7, the annotations of some samples extracted from a dynamic dataset are presented.

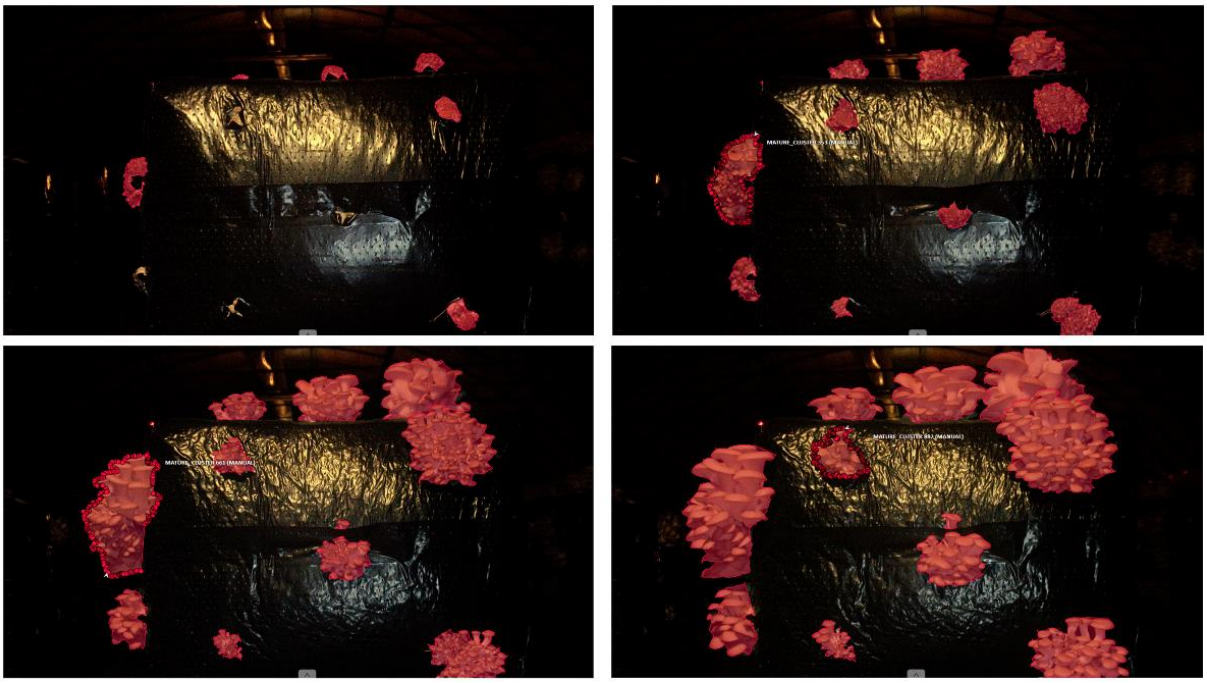

Figure 7. Annotated mushroom clusters on images obtained from the dynamic timelapse dataset.

About 330 images were manually annotated and used for training the deep learning (DL) models. In total, 3151 mushroom cluster instances were manually annotated, many of them corresponding to the same mushroom cluster but in different growth stages.

5. Instance segmentation architecture of Mask R-CNN

The state-of-the-art instance segmentation architecture of Mask R-CNN [3] was selected, trained, and evaluated using the mushrooms dataset. This architecture is widely recognized as a major breakthrough in the field of DL instance segmentation. It builds upon the object detection architecture of Faster R-CNN [4] by incorporating a fully convolutional network (FCN) specifically designed for semantic segmentation, as depicted in the Mask Head component shown in Figure 8. To the best of our knowledge, very few studies have been conducted on oyster mushrooms, utilizing DL instance segmentation architecture. In a relevant study by Moysiadis et al. [5], the object detection YOLOv5 [6] architecture was used to detect and classify oyster mushrooms into three growth stages. Then, a Mask R-CNN architecture was trained to produce masks of mushroom clusters and monitor their growth by calculating the change of pixels per instance. The instance segmentation Mask R-CNN model was applied to images collected from the top of the

substrate, limiting the visible parts of the clusters, while no information on the number of mushroom instances present in the cluster was extracted. Moreover, no environmental data were collected and considered throughout the study. On the contrary, the MUSHNOMICS project collects two kinds of mushroom data static clusters and dynamic clusters, the former from multiple angles and the latter from the front, which capture clusters from a greater variety of angles than only the top as it can be seen in Figure 6. Finally, environmental variables of temperature, humidity, and CO_2 are correlated with the actual growth of the mushrooms, providing valuable insight into the effects of climatic conditions inside the mushroom farm on the growing clusters.

Various studies have been conducted using Mask R-CNN with different target crops than mushrooms. Ge et al. [7] used Mask R-CNN to detect strawberries and classify them into four classes, three for their ripeness state and one for existent deformations. Aiming at growth monitoring of lettuce, Lu et al. [8] deployed a Mask R-CNN model for lettuce instances detection inside a greenhouse for 20 days using time-lapse images. Lee et al. [9] explored yield monitoring in tomatoes, with the use of a pre-trained Mask R-CNN model, examining a correlation between detected fruit dimensions and actual mass. Ganesh et al. [10] used a Mask R-CNN combining RGB and HSV images as input datasets for the detection and localization of oranges.

With a deeper examination of the Mask R-CNN architecture, it can be categorized as a two-stage detector. The first stage is used to identify areas of interest (ROIs) inside the image that may potentially contain instances of interest, whilst the second stage is responsible for generating the final predictions of class, bounding box coordinates, and detailed pixel-wise instance masks within each ROI. In general, the architectural design of Mask R-CNN may be categorized into three components if the feature extractor component is regarded as a separate entity and not part of the first stage. This architecture is illustrated visually in Figure 8.

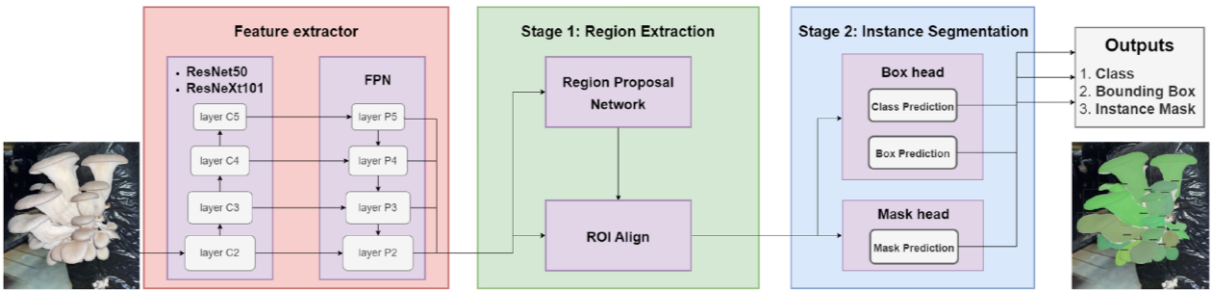

Figure 8. Mask R-CNN architecture, divided into distinct parts (image from Charisis et al. [11]).

The initial component of the network, referred to as the backbone, functions as a mechanism to extract features from the input image. This is achieved by using convolutional neural networks (CNNs) in a fully convolutional network (FCN) architecture. In the case of Mask R-CNN, the backbone consists of a network composed of ResNet [12] and Feature Pyramid Network (FPN) [13], which is depicted in Figure 9. The ResNet architecture has demonstrated its efficacy in addressing the issue of gradient vanishing that often arises in deeper architectures containing multiple layers. This important characteristic enables the efficient utilization of multi-layer networks, such as ResNet50 and ResNet101. Furthermore, the ResNet layout follows a

bottom-up approach, wherein it is capable of detecting both low-level characteristics in the initial layers and high-level features in the last layers. The integration of a top-down architecture is achieved with the introduction of the FPN, allowing for the effective combination of low- and high-level features in a multi-scale feature fusion within the backbone.

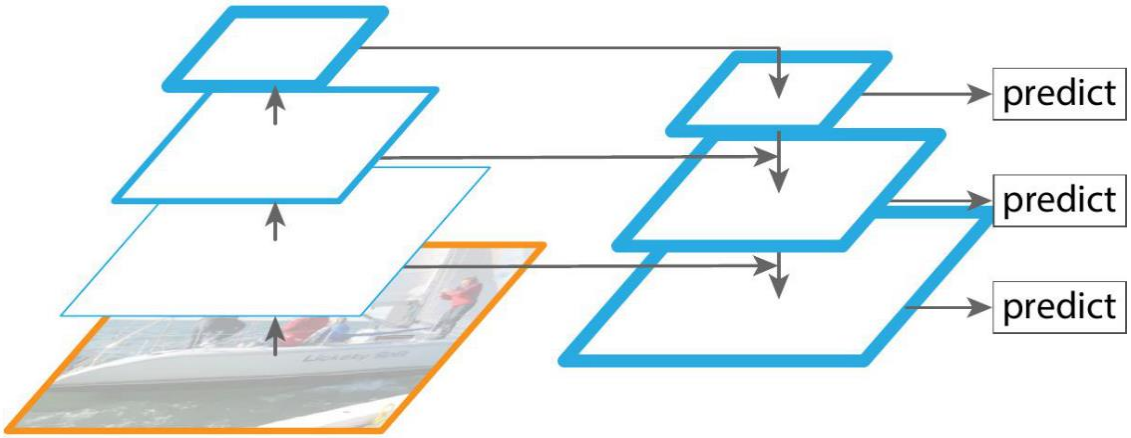

Figure 9. FPN architecture, divided into distinct parts. The left part of the architecture is a backbone network (usually ResNet or similar implementation) and the right part is the FPN addition. Depending on the application one or more of the "predict" stages can be utilized as a feature from the following components of an architecture (image from [13]).

The feature map generated by the backbone network is utilized as the input for the next network components, namely the Region Proposal Network (RPN) and Region of Interest (ROI) Align. The task of RPN involves the identification of instances inside an image, by utilizing the feature map and a set of predefined boxes, known as anchors, which have a certain size and scale. The anchors are iteratively passed through the feature map and each position is evaluated to ascertain whether a potential object is present or absent by comparing it to the ground truth data. The result derived from the Reverse Polish Notation (RPN) contains a collection of anchors and two additional attributes: the anchor's classification (foreground or background) and the coordinates of the anchor's bounding box. To alleviate the computational cost associated with analyzing all anchors, a selection process is conducted that prioritizes the identification of anchors with the highest level of confidence in containing an object. The task of ROI Align involves the extraction of features from the feature map generated by the backbone. These features correspond to the proposed areas identified by the RPN. The extracted features are then converted into arrays of fixed sizes to enable standardized processing in the next and final part of the network.

The fix-sized arrays resulting from the processing of the preceding components are next fed into two separate branches. One branch generates the final class prediction and bounding box coordinates (referred to as the box head branch), while the other branch produces the pixel-wise mask of the detected instance (known as the mask head branch). The class and bounding box coordinates are generated by the utilization of fully connected networks for classification and regression, respectively, whereas, for the instance mask, a small CNN component is implemented.

6. Performance evaluation metrics

The detection performance was measured quantitatively using metrics such as precision (P) and recall (R). The detection results were divided into four categories: true positive (TP), false positive (FP), true negative (TN), and false negative (FN). In this context, TP stands for the correct detection of oyster mushrooms, FP for the incorrect detection of oyster mushrooms, and FN for the number of raw oyster mushrooms that were not recognized within the cluster. Precision (P) and recall (R) are defined as follows:

$$Precision = \frac{TP}{TP + FP} \quad (1)$$

$$Recall = \frac{TP}{TP + FN} \quad (2)$$

To calculate the performance metrics, equation 3 defines the Intersection over Union (IoU). The IoU score measures the overlap between a predicted mask and the ground truth mask. It is a value that falls between 0.0 and 1.0, with a prediction regarded as a solid result if IoU is greater than 0. Figure 10 shows how these performance metrics were computed, the Intersection over Union (IoU) was defined in a mushrooms cluster.

$$IoU = \frac{Area\ of\ Inersection}{Area\ of\ Union} \quad (3)$$

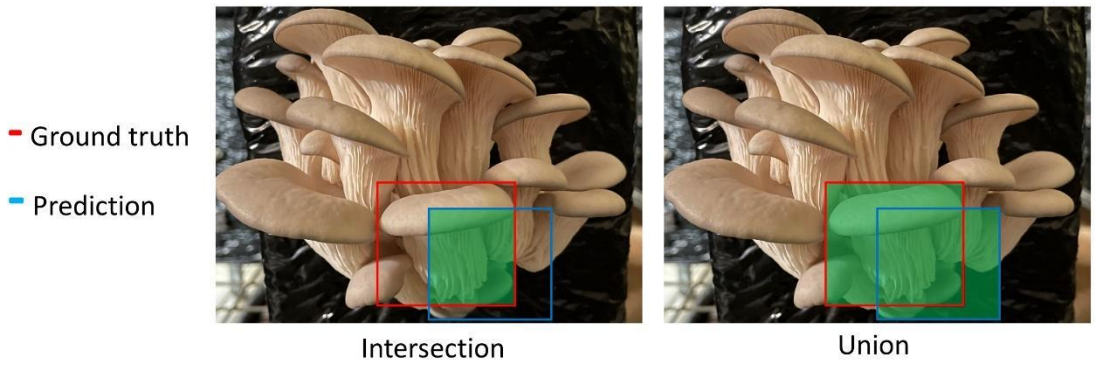

Figure 10. IoU visualization on a single oyster mushroom instance in a cluster (image from Charisis et al. [11]).

Based on the IoU value, each prediction will be classified as a TP or correct prediction if the value exceeds a specified threshold or as FP if the value falls below thresholds. The accuracy of the prediction made from the model increases as the value of these two metrics increases.

The performance was evaluated using standard COCO metrics, by calculating the mean average precision (mAP) which determines the average precision for each class, namely in this case, the mushroom class. This is done across 10 IoU thresholds, ranging from 0.5 to 0.95 with an increment of 0.05. The mAP is computed from the mean of these class-specific average precision values.

7. MUSHNOMICS timelapse tracking algorithm

The timelapse application aims to track the growth of mushroom clusters in a series of time-lapse images. This aim was achieved by first capturing static images containing single or multiple mushroom clusters. As described earlier, these images were manually annotated to track the outlines of whole clusters and single mushrooms with the annotated images being used for the training and validation of Mask-RCNN neural networks which can be used to automatically recognize whole clusters and single mushrooms in images.

The developed tracking algorithm has enabled the detailed monitoring of the cluster growth throughout the cultivation process. As the timelapse image data consists of multiple successive images, a cluster tracking algorithm is required for the clusters to be matched with themselves in previous timesteps. The algorithm consists of three main array data structures that keep track of the clusters' center coordinates that have already been found and the center coordinates of the new cluster detections. The comparison of the center coordinates is a solid indicator to determine if a cluster is the same over consecutive images.

The first array structure is the "baseline", in which the information regarding every detected cluster is stored and acts as a reference for new instances or instances that were detected earlier but in the current image are not. It is updated every timestep to include new detections and update the positions of the existing clusters. The second array, named "previous" structure contains the information on the center coordinates of clusters of the previous timestep. The third array, named "current" contains the information on the center coordinates of clusters detected in the current timestep.

The "current" array is first compared to the "previous" array to match the detected clusters. It is not expected that the center coordinates of a cluster exhibit great differences between two successive timesteps. Based on this behavior, most detected clusters correlate in this first comparison. If not all of the clusters in "current" are correlated with clusters in the "previous" array, it is due to a new cluster that has been detected and must be recorded. If a previously detected cluster is not detected in the current timestep, its information is passed on from the old "baseline" array into the updated "baseline" array.

The image files and their information are imported into the script where they are saved in separate arrays. The cluster detector Mask R-CNN model is then applied to each of the images and a list of results for each recognized mushroom cluster in the image is produced. These results are filtered based on the confidence of the model that the detected object is a cluster and not background, with only clusters of confidence greater than 90% being considered. The detected mushrooms are in binary mask format, where pixels in the image that were recognized by the neural network as "part of a cluster" are saved as "True" whereas the other "non-cluster" pixels are saved as "False".

The results of the tracking algorithm can be seen in Figure 14, where 2 representative timesteps are presented, with the corresponding numbers of the detected clusters at their center.

8. Performance of MUSHNOMICS algorithms

In Table 1 the performance metrics are presented for the models developed either by using the static or dynamic-timelapse image datasets. A better understanding of the performance of each model can be supported by observing the detection results produced by the Mask R-CNN based MUSHNOMICS algorithms. Hence, some representative prediction results are presented in the following Figures for both static (Figures 11, 12) and dynamic (Figure 13) image datasets, allowing for a visual interpretation of the performance of the trained Mask R-CNN models.

Table 1. Performance evaluation metrics for both Mask R-CNN trained models.

	mAP	Recall
Static image dataset	0.707	0.755
Timelapse image dataset	0.766	0.786

The timelapse model exhibits better performance as its task is relatively easier than the static model, which has to tackle with the difficult problem of occluded mushroom instances.

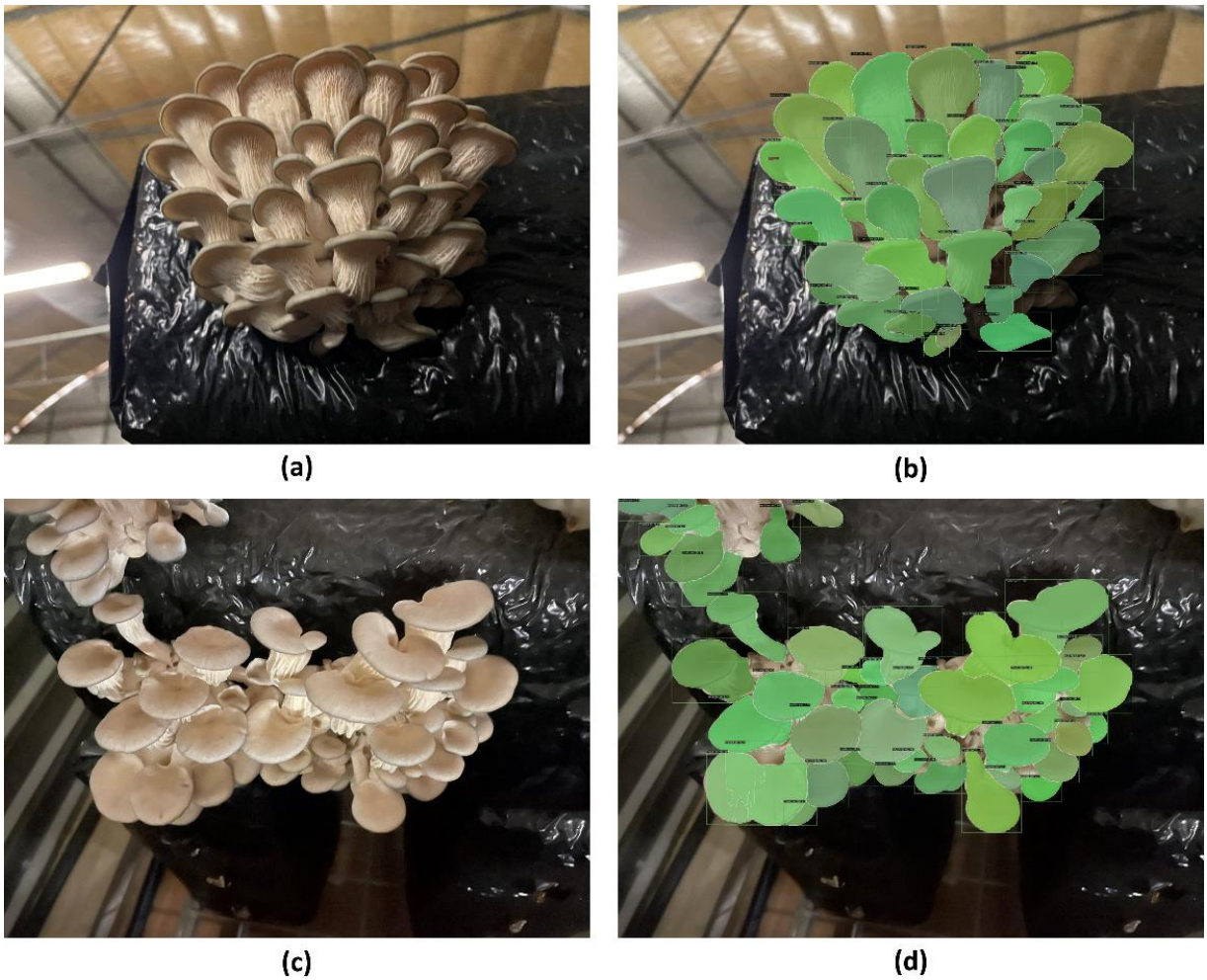

Figure 11. Mask R-CNN based detection of single mushrooms in a cluster on image datasets taken from different angles, bottom (a, b) and top (c, d).

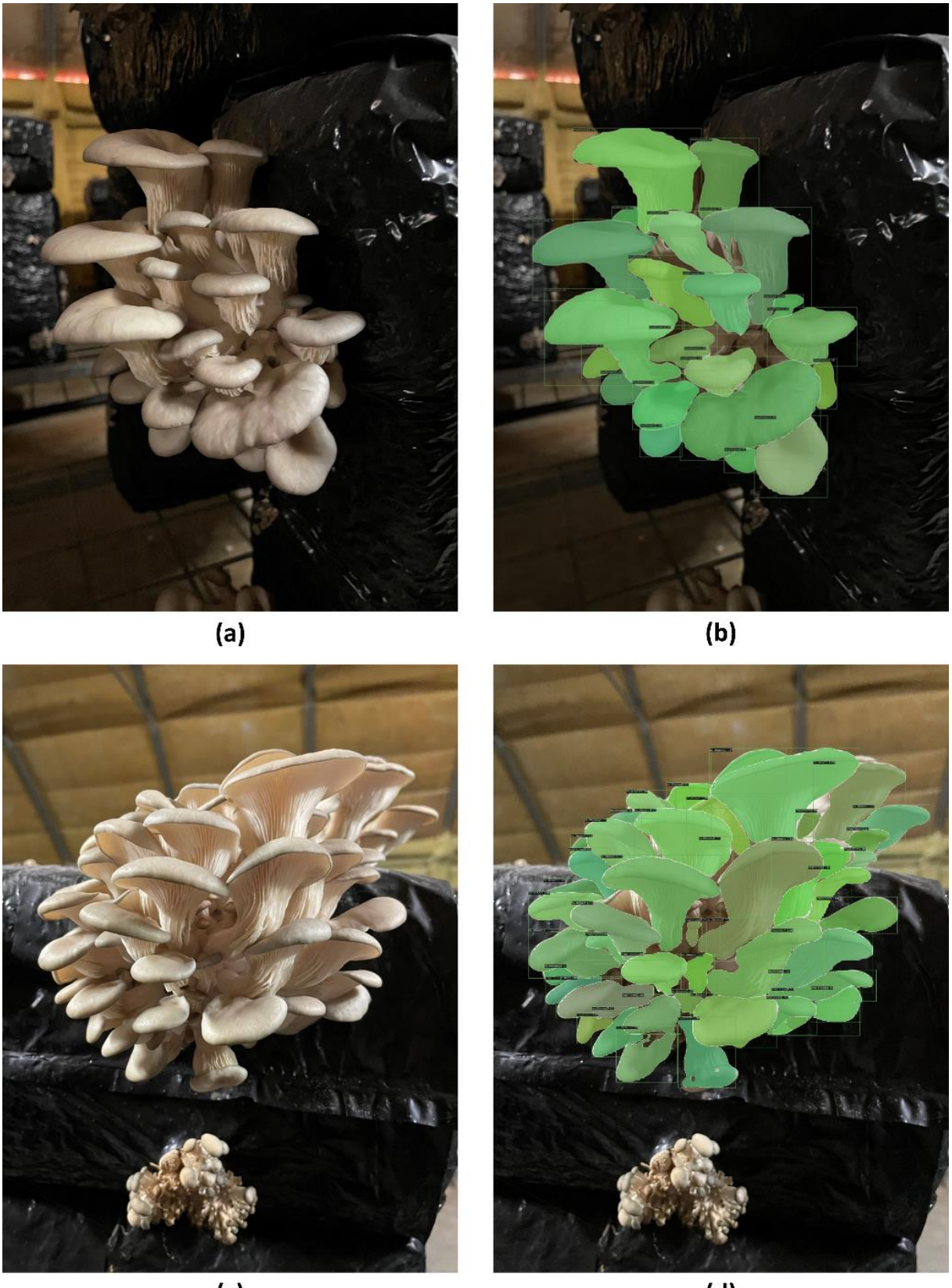

Figure 12. Mask R-CNN based detection of single mushrooms in a cluster on image datasets taken from different angles, side (a, b) and front (c, d).

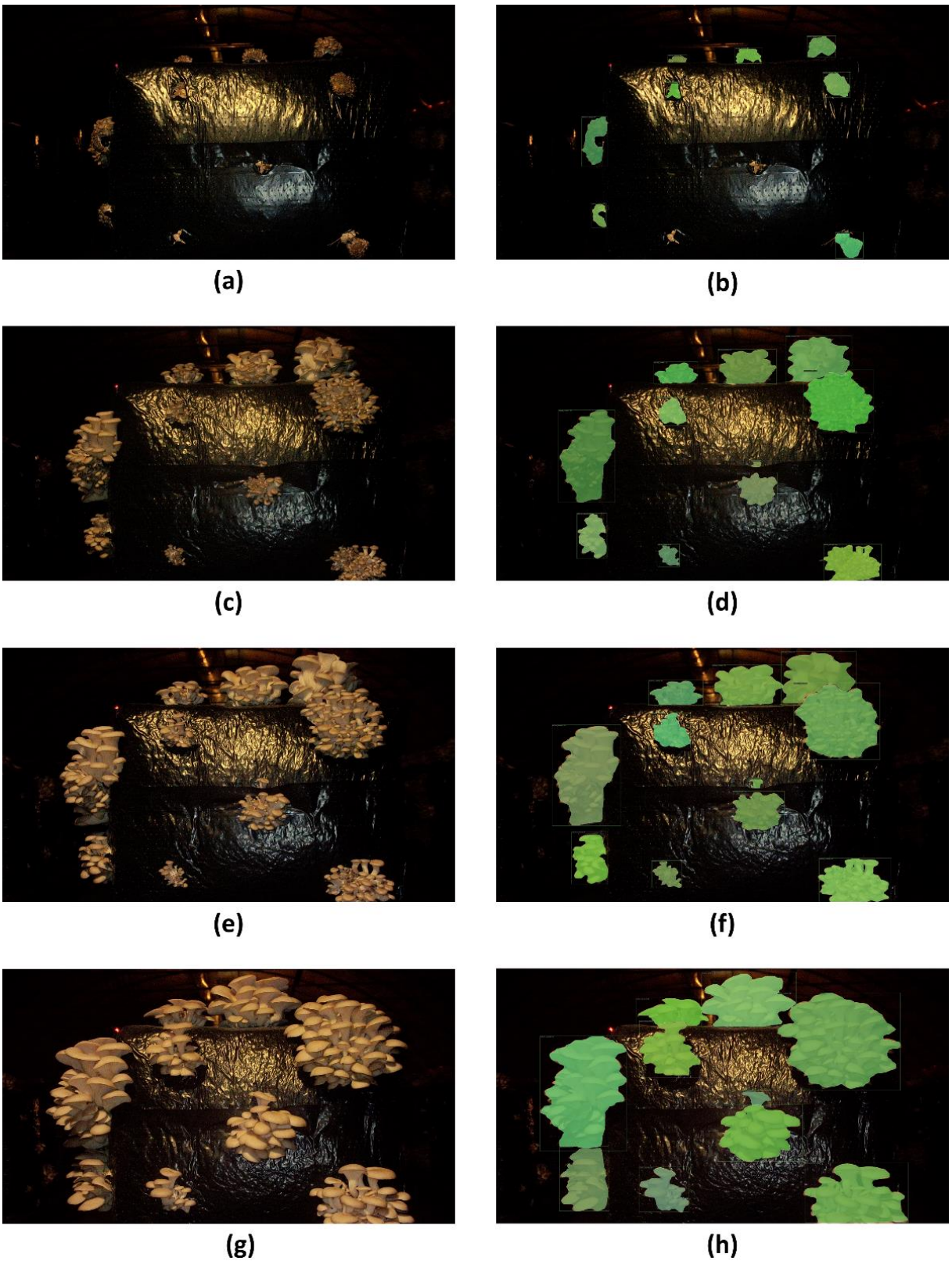

Figure 13. Mask R-CNN based detection of mushroom clusters on image datasets acquired in a time lapse manner.

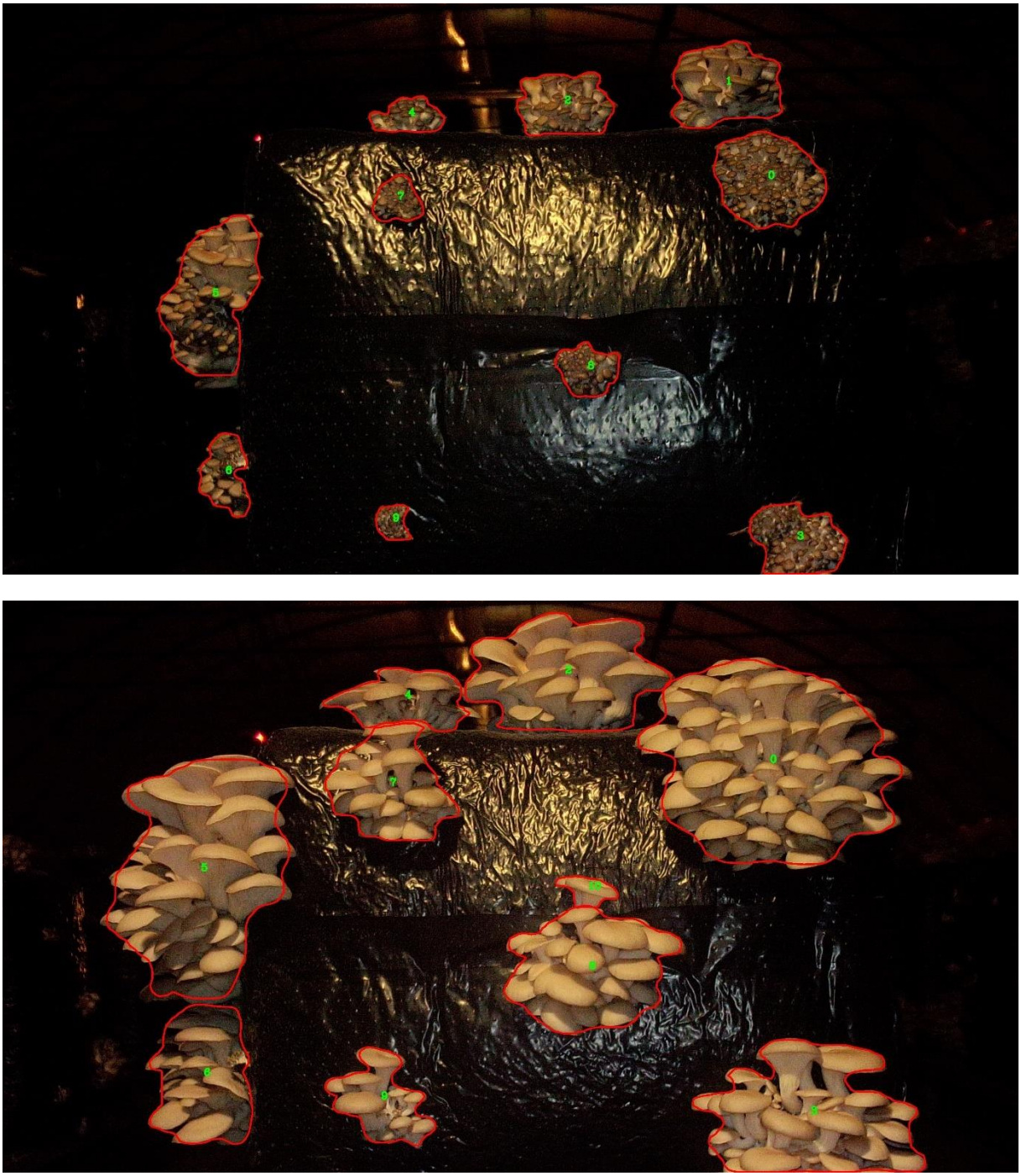

Figure 14. MUSHNOMICS tracking algorithm at selected timesteps of mushroom growth.

Each mushroom cluster is marked with a number as shown in Figure 14. The images are saved with their recognized clusters being outlined on the image and numbered according to their position in the baseline array. This is carried out to visualize the previous steps and validate their functions. Figure 15 exemplarily shows an image being sorted with its reference array and pixel coordinates (to track the changes in the array) together with the growth tracking curves of eleven mushroom clusters grown on a wheat straw substrate block. After the marked images and the cluster images are saved the tracking of growth can be carried out. The size of each cluster is calculated in terms of pixel number to track the growth of each mushroom cluster through the

different images. The growth rate of the clusters is presented in relative terms and serves as a proof of concept for tracking mushroom growth digitally. In addition, the clusters are categorized as small, medium and large providing an overview of how suitable each cluster is for harvest based on its relative size.

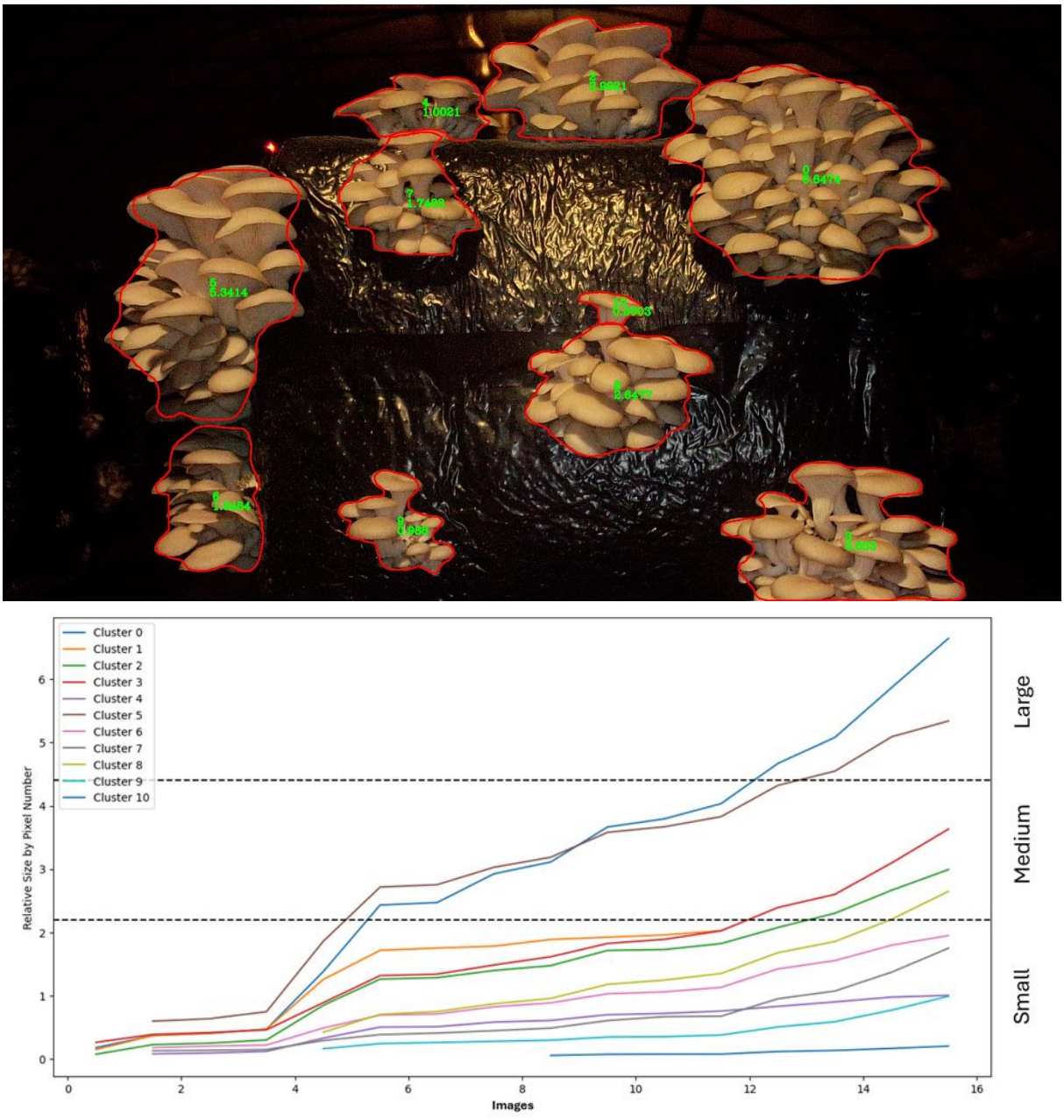

Figure 15. Mushroom cluster tracking predictions and pixel-wise relative growth index.

The deep instance segmentation models based on Mask R-CNN presented here, constitute the prototype MUSHNOMICS algorithms and are utilized by the MUSHNOMICS platform as supportive modules for researchers and stakeholder users. The MUSHNOMICS algorithms were then validated using a new dataset acquired from the production trials carried out in the MUSHNOMICS module (T3.3) using the novel MUSHNOMICS substrates.

References

[1] Wingscapes TIMELAPSECAM pro [WWW Document], n.d. . Wingscapes. URL https://www.wingscapes.com/timelapsecam-pro (accessed 1.16.24).

[2] Cvat [WWW Document], n.d. . CVAT. URL https://www.cvat.ai/ (accessed 1.16.24).

[3] He, K., Gkioxari, G., Dollar, P., Girshick, R., 2017. Mask R-CNN. 2017 IEEE International Conference on Computer Vision (ICCV). doi:10.1109/iccv.2017.322

[4] Ren, S., He, K., Girshick, R., Sun, J., 2017. Faster R-CNN: Towards real-time object detection with region proposal networks. IEEE Transactions on Pattern Analysis and Machine Intelligence 39, 1137–1149. doi:10.1109/tpami.2016.2577031

[5] Moysiadis, V., Kokkonis, G., Bibi, S., Moscholios, I., Maropoulos, N., Sarigiannidis, P., 2023. Monitoring mushroom growth with machine learning. Agriculture 13, 223. doi:10.3390/agriculture13010223

[6] Redmon, J., Divvala, S., Girshick, R., Farhadi, A., 2016. You only look once: Unified, real-time object detection. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). doi:10.1109/cvpr.2016.91

[7] Ge, Y., Xiong, Y., From, P.J., 2019. Instance segmentation and localization of strawberries in farm conditions for automatic fruit harvesting. IFAC-PapersOnLine 52, 294–299. doi:10.1016/j.ifacol.2019.12.537

[8] Lu, J.-Y., Chang, C.-L., Kuo, Y.-F., 2019. Monitoring growth rate of lettuce using deep convolutional neural networks. 2019 Boston, Massachusetts July 7- July 10, 2019. doi:10.13031/aim.201900341

[9] Lee, J., Nazki, H., Baek, J., Hong, Y., Lee, M., 2020. Artificial intelligence approach for Tomato Detection and mass estimation in Precision Agriculture. Sustainability 12, 9138. doi:10.3390/su12219138

[10] Ganesh, P., Volle, K., Burks, T.F., Mehta, S.S., 2019. Deep orange: Mask R-CNN based Orange Detection and segmentation. IFAC-PapersOnLine 52, 70–75. doi:10.1016/j.ifacol.2019.12.499

[11] Charisis, C., Gyalai-Korpos, M., Nagy, A.S., Karantzalos, K., Argyropoulos, D., 2023. 48. detecting and locating mushroom clusters by a mask R-CNN model in farm environment. Precision agriculture '23. doi:10.3920/978-90-8686-947-3_48

[12] He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). doi:10.1109/cvpr.2016.90

[13] Lin, T.-Y., Dollar, P., Girshick, R., He, K., Hariharan, B., Belongie, S., 2017. Feature Pyramid Networks for Object Detection. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). doi:10.1109/cvpr.2017.106