Contributions to computer vision and machine learning for plant variety testing

Mouad Zine El Abidine

UNIVERSITE BRETAGNE LOIRE / MATHSTIC

université angers

THÈSE DE DOCTORAT DE

L'UNIVERSITÉ D'ANGERS
COMUE UNIVERSITÉ BRETAGNE LOIRE

ÉCOLE DOCTORALE N° 601
Mathématiques et Sciences et Technologies de l'Information et de la Communication
Spécialité : *Signal, Image, Vision*

Par

«Mouad Zine El Abidine»

«Contributions to computer vision and machine learning for plant variety testing»

Thèse présentée et soutenue à « INRAE - Angers», le « 04/10/2022 »
«LARIS - Laboratoire Angevin de Recherche en Ingénierie des Systèmes»

Rapporteurs avant soutenance :

Pr. Jean-Pierre DA COSTA, IMS, Bordeaux, France
Pr. Adel AFIANE, INSA Val de Loire, France

Composition du Jury :

Président :	
Examinateur :	Pr. Manuela ZUDE-SASSE, Postdam University, Germany.
Examinateur :	Pr. Gerhard BUCK SORLIN, Institut Agro, Rennes-Angers, France.
Examinateur :	Dr. Maria-José ARANZANA, IRTA, Catalunya, Spain.
Dir. de thèse :	Pr. David ROUSSEAU, Université d'Angers, France.
Co-encadrants :	Dr. Pejman RASTI, Université d'Angers, France.
	Dr. Helin DUTAGACI, Eskisehir Osmangazi University, Eskisehir, Turkey.

Invité(s) :

Dr. François LAURENS, INRAe, Angers, France.

Acknowledgement

First, I would like to express my sincere gratitude to my mentor and my supervisor, Pr. David Rousseau. During my journey with him, I learned to control my energy, analyze carefully, improvise "when it's needed" and believe in myself. He was very patient, supportive and enthusiastic. It was truly an honor to be one of his students.

I would like to thank also my supervisor, Dr. Helin Dutagaci, for her continuous support. She is a very wise person, and we shared unforgettable moments, especially during the data acquisition in the orchards while eating the apples.

Thanks also go to my supervisor, Dr. Pejman Rasti. He supported me and taught me a lot. Aside from the professional relationship, Pejman is a great friend, a very experienced person who always has invaluable advice.

I would like to extend my deepest appreciation to my committee members: Pr. Manuela Zude-Sasse, Dr. Maria Aranzana, Pr. Gerhard Buck-Sorlin, Pr. Jean-Pierre Da Costa and Pr. Adel Afiane, who gently accepted to evaluate my work and attend my Ph.D. defense.

I also had the great pleasure of working with François Laurens, who inspired me a lot with his charismatic personality and management of projects, and with Roland Robic, for putting his heart into achieving the tasks.

Thanks should go to all my friends and colleagues in INRAe, Polytech Angers and especially the ImHorPhen team, including Hadhami Garboug, who shared with me the office for three years and Natalia Sapoukhina for her constant advice and the fruitful discussions.

I would like to express my sincere gratitude to my family, including my sister Fatima Zohra and her adorable daughter Rhita and my cousin Hajar El Hamdouchi, for her encouragement, which never let me down.

The special thanks go to my parents, for believing in me and giving me the support and love to fulfill my dreams. I dedicate my Ph.D. to my father, Pr. Abdenbi Zine El Abidine, for his unwavering efforts. You are the best dad in the universe.

Last and foremost, I would like to thank "Allah" for blessing me and helping me to choose the right path.

NOMENCLATURE

$(\Delta_x, \Delta_y, \Delta_z)$ Edge lengths of the voxels for voxelization of a point cloud

$(\hat{x}, \hat{y}, \hat{z})$ Coordinates of a 3D point $\hat{p}$

(A, B, C, D) Parameters of the trellis-plane

(I_s, J_s) Location of the s^{th} detected peak in IG

(N_x, N_y, N_z) Size of B and S

(x, y, z) Coordinates of a 3D point p

$(x_{r,1}, y_{r,1}, z_{r,1})$ Coordinates of the point $p_{r,1}$

$(x_{r,2}, y_{r,2}, z_{r,2})$ Coordinates of the point $p_{r,2}$

Γ Set of semantic labels

γ_i Predicted semantic label of the i^{th} point p_i

γ_i^{GT} Ground truth semantic label of the i^{th} point p_i

$\hat{L}_j = (0, \hat{y}_j, 0)$ Location of verified trunk j

$\hat{L}_s = (0, \hat{y}_s^{ct}, 0)$ Location of candidate trunk s

$\hat{p}$ A 3D point in PC_w^{TP}

$\hat{p}_e$ e^{th} end-point of C_c

$\hat{p}_{q,j}$ Intersection point of q^{th} trellis-line and j^{th} trunk

$\hat{p}_{s,bottom}$ Bottom point of SK_s along the Z-axis

$\hat{p}_{s,top}$ Top point of SK_s along the Z-axis

$\hat{y}_s^{ct}$ y coordinate of the s^{th} candidate trunk location in $\mathcal{CT}$

$\hat{z}_q$ Height of tl_q

$\mathcal{A}^{\mathcal{GT}}$ Set of ground truth apples

$\mathcal{A}$ Set of detected apples

$\mathcal{CC}$ Set of connected components of S_{trees}

$\mathcal{CT}$ Set of candidate trunk locations

$\mathcal{IG}_{i,j}$ Set of points in PC_{ts} projected to the ground in the grid (i, j)

$\mathcal{L}_{HL}$ Set of 3D horizontal lines detected through applying Hough Transform to IH

$\mathcal{L}_{TL}$ Set of trellis-lines

$\mathcal{T}$ Set of located trees in the scene

τ_a Predicted tree identity of the a^{th} apple in $\mathcal{A}$

τ_c Predicted tree identity of the c^{th} connected component C_c

τ_g Ground truth tree identity of the g^{th} apple in $\mathcal{A}^{GT}$

τ_i Predicted tree identity of the i^{th} point p_i

τ_i^{GT} Ground truth tree identity of the i^{th} point p_i

$\{(C_f, \tau_f)\}$ Connected components already assigned to a tree

$\{C_{c,d}\}$ Set of connected components of C_c after cutpoints are removed

ACC Accuracy of apple assignment to trees

B Binary 3D volumetric form of PC_w^C

B_s Binary 3D volumetric form of PC_s^{CT}

B_{trees} Binary 3D volumetric form of PC_w^{trees}

BVP Best View Projection

C_c c^{th} connected component in $\mathcal{CC}$

CA Class Accuracy

CNN Convolutional Neural Network

CP Connecting path between adjacent trees

$CTIFL$ Centre Technique Interprofessionnel des Fruits et Légumes

$d(p, hl_r)$ Distance between point p and the line hl_r

d_R^{cc} The minimum distance of the ColorChecker tripod stick to the tree row

d_s^{SP} Length of main axis SP_s

d_T^{cc} The distance of the ColorChecker tripod stick to a designated tree

DUS Distinctness, Uniformity And Stability

$ECPGR$ European Cooperative Programme For Plant Genetic Resources

EO Examination Offices

$F1$ F1 score

FN Number of false negatives

FP Number of false positives

GAN Generative Adversarial Networks

$GEVES$ Groupe d'Etude et de contrôle des Variétés Et des Semences

hl_r r^{th} horizontal line in $\mathcal{L}_{HL}$

IG Histogram of points in PC_{ts} projected to the ground

IH Binary image resulting from projecting S to the YZ-plane

$INRAe$ Institut National De Recherche Pour L'agriculture, L'alimentation Et L'environnement

$INVITE$ INnovations In Plant VarIety Testing in Europe

IoU Intersection over Union

$IPPN$ International Plant Phenotyping Network

ITB Institute Technique De La Betterave

l_e Line fitted to the points around p_e

L_j Location of j^{th} tree in the scene corresponding to (x_j, y_j, z_j) coordinates of the base of the tree

LDA Linear Discriminant Analysis

$ls_{j,j+1}$ The line defined by the points $\hat{p}_{q,j}$ and $\hat{p}_{q,j+1}$

MDS Multidimensional Scaling

N_e Number of end-points of C_c

N_F Number of connected components already assigned to a tree

N_P Number of detected peaks in IG and number of canditate trunk locations in $\mathcal{CT}$

N_W Number of points in PC_w^C

N_{apples} Number of apples in $\mathcal{A}$

N_{apples}^{GT} Number of apples in $\mathcal{A}^{\mathcal{GT}}$

N_{comp} Number of connected components in $\mathcal{CC}$

N_{comp}^c Number of connected components of C_c after cutpoints are removed

N_c Number of trees spanned by a connected component C_c

N_{HL} Number of horizontal lines in $\mathcal{L}_{HL}$

N_{TL} Number of trellis-lines in $\mathcal{L}_{TL}$

n_{TP} Unit normal of the trellis-plane

N_{trees} Number of verified trees in the scene

N_{trees}^{GT} Number of ground truth trees

OT Optimal Transport

p A 3D point

p_a^α Position of the a^{th} apple in $\mathcal{A}$

$p_g^{\alpha,GT}$ Position of the g^{th} ground truth apple in $\mathcal{A}^{\mathcal{GT}}$

p_i i^{th} point in PC_w^C

$p_{r,1}$ A point on hl_r

$p_{r,2}$ A point on hl_r

PC A 3D colour point cloud

PC_h Reconstructed harvest point cloud before calibration

PC_h^C Calibrated harvest point cloud

PC_s^{CT} Subset of PC_w^{TP}; set of points within a cylindrical region around the s^{th} candidate trunk location

PC_w Reconstructed winter point cloud before calibration

PC_w^C Calibrated winter point cloud

PC_w^{TP} Winter point cloud aligned to the trellis-plane

$PC_{j,j+1}^q$ Cylindrical region around the q^{th} trellis-line between the points $\hat{p}_{q,j}$ and $\hat{p}_{q,j+1}$

PC_{tr} Subset of PC_w^C; set of points within 1cm distance to the horizontal lines on the trellis-plane

PC_{ts} Subset of PC_w^{TP}; set of points within 5cm distance to the trellis-plane

PC_{wh}^C Winter point cloud aligned to PC_h^C

PC_w^{trees} Subset of PC_w^{TP}; set of points with trellis wires and support pole removed

PCA Principal Component Analysis

Pr Precision

R Rotation matrix to transform PC_w^C to PC_w^{TP}

$R-CNN$ Regions-Convolutional Neural Network

R^{wh} Rotation matrix to align PC_w^C to PC_h^C

Re Recall

S Skeleton of B

S_s Skeleton of B_s

S_{trees} Skeleton of B_{trees}

$SBIR$ Sketch-Based Image Retrieval

SDR Sufficient Dimension Reductions

SK_s Set of points on the skeleton of s^{th} trunk candidate

SP_j Set of points on the main axis of j^{th} verified trunk

SP_s Set of points on the main axis of s^{th} trunk candidate

SVM Support Vector Machine

T^{wh} Translation vector to align PC_w^C to PC_h^C

T_j j^{th} tree in $\mathcal{T}$

t_j Identity of j^{th} tree in $\mathcal{T}$

tl_q q^{th} trellis-line in $\mathcal{L}_{TL}$

TN Number of true negatives

TP Number of true positives

TP_C Number of true positive apples correctly assigned to respective trees

u_X X-axis of the reference frame aligned to the trellis-plane

u_Y Y-axis of the reference frame aligned to the trellis-plane

u_Z Z-axis of the reference frame aligned to the trellis-plane

$UPOV$ International Union for the Protection of New Varieties of Plants

$VaDiPom$ Pip Fruit Diversity Valorization

VCU Value For Cultivation And Use

x_{max} Maximum of the x coordinates of the points in a point cloud

x_{min} Minimum of the x coordinates of the points in a point cloud

y_{max} Maximum of the y coordinates of the points in a point cloud

y_{min} Minimum of the y coordinates of the points in a point cloud

z_{max} Maximum of the z coordinates of the points in a point cloud

z_{min} Minimum of the z coordinates of the points in a point cloud

HSV Hue, Saturation, and Value components of the colour of a 3D point

RGB Red, Green, and Blue channels of the colour of a 3D point

Table of Contents

1 Introduction **25**

1.1 Problematic . . . 25

1.2 Type of tests in variety testing . . . 27

1.2.1 Distinctness, uniformity and stability (DUS) . . . 27

1.2.2 Value for cultivation and use (VCU) . . . 28

1.3 Why variety testing needs specific machine learning methodologies and computer vision tools. . . . 28

1.4 Our contributions . . . 30

1.5 Structure of this document . . . 33

2 Computer vision for variety testing orchards **35**

2.1 Introduction . . . 35

2.2 Materials and methods . . . 38

2.2.1 Experimental field . . . 39

2.2.2 Data acquisition and 3D reconstruction . . . 40

2.2.3 Calibration and extraction of region of interest . . . 43

2.2.4 Separation of individual trees . . . 43

2.2.5 Apple detection . . . 57

2.2.6 Assigning apples to individual trees . . . 57

2.2.7 Ground truth and evaluation metrics . . . 58

2.3 Results . . . 61

2.3.1 Evaluation of detection of trellis wires, tree trunks and support poles 61

2.3.2 Evaluation of apple detection and assignment to individual trees . . 66

2.3.3 Processing time . . . 68

2.4 Discussion . . . 68

2.5 Conclusion and perspectives . . . 73

3 Toward the use of machine learning in variety testing **75**

3.1 Introduction . . . 75

3.2 Apple color characterization . . . 76
3.2.1 Materials and methods . . . 77
3.2.2 Color features . . . 81
3.2.3 Classification setups . . . 84
3.2.4 Classification results . . . 84
3.2.5 Multi-class classification between non-gala mutant varieties . . . 85
3.2.6 Conclusion . . . 89
3.3 Apple shape charaterisation . . . 92
3.3.1 Materials and methods . . . 92
3.3.2 Results . . . 97
3.3.3 Reference-based classification approach . . . 97
3.3.4 Discussion . . . 98
3.3.5 Conclusion . . . 100
3.4 Conclusion and perspective . . . 101

4 Processing ordinal data in variety testing 103
4.1 Introduction . . . 103
4.2 Method . . . 106
4.3 Results on dimension reduction . . . 109
4.3.1 Simulated data . . . 109
4.3.2 Real ordinal data . . . 111
4.4 Ordinality metrics . . . 113
4.4.1 Deviation from ordinality metric . . . 113
4.4.2 Inter-class intersection metric . . . 114
4.5 Results on ordinality metrics . . . 115
4.6 Application to variety testing . . . 117
4.7 Conclusion . . . 119

5 Conclusion and Perspectives 121
5.1 Synthetic view of methodological contributions . . . 121
5.2 Engineering contributions . . . 122
5.3 Discussion . . . 123
5.4 Publications . . . 125

Bibliography 126

6 Annex A **127**
6.1 Annotated datasets and machine learning models . . . 127
6.1.1 Images of apple trees in harvest period . . . 127
6.1.2 3D point cloud of apple trees in harvest period . . . 128
6.1.3 3D point cloud of apple trees in winter period . . . 128
6.1.4 Images of apple trees in flowering period . . . 129
6.1.5 Images of apple fruits for measurement of shape . . . 130
6.1.6 Images of apple fruits for measurement of color . . . 130
6.1.7 Images of apple fruits for measurement of russeting in indoor and outdoor . . . 131
6.2 Low-cost acquisition system . . . 132
6.3 Applications . . . 132
6.3.1 PanoVar . . . 132
6.3.2 Ordinalysis . . . 135

7 Annex B **137**
7.1 ColorChecker detection from 3D color point clouds . . . 137
7.2 Calibration of the 3D point cloud using the ColorChecker . . . 140
7.2.1 Re-scaling the point cloud . . . 141
7.2.2 Rotating the point cloud to a canonical reference frame . . . 141
7.2.3 Extraction of region of interest . . . 141
7.2.4 Translating the origin of the reference frame . . . 142

8 Annex C **149**

9 Annex D **151**
9.1 Classical machine learning . . . 151
9.2 Deep learning . . . 151
9.3 Image annotation . . . 152
9.4 Image classification . . . 153
9.5 Segmentation . . . 153
9.5.1 Object detection . . . 154

10 Annex E **155**

11 Annex F **165**

12 Annex G **175**

13 Annex H **185**

13.1 Motivation and significance 185
13.2 Software description 186
13.2.1 Software architecture 186
13.2.2 Programming environment 187
13.2.3 Software functionality 187
13.3 Applications 188
13.3.1 Selection of the best dimension reduction technique 188
13.3.2 Ordinality in images 189
13.4 Impact 191
13.4.1 Tutorial 192
13.5 Conclusion 192

List of Figures

1.1 The process of planting authorized varieties starts with variety selection through plant breeding to variety testing to certify the candidate's varieties as new varieties. The varieties are then tested on a large scale by technical institutes so that farmers can plant and produce food using the certified new varieties. 25

1.2 (a): High-throughput phenotyping platform in plant breeding [1]. (b): Example of measurements conducted manually, in variety testing [2]. (c): Robots used by technical institutes. The illustrated one is called Phenoarch used in Arvalis technical institutes [3]. (d): Robots and drones used by farmers [4]. 26

1.3 A screenshot of UPOV catalog of DUS tests for apples. There are two formats of instructions: wordy and schemes. 28

1.4 Sorting apple machine used for precision agriculture, located at the experimental unit of INRAe. 30

1.5 Types of agriculture. Our field of interest is the arboriculture, highlighted in red. 31

2.1 Apple detection algorithms usually estimate the cumulative apple count from the harvest season. Our aim is to count the number of apples on each individual tree. The main idea is to register the 3D model from the harvest period (a) with the delineated 3D model from the winter period (c) to align the branches with the detected apples (d). We assign a different label to each delineated tree as an output of the automatic tree separation algorithm we perform on the winter model (c). Finally, the detected apples from the harvest model are mapped to their closest branches, and membership of each apple to an individual tree is determined (e). 38

2.2 Pipeline proposed to assign apples to individual trees. (a) Image acquisition of apple trees in winter and harvest period. (b) Calibration of 3D models and extraction of region of interest. (c) Registration of calibrated models from winter and harvest period. (d) Separation of individual trees in winter point cloud. (e) Apple detection from harvest point cloud. (f) Distance map to assign apples to individual segmented trees. 39

2.3 Data acquisition and point cloud calibration modules corresponding to (a) and (b) in Fig. 2.2. (a) Multi-view image acquisition. (b) Apple orchard images acquired in winter and harvest periods. (c) 3D colour point cloud reconstructions (PC_w and PC_h) of orchard scenes with zoom on the ColorChecker. (d) 3D colour point clouds after calibration and extraction of region of interest (PC_w^C and PC_h^C). See Annex B for details of the calibration process. 42

2.4 Block diagram for detection and removal of trellis wires and the water-pipe. 53

2.5 (a) Block diagram for separating individual trees. (b) Illustration of Step 5 for splitting touching trees, (c) Illustration of Step 6 for labeling floating components. 54

2.6 Ground truth. (a) Manually labeled point cloud for assessment of trellis wire, tree trunk and support pole detection, (b) Harvest point cloud with ground truth apple locations, (c) Point cloud manually segmented to individual trees. 60

2.7 (a) Calibrated point clouds, (b) Manually generated Ground Truth (cyan:trellis wires, red: tree trunks, black: support poles), (c) Semantic labels obtained by our method for automatic detection of trellis wires, tree trunks, and support poles 62

2.8 Performance of the method for detection of trellis wires, tree trunks and support poles in terms of Recall (Re), Precision (Pr), F1 score ($F1$), Intersection over Union (IoU), and Class Accuracy (CA) (in %). 65

2.9 Apple detection performance in terms of Recall (Re) and Precision (Pr) (in %). 67

2.10 Accuracy of assigning apples to the correct apple trees in 3D models. . . . 67

2.11 (a), (b) True positives, false negatives, and false positives obtained with colour-based apple detection method for two sample scenes. (c), (d) Registration of harvest and winter clouds for the two scenes. Each separated tree is shown with a different colour. (e), (f) Assignment of true positives to their corresponding trees for the two scenes. 69

3.1 Acquisition system. Upper panel: Machine equipped with a conveyor belt, used for the acquisition of images of apples with a high surface coverage. Lower panel: view of the acquired images of apples after segmentation from the background. 78
3.2 Images representing the 8 non-mutant Gala varieties in the order of appearance in the list of Tab. 3.1. 79
3.3 Images showing a subset of each Gala Mutant, in the same order of appearance as in Tab. 3.2, from left to right and by line, except for the mutant *X8594* which is completely yellow and highly distinctive. 80
3.4 Images representing histograms of non-mutant Gala varieties. From left to right, by row: variety30, variety37, variety40, variety41, variety42, variety44, variety46 and variety47. 81
3.5 Images representing histograms of mutant Gala varieties. From left to right, by row: X4111, X4410, X4712, X6716, X7440, X7812, X8125, X9214. 81
3.6 Reproduced and modified view of the ECPGR catalog. Apple shape sketches in the catalog of variety testing. The classes considered in this work are highlighted in the red rectangle. 93
3.7 Acquisition device: HP Scanjet Pro 4500 fn1. 94
3.8 Histogram of the distribution of classes (*Flat*, *Globose* and *Oval*) assigned by experts. 94
3.9 Reference-based classification approach: (a1): reference sketches Dataset, (a2) RGB images Dataset to be classified. (b1): edge detection. (b2): rescaling of the aspect ratio of sketches of each class. (c) rescaled sketches. (d1) shape features extraction. (e): similarity measure. (f): classification results. 96
3.10 Prediction curves of test set of cured data via model-based classification and reference-based classification using reference sketches, centers representatives and rescaled reference sketches, after training on cured dataset. . 99
3.11 Same as in Fig. 3.10 with only two classes *Flat* and *Oval*. 100

3.12 Two different variety testing catalogs. (a): Catalog delivered by The European Malus GERMPLASM Workshop (ECPGR 2009). (b): Catalog delivered by UPOV (2006). 101

4.1 Ordinal classification problem. The original data is shown in (a). The class centers are enclosed with black circles. The segments joining the class centers are in black color. The objective is to find the optimum viewpoint on the view sphere (b) such that the adjacent class centers are seen as apart as possible. The colors on the sphere are indicative of the objective function defined in 4.2. The 2D visualization (c) of the data is obtained through projecting the data to the plane defined by the optimum viewpoint. 105

4.2 Proposed best view point (BVP) algorithm in action with ordinal datasets by comparison with other classical dimensionality reduction techniques. Panels (a) and (b) are two views of the synthetic 3D ordinal data set. Panels (c) to (i) show results of dimensionality reduction from 3D to 2D. The black circles with numbers correspond to class labels. 108

4.3 Same as Fig. 4.2 with an ordinal dataset composed of partial swiss rolls. . 110

4.4 2D visualization of the dataset contact-lenses. 111

4.5 2D visualization of the dataset pasture 112

4.6 2D visualization of the dataset squash-stored 112

4.7 2D visualization of the dataset newthyroid 112

4.8 2D visualization of the dataset bondrate 112

4.9 2D visualization of the dataset car 112

4.10 Pipeline to illustrate the use of *Ordinalysis* at automating variety testing protocols. (1): ordinal data presented in Annex E. (2): interactive space to visualize image data in Ordinalysis and select the type of features to be extracted. (3): select the dimension reduction techniques. (4a): visualization of the latent space for each dimension reduction technique and the associate plots of both ordinal metrics. (4b): values of ordinal metrics exported as CSV file. 118

5.1 Traits of the distinctness tests mentioned in UPOV catalog. Traits highlighted in green benefited from methodological contributions and engineering tools proposed in this thesis. 123

6.1 (a): image of apple tree. (b): rectangles drawn around the apples. 127
6.2 (a): 3D point cloud of apple trees. (b): 3D point cloud of apple trees with ground truth apple locations. 128
6.3 (a): 3D point cloud of apple trees in winter. (b): manually generated Ground Truth (blue: trellis wires, red: tree trunks, black: support poles, green: branches). (c): labels obtained by our semantic segmentation method (see chapter 2). 129
6.4 (a): images of apples trees in the flowering period. (b): pixels of flowers in the first row are marked with white pixels. 129
6.5 10 cut apples classified as Globose. 130
6.6 Mutant of Gala. 130
6.7 (a): apple without russeting (indoor). (b): apple with russeting (indoor). (c): apple without russeting (outdoor). (d): apple with russeting (outdoor). 131
6.8 (a) Low-cost sorting machine, (b) real-time apples detection and segmentation. 132
133figure.caption.119
6.10 A screenshot from *PanoVar*, where examiners are scoring the shape of cut apples. The sketches represent the scoring scale defined in the UPOV catalog.134

7.1 The ColorChecker object 137
7.2 3D RGB point cloud of an apple orchard scene including a ColorChecker . 138
7.3 Block diagram of the procedure for ColorChecker detection from 3D RGB point cloud 143
7.4 The localized centers of color patches on the ColorChecker are used to estimate the correct scale and the "upward" and "leftward" directions. . . . 144
7.5 The axes of the canonical reference frame to which the point cloud is rotated. Y-axis is parallel to the tree row and oriented towards left. Z-axis is orthogonal to the ground. 144
7.6 Estimation of the Z-axis of the new reference frame. The points below the ColorChecker chart and on the tripod stick (shown in red) are processed with PCA to extract the principal direction. 145
7.7 (a) The origin of the reference frame is moved temporarily to the base of the ColorChecker. (b) After the localization of the designated tree trunk, the origin is set at the base of the designated tree. 145

7.8 The histogram of the points projected to the XY-plane of the ROI. The peaks correspond to the tree trunks. The local region around the position of the designated tree to the ColorChecker is searched for a peak. 146
7.9 Calibration of the point cloud and extraction of region of interest. 147

8.1 **First column:** Calibrated point clouds, **Second Column:** Manually generated Ground Truth (blue: trellis wires, red: tree trunks, black: support poles, green: branches), **Third Column:** Labels obtained by our semantic segmentation method. 149

9.1 The plateau of performance of machine learning and deep learning in function of the amount of data. 152
9.2 Panel of computer vision tasks and associated level of annotation requested for supervised machine learning. 153

13.1 Illustrative schema to use *Ordinalysis*, to process image ordinal data. (1): create the feature space by selecting the image dataset and the type of features. (2): select the dimension reduction (3): visualize ordinality and the ordinal metrics. 187
13.2 Visualizing the dimension reduction techniques: TSNE, LDA, LSDA and BVP, applied on the ordinal feature space: car, presented in [174, 175]. . . 189
13.3 Illustration figure of the associate images to the scale used by the VCU examiners to measure the propagation of powdery mildew in melon foliar disks. 190
13.4 Visualizing the dimension reduction techniques: PCA, TSNE, and BVP, applied on (a): LBP features, (b): deep features (VGG16) and (c): deep features (ResNet50) of the powdery mildew datasets (blue: resistant, cyan: moderate, yellow: severe). 191

List of Tables

2.1 Number of trees in the scenes and number of images acquired in winter and harvest periods. 41
2.2 Performance of the method for detection of trellis wires, tree trunks and support poles in terms of Recall (Re), Precision (Pr), F1 score ($F1$), Intersection over Union (IoU), and Class Accuracy (CA). NP is for non-present. 64
2.3 Apple detection performance in terms of Recall (Re) and Precision (Pr). . 66
2.4 Average processing times of the steps of the pipeline in seconds. 68
2.5 Machine characteristics. 68

3.1 Number of images of 8 reference, registered varieties corresponding to Non-Gala Mutant dataset. 79
3.2 Number of images of 9 reference, registered varieties corresponding to Gala Mutant dataset. 80
3.3 Results obtained by SVM with Gala mutant dataset and all features. . . . 85
3.4 Results obtained by LDA and Random Forest with Gala mutant dataset and all features. 86
3.5 Results obtained by SVM with Gala mutant dataset and optimal transport only. 86
3.6 Results obtained by LDA and Random Forest with Gala mutant dataset and optimal transport only. 87
3.7 Results obtained by SVM with non Gala mutant dataset and all features. . 88
3.8 Results obtained by LDA and Random Forest with non Gala mutant dataset and all features. 88
3.9 Results obtained by SVM with non Gala mutant dataset and optimal transport only. 89
3.10 Results obtained by LDA and Random Forest with non Gala mutant dataset and optimal transport only. 89
3.11 Results obtained by SVM on individual data and subsets of apples, with optimal transport only and all features. 90

3.12 Results obtained by LDA on individual data and subsets of apples, with optimal transport only and all features. 90
3.13 Results obtained by Random Forest on individual data and subsets of apples, with optimal transport only and all features. 91
3.14 Measuring accuracy (ACC) of reference-based approach on expert 1 (E1), expert 2 (R2) and cured data (CD), to all types of sketches. 97
3.15 Same as in Table 1 but with only two classes *Flat* and *Oval*. 98
3.16 Translation between variety testing catalogs. 100

4.1 Real ordinal datasets used for the experiments [146, 147] (I is the total number of instances, Q is the dimensionality of the original data and K is the number of classes). 111
4.2 *Inter-Class intersection* and *Deviation from ordinality* values extracted from latent spaces generated after applying dimension reduction techniques (PCA, TSNE, LDA, ISOMAP, MDS, LSDA and BVP) on synthetic dataset in Fig. 4.2a. 116
4.3 *Inter-Class intersection* and *Deviation from ordinality* values extracted from latent spaces generated after applying dimension reduction techniques (PCA, TSNE, LDA, ISOMAP, MDS, LSDA and BVP) on synthetic dataset in Fig. 4.3. 116
4.4 *Inter-Class intersection* and *Deviation from ordinality* values extracted from latent spaces generated after applying dimension reduction techniques (PCA, TSNE, LDA, ISOMAP, MDS, LSDA and BVP) on pasture dataset. 116
4.5 *Inter-Class intersection* and *Deviation from ordinality* values extracted from latent spaces generated after applying dimension reduction techniques (same as in table 4.4) on bondrate dataset. 117
4.6 *Inter-Class intersection* and *Deviation from ordinality* values extracted from latent spaces generated after applying dimension reduction techniques (same as in table 4.4) on contact lenses dataset. 117
4.7 *Inter-Class intersection* and *Deviation from ordinality* values extracted from latent spaces generated after applying dimension reduction techniques (same as in table 4.4) on newthyroid dataset. 117
4.8 *Inter-Class intersection* and *Deviation from ordinality* values extracted from latent spaces generated after applying dimension reduction techniques (same as in table 4.4) on squash-stored dataset. 117

13.1 Code metadata 185
13.2 *Inter-Class intersection* values extracted from LBP, Deep features of VGG16 and deep features of ResNet50 latent spaces generated after applying the dimension reduction techniques: PCA, TSNE and BVP. 190

Chapter 1

Introduction

1.1 Problematic

In industrialized countries, farmers have to use authorized varieties. The process of authorizing a new variety to be on the market includes four main entities (see Fig. 1.1). First, the breeders create candidate varieties, either by classical crossing or by genomic manipulations. The candidate varieties pass through a set of tests called variety testing, conducted by examination offices in charge of certifying the quality of these new candidates and determining how they depart from existing varieties. In France, this is the role of the "groupe d'étude et de contrôle des variétés et des semences" (GEVES) located in Angers. Once validated by the examination offices, the candidate varieties are certified as new varieties and registered to an official catalog, where all commercialized varieties are listed. The new varieties still need to be tested on larger scales. In France, this is the responsibility of the technical institutes (such as Arvalis for main crops, ITB for sugar beet, or CTIFL for fruits and vegetables). The technical institutes also advise farmers on optimally using these new varieties.

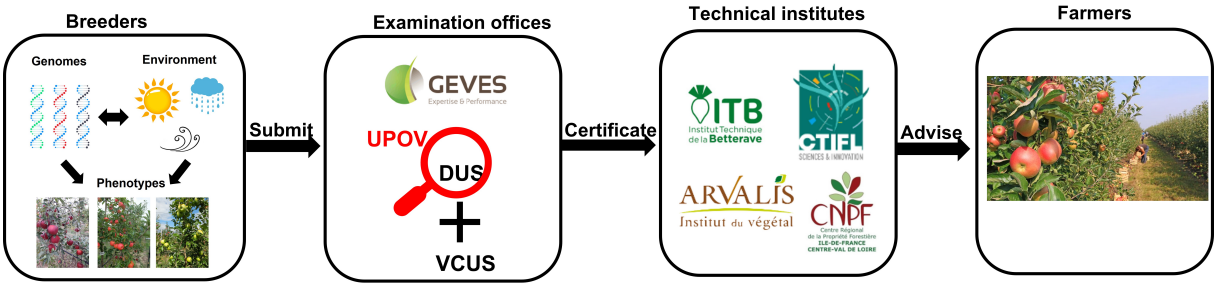


Figure 1.1 – The process of planting authorized varieties starts with variety selection through plant breeding to variety testing to certify the candidate's varieties as new varieties. The varieties are then tested on a large scale by technical institutes so that farmers can plant and produce food using the certified new varieties.

The steps in Fig. 1.1, corresponding to tasks of breeders, technical institutes and

farmers, already include automation and numerical practices based on computer vision in their protocols, unlike measurements in variety testing, that are conducted based on visual inspection (see Fig. 1.2).

Figure 1.2 – (a): High-throughput phenotyping platform in plant breeding [1]. (b): Example of measurements conducted manually, in variety testing [2]. (c): Robots used by technical institutes. The illustrated one is called Phenoarch used in Arvalis technical institutes [3]. (d): Robots and drones used by farmers [4].

Computer vision is the domain of information sciences that aims to extract information automatically from images via computers. Nowadays, it is often coupled with machine learning, which automatically defines rules for this information extraction based on data.

Computer vision is currently widely used in precision agriculture and plant breeding [5, 6, 7, 8, 9, 10]. In plant breeding, the transition to numerical practices involves training technicians to handle computer vision tools during workshops.

For instance, in France, the french network of high-throughput plant phenotyping (PHENOME) is a project that aims to equip the french scientific community with an infrastructure able to measure the agronomic characteristics of plants using precise and high throughput methods.

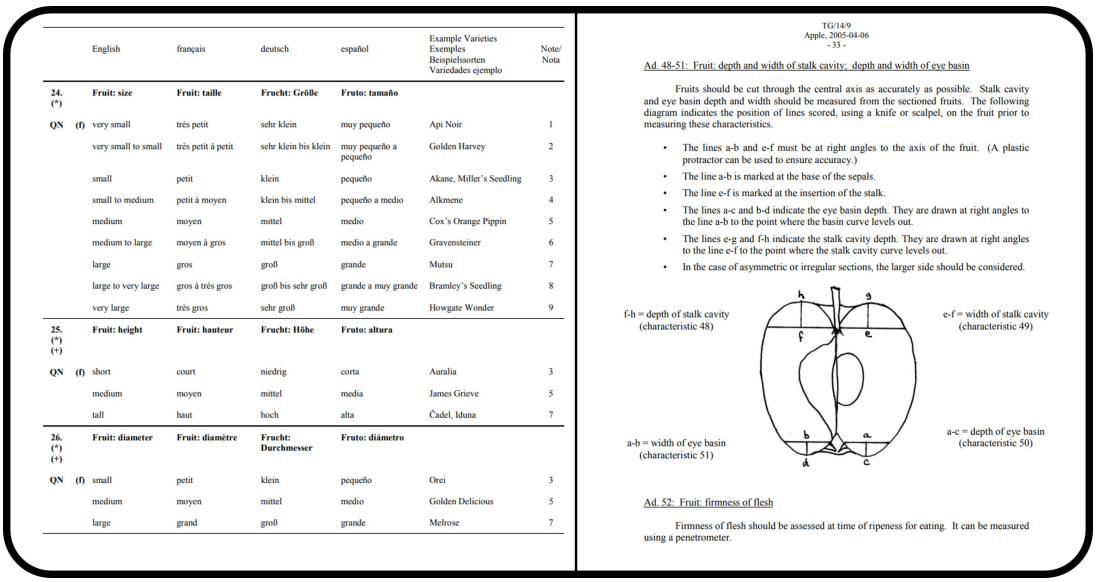
Likewise, the international plant phenotyping network (IPPN) is an association created to increase the visibility and impact of plant phenotyping and enable cooperation by fostering communication between stakeholders in academia, industry, government, and the public. Through workshops and symposia, IPPN seeks to establish different working groups and distribute all relevant information about plant phenotyping on a web-based platform.

With the demonstrated potential of computer vision in plant breeding and precision agriculture, its low exploitation in variety testing earlier seems questionable, especially since the need to increase the speed and accuracy of variety testing is urgent to certificate varieties able to adapt to varying climate conditions. This absence of numerical transition of practices is explained by the specificity and challenges of variety testing, hence the non-use of computer vision tools developed in precision agriculture and plant breeding. Before listing the constraints and limitations of the current practices, we define the categories of variety testing in the next section.

1.2 Type of tests in variety testing

1.2.1 Distinctness, uniformity and stability (DUS)

During the DUS tests, examiners follow instructions and guidelines set by the International Union for the Protection of New Varieties of Plants (UPOV), to extract relevant characteristics (e.g. plant height, leaf shape, time of flowering) to ensure that a new variety is distinct from existing varieties, that its characteristics are uniform, and that the variety is stable with consistent phenotypic characteristics from one generation to the next [2]. The instructions are gathered in the official catalog of DUS variety testing (see Fig. 1.3). Instructions are either sentences or schemes within the measurement to conduct.



		English	français	deutsch	español	Example Varieties Exemples Beispielsorten Variedades ejemplo	Note/ Nota
24. **(*)**		**Fruit: size**	**Fruit: taille**	**Frucht: Größe**	**Fruto: tamaño**		
QN	**(f)**	very small	très petit	sehr klein	muy pequeño	Api Noir	1
		very small to small	très petit à petit	sehr klein bis klein	muy pequeño a pequeño	Golden Harvey	2
		small	petit	klein	pequeño	Akane, Miller's Seedling	3
		small to medium	petit à moyen	klein bis mittel	pequeño a medio	Alkmene	4
		medium	moyen	mittel	medio	Cox's Orange Pippin	5
		medium to large	moyen à gros	mittel bis groß	medio a grande	Gravensteiner	6
		large	gros	groß	grande	Mutsu	7
		large to very large	gros à très gros	groß bis sehr groß	grande a muy grande	Bramley's Seedling	8
		very large	très gros	sehr groß	muy grande	Howgate Wonder	9
25. **(*)** **(+)**		**Fruit: height**	**Fruit: hauteur**	**Frucht: Höhe**	**Fruto: altura**		
QN	**(f)**	short	court	niedrig	corta	Auralia	3
		medium	moyen	mittel	media	James Grieve	5
		tall	haut	hoch	alta	Čadel, Iduna	7
26. **(*)** **(+)**		**Fruit: diameter**	**Fruit: diamètre**	**Frucht: Durchmesser**	**Fruto: diámetro**		
QN	**(f)**	small	petit	klein	pequeño	Orei	3
		medium	moyen	mittel	medio	Golden Delicious	5
		large	grand	groß	grande	Melrose	7

Ad. 48-51: Fruit: depth and width of stalk cavity; depth and width of eye basin

Fruits should be cut through the central axis as accurately as possible. Stalk cavity and eye basin depth and width should be measured from the sectioned fruits. The following diagram indicates the position of lines scored, using a knife or scalpel, on the fruit prior to measuring these characteristics.

- The lines a-b and e-f must be at right angles to the axis of the fruit. (A plastic protractor can be used to ensure accuracy.)
- The line a-b is marked at the base of the sepals.
- The line e-f is marked at the insertion of the stalk.
- The lines a-c and b-d indicate the eye basin depth. They are drawn at right angles to the line a-b to the point where the basin curve levels out.
- The lines e-g and f-h indicate the stalk cavity depth. They are drawn at right angles to the line e-f to the point where the stalk cavity curve levels out.
- In the case of asymmetric or irregular sections, the larger side should be considered.

f-h = depth of stalk cavity (characteristic 48)

e-f = width of stalk cavity (characteristic 49)

a-b = width of eye basin (characteristic 51)

a-c = depth of eye basin (characteristic 50)

Ad. 52: Fruit: firmness of flesh

Firmness of flesh should be assessed at time of ripeness for eating. It can be measured using a penetrometer.

Figure 1.3 – A screenshot of UPOV catalog of DUS tests for apples. There are two formats of instructions: wordy and schemes.

1.2.2 Value for cultivation and use (VCU)

VCU tests are carried out to evaluate a variety's suitability for growing in given agro-climatic conditions and the use made of harvested crops and products produced from that variety [2]. Candidate varieties are evaluated based on:

- Yield level;
- End-use quality (protein content and micro-malting quality in winter barley, oil and protein content in rapeseed, animal feed quality in forage maize, etc.);
- Resistance to disease, pests, environmental aggressors such as wind and winter cold;

While DUS is managed by an intergovernmental authority (UPOV), VCU tests are rather governed by a national authority of each country. Lately, efforts have been made to unify the VCUs tests, at least at the European level.

1.3 Why variety testing needs specific machine learning methodologies and computer vision tools.

As illustrated in Fig. 1.2, most measurements in variety testing are conducted manually based on a visual inspection. In addition to being slow, the accuracy of the experts'

scoring systems is rather subjective because of visual perception limitations. This can also impact the reproducibility of the results. This is very important to acknowledge because often, varieties are planted in several sites to monitor their evolutions during their interactions with the local climate. The comparison between the performance of the same variety by the different examination offices is therefore questionable, as there are no guarantees that the measurement conditions were identical.

Limitations in current practices of variety testing call for the use of computer vision to increase the speed of the tests, the accuracy of the measurements, and to guarantee uniform measurement conditions for reproducibility of the results. However, there are specific challenges that restrains the shift to numerical practices.

DUS Variety testing follows guidelines and instructions imposed by the UPOV. These instructions are set through a voting system, including all stakeholders and members of the organization. On the other hand, VCU tests follow specific rules defined by each country. Such decision systems complexify the possibility of proposing alternative scoring systems or shifting from manual to automated measurements based on computer vision.

All examination offices in variety testing are state-owned. Budgets assigned for variety testing are relatively low compared to the agro-industry. This impacts the possibility of incorporating new technologies and investing in training examiners who often do not have a background in computer vision or image processing tools.

Measurements in variety testing are conducted on many crops and a smaller scale than in precision agriculture. Exploiting existing computer vision tools of precision agriculture in variety testing is therefore not straightforward nor cost-effective. For instance, at INRAe of Angers, technicians use a sorting machine (see Fig. 1.4). This machine uses sensors mounted in a black box to acquire images and perform image processing in real-time for sorting. This is not suitable for variety testing for the following reasons. First, the machine costs around ($\approx 100k€$) and is applied only for sorting. Hence, it is not generalizable to all post-harvest measurements conducted in variety testing. Secondly, such commercialized systems do not share the raw data that produce the measurements. This represents a limitation in variety testing because the data can be requested as proof for the client to debate the decision of examination offices. Data can also be exploited in uniformity tests as the results are compared over five years. Third, there is no guarantee that the measurements of traits performed by the sorting machine follow the same instructions and the scale in the UPOV. This makes them unusable in variety testing. To shift to numerical practices, the examination offices need, therefore to develop new

computer vision tools that can be generalizable and affordable.

Figure 1.4 – Sorting apple machine used for precision agriculture, located at the experimental unit of INRAe.

1.4 Our contributions

During this PhD, we contribute to computer vision and machine learning for plant variety testing by developing generalizable methodologies adapted to the specificity and constraints of variety testing with a focus on low-cost solutions. The materials developed were shared among the examiners of variety testing in the European project INVITE (https://www.h2020-invite.eu/).

To demonstrate the possible transition to numerical practices based on computer vision and machine learning in variety testing, we targeted arboriculture (see Fig. 1.5), particularly apple fruits, because of the importance of this crop in the Maine-et-Loire as it is a province where the agro-industry of apples is very active. We also selected apples because of the proximity to examiners conducting DUS tests on varieties of apples. In Angers, Pip Fruit Diversity Valorization (VaDiPom) is a research team located at INRAE. Their main objectives are valorizing apple diversity, breeding programs and conducting variety testing for apple varieties' commercialization. VaDiPom is a member from the so-called European Refpop community [11]. Refpop examiners participated in providing

data during this thesis. Our exchanges and collaboration were operated in the context of the European project INVITE.

INVITE is a 5-year European Union-funded project, that aims to valorize and promote varieties that are more adapted to sustainable management practices and more resilient to climate change. The selected species in this project are apple, fodder grass, sunflower, soybean, wheat, maize, potato, tomato, oilseed rape, and lucerne, representing the main features of propagation, food and having an important breeding activity at the European level. Among the missions of INVITE, developing new phenotyping tools to enhance the speed, accuracy and efficiency of variety testing. This meets our purpose of demonstrating the possibility of incorporating computer vision and machine learning in variety testing.

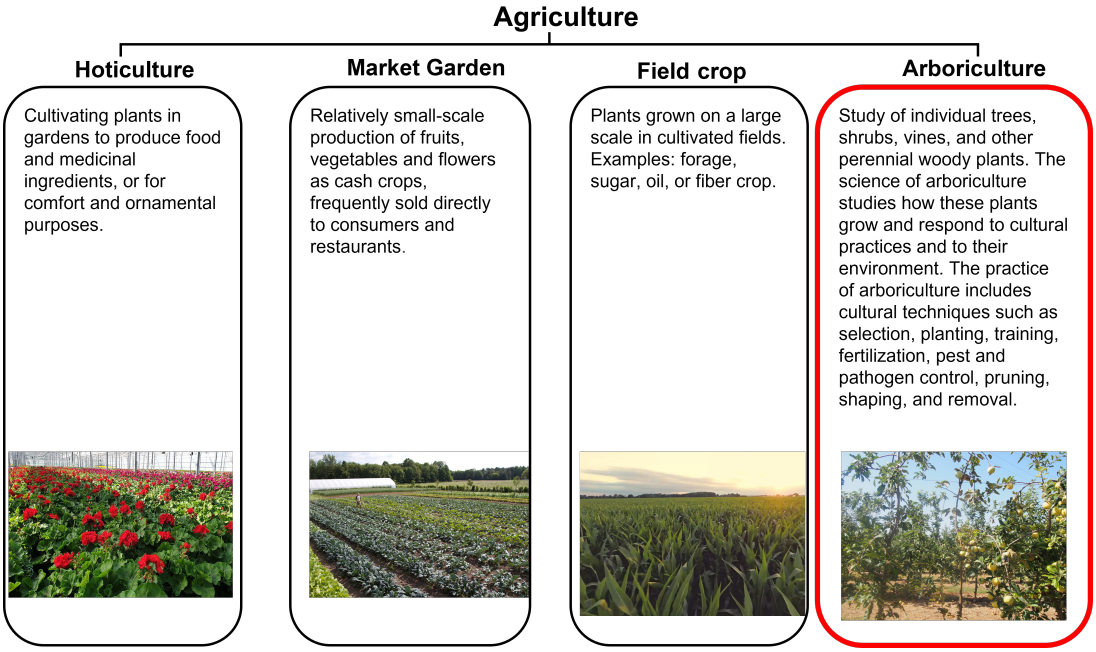


Figure 1.5 – Types of agriculture. Our field of interest is the arboriculture, highlighted in red.

Regarding our contributions in variety testing, We chose not to prioritize VCUs tests because the scoring scale is often not normalized across all examination offices and is specific to each country. In such a scenario, proposing automated pipelines will be suited to a particular country and not generalizable.

To contribute on an international level, we target DUS traits, more precisely, the distinctness test of DUS protocols. The stability is a comparison of the performance of a variety over five years. We could not address this category due to the limited time of

the PhD. Uniformity is a comparison between the performance of the same population of varieties. This test can benefit from works achieved in the automation of the distinctness test.

The objective of the distinctness tests is to verify that the candidate varieties are different from the reference varieties of the official catalog, based on measurements of traits. Machine learning can fit perfectly in this situation because the distinctness test can be seen as a classification problem (see Annex A). With several measured traits, the candidate and reference varieties can be represented in a high dimensional point cloud, where each axis refers to a measurement. The classifier in the high dimensional point cloud can quantify the separability between the varieties.

The distinctness tests are conducted during the pre-harvest and post-harvest periods. In each category, trait measurements need to be automated to address the distinctness test as a classification problem in machine learning. In the pre-harvest period, the tools developed must be adapted to the specificity of the variety testing orchards. Unlike orchards dedicated to producing food, trees in variety testing orchards are planted tightly (to reduce the investment cost). Therefore, the branches of neighbor trees intersect, making the automating of measurements complex. In chapter 2 of this document, we present a 3D computer vision pipeline developed to separate trees. This work is a foundational step for the automation of apple traits such as estimation of flowering intensity, estimation of the mean color of fruits and estimation of the mean size of fruits (see Fig. 5.1).

During the post-harvest distinctness tests, the controlling conditions are normalized and the automation of trait measurements is less complex. To demonstrate the possibility to incorporate computer vision and machine learning in the distinctness tests of DUS variety testing, we selected color and shape traits (see chapter 3).

During a numerical distinctness test, the varieties are represented by descriptors in the feature space (for definitions of machine learning terminology, see Annex D). Some traits, such as color or stripes, are scored on an ordinal scale (see Fig. **??**). Therefore, the ordinality is also present in the point cloud representing the varieties. We developed, in chapter 4, a dimension reduction technique to visualize the ordinality in the feature space and two quantification metrics to understand the decision of machine learning algorithms. These methods will be demonstrated on variety testing, but can be applied to any application involving ordinal measurements.

1.5 Structure of this document

The document is organized in three chapters (2,3,4) before a concluding chapter. In these three chapters, we present our main methodological contributions. In chapter 2, we present a 3D computer vision pipeline developed to delineate the trees in an apple variety testing orchards, to overcome a specificity of variety testing orchards and open prospects for automating DUS traits, accurately. In chapter 3, we present supervised and unsupervised methods to incorporate machine learning in the distinctness tests of the DUS variety testing. In chapter 4, we present a dimension reduction technique and quantification metrics to visualize and quantify ordinality in the latent space. These three chapters deal with variety testing matters and are all applied on apple variety testing protocols. As a disclaimer, we stress that they are methodological contributions which are largely independent from each others. Consequently, we do not provide a centralized state-of-the-art chapter, but rather provide bibliographic state-of-the-art sections in the introduction of each chapter. A common element in the manuscript stands on the adaptation of machine learning methods. To avoid expanding the introduction of this document, we present in Annex D, a brief explanation of the terminology of machine learning used in this thesis.

In addition to the main methodological contributions of this work, we have developed engineering contributions, which are proposed in Annex A. The methods in chapter 4 were incorporated in a ready-to-use application called *Ordinalysis*. The impact of this application will be demonstrated on the measurements of the resistance of melon varieties to powdery mildew, in Annex H. In Annex F, we exploit the transfer learning from indoor data to outdoor data, to reduce the cost of annotation and enhance the robustness of a machine learning model at recognizing the russeting on apple fruits in the orchard. Annex B and C detail the protocol to perform the 3D reconstruction and the visualizations associated with the work done in chapter 2.

Chapter 2

Computer vision for variety testing orchards

2.1 Introduction

In this chapter, we propose a novel methodology to delineate apple trees in a trellis structured orchard. This structure is very common in variety testing orchards where each tree represents a single different variety and therefore has to be characterized separately. The pipeline proposed is tested for the counting of apples (Fig. 2.1). The material presented in this chapter has been published in [12].

Our strategy is to reconstruct 3D models of the same set of trees twice a year, once during the winter period and once during the harvest period. We perform delineation of individual trees on the leaf-off model from winter, which we refer to as *winter point cloud*. We detect tree trunks and identify the branches connected to them using winter point cloud. We employ the 3D model from the harvest period, which we call *harvest point cloud*, to localize apples. We determine the tree-membership of each apple in the harvest point cloud by mapping their locations onto the winter point cloud, where individual trees are separated. This approach of registering data from two different time instances for fruit counting is another novelty we introduce to the field. We also propose the use of a known calibration object to facilitate the registration of two point clouds and to recover the true metric sizes of the important structures in the scenes.

The main contributions of this study are:

- Addressing the problem of apple counting on individual trees from 3D colour point clouds.
- As a way to map detected apples to individual trees, alignment of harvest point cloud to the winter point cloud, where individual trees are automatically delineated.
- A complete pipeline for detecting and removing trellis wires and support poles, detecting tree trunks and delineating crowns of individual trees in winter point

clouds.

- The use of a calibration object for correct scaling and alignment of point clouds acquired in different time instances.

Individual tree delineation is the process of separating individual trees, including trunk detection and crown boundary delineation; i.e. identifying the trunk and branches belonging to a single tree [13]. Delineation of trees in dense orchards or forests is a challenging task due to interlacing and touching branches of adjacent trees, particularly when there is high variation among the trees in terms of crown size and shape [13]. Occlusion caused by dense leaf cover during harvest period further complicates the delineation of trees. Using leaf-off data collected during winter can alleviate the occlusion and facilitate the capture of trunk and branch geometry [14, 15].

The architectural structure that determines the connectivity of the branches to a particular tree trunk becomes ambiguous in 2D images, even during winter period. 2D projection causes loss of shape and connectivity information of the branches of neighbouring trees. Processing 3D point clouds is more adequate for our application since 3D data enables a detailed analysis of the geometric structure of trees and localisation of branches and fruits in the 3D world.

Computer vision techniques aiding management of fruit orchards range from complete processing pipelines to algorithms performing single tasks such as tree localisation [16, 17, 18, 19, 20, 21, 22, 23, 24]. A vision system was developed by [16] to reconstruct 3D fruit trees and identify branch structure and traits for automatic pruning. In [17] an automatic trunk-detection system using an infrared sensor was introduced. Medeiros et al. [18] employed a laser sensor to model dormant fruit trees and identify primary branches for automatic pruning. In [19] Regions-Convolutional Neural Network (R-CNN) was applied on depth images for detection of branches of apple trees and localisation of shaking points to guide a harvesting machine. Zeng et al. [20] developed an algorithm to segment trellis wires, support poles, and tree trunks in sparse LiDAR point clouds acquired from trellis-structured apple orchards. In order to optimise the mechanisation of fruitlet and blossom thinning, Nielsen et al. [21] used LiDAR and stereo vision together for obtaining 3D models of orchard rows of trees. They fitted mixtures of Gaussians to the point cloud to cluster the trees into Gaussian shaped cylinders. In [22], LiDAR data was used for individual tree separation through a hidden semi-Markov model. Their objective was to develop a pipeline for building detailed orchard maps and an algorithm to match subsequent LiDAR tree scans to the prior database, enabling correct data association

for precision agricultural applications. In [23], a procedure for segmenting canopy to individual trees was proposed. The procedure involved octree construction, clustering, trunk detection and Ncut segmentation. 3D data was obtained with terrestrial laser scanning (TLS) and mobile laser scanning (MLS). In [24], a tree trunk detection pipeline was proposed for identifying individual trees in a trellis structured apple orchard, using ground-based LiDAR and image data. Hough transformation was performed on 3D point cloud to search for trunk candidates. These candidates were projected into the camera images, where pixel-wise classification was used to update their likelihood of being a tree trunk. Detection was achieved by using a hidden semi-Markov model to leverage from the contextual information provided by the repetitive structure of the orchard.

Regarding the detection and counting of apples, while the majority of computer vision techniques for fruit detection and counting relied on RGB (Red, Green, Blue) images, other types of data including RGB-Depth images [25, 26, 27, 28, 29, 30], spectral images [31], thermal images [32, 33, 34, 35, 36] images or LiDAR (Light Detection and Ranging) data [37] have also been used. In traditional approaches for fruit detection through such sensor information, relevant information is extracted from each data instance separately according to a manually predefined algorithm. The representative quantitative information obtained in this manner is generally referred to as a hand-crafted feature. Hand-crafted approaches can involve techniques such as colour thresholding, colour space clustering, shape analysis, blob detection, circular Hough transform, Ncut algorithm, employment of Histogram of Oriented Gradients (HOG), Local Binary Patterns (LBP) and Upright Speeded Up Robust Features (U-SURF) for separating fruits from the canopy [38, 39, 40, 27, 41, 26, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51]. Recently, deep learning methods have become commonplace for fruit detection and counting [52, 53, 54, 55, 56, 57, 58, 25, 59, 28, 60, 30, 36, 61]. Deep neural networks are employed to learn predictors from a set of training data through optimising the parameters of feature extraction and localisation of fruits simultaneously. After prediction, further processing, such as circular Hough transform and watershed transform [43] for verification and filtering of multiple counts through 3D (3-Dimensional) reconstruction [62, 41, 57] can be applied to extract the final fruit count.

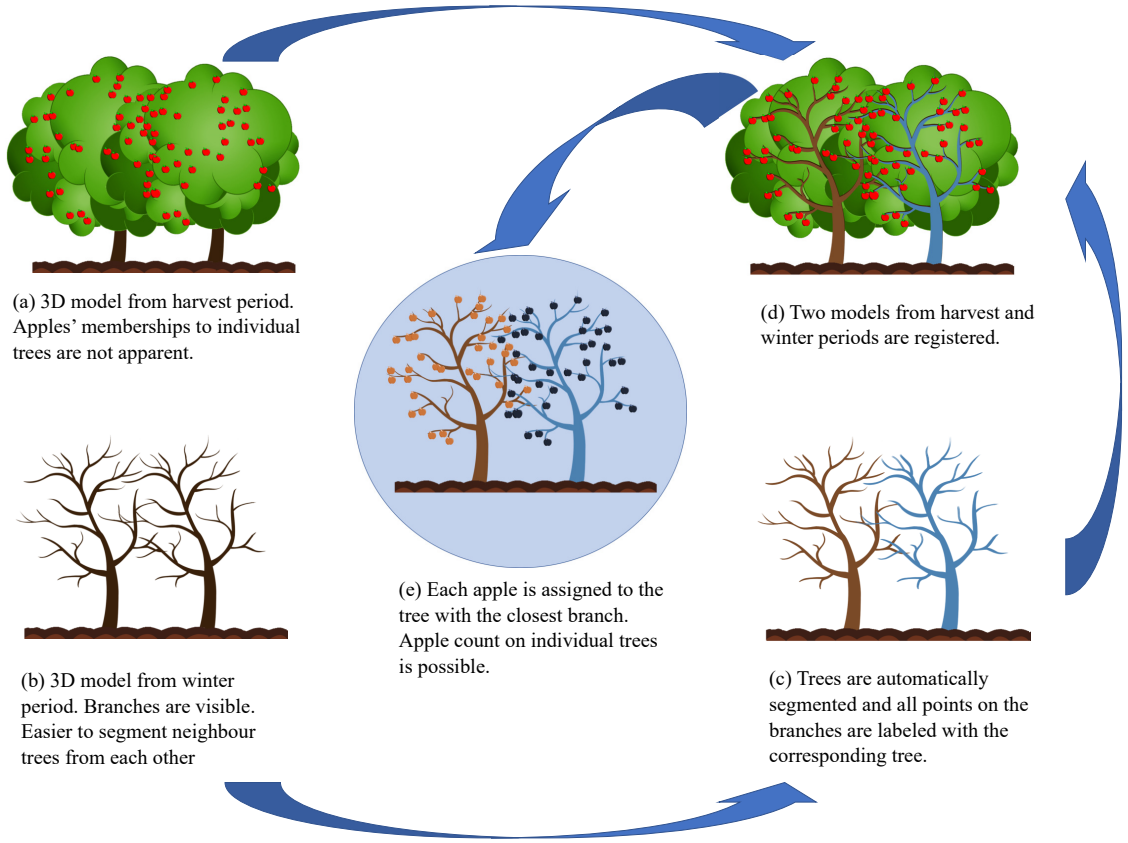


Figure 2.1 – Apple detection algorithms usually estimate the cumulative apple count from the harvest season. Our aim is to count the number of apples on each individual tree. The main idea is to register the 3D model from the harvest period (a) with the delineated 3D model from the winter period (c) to align the branches with the detected apples (d). We assign a different label to each delineated tree as an output of the automatic tree separation algorithm we perform on the winter model (c). Finally, the detected apples from the harvest model are mapped to their closest branches, and membership of each apple to an individual tree is determined (e).

2.2 Materials and methods

We developed a point cloud processing pipeline (Fig. 2.2) in order to locate and count apples on individual trees. We use a colour camera for capturing images of target trees in the orchard from multiple views during both winter and harvest periods (Fig. 2.2 (a)). These images are processed by a structure from motion algorithm to reconstruct winter and point clouds. The two point clouds are prepared for initial alignment which we refer to as *calibration of point clouds* (Fig. 2.2 (b)). A novelty of our pipeline is the use of a ColorChecker during acquisition. The ColorChecker serves both as a reference for removal

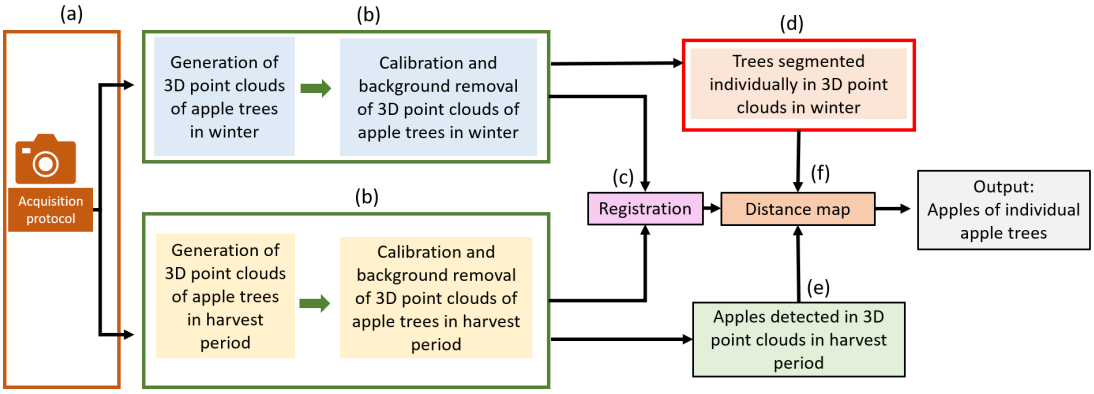


Figure 2.2 – Pipeline proposed to assign apples to individual trees. (a) Image acquisition of apple trees in winter and harvest period. (b) Calibration of 3D models and extraction of region of interest. (c) Registration of calibrated models from winter and harvest period. (d) Separation of individual trees in winter point cloud. (e) Apple detection from harvest point cloud. (f) Distance map to assign apples to individual segmented trees.

of irrelevant background information and as a calibration tool. Calibration of the point cloud, in our case, involves 1) re-scaling the point cloud to the correct metric scale, 2) orienting the point cloud to a canonical reference frame, 3) extraction of region of interest, and 4) re-centering the point cloud to a predetermined position. The estimated scale allows us to impose metric parameters on the pipeline such as range of separation between trees, separation between trellis wires, diameter of trellis wires, diameter of tree trunks, the expected pole diameter and height, etc. The orientation and re-centering facilitate trellis wire removal, tree trunk detection, and delineation of tree crowns (Fig. 2.2 (d)). The calibration of both harvest and winter point clouds is also crucial for their correct registration (Fig. 2.2 (c)). We employ a colour-based apple detection algorithm to locate the apples in the harvest point cloud (Fig. 2.2 (e)). Finally, we map the detected apples onto the winter cloud via distance calculation to assign them to their bearing trees. We give detailed explanations of each module of our pipeline in the following subsections.

2.2.1 Experimental field

The experiments were conducted in a dense apple orchard, dedicated to variety testing at INRAe-Angers (latitude: 47.48226°N, longitude: 0.6152°E) in France. The orchard was composed of 4 years old apple trees organised in I-trellis structure with support poles. Our target trees were arranged in a row, where each tree was a mutant, being tested to be established as a new apple variety. The spacing between trees was 1m in average and

the height of the trees ranged from 1 to 3m. The variation of the crown shape among the trees was high.

2.2.2 Data acquisition and 3D reconstruction

Fig. 2.3 illustrates the data acquisition and point cloud calibration processes of our pipeline, corresponding to the modules (a) and (b) in Fig. 2.2. We obtained 3D colour point clouds of seven scenes from the orchard through a multi-view reconstruction process. A *scene*, in our study, refers to part of an orchard row; i.e. a set of adjacent trees in the same row. Each scene contained 4 to 5 apple trees in our experiments, although our algorithm is capable of processing an entire orchard row. The number of trees in each scene is given in Tab. 2.1.

A 3D colour point cloud (or a 3D RGB point cloud) PC is a set of 3D points, where each point is represented by its coordinates (x, y, z) and its colour (R, G, B). Here, (R, G, B) refers to the values of red, green and blue channels.

We captured multiple RGB images of size 3000×4000 pixels, of a scene with a colour camera (Fujifilm X20, Fujifilm Corporation, Tokyo, Japan) in both winter and harvest periods to reconstruct the point clouds. We acquired images from only one side of the orchard row; although it is possible to follow the procedure proposed in [63] to reconstruct and register two sides of a row. Table 2.1 lists the number of images used for 3D reconstruction of the scenes from winter and harvest periods. In this study, we captured multiple images from the scene manually, choosing the viewpoints and viewing angles (i.e. camera positions and orientations) to get visual information covering the scene from top to bottom and from various sides of the trees. It is important to guarantee that there is enough overlap between pairs of images for a successful 3D reconstruction. This process can be automated in a more systematic manner with path planning, using a drone [64] or a land robot equipped with multiple cameras [16].

The multi-view images were used to reconstruct 3D colour point clouds of the scenes through VisualSFM [65, 66] and PMVS/CMVS tool [67, 68]. VisualSFM is a freely available software [65, 66] that performs Structure from Motion (SfM) to estimate unknown camera locations and orientations. It provides a sparse point cloud of the scene through keypoint matching and triangulation. In order to obtain a dense point cloud, we used PMVS/CMVS tool, another freely-available software [67, 68]. This tool takes as input the images and the camera parameters computed by VisualSFM and provides a dense reconstruction of the scene through multi-view stereo. For introductory and in-depth

information on the techniques of SfM and multi-view stereo, we refer the reader to the textbook of Hartley and Zisserman [69].

Before capturing the images of each scene, we installed a calibration object (ColorChecker Passport Photo 2, X-rite, Great Lakes, Midwestern US) mounted on a tripod stick at a known position. We placed the tripod stick in front of the trees facing the camera, such that the ColorChecker pattern is almost parallel to the tree row Fig. 2.2 (b). When the ColorChecker stick was installed, we manually measured two distances with a tape measure: d_R^{cc}: the minimum distance of the tripod stick to the tree row, and d_T^{cc}: the distance to a designated target tree. These values are necessary for the calibration process of the point clouds.

The reconstructed harvest point cloud and winter point cloud of a scene are referred to as PC_h and PC_w respectively. Point clouds of a sample scene are given in Fig. 2.3 (c) with the ColorChecker objects zoomed in.

Table 2.1 – Number of trees in the scenes and number of images acquired in winter and harvest periods.

	# trees	# images (winter)	# images (harvest)
Scene 1	5	236	364
Scene 2	5	189	382
Scene 3	5	221	380
Scene 4	4	183	374
Scene 5	5	206	380
Scene 6	4	199	376
Scene 7	4	227	376

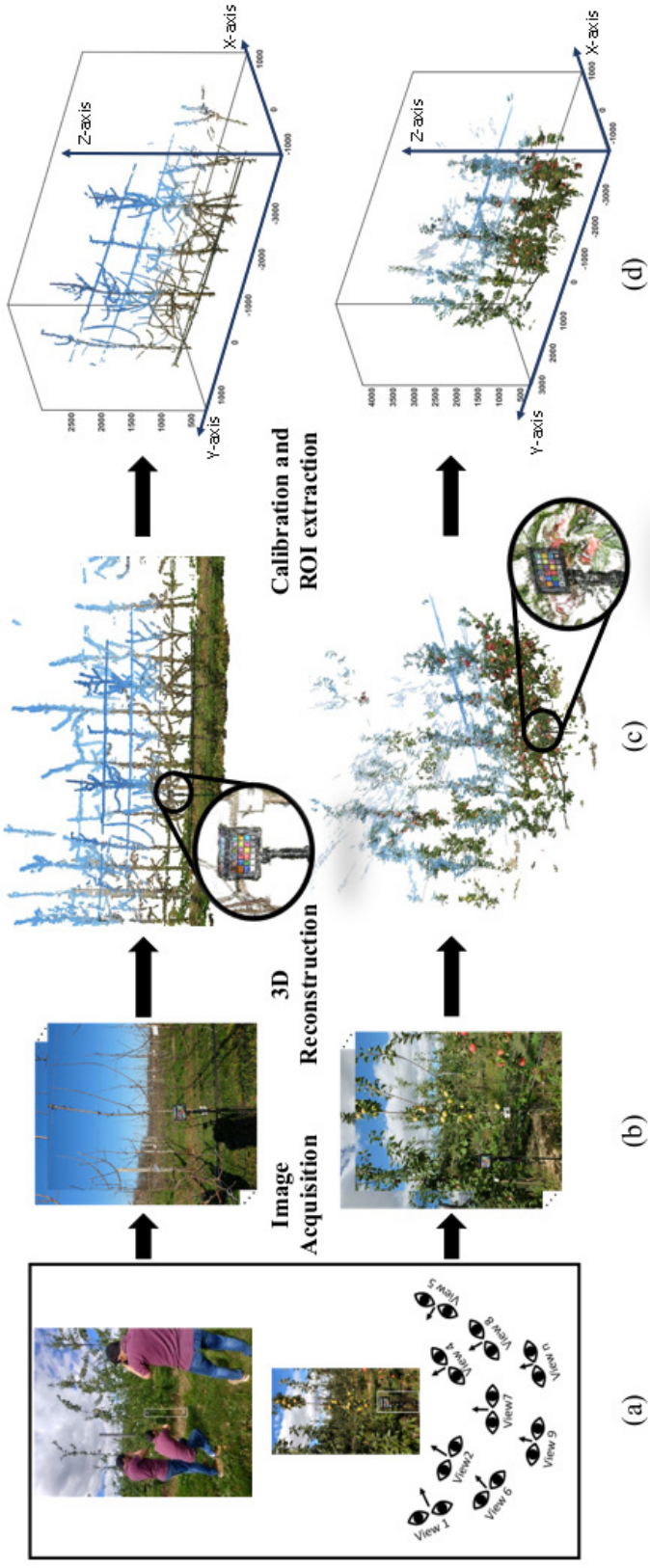


Figure 2.3 – Data acquisition and point cloud calibration modules corresponding to (a) and (b) in Fig. 2.2. (a) Multi-view image acquisition. (b) Apple orchard images acquired in winter and harvest periods. (c) 3D colour point cloud reconstructions (PC_w and PC_h) of orchard scenes with zoom on the ColorChecker. (d) 3D colour point clouds after calibration and extraction of region of interest (PC_w^C and PC_h^C). See Annex B for details of the calibration process.

2.2.3 Calibration and extraction of region of interest

The calibration of the point clouds from harvest and winter periods provides an initial alignment, which is fundamental for the success of the registration of the two point clouds. Having the point cloud with the accurate scale also enables us to fix parameters, such as trunk diameter, tree height, separation between trees, according to the range of expected metric sizes of the structures in the scene.

The ColorChecker is usually employed as a colour reference to obtain accurate colours from images under varying lighting conditions [70]. In this work, we do not use the ColorChecker for this purpose. Instead, we use it as a distinct reference pattern to geometrically calibrate the raw point clouds. We developed an algorithm for automatic detection of the ColorChecker, together with the tripod stick it is mounted on, from 3D colour point clouds. The description of this algorithm can be found in Annex B. The 3D locations of the centers of the colour patches of the ColorChecker chart are used to guide the calibration of the point cloud.

The geometric calibration process takes as input the harvest and winter point clouds (PC_h and PC_w) and produces the calibrated point clouds (PC_h^C and PC_w^C), as shown in Fig. 2.3 (d). The details of the calibration process are given in Annex B. In summary, the calibration process consists of 1) estimation of the true scale and re-scaling the point cloud; 2) re-defining a canonical reference frame and rotating the point cloud to this new frame; 3) extraction of region of interest, which corresponds to the set of trees just behind the ColorChecker; and 4) moving the origin of the reference frame to the base of the designated tree. The canonical reference frame is defined such that Y-axis is parallel to the tree row and Z-axis is orthogonal to the ground.

2.2.4 Separation of individual trees

In this section, we describe the procedure to separate the trees from each other in the winter scenes. This procedure involves localisation of target tree trunks, finding the points on the tree trunks, detecting and removing trellis wires, the water pipe, and the support poles. After the trees are localised and irrelevant points are removed, the tree membership of all the remaining points are determined.

Let the number of points in the calibrated winter point cloud $PC_w^C = \{p_1, p_2, ..., p_{N_W}\}$ be N_W. We aim to map each point p_i to a semantic label γ_i, $i = 1, 2, ..., N_W$ where $\gamma_i \in \Gamma$. Γ is the set of four semantic labels: $\Gamma = \{$ *"Tree trunk"*, *"Branch"*, *"Trellis wire+Water*

pipe", "Support pole"}. The process of automatically labeling the points in the cloud with one of these four classes is called *semantic segmentation* of the scene. The rationale for a semantic segmentation stage is to remove irrelevant structures and to eliminate the connectivity between adjacent trees caused by trellis wires and the water pipe.

In conjunction with semantic segmentation, we also detect trees in the scene and locate their trunks. Let the set of verified trees in the scene be denoted as $\mathcal{T}$. Each tree T_j in $\mathcal{T}$ is represented by its tree identity $t_j \in \{1, 2, ..., N_{trees}\}$ and its location L_j, for $j = 1, 2, ..., N_{trees}$. The location of a tree corresponds to the coordinates of its base $L_j = (x_j, y_j, z_j)$, $j = 1, 2, ..., N_{trees}$ measured in the canonical reference frame.

After removing the irrelevant structures (trellis wires, water pipe and support pole) we delineate the trees in the winter point cloud. The final output of the tree separation algorithm is the assignment of each trunk and branch point in the calibrated winter point cloud PC_w^C to one of the trees in the set $\mathcal{T}$.

Detection of trellis wires and tree trunks

The procedure for detecting points on trellis wires is based-on estimation of the *trellis-plane* and the *trellis-lines* along the trellis wires and operating on the points close to these estimates. Candidate trunk locations are detected along the trellis-plane based on point density. The points in a cylindrical region along each candidate location is separately skeletonised. The skeleton and the points surrounding it are examined to verify tree trunk position and to detect the presence of a supporting pole. 3D points belonging to the trunk of each individual tree and support pole are identified and labeled. Regions between tree trunks along the initial line estimates are re-examined through 3D line fitting to increase the precision of the detection and removal of the points that belong to the trellis wires. The steps of the procedure are shown in Fig. 2.4 and detailed below:

Step 1: Voxelization The calibrated winter point cloud PC_w^C is converted to binary volumetric form, where a voxel takes the value 1 if the voxel is occupied by the points in PC_w^C. Specifically, we fit a regular 3D grid to the bounding box defined by the minimum and maximum coordinate values (x_{min}, x_{max}), (y_{min}, y_{max}), (z_{min}, z_{max}) of the points in PC_w^C. Each cell, i.e. voxel, of the grid has edge lengths of $\Delta_x = \Delta_y = \Delta_z = 5mm$. On this grid, we define a 3D array B of size $N_x \times N_y \times N_z$, where

$$
\begin{aligned}
N_x &= \lfloor \frac{x_{max} - x_{min}}{\Delta_x} \rfloor + 1; \\
N_y &= \lfloor \frac{y_{max} - y_{min}}{\Delta_y} \rfloor + 1; \\
N_z &= \lfloor \frac{z_{max} - z_{min}}{\Delta_z} \rfloor + 1.
\end{aligned} \tag{2.1}
$$

Here $\lfloor \cdot \rfloor$ is the floor function. The 3D volumetric form of the point cloud corresponds to the binary function B computed as

$$
B(k,l,m) = \begin{cases} 1, & \text{if } \exists p = (x,y,z) \in PC_w^C : \\ & \lfloor \frac{x - x_{min}}{\Delta_x} \rfloor = k \ \ \& \ \ \lfloor \frac{y - y_{min}}{\Delta_y} \rfloor = l \ \ \& \ \ \lfloor \frac{z - z_{min}}{\Delta_z} \rfloor = m \\ 0, & \text{otherwise,} \end{cases} \tag{2.2}
$$

for $k = 0, ..., N_x - 1$, $l = 0, ..., N_y - 1$, and $m = 0, ..., N_z - 1$. In Fig. 2.4 (Step 1), the volumetric model of a sample scene is visualized. Only the voxels with value "1" are shown.

Step 2: Skeletonisation We extract the skeleton of the volumetric model B using medial axis thinning algorithm given in [71]. Formally, the skeleton of a 3D object is the set of the centers of all inscribed maximal spheres where these spheres touch the object boundary at one than more point [71]. The skeletonisation process produces another binary 3D grid S of size $N_x \times N_y \times N_z$, where the structures in B are pruned to curves with thickness of one voxel. In Fig. 2.4 (Step 2), the skeleton of a sample scene is shown.

Step 3: Projection and Hough Transform The skeleton defined in the binary 3D grid S is projected to the YZ-plane (parallel to the tree row) as a binary image, IH of size $N_y \times N_z$:

$$
IH(l,m) = \begin{cases} 1, & \text{if } \sum_{k=0}^{N_x - 1} S(k,l,m) > 0 \\ 0, & \text{otherwise,} \end{cases} \tag{2.3}
$$

for $l = 0, ..., N_y - 1$, and $m = 0, ..., N_z - 1$.

In Fig. 2.4 (Step 3), the projected binary image of a sample scene is shown. We apply 2D Hough Transform [72] to IH to extract main horizontal lines in the binary image. The peaks greater than 20% of the maximum value in the Hough parameter space, and with angle with the horizontal axis less than 10° are selected as the main horizontal lines. These horizontal lines correspond to candidates for the trellis-lines in the scene.

Step 4: Estimation of the trellis-plane

The detected horizontal lines are back-projected to the 3D space of the point cloud

PC_w^C, as shown with red lines in Fig. 2.4 (Step 4). Let the set of these horizontal 3D lines be $\mathcal{L}_{HL} = \{hl_1, hl_2, ..., hl_{N_{HL}}\}$, where N_{HL} is the number of horizontal lines. Each 3D line is defined by a pair of points on it, as $hl_r = (p_{r,1}, p_{r,2})$, with $p_{r,1} = (x_{r,1}, y_{r,1}, z_{r,1})$ and $p_{r,2} = (x_{r,2}, y_{r,2}, z_{r,2})$. We retrieve the points in PC_w^C with distance 1cm to these lines, and form the subset:

$$PC_{tr} = \{p = (x, y, z) \in PC_w^C : \min_{r=1,..,N_{HL}} d(p, hl_r) < 1cm\}. \tag{2.4}$$

The distance $d(p, hl_r)$ between a point p and the line hl_r is calculated as:

$$d(p, hl_r) = \frac{\|(p - p_{r,1}) \times (p - p_{r,2})\|}{\|p_{r,2} - p_{r,1}\|}, \tag{2.5}$$

where $\times$ is the cross product operation, and $\|\cdot\|$ is the Euclidean norm. We fit a plane to the points in PC_{tr} using M-estimator SAmple Consensus (MSAC) algorithm given in [73], which is a variant of RANdom SAmple Consensus (RANSAC) algorithm. Maximum distance for a point to be an inlier is set to be 0.5cm. The output of the algorithm is a plane model (A, B, C, D), where the parameters define the plane equation $Ax+By+Cz+D = 0$. The unit vector $n_{TP} = (A, B, C)$ corresponds to the normal of the plane. We refer to this plane as the *trellis-plane* on which trellis wires and tree trunks are located. Fig. 2.4 (Step 4) shows the trellis-plane fitted to the points in PC_{tr} for a sample scene.

The trellis-plane plays an important role in the following steps. We rotate the calibrated winter point cloud PC_w^C to a new reference frame such that the new YZ plane coincides with the trellis-plane and Y-axis is parallel to the trellis-lines. The new Y-axis is computed as the average of the direction vectors of the horizontal lines in $\mathcal{L}_{HL}$:

$$u_Y = \frac{\sum_{r=1}^{N_{HL}} (p_{r,2} - p_{r,1})}{\|\sum_{r=1}^{N_{HL}} (p_{r,2} - p_{r,1})\|} \tag{2.6}$$

The new Z-axis is orthogonal to the normal of the trellis-plane and the average direction of the trellis-lines:

$$u_Z = u_Y \times n_{TP}, \tag{2.7}$$

and the new X-axis is

$$u_X = u_Y \times u_Z \tag{2.8}$$

We transform each point $p = (x, y, z)$ in the calibrated winter cloud PC_w^C using the

rotation matrix R defined in Eq. (2.9), and obtain a point cloud of the same size, PC_w^{TP}. We refer to this point cloud as the winter point cloud aligned to the trellis-plane.

$$PC_w^{TP} = \{\hat{p} = (\hat{x}, \hat{y}, \hat{z}) = pR : p \in PC_w^C\}; \quad R = \begin{bmatrix} u_X \\ u_Y \\ u_Z \end{bmatrix} \tag{2.9}$$

The origin of the new reference frame remains at the base of the target tree (see Annex B). With the transformation, the trellis-plane coincides with the $\hat{x} = 0$ plane in the new reference frame. This ensures that the $\hat{x}$ coordinate of each tree trunk is close to 0. Notice that this transformation is applied only to the winter point cloud. Once the semantic segmentation of the winter cloud is achieved and the trees are delineated, the points are transformed back to their original positions using $p = \hat{p}R^{-1}$.

Step 5: Merge lines The detected horizontal lines are on the trellis-plane; hence, they are located on the $\hat{x} = 0$ plane in the new reference frame. Their average direction is parallel to the Y-axis. Hence, we represent each line $hl_r \in \mathcal{L}_{HL}$ with the direction vector $(0, 1, 0)$ and a point on the line $(0, 0, \hat{z}_r)$. The value $\hat{z}_r$ indicates the height of a horizontal line on the trellis-plane and is calculated as:

$$\hat{p}_{r,1} = (\hat{x}_{r,1}, \hat{y}_{r,1}, \hat{z}_{r,1}) = p_{r,1}R; \quad \hat{p}_{r,2} = (\hat{x}_{r,2}, \hat{y}_{r,2}, \hat{z}_{r,2}) = p_{r,2}R; \tag{2.10}$$

$$\hat{z}_r = \frac{\hat{z}_{r_1} + \hat{z}_{r_2}}{2} \tag{2.11}$$

We merge the lines into parallel lines on the trellis-plane, each separated by at least 30cm to create the set of trellis-lines $\mathcal{L}_{TL} = \{tl_1, ..., tl_{N_{TL}}\}$. Each line is represented with the direction vector $(0, 1, 0)$ and a point on the line $(0, 0, \hat{z}_q)$, with $q = 1, ..., N_{TL}$. We use the following procedure to cluster the horizontal lines in $\mathcal{L}_{HL}$ into trellis-lines in $\mathcal{L}_{TL}$: We first sort the horizontal lines with ascending height. We start from the bottom line on the trellis-plane, and initialise $\hat{z}_1$ to the height of the first horizontal line. If the distance between the closest horizontal line is less than 30cm, we add the line to the group and update $\hat{z}_1$ to the average height of the group. Otherwise, we create a new group and proceed to the next line. In our experiments, the horizontal lines were grouped into 4 lines for all the winter scenes. Fig. 2.4 (Step 5) shows the resulting trellis-lines in red colour for a sample winter scene. In the rest of the paper we fix $N_{TL} = 4$. The four height values $\{\hat{z}_1, \hat{z}_2, \hat{z}_3, \hat{z}_4\}$ will be used to specify the locations of the trellis-lines.

Step 6: Trunk candidate localisation To localise candidate tree trunks along the trellis-plane we limit the search space within 5cm distance to the trellis-plane. We extract a subset of points PC_{ts} from PC_w^{TP}:

$$PC_{ts} = \{\hat{p} = (\hat{x}, \hat{y}, \hat{z}) \in PC_w^{TP} : |\hat{x}| < 5cm\} \tag{2.12}$$

Fig. 2.4 (Step 6) shows PC_{ts} of a sample scene. We define a regular 2D grid, IG on the $z = 0$ plane, which is parallel to the ground. Each cell of the grid has edge length $\hat{\Delta}_x = \hat{\Delta}_y = 1cm$. We compute the number of points in PC_{ts} falling into each cell:

$$\mathcal{IG}_{i,j} = \{\hat{p} = (\hat{x}, \hat{y}, \hat{z}) \in PC_{ts} : \lfloor \frac{\hat{x} - \hat{x}_{min}}{\hat{\Delta}_x} \rfloor = i \quad \& \quad \lfloor \frac{\hat{y} - \hat{y}_{min}}{\hat{\Delta}_y} \rfloor = j\}; \tag{2.13}$$

$$IG(i, j) = |\mathcal{IG}_{i,j}|, \tag{2.14}$$

where $\hat{x}_{min}$ and $\hat{y}_{min}$ are the minimum of the $\hat{x}$ and $\hat{y}$ coordinates of the points in PC_{ts}, and $|\mathcal{X}|$ denotes the number of elements in the set $\mathcal{X}$.

IG is the histogram of the points in PC_{ts} projected to the ground. The points on the tree trunks form the densest regions in the histogram correspond to the peaks of IG. The locations of the peaks are detected via non-maximum suppression [74] as $\{(I_1, J_1), (I_2, J_2), ..., (I_{N_P}, J_{N_P})\}$, where N_P is the number of detected peaks.

The set of candidate trunk locations in the 3D space are then defined as $\mathcal{CT} = \{(0, \hat{y}_1^{ct}, 0), (0, \hat{y}_2^{ct}, 0), ..., (0, \hat{y}_{N_P}^{ct}, 0)\}$; with $\hat{y}_1 < \hat{y}_1 < ... < \hat{y}_{N_P}^{ct}$. Recall that the trunks intersect with the trellis-plane. $\hat{y}_s^{ct}$ for $s = 1, ..., N_P$ is calculated as:

$$\hat{y}_s^{ct} = J_s \hat{\Delta}_y + \hat{y}_{min}. \tag{2.15}$$

Fig. 2.4 (Step 6) shows the locations of the candidate trunks as vertical purple lines passing through $(0, \hat{y}_s^{ct}, 0)$.

Step 7: Trunk verification Not all the peaks detected in the previous step correspond to tree trunks. In this step, we examine the points at each candidate trunk location to verify whether it is a tree trunk, a support pole, or neither. We construct the set of trees $\mathcal{T}$ using the verified trunks. Each tree T_j in $\mathcal{T}$ is represented by its tree identity $t_j \in \{1, 2, ..., N_{trees}\}$ and the location of its base $\hat{L}_j = (\hat{x}_j, \hat{y}_j, \hat{z}_j)$, for $j = 1, 2, ..., T_{N_{trees}}$. The procedure for constructing the set of detected trees is given in Algorithm 1, and explained below:

Algorithm 1: Tree trunk verification

Data: PC_w^{TP}: The winter point cloud aligned to the trellis-plane;
$(0, \hat{y}_s^{ct}, 0)$: Candidate tree trunk locations for $s = 1, ..., N_P$
Result: $\mathcal{T} = \{T_1, ..., T_{N_{trees}}\}$: Set of detected trees;
N_{trees}: Number of detected trees;
t_j: Tree identity of $T_j \in \mathcal{T}$;
$\hat{L}_j = (\hat{x}_j, \hat{y}_j, \hat{z}_j)$: Location of $T_j \in \mathcal{T}$;
SP_j: Set of points on the main axis of T_j

Initialise $\mathcal{T} = \emptyset$; $N_{trees} = 0$; $j = 0$;
for $s \leftarrow 1$ *to* N_P **do**
    Extract the point set PC_s^{CT} using Eq. (2.16);
    Convert PC_s^{CT} to binary volumetric form B_s through voxelization;
    Compute the skeleton S_s of B_s using medial axis thinning [71];
    Obtain SK_s by retrieving the 3D points on the skeleton S_s ;
    Find the top and bottom points in SK_s with the largest and smallest z-coordinates and designate them as $\hat{p}_{s,top}$ and $\hat{p}_{s,bottom}$;
    Extract the shortest path between $\hat{p}_{s,top}$ and $\hat{p}_{s,bottom}$ using Breadth-first search [75] ;
    Collect the points on the shortest path to form the main axis SP_s;
    Calculate the length d_s^{SP} of SP_s;
    if $d_s^{SP} > 1m$ **then**
        Run Support Pole Detection Algorithm on s (Section 19) ;
        if *s is not a Support Pole* **then**
            $j \leftarrow j + 1$; $N_{trees} \leftarrow N_{trees} + 1$; $t_j = j$;
            $\hat{L}_j = (0, \hat{y}_s^{ct}, 0)$;
            $SP_j = SP_s$;
            $T_j = (t_j, \hat{L}_j, SP_j)$;
            $\mathcal{T} \leftarrow \mathcal{T} \cup T_j$

We first initialise the set of trees as $\mathcal{T} = \emptyset$ and the number of tree trunks as $N_{trees} = 0$. For each candidate trunk indexed with s, we define a cylindrical region, with radius 15 cm, centered at the candidate trunk location $(0, \hat{y}_s^{ct}, 0)$, along the trellis-plane. We extract the points inside this region from PC_w^{TP}:

$$PC_s^{CT} = \{\hat{p} = (\hat{x}, \hat{y}, \hat{z}) \in PC_w^{TP} : \sqrt{\hat{x}^2 + (\hat{y}^2 - \hat{y}_s^{ct})^2} < 15cm\} \tag{2.16}$$

The point cloud PC_s^{CT} is converted to binary volumetric form B_s with voxel size $\hat{\Delta}_x = \hat{\Delta}_y = \hat{\Delta}_z = 5mm$. Then, the skeleton S_s is extracted from B_s using medial axis thinning algorithm given in [71]. The points on the skeleton are retrieved from the point cloud PC_s^{CT}, and denoted as SK_s.

The two top and bottom points of the set SK_s along the Z-axis $\hat{p}_{s,top}$ and $\hat{p}_{s,bottom}$ are retrieved. The points on the shortest path between these two points is computed using the Breadth-first search algorithm described in [75]. We refer to the set of the points on the shortest path as the *main axis* of the s^{th} trunk, and denote it as SP_s. Fig. 2.4 (Step 7) shows the skeleton with black dots and the points on the shortest path with blue dots for a candidate trunk location.

If the length of the shortest path d_s^{SP} is less than 1m, then the candidate trunk location is discarded. Otherwise, it is passed to the support pole detection procedure described in Section 19. If it is not identified as a support pole, then we update $N_{trees} \leftarrow N_{trees} + 1$, and insert the verified trunk into $\mathcal{T}$. We also store the main axis of the verified trunk. See Algorithm 1 for the formation of the set $\mathcal{T}$.

Step 8: Extraction of trunk points The previous step gives the attributes of each tree $T_j = (t_j, \hat{L}_j, SP_j) \in \mathcal{T}$. The main axis of the j^{th} detected tree is represented by the set of points SP_j. We label a point $\hat{p}_i$ in the point cloud PC_w^{TP} as *"Tree trunk"* if its distance to the main axis of one of the trees is less than 3cm. Specifically:

$$\gamma_i = \textit{"Tree trunk"} \quad \text{if} \quad \min_j \min_{\hat{p} \in SP_j} \|\hat{p} - \hat{p}_i\|^2 < 3cm \tag{2.17}$$

Fig. 2.4 (Step 8) shows the points semantically labeled as *"Tree trunk"* in a winter scene.

Step 9: Locating the intersection points of trellis wires and tree trunks
In Step 4, the set of trellis-lines $\mathcal{L}_{TL} = \{tl_1, tl_2, tl_3, tl_4\}$ is determined. Recall that each

line is represented with the direction vector $(0, 1, 0)$ and a point on the line $(0, 0, \hat{z}_q)$, with $\hat{z}_1 < \hat{z}_2 < \hat{z}_3 < \hat{z}_4$. Now, having located the trunks at $\hat{L}_j = (0, \hat{y}_j, 0)$ with $\hat{y}_1 < \hat{y}_2 < ... < \hat{y}_{N_{trees}}$, we find the points where the trellis wires intersect with the trunk locations. For a trellis-line with index q and a trunk location with index j, we find the point $\hat{p}_{q,j} \in PC_w^{TP}$ closest to the location $(0, \hat{y}_j, \hat{z}_q)$. Fig. 2.4 (Step 9) shows the trellis-lines, located tree trunks and the intersection points $\hat{p}_{q,j}$ for a winter scene.

Step 10: Finding the end points of trellis wires The end-points corresponding to the trellis wires in the scene are determined by finding the closest points to the trellis-lines at the two extremes of the point cloud along the Y-axis. Specifically, for a trellis-line with index q, we locate two points $\hat{p}_{q,0} \in PC_w^{TP}$ and $\hat{p}_{q,N_{trees}+1} \in PC_w^{TP}$, which are closest to the locations $(0, \hat{y}_{min}, z_q)$ and $(0, \hat{y}_{max}, z_q)$, respectively. Here, $\hat{y}_{min}$ and $\hat{y}_{max}$ are the minimum and maximum Y-coordinates of the points in PC_w^{TP}. Fig. 2.4 (Step 10) shows the end points for a winter scene with red and yellow dots.

Step 11: Line fitting to find the points on trellis wires and the water-pipe The region between each adjacent intersecting points of trellis-lines and the trunks are examined for a precise determination of the points on the trellis wires and the water-pipe. For each pair of intersecting points $\hat{p}_{q,j}$ and $\hat{p}_{q,j+1}$, $q = 1, ..., 4 \quad j = 0, 1, ..., N_{trees}$ we extract the points:

$$\begin{aligned} PC_{j,j+1}^q = &\{\hat{p} = (\hat{x}, \hat{y}, \hat{z}) \in PC_w^{TP} : \\ &(\hat{y}_j + 4cm < \hat{y} < \hat{y}_{j+1} - 4cm) \ \ \& \ \ (d(\hat{p}, ls_{j,j+1}) < 10cm)\} \end{aligned} \tag{2.18}$$

where $ls_{j,j+1}$ is the line defined by the points $\hat{p}_{q,j}$ and $\hat{p}_{q,j+1}$, and $d(\hat{p}, ls_{j,j+1})$ is the distance between point $\hat{p}$ and line $ls_{j,j+1}$. This region corresponds to a cylinder of radius 10cm with axis $ls_{j,j+1}$. We set an offset value of 4cm from the trunk locations not to include trunk points to the search region for trellis wire points.

Using MSAC algorithm given in [73], we fit two lines to the points in $PC_{j,j+1}^0$ corresponding to the regions along the lowest trellis-line (one for the trellis wire and one for the water-pipe). One line is fitted to the points for the rest of the regions $PC_{j,j+1}^q$ with $q = 2, 3, 4$. For a point to be an inlier, the maximum distance to the fitted line is set to be 7cm for $q = 0$ and 4cm for $q = 2, 3, 4$. Fig. 2.4 (Step 11) shows points in two regions along the trellis wires in blue and the lines fitted to them in black.

If a point $\hat{p}_i \in PC_w^{TP}$ is an inlier of one of the fitted lines we set its semantic label as $\gamma_i =$ *"Trellis wire + Water pipe"*.

Step 12: Removal of detected trellis wire points

The detected trellis wire points and the points on the support pole, if there is any, are removed from the point cloud to form the set:

$$\begin{aligned} PC_w^{trees} = &\{\hat{p}_i \in PC_w^{TP} : \\ &(\gamma_i \neq \textit{"Trellis wire + Water pipe"}) \ \& \ (\gamma_i \neq \textit{Support pole"})\} \end{aligned} \tag{2.19}$$

The procedure for retrieving points on the support pole is given in Section 19. The point cloud PC_w^{trees} is supposed to include only the points on the trees. In Fig. 2.4 (Step 12) the points labeled as *"Trellis wire+Water pipe"* are shown in dark blue. Also the resulting PC_w^{trees} is given for a sample winter scene.

Detection of Support Poles

During the procedure for trellis wire detection and localisation of tree trunks, we examine each trunk candidate to determine whether it corresponds to a support pole or an actual tree trunk. We consider the points in a vertical cylindrical region of radius 15cm centered at the candidate trunk location. We partition the points into horizontal slices of height 2cm. We project the points in each slice onto the XY-plane (the ground plane) and fit a circle of radius 4.5cm (the actual radius of a support pole in the orchard) to the projected points, and estimate the center. The centers of the slices form the axis of the candidate support pole and the new cylindrical region. We count the points in the cylindrical shell with inner and outer radii, $4.5-0.5$ and $4.5+0.5$cm, and with height 2.3m (the actual height of a support pole). If the ratio of this number to the total number of points in the initial cylindrical region is higher than 0.8, then we declare that the structure corresponds to a support pole. We label the points in the cylindrical shell as pole points.

Identifying tree membership of points (Tree Separation)

This module of our pipeline is responsible for delineating the trees in PC_w^{trees}, which is the point cloud with trellis wires, the water-pipe and the support pole removed. The output of the delineation process is the assignment of each point in PC_w^{trees} to one of the trees $T_j = (t_j, \hat{L}_j, SP_j) \in \mathcal{T}$.

The main steps of the tree separation process is given in Fig. 2.5. We convert PC_w^{trees} to binary volumetric form and apply skeletonisation to conduct a connectivity analysis. We delineate adjacent trees if they are touching and we assign isolated connected components to one of the two nearest trees through a set of rules. The details of the steps are as follows:

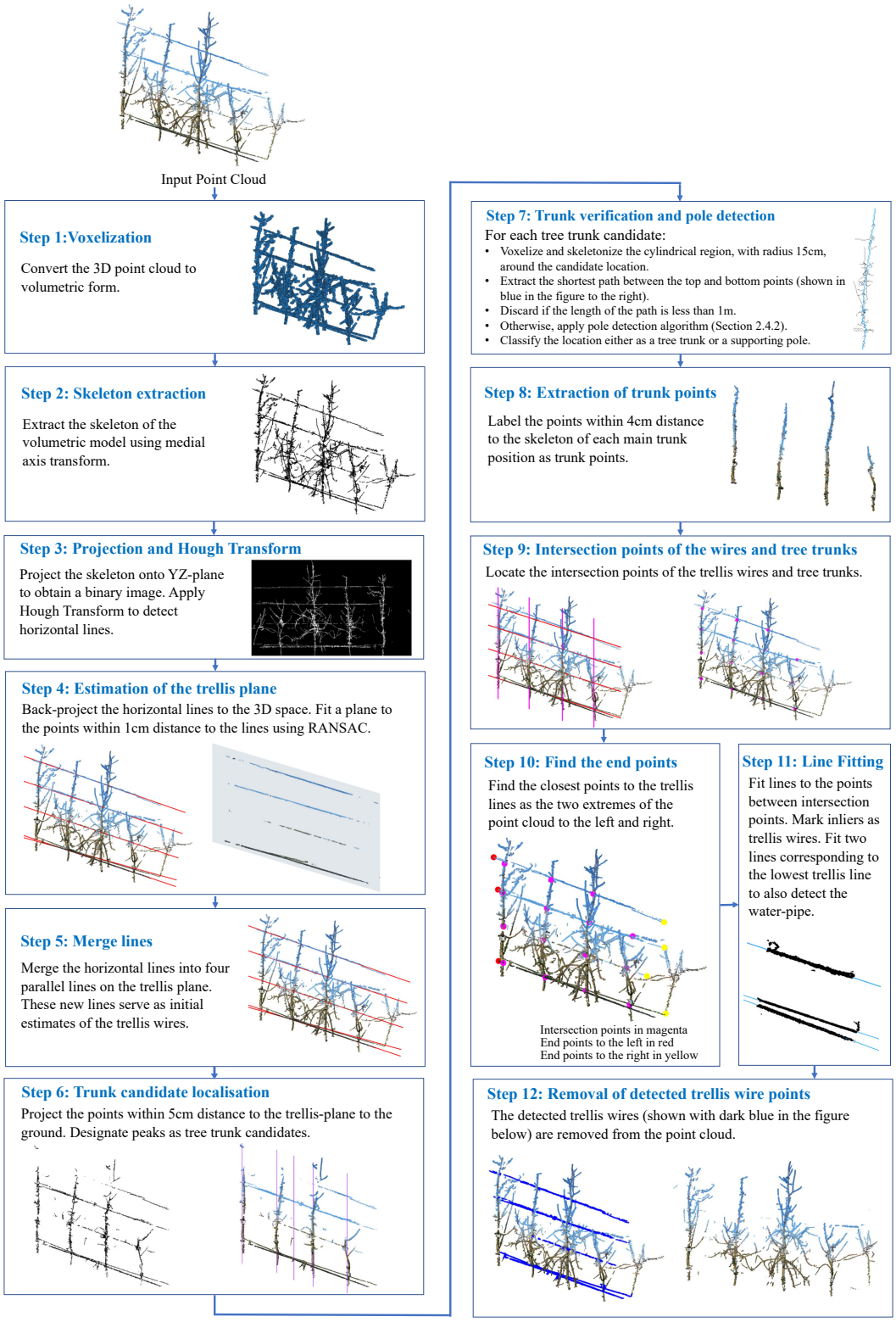


Figure 2.4 – Block diagram for detection and removal of trellis wires and the water-pipe.

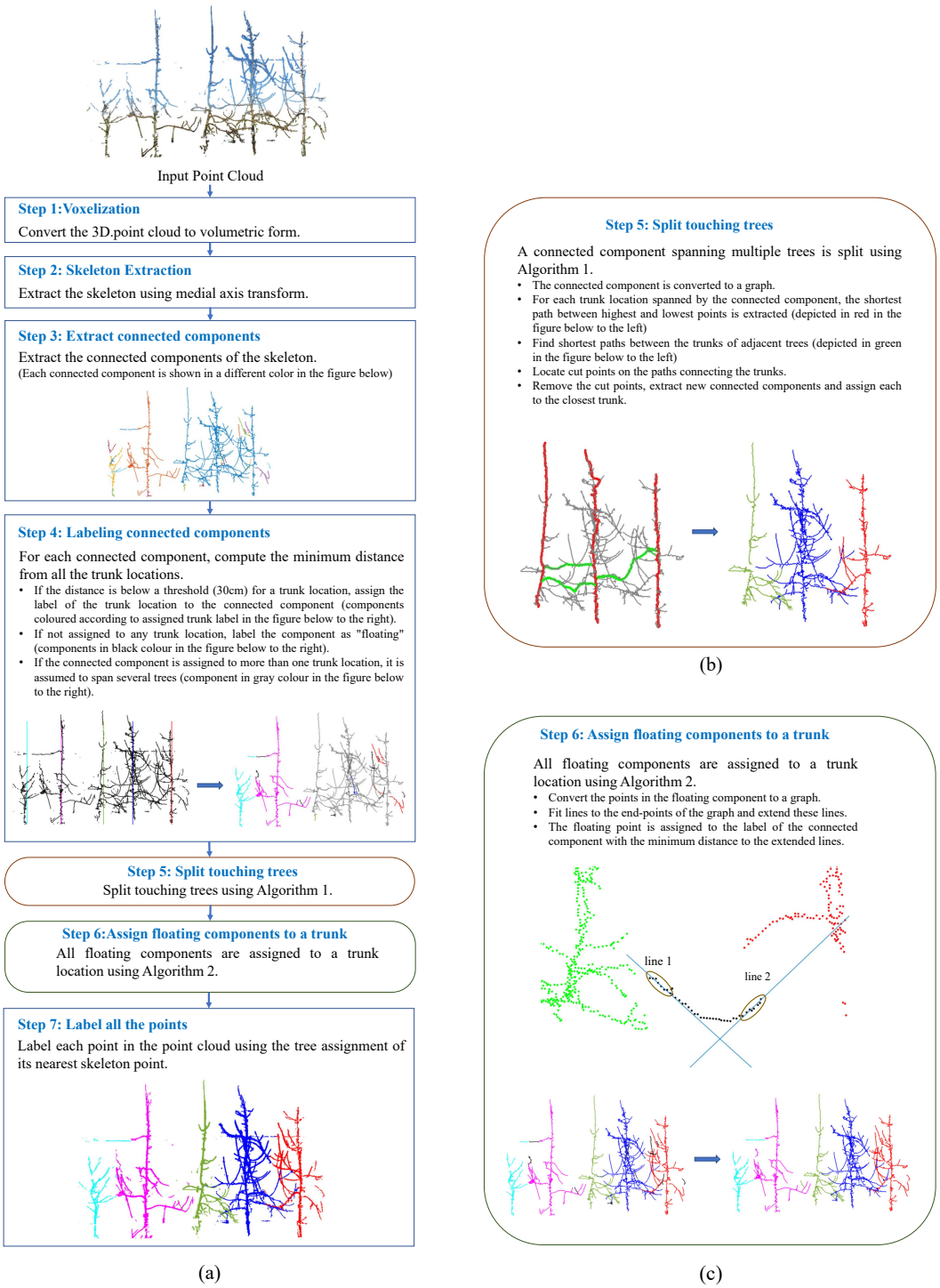


Figure 2.5 – (a) Block diagram for separating individual trees. (b) Illustration of Step 5 for splitting touching trees, (c) Illustration of Step 6 for labeling floating components.

Step 1: Voxelization The point cloud PC_w^{trees} is converted to binary volumetric form B_{trees} with voxel size $\hat{\Delta}_x = \hat{\Delta}_y = \hat{\Delta}_z = 5mm$.

Step 2: Skeletonisation The skeleton S_{trees} is extracted from B_{trees} using medial axis thinning algorithm given in [71].

Step 3: Extraction of connected components Connected components of the skeleton S_{trees} are extracted using flood fill algorithm [76]. We denote the set of connected components as $\mathcal{CC} = \{C_1, C_2, ..., C_{N_{comp}}\}$, where C_c is the c^{th} connected component and N_{comp} is the number of connected components. Fig. 2.5 shows each connected component in a sample S_{trees} in a different colour.

Step 4: Labeling connected components We compute the minimum distance of each connected component C_c to all trunk locations $\hat{L}_j = (0, \hat{y}_j, 0)$. If this distance is below 30cm, then we assign C_c to t_j. Fig. 2.5 (Step 4) gives the trunk locations as lines in different colours and the connected components coloured according to the assigned tree for a sample S_{trees}.

After this procedure a connected component might be assigned to 1) only one tree, 2) to multiple trees, or 3) none of the trees. If the connected component is assigned to multiple trees, it is assumed to be spanning several trees that are touching each other. We label the connected components not assigned to any tree as "floating". The floating components are shown in black colour in Fig. 2.5 (Step 4).

Step 5: Splitting touching trees For a connected component C_c spanning N_c trees $\{T_j\}$, $j = j_1^c, ..., j_{N_c}^c$, we run Algorithm 2. Before running the algorithm, we update SP_j, *main axis* of the j^{th} trunk, together with the points $\hat{p}_{j,top}$ and $\hat{p}_{j,bottom}$. Recall that $\hat{p}_{j,top}$ and $\hat{p}_{j,bottom}$ are the top and bottom points of the skeleton of the j^{th} tree trunk and SP_j is the shortest path connecting them. Fig. 2.5 (b) shows a connected component spanning three trees. The main axes of them are plotted in red colour, on the left.

Algorithm 2 takes as input the set of trees identities $\{T_j\}$, $j = j_1^c, ..., j_{N_c}^c$ spanned by the connected component C_c. For each adjacent tree pair T_j, T_{j+1}, the shortest path between their top points $\hat{p}_{j,top}$ and $\hat{p}_{j+1,top}$ is extracted. We call this path CP, the connecting path, which contains the touching point of branches from trees T_j and T_{j+1}. Each such path is searched for a cut-point to separate the connected adjacent trees. The cut-point is removed from the component C_c to break the connectivity at that point. The process is repeated and CP is updated until there remains no connected path between $\hat{p}_{j,top}$ and

$\hat{p}_{j+1,top}$. Fig. 2.5 (b) depicts the connecting paths CP between adjacent trees with green dots.

After all connecting paths are extracted and the cut-points are found and removed, detached connected components $\{C_{c,d}\}$; $d = 1, ..., N^c_{comp}$ of C_c are extracted. Then each connected component is assigned to the tree identity of the closest tree trunk. Fig. 2.5 (b) shows the detached connected components each coloured according to its tree identity.

It is challenging to determine the point where branches from two trees touch each other. Many architectural and morphological rules concerning apple tree branches can be incorporated. However, here, we use a simple heuristic based on the assumption that the point that changes direction along the z-axis (upwards or downwards) corresponds to a meeting point along the path. We select the global extremum of the z-coordinate as the cut-point of the connecting path.

Step 6: Assigning floating components to a tree The tree membership of a floating component C_c is determined using Algorithm 3. Before running Algorithm 3, we identify the set $\mathbb{C} = \{(C_1, \tau_1), ..., (C_{N_F}, \tau_{N_F})\}$ of connected components already assigned to a tree. Here $\tau_f \in \{t_1, ..t_{N_{trees}}\}$ is the tree identity of the component C_f. We determine the two closest components in $\mathbb{C}$ to the floating component C_c. If the distance to one connected component is more than 3 times than the distance to the other component, we assign the points in C_c to the tree identity of the closest component. Otherwise, we locate the end-points in C_c, fit lines to these end-points and extend these lines, as shown in Fig. 2.5 (c). The minimum distance of the two closest connected components to these lines are calculated. The floating component is then assigned to the tree identity of the connected component with the minimum distance to the extended lines. Algorithm 3 gives the details of the process.

Step 7: Labeling all points with tree identities After Steps 5 and 6, all connected components in S_{trees} are assigned to a tree label $\tau_c \in \{t_1, ..t_{N_{trees}}\}$. Recall that the connected components are extracted from the skeleton S_{trees} of the point cloud PC^{trees}_w. For each point $\hat{p} \in PC^{trees}_w$, we locate the closest component of S_{trees} and assign the tree identity of the component to the point $\hat{p}$. Fig. 2.5 (Step 7) shows the points of a sample PC^{trees}_w coloured according to their tree identities.

Recall that PC^{trees}_w is a subset of PC^{TP}_w, which is the winter point cloud aligned to the trellis-plane. To find the tree identities of the points in the calibrated winter cloud PC^C_w, we first apply $p = \hat{p}R^{-1}$ to each point $\hat{p} \in PC^{trees}_w$ with tree identity $\tau \in \{t_1, ..., t_{N_{trees}}\}$. Then, we retrieve the closest point $p_i \in PC^C_w$ to p and set $\tau_i = \tau$.

2.2.5 Apple detection

To detect apples, we applied simple colour thresholding to the calibrated 3D colour point cloud of the harvest scene PC_h^C. First, the RGB colours of points are converted to HSV (Hue, Saturation, Value) representation. The points in the hue range [0.15-0.2] are assumed to correspond to green/yellow apple points. The red apple points are assumed to be in the hue range [0-0.05] and [0.95-1]. The points with hue values in these ranges are retrieved and converted to volumetric form. The connected components of the volumetric form and their bounding boxes are extracted. The centers of these bounding boxes are mapped to the 3D space of PC_h^C and are considered to be the locations of detected apples. We denote the set of detected apples in a harvest scene as $\mathcal{A} = \{p_1^\alpha, ..., p_{N_{apples}}^\alpha\}$, where p_a^α is the location of a detected apple.

Although our apple detection approach is primitive, it provides recall rates in the range of 74% to 90% (see Section 2.3.2). This level of detection success is sufficient to demonstrate the effectiveness of our approach for assigning retrieved apples to their respective trees.

2.2.6 Assigning apples to individual trees

The main objective of this work is to automatically assign detected apples to their respective trees; i.e. to determine the tree identity $\tau_a \in \{t_1, ..., t_{N_{trees}}\}$ of each detected apple $p_a^\alpha \in \mathcal{A}$. To this end, we align calibrated winter cloud PC_w^C and summer cloud PC_h^C and assign apple p_a^α detected from PC_h^C to the tree identity of the closest branch point in the aligned winter cloud.

Since both point clouds were transformed, through calibration, to a common reference frame with the origin at the base of a reference tree (see Annex B for details), they are already initially aligned. We apply the standard Iterative Closest Point (ICP) algorithm [77] to improve the alignment. Point to point metric is used to minimise the alignment error. ICP returns the transformation parameters; a rotation matrix R^{wh} and a translation vector T^{wh} that align the points in PC_w^C to the points in PC_h^C:

$$PC_{wh}^C = \{p_i' = p_i R^{wh} + T^{wh} : (p_i \in PC_w^C) \ \ \& \ \ \tau_i \in \{t_1, ..., t_{N_{trees}}\}\}. \tag{2.20}$$

Once the transformed winter point cloud PC^C_{wh} is obtained, the closest branch point in PC^C_{wh} labeled with a tree identity to the apple location p^α_a is retrieved:

$$i^* = \arg \min_{p'_i \in PC^C_{wh}} \|p'_i - p^\alpha_a\|; \tag{2.21}$$

and the tree identity of apple a is set as

$$\tau_a = \tau_{i^*}. \tag{2.22}$$

2.2.7 Ground truth and evaluation metrics

To provide ground truth for evaluation of our semantic segmentation scheme, we manually labeled each point $p_i \in PC^C_w$ with one of the following semantic labels: $\gamma^{GT}_i \in \{$*"Tree trunk"*, *"Branch"*, *"Trellis wire+Water pipe"*, *"Support pole"*$\}$. We used Cloud Compare (2.11, GPL software, 2020) to label the point cloud. Fig. 8.1-(a) shows a sample winter scene with points coloured according to their manually annotated ground truth labels.

We evaluated the performance of the semantic segmentation module described in Section 2.2.4 using Recall (Re), Precision (Pr), F1 score ($F1$), Intersection over Union (IoU), and Class Accuracy (CA), defined as

$$Re = \frac{TP}{TP + FN} \tag{2.23}$$

$$Pr = \frac{TP}{TP + FP} \tag{2.24}$$

$$F1 = 2 \times \frac{Pr \times Re}{Pr + Re} \tag{2.25}$$

$$IoU = \frac{TP}{TP + \ FN + \ FP} \tag{2.26}$$

$$CA = \frac{TP + TN}{TP + TN + FP + FN}\ , \tag{2.27}$$

where TP, TN, FP and FN, correspond to the number of True Positives, True Negatives, False Positives, and False Negatives, respectively. These cases for the *"Tree trunk"* are determined as follows:

$$\text{Case}_i = \begin{cases} \text{True Positive} & \text{if } \gamma_i = \gamma_i^{GT} = \textit{"Tree trunk"} \\ \text{True Negative} & \text{if } (\gamma_i \neq \textit{"Tree trunk"}) \;\&\; (\gamma_i^{GT} \neq \textit{"Tree trunk"}) \\ \text{False Positive} & \text{if } (\gamma_i = \textit{"Tree trunk"}) \;\&\; (\gamma_i^{GT} \neq \textit{"Tree trunk"}) \\ \text{False Negative} & \text{if } (\gamma_i \neq \textit{"Tree trunk"}) \;\&\; (\gamma_i^{GT} = \textit{"Tree trunk"}), \end{cases} \tag{2.28}$$

where γ_i^{GT} is the ground truth label of point p_i and γ_i is the label predicted by our automatic semantic segmentation scheme. The cases for *"Support pole"* and *"Trellis wire+Water pipe"* are obtained in a similar manner.

In order to assess the performance of the colour-based apple detection approach, we manually marked the apple positions in the harvest point clouds and obtained the set of points $\mathcal{A}^{GT} = \{p_g^{\alpha,GT}\}; g = 1, ..., N_{apples}^{GT}$. In Fig. 8.1-(b), a harvest point cloud with ground truth apple positions is shown. For evaluation, we used Recall (Re) and Precision (Pr) metrics, defined in Eq. (2.23) and (2.24). Here, the True Positives correspond to the cases where a ground truth apple is correctly localised. The False Positives are wrong detections returned by the algorithm. The False Negatives correspond to the ground truth apple locations missed by the algorithm. A detection $p_a^{\alpha} \in \mathcal{A}$ is considered a True Positive if there is a ground truth apple $p_g^{\alpha,GT} \in \mathcal{A}^{GT}$ such that $\|p_a^{\alpha} - p^{\alpha,GT}\| < 10cm$ and there is no other detected apples closer to $p^{\alpha,GT}$. We pair the indices (a, g) to indicate that $p_a^{\alpha} \in \mathcal{A}$ corresponds to $p_g^{\alpha,GT} \in \mathcal{A}^{GT}$. The number of False Positives and False Negatives are then calculated as:

$$FP = N_{apples} - TP \tag{2.29}$$

$$FN = N_{apples}^{GT} - TP \tag{2.30}$$

where TP is the number of True Positives, N_{apples} is the number of detected apples in $\mathcal{A}$ and N_{apples}^{GT} is the number of ground truth apples in $\mathcal{A}^{GT}$.

The end result of our apple assignment pipeline is the tree identity of each detected apple, indicating which tree it belongs to. In order to evaluate assignment performance, we provided the correct tree identities of the ground truth apples via manual inspection; i.e. we determined $\tau_g \in \{1, .., N_{trees}\}$ for each $p_g^{\alpha,GT} \in \mathcal{A}^{GT}$. We computed the accuracy of the apple assignment (ACC) as the ratio of the number of correctly assigned true positives TP_C to the total number of true positives TP in the scene:

$$ACC = \frac{TP_C}{TP} \tag{2.31}$$

A detection $p_a^{\alpha} \in \mathcal{A}$ is considered to be a correctly assigned true positive if its tree

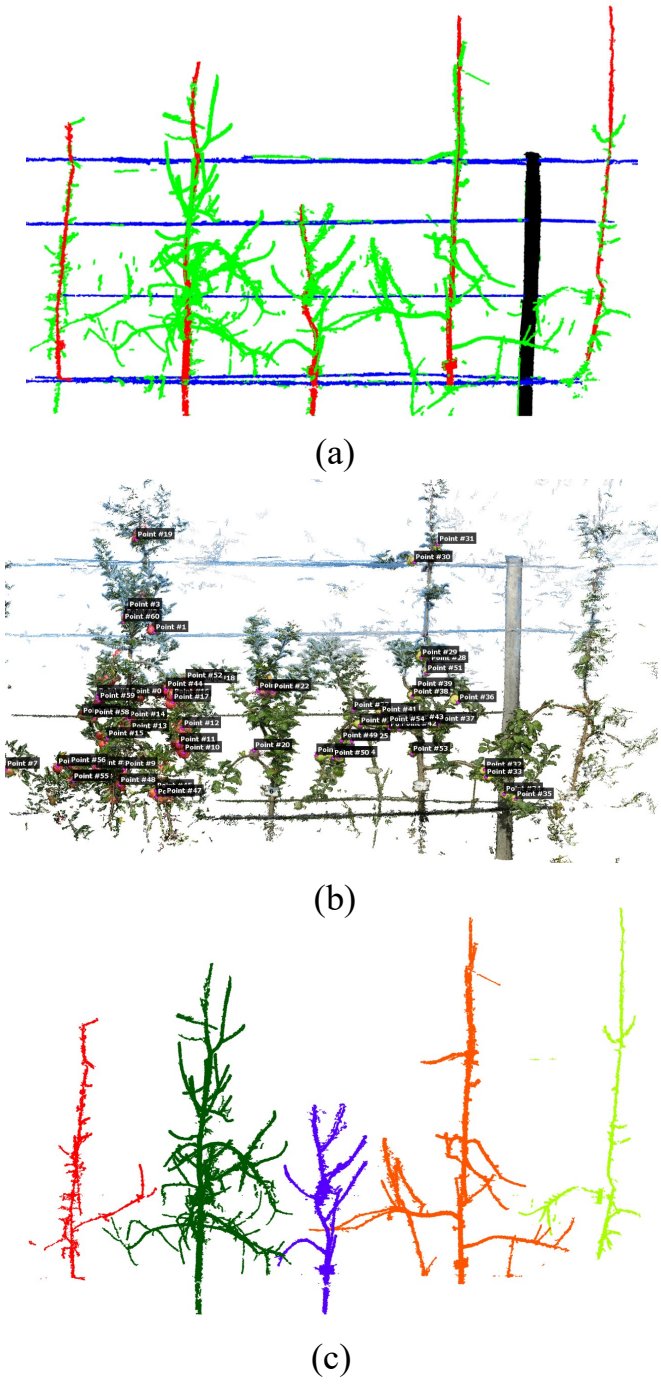


Figure 2.6 – Ground truth. (a) Manually labeled point cloud for assessment of trellis wire, tree trunk and support pole detection, (b) Harvest point cloud with ground truth apple locations, (c) Point cloud manually segmented to individual trees.

identity τ_a, determined by Eq. (2.21) and (2.22), is equal to the tree identity τ_g of its matched ground truth apple $p_g^{\alpha,GT} \in \mathcal{A}^{GT}$.

Recall that we assigned each apple $p_a^{\alpha} \in \mathcal{A}$ to the tree identity τ_{i^*} of the closest branch point p_{i^*} in the aligned winter cloud through Eq. (2.21) and (2.22). In order to decouple the apple assignment errors due to branch deformation between winter and summer trees and errors due to our automatic tree separation method, we performed the apple assignment procedure on two types of data:

1. Manually Separated: We manually separated the winter point clouds into individual trees and provided the ground truth tree identities $\tau_i^{GT} \in \{1, ..., N_{trees}^{GT}\}$ of the trunk and branch points in the winter cloud. We used CloudCompare (2.11, GPL software, 2020) for annotation. One example is shown in Fig. 8.1-(c).
2. Automatically Separated: We used the tree identities $\tau_i \in \{1, ..., N_{trees}\}$ of the trunk and branch points in the winter cloud predicted by our automatic tree separation procedure.

2.3 Results

We first report the results of the semantic segmentation method, which detects the trellis wires, tree trunks and support poles. Then, we provide the performance of the apple detection method and the assignment procedure of apples to individual trees in the scene.

2.3.1 Evaluation of detection of trellis wires, tree trunks and support poles

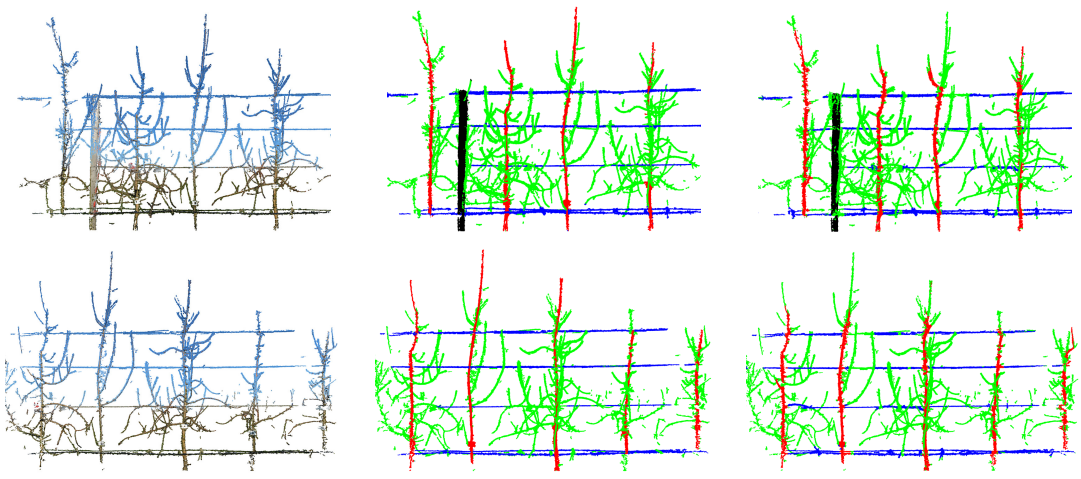
In Fig. 2.7, we give visual results of our semantic segmentation method for two winter scenes. The visual results for all the seven scenes can be found in Annex C. We can observe that all the trees in the scenes of the apple orchard, the trees were correctly localised. The number of detected tree trunks and the actual number of trees were equal for all seven scenes; $N_{trees} = N_{trees}^{GT}$.

Table 2.2 provides quantitative evaluation of our semantic segmentation method. In Fig. 2.8, the results are given as bar graphs. The recall and precision values for the trellis wires are satisfactory. All the support poles in the scenes were correctly identified and segmented with over 90% success. The recall rate for the trunks is over 90% for all

Algorithm 2: Separation of a connected component into multiple trees

Data: C_c: Connected component spanning multiple trees;
$\{T_j\}$: Trees spanned by C_c; $j = j_1^c, ..., j_{N_c}^c$;
$\{SP_j\}$: Main axes of trees;
$\{\hat{p}_{j,top}\}$: Top points of the main axes
Result: $\{C_{c,d}\}$: Detached connected components each assigned to a tree;
$d = 1, ..., N_{comp}^c$

```
for j ← j_1^c to j_{N_c}^c − 1 do
    CP ← ∅ ;
    Extract the shortest path CP between the points p̂_{j,top} and p̂_{j+1,top};
    while CP ≠ ∅ do
        CP ← (CP \ SP_j) \ SP_{j+1} ;
        Select the global extremum of the z-coordinate in CP as the cut-point;
        Remove the cut-point from C_c ;
        Extract the shortest path CP between the points p̂_{j,top} and p̂_{j+1,top};
Apply connected components to C_c to obtain {C_{c,d}} ;
Assign each connected component C_{c,d} to the closest tree trunk;
```



(a) (b) (c)

Figure 2.7 – (a) Calibrated point clouds, (b) Manually generated Ground Truth (cyan:trellis wires, red: tree trunks, black: support poles), (c) Semantic labels obtained by our method for automatic detection of trellis wires, tree trunks, and support poles

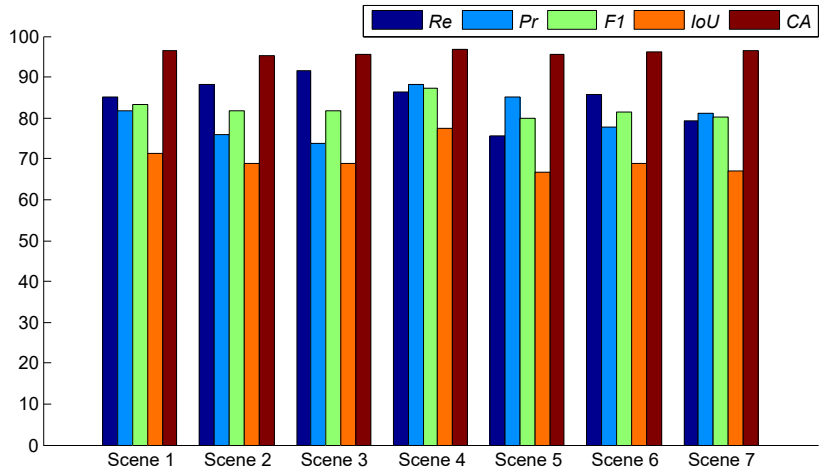
Algorithm 3: Assignment of a floating branch to a neighbouring tree.

Data: C_c: Floating connected component;
$\mathbb{C} = \{(C_f, \tau_f)\}$: Connected components already assigned to a tree
$(f = 1, 2, ..., N_F)$
Result: $\tau_c \in \{t_1, ..t_{N_{trees}}\}$: Tree identity of C_c

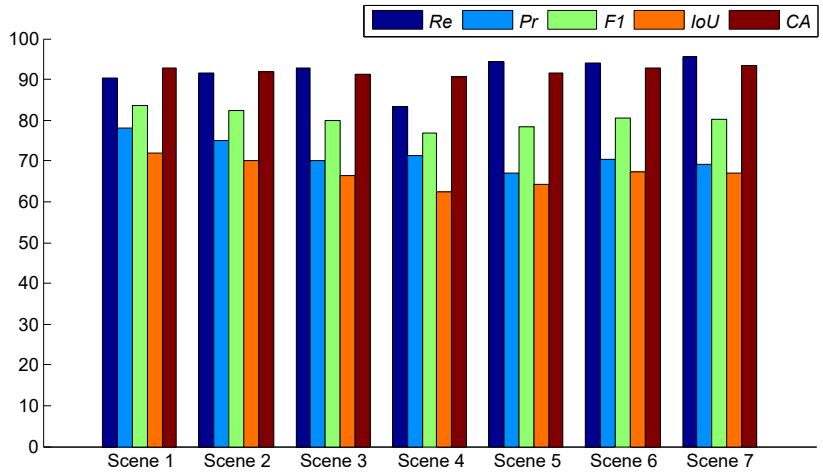
for $f \leftarrow 1$ *to* N_F **do**
    Calculate the minimum distance d_f between the points in C_c and the points in C_f;
$d^{F1} \leftarrow$ Minimum of d_f; $d^{F2} \leftarrow$ Next minimum of d_f;
$C^{F1} \leftarrow$ Component with d^{F1} ; $C^{F2} \leftarrow$ Component with d^{F2};
$\tau^{F1} \leftarrow$ Tree identity of C^{F1} ; $\tau^{F2} \leftarrow$ Tree identity of C^{F2};
if $\frac{d^{F2}}{d^{F1}} > 3$ **then**
    $\tau_c \leftarrow \tau^{F1}$;
else
    Extract the end-points p_e of C_c, $e = 1, 2, ..., N_e$;
    for $e \leftarrow 1$ *to* N_e **do**
        Extract K nearest neighbours of p_e with $K = 10$;
        Fit a line l_e to the neighbours;
        $d^{eF1} \leftarrow$ Minimum distance of the points in C^{F1} to the line l_e;
        $d^{eF2} \leftarrow$ Minimum distance of the points in C^{F2} to the line l_e;
    if $min\{d^{eN1}\} < min\{d^{eN2}\}$ **then**
        $\tau_c \leftarrow \tau^{F1}$
    else
        $\tau_c \leftarrow \tau^{F2}$;

Table 2.2 – Performance of the method for detection of trellis wires, tree trunks and support poles in terms of Recall (Re), Precision (Pr), F1 score ($F1$), Intersection over Union (IoU), and Class Accuracy (CA). NP is for non-present.

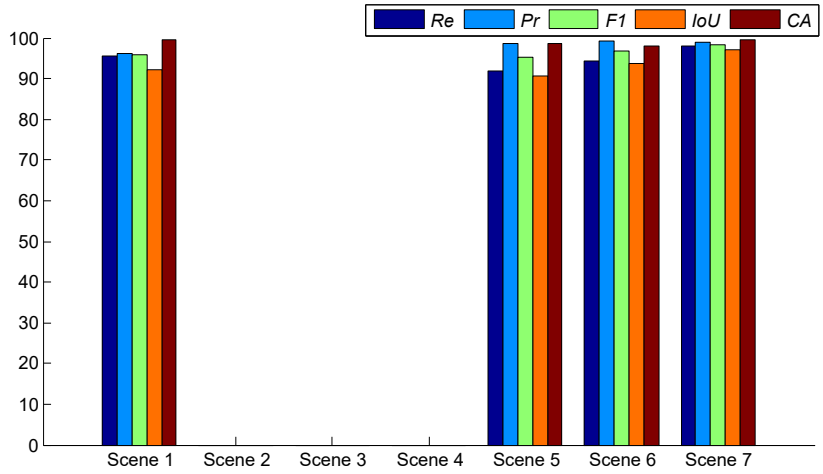
	Trellis wires				
	% Re	% Pr	% $F1$	% IoU	% CA
Scene 1	84.98	81.61	83.26	71.32	96.42
Scene 2	88.16	76.01	81.63	68.96	95.19
Scene 3	91.48	73.65	81.61	68.93	95.52
Scene 4	86.48	88.20	87.33	77.51	96.88
Scene 5	75.47	85.21	80.04	66.73	95.64
Scene 6	85.82	77.75	81.59	68.90	96.06
Scene 7	79.24	81.16	80.19	66.93	96.48
	Tree trunks				
	% Re	% Pr	% $F1$	% IoU	% CA
Scene 1	90.26	77.97	83.67	71.92	92.83
Scene 2	91.49	74.89	82.36	70.01	91.77
Scene 3	92.77	70.03	79.81	66.40	91.31
Scene 4	83.23	71.23	76.76	62.29	90.55
Scene 5	94.25	67.03	78.34	64.40	91.56
Scene 6	94.02	70.39	80.51	67.37	92.78
Scene 7	95.47	69.19	80.24	66.99	93.27
	Support poles				
	% Re	% Pr	% $F1$	% IoU	% CA
Scene 1	95.50	96.24	95.87	92.07	99.44
Scene 2	NP	NP	NP	NP	NP
Scene 3	NP	NP	NP	NP	NP
Scene 4	NP	NP	NP	NP	NP
Scene 5	91.83	98.74	95.16	90.77	98.65
Scene 6	94.44	99.26	96.79	93.78	98.15
Scene 7	97.94	98.96	98.45	96.94	99.64



(a) Trellis wires



(b) Tree trunks



(c) Support Poles

Figure 2.8 – Performance of the method for detection of trellis wires, tree trunks and support poles in terms of Recall (Re), Precision (Pr), F1 score ($F1$), Intersection over Union (IoU), and Class Accuracy (CA) (in %).

but one scene, meaning that most of the trunk points are retrieved. The precision rates are satisfactory for our purposes. The less than perfect precision is due to the fact that branching points close to the tree trunks are also classified as trunks by our method.

It should be recalled that our aim is not to provide a perfect segmentation, but rather 1) to detect and remove the trellis wires to break connectivity between adjacent trees, 2) to locate the tree trunks correctly to be able to separate individual trees, and 3) to remove the support poles. For the purposes of our application, these aims were achieved with this level of automatic point labeling of the scene.

2.3.2 Evaluation of apple detection and assignment to individual trees

The precision and recall values obtained with colour-based apple detection are given in Tab. 2.3 and, also presented in Fig. 2.9 as a bar graph. Despite the simplicity of the detection approach, we achieved over 90.75% recall; i.e. most of the apples in the ground truth were retrieved. The false negatives occurred since we did not post-process the connected components for resolving clusters of apples. The over-detection (precision 65,37%) can be explained by the sensitivity of the colour-based algorithm and the lack of shape-based apple verification. Fig. 2.11 (a) and (b) visually illustrate the performance of our apple detection method on two sample scenes.

Table 2.3 – Apple detection performance in terms of Recall (Re) and Precision (Pr).

3D scenes	% Re	% Pr
Scene 1	74.50	61.29
Scene 2	87.34	62.16
Scene 3	88.54	58.21
Scene 4	90.00	48.64
Scene 5	90.62	58.58
Scene 6	77.41	65.62
Scene 7	80.85	66.66

Our main task is to correctly assign the detected apples to the individual trees they belong to. As we have stated earlier, we performed the assignment procedure to two types of data: 1) The winter point clouds which are manually segmented to individual trees, and 2) The winter point clouds where the trees are segmented using our automatic

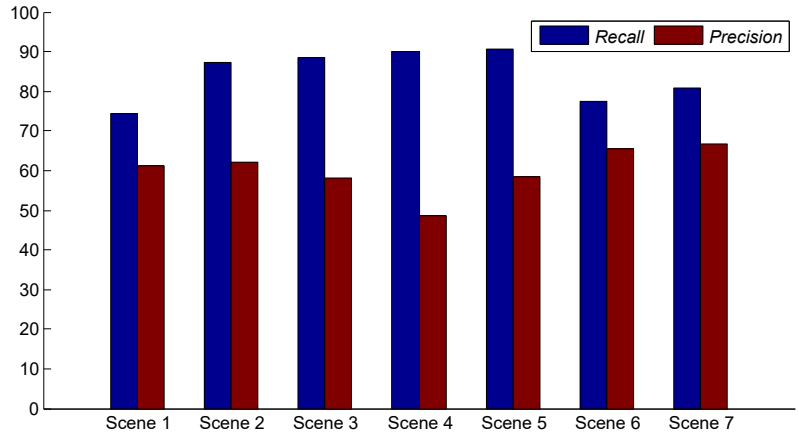


Figure 2.9 – Apple detection performance in terms of Recall (Re) and Precision (Pr) (in %).

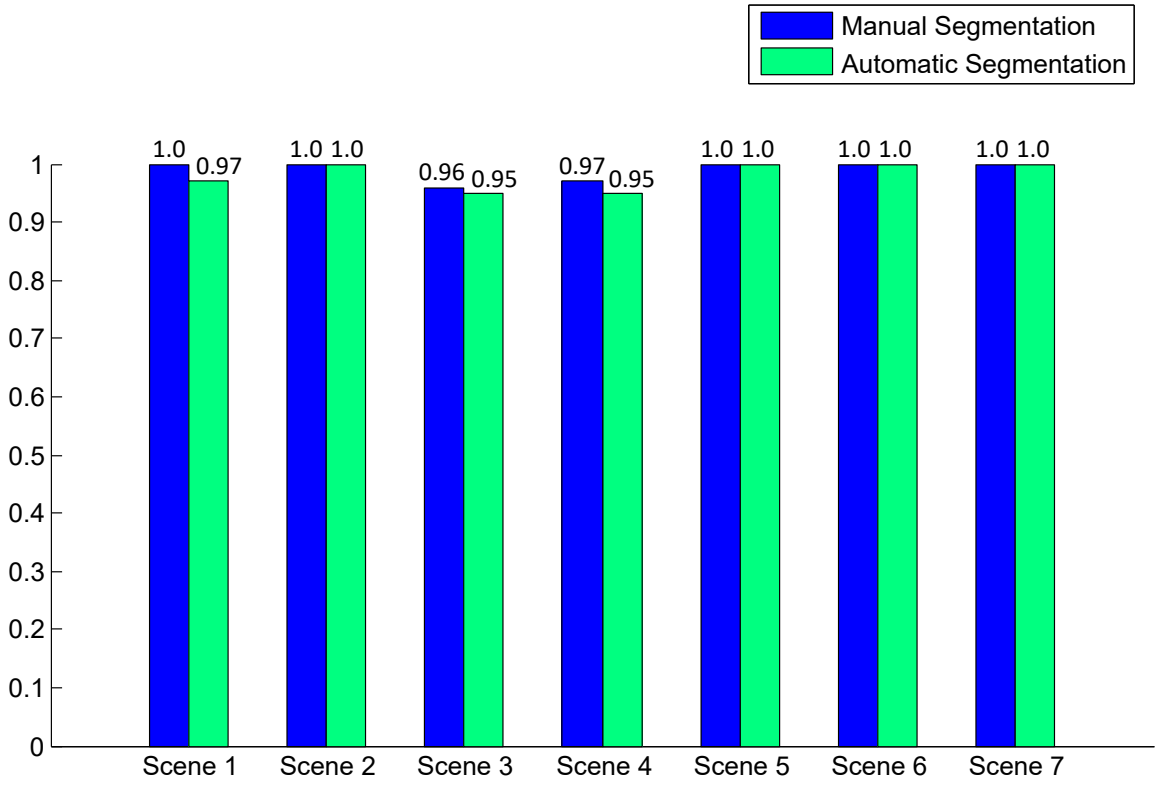


Figure 2.10 – Accuracy of assigning apples to the correct apple trees in 3D models.

tree separation method. Fig. 2.10 shows the assignment accuracy (ACC) on both type of data. The performance is high for both cases (100% on four scenes). With automatic tree separation, a performance drop of less than 3% is observed, demonstrating that our automatic pipeline was able to detach individual trees and correctly assign the detected

apples.

Fig. 2.11 (c) and (d) show the registration result of winter and harvest point clouds for two sample scenes. Each separated tree in the winter clouds is shown in a different colour. In Fig. 2.11 (e) and (f), the detected apples are shown with the colour of their corresponding tree labels.

2.3.3 Processing time

The average processing times, together with the standard deviations, for the steps of the pipeline are provided in Tab. 2.4. Given a 3D point cloud reconstructed through SFM, the total time for processing the point cloud is approximately 3 minutes. We also give the characteristics of the machine we used to process the data in Tab. 2.5.

Table 2.4 – Average processing times of the steps of the pipeline in seconds.

Step	**Processing time**
Calibration and ROI extraction	16.53 ± 2.02 s
Detection of tree trunks, trellis wires and support poles	50.20 ± 2.42 s
Tree separation	18.41 ± 2.56 s
Apple detection and assignment	118.40 ± 3.78 s
Total	203.54 ± 10.78s

Table 2.5 – Machine characteristics.

RAM	**Processor**	**GPU membership**
64 GB	Intel® Xeon® Silver 4114 CPU @2.20 GHz and 2.19 GHz (2 processors)	Quadro P4000 GPU of 8 GB

2.4 Discussion

The full pipeline presented and tested in this manuscript achieves great performance for assigning apples to individual trees in dense orchards. The main strategy is aligning summer and winter point clouds. The sub-steps of the pipeline, for which we chose standard approaches for implementation, are open to improvement for further performance increase.

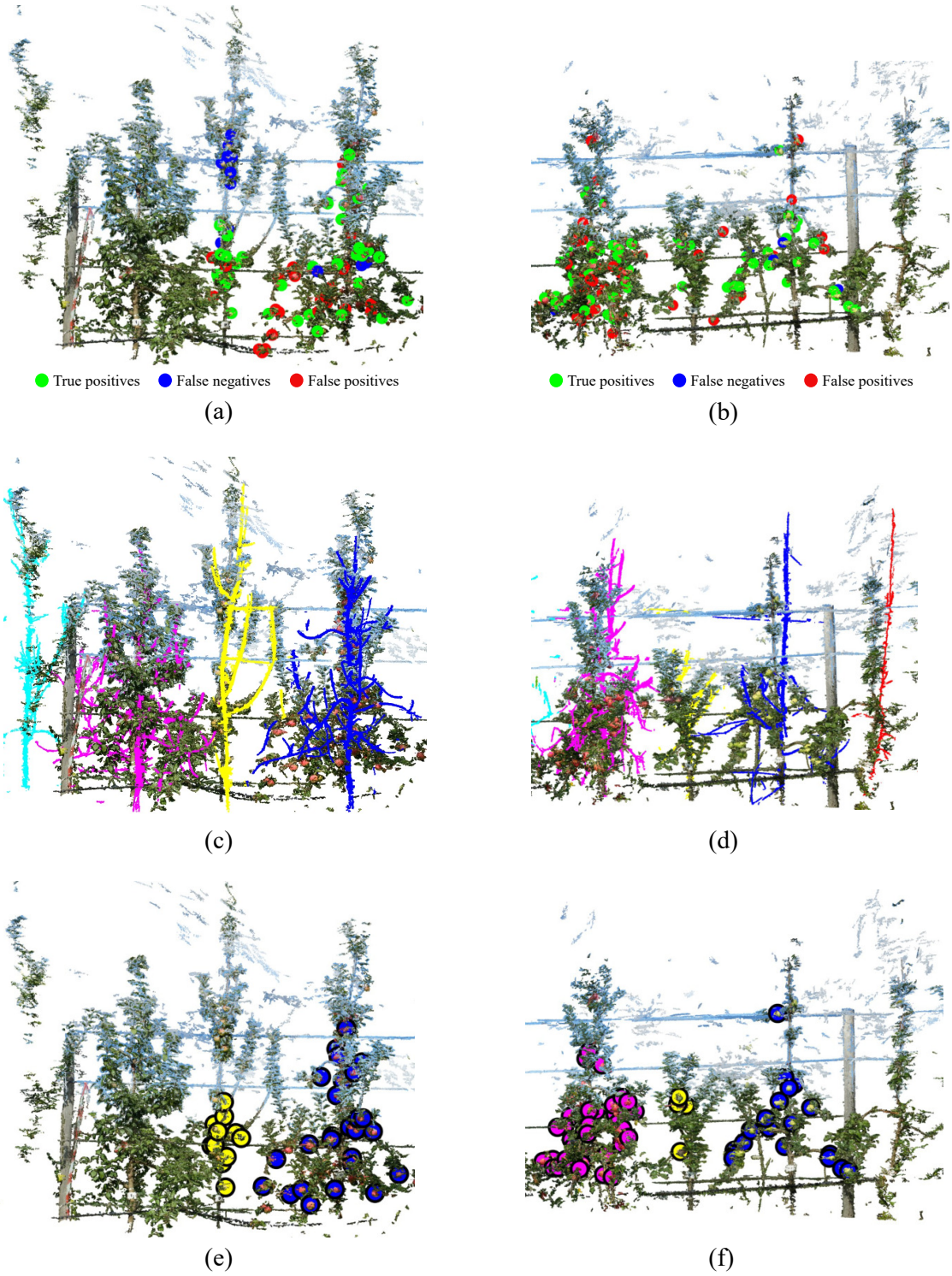


Figure 2.11 – (a), (b) True positives, false negatives, and false positives obtained with colour-based apple detection method for two sample scenes. (c), (d) Registration of harvest and winter clouds for the two scenes. Each separated tree is shown with a different colour. (e), (f) Assignment of true positives to their corresponding trees for the two scenes.

Images were acquired manually with a standard camera. This is a rather time consuming process for producing hundreds of images per tree. The speed of acquisition can be increased and the amount of images can be optimised by a drone with a camera or a land robot with multiple cameras and automatic navigation via GPS localisation [78]. The object of reference for calibration and registration of the summer and winter point cloud was chosen to be the X-Rite ColorChecker, since it is a standard tool in the computer vision community. In principle, any reference object with a distinctive geometric pattern could serve the same purpose.

The deformation we observed with our data (young trees of four years old) becomes even more pronounced for older trees. Registration of winter and summer calibrated point clouds could be performed efficiently with non-rigid registration while dealing with older trees, where the deformation during summer could be larger due to increased fruit load. Non-rigid registration is widely used in medical imaging when data from two different modalities, such as MRI and X-Ray images, should be registered. Non-rigid deformation between the image sets are commonly observed due to movement of the patient or artifacts of the imaging systems. The literature on non-rigid registration of medical images can thus be revisited for our plant imaging problem [79]. To avoid having a too large exploration space for this non-rigid registration, one could also use botanical and physical knowledge on the development of trees. The size and weight of the fruits is important because it can cause arching of the branches, therefore, a deformation of the architecture. Another factor that alters the architecture is the secondary growth of the branches. Expert knowledge on such processes can be used to constrain the deformation space and fix the hyperparameters of the non-rigid registration algorithms.

We based the evaluation of the registration of winter and harvest point clouds on the rate of correct assignment of the apples to their corresponding trees. A thorough evaluation of the matching error is possible through computing distances between corresponding keypoints in the two point clouds. However, such a procedure necessitates manually establishing ground truth correspondences between well-located keypoints. It will be a worthy endeavor to measure the registration error and decouple errors due to changing structures in the trees and errors due to the limitations of the matching method. The decoupling can be done by locating keypoints both on the fixed structures (e.g. on the trellis structure) and on the trees (e.g. branching locations).

Another issue is the applicability of our method to other orchards and other fruit trees. The registration method we proposed relies on the initial alignment of two point clouds,

each of which are calibrated separately. The calibration procedure operates on three requirements: 1) There should be a reference pattern (in our case, the ColorChecker) in place to recover the scale and to establish upwards and leftwards directions; 2) The trees should be organised in a row to establish the -y- direction; and 3) The relative position of a designated tree to the reference pattern should be known to locate the origin. As long as these requirements are met, our calibration procedure will be applicable to fruit orchards organised as rows.

The parameters of our method were fixed for all the scenes we processed in our experiments. We avoided fine tuning the parameters required by some standard procedures such as voxelization and Hough line extraction via trusting the added robustness of our further processing steps. For example, for the grid size for voxelization, the choices of both 5mm and 10mm work effectively for extraction of a representative skeleton, although the latter gives a coarser skeleton. The grid size should not be too small compared to the point resolution or to cause computational overload; and it should not be too large to cause merging unconnected structures into one voxel. The dependency on parameters in Hough transform on extracting candidate lines is controlled by the line merging step to extract the expected four trellis lines. These parameters will not require adjustment for application to apple orchards with a similar trellis-wire organisation.

For other parameters, such as the inlier distance to detect the trellis-plane (1cm – to cover the thickness of a trellis wire), trunk candidate search distance (5cm – to cover the thickness of a trunk), and the minimum distance between successive trellis wires (30cm), we used the actual metric quantities of these structures, again avoiding fine-tuning. For application to other orchards, these parameters can be adjusted according to geometric priors such as trellis-wire thickness, distance between trellis-wires and expected trunk thickness.

The point clouds used in our experiments spanned four to five trees reconstructed using manually acquired images. The extension of the method to process the whole tree row in an automatic manner is possible through installation of multiple low-cost reference patterns along the row. Such patterns can also be installed per tree, together with a QR code as a tree identifier. Using multiple patterns will help reduce the error in the directions of the fixed frame through multiple estimations. They can also be used to reset the reconstruction process of SFM at predetermined locations to avoid the accumulative drift through providing multiple point clouds covering overlapping regions along the tree row. These point clouds can be processed separately, and if necessary, can be calibrated

and registered to obtain the entire row model. Our future work includes installation of one board with printed ColorChecker structure and a QR code at each tree and the analysis of the performance of suggested solutions.

In this work, we used connectivity analysis and simple heuristics to disconnect touching trees. Alternatively, the identification of each tree unit can be achieved using the architectural criteria specific to each tree. They are linked to the basic architectural models defined for each taxon [80]. They are supplemented by the growth conditions specific to each tree and are assessed by the diameter, length, age and branching angles of the branches but also by the location of inflorescences and fruits.

The apple detection algorithm chosen in this work was extremely simple and it will be necessary to revisit the huge literature on apple detection to improve the performance, specially on groups of apples or to reduce the amount of false positives. State-of-the-art methods employing deep learning architectures, such as [81, 82, 83] have been highly successful. It is possible, through the projection parameters estimated by the multi-view reconstruction process, to combine image-based fruit detection methods with 3D point reconstruction of the scene. Indeed, [84] proposed a method where apple detection is performed on 2D images and detected apples in images are used to segment the apples in the 3D reconstruction. A similar method that establishes correspondences between 3D points and the pixels in 2D images and classifies each 3D point as apple/non-apple can be integrated into our scheme.

One of the common issues in image-based fruit detection methods is the occlusion of fruits caused by leaves and other fruits [85]. Imaging trees from various viewpoints greatly increases the possibility that occluded fruits will be visible in more than one image. However, association of the same fruits occurring in different images is necessary for 2D image-based methods to avoid double-counting. The 3D reconstruction pipeline we used in this work greatly alleviates the occlusion problem by utilising multi-view reconstruction and inherently registering multiple sightings of a single fruit. A further research question can be formulated as the systematic investigation of the severity of occlusion with employing the ground truth count of harvested apples and the analysis of the impact of the imaging systems and acquisition protocols on capturing heavily occluded fruits.

Our pipeline enables the assignment of apples to the trees that bear them. This makes it possible to assess the production and the quality of the fruiting body in variety testing applications and also in the agronomic management of orchards. We know that fruiting

is the expression of primary and secondary growth followed by a flowering process with the formation of inflorescences and flowers. One, two or three years old axes that are part of the overall architecture of the tree carry these inflorescences. In this biological process, Laury et al. [86, 87] showed the importance of the age of branches, their position in the architecture and secondary growth on the fruit load of the tree. Our pipeline opens the way to acquire data at different developmental stages, analyse the architecture of individual trees, track primary and secondary growth, determine their axes of different ages. The location of the fruits and the identification of the characteristics of the axes that carry them, supplemented by a temporal monitoring of the architectural development could make it possible to obtain information to manage and improve the agronomic management of fruit trees.

2.5 Conclusion and perspectives

In this chapter, we presented a pipeline to reconstruct 3D architecture of trees, detect fruits and associate each fruit to the correct tree. The accuracy of correct assignment of fruits reached 97% on seven apple tree varieties. This work is a foundational step for the automation of apple traits in the orchards.

The pipeline is rather time-consuming in its current state, both in terms of data acquisition and data processing. A logical perspective would now be to increase further the throughput of the acquisition protocol, ergonomy and speed of the computational steps. These engineering tasks fall outside the scope of this PhD.

As an alternative perspective, we envision a different approach for image acquisition in the orchard with cameras. In this new approach, a full 3D reconstruction of the scene is not targeted. Instead, a video locally scanning the object of interest is produced via a connected stick handheld by the variety testing examiner. Such approach will have several benefits, such as:

- avoiding reconstruction and processing of 3D models, which is heavy;
- the approach is adapted to the protocols of examiners as they can acquire data while performing their examinations;
- the annotation will be performed within the acquisition of data which constitutes a major gain of time.

Chapter 3

Toward the use of machine learning in variety testing

3.1 Introduction

During the distinctness test of the DUS variety testing, the examiners decide whether the candidate variety is different from the reference ones. The specificity of the distinctness test is that the measurements of traits follow a strict scoring system set by UPOV. On the other hand, the limitation of the decision-making in the current practices in the distinctness test remains in the comparison between the varieties. Often, examiners use Gaussian statistical models while there is no certainty that the measurements follow a Gaussian distribution.

Machine learning can fit well to shift toward automated measurement for the following reasons. Using data annotated by examiners, measurements of traits can be automated while respecting the scoring scale set by UPOV. In addition, the distinctness test can be addressed as a classification problem. After automating the measurement of each trait, machine learning can process the high-dimensional space where all traits of the candidate and the reference varieties are measured and quantify the separability between the varieties. Another advantage of machine learning is that the decision about the distinctness is made based on the data itself (images and machine learning descriptors) and not relying on an invalidated statistical hypothesis.

In the first section of this chapter, we introduce a methodology to incorporate supervised machine learning in the distinctness test of apple varieties based on color information.

In the second section, we propose a new alternative methodology based on non-supervised machine learning suitable for variety testing protocols. We demonstrate its potential in the distinctness test of apple varieties based on the shape information. The material presented in this chapter has been published in [88] and [89].

3.2 Apple color characterization

Color difference evaluation has been extensively studied by CIE which provides a set of formulae found in accordance with human perception for more or less simple uniform images with controlled background and illumination [90]. In our case, we face the question of differentiation of a large set of textured images of apples which characterizes a variety from another large set of images. Comparing pairs of individual images of apples with the standard CIE approach would be a rather brute force approach which is not followed by the experts in charge of variety examinations. We rather decided to mimic their current visual approach of these experts, which is to compare two images each containing a large set of sample apples and decide whether the two sets belong to the same variety or not. We thus address the problem as a statistical one and consider distinctness as equivalent to deciding if a set of 3D color histograms representing an apple variety is sufficiently "distant" from a set of other 3D color histograms corresponding to other sets of apples.

3D color histograms of high-resolution images are dense point clouds and are difficult to be properly visualized for assessment of the density variations [91]. As a consequence, 3D color histograms have been characterized in many ways in the literature [92]: histogram intersection, dominant color descriptor, color correlogram, color co-occurrence matrix, dominant color descriptor, chromaticity, fractal dimension [93, 94] and fractional anisotropy [95]. Color histogram matching can also be performed with the help of metrics developed in information theory such as the Shannon mutual information and their variants [96, 97] or optimal transport [98, 99] to probe the discrepancy between two statistical distributions. For a first step of variety testing in the domain of machine learning, we decided to select a small subset of these features, since we do not claim to provide an exhaustive analysis of the appropriate features. Here, we rather aim to set the machine learning scheme and give proof of feasibility in the automation of distinctness.

From the computer vision generic perspective, the applied task considered in this communication may relate to identification problems such as image retrieval or object tracking [100, 101, 102, 99]. In our case again, the identification problem is related to a population of images, while in image retrieval or object tracking [92, 102] the task mostly targets identifying single image or objects. Fruit characterization is a field of machine vision in itself, which has received considerable attention either for species identification or quality control (see [103, 104, 105, 106] for reviews). Again the statistical situation considered here is different since the goal is not to sort each individual apple but to

determine if a set of apples can be considered as distinct or not from others. Most related works to what we propose can be seen as [107, 108] where supervised machine learning is proposed to classify existing varieties. In our case, while considering other crops and testing other features, we deal with the situation where a new set of plant has to be identified as similar to existing ones or distinct up to the point where it deserves to be designated as a new variety.

3.2.1 Materials and methods

Images Acquisition

The acquisition of the images of the different varieties of apples was carried out with the help of a conveyor machine, allowing to move the fruits in translation while carrying out a rotation (see Fig. 3.1). A camera, located at the top of the conveyor belt of the machine with a perpendicular viewing direction, took pictures of the apples in rotation, which allowed us to have multiple images providing almost an entire coverage of each apple. Approximately 9 to 10 views of the same apple were captured thanks to this rotation-translation process. These multiple views are important since apples may have several major colors on their skins. With the standard visual approach, experts have to manually rotate the apples to have a full perception of the variation of color on a single apple. Here, the machine presented in Fig. 3.1 can acquire a set of 30 apples in a couple of minutes. Images were acquired in burst mode with a Canon camera (10.1 megapixels resolution) controlled by a simple Raspberry-pi minicomputer. Apples were segmented automatically from the background as visible in Fig. 3.1 and assembled in multiple view images of 30 apples as shown in Figs. figs. 3.2 and 3.3. This machine, developed for this study, is much simpler and low cost (approximately ($\approx 10k€$) versus ($\approx 100k€$) for classical apple sorting machines) than any commercial system since it does not need to incorporate any sorting mechanism. Also, by contrast with most commercial systems , access to raw image format, i.e. uncompressed format, is possible.

Datasets

Currently, when experts of EO are performing distinctness, they observe directly with their own eyes boxes of 30 apples of each tested variety and reference varieties manually positioned in the same room, and they decide from a pure subjective perception if these sets are distinct or not from one another. An objective of this work is to produce a step

Figure 3.1 – Acquisition system. Upper panel: Machine equipped with a conveyor belt, used for the acquisition of images of apples with a high surface coverage. Lower panel: view of the acquired images of apples after segmentation from the background.

toward automation of such examination through the use of computer vision applied to images such as Figs. 3.2 and 3.3, which are automatically produced after acquisition on the system of Fig. 3.1. Two datasets were produced for this study to test the proposed machine vision approach for distinctness evaluation.

Non-Gala Mutant varieties

We first created a dataset of images of apples with highly distinct color distributions. The dataset is composed of 1293 images of apples belonging to 8 varieties (see Fig. 3.2) which we refer to as non-Gala Mutants. These varieties correspond to varieties identified as distinct from each other by the official examining offices. These varieties are not named. They simply have a reference number to identify them. The number of images per variety is given in Tab. 3.1.

Variety	Images
variety30	113
variety37	127
variety40	133
variety41	99
variety42	106
variety44	312
variety46	215
variety47	188

Table 3.1 – Number of images of 8 reference, registered varieties corresponding to Non-Gala Mutant dataset.

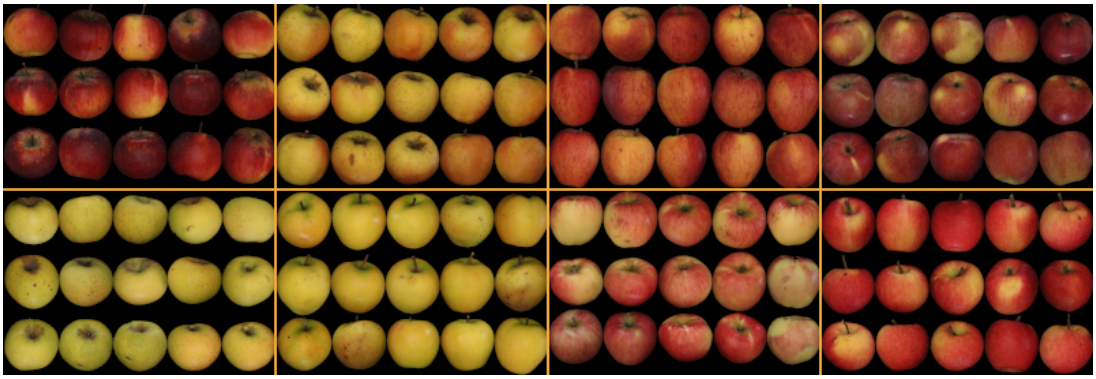


Figure 3.2 – Images representing the 8 non-mutant Gala varieties in the order of appearance in the list of Tab. 3.1.

Gala mutants

As a complement to the first dataset, we built a second dataset containing 4040 images of apples belonging to 9 different mutants of the variety Gala. These mutants are similar to each other in terms of color content, as shown in Fig. 3.3. The details of these mutants are given in Tab. 3.2 where first column gives the encrypted reference of each corresponding mutant. These mutants are also considered distinct from each other by experts of EO, but they somehow reach the limit of what they consider as distinct.

3D RGB Histograms

With our objective being to differentiate apples mainly based on the color distribution, we extracted features from the RGB histogram of the images represented in 3 dimensions (one axis by component color). We calculated the RGB histogram of each image, and

References	**Images**
X4111	597
X4410	438
X4712	716
X6716	684
X7440	444
X7812	259
X8125	329
X8594	343
X9214	230

Table 3.2 – Number of images of 9 reference, registered varieties corresponding to Gala Mutant dataset.

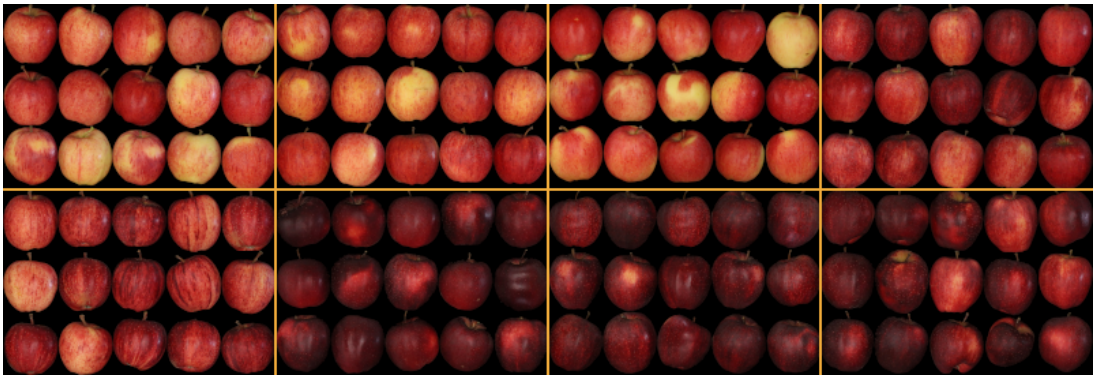


Figure 3.3 – Images showing a subset of each Gala Mutant, in the same order of appearance as in Tab. 3.2, from left to right and by line, except for the mutant *X8594* which is completely yellow and highly distinctive.

to obtain the RGB histogram of a variety, we simply calculated the sum of the RGB histograms of the images belonging to the variety. We can see the corresponding summed histogram of each of the non-Gala mutants in Fig. 3.4 and of the Gala mutants (except X8594) in Fig. 3.5. It is interesting to see that despite the loss of spatial localization in RGB histograms, a contrast between colors is clearly visible in this representation with the non-Gala mutants in Fig. 3.4. However, the contrast is much more difficult to perceive with RGB histograms in the case of the Gala mutants, which represents a clearly challenging classification task.

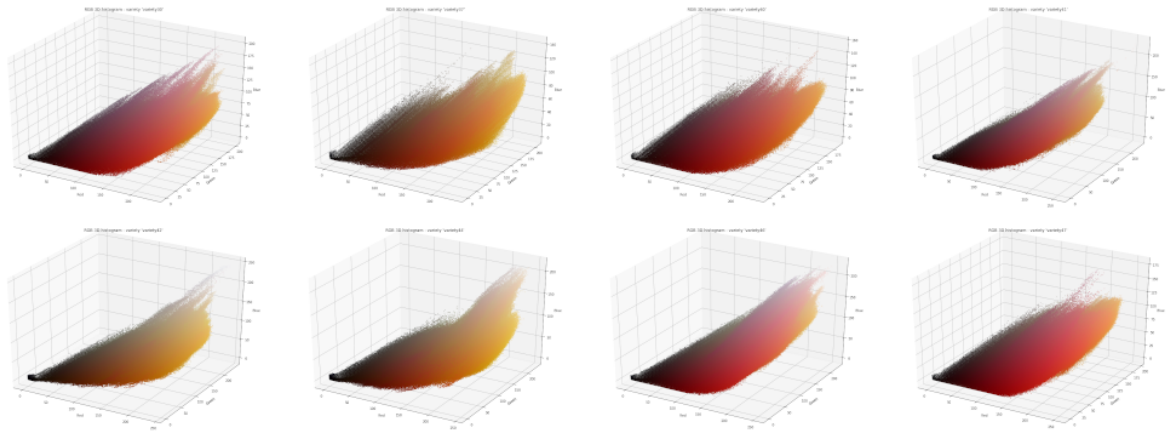


Figure 3.4 – Images representing histograms of non-mutant Gala varieties. From left to right, by row: variety30, variety37, variety40, variety41, variety42, variety44, variety46 and variety47.

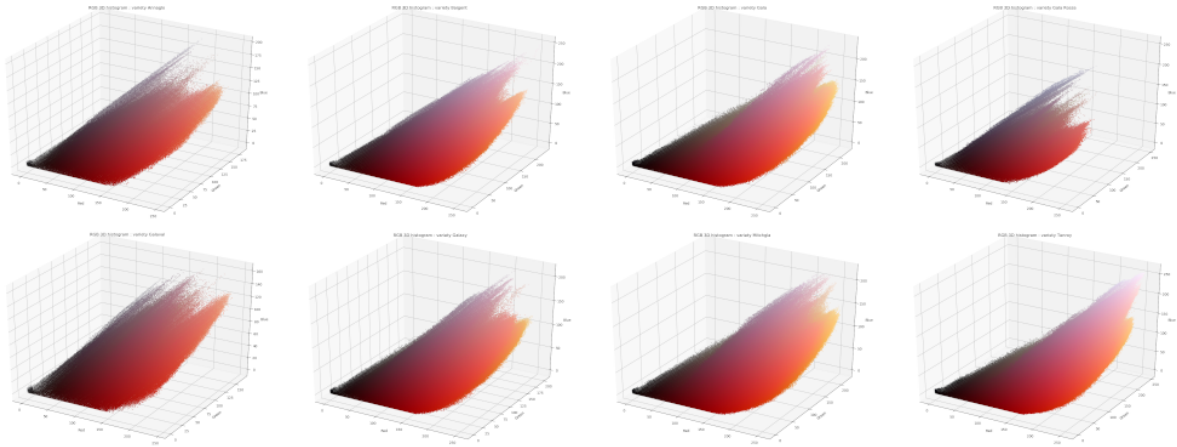


Figure 3.5 – Images representing histograms of mutant Gala varieties. From left to right, by row: X4111, X4410, X4712, X6716, X7440, X7812, X8125, X9214.

3.2.2 Color features

Once the histogram of each image and each variety is built, we extract features, allowing us to characterize several aspects of the histograms. The same features were used for both datasets. We present the features used for this study in the rest of this section.

Average and variance of colors: The first two descriptors are the mean and the variance of the colors. It seems quite intuitive to use them since they give us respectively the average color of the variety of apples and the contrast of yellow and red regions are

captured via the variance. These two values are easily calculated. Let I be an image of size $m \times n$ represented in the RGB space by an array of RGB vectors $P = (p_{i,j})$ where $p_{i,j} = (r_{i,j}, g_{i,j}, b_{i,j}) \in 0, 255^3$ is the color of the pixel of coordinates $(i, j) \in 0, m-1 \times 0, n-1$ of the image I. Denoting the mean and variance of the colors respectively by $\bar{P}$ and V the mean and the variance of the image I, we have:

$$\bar{P} = \frac{1}{m \times n} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} c_{i,j}, \quad V = \frac{1}{m \times n} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (c_{i,j} - \bar{c})^2 \tag{3.1}$$

Fractional Anisotropy: Fractional anisotropy is a number in the interval $[0, 1]$ which reflects the degree of anisotropy of the shape of the point cloud formed by the 3-dimensional histogram. This scalar gives a measure of the stretch of the point cloud in various directions. If its value is 1, it means that the points would all be distributed along a perfectly linear axis. If its value is zero, it means that the points are distributed homogeneously in all directions. Thus, a sphere has a zero fractional anisotropy, an ellipse has a fractional anisotropy between 0 and 1 and a straight line has a fractional anisotropy equal to 1. After obtaining the eigenvalues λ_1, λ_2 and λ_3 of the PCA (Principal Components Analysis) on the 3-dimensional histogram of an image, we can easily calculate the fractional anisotropy, denoted as FA, via the formula

$$FA = \sqrt{\frac{1}{2}} \frac{\sqrt{(\lambda_1 - \lambda_2)^2 + (\lambda_2 - \lambda_3)^2 + (\lambda_3 - \lambda_1)^2}}{\sqrt{\lambda_1^2 + \lambda_2^2 + \lambda_3^2}} \tag{3.2}$$

Fractal box counting dimension: The fractal box counting method [109] subdividing the 3D color cube $0, 255^3$ into '*box*' of the edge length r counts the number of boxes $N(r)$ necessary to cover each color cell occupied by the point cloud making up the 3D histogram. This number of boxes $N(r)$ has been found to follow a law of the form r^{-D} where D is the fractal dimension of the histogram [94, 93]. This number ranging between 2 and 3 for natural images, provides a description of the structure and density of the point cloud. A smaller value of fractal dimension indicates that although the histogram is distributed throughout the color space, there remain empty regions.

Mutual entropy: Mutual entropy [110] allows us to compute the information common to 2 histograms. We include this mutual entropy as the measure of the color similarity between an image and the target variety. To calculate the mutual entropy, also called

joint entropy, between two histograms, we use the following formula

$$ME = -\sum q(x) \log(\frac{q(x)}{p(x)}) \tag{3.3}$$

where p and q represent the respective pixel distributions represented in the histograms of the two images.

Cost of optimal transport: As the last feature, we propose to include optimal transport [111] which provides a way of transporting a set of points to another set in the least expensive way possible. In our case, minimizing the total distance between the two sets of points fits with the capability of optimal transport. Since we work on 3-dimensional histograms of our images, we can measure the cost in terms of the distance between the histogram of an image and the average histogram of a target variety. If we assume we have k varieties (here $k = 8$ or 9 depending on the dataset used), we get k values representing the cost of moving our image to the k varieties. These k values are treated as color features, each representing a measure of the probability of the image to belong to the corresponding variety. The lower the cost, the closer the histograms are in terms of structure. Let $\mu = \sum_{i=0}^{m-1} p_i \delta_{a_i}$ and $\nu = \sum_{j=0}^{n-1} q_j \delta_{b_j}$ be two discrete measures associated with two histograms, and let c be a cost function for which we note $c_{ij} = c(a_i, b_j)$ for all $i, j \in 0, m-1 \times 0, n-1$. We then try to minimize $\iint c(x, y)\, dx\, dy$. By noting $b = \{p_0, p_1, \ldots, p_{m-1}, q_0, q_1, \ldots, q_{n-1}\}^T$ and $c = \{c_{11}, c_{12}, \ldots, c_{1(n-1)}, \ldots, c_{m1}, c_{m2}, \ldots, c_{(m-1)(n-1)}\}^T$, the problem then becomes a minimization problem of $c^T x$ under the constraints

$$\begin{cases} A.x = b \\ \forall i, j, x_{ij} \geq 0\ . \end{cases} \tag{3.4}$$

In practice, we calculated this cost using the python package named POT (Python Optimal Transport, https://pythonot.github.io/). The cost is computed using the method based on earth mover's distance, from 3D histograms. This algorithm has 2 advantages : histograms do not need to be normalized and they do not need to be of the same size [112]. All in all, the feature space is composed of features of various dimensions. The optimal transport feature is a vector for which each component is the value of the norm of the cost between two varieties. Therefore, if the dataset is composed of k varieties, the optimal transport is a vector with k components. The other features can be scalars as fractal dimension or fractional anisotropy, 3 component vectors as RGB means and RGB

variance, or a vector of the same dimension as the optimal transport for mutual entropy.

3.2.3 Classification setups

In this part, we detail the machine learning classification setups tested to assess distinctness with both apple datasets presented in the previous section.

Multi-class classification A first setup is simply to perform a multi-class classification between the varieties, allowing to assess if the varieties are distinguishable between them. This is a "one versus one" approach, where the tested variety is tested against all the existing ones individually. For this, we simply separated an initial dataset of images to create 2 sub-datasets: a test sub-dataset containing $\frac{1}{3}$ of the images, and a training sub-dataset containing the rest. These 2 subsets were used respectively to train the supervised classifiers and to test their efficiency to distinguish the different varieties. For this classification, two sets of features were used, a set containing all features and the other set containing only the most relevant features among all the tested features.

Binary classification The second classification setup was used to test if our model was able to differentiate the two apple datasets. This is a "one versus all" approach, where the one tested corresponds to the variety compared with all the existing registered varieties at once. We gathered the 2 datasets presented previously, thus constituting a dataset of 5333 images, with 4040 images of Gala mutants and 1293 non-Gala mutants which are our 2 classes. We separated the dataset into test and training sub-sets with a 50-50 ratio. To mimic the procedure, experts currently follow for apple variety testing, the algorithm made an individual prediction on each apple and a majority voting over subsets of 30 apples.

3.2.4 Classification results

Multi-class classification between Gala mutant varieties

We first performed the multi-class classification between Gala mutant apples only, in order to verify that it was indeed possible to distinguish these 9 registered varieties between them. We first separated the data into test and training sets with a ratio of $\frac{2}{3}$ for the training set, then we used 3 different supervised classifiers: Support vector machine (SVM), Random Forest and Linear Discriminant Analysis (LDA).

The classification results visible in Tabs. 3.3 and 3.4 show that the Gala mutant varieties are distinguishable in terms of color. These results are of the same order of magnitude for the three tested classifiers. However, SVM gives an F1-score over 97%, and perform slightly better than others.

A forward analysis, testing the performance of each type of feature, identified that the best features for the classification happened to those from optimal transport. As visible in Tabs. 3.5 and 3.6, these features alone do not allow us to obtain an entirely satisfactory classification. However, they are relatively efficient since they yield a classification accuracy of about 50%. Since our dataset has 9 distinct classes, a random classification of the data would give 11% accuracy. The relative superiority of optimal transport toward the other features can be explained since all histograms share the same elongated shape centered on a red-yellow barycenter.

SVM : All Features	
Precision Score	97,13 %
Recall Score	97,10 %
F1-Score	97,07 %
Parameters	
Kernel	linear
Penalty	l2
C	2.0
Loss	squared_hinge
Dual	False
Tol	0.0001
Multi_class	crammer_singer
Class_weight	None
Max_iter	-1
Data Transformation	Normalized

Table 3.3 – Results obtained by SVM with Gala mutant dataset and all features.

3.2.5 Multi-class classification between non-gala mutant varieties

In a second step, we performed the same classification method as in the previous section, this time using the dataset composed of the non-Gala mutant varieties.

LDA : All Features	
Precision Score	93,39 %
Recall Score	93,32 %
F1-Score	93,30 %
Parameters	
Solver	SVD
Shrinkage	None
Priors	None
n_components	1
store_covariance	False
tol	0.0001
Data Transformation	Normalized

Random Forest : All Features	
Precision Score	94,13 %
Recall Score	93,99 %
F1-Score	93,92 %
Parameters	
n_estimators	250
criterion	entropy
max_depth	None
max_leaf_nodes	None
bootstrap	False
Data Transformation	Normalized

Table 3.4 – Results obtained by LDA and Random Forest with Gala mutant dataset and all features.

SVM : OT Features	
Precision Score	50,48 %
Recall Score	50,26 %
F1-Score	46,34 %
Parameters	
Kernel	linear
Penalty	l2
C	8.9
Loss	squared_hinge
Dual	True
Tol	0.0001
Multi_class	None
Class_weight	balanced
Max_iter	1,00E+06
Data Transformation	Normalized

Table 3.5 – Results obtained by SVM with Gala mutant dataset and optimal transport only.

For this dataset, the results obtained in Tabs. 3.7 and 3.8 are also very satisfactory, with F1-scores close to 90%. Once again, the SVM with a polynomial kernel gives the best results with a precision score of 93.76%. For the Non-Gala mutants, these results show that the varieties composing this dataset are clearly distinguishable, as also confirmed by the experts from EO since these are registered as official varieties. The precision score is

LDA : OT Features	
Precision Score	48,39 %
Recall Score	50,56 %
F1-Score	46,56 %
Parameters	
Solver	lsqr
Shrinkage	auto
Priors	None
n_components	1
store_covariance	False
tol	0.0001
Data Transformation	Normalized

Random Forest : OT Features	
Precision Score	52,35 %
Recall Score	53,38 %
F1-Score	52,49 %
Parameters	
n_estimators	40
criterion	entropy
max_depth	None
max_leaf_nodes	None
bootstrap	True
Data Transformation	Normalized

Table 3.6 – Results obtained by LDA and Random Forest with Gala mutant dataset and optimal transport only.

logically found a bit lower than for the non-Gala mutant datasets, since the contrast in color between varieties is lower.

Again we performed a forward analysis which established optimal transport as providing the most significant features. As visible in Tabs. 3.9 and 3.10, the classification only based on these optimal transport features reached their best results with SVM with a polynomial kernel of degree 3, which gives a precision score of 71.25%. Globally, the result observed with the well-contrasted dataset non-Gala mutant are robustly conserved when the method is transposed to less contrasted apples, such as the ones of the Gala mutants. This demonstrates the high potential of a machine learning framework equipped with color features for variety testing.

Binary classification with the 2 collected datasets

Once we observed that both datasets were well distinguishable, we focus on the most difficult dataset and explore the potential of our framework to determine whether a set of test images corresponds to a certain Gala mutant or not. To mimic the way experts perform their scoring, we decided to focus not only on individual classification of apples, but also on a group classification from the same variety. To do so, we selected images from the test data and by a random draw without replacement of apples of the same variety, to create subsets of 30 apples. This number exactly corresponds to the size of the group of apples chosen by the experts when they observe groups of apple for distinctness. Once our model is trained on classification of individual apple images, we tested its efficiency

SVM : All Features	
Precision Score	93,76 %
Recall Score	93,50 %
F1-Score	93,51 %
Parameters	
Kernel	poly
Degree	2
Penalty	l2
C	12
Gamma	auto
Coef0	0.5
shrinking	False
tol	0.0001
Class_weight	None
Max_iter	-1
Data Transformation	Normalized

Table 3.7 – Results obtained by SVM with non Gala mutant dataset and all features.

LDA : All Features	
Precision Score	89.50%
Recall Score	89.10%
F1-Score	88.96%
Parameters	
Solver	lsqr
Shrinkage	auto
Priors	None
n_components	1
store_covariance	False
tol	0.0001
Data Transformation	Normalized

Random Forest : All Features	
Precision Score	89,35 %
Recall Score	89,10 %
F1-Score	88,81 %
Parameters	
n_estimators	90
criterion	entropy
max_depth	None
max_leaf_nodes	None
bootstrap	True
Data Transformation	Normalized

Table 3.8 – Results obtained by LDA and Random Forest with non Gala mutant dataset and all features.

on the subsets through majority voting.

As can be seen in Tabs. 3.11 to 3.13, we get 100% F1-score with all classifiers when all features are employed. Consistent with the results of the previous section, optimal transport again appeared as the most important features in a forward analysis. With optimal transport only, Random Forest gives the best results in individual classification, with a precision of 88.13% and an F-score of 75.67%.

SVM : OT Features	
Precision Score	71,25 %
Recall Score	70,53 %
F1-Score	67,22 %
Parameters	
Kernel	poly
Degree	3
Penalty	l2
C	20
Gamma	auto
Coef0	0.5
shrinking	False
tol	0.0001
Class_weight	None
Max_iter	-1
Data Transformation	Normalized

Table 3.9 – Results obtained by SVM with non Gala mutant dataset and optimal transport only.

LDA : OT Features	
Precision Score	52,96 %
Recall Score	61,72 %
F1-Score	55,20 %
Parameters	
Solver	svd
Shrinkage	None
Priors	None
n_components	1
store_covariance	False
tol	0.0001
Data Transformation	Normalized

Random Forest : OT Features	
Precision Score	57,78 %
Recall Score	59,40 %
F1-Score	58,12 %
Parameters	
n_estimators	40
criterion	gini
max_depth	None
max_leaf_nodes	None
bootstrap	True
Data Transformation	Normalized

Table 3.10 – Results obtained by LDA and Random Forest with non Gala mutant dataset and optimal transport only.

3.2.6 Conclusion

In this section, we have demonstrated, on a use-case dedicated to apples, a supervised machine learning scheme to identify if a new candidate for variety registration could be considered as distinct or not from an existing set of varieties. Two datasets corresponding to highly contrasted varieties and varieties contrasted at the limit of what would be

SVM – Individual classification – OT only		**True labels**		**Scores**	
		0	1	Precision	51,16 %
Predictions	0	1014	117	Recall	73,88 %
	1	316	331	F1-Score	60,46 %
SVM – Subsets classification – OT only		**True labels**		**Scores**	
		0	1	Precision	100,00 %
Predictions	0	14	0	Recall	100,00 %
	1	0	14	F1-Score	100,00 %
SVM – Individual classification – All features		**True labels**		**Scores**	
		0	1	Precision	100,00 %
Predictions	0	1330	0	Recall	100,00 %
	1	0	448	F1-Score	100,00 %
SVM – Subsets classification – All features		**True labels**		**Scores**	
		0	1	Precision	100,00 %
Predictions	0	14	0	Recall	100,00 %
	1	0	14	F1-Score	100,00 %

Table 3.11 – Results obtained by SVM on individual data and subsets of apples, with optimal transport only and all features.

LDA – Individual classification – OT only		**True labels**		**Scores**	
		0	1	Precision	82,26 %
Predictions	0	1286	244	Recall	45,54 %
	1	44	204	F1-Score	58,62 %
LDA – Subsets classification – OT only		**True labels**		**Scores**	
		0	1	Precision	100,00 %
Predictions	0	14	10	Recall	28,57 %
	1	0	4	F1-Score	44,44 %
LDA – Individual classification – All features		**True labels**		**Scores**	
		0	1	Precision	100,00 %
Predictions	0	1330	0	Recall	100,00 %
	1	0	448	F1-Score	100,00 %
LDA – Subsets classification – All features		**True labels**		**Scores**	
		0	1	Precision	100,00 %
Predictions	0	14	0	Recall	100,00 %
	1	0	14	F1-Score	100,00 %

Table 3.12 – Results obtained by LDA on individual data and subsets of apples, with optimal transport only and all features.

considered distinct have been tested. Distinctness was found in perfect accordance with the human expert. This demonstrates the possibility to introduce more objective and higher-throughput approaches in the domain of variety testing. We found that among the tested features, optimal transport was producing the most adapted features, i.e. which contributed the most in the correct decision-making. It is essential to notice that all these results were obtained based on a color histogram, i.e. with a total loss of spatial information.

This first step opens various ways of further investigations. A limit of the result pre-

RF – Individual classification – OT only		True labels		Scores	
		0	1	Precision	88,13 %
Predictions	0	1291	151	Recall	66,29 %
	1	40	297	F1-Score	75,67 %
RF – Subsets classification – OT only		True labels		Scores	
		0	1	Precision	100,00 %
Predictions	0	14	0	Recall	100,00 %
	1	0	14	F1-Score	100,00 %
RF – Individual classification – All features		True labels		Scores	
		0	1	Precision	100,00 %
Predictions	0	1330	0	Recall	100,00 %
	1	0	448	F1-Score	100,00 %
RF – Subsets classification – All features		True labels		Scores	
		0	1	Precision	100,00 %
Predictions	0	14	0	Recall	100,00 %
	1	0	14	F1-Score	100,00 %

Table 3.13 – Results obtained by Random Forest on individual data and subsets of apples, with optimal transport only and all features.

sented so far stands in the absence of negative data, i.e. unregistered varieties in our dataset. Access and diffusion of such historical data is complex from a legal point of view when dealing with EO. A workaround approach could consist in simulating fake unregistered varieties from an existing dataset. This requires enlarging the datasets used in this article and we are currently investigating this direction. On the machine learning side, several alternatives could be considered. We selected classical shallow learning algorithms (SVM, random forest and LDA). We produced binary decisions in accordance with the essence of distinctness which is a binary trait. All the tested algorithms could also provide probabilities and confidence intervals which would provide more insights. Such output, although not currently in practice in variety testing, would nonetheless be very useful specially to provide arguments to the breeders when new variety candidates are rejected by EO. The set of hand-crafted features could be extended to additional color features mentioned in the related work section. Also, all the analyses were performed in the native RGB color space and other color spaces more suitable to fit with the human perception could also be tested with the approach introduced in this paper. Alternatively, deep learning approaches could be considered. An obvious match would be with generative adversarial networks (GAN) [113] where the discriminator network could decide if a variety is distinct from another after the GAN would have been trained to reproduce images of already registered varieties.

In the next section, we propose an alternative approach, different from the one presented in this section. This approach is based on non-supervised machine learning, to

incorporate machine learning in the distinctness tests of the DUS variety testing protocols. The approach is evaluated for the shape characterization test.

3.3 Apple shape charaterisation

Coupling computer vision with supervised machine learning techniques has been proven to be successful for many phenotyping tasks. However, supervised machine learning being dependent on data for building models for inference, is not suitable to mimic the current protocol for variety testing, where experts establish their ratings in reference to the visual comparison between sketches and the inspected real plants. Also, since the expert can disagree on their ratings due to subjective interpretations of the official sketches, supervised machine learning using ground truth provided by an expert may embed some bias to their inference models.

In this work, we investigate the approach of directly using reference sketches in an official catalog for quantitative matching with images of plants to be assessed (see Fig. 3.9). We test this approach, which is novel in the context of variety testing, on the problem of apple shape assessment. The closest related problem in computer vision is sketch-based image retrieval (SBIR), where the objective is the retrieval of related images from a database given a sketch query (see [114] for a recent review). The SBIR method combines information from both datasets (sketches and RGB images) for high image retrieval accuracy. Rather than image retrieval, we target the classification of RGB images of apples by quantitatively matching them with catalog sketches.

Shape-based classification of fruits based on supervised machine learning has been widely studied in the literature (see [115, 116, 117, 118] for recent reviews). While we use classical shape features to characterize the shape of apples, we do not promote supervised machine learning techniques.

3.3.1 Materials and methods

Reference Sketches

The apple shape classification tool currently follows the UPOV rules. In this variety testing framework, experts inspect cut apple shapes by comparing them to the reference sketches in the official variety testing catalog (see Fig. 3.6). Three main classes are used to designate the shape of apples: *Flat*, *Globose* and *Oval*. For each category, there are sub-

categories such *Flat-Globose*, *Oblong* and *Ellipsoid*. In this study, we target classification of apple images to one of the three broad categories: *Flat*, *Globose* and *Oval*.

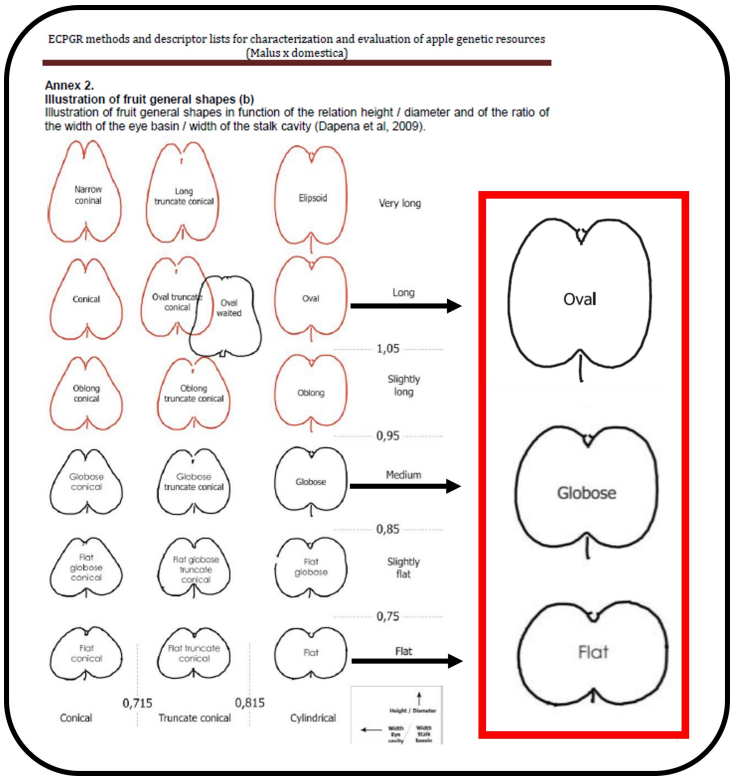


Figure 3.6 – Reproduced and modified view of the ECPGR catalog. Apple shape sketches in the catalog of variety testing. The classes considered in this work are highlighted in the red rectangle.

Data set

The apple images assessed in this work are extracted from the dataset described in [119]. The image acquisition procedure is shown in Fig. 3.7. As an image acquisition procedure, apples from the so-called Refpop population [11]) are cut along their medial axis, placed on their flat, freshly cut side in groups of 6 on an HP Scanjet Pro 4500 fn1 with a resolution of 1200 x 1200 dpi. Since the contrast between the apples and the background is strong, we used simple thresholding on the brightness channel of HSB color space to segment the apples from the background. The bounding box of each individual apple is obtained through connected component analysis. A simple edge detection via Sobel filter is applied to produce a binary image highlighting the boundaries of the apples (see Fig. 3.9).

Figure 3.7 – Acquisition device: HP Scanjet Pro 4500 fn1.

From the original dataset of [119], 1821 images were selected and classified independently by two experts. Figure 3.8 shows the distribution of the classes (*Flat*, *Globose* and *Oval*) corresponding to the annotations of the two experts. Three sets of class labels resulted from this annotation. The class labels provided by expert 1, by expert 2, and a subset labeled in agreement by both experts (600 images) were kept. This quantifies the inter-variability between experts and the current consequences of subjective rating. This also shows the intrinsic difficulty of the visual tasks raised to experts in variety testing where experts only agree in 30% of the cases.

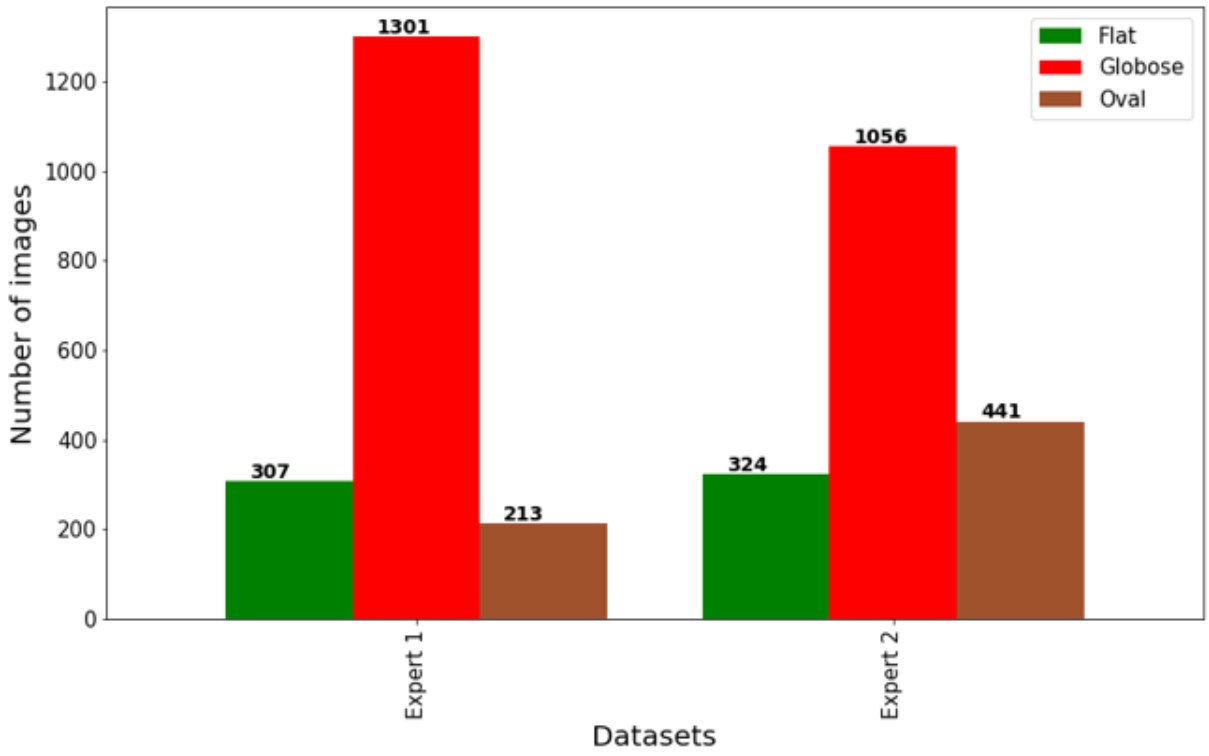


Figure 3.8 – Histogram of the distribution of classes (*Flat*, *Globose* and *Oval*) assigned by experts.

Shape descriptors

In this article, since our aim is not to provide new features to characterize apple shape but rather to investigate the possibility to use the reference sketches for apple shape classification, we used some existing shape descriptors [115]. We used chain-code histogram, elliptical Fourier descriptors [120] and Frechet's Ratio with the following hyperparameters. A connectivity of 8, was chosen for the chain-code. The ten first harmonics of elliptical Fourier descriptors were kept. All features were normalized to 1 to allow the use of Euclidean distance in the produced feature space.

Experiments

We evaluated two approaches for apple classification: i) reference-based classification approach and ii-) model-based classification approach, where a support vector machine (SVM) is used as the machine learning method. The details are provided in the following subsections.

Reference-based classification approach

In this experiment, we mimic the way variety testing experts use catalogs. We perform a multi-class classification by computing the Euclidean distance between features of cut apple query image and features from a reference instance of each class: one for *Flat*, one for *Globose* and one for *Oval*. A visual abstract of the pipeline is given in Fig. (3.9). Three types of references were tested i) the original sketches from the official catalog of ECPGR, ii) the contours from real apples each representing a class, iii) The ECPGR sketches rescaled.

First, we considered the original sketches from the official catalog of ECPGR. As the second option, instead of the sketches provided in a catalog, we considered the contours of representative apples as reference shapes. The representative apples are chosen from the dataset of real apple images. For each class, the apple with the aspect ratio closest to the class average is selected as the class representative. As the third option, we modified ECPGR sketches such that the aspect ratio of each reference sketch becomes equal to the corresponding class average. This operation bridges the gap between the aspect ratios of the sketches and the distribution of the aspect ratio of the apple variety to be tested.

The average aspect ratio was computed in the following unsupervised way. We assigned the centroids of each cluster as the class average aspect ratio to the corresponding class,

knowing the ordinal relation between classes. This sorting was obviously not perfect (otherwise, the task would have been done).

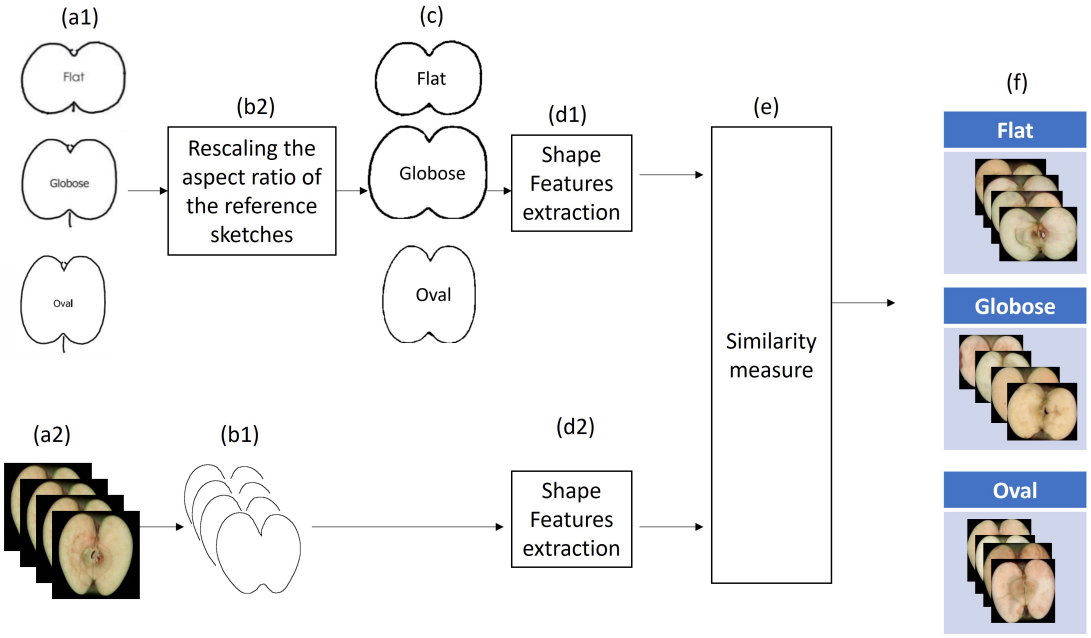


Figure 3.9 – Reference-based classification approach: (a1): reference sketches Dataset, (a2) RGB images Dataset to be classified. (b1): edge detection. (b2): rescaling of the aspect ratio of sketches of each class. (c) rescaled sketches. (d1) shape features extraction. (e): similarity measure. (f): classification results.

Model-based classification approach

In this approach, we do not follow the ECPGR catalog and rather adopt a supervised machine learning technique. A model is trained to classify the shape of images based on a training set composed of annotated RGB images. Since our goal is not to claim optimal performances but rather to provide a comparison with our proposed reference-based approach, we selected a basic machine learning model, an SVM, with a linear kernel. To quantify the sensitivity of the chosen data in the training, multiple runs of the classification experiment were conduced for various values of the train-test split of the data set and a 10-fold cross-validation. The average value and standard deviation of the performances of classification were recorded. To quantify the inter-variability between experts annotations, the experiment was repeated with labels provided by the two experts for both annotators and with the curated data set containing apples with agreed labels only.

Metric

All the classification experiments were evaluated using the accuracy metric:

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \ . \tag{3.5}$$

To rely on this metric, the classes *Flat*, *Globose* and *Oval*, on both experts annotated datasets and cured dataset were balanced with 200 images for each class.

3.3.2 Results

3.3.3 Reference-based classification approach

Table 3.14 provides quantitative evaluation of reference-based approach results. One can observe low accuracy when the sketches ECPGR catalog as references are used. A noticeable improvement of 9 to 13 % of accuracy is brought when these reference sketches are rescaled. The best performance is obtained when the class representatives are selected as a reference. However, it is to be noticed that the gain of performance is only of a 2 to 4 % by comparison with the unsupervised rescaling of the official sketches. The performance are culminating at 77% of accuracy when the data are cured. We reproduced the experiment after withdrawing the intermediate class *Globose*. Results given in Tab. 3.15 show similar trends as in Tab. 3.14 but with much higher accuracy around 95%. We still observe a gain of 2 to 3% after rescaling the reference sketches of the ECPGR catalog with our proposed supervised approach.

Table 3.14 – Measuring accuracy (ACC) of reference-based approach on expert 1 (E1), expert 2 (R2) and cured data (CD), to all types of sketches.

Sketches	% ACC_{E1}	% ACC_{E2}	% ACC_{CD}
Reference sketches	58%	55%	60%
Classes representatives	69%	67%	77%
Rescaled reference sketches	67%	63%	73%

Comparison with model-based approach

Figure 3.10 shows the performance of the model-based approach as a function of the ratio of the size of the test set to the size of the training set. As trivially expected, the performances drop progressively with increasing standard deviation when the amount of

Table 3.15 – Same as in Table 1 but with only two classes *Flat* and *Oval*.

Sketches	% ACC_{E1}	% ACC_{E2}	% ACC_{CD}
Reference sketches	94%	90%	95%
Classes representatives	97%	96%	98%
Rescaled reference sketches	97%	95%	97%

data in the training data set reduces. The plateau of performance for low test/training ratio is around 93% for the cured data sets but drops around 43% with a huge standard deviation of 14% when only one instance is kept. This is to be compared with the reference-based classification approach explained in the previous subsection, where only one image per class was used.

The results of predictions on the test set of cured data, using the model-based approach and the reference-based approach toward reference sketches, centers representatives and rescaled reference sketches, are presented in Fig. 3.10. It is important here to recall that the reference-based approach is purely unsupervised and, therefore, fully automatic in the case of testing of a new variety while the model-based approach requires labor-intensive annotation of the newly introduced data set.

The performance of the reference-based classification is found to be stable with the amount of data used to compute the rescaling aspect ratio and outperforms the model-based approach when less than 30% of the data sets are not annotated. This experiment was carried out again while withdrawing the intermediate *Globose* class with similar results shown in Fig. 3.11. The difference of plateau of performance between the model-based approach and the reference-based approach vanishes, and there is here no clear advantage in annotating the images to train a model. The current approach based on sketches can directly be automated with the unsupervised approach proposed in this work.

3.3.4 Discussion

As illustrated in Fig. 3.12, variety testing catalogs may differ from one country to another, as well as between germplasm curators and breeders, or the sketches of reference may evolve. This situation may cause difficulty of comparison of the results over the time or even non-communicability between countries not sharing the same references. The instance-based approach described in this article may actually serve to decipher this babel tour-like problem. As shown in Tab. 3.16, we investigated the possibility of automatic translation of one catalog to another. For this purpose, we provided the nearest reference

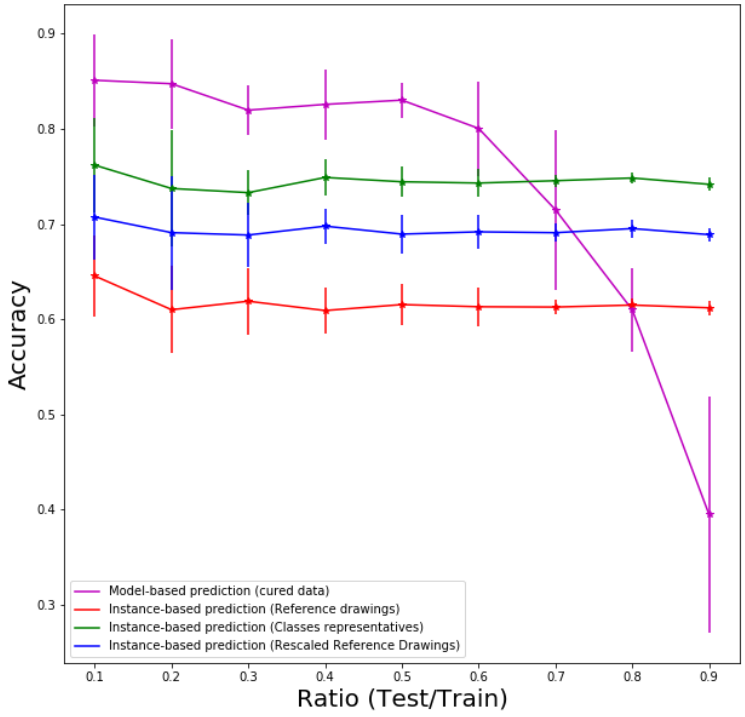


Figure 3.10 – Prediction curves of test set of cured data via model-based classification and reference-based classification using reference sketches, centers representatives and rescaled reference sketches, after training on cured dataset.

sketches to a query reference sketch from a different catalog.

In Tab. 3.16, some catalogs' characteristics are found to be directly almost perfectly matching with each other. In other cases, the designation used in one catalog does not match with the designation of another catalog. This translation based on pure objective features enables overcoming the semantic gap that the multiplicity of catalogs may cause. Consequently, our approach may not only be used to provide objective measurements while following the current variety testing practices, but it can also be used when measurements based on different visual references need to be shared. It is here again important to stress that this translation from one catalog of reference to another, would not be directly accessible with supervised machine learning. Indeed this would require training models with an expert looking at sketches from different catalogs. Then it would require to compare this subjective rating result on testing data. By contrast with the unsupervised

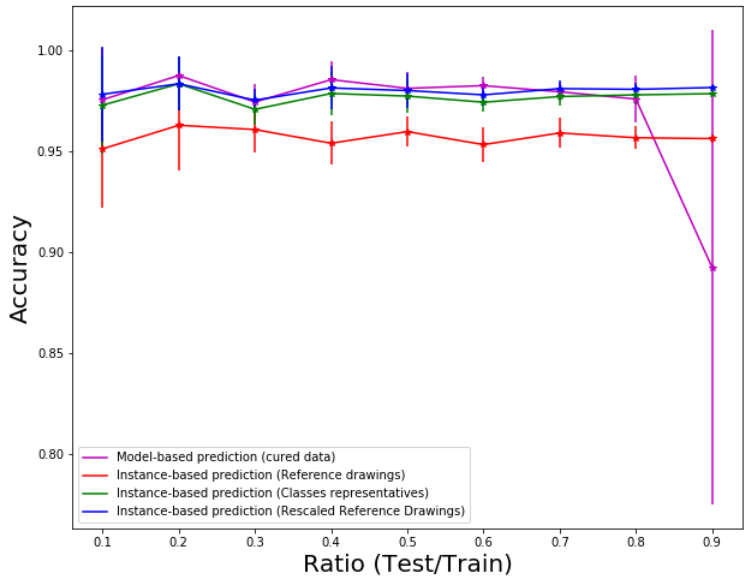


Figure 3.11 – Same as in Fig. 3.10 with only two classes *Flat* and *Oval*.

instance-based model proposed in this article, translation from one catalog to another is almost instantaneous since only the objective similarity between reference sketches needs to be computed.

Table 3.16 – Translation between variety testing catalogs.

ECPGR Catalog	**Upov Catalog**					
Flat	*Flat*	*Oblate*	*Globose-Canonical*	*Ellipsoid*	*Oblong*	*Globose*
Globose	*Globose*	*Globose-Canonical*	*Ellipsoid*	*Oblong*	*Oblate*	*Flat*
Oval	*Globose*	*Ellipsoid*	*Oblong*	*Globose-Canonical*	*Oblate*	*Flat*
Ellipsoid	*Oblong*	*Ellipsoid*	*Globose-Canonical*	*Globose*	*Oblate*	*Flat*
Flat-Globose	*Oblate*	*Flat*	*Ellipsoid*	*Globose-Canonical*	*Oblong*	*Globose*
Oblong	*Globose*	*Oblong*	*Globose-Canonical*	*Oblate*	*Ellipsoid*	*Flat*

3.3.5 Conclusion

In this work, we have demonstrated the possibility of using reference sketches in a variety testing catalog, to help the transition from pure manual inspection toward automated computational practices. The sketches can serve as references for quantitative matching to classify images of plant instances. Rescaling of the aspect ratio of the reference sketches was shown to be helpful in boosting the performance of classification.

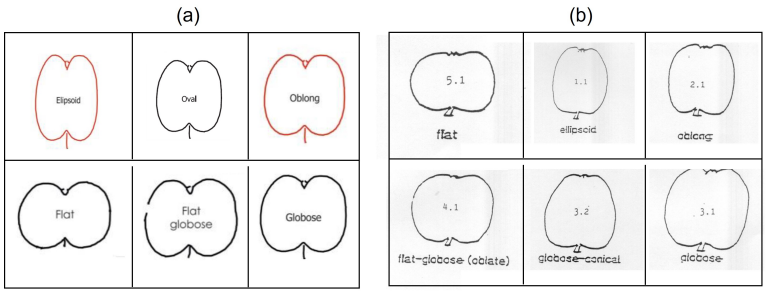


Figure 3.12 – Two different variety testing catalogs. (a): Catalog delivered by The European Malus GERMPLASM Workshop (ECPGR 2009). (b): Catalog delivered by UPOV (2006).

Reference-based approach was shown to be better suited in variety testing as compared to a model-based approach using standard supervised machine learning methods, since the latter requires intensive manual annotation and therefore brings no gain of efficiency to the current practice of manual inspection.

3.4 Conclusion and perspective

In this chapter, we have demonstrated the possibility to incorporate machine learning in the distinctness test of the DUS protocol, via the supervised approach in the automation of the measurement of color and the unsupervised approach in the automation of the measurement of shape.

While supervised machine learning has shown strong potential, the annotation phase can be heavy for some specific applications. The proposed method in the second section can be a good alternative, especially in variety testing. The examiners during the distinctness test follow the instructions in the catalog of UPOV. These guidelines can be addressed as a numerical ground truth. We demonstrated the success of this approach using sketches.

Apple varieties are registered based on a large set of traits. In this chapter, we considered only color and shape. However, it would be interesting to extend the scheme to incorporate more parameters such as texture (stripes on apple skin), size, etc. As a perspective, with more extended traits, rather than evaluating the distinctness based on each trait, the apple varieties could be represented in a high dimensional point cloud with axes referring to descriptors of the traits. In this scenario, the proposed approach in the first section, with optimal transport, can be adapted to such higher dimensional space

and thus offers a generic framework to extend the quest for automation in variety testing and help numerically and visually the examen offices to decide on the existence of the tested variety in the registration catalog.

Chapter 4

Processing ordinal data in variety testing

4.1 Introduction

In this chapter, we continue the investigation of the potential use of machine learning for variety testing. In a numerical test, the distinctness test is a classification problem where varieties are represented through their traits in the feature space. Some traits are scored on ordinal scales. In such a case, the structure of the feature space will also follow an order. Hence, ordinality can serve as a criterion to help the examiners verify if the selected machine learning strategy respects the meaningful order of the data.

To assess ordinal data in the feature space, specific tools must be developed. Ordinal classification or ordinal regression tools are already found in the literature [121]. However, we realized that dimension reduction techniques for ordinality and quantification of ordinality were still missing. We introduce the method we have developed for these tasks in this chapter.

A typical structure of a machine learning pipeline includes feature extraction from raw data and dimension reduction to produce a latent space in which decisions are made by the machine. These steps can be trained sequentially or in an end-to-end framework in deep learning approaches [122]. In addition to efficiency, a critical quality expected from machine learning algorithms lies in the interpretability of their decisions [123, 124]. Understanding the behavior of a machine learning model is especially mandatory for applications with an impact on daily human lives, such as medical applications [125, 126, 127] but also the selection of plant varieties.

Interpretability methods have been classified [128] in three main approaches according to whether they are applied before (pre-model), during (in-model), or after (post-model) building the machine learning model. To investigate the state of ordinality, we target the "Pre-model" interpretability, which refers to the understanding of the organization of the

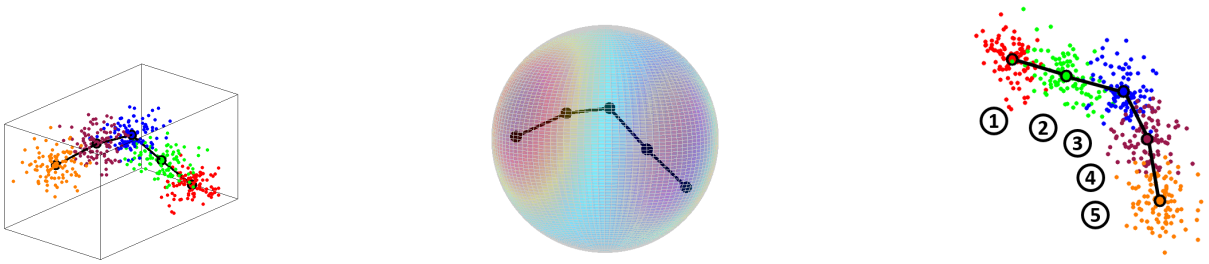
latent space. This space will typically be visualized in 2D or 3D to observe how instances, to be learned, are organized. In such a reduced space, the only structure of dimensions less than 2 or 3 can actually be fully interpreted. This can appear as a strong limitation, but it actually includes all the problems where the latent space is expected to be structured in 1D. This special case includes the niche of ordinal classification problems.

Ordinal classification refers to the classification problems where there is a natural order between categories [129]. Examples to ordinal classification are ranking the severity of a disease [130], prediction of movie preferences [131] or classification of images involving ordinal quantities [132]. The categories are usually represented with one-dimensional discrete values following their inherent order. It is expected that the features used to predict the ordinal categories of the instances also possess an intrinsic order in the high-dimensional space. In order to visualize and assess whether these features follow the ordinality of the categories, dimensionality reduction can be used.

Many dimensionality reduction techniques are available to transform the high-dimensional data into a low-dimensional space for interpretation of the prediction model. Principal component analysis (PCA) is the leading choice. Other unsupervised techniques include multidimensional scaling (MDS) [133], Isomap [134], non-negative matrix factorization (NMF) [135] and t-distributed stochastic neighbor embedding (t-SNE) [136]. When class information of the training data is available, supervised techniques such as Linear Discriminant Analysis (LDA) [137] or Locality Sensitive Discriminant Analysis (LSDA) [138] are more effective in retaining meaningful properties of the data that predict the categories. Although these techniques can be very useful, they do not incorporate the ordinal structure of the categories into their original formulation for ordinal classification problems.

There are very few works on dimensionality reduction specific to ordinal classification. A variant of Kernel Discriminant Analysis has been proposed [139] to find the projection that simultaneously results in a high separation between classes and maintains the class order. In [140], a supervised method based on sufficient dimension reductions (SDR) is developed to regress the response of underlying Gaussian latent variables to ordered categorical variables. However, this method is more suitable for dimension reduction in regression problems for predictor selection and better prediction rather than visualization of the available features.

In this chapter, we propose an intuitive dimensionality reduction technique, which we call *Best-view Projection* (BVP), for ordinal classification without imposing a regression



(a) 3D input data (b) View sphere (c) Projected data

Figure 4.1 – Ordinal classification problem. The original data is shown in (a). The class centers are enclosed with black circles. The segments joining the class centers are in black color. The objective is to find the optimum viewpoint on the view sphere (b) such that the adjacent class centers are seen as apart as possible. The colors on the sphere are indicative of the objective function defined in 4.2. The 2D visualization (c) of the data is obtained through projecting the data to the plane defined by the optimum viewpoint.

model. The main motivation is to propose a complementary tool to the classical dimensionality reduction techniques, which fail for special configurations of data distribution in ordinal classification tasks. We formulate our approach as finding the best viewpoint in the feature space such that the viewer can "see" the direction of ordinality as clearly as possible. Inspired by the work for human skeleton visualization in [141], we determine the optimal viewpoint via maximization of the squared distances between centers of adjacent classes in the projected space. The dimensionality is reduced by one (i.e. from N to $N-1$) by projecting the features to the lower dimensional space defined by the optimum viewpoint. The process is repeated until the desired dimensionality is achieved. A major advantage of our method is that it does not require any parameters to be tuned. We provide a qualitative and quantitative comparison of our BVP method with a number of classical dimensionality reduction techniques on simulated and real ordinal datasets, through two introduced ordinal metrics.

In the last section of this chapter, we present briefly, *Ordinalysis*, a standalone application incorporating an end-to-end machine learning pipeline, dedicated for the automation of the variety testing protocols following an ordinal scale. More details about *Ordinalysis* can be found in Annex H. The material presented in this chapter has been valorized in [142] and in [143].

4.2 Method

Let us consider an ordinal classification problem illustrated in Fig. 4.1, where the features are in 3D space and the categories are ordered. We would like to find a viewpoint on the view sphere such that when viewed from that point, the adjacent class centers seem as apart as possible from each other. Our BVP method finds the optimum viewpoint that maximizes the projected square distances between adjacent class centers and projects the data points to the space defined by the optimum viewpoint.

To generalize the problem for N-dimensional space, let us first suppose that we have K classes, ordered and identified as $k = 1, 2, ..., K$. A class l is adjacent to class k if $l = k - 1$ or $l = k + 1$. The instances of class k are represented as N-dimensional column vectors denoted as $x_i^k \in \mathbb{R}^N$, with $i = 1, 2, ..., I_k$, where I_k is the number of instances in class k. The class centers are denoted as c_k corresponding to the arithmetic mean of the instances in class k. For the sake of simplifying the equation of the view sphere, the data is translated beforehand such that the origin of the N-dimensional space corresponds to $\frac{1}{K}\sum_{k=1}^{K} c_k$, i.e. the mean of the class centers.

Let us define the n-sphere ($n = N - 1$) in the N-dimensional space as $\mathcal{S} = \{v \in \mathbb{R}^N : \|v\| = 1\}$. Given a viewpoint $v \in \mathcal{S}$, we can define an orthogonal projection $P : \mathbb{R}^N \to \mathbb{R}^N$, whose $N-1$ columns are defined by the vectors orthonormal to v, and whose last column is equal to v. Then, a point $x \in \mathbb{R}^N$ can be projected to the $N-1$-dimensional space defined by v by computing $y = Px$ and dropping the last component of y. This point, $\bar{x}(v) \in \mathbb{R}^{N-1}$ can be interpreted as point x as seen from the viewpoint v. Its component parallel to v is invisible to the viewer.

Our objective is to find the viewpoint v^* on the n-sphere such that the sum of the squared distances between the centers of the adjacent classes is maximized. If we define $\bar{c}_k(v) \in \mathbb{R}^{N-1}$ to be the projected center of class k in the $N-1$-dimensional space defined by viewpoint v, we search for v^* maximizing

$$G(v) = \sum_{k=1}^{K-1} \|\bar{c}_{k+1}(v) - \bar{c}_k(v)\|^2 \tag{4.1}$$

subject to the constraint $\|v\| = 1$. Maximizing G(v) is equivalent to solving the following minimization problem:

$$\text{Minimize } F(v) = \sum_{k=1}^{K-1} [v^T(c_{k+1} - c_k)]^2 \text{subject to } \|v\| = 1\ . \tag{4.2}$$

Algorithm 4: Find the optimum viewpoint.

Data: Class centers: c_k, $k = 1, 2, ..., K$
Result: Optimum viewpoint: v^*

Initialize v_0 randomly such that $\|v_0\| = 1$;
MaxIter = 100; $\epsilon = 10^{-5}$; $\gamma_0 = 0.05$; $j = 0$;
while $j <$ *Maxiter* **do**
    Calculate $\nabla F(v_j)$ using 4.5 and 4.6;
    if $j > 0$ **then**
        Calculate γ_j using 4.7;
    end
    $\hat{v}_j = v_j - \gamma_j \nabla F(v_j)$;
    $v_{j+1} = \frac{\hat{v}_j}{\|\hat{v}_j\|}$;
    if $\cos^{-1}(v_{j+1}^T v_j) < \epsilon$ **then**
        $v^* = v_{j+1}$;
        break;
    end
    $j \leftarrow j + 1$
end

This is an optimization problem where the search space is constrained to a smooth Riemannian manifold. We use gradient descent together with the retraction formulation for a spherical manifold in [144] to find v^*. The procedure is given in Algorithm 4. We randomly pick a viewpoint v_0 on $\mathcal{S}$ for initialization and update it as:

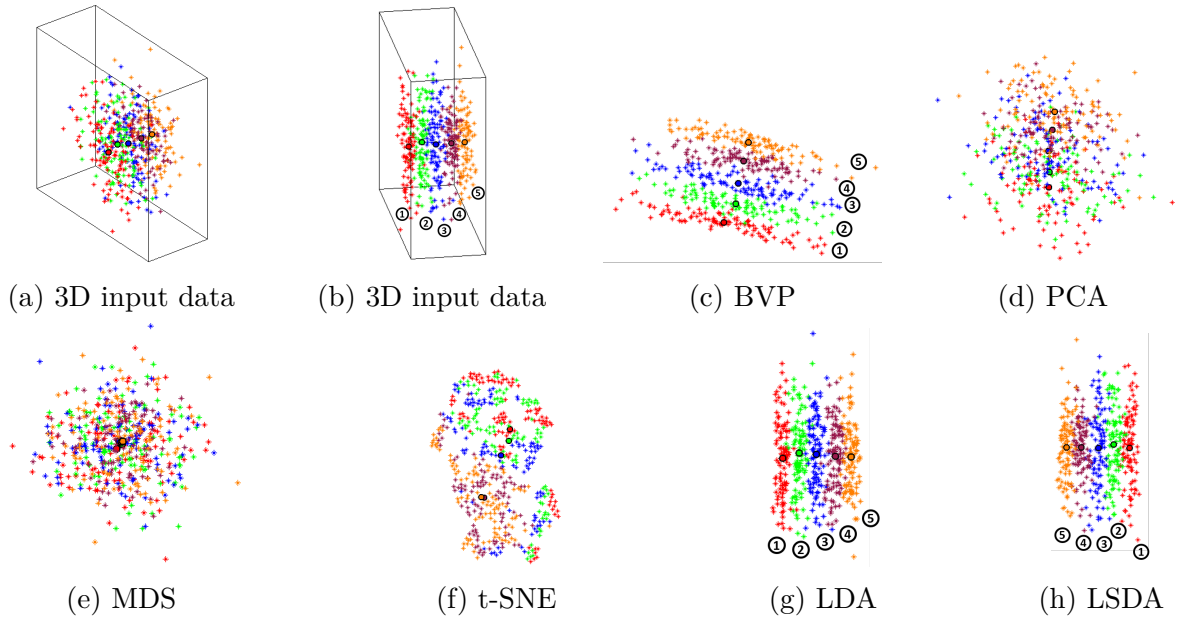
$$v_{j+1} = \mathrm{Retr}_{v_j}(\eta_j) \tag{4.3}$$

$$\eta_j = -\gamma_j \nabla F(v_j) \; . \tag{4.4}$$

The gradient of $F(v)$ is equal to

$$\nabla F(v) = 2A^T A v \tag{4.5}$$

$$A = \begin{bmatrix} (c_2 - c_1)^T \\ (c_3 - c_2)^T \\ \vdots \\ (c_K - c_{K-1})^T \end{bmatrix} \tag{4.6}$$



(a) 3D input data (b) 3D input data (c) BVP (d) PCA

(e) MDS (f) t-SNE (g) LDA (h) LSDA

Figure 4.2 – Proposed best view point (BVP) algorithm in action with ordinal datasets by comparison with other classical dimensionality reduction techniques. Panels (a) and (b) are two views of the synthetic 3D ordinal data set. Panels (c) to (i) show results of dimensionality reduction from 3D to 2D. The black circles with numbers correspond to class labels.

We update step size γ_j according to the formula [145]:

$$\gamma_j = \frac{\left|(v_j - v_{j-1})^T |\nabla F(v_j) - \nabla F(v_{j-1})|\right|}{\|\nabla F(v_j) - \nabla F(v_{j-1})\|^2} \,. \tag{4.7}$$

The retraction for the sphere can be chosen as [144]:

$$\mathrm{Retr}_v(\eta) = \frac{v + \eta}{\|v + \eta\|} \,. \tag{4.8}$$

The iteration is stopped when the angle between v_j and v_{j+1} is smaller than ϵ, and the optimum viewpoint v^* is set to be equal to v_{j+1}. For all experiments, ϵ is set to 10^{-5}.

The cluster centers are then projected to the $N-1$-dimensional space defined by v^*. The whole procedure is repeated until the desired dimensionality is achieved.

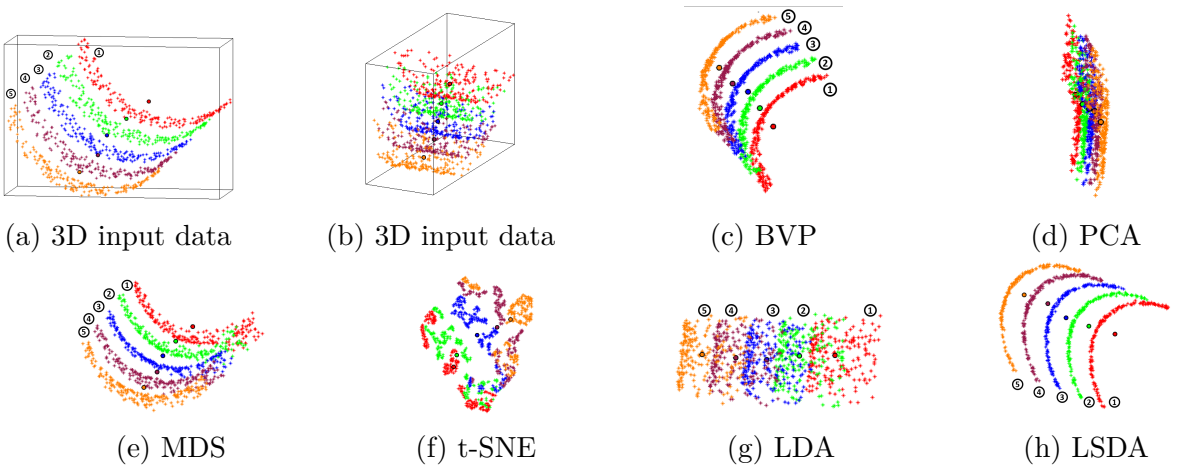
4.3 Results on dimension reduction

Since we present our dimensionality reduction method as a complementary visualization tool for ordinal datasets rather than reducing the dimension of inputs of classification techniques, we provide visual results only. We compare the best-view projection method with three unsupervised methods 1) PCA, 2) MDS, 3) T-SNE) and two supervised methods 4) LDA, 5) LSDA. The comparison is based on: 1) whether the classes are well-separated, 2) whether the ordinality between classes is preserved, and 3) whether the distribution of the data is informative in the low-dimensional space. First, we present results with simulated data where the dimensionality is reduced from 3 to 2. Then, we provide comparisons with real ordinal datasets of higher initial dimensions.

4.3.1 Simulated data

We created two 3-dimensional ordinal datasets and employed our best-view projection method and other six algorithms to reduce the dimensionality to 2. Figures 4.2a and 4.2b show the first dataset, where there are five ordinal classes and the within-class distribution is Gaussian. 100 instances were generated for each class. The class labels are given in Fig. 4.2a. In this example, the direction of ordinality in the original space is not aligned with the principal axes of variation of the whole data; hence PCA fails to appropriately reduce the dimensionality as seen in Fig. 4.2d. MDS tries to place data points into 2D space such that the pairwise distances are preserved as much as possible. It does not take into account the class labels, and it fails when the instances of adjacent classes are close to each other in the original space (Fig. 4.2e). In t-SNE, local neighborhood of points are embedded to capture the local structure of the data together with clusters at several scales. We experimented thoroughly with varying the perplexity parameter, which is a measure of the effective number of neighbors used in the algorithm. We give the best result, with perplexity 40, in Fig. 4.2f. Although t-SNE manages to group together samples of the same class in local clusters, the global ordinality present in the data is lost in the resulting 2D space.

Notice that PCA, MDS and t-SNE are unsupervised techniques, which are great for revealing the important global or local structure of data. However, the class distribution is not necessarily aligned with that structure in many cases, as in this example. The supervised techniques, LDA (Fig. 4.2g) and LSDA (Fig. 4.2h) are able to reduce the dimensionality to 2 with good class separation while preserving the ordinality. Our BVP



(a) 3D input data (b) 3D input data (c) BVP (d) PCA

(e) MDS (f) t-SNE (g) LDA (h) LSDA

Figure 4.3 – Same as Fig. 4.2 with an ordinal dataset composed of partial swiss rolls.

algorithm performs very similarly to LDA and LSDA (Fig. 4.2c). What it does is essentially to rotate of the 3D data given in Fig. 4.2a until it finds the best view that separates the adjacent cluster centers as well as possible.

The second simulated dataset is shown in Figs. 4.3a and 4.3b. Instances of each class belong to a 3D partial Swiss roll. 200 instances were generated for each class. The classes are separated by an offset in 3D in accordance to their ordinality. Similar to the first dataset, PCA is not effective (Fig. 4.3d) since the principal axis of global data distribution is not aligned with the direction of ordinality. MDA is successful in this case Fig. (4.3e) due to the fact that there is greater separation between instances from different classes in the original space as compared to the first dataset. For t-SNE, the best configuration was obtained with perplexity 30. In accordance with its objective, t-SNE gathered the data in local clusters in the 2D space, preserving the separability and ordinality to some degree Fig. 4.3f; however, the global nature of the data is not observable.

For this dataset, we observe that LDA failed to reduce the dimensionality properly (Fig. 4.3g). LDA searches for a projection that minimizes the distances of instances of each class to its center, and in this case, the class centers are located closer to the instances of adjacent classes in the original space. The result is a 2D configuration where the class separation is lost. Our BVP method does well in this case (Fig. 4.3c) as does LSDA (Fig. 4.3c), projecting the data such that the separation and ordinality between classes are preserved together with an informative distribution in the reduced space.

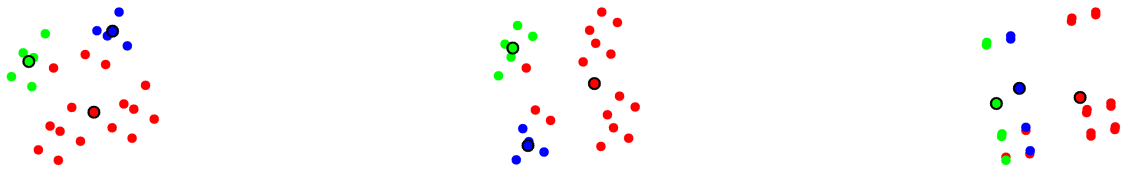
4.3.2 Real ordinal data

We tested our dimensionality reduction technique on real ordinal classification datasets [146, 147]. Due to lack of space, we only give visual comparisons with the two supervised techniques; LDA and LSDA. The datasets and their properties are given in Tab. 4.1.

Table 4.1 – Real ordinal datasets used for the experiments [146, 147] (I is the total number of instances, Q is the dimensionality of the original data and K is the number of classes).

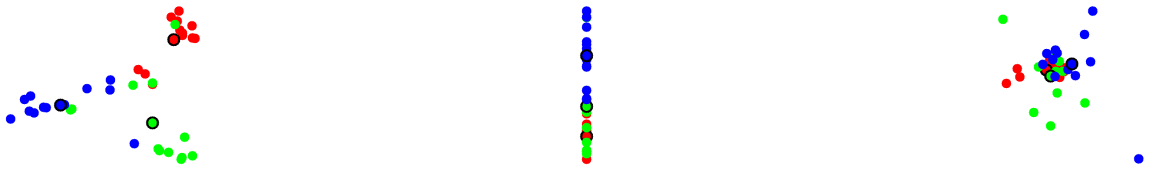
Dataset	I	Q	K	**Class Distribution**
contact-lenses	24	6	3	(15,5,4)
pasture	36	25	3	(12,12,12)
squash-stored	52	51	3	(23,21,8)
newthyroid	215	5	3	(30,150,35)
car	1728	21	4	(1210,384,69,65)
bondrate	57	37	5	(6,33,12,5,1)

Figure 4.4 through 4.9 show the dimensionality reduction results obtained by BVP in comparison to LDA and LSDA. For all cases, BVP was able to provide a glimpse of the data distribution and relative relations of the classes with respect to each other. LDA showed good performance in some cases (Figs. 4.4b and 4.7b). In the other cases, LDA pulled the instances close to the class centers, causing a loss of information within class distribution. Not originally designed for dimensionality reduction for visualization, LSDA did not retain class separability in 2 dimensions for the real datasets, except for the dataset newthyroid (Fig. 4.7c). These results demonstrate that for many ordinal datasets, classical dimensionality reduction techniques may fail to provide a proper visualization of the high-dimensional features, and our method can be used as an alternative tool to map high-dimensional data to 2D for interpretation.



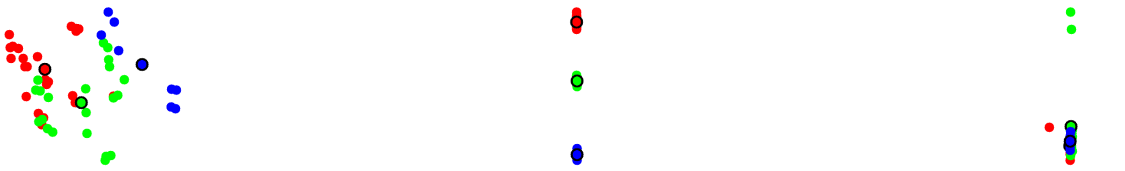
(a) BVP (b) LDA (c) LSDA

Figure 4.4 – 2D visualization of the dataset contact-lenses.



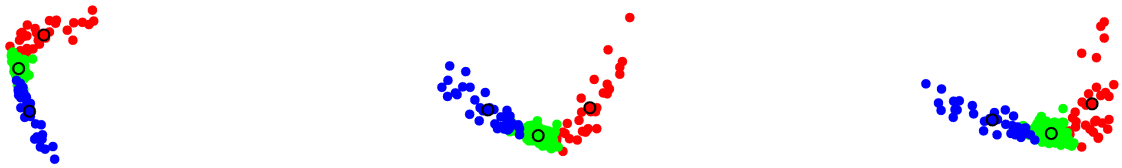
(a) BVP (b) LDA (c) LSDA

Figure 4.5 – 2D visualization of the dataset pasture



(a) BVP (b) LDA (c) LSDA

Figure 4.6 – 2D visualization of the dataset squash-stored

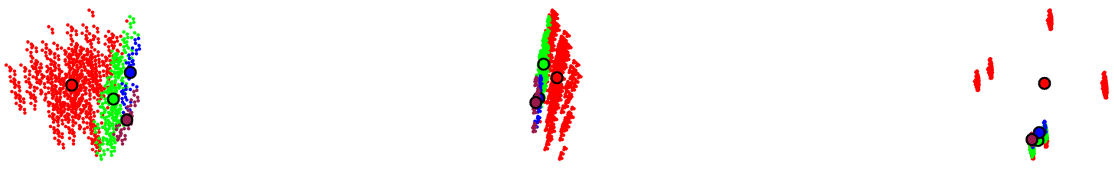


(a) BVP (b) LDA (c) LSDA

Figure 4.7 – 2D visualization of the dataset newthyroid

(a) BVP (b) LDA (c) LSDA

Figure 4.8 – 2D visualization of the dataset bondrate



(a) BVP (b) LDA (c) LSDA

Figure 4.9 – 2D visualization of the dataset car

4.4 Ordinality metrics

To go beyond the sole visualization of the dimension reduction, we now target to quantify ordinality in the reduced latent space. In the literature, several researches have developed specific metrics dedicated to ordinal classification problems [148, 149, 150, 128]. Nevertheless, these references focus on the characterization of the classification performance themselves, while we care here about the interpretability of the latent space before classification. Related works [151] proposed a framework, to solve the performance versus interpretability trade-off in the context of ordinal problems. Although this work covers interpretability of ordinality, it is related to In-Model and equation-Model interpretability techniques [128] while we focus on Pre-Model here. As the most related work, [152] focused on the intersection between instances of ordinal data in the latent space. In this investigation, the authors proposed a projection method from N-dimensional latent space to 1-dimensional latent space, using insights about the class distribution obtained from pairwise distance calculation between instances of all classes. The idea in [152] is to project an instance in the 1D interval of a given class, according to its distance to instances of other classes. A threshold is set manually, to split the interval on segments of classes. The output projection is then used to perform an ordinal regression.

By contrast to the pairwise method in [152], we propose metrics to quantify the intersection between classes by penalizing the ordinal distance of these misclassifications. This can not be deduced from [152], where all instances are mapped in their corresponding class interval. In addition, unlike the thresholds selected manually in [152], our proposed metric is fully automatic. Moreover, we complement our new metric of ordinal intersection between classes with another metric assessing the ordinality at the level of the centroids of the clusters of the classes. This can help to discriminate ordered and unordered latent space for noise-free datasets having no class intersection in the latent space. This aspect was not taken into account in [152] which assumes that centroids in latent space are already well aligned and does not quantify their order in this latent space. We detail the expression of these two metrics in the next section.

4.4.1 Deviation from ordinality metric

The deviation from ordinality (DFO) is a metric that quantifies how much the order of the centroids departs from the expected order after dimension reduction. Technically, it compares the position of centroids in the path connecting them in the expected order

$k = \{1, \ldots, K\}$ (reference path), with the position of the same centroids in the shortest path. The shortest path is a path where the nearest centroids are connected to each other based on the Euclidean distance. This metric can be considered as an edit distance metric. Several edit distance metrics already exist in the literature [153]. Here, we propose a simple binary output: ordered or unordered. The mathematical formulation of deviation from ordinality metric is a simple subtraction between order of centroids in shortest and reference paths

$$DFO^k = \frac{p_k^{ref} - p_k^{short}}{K-2} \ , \tag{4.9}$$

where p_k^{ref} is the position of centroid k in the reference path and p_k^{short} is the position of centroid k in the shortest path. The subtraction is normalized by the maximum distance $K-2$. Centroid of class 1 is chosen as the starting point for the reference path and the shortest path, hence the normalization by $K-2$ provides a metric between 0 and 1. DFO^k equals to 0 if the centroid k has the same position in both paths (ordered case) while DFO equals 1 if the centroid k is displaced to position K (the extreme case). An average value of the DFO^k over all k can then be computed to provide a global assessment in addition to the local order DFO^k associated to each class.

4.4.2 Inter-class intersection metric

We now introduce a second metric to quantify ordinality coined as *Inter-Class intersection*. The metric has two outputs: the first scalar quantifies the severity of intersection between classes and the second is a binary scalar which states if the intersection is only between adjacent classes or also between non-adjacent classes.

The definition of intersection between classes depends on the decision boundaries for classes. In this work, we assume elliptic regions for data are reduced to two dimensions to account for second-order statistics of the data. The inter-class intersection metric we propose can be generalized to higher dimensions (e.g. 3D ellipsoids) and other decision boundaries such as polygonal shapes.

Here, the computation of the *Inter-Class intersection* is performed in the following way. First, a boundary B_k (ellipse) is computed around each class l_k (algorithm 5). The ratio a_j^k of instances x_i^k inside the boundary B_k is evaluated by counting the number of instances of the class l_k inside B_k, normalized by the dimension I_k. The output is a confusion matrix $K \times K$. The second step is to penalize boundaries in the confusion matrix, containing instances of non-adjacent classes (equation 4.10). This is achieved by

multiplying the ratio a_j^k by the square distance $(j-k)^2$ in the matrix $K \times K$. Normalization is applied on all matrix-elements by dividing over the sum of square distances $(j-k)^2$, so that the *Inter-Class intersection* value is between 0 and 1, as given in

$$IC^{B_k} = \frac{\sum_{j=1}^{K} a_j^k \times ((j-k))^2}{\sum_{j=1}^{K} (j-k)^2} \text{for k=}\{1,\ldots,\text{K}\} \; . \tag{4.10}$$

Inter-Class intersection IC^{B_k} equals to 0 when there is no intersection between classes. To be able to separate the case of intersection only between adjacent classes and intersection between non-adjacent classes, we add complementary information through a binary scalar (BS). If all the non-diagonal and the non-adjacent cell values in the matrix $K \times K$ are equal to 0, the binary output is equal to 0. Otherwise, the binary output is equal to 1.

Algorithm 5: Pseudo-code to compute the *Inter-Class intersection* metric.

Data: Coordinates of instances x_j^k of all classes l_k.
Result: KxK matrix containing the percentage of instances x_j^k of all class l_k in each boundary B_k

1 Fit an ellipse B_k over instances x_j^k, by computing the covariance matrix and eigen vectors and value; find instances x_j^k of class l_k inside the boundary B_k;
2 Normalize the number of instances x_j^k found by the dimension I_k of the class l_k;
3 Save all ratios in an KxK matrix, where K is the number of classes;

4.5 Results on ordinality metrics

The two ordinal metrics of the previous section have been applied on the synthetic and ordinal data of sections 4.3.1 and 4.3.2. The results are provided in Tabs. 4.2 to 4.7. The quantitative results are in accordance with the visualizations in Figs. 4.2 to 4.9. It appears that BVP is providing excellent results, with almost no deviation from ordinality and low mean inter-class intersection. By comparison with the other classical dimension reduction methods, BVP provides better results than PCA, TSNE, ISOMAP and MDS. The closest quantitative results of BVP with existing methods is with LDA and LSDA.

On some datasets (such as bondrate), BVP shows a deviation from ordinality not better than LDA and LSDA. However, in other datasets (such as contact lenses), BVP outperforms LDA and LSDA. This demonstrates the complementary role of BVP in rela-

Table 4.2 – *Inter-Class intersection* and *Deviation from ordinality* values extracted from latent spaces generated after applying dimension reduction techniques (PCA, TSNE, LDA, ISOMAP, MDS, LSDA and BVP) on synthetic dataset in Fig. 4.2a.

DRT	IC^{B1}	IC^{B2}	IC^{B3}	IC^{B4}	IC^{B5}	**Mean IC**	**BS**	DFO^2	DFO^3	DFO^4	DFO^5	**Mean DFO**
ℒPCA	0.99	0.48	0.33	0.49	0.96	0.65	1	0	0	0	0	0
ℒTSNE	0.98	0.49	0.33	0.50	0.84	0.63	1	0	0	0	0	0
ℒLDA	0.01	0.01	0.01	0.01	0.00	**0.01**	0	0	0	0	0	0
ℒISOMAP	0.99	0.48	0.33	0.49	0.93	0.64	1	0	0	0	0	0
ℒMDS	1.00	0.47	0.32	0.49	0.98	0.65	1	0	0	0	0	0
ℒLSDA	0.02	0.02	0.02	0.02	0.01	**0.01**	0	0	0	0	0	0
ℒBVP	0.03	0.03	0.03	0.04	0.02	**0.03**	1	0	0	0	0	0

Table 4.3 – *Inter-Class intersection* and *Deviation from ordinality* values extracted from latent spaces generated after applying dimension reduction techniques (PCA, TSNE, LDA, ISOMAP, MDS, LSDA and BVP) on synthetic dataset in Fig. 4.3.

DRT	IC^{B1}	IC^{B2}	IC^{B3}	IC^{B4}	IC^{B5}	**Mean IC**	**BS**	DFO^2	DFO^3	DFO^4	DFO^5	**Mean DFO**
ℒPCA	0.08	0.12	0.18	0.12	0.10	**0.12**	1	0	0	0	0	0
ℒTSNE	0.11	0.06	0.27	0.18	0.25	0.18	1	0	0	0	0	0
ℒLDA	0.09	0.12	0.18	0.12	0.10	**0.12**	1	0	0	0	0	0
ℒISOMAP	0.01	0.05	0.11	0.09	0.11	**0.08**	1	0	0	0	0	0
ℒMDS	0.09	0.13	0.18	0.12	0.10	0.13	1	0	0	0	0	0
ℒLSDA	0.08	0.12	0.19	0.12	0.10	**0.12**	1	0	0	0	0	0
ℒ BVP	0.09	0.12	0.19	0.12	0.10	**0.12**	1	0	0	0	0	0

Table 4.4 – *Inter-Class intersection* and *Deviation from ordinality* values extracted from latent spaces generated after applying dimension reduction techniques (PCA, TSNE, LDA, ISOMAP, MDS, LSDA and BVP) on pasture dataset.

DRT	IC^{B1}	IC^{B2}	IC^{B3}	**Mean IC**	**BS**	DFO^2	DFO^3	**Mean DFO**
ℒ PCA	0.12	0.3	0.35	**0.25**	1	0	0	0
ℒ TSNE	0.3	0.4	0.448	0.38	1	0	0	0
ℒ LDA	0	0	0	**0**	0	0	0	0
ℒ ISOMAP	0.05	0.384	0.334	0.26	1	0	0	0
ℒ MDS	0.12	0.3	0.35	0.26	1	0	0	0
ℒ LSDA	0.72	0.35	0.936	0.67	1	0	0	0
ℒ BVP	0.05	0.34	0.27	**0.22**	1	0	0	0

tion with the existing literature on dimension reduction. It is important to underline the quality of BVP on the Inter-class intersection metrics. Indeed, BVP is designed based on a metric applied on the centroids of the cluster and does not take into account the dispersion around these clusters. The encouraging results found on Inter-class intersection indicate that BVP also has the potential to be used for classification purposes. This is also in agreement with its good performance in comparison with LDA which is specifically designed for classification. These are interesting perspectives that we currently develop.

Table 4.5 – *Inter-Class intersection* and *Deviation from ordinality* values extracted from latent spaces generated after applying dimension reduction techniques (same as in table 4.4) on bondrate dataset.

DRT	IC^{B1}	IC^{B2}	IC^{B3}	IC^{B4}	Mean IC	BS	DFO^2	DFO^3	DFO^4	Mean DFO
$\mathcal{L}$ PCA	0.81	0.42	0.42	0.97	0.66	1	0.5	0.5	1	0.67
$\mathcal{L}$ TSNE	0.98	0.42	0.43	0.99	0.70	1	1	0	1	0.67
$\mathcal{L}$ LDA	0.05	0.10	0.02	0.01	**0.04**	0	0	0	0	0
$\mathcal{L}$ ISOMAP	0.78	0.42	0.42	0.93	0.64	1	0.5	0.5	1	0.67
$\mathcal{L}$ MDS	0.81	0.42	0.42	0.97	0.66	1	0.5	0.5	1	0.67
$\mathcal{L}$ LSDA	0.95	0.42	0.38	0.49	**0.56**	1	0	0.5	0.5	0.33
$\mathcal{L}$ BVP	0.66	0.42	0.43	0.97	**0.62**	1	0.5	0.5	1	0.67

Table 4.6 – *Inter-Class intersection* and *Deviation from ordinality* values extracted from latent spaces generated after applying dimension reduction techniques (same as in table 4.4) on contact lenses dataset.

DRT	IC^{B1}	IC^{B2}	IC^{B3}	Mean IC	BS	DFO^2	DFO^3	Mean DFO
$\mathcal{L}$ PCA	1	0.4	1	0.8	1	0	0	0
$\mathcal{L}$TSNE	1	0.28	1	0.76	1	1	1	1
$\mathcal{L}$LDA	1	0.01	0.10	**0.37**	0	0	0	0
$\mathcal{L}$ISOMAP	1	0.4	1	0.8	1	1	1	1
$\mathcal{L}$MDS	1	0.4	1	0.8	1	1	1	1
$\mathcal{L}$LSDA	0.92	0.01	0.10	**0.35**	0	1	1	1
$\mathcal{L}$BVP	0.76	0.01	0.10	**0.29**	0	0	0	0

Table 4.7 – *Inter-Class intersection* and *Deviation from ordinality* values extracted from latent spaces generated after applying dimension reduction techniques (same as in table 4.4) on newthyroid dataset.

DRT	IC^{B1}	IC^{B2}	IC^{B3}	Mean IC	BS	DFO^2	DFO^3	Mean DFO
$\mathcal{L}$ PCA	0.12	0.04	0.11	**0.09**	0	0	0	0
$\mathcal{L}$TSNE	0.03	0.24	0.02	0.10	0	0	0	0
$\mathcal{L}$LDA	0.05	0.04	0.18	**0.09**	0	0	0	0
$\mathcal{L}$ISOMAP	0.43	0.04	0.18	0.21	1	0	0	0
$\mathcal{L}$MDS	0.50	0.04	0.36	0.30	1	0	0	0
$\mathcal{L}$LSDA	0.08	0.04	0.17	0.10	0	0	0	0
$\mathcal{L}$BVP	0.09	0.04	0.13	**0.09**	0	0	0	0

Table 4.8 – *Inter-Class intersection* and *Deviation from ordinality* values extracted from latent spaces generated after applying dimension reduction techniques (same as in table 4.4) on squash-stored dataset.

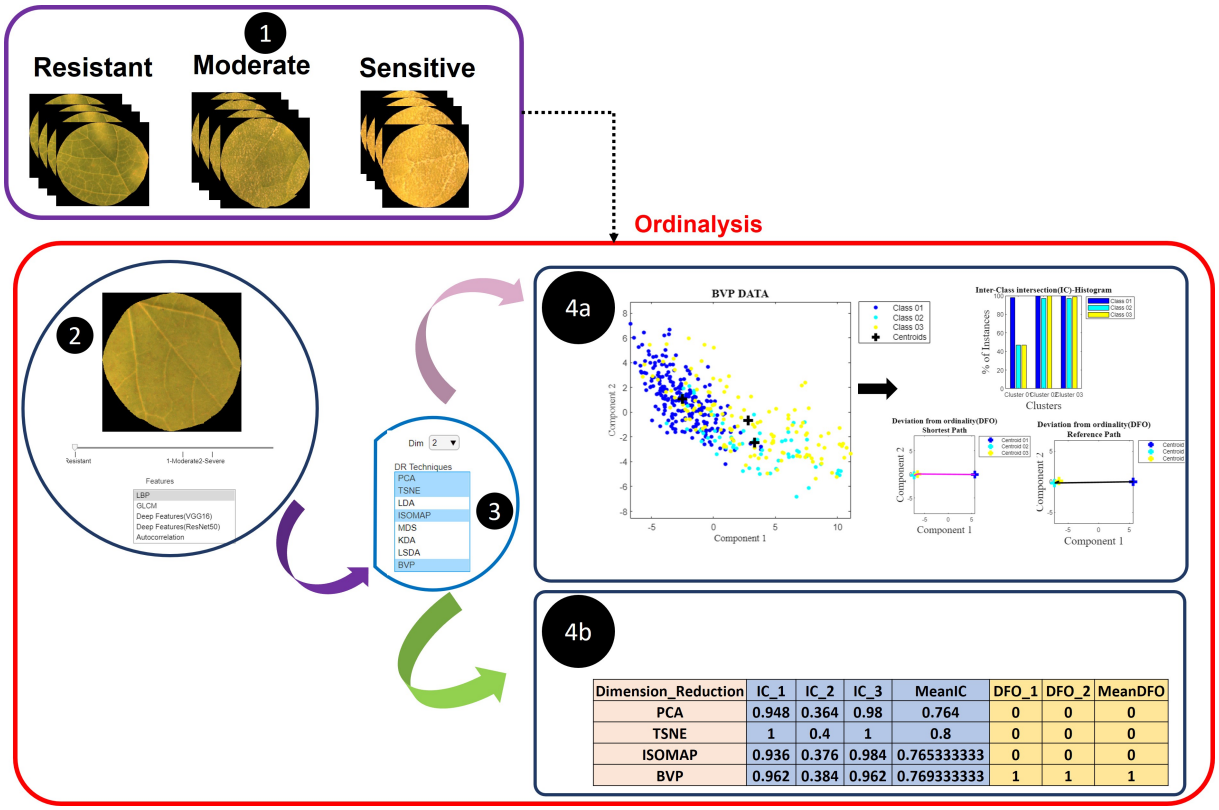
DRT	IC^{B1}	IC^{B2}	IC^{B3}	Mean IC	BS	DFO^2	DFO^3	Mean DFO
$\mathcal{L}$ PCA	0.60	0.40	0.20	**0.40**	1	0	0	0
$\mathcal{L}$TSNE	1	0.40	0.42	0.61	1	0	0	0
$\mathcal{L}$LDA	0	0	0	**0**	0	0	0	0
$\mathcal{L}$ISOMAP	0.60	0.40	0.21	0.40	1	0	0	0
$\mathcal{L}$MDS	0.78	0.40	0.24	0.47	1	0	0	0
$\mathcal{L}$LSDA	0.97	0.39	0.75	0.70	1	1	1	1
$\mathcal{L}$BVP	0.37	0.39	0.04	**0.27**	1	0	0	0

4.6 Application to variety testing

To enable examiners to visualize the ordinal feature space in the numerical distinctness test, we have developed a standalone application called *Ordinalysis*. This application

incorporates the dimension reduction technique BVP and ordinality metrics presented in this chapter. The added value of *Ordinalysis* remains in the non-dependency on the coding experience of the user.

In Annex E, we presented a pipeline to automate the measurement of the resistance of melon varieties to powdery mildew pathogen. In this section, we demonstrate the potential of *Ordinalysis* at selecting the best machine learning training strategy. We verify if the visual results and the quantification metrics will select the same machine learning strategy using the results obtained in Annex E. Fig. 4.10 presents the workflow to process the powdery mildew ordinal data in *Ordinalysis*. The results obtained are presented in Annex H.



Dimension_Reduction	IC_1	IC_2	IC_3	MeanIC	DFO_1	DFO_2	MeanDFO
PCA	0.948	0.364	0.98	0.764	0	0	0
TSNE	1	0.4	1	0.8	0	0	0
ISOMAP	0.936	0.376	0.984	0.765333333	0	0	0
BVP	0.962	0.384	0.962	0.769333333	1	1	1

Figure 4.10 – Pipeline to illustrate the use of *Ordinalysis* at automating variety testing protocols. (1): ordinal data presented in Annex E. (2): interactive space to visualize image data in Ordinalysis and select the type of features to be extracted. (3): select the dimension reduction techniques. (4a): visualization of the latent space for each dimension reduction technique and the associate plots of both ordinal metrics. (4b): values of ordinal metrics exported as CSV file.

4.7 Conclusion

In this chapter, we presented a novel and intuitive dimension reduction technique for the visualization of high-dimensional data for ordinal classification. We provided visual comparisons with various dimensionality reduction techniques on both simulated and real datasets, and demonstrated that BVP is capable of retaining separability and class order in cases where the other dimension techniques failed. This visualization step is important for ordinal classification to ensure that the latent space, on which a final classification is to be performed by a machine, is interpretable to the human eye [128]. We also provided two metrics to quantify the ordinality in this reduced latent space. The whole set of tools (dimension reduction and ordinality metrics) are made available as a standalone application called, *Ordinalysis* (The description of the application and its application to variety testing are presented in Annex H).

Chapter 5

Conclusion and Perspectives

5.1 Synthetic view of methodological contributions

Variety testing is a set of tests performed by examiners to certify candidate varieties to be registered in the catalog of varieties for commercialization. Currently, these tests are conducted manually. The specific contraints in variety testing have limited so far the transition toward numerical measurements based on computer vision. On the other hand, the tools used in precision agriculture and plant breeding are not straightforward and cost-effective to be exploited in variety testing.

In this thesis, we contributed to computer vision and machine learning for plant variety testing by proposing methodological solutions and engineering tools to incorporate machine learning and computer vision in variety testing. We targeted the DUS variety testing, particularly the distinctness test. Our contributions were organized into three parts.

In the first part, we proposed a computer vision pipeline based on point cloud processing to delineate apple trees in variety testing orchards [12]. This work's novelty lies in fusing the 3D reconstruction of the orchard in the winter and the harvest periods. We delineated individual trees on the leaf-off model from winter and employed the 3D model from the harvest period to localize apples. We determined the tree membership of each apple in the harvest point cloud by mapping their locations onto the winter point cloud, where individual trees are separated.

In the second part, we considered the distinctness as a classification problem. We demonstrated the efficiency of supervised machine learning via the optimal transport at separating the mutant and non-mutant of Gala based on color information [88]. Alternatively, we proposed a novel unsupervised machine learning methodology to automate shape measurement, directly using sketches from the catalog of variety testing, as a numerical ground truth [89].

In the third part, we proposed a dimension reduction technique called best-view pro-

jection (BVP) and two quantification metrics (deviation from ordinality and inter-class intersection) to visualize and quantify the ordinality in the feature space [142, 143]. We demonstrated the impact of BVP and the quantification metrics at selecting the best training strategy to automate the measurement of the resistance of melon varieties to powdery mildew pathogen.

5.2 Engineering contributions

In this thesis, in addition to the proposed methodological contributions, we also developed computer vision and machine learning tools ready to be used, for traits highlighted in green in Fig. 5.1. These tools include annotated datasets, 3D models, a sorting machine, a web application, a standalone application and deep learning models (see Annex A). These resources were shared with the community of variety testing in the European project INVITE (https://www.h2020-invite.eu/). The process of automating the measurement of traits using computer vision and machine learning includes three steps: acquisition, annotation, and modeling. Our tools are dedicated to serve the examination offices in each step.

We contributed in the step of data creation by developing a low-cost sorting machine ($\approx 10k€$) to acquire images of fruits at a high rate. Such a robot fits with the practices of measurements in variety testing, in particular, the ones in the post-harvest distinctness test, unlike the costly commercialized robots developed in precision agriculture ($\approx 100k€$), where raw data that produce the measurement is inaccessible.

Annotation is known for being the bottleneck in computer vision and it can be costly for examination offices. We aim to share annotated data to democratize the use of computer vision in variety testing. In addition, annotated data can also serve for transferring the knowledge of machine learning models between similar crops [154]. For instance, one examination office assessing the varieties of pears can use the annotated data of apples to train machine learning to automate the measurements of traits of pears. We also contribute at the creation of annotated data by developing *PanoVar* (https://panovar.org/). *PanoVar* is a web application dedicated to speeding up annotation. It offers a shared space for examiners to conduct their measurements interactively, compares their scoring and runs automatic measurements through trained machine learning models. We aim that this application will help examination offices and prevent them from paying companies to build costly applications.

On the other hand, machine learning models can serve as examination offices to compare the efficiency of existing commercialized tools by comparing their results with the ones of the machine learning and deep learning models developed in this thesis. In this context, we also developed *Ordinalysis* that can impact the selection of training strategies. *Ordinalysis* is a standalone application that was developed to visualize and quantify the ordinality in the feature space where machine learning algorithms make decisions. This application can serve examiners in the distinctness test to visualize the varieties and verify if the meaningful order of data is respected. More details about this application are available in Annex A and H.

Tree	Fruit	Flower
type, habit ,type of bearing	**size, height, diameter**	arrangement of petals
One-year-old shoot: length of internode, color on sunny side	size of eye	aposition of stigmas relative to anthers
leaf blade: length, width ratio width/length intensity of green color, incisions of margin (upper half)	length of sepal	**time of the pic of flowering**
	shape	**time of ending of flowering**
	colour	
	width of stripes	
	area of russet	
	lenticels: number and size	
	stalk: length, thickness depth of cavity, width of cavity	
	eye basin (depth, width)	

Figure 5.1 – Traits of the distinctness tests mentioned in UPOV catalog. Traits highlighted in green benefited from methodological contributions and engineering tools proposed in this thesis.

5.3 Discussion

In this thesis, we provided specific perspectives at the end of each chapter. In this section, we rather discuss how to adjust the variety testing practices to decrease the complexity of the proposed computer vision and machine learning methodologies.

In this thesis, we have demonstrated that, indeed, computer vision and machine learning can fit perfectly to shift toward numerical practices and measurements in variety testing. The incorporation of these techniques could even be more successful if there were less constraints in variety testing. For instance, the pipeline proposed in chapter 2 was

complex because the trees were planted tightly, which is a specificity of the variety testing orchards. One can imagine that if the trees were well separated, there would be no need to afford computation power to run 3D point cloud processing. Instead, machine learning can be implemented easily to automate measurements of traits on 2D images. Another illustration of the complexity of variety testing relies in the current scoring system. At the time when the catalog of variety testing was introduced, the commercialized varieties had different genotyping signatures. Later, several new varieties were created via the crossing between the registered varieties, leading to a common genotyping signature between all the varieties. This impacted variety testing because the scoring system to measure the traits (phenotypes) was not efficient to capture the difference between the varieties. Recently, UPOV introduced intermediate classes in the scoring system to help examiners to capture the minimal difference between varieties. However, the threshold to differentiate these intermediates scores is often neither perceptible by the examiners nor by the computer vision and machine learning algorithms that process the images. An alternative method would be to restore the old scoring system and use the machine learning descriptors to capture the slight difference between the varieties.

The quality of the variety testing catalog can also impact the transition to automated measurements of trait. In chapter 3, we proposed an idea to use the sketches in the catalog of variety testing as a numerical ground truth to automate the measurements of fruit shape. In such a way, we avoid asking examiners to annotate images. However, the difficulty faced is that the quality of the sketches is mediocre. All of them were just drawn with no respect to the similarity to the real fruit shape. In the spirit of improving variety testing protocols, we invite UPOV to revisit these catalogs. For instance, for apple shape, a proposition for the examiners is to select the actual shape of apples per category (Flat, ellipsoid, etc.) and assign it as a reference in the catalog.

5.4 Publications

Journal Articles

- Zine-El-Abidine Mouad, Helin Dutagaci, Gilles Galopin, and David Rousseau. "Assigning apples to individual trees in dense orchards using 3D colour point clouds." Biosystems Engineering 209 (2021): 30-52.
- Zine-El-Abidine Mouad, Helin Dutagaci, Pejman Rasti, and David Rousseau. "Apple shape classification based on drawings in variety testing catalogs." Biosystems Engineering (2022). **(Under review)**
- Zine-El-Abidine Mouad, Helin Dutagaci, and David Rousseau."*Ordinalysis*", SoftwareX, 2022. **(Under review)**

International conferences

- Geoffroy Couasnet, Mouad Zine El Abidine, François Laurens, Helin Dutagaci, and David Rousseau. "Machine learning meets distinctness in variety testing." In Proceedings of the IEEE/CVF International Conference on Computer Vision (**ICCV**), pp. 1303-1311. 2021.
- Zine-El-Abidine Mouad, Helin Dutagaci, and David Rousseau. "Dimensionality Reduction for Ordinal Classification." In 29th European Signal Processing Conference (**EUSIPCO**), pp. 1531-1535. IEEE, 2021.
- Zine-El-Abidine Mouad, Sabine Merdinoglu-Wiedemann, Pejman Rasti, Helin Dutagaci, and David Rousseau. "Machine Learning-Based Classification of Powdery Mildew Severity on Melon Leaves." In 9th International Conference on Image and Signal Processing (**ICISP**), 2020.
- Zine-El-Abidine Mouad, Helin Dutagaci, and David Rousseau. "Reducing the complexity of annotation through transfer learning from indoor to outdoor. Application to Russeting classification." In 31st International Horticultural Congress (**IHC**), 2022.**(in press)**
- Zine-El-Abidine Mouad, Helin Dutagaci, and David Rousseau. "Automatic apple detection in orchards with computer vision and machine learning." In 31st International Horticultural Congress (**IHC**), 2022. **(in press)**
- Zine-El-Abidine Mouad, Helin Dutagaci, and David Rousseau. "Motion-based acquisition to speed up annotation. Application to Russeting classification." In 7th International Plant Phenotyping Symposium (**IPPS**), 2022. **(in press)**

National conferences

- Zine-El-Abidine Mouad, Helin Dutagaci, Pejman Rasti and David Rousseau. Vers une quantification de l'interprétabilité des espaces latents. 27ème **GRETSI**, 2022. **(in press)**

Chapter 6

Annex A: Annotated datasets, machine and deep learning models and application developed during this PhD

6.1 Annotated datasets and machine learning models

6.1.1 Images of apple trees in harvest period

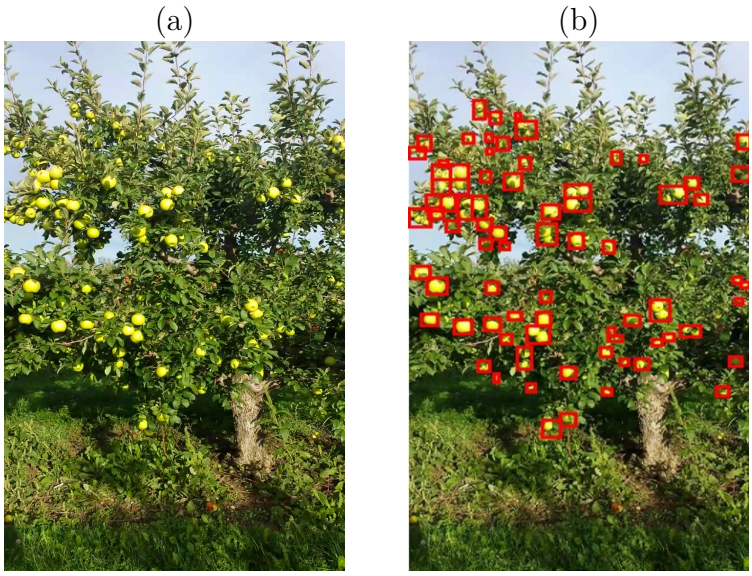


Figure 6.1 – (a): image of apple tree. (b): rectangles drawn around the apples.

- size of data: 700 images and 700 text files containing the coordinates of the apples in the corresponding images.

- type of annotation: coordinates of rectangles drawn around apples;
- computer vision task: detection;
- potential variety testing applications: apple counting;
- machine and deep learning models: YoloV4 Tiny [155];

6.1.2 3D point cloud of apple trees in harvest period

Figure 6.2 – (a): 3D point cloud of apple trees. (b): 3D point cloud of apple trees with ground truth apple locations.

- size of data: 7 3D point cloud of apple trees in harvest period;
- annotation: ground truth apple locations;
- computer vision task: detection;
- potential variety testing applications: fruit counting, estimation of volume, localization for fruit picking;
- algorithm: 3D point cloud processing;

6.1.3 3D point cloud of apple trees in winter period

- size of data: 7 3D point clouds of apple trees in winter;
- annotation: points of the branches, the trunks, the trellis-wires and the poles are respectively colored differently;
- computer vision task: segmentation;
- potential variety testing applications: Localization of trees in the orchards, estimation of the type, the habit and the dimensions of trees;
- algorithm: point cloud processing;

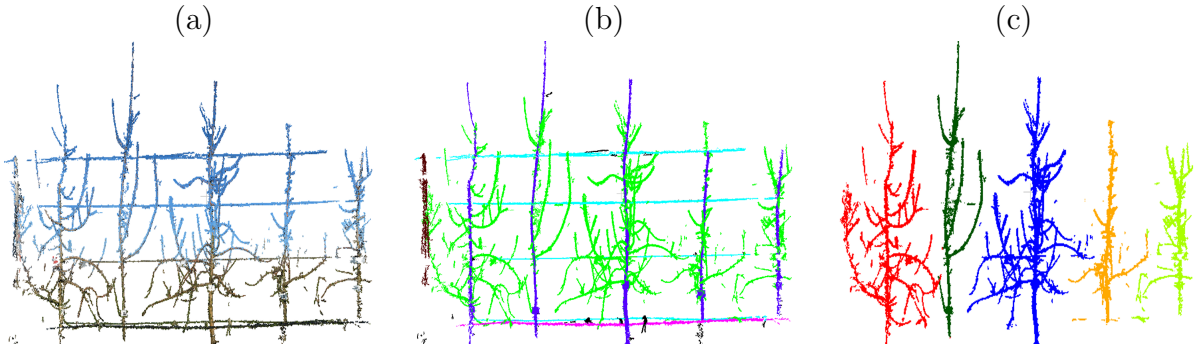


Figure 6.3 – (a): 3D point cloud of apple trees in winter. (b): manually generated Ground Truth (blue: trellis wires, red: tree trunks, black: support poles, green: branches). (c): labels obtained by our semantic segmentation method (see chapter 2).

6.1.4 Images of apple trees in flowering period

Figure 6.4 – (a): images of apples trees in the flowering period. (b): pixels of flowers in the first row are marked with white pixels.

- size of data: 50 images and 50 masks;
- annotation: pixels that belong to flowers are colored differently;
- computer vision task: detection;
- potential variety testing applications: estimation of intensity, the beginning and the end of flowering;
- machine and deep learning models: UNET [156];

Figure 6.5 – 10 cut apples classified as Globose.

6.1.5 Images of apple fruits for measurement of shape

- size of data: 1800 images;
- annotation: classify the image into: Flat, Globose or Oval;
- computer vision task: classification;
- machine and deep learning models: shape features (see chapter 3) and support vector machine [157];

6.1.6 Images of apple fruits for measurement of color

Figure 6.6 – Mutant of Gala.

- size of data: 4800 images including mutant and non-mutant of Gala;

- annotation: the name of the mutant and non-mutant of Gala;
- computer vision task: classification;
- machine and deep learning models: color features and support vector machine [157];

6.1.7 Images of apple fruits for measurement of russeting in indoor and outdoor

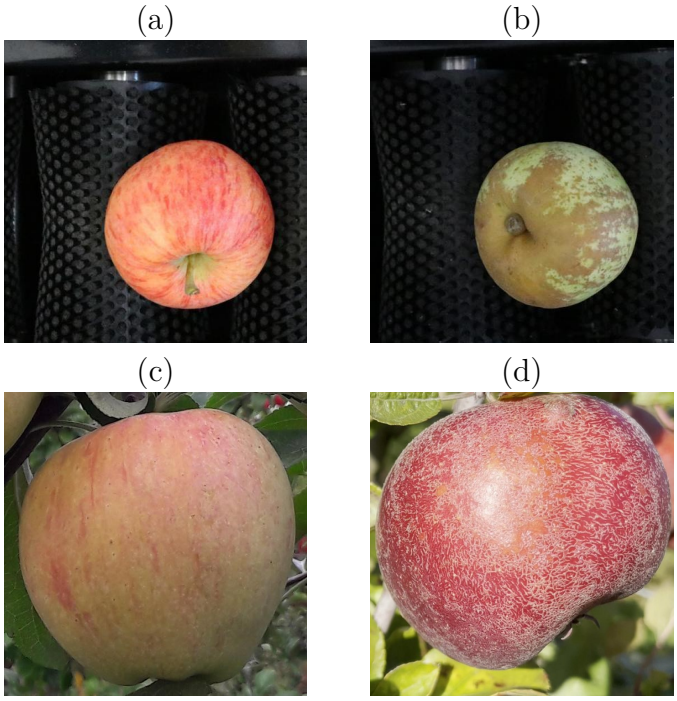


Figure 6.7 – (a): apple without russeting (indoor). (b): apple with russeting (indoor). (c): apple without russeting (outdoor). (d): apple with russeting (outdoor).

- size of data: 1000 images;
- annotation: classify the image into: with russeting, without russeting;
- computer vision task: classification;
- machine and deep learning models: CNN [158];

6.2 Low-cost acquisition system

To incorporate numerical practices in the post-harvest measurements during the distinctness tests, we developed a conveyor machine (see Fig. 6.8(a)). Technical details about this machine's functionality can be found in section 3.2.1 of this document. In addition, an object detection deep learning model was built to segment the apples from the background in real-time, leading to fast processing of images (see Fig. 6.8(b)).

(a) (b)

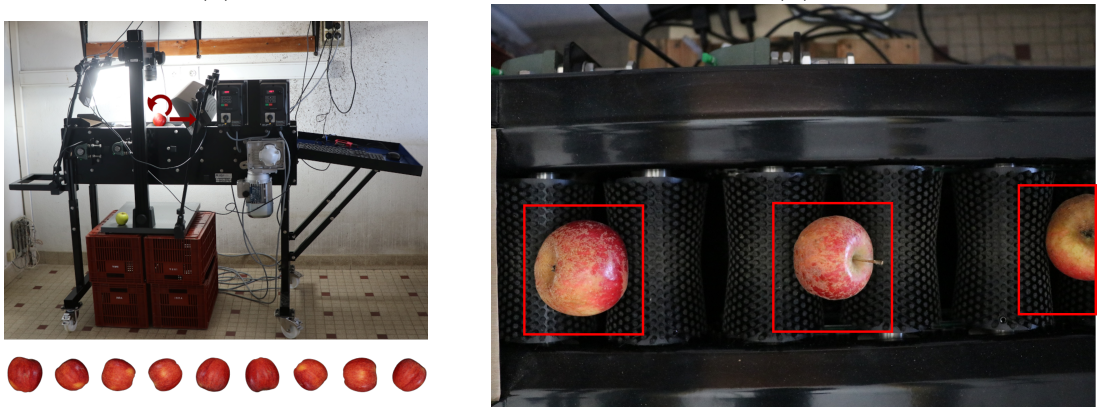


Figure 6.8 – (a) Low-cost sorting machine, (b) real-time apples detection and segmentation.

6.3 Applications

6.3.1 PanoVar

Motivation

In order to incorporate supervised machine learning in variety testing, models must be trained to measure traits. Training requires the existence of annotated data. The current protocol to ask examiners to annotate data is slow and inefficient. In this thesis, we developed a web application called *PanoVar* (https://panovar.org/) (see Fig. 6.9). *PanoVar* is dedicated to speed up the process of creating annotated data. It allows the examiners to upload images and score each trait following the scoring scale indicated in the UPOV catalog. The benefits of *PanoVar* are the followings:

Figure 6.9 – A screenshot of the first page of *PanoVar* (https://panovar.org/).

- annotate the traits numerically: it makes the comparison with the machine learning scoring fair as both assess the trait on the same images, unlike the current comparison between manual annotation and machine learning prediction;
- visualize the variability between examiners: comparing the scores assigned to the same images between examiners. It also allows understanding the errors of machine learning.
- speed up the annotation: the scoring is interactive. The examiners must click on the image and select the corresponding class for the measured trait. The Fig .6.10 shows a screenshot from *PanoVar*, where examiners are scoring the shape of cut apples;
- correcting the annotations: *PanoVar* allows the examiners to upload the annotations for verifications or modifications;
- creates historical data: the list of the images names and the corresponding classes can be exported in a CSV file. Such files can serve for stability tests in DUS variety testing;

Programming environment

PanoVar was built using the web application framework *FLASK* 1.1.1. Flask is a web application framework written in Python.

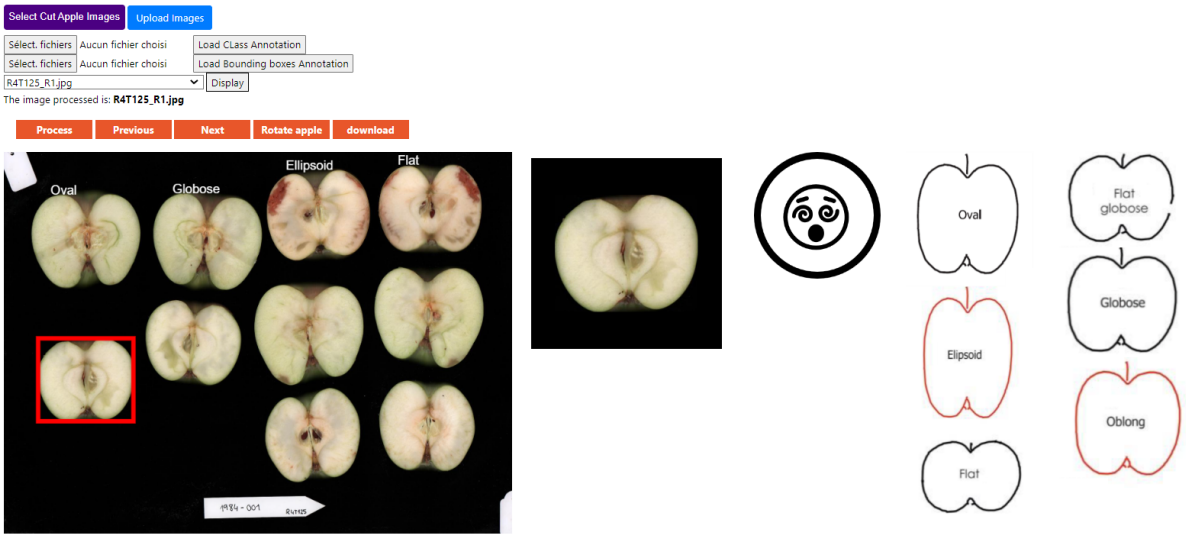


Figure 6.10 – A screenshot from *PanoVar*, where examiners are scoring the shape of cut apples. The sketches represent the scoring scale defined in the UPOV catalog.

Impact

Examiners already tested *PanoVar* in creating the annotated data exploited in chapter 3.2. In addition, the application was also presented to variety testing community in the European project INVITE (https://www.h2020-invite.eu/). The feedbacks were very positive. There was an agreement that the collaboration should continue to adjust *PanoVar* to fit the needs of examiners.

Perspectives

For the near future, there are two perspectives:

- implement training of machine learning models in *PanoVar* and allow examiners to perform both, manual measurements (as it can be done currently) and automatic measurements.
- performs the distinctness test numerically using the machine learning models. Examiners will import the images of the candidates and the reference varieties and run the test in *PanoVar*.

Tutorial

A tutorial video for *PanoVar* was recorded. It is available at this link.

6.3.2 Ordinalysis

Ordinalysis is a software that enables to perform dimension reduction, visualization and quantitative analysis of ordinality. It is provided as a standalone executable file with a video tutorial. Technical details and illustration of Ordinalysis's applications are shown in Annex H.

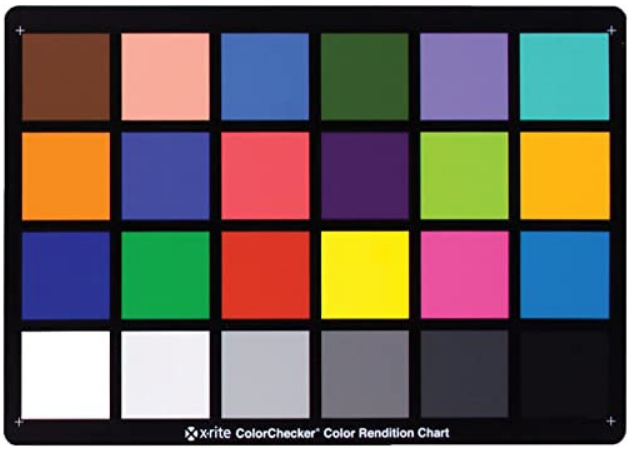
Chapter 7

Annex B: Calibration of 3D Color Point Clouds of Apple Orchard Scenes

7.1 ColorChecker detection from 3D color point clouds

The ColorChecker detection algorithm is devised to estimate the positions of the color patches of the ColorChecker (ColorChecker Passport Photo 2, X-rite, Great Lakes, Midwestern US) (Fig. 7.1a) in a 3D RGB point cloud. The ColorChecker includes a 24-patch color reference target (Fig. 7.1b).

(a) X-Rite's ColorChecker Passport Photo 2



(b) 24-patch color reference target of the ColorChecker

Figure 7.1 – The ColorChecker object

Fig. 7.3 gives the block diagram of the procedure that takes a 3D color point cloud as the input and localizes the ColorChecker, if there is any. A color point cloud PC is a set of 3D points, where each point $p_m \in PC$ is represented by its coordinates (x, y, z) and its

(a) 3D RGB point cloud (b) ColorChecker zoomed

Figure 7.2 – 3D RGB point cloud of an apple orchard scene including a ColorChecker

color (R, G, B). Here, (R, G, B) refers to the values of red, green and blue channels. The correct metric scale of the point cloud is not known beforehand; therefore, we re-scale the coordinates of the points in PC such that the maximum pairwise distance of the points is 100:

$$(x', y', z') = \frac{100}{d_{max}}(x, y, z) \tag{7.1}$$

where, d_{max} is the maximum pairwise distance in the original point cloud:

$$d_{max} = \max_{p_m, p_n \in PC} \|p_m - p_n\| \tag{7.2}$$

The objective of the procedure is to locate the centers of the color patches $c_{i,j}$, $i = 1, ..., 4$, $j = 1, ..., 6$ of the ColorChecker (Fig. 7.4).

The details of our ColorChecker detection procedure are given below:

Step 1: Color thresholding The colors of the points in PC are converted to HSV (Hue, Saturation, Value) representation. Points within the hue range of [0.75 0.87] and saturation range of [0.35 0.55] are extracted. Points with colors in these ranges correspond to

the purple points in the point cloud.

Step 2: Extraction of connected components Connected components of the resulting purple points are considered to be candidate regions for the purple patch of the ColorChecker. After obtaining the connected components, small components with less than 20 points are discarded. Starting from the largest one, Steps 3 to 8 are applied to the connected components until a ColorChecker pattern is detected in Step 8.

Step 3: Plane fitting A plane is fitted to the points in the connected components using M-estimator SAmple Consensus (MSAC) algorithm given in [73], which is a variant of RANdom SAmple Consensus (RANSAC) algorithm. Maximum distance for a point to be an inlier is set to be 0.01. Fig. 7.3 (Step 3) shows the inlier points for a candidate purple patch.

Step 4: Diagonal length estimation The center $\mathbf{c}$ of the the inlier points is calculated by averaging their coordinates. The maximum of the distances between pairwise inlier points is estimated to be the length D of the diagonal of the candidate purple patch. Fig. 7.3 (Step 4) illustrates $\mathbf{c}$ and D for a candidate purple patch.

Step 5: ROI extraction The region of the point cloud within distance R from the center of the candidate purple patch, where $R = 8D$, is considered to be the region of interest (ROI). This region has the potential of including a ColorChecker. Fig. 7.3 (Step 5) shows the points in the region of interest around a candidate purple patch.

Step 6: Plane fitting to the points in ROI MSAC is applied to fit a plane to the points in the region of interest. Fig. 7.3 (Step 6) shows the inlier points of the fitted plane for a candidate ROI.

Step 7: Projection onto an image The inlier points of the plane are projected onto a color image. The size of the image grid is $M \times N$, where $max(M, N) = 200$. Each pixel gets the average color of the 3D points projected onto it. A pixel with no points projected onto it is interpolated with nearest neighbor interpolation. Fig. 7.3 (Step 7) shows a sample projected image.

Step 8: Application of 2D ColorChecker Detector The ColorChecker detection software developed by Hirakawa [159] is utilized to check whether there exists a ColorChecker in the image. This software operates on 2D color images and localizes the 2D center of each color patch. The detected centers for a sample image are depicted with black circles in Fig. 7.3 (Step 8). If no ColorChecker is found, the next connected component from Step 2 is processed. If all connected components are processed without locating a ColorChecker, the procedure is terminated.

Step 9: Back-projection to 3D space The 2D patch centers are back-projected to 3D space to obtain the centers of the color pathces $c_{i,j}$, for $i = 1, ..., 4$ and $j = 1, ..., 6$. In Fig. 7.3 (Step 9) the centers are depicted with black circles.

7.2 Calibration of the 3D point cloud using the ColorChecker

Calibration of a point cloud, in our work, corresponds to the procedure that involves 1) re-scaling the point cloud to the correct metric scale, 2) orienting the point cloud to a canonical reference frame, 3) extraction of region of interest, and 4) re-centering the point cloud to a predetermined position. We apply calibration to both the harvest point cloud PC_h and the winter point cloud PC_w to obtain their calibrated versions PC_h^C and PC_w^C.

The 3D locations of the centers of the ColorChecker patches are used to determine the correct scale of the point cloud and "upward" and "leftward" directions relative to the ColorChecker. The distance between the adjacent centers $c_{i,j}$ and $c_{i,j+1}$ on the same row is s_{ij}. The average of s_{ij} for $i = 1, ..., 4$ and $j = 1, ..., 5$, which is denoted as $\bar{s}$, is used to re-scale the point cloud.

The upward vector v_{UP} corresponds to the direction pointing away from the ground of the apple orchard, and the leftward vector v_{LEFT} is towards the left along the apple tree row when one faces the row. They are estimated as:

$$v_{UP} = \frac{1}{6}\sum_{j=1}^{6}(c_{1j} - c_{4j}); \qquad v_{LEFT} = \frac{1}{4}\sum_{i=1}^{4}(c_{i1} - c_{i6}). \tag{7.3}$$

7.2.1 Re-scaling the point cloud

The actual value of the distances between centers of adjacent patches on the rows of the ColorChecker chart is 15mm. All the coordinates of the point cloud are thus multiplied by $\frac{15}{\bar{s}}$ to get the values of the coordinates in mm units. After this point, all size parameters, such as the height of the ColorChecker tripod, involved in the processing of the point cloud are adjusted in accordance with their true metric values.

7.2.2 Rotating the point cloud to a canonical reference frame

We rotate the point cloud to a canonical reference frame, where Z-axis corresponds to the direction orthogonal to the ground and pointing upwards and Y-axis is parallel to the tree row and oriented towards left (Fig. 7.5).

In order to estimate the axes of the new reference frame, the stick of the tripod of the ColorChecker, and the vectors v_{UP} and v_{LEFT} defined in Eq. 7.3 are used. The stick of the tripod is perpendicular to the ground; therefore the 3D points corresponding to the stick are used for estimation of the Z-axis. The center of the ColorChecker chart is calculated by averaging the detected centers of the color patches. Then, points that are in the distance range [10-65]cm from the chart center in the direction $-v_{UP}$ are gathered (red points in Fig. 7.6). The principal direction of these points is determined through principal component analysis, and is established as the Z-axis of the canonical reference frame. X-axis and Y-axis are then computed through the cross products:

$$X_{axis} = -Z_{axis} \times V_{LEFT}; \qquad Y_{axis} = Z_{axis} \times X_{axis}. \tag{7.4}$$

The origin of the reference frame is temporarily moved to the base of the tripod, which is 1m away from the center of the ColorChecker chart in -Z direction. The temporary reference frame is shown in Fig. 7.7 (a).

Fig. 7.9 (a) and (b) show an uncalibrated point cloud. After re-scaling and rotating to the estimated reference frame, we obtain the point cloud shown in Fig. 7.9 (c), (d), and (e).

7.2.3 Extraction of region of interest

First, points with Z coordinates less than 25cm are discarded from the calibrated point cloud. This way, both ground points and irrelevant objects close to the ground are removed. Fig. 7.9 (f) shows a calibrated point cloud with ground points removed.

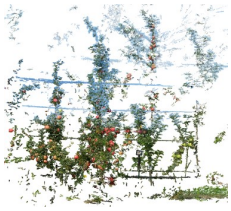
We are interested in processing apple trees in the row located behind the ColorChecker. When the ColorChecker stick was installed, two distances were manually measured with a tape measure: d_R^{cc}: the minimum distance of the tripod stick to the tree row, and d_T^{cc}: the distance to a designated tree. All points with X-coordinates greater than $d_R^{cc} + 1500$m are removed from the point cloud. The remaining points correspond to the target tree row, which is closest to the ColorChecker. Fig. 7.9 (g) shows the region of interest extracted from a calibrated point cloud.

Finally, we remove the ColorChecker object from the scene by removing all the points that satisfy the conditions $(X^2 + Y^2) < R_{CC}^2$ and $Z < H_{CC}$, with $R_{CC} = 20cm$ and $H_{CC} = 120cm$. Recall that the origin of the point cloud is at the base of the ColorChecker tripod at this point.

7.2.4 Translating the origin of the reference frame

The origin of the reference frame is carried from the base of the ColorChecker tripod to the base of the designated tree, as shown in Fig. 7.7. All the points in the region of interest are projected to the ground; i.e. a histogram of points in the XY-plane is created (Fig. 7.8). The XY-plane is converted to a regular grid; and the number of points contained in each grid is calculated. The peaks of the histogram correspond to the tree trunk locations. The distance of the designated tree to the base of the ColorChecker tripod, d_T^{cc}, is known. Since the distance of the ColorChecker to the tree row, d_R^{cc}, was also measured, the location of the designated tree with respect to the ColorChecker is known (Fig. 7.8). A search region of radius 30cm around the location of the designated tree is examined. The maximum peak location of the histogram in this region is marked as the base of the designated tree. The point cloud is translated such that the origin of the reference frame coincides with this location.

The final region of interest of a calibrated point cloud, re-centered at the base of the designated tree is given in Fig. 7.9 (h).



Input Point Cloud

Step 1: Color Thresholding

Extract purple points by applying color thresholding.
Hue range: [0.75 0.87] Saturation range: [0.35 0.55]

Step 2: Connected Components

Apply connected components to the purple points to locate candidate purple patches of the ColorChecker.

Step 3: Plane Fitting

Apply MSAC to the points in the candidate connected component to fit a plane.

Step 4: Diagonal Length Estimation

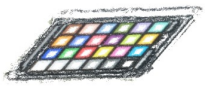
Find the maximum of the pairwise distances (D) among the inliers in the candidate planar purple patch.
Find the center point ($\boldsymbol{c}$) of the inliers.

Step 5: Extract ROI

Extract the Region of Interest (ROI), which is defined as all the points in the point cloud within distance R to the center ($\boldsymbol{c}$) of the candidate purple patch ($R = 8D$).

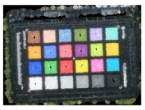
Step 6: Plane Fitting

Apply MSAC to the points in the ROI to fit a plane.



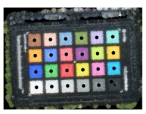
Step 7: Projection onto an image

Project the inliers of the plane to a color image. Apply nearest neighbour interpolation for empty pixels.



Step 8: Apply 2D ColorChecker Detector

Apply "Colorchecker Finder" by Hirakawa [2] to detect the centers of the color patches in the 2D color image.



ColorChecker detected?

YES

NO

Next candidate purple patch exists?

YES

NO

No ColorChecker detected.

Step 9: Back-projection to 3D space

Back-project the detected 2D centers of the patches to the original 3D space.

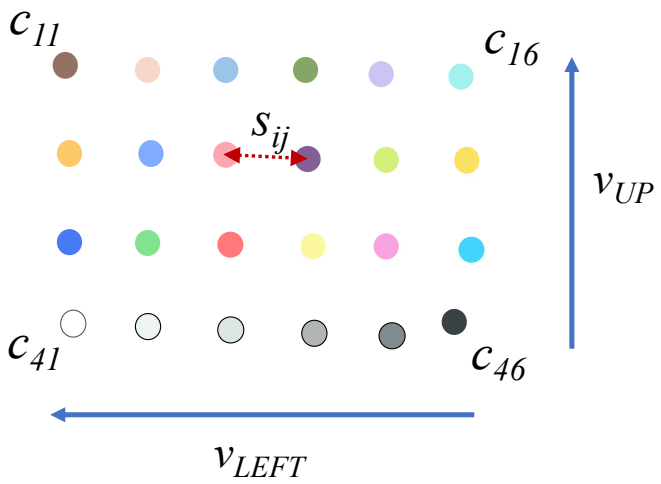


Figure 7.4 – The localized centers of color patches on the ColorChecker are used to estimate the correct scale and the "upward" and "leftward" directions.

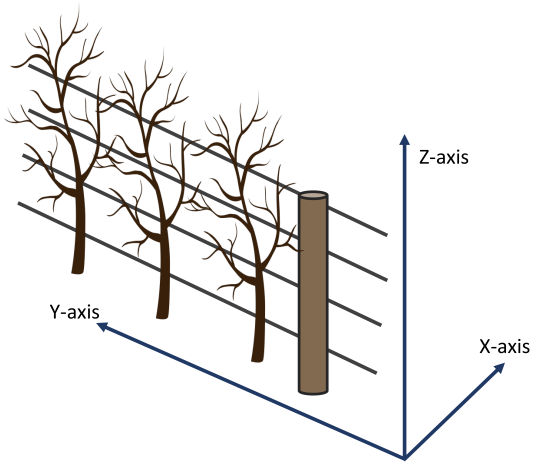


Figure 7.5 – The axes of the canonical reference frame to which the point cloud is rotated. Y-axis is parallel to the tree row and oriented towards left. Z-axis is orthogonal to the ground.

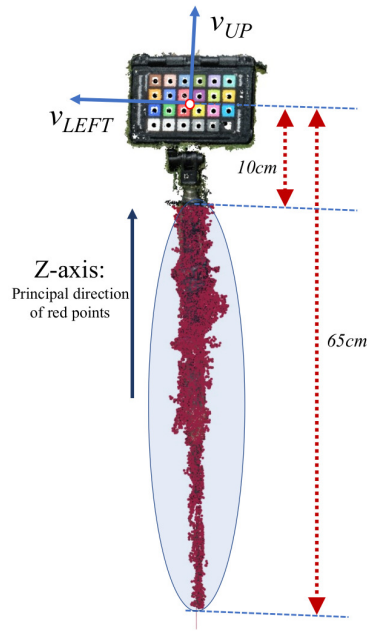
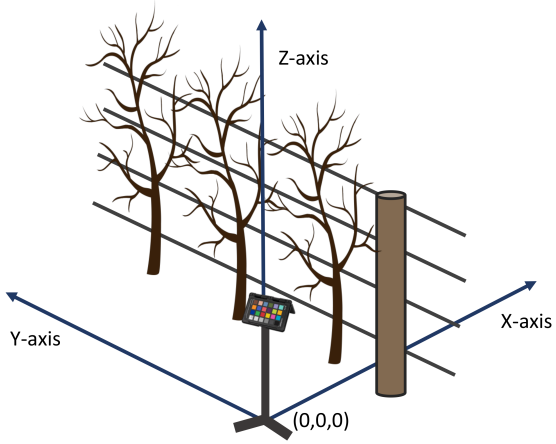
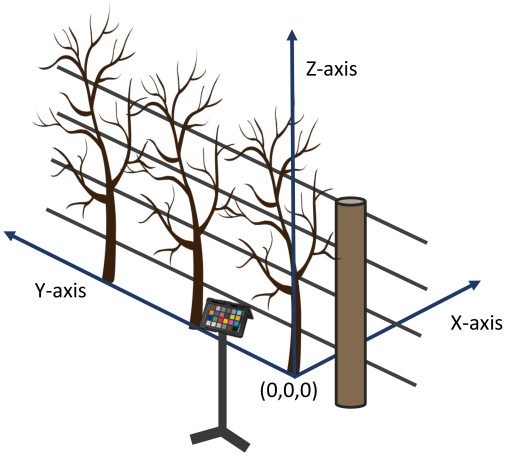


Figure 7.6 – Estimation of the Z-axis of the new reference frame. The points below the ColorChecker chart and on the tripod stick (shown in red) are processed with PCA to extract the principal direction.



(a) Origin at the base of ColorChecker tripod.



(b) Origin at the base of the designated tree

Figure 7.7 – (a) The origin of the reference frame is moved temporarily to the base of the ColorChecker. (b) After the localization of the designated tree trunk, the origin is set at the base of the designated tree.

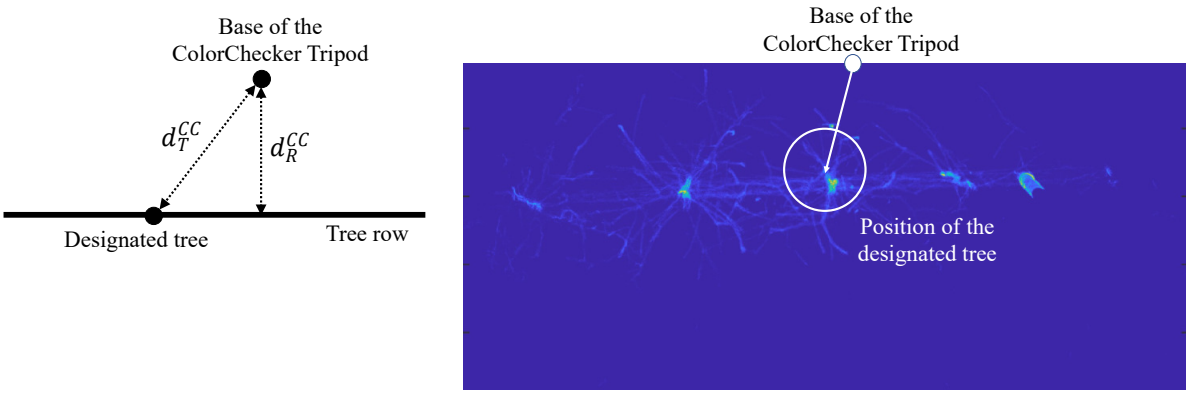
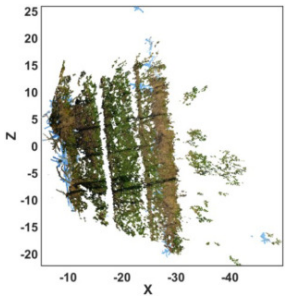
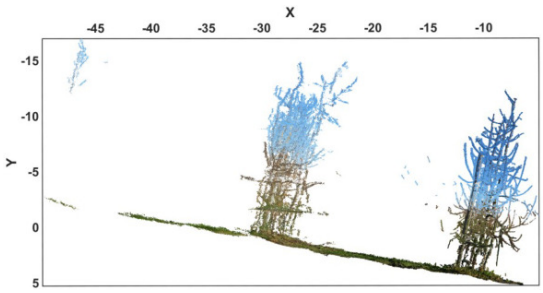


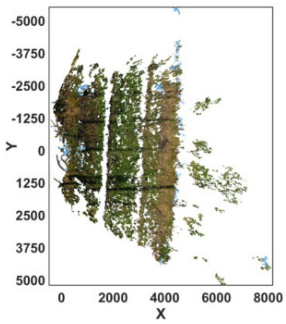
Figure 7.8 – The histogram of the points projected to the XY-plane of the ROI. The peaks correspond to the tree trunks. The local region around the position of the designated tree to the ColorChecker is searched for a peak.



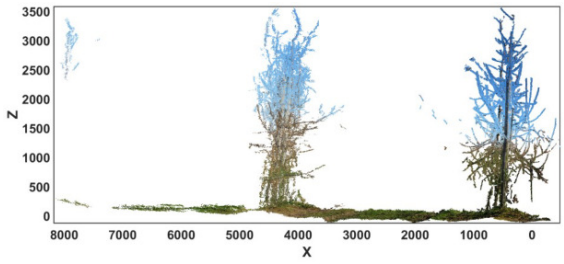
(a) Uncalibrated point cloud seen from the top.



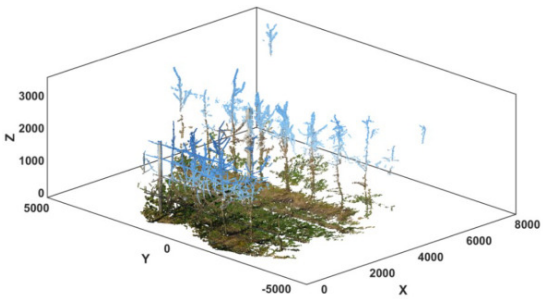
(b) Uncalibrated point cloud seen from the side. The coordinate frame and the scale are arbitrary.



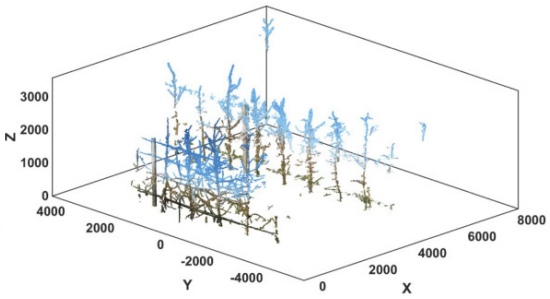
(c) Point cloud after scaling and rotation to the new reference frame, seen from the top. The apple row is parallel to Y-axis.



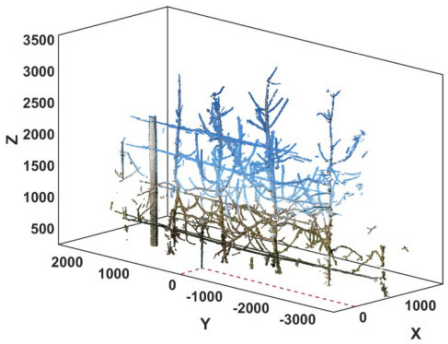
(d) Point cloud after scaling and rotation to the new reference frame, seen from the side. Z-axis is orthogonal to the ground. The scale is measured in millimeters.



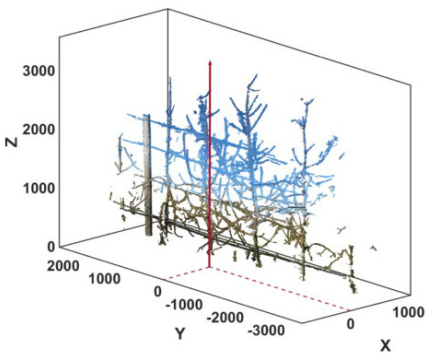
(e) Point cloud after scaling and rotation to the new reference frame. The center of the coordinate frame is temporarily moved to the base of the ColorChecker Tripod.



(f) Point cloud after ground removal. Points with Z coordinates less than 25cm are removed.



(g) Region of interest which is defined as the tree row just behind the ColorChecker.



(h) Final region of interest of the calibrated point cloud. The center of the reference frame is moved to the base of the designated tree.

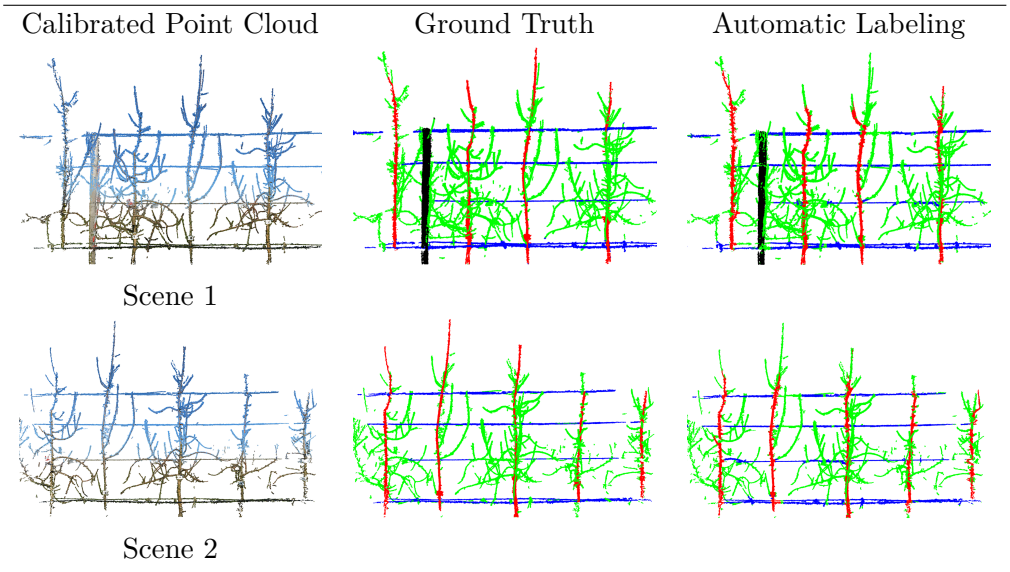
Figure 7.9 – Calibration of the point cloud and extraction of region of interest.

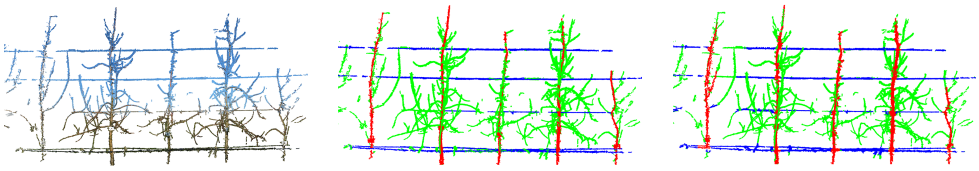
Chapter 8

Annex C: Visual results for detection of trellis wires, tree trunks and support poles

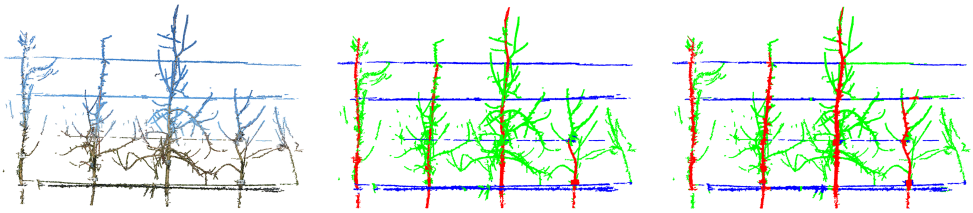
In the chapter 2, we proposed a methodology to separate apple trees in the orchards. The pipeline relies on the detection of the trellis wires, the tree trunks, the support poles and the branches, in the 3D point clouds of apple trees in the winter. To evaluate our detection algorithms, we generated a ground truth associated to the seven 3D point clouds assessed. The Fig. 8.1 shows the calibrated point clouds, the ground truth generated and the automatic labeling using the detection algorithms.

Figure 8.1 – **First column:** Calibrated point clouds, **Second Column:** Manually generated Ground Truth (blue: trellis wires, red: tree trunks, black: support poles, green: branches), **Third Column:** Labels obtained by our semantic segmentation method.

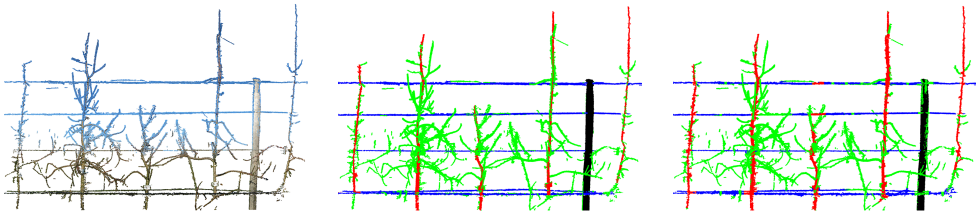




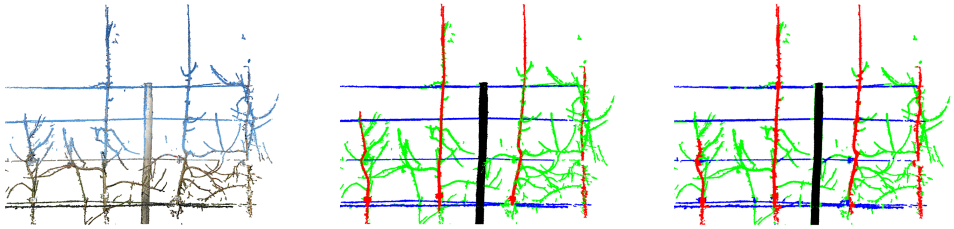
Scene 3



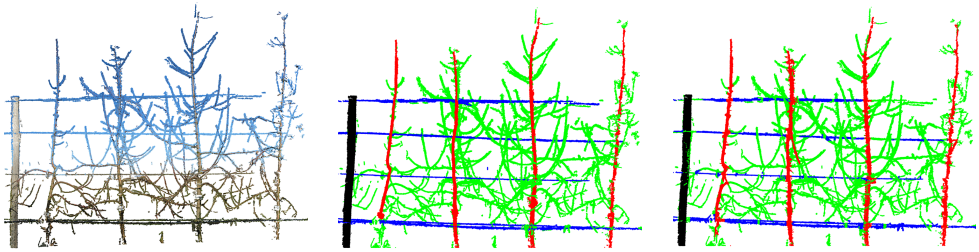
Scene 4



Scene 5



Scene 6



Scene 7

Chapter 9

Annex D: Terminology of machine learning

9.1 Classical machine learning

Machine Learning is an application of artificial intelligence (AI) and computer science. It provides systems with the ability to automatically learn and improve from experience without being explicitly programmed. The improvement of knowledge of machine learning algorithms comes from the use of data. The data are transcoded by feature vectors. The feature space is the reference frame where data are represented. Often this feature space is in $\mathbb{R}^n$ where $n > 3$. Dimension reduction techniques [160] such as PCA, TSNE and LDA, are used to project the feature space from $\mathbb{R}^n$ to $\mathbb{R}^3$ or $\mathbb{R}^2$, to visualize the structure of data in the reduced space. There are several ways of learning the structure of the data in the feature space including supervised learning, unsupervised learning, self-supervised learning and reinforcement learning. In this thesis, we addressed only supervised and unsupervised machine learning.

Supervised machine learning refers to the algorithms where you have input variables x and an output variable Y called labels, and you use an algorithm to learn the mapping function from the input to the label. The goal is to approximate the mapping function so well that you can predict the correct Y for a new input data x. On the other hand, unsupervised machine learning seeks to identify structure among unlabeled data x.

9.2 Deep learning

Although classical machine learning algorithms are robust, they still require human intervention to set the features and there is no guarantee that these features are the most discriminate ones to solve a problem. The classical machine learning algorithms are often not complex and have few parameters to optimize in the training phase. Therefore, they

reach saturation in their performance levels (see Fig. 9.1), despite the increase of the dataset size.

On the other hand, the performance of deep learning, which is a subset of a family of machine learning methods based on artificial neural networks, increases with the addition of data. This is explained by the complexity of deep learning algorithms that have millions of parameters to adjust and therefore require large datasets to be robust. Deep learning architectures consist of end-to-end transformations of the raw data that are chained together top to bottom. Unlike in classical machine learning, the features are selected by the deep learning architectures from the raw data. Currently, in computer vision applications, supervised machine learning algorithms including deep learning are the most robust ones. However, they require image annotation.

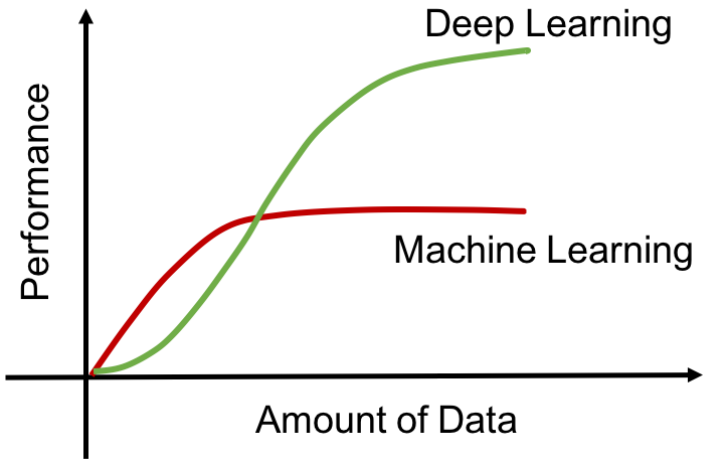


Figure 9.1 – The plateau of performance of machine learning and deep learning in function of the amount of data.

9.3 Image annotation

Machine learning algorithms, when used in supervised ways, require considerable amount of annotated data. Annotation of images is the process of generating a ground-truth Y for each input variable x. There are three different levels of image annotation, each one refers to a computer vision task. Figure 9.2, illustrates these different levels.

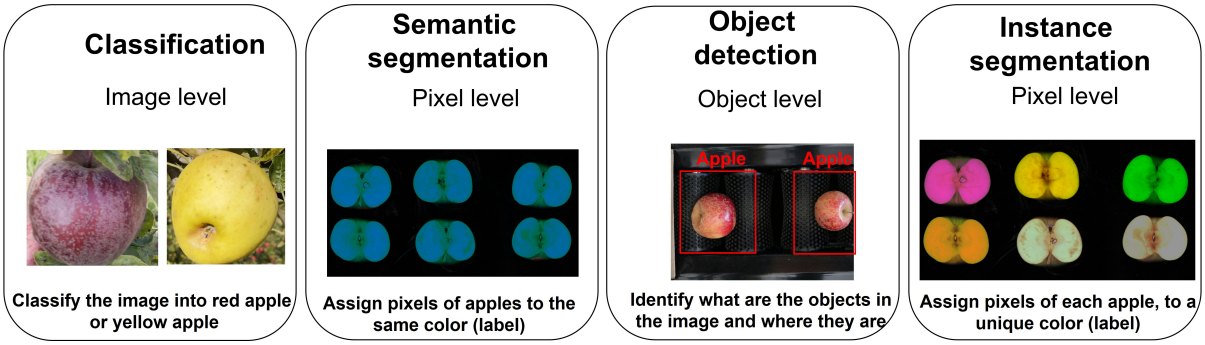


Figure 9.2 – Panel of computer vision tasks and associated level of annotation requested for supervised machine learning.

9.4 Image classification

Image classification (see Fig. 9.2) refers to a predictive modeling problem where a class label is predicted for a given sample image. A model will use the training dataset (images and labels) and will calculate how to best map examples of input data to specific class labels. Some popular examples of classification algorithms are: k-Nearest Neighbors [161], Decision Trees [162], Support Vector Machine [157], Naive Bayes [163] and convolutional neural network (CNN) [158].

9.5 Segmentation

Image segmentation (see Fig. 9.2) is the task of clustering parts of an image together that belong to the same object class. This process is also called pixel-level. There are two types of image segmentation: semantic segmentation and instance segmentation. Semantic segmentation labels each pixel in the input image with a category label. It does not differentiate instances, only cares about categories. It is a pixel-wise classification. On the other hand, instance segmentation goes further and distinguishes instances of the same semantic class. Some common image segmentation techniques in computer vision and machine learning [164] are: thresholding method, edge-based segmentation, region-based segmentation and cluster-based image segmentation and deep learning-based segmentation.

9.5.1 Object detection

Object detection is the task of determining where objects are located in a given image (object localization) and which category each object belongs to (object classification). As shown in Fig. 9.2, we usually use a bounding box to describe the spatial location of an object. The bounding box is a rectangle, which is often determined by the x and y coordinates of the upper-left corner of the rectangle and the such coordinates of the lower-right corner. There are two families of object detection architectures [165]: one-stage methods and two-stage methods.

The one-stage methods such as SSD [166] or YOLO [167] require only a single pass through the neural network and predict all the bounding boxes in one shot. On the other hand, the two-stage methods such as R-CNN [168] extract many region proposals from the input image, use a convolutional neural network to perform forward propagation on each region proposal to extract its features, then use these features to predict the class and bounding box of this region proposal.

Chapter 10

Annex E: Machine Learning-Based Classification of Powdery Mildew Severity on Melon Leaves

Machine Learning-Based Classification of Powdery Mildew Severity on Melon Leaves

Mouad Zine El Abidine[1], Sabine Merdinoglu-Wiedemann[2], Pejman Rasti[1,3], Helin Dutagaci[1], and David Rousseau[1]( )

[1] LARIS, UMR INRAE IRHS, Université d'Angers, 62 Avenue Notre Dame du Lac, 49000 Angers, France
david.rousseau@univ-angers.fr

[2] INRAE-Université de Strasbourg, 21 rue de Herrlisheim, 68000 Colmar, France

[3] Department of Big Data and Data Science, école d'ingénieur informatique et environnement (ESAIP), Saint Barthelemy d'Anjou, France

Abstract. Precision agriculture faces challenges related to plant disease detection. Plant phenotyping assesses the appearance to select the best genotypes that resist to varying environmental conditions via plant variety testing. In this process, official plant variety tests are currently performed in vitro by visual inspection of samples placed in a culture media. In this communication, we demonstrate the potential of a computer vision approach to perform such tests in a much faster and reproducible way. We highlight the benefit of fusing contrasts coming from front and back light. To the best of our knowledge, this is illustrated for the first time on the classification of the severity of the presence of a fungi, powdery mildew, on melon leaves with 95% of accuracy.

Keywords: Machine learning · Classification · Plant disease

1 Introduction

During the last decades, precision agriculture benefited from advances in robotics [1,2], computer vision [3] and artificial intelligence [4,5] to automate the monitoring of crops [6] and harvesting [7]. However, some activities of major importance for agriculture are still to take benefit from these advances. One such activity is plant variety testing. To register and protect a new variety in a country, a plant breeding company has to follow a process managed by a national examination office within an official framework. The national examination offices run tests to register new varieties in the official catalogue, protect them with «plant variety rights» and post control of certified seed lots. Currently, most of these tests are based on manual measurements performed with visual inspection. This is an issue for the sake of efficiency due to the time consuming nature of these tests. In this context, we focus on one of these plant variety tests. We propose an automated algorithm to detect and quantify the presence of powdery mildew

A. El Moataz et al. (Eds.): ICISP 2020, LNCS 12119, pp. 74–81, 2020.
https://doi.org/10.1007/978-3-030-51935-3_8

on melon leaves via in vitro imaging to assess the resistance capability of the tested varieties. Powdery mildew is a fungal disease infecting melon leaves and causing a major reduction of yield. The typical symptoms of powdery mildew are white colonies on the leaf surface consisting of mycelium and spores of the fungal pathogen. We first describe the current manual method and then explain the computer vision procedure that we propose, based on machine learning and fusion of front and back light images. The performance of this automated procedure is compared with the manual method and previous automated methods before conclusion.

2 Related Work

Plant disease detection using deep learning has attracted many attention in the recent past [8–10] due to the large variety of conditions in which diseases can be studied including conditions of plant-pathogen interactions (virus, bacteria, fungi), environmental conditions (field, controlled environment, in vitro, ...), imaging modalities (RGB, thermal, fluorescent, ...) and observation scales (tissue, leaf, canopy, ...). As closely related work, for powdery mildew detection observed on foliar disk in vitro, a spatio-spectral analysis based on hyperspectral images of wine grapes was developed to classify powdery mildew infection levels [11]. An accuracy of 87% was reported to classify "healthy", "infected" and "severely" diseased bunches. In another work, a machine vision based phenotyping system was developed to assess the severity of grapevine powdery mildew [12]. The system is based on a high-resolution camera and a long working distance macro-focusing lens. The system acquires an image of each foliar disk inside a Petri dish and requires an XY motorised stage to move above one foliar disk to another. A GoogleNet neural network architecture was trained on 9920 images of two classes "infected" and "not infected". The training lasted 3.4 h. The resulting CNN had a classification accuracy of 94.3%. By contrast with these methods, our computer vision system requires only a simple RGB camera with standard resolution and a lighting device. This simplicity and low-cost is important for dissemination of the method as the system is dedicated to pathology tests performed by biologists. Also, acquisition time is important as the global objective is to implement a high-throughput phenotyping system. Unlike microscope-based images of [12] that catches only a single foliar disk at a time, we acquire 9 foliar disks in the same snapshot. The previous works have improved disease detection accuracy by investing in the imaging system (hyper-spectral camera and microscope). On the side of optics, we propose to fuse front and back light images to improve classification accuracy and thus implement a high-throughput phenotyping system at a relatively low cost.

3 Current Manual Procedure

The current manual procedure to assess melon leaves resistance to powdery mildew is as follows. First, biologists extract foliar disks from melon leaves (as

illustrated in Fig. 1(a)) and position them inside a Petri dish. A control foliar disk (very sensitive to powdery mildew), positioned at the center of the Petri dish, serves to validate the presence of powdery mildew. In the next step, foliar disks are inoculated with powdery mildew powder inside Petri dishes and left for an incubation period of 10 days. After that, biologists use a binocular loop to visualize leaves and assign an ordinal score according to powdery mildew density on the leaf surface as shown in Fig. 1(b). The encoding of the scoring is provided in Table 1.

Table 1. Annotation scale of powdery mildew propagation on melon leaves.

Score assigned to powdery mildew density	Observation
Resistant	One spore of powdery mildew
Moderate	50% of the leaf is infected
Severe	Leaf is totally infected

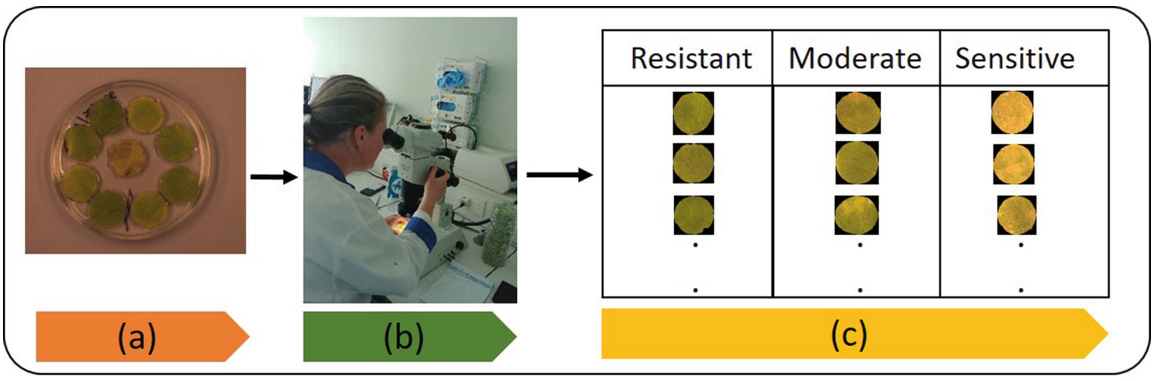


Fig. 1. Schematic visualisation of the current procedure. (a): Petri dish containing melon foliar disks inoculated with powdery mildew. (b): a biologist visualizes and annotates powdery mildew propagation using a binocular loop. (c): data generated after assessment following the annotation scale of Table 1. Each foliar disk is saved in a CSV file with its corresponding class.

4 Proposed Computer Vision Procedure

To automate the manual procedure described in the previous section, we propose to follow the pipeline given in Fig. 2. 70 Petri dish images are acquired with a digital color camera with resolution of 2448 by 2050 pixels. The size of each foliar disk is approximately 120 000 pixels. The camera is positioned vertically above the Petri dish as shown in Fig. 2(a). As illustrated in Fig. 2(b), Petri dish

images are acquired under two lighting techniques, front and back light. After RGB to HSB conversion, the brightness channel of both images (front and back light) are fused in a linear blending to enhance the contrast between the lymb and the fungi.

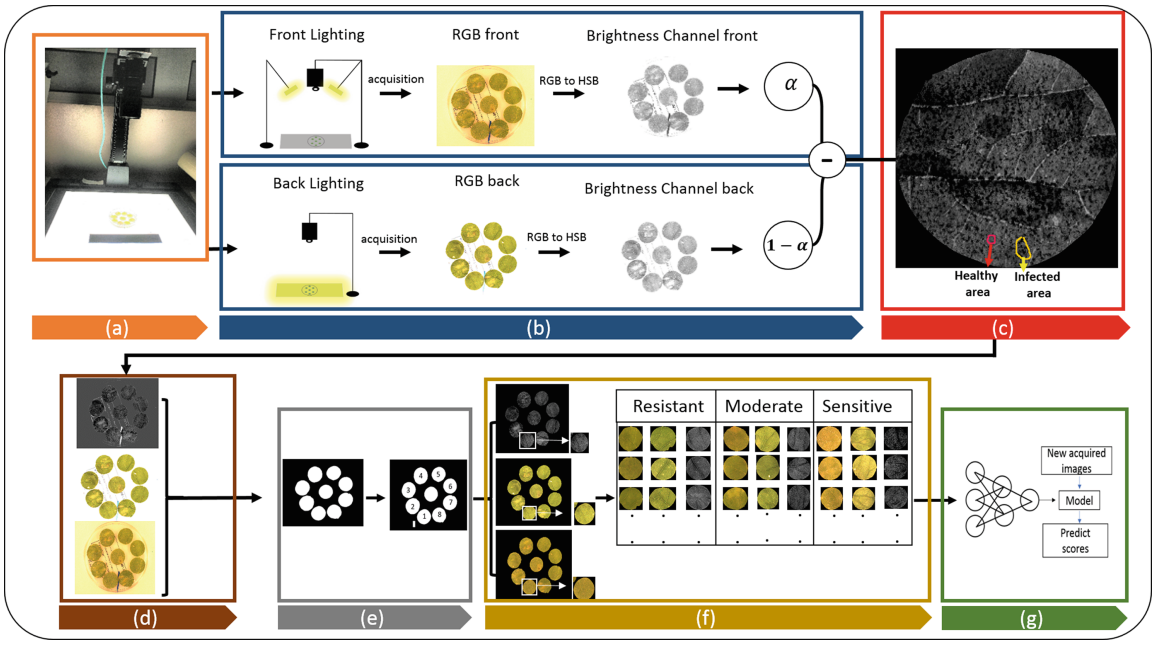


Fig. 2. Visual scheme of the proposed computer vision procedure. (a): imaging device. (b): fusing images following a linear blending. (c): generated fused image. (d): front, back and fused data sets. (e): The foliar disks are segmented from Petri dishes using the convolutional neural network architecture for semantic segmentation UNET [13] and cropped individually. (f): foliar disks are assigned to their corresponding ground truth, produced by an expert in the current manual procedure. (g): feeding training images to a supervised machine learning classification algorithm.

The partition of foliar disk images per class in front, back and fused data sets is as follows: 180 images for "Resistant" class, 62 images for "Moderate" class and 131 images for "Sensitive" class. This data set is rather small in this work (compared to standard large data sets in machine learning). This constitutes a possible limit to the use of an end-to-end deep learning method due to the tendency of overfitting. Instead, a shallow supervised learning scheme based on the concatenation of a deep-feature extraction [14] stage followed by a linear support vector machine classifier was used for the comparison of the performance with different images (fused, front and back light). Deep features trained on ImageNet from well-known architectures were tested in this study including VGG16 and Resnet50. In addition, a small end-to-end CNN model with the architecture shown in Fig. 3 was fine-tuned on a validation data set of 20% of training images. The accuracy of the classification of all tested models was computed per class from the confusion matrix. Due to the lack of enough data and the imbalance classes in our data set, a data augmentation was used

to improve the classification accuracy and to be able to compare fairly with a deep learning architecture. Data were augmented to force invariance to rotation since the leaves are randomly positioned in the Petri dish invariance to shearing and zoom to allow for robustness to some plasticity of the leaf tissue. The mix parameter α in the linear blending to fuse front and back light images was chosen to maximize the contrast between powdery mildew and healthy lymb computed with the Fisher ratio which is defined as

$$FR = \frac{(\mu_2 - \mu_1)^2}{\sigma_2^2 + \sigma_1^2} \tag{1}$$

where μ is the mean pixel value in the selected area and σ is the standard deviation of the pixel values in the selected area. The Fisher ratio was computed on fused images generated by varying α from 0 to 1. The optimal value of α for which Fisher ratio is maximum, equals to 0.1. This value was applied in the linear blending to generate fused images for classification.

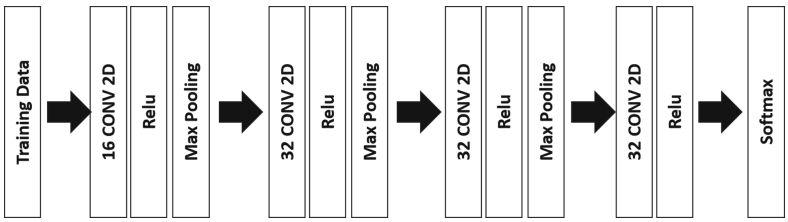


Fig. 3. CNN architecture proposed.

5 Results

Classification performances of the supervised machine learning algorithm described in Sect. 4, are given in Table 2 and Table 3. Best results, highlighted in blue were systematically obtained with the fused images for the three tested classifiers with or without data augmentation. Highest scores are obtained with association of deep features from Resnet50 coupled with a linear SVM with data augmentation. Other classical classifiers such as random forest or non linear SVM were also tested (not shown) but results were not significantly improved. The confusion matrix for this best classifier is illustrated in Fig. 4 which shows that most errors come from the confusion between moderate and sensitive classes. Finally, these two classes are merged by biologists when varieties are registered officially. The classification accuracy achieved in classification of resistant and sensitive levels are provided in Table 3 with best performances culminating at 95% accuracy.

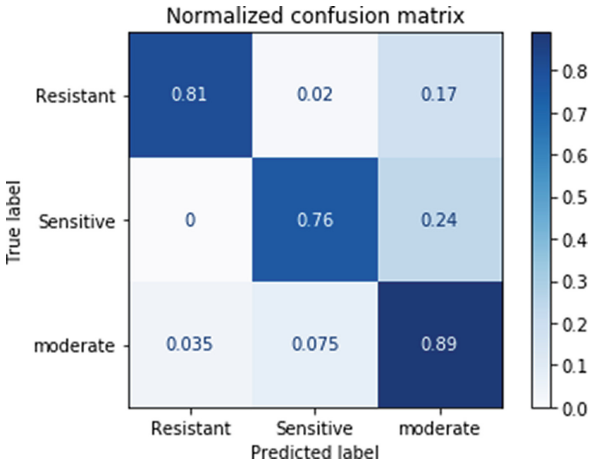


Fig. 4. Confusion Matrix of Resnet50 algorithm to classify powdery mildew on 3 infection levels: resistant & moderate & sensitive.

Table 2. Classification accuracy for 3 infection levels: resistant & moderate & sensitive.

Architecture	Training approach	Front	Back	Fused	Train/test
VGG16	Raw Data	0.61	0.53	0.7	135/88
	Data augmentation	0.69	0.49	0.75	1000/200
Resnet50	Raw Data	0.67	0.59	0.79	135/88
	Data augmentation	0.71	0.62	0.82	1000/200
Proposed CNN	Raw Data	0.56 ± 0.06	0.40 ± 0.04	0.67 ± 0.04	135/88
	Data augmentation	0.64 ± 0.05	0.48 ± 0.07	0.78 ± 0.02	1000/200

Table 3. Classification accuracy for 2 infection levels: resistant & sensitive.

Architecture	Training approach	Front	Back	Fused	Train/test
VGG16	Raw Data	0.82	0.72	0.86	280/100
	Data augmentation	0.81	0.74	0.83	1000/200
Resnet50	Raw Data	0.86	0.81	0.94	280/100
	Data augmentation	0.86	0.84	0.95	1000/200
Proposed CNN	Raw Data	0.71 ± 0.08	0.65 ± 0.02	0.86 ± 0.04	280/100
	Data augmentation	0.79 ± 0.11	0.68 ± 0.01	0.92 ± 0.01	1000/200

6 Discussion

The previous section presented successful results for the classification of the presence of powdery mildew in foliar disks containing melon leaves. The obtained performances are similar to the recently published work on the classification of powdery mildew in two [11] or three classes [12] as presented in the related work section. It is to be noticed that the closest related method of [12] is applied to another crop but in a similar in vitro imaging conditions protocol. While neural networks are also used as the main element of the image processing pipeline, [12] notably differs from our approach. The work of [12] focuses on a metrological measurement of the powdery mildew performed with a high resolution imaging system enabling to detect individual mycelium. By contrast, we propose an ordi-

nal classification of the foliar disk corresponding to the final annotation of an expert. We investigated the possibility of addressing this less demanding task by considering foliar disks as a texture with a much lower spatial resolution. Working at such degraded resolution could constitute a risk of loosing accuracy specially at the low grade of the development of the powdery mildew. However, we demonstrated that this was not the case when considering the final score recorded in variety testing which only keeps two classes (resistant, sensitive). Our method is especially suitable for high-throughput application of variety testing to avoid an overwhelming increase of data while keeping the accuracy of the tests at the current level. The performance of the classical CNN architecture is promising and should exceed the 95% accuracy of Resnet50 in case more training images were provided. A comparison on the same samples of our classification approach with the metrological quantification of [12] would be an interesting perspective.

7 Conclusion and Future Work

In this paper, we presented a computer vision-based approach to automate a plant variety test performed to quantify the severity of powdery mildew infection levels on melon leaves. We demonstrated that fusing front light and back light images improved powdery mildew contrast. This fusion resulted an improvement of 10% accuracy with a very low-cost imaging system. Also, we highlighted the achievement of this performance level with a standard spatial resolution, while the state of the art on this problem reported the use of microscopic resolution to track individual mycellium. The use of deep features Resnet50 coupled with a standard SVM achieved an accuracy of 95%. This automated approach is expected to improve the speed and accuracy of disease detection and could be extended to other in vitro pathology tests. The fusion of front and back light was limited in this communication to a simple linear blending. In the future, we plan to explore various approaches of image fusion [15] to optimize the combination of front and back light images.

Acknowledgements. Authors thank INRAE for funding, V. Grimault, S. Perrot S. Houdault and H. Péteul from GEVES (French authority in variety testing) for acquisition trials.

References

1. Billingsley, J., Visala, A., Dunn, M.: Robotics in agriculture and forestry. In: Siciliano, B., Khatib, O. (eds.) Springer Handbook of Robotics, pp. 1065–1077. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-30301-5_47
2. Emmi, L., Gonzalez-de-Soto, M., Pajares, G., Gonzalez-de-Santos, P.: New trends in robotics for agriculture: integration and assessment of a real fleet of robots. Sci. World J. **2014**, 21 (2014)
3. Rousseau, D., et al.: Multiscale imaging of plants: current approaches and challenges. Plant Methods **11**(1), 6 (2015)

4. Kamilaris, A., Prenafeta-Boldú, F.X.: Deep learning in agriculture: a survey. Comput. Electron. Agric. **147**, 70–90 (2018)
5. Patrício, D.I., Rieder, R.: Computer vision and artificial intelligence in precision agriculture for grain crops: a systematic review. Comput. Electron. Agric. **153**, 69–81 (2018)
6. Negash, L., Kim, H.-Y., Choi, H.-L.: Emerging UAV applications in agriculture. In: 2019 7th International Conference on Robot Intelligence Technology and Applications (RiTA), pp. 254–257. IEEE (2019)
7. Zhuang, J., et al.: Computer vision-based localisation of picking points for automatic litchi harvesting applications towards natural scenarios. Biosyst. Eng. **187**, 1–20 (2019)
8. Mohanty, S.P., Hughes, D.P., Salathé, M.: Using deep learning for image-based plant disease detection. Front. Plant Sci. **7**, 1419 (2016)
9. Ferentinos, K.P.: Deep learning models for plant disease detection and diagnosis. Comput. Electron. Agric. **145**, 311–318 (2018)
10. Singh, A.K., Ganapathysubramanian, B., Sarkar, S., Singh, A.: Deep learning for plant stress phenotyping: trends and future perspectives. Trends Plant Sci. **23**(10), 883–898 (2018)
11. Knauer, U., Matros, A., Petrovic, T., Zanker, T., Scott, E.S., Seiffert, U.: Improved classification accuracy of powdery mildew infection levels of wine grapes by spatial-spectral analysis of hyperspectral images. Plant Methods **13**(1), 47 (2017). https://doi.org/10.1186/s13007-017-0198-y
12. Bierman, A., et al.: A high-throughput phenotyping system using machine vision to quantify severity of grapevine powdery mildew. Plant Phenom. **2019**, 9209727 (2019)
13. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
14. Vapnik, V.: The Nature of Statistical Learning Theory. Springer, New York (2013)
15. Sun, S.: A survey of multi-view machine learning. Neural Comput. Appl. **23**(7–8), 2031–2038 (2013). https://doi.org/10.1007/s00521-013-1362-6

Chapter 11

Annex F: Transfer learning from indoor to outdoor in Variety Testing: application to Russeting detection

Transfer learning from indoor to outdoor in Variety Testing: application to Russeting detection

Mouad Zine El Abidine[1], Helin Dutagaci[1,2], David Rousseau[1]

1: Université d'Angers, LARIS, UMR INRAe IRHS,
62 Avenue Notre Dame du Lac, 49000 Angers, France.
2. Eskisehir Osmangazi University, Department of Electrical-Electronics Engineering, Eskisehir, Turkey
Corresponding author: david.rousseau@univ-angers.fr

Abstract:

Post-harvest measurements in digital horticulture are nowadays well-calibrated. They are done indoors, i.e., in a controlled environment, with chosen lighting systems and optimized positioning of the sensors to probe the harvested items. In outdoor conditions, the deployment of digital horticulture with sensors and cameras remains challenging due to uneven illumination of the scenes or the spatial 3D complexity of the plants. In this contribution, we consider the important problem of russeting detection on apples. This is an important trait that impacts the quality of the fruit for the consumer and is commonly measured in variety testing trials. We investigate the possibility of using indoor post-harvest russeting data (RGB images of individual apples) to help the detection of russeting on apple images in orchards. This is obtained with a shallow learning approach. First, a supervised model quantifies the amount of russeting in indoor data based on the low-cost apple sorting machine described in [8]. The indoor data set is then merged with the outdoor data set in the training phase to enhance the presence of russeting texture information through deep features. The outdoor data set includes images with several apples encompassed in a single image. The apples are first detected with a standard object detection algorithm (YOLOv4 Tiny). Inference on these detected objects is then boosted with the use of the indoor dataset.

Keywords: Transfer learning, Supervised classification, Russeting, Horticulture.

Introduction:

Computer vision is currently extensively applied in digital horticulture [1]. When coupled with machine learning, it enables high performance for a large variety of informational tasks such as classification, object detection or segmentation of wide interest to automate agronomic measures. These annotations are still widely used in the supervised mode, where the machine is trained on labeled examples. Consequently, the bottleneck in the design of computer vision solutions shifts from features extraction to data annotation. There are different ways to reduce this need for data annotation, including self-supervised methods [2], training on synthetic data sets (either produced by data augmentation [3], virtual environment [4] or generative models [5]) and in transfer learning [6]. In transfer learning, a model is trained on data similar to the one of final interest and then just fine-tuned with a smaller amount of target data. In digital

horticulture, computer vision is used in two main environments: indoor conditions for post-harvest measurement and outdoor conditions for pre-harvest measurement. Indoor conditions are obviously easier because they are obtained with controlled lighting and with standardized field of views. Also, post-harvest measurement can usually be performed with fewer constraints in terms of time since the harvest products can be stored. By contrast, outdoor data is much more challenging due to uneven illumination, variability of field of view from one field to another and constraints in terms of acquisition timing to be synchronized with the phenological stages of the horticultural crops. A natural question is therefore to address the possible use of indoor data for transfer learning to boost machine learning applied on outdoor data.

We propose to investigate this generic question to transfer learning from indoor data to outdoor data in a specific variety testing experiment. We consider the detection of rEuusseting in apples. Russeting is a phenomenon that affects apples and pears, causing slightly harder patches of brown on the skin of the fruit. It doesn't harm the fruit. It is an expected feature, but it's not always welcome, and it can even decrease the economic value of the variety.

Data acquisition

In this section, we describe steps (a1) and (a2) of the pipeline in Fig. 1. Overall, 50 images of 20 varieties of apple trees, from the so-called Refpop population [7], were acquired during the harvest period, using a color camera with a resolution of 2000×4000 pixels. After the harvest period, another 20 varieties different from the one acquired outdoor, each represented by 30 apples (as shown in figure 2), were acquired using a conveyor machine developed in [8]. The machine allows moving the fruits in translation while carrying out a rotation (see Fig. 3). A camera, positioned in top view from the conveyor belt of the machine, took pictures of the apples in rotation, which allowed multiple images with an entire coverage of each apple. Approximately 9 to 10 views of the same apple were captured thanks to this rotation-translation process. These multiple views are important since russeting may appear at any location, which explains why experts have to manually rotate the apples to have an exact assessment. The machine presented in Fig. 3 can acquire a set of 30 apples in a couple of minutes. Images were acquired in burst mode with a Canon camera (10.1 megapixels resolution) controlled by a simple Raspberry-pi minicomputer. Apples were segmented automatically from the background, as visible in Fig. 3. Overall, 23000 segmented apple images were acquired. Figure 1 (a1) shows an example of acquired images.

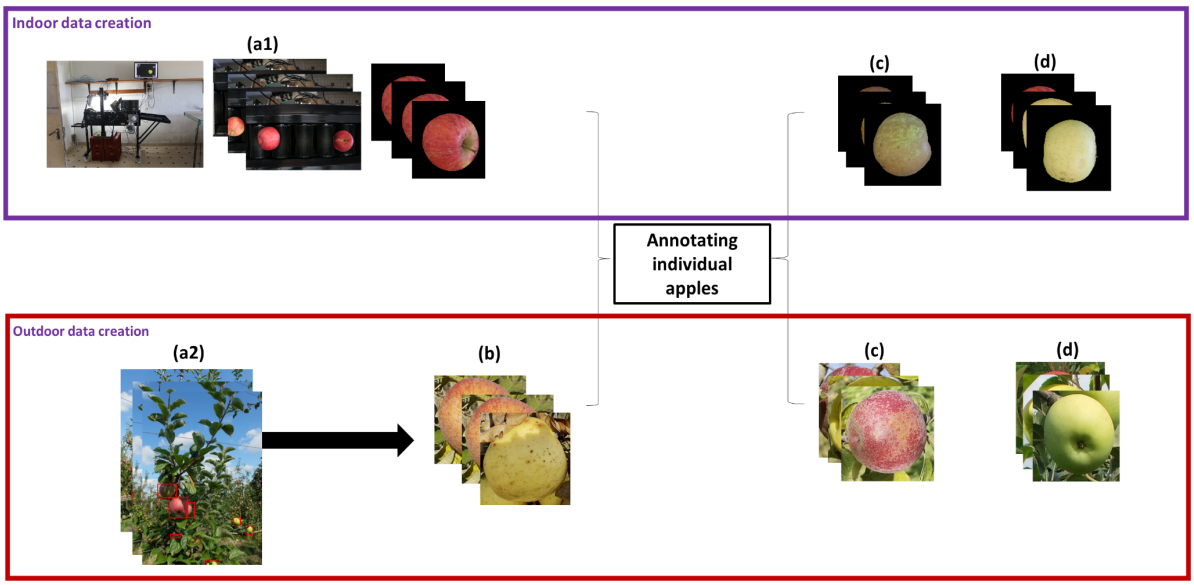


Figure 1: computer vision pipeline followed to create annotated data. (a1): indoor acquisition. (a2): outdoor acquisition. (b): cropping apples from indoor and outdoor data. (c): annotated apples in the class unhealthy (containing Russeting). (d): annotated apples in the class healthy.

Figure 2: Example of input indoor data. 30 representative apples of a variety, stacked in a box..

Figure 3: Acquisition system. Upper panel: Machine equipped with a conveyor belt, used for the acquisition of images of apples with a high surface coverage. Lower panel: view of the acquired images of apples after segmentation from the background.

Preprocessing

After the acquisition of outdoor data, apples need to be cropped from apple tree images to be annotated and fed to the classification model. To save time, we exploited an already existing annotated dataset and trained a YOLO architecture to detect apples on outdoor acquired images of apple trees. YOLO (You Only Look Once) is a state-of-the-art, real-time object detection architecture [9]. We opt for a YOLO model because it requires only a small dataset to achieve good results. YOLO reframes object detection as a single regression problem, straight from image pixels to bounding box coordinates and class probabilities. A single convolutional network simultaneously predicts multiple bounding boxes and class probabilities for those boxes [9]. We trained Yolov4 Tiny using a batch size of 8, one subdivision and 6000 epochs. Figure 4 shows the learning curve on the validation data. The model achieved a mean-average precision of 68%. The detection result was sufficient to crop a few apples from apple tree images, therefore, we didn't test other approaches to improve the accuracy. Figure 5 shows an illustrative example of the predicted bounding boxes on an apple tree image from the test set. Using the bounding boxes coordinates, we cropped the apples, as visible in figure 1(b).

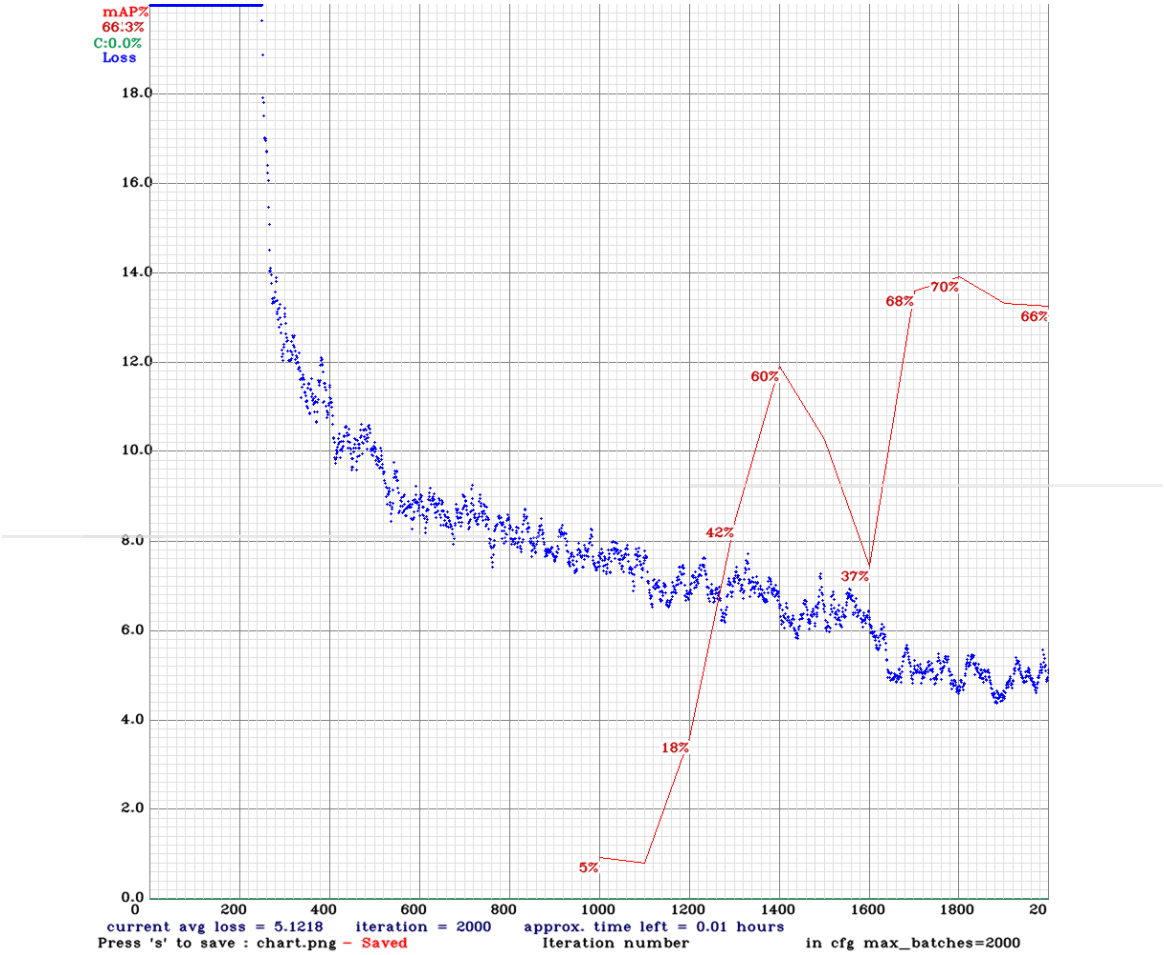


Figure 4: Learning curve of performance as a function of the number of iteration of the model for apple detection in apple trees. The blue line stands for the loss function, while the red line stands for the metric: mean-average precision.

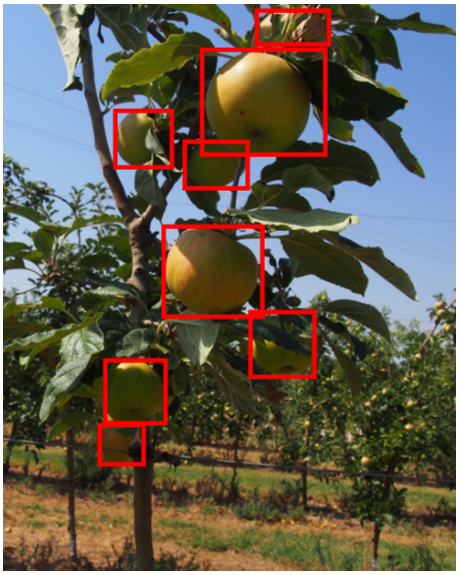


Figure 5: Example of predicted bounding boxes around apples on an apple tree image from the test set.

Data annotation

From the original dataset cropped, 700 apple images from indoor data and 550 apple images from outdoor data were selected to be annotated to healthy and unhealthy (containing Russeting) classes. In the literature, several annotation tools exist to perform object detection annotation. For the sake of simplicity, we targeted online tools such as Labelbox LabelMe, Cytomine, Dataturks, VGG image annotator. The cited annotation tools are well-known for their interactivity with annotators. However, they do not incorporate one important feature: the ability of individual volunteers, to participate in the annotation campaign. Sharing the project with internal collaborators can be a heavy and labor-intensive task. Our objective is to invite people (experts and none-experts) to perform the task required in the project. This strategy of annotation motivated us to opt for the platform "Zooniverse".

As stated in their website: Zooniverse is the world's largest and most popular platform for people-powered research. This research is made possible by volunteers — more than a million people around the world who come together to assist professional researchers...At the Zooniverse, anyone can be a researcher You don't need any specialized background, training, or expertise to participate in any Zooniverse projects. We make it easy for anyone to contribute to real academic research, on their own computer, at their own convenience....Our projects combine contributions from many individual volunteers, relying on a version of the 'wisdom of crowds' to produce reliable and accurate data. By having many people look at the data we often can also estimate how likely we are to make an error.

For both dataset categories, we build, separately, an annotation project in Zooniverse and set instructions in the "About" and "FAQ" sections, to guide annotators (volunteers) to perform the task correctly; i.e. classify an apple as healthy or unhealthy (containing Russeting). Table 1 presents the distribution of annotated data in healthy and unhealthy classes.

Data categories	Healthy	Unhealthy (Russeting)
Indoor Data	**350**	**350**
Outdoor Data	**250**	**250**

Table 1: Distribution of images in each class for indoor and outdoor data.

Methodology: transfer learning from indoor to outdoor data in supervised binary classification

To show the added value of indoor data on the improvement of classification accuracy of the outdoor data, we opt for a classical machine learning model. The data set presented in Table 1 is rather small compared to standard large data sets in machine learning. This constitutes a possible limit to the use of an end-to-end deep learning method due to the tendency of overfitting. Instead, a shallow supervised learning scheme based on the concatenation of a deep-feature extraction [10] stage followed by a linear support vector machine classifier was used. Deep features trained on ImageNet from well-known architectures were tested in this study, including VGG16 and Resnet50. Three experiments were conducted . In the first one, a model was trained and tested on indoor data. We expect that the plateau of performance will be the reference, as the images were taken in a controlled environment. The second experiment concerns training the model on outdoor data. In the third experiment, we added all indoor data to outdoor data in the training set, and we tested on only outdoor data. We investigate if the added set of indoor data helps to improve the classification accuracy obtained in experiment 2. To quantify the sensitivity to the choice of the data chosen for the training, the experiment was conducted for various values of the train-test split of the data set and a 10-fold cross-validation. The average value and standard deviation of the performances of classification were recorded.

Results and discussion :

The plateau of performance of each experiment is shown in Figure 6. As expected, the model, trained on indoor data and tested on indoor data, provides the best performance among experiments. The uniformity of the background and the normalized acquisition conditions guarantee almost identical features for the same class of data, ensuring a correct classification accuracy.
Regarding the results of experiments 2 and 3, we observe that for small training sets (up to 0.45 of the training set), the model trained on outdoor data performed better than the model trained on outdoor and indoor data. This behavior happens because the background of outdoor data is moderately present in the training set. While the region of interest is common between indoor and outdoor images, the properties of the images are not the same. Illumination, cluttered apples and shadow, are factors that should be learned in the training set to separate the background from the foreground in test images and recognize the region of interest, i.e, Russeting. With few instances of the outdoor data in the training set, the model cannot correctly perform the classification task. By contrast, when the training set has a balanced combination of indoor and outdoor data (starting from 0.45 of the training set), the model can learn the context of the outdoor data background while adding more information about the region of interest from deep features of indoor data. Figure 7 shows examples of correct and wrong classification results in experiment 3. We noticed from a visual inspection that high illumination (the reflection of the sun) or shadow could lead the model to fail in some cases. Such limiting factors could be overcome by acquiring more apples in these specific illumination conditions or performing data augmentation by randomly enhancing illumination and adding shadow to existing images, as demonstrated in [11].

The gain of performance found with fusion of indoor and outdoor data is significant and interesting if one considers that indoor data are much more easy to produce. In the example chosen for illustration this gain of performance was however somehow limited. One reason is that the performances with pure outdoor data and pure indoor data were already rather high. As a consequence the maximum gain to be expected in the fusion was limited. The level of complexity of the informational tasks controls the expected gain by transfer learning.

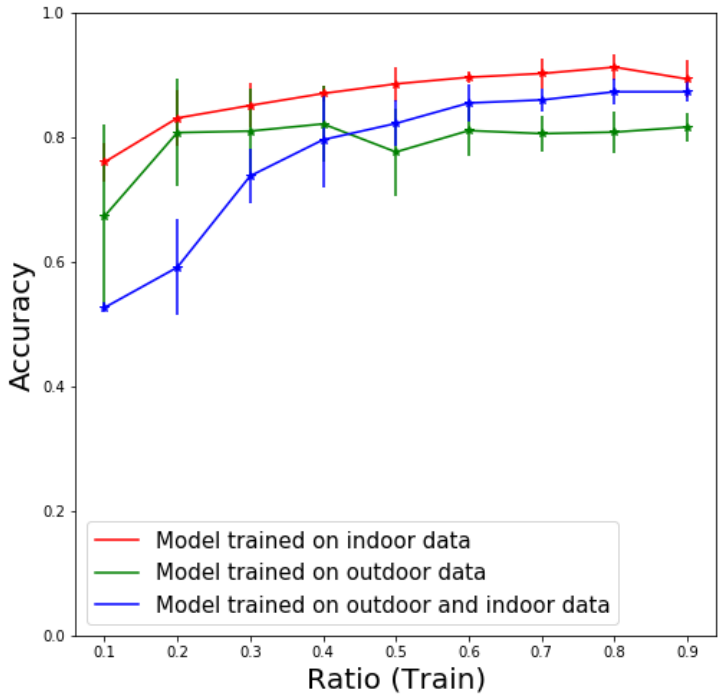


Figure 6: Evolution of the performance as a function of the Ratio (train/test) of the dataset.

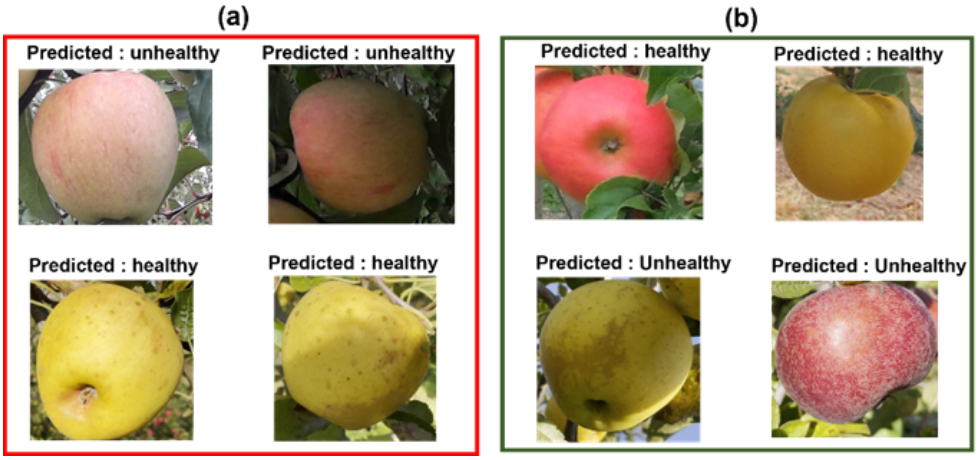


Figure 7: Examples of (a): wrong classification results and (b): correct classification results.

Conclusion

This work investigated the possible value of indoor data from a controlled environment to outdoor conditions in computer vision-based horticultural phenotyping. Using a shallow learning algorithm, we illustrated transfer learning on an example of apple russeting phenotyping. A gain of +5 % in detection performance was shown. This is encouraging. In our case the model based with only outdoor data was already found with rather large performances. This explains the rather limited gain obtained. It would be interesting to revisit the study presented here with more difficult tasks.

References

[1] Yang, Biyun, and Yong Xu. "Applications of deep-learning approaches in horticultural research: A review." *Horticulture Research* 8 (2021).

[2] Jaiswal, Ashish, et al. "A survey on contrastive self-supervised learning." *Technologies* 9.1 (2020): 2.

[3] Montserrat, Daniel Mas, et al. "Training object detection and recognition CNN models using data augmentation." *Electronic Imaging* 2017.10 (2017): 27-36.

[4] Nikolenko, Sergey I. "Synthetic Simulated Environments." *Synthetic Data for Deep Learning*. Springer, Cham, 2021. 195-215.

[5] Tanaka, Fabio Henrique Kiyoiti dos Santos, and Claus Aranha. "Data augmentation using GANs." arXiv preprint arXiv:1904.09135 (2019).

[6] Weiss, Karl, Taghi M. Khoshgoftaar, and DingDing Wang. "A survey of transfer learning." *Journal of Big data* 3.1 (2016): 1-40.

[7] M. Jung, M. Roth, M. J. Aranzana, A. Auwerkerken, M. Bink, C. Denance´, C. Dujak, C.-E. Durel, C. F. i Forcada, C. M. Cantin et al., "The apple refpop—a reference population for genomics-assisted breeding in apple," Horticulture research, vol. 7, no. 1, pp. 1–16, 2020.

[8] G. Couasnet, M. Z. El Abidine, F. Laurens, H. Dutagaci, and D. Rousseau, "Machine learning meets distinctness in variety test- ing," in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 1303–1311.

[9] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 779–788.

[10]V. Vapnik, The nature of statistical learning theory. Springer science & business media, 2013.

[11] Garbouge, Hadhami, Pejman Rasti, and David Rousseau. "Deep learning-based detection of seedling development from indoor to outdoor." *International Conference on Systems, Signals and Image Processing*. Springer, Cham, 2021.

Chapter 12

Annex G: Automatic apple detection in orchards with computer vision and machine learning

Automatic apple detection in orchards with computer vision and machine learning

Mouad Zine El Abidine[1], Ali Ahmad[1], Helin Dutagaci[1,2], David Rousseau[1]

1: Université d'Angers, LARIS, UMR INRAe IRHS,
62 Avenue Notre Dame du Lac, 49000 Angers, France.
2. Eskisehir Osmangazi University, Department of Electrical-Electronics Engineering, Eskisehir, Turkey
Corresponding author: david.rousseau@univ-angers.fr

Abstract:

Computer vision and artificial intelligence promise to revolutionize horticulture with automation and more objective measurements of traits of agronomical importance. While many algorithms are already published in this domain, the transferability of this literature to the techno providers or the final users is often limited due to the absence of sharing of the software or of the data set used to train these softwares. Some major initiatives to address the lack of reproducibility, in the plant domain at large, have so far been limited to plant models (Arabidopsis) or major crops (Wheat). Horticulture is therefore waiting for more similar initiatives. In this work, we present an annotated data set on apple detection that we make publicly available. This data set produced in collaboration with a group of European variety testing offices on several sites is associated with a baseline algorithm (deep learning) already providing reasonable results. The challenge carried out by these data set is typical of an orchard environment with a background creating a major clutter with the targeted foreground. The test of new algorithms is made accessible via the deployment of a data challenge related to this data set. We will present the result of the data challenge which is open during the academic year 2021-2022 to master students in data science.

Keywords: Deep learning, Data Challenge, Horticulture, Fruit Detection

Introduction:

Apple is an important horticultural crop and its production has a clear impact on the economy. Breeders aim to produce apple varieties with an appealing texture, taste and color. Among the traits to quantify an apple variety: the counting of fruits. Classical apple counting is conducted by sampling a fixed percentage (e.g. 5 or 10%) of trees randomly or systematically and extrapolating the counts on these trees for total yield estimation of the entire orchard [1]. This sampling and extrapolation process, in addition to being time-consuming and labor-intensive, does not always produce the desired precision of yield estimation. Recently, deep learning methods have become commonplace for fruit detection and counting [2]–[15]. Deep neural networks are employed to learn predictors from a set of training data by optimizing the parameters of feature extraction and localization of fruits simultaneously. After prediction, further processing, such as circular Hough transform and watershed transform [16] for verification and filtering of multiple counts through 3D (3-Dimensional) reconstruction [7], [17], [18] can be applied to extract the final fruit count. Although the cited works developed robust models for apple detection, the lack of open access code lines and sharing of annotated data, motivated us to organize a data challenge where research teams in digital phenotyping are welcome to test their models on our annotated Dataset. It should be mentioned that [19] has already published a benchmark annotated dataset

for apple detection and segmentation. The acquisition distance was the same, therefore there is no variation in apple scales. In this work, we present an annotated data set on apple detection that we make publicly available. We make accessible the test of algorithms to process these images through a data challenge. We provide a baseline solution and present the associated results. To encourage the community of precision horticulture to engage more in such practices, which are common in computer vision, we describe the creation of the data challenge itself. The result of the data challenge will be announced during the IHC2020 event.

Data acquisition

The data used in this communication was initially acquired for our article [20]. As image acquisition procedure, apple trees from the so-called Refpop population [21], were acquired using a DSLR camera with a resolution of 2000×4000. As described in Figure1, the acquisition was conducted from multi-views, to get visual information covering the apple tree from top to bottom and from various sides of the trees to guarantee that all fruits appear in the frames. In addition, we include in the dataset a subset from the benchmark Dataset for Apple Detection and Segmentation [19]. Table 1 presents the repartition of acquired images according to the distance between apple trees and the camera.

Acquisition categories	Number of images
Close-view images	**1738**
Front-view images	**3300**
Far view images	**712**

Table 1: Number of images acquired for each acquisition category. The category is de ned by the distance of the apple trees to the camera.

Figure 1: Multi-view image acquisition protocol. Apple tree is acquired from all angles to capture all fruits.

Data annotation

From the original dataset in table 1, 678 images were selected for annotation. In the literature, several annotation tools exist to perform object detection annotation. For the sake of simplicity, we targeted online tools such as Labelbox LabelMe, Cytomine, Dataturks, VGG image annotator. The cited annotation tools are well-known for their interactivity with annotators. However, they do not incorporate one important feature: the ability of individual volunteers, to

participate in the annotation campaign. Sharing the project with internal collaborators can be a heavy and labor-intensive task. Our objective is to invite people (experts and non-experts) to perform the task required in the project. This strategy of annotation motivated us to opt for the platform *"Zooniverse"*.

As stated in their website: *Zooniverse* is the world's largest and most popular platform for people-powered research. This research is made possible by volunteers — more than a million people around the world who come together to assist professional researchers...At the *Zooniverse*, anyone can be a researcher You don't need any specialized background, training, or expertise to participate in any *Zooniverse* projects. We make it easy for anyone to contribute to real academic research, on their own computer, at their own convenience....Our projects combine contributions from many individual volunteers, relying on a version of the 'wisdom of crowds' to produce reliable and accurate data. By having many people look at the data we often can also estimate how likely we are to make an error. We build our annotation project in *Zooniverse* and set instructions in the "about" and "FAQ" sections, to guide annotators (volunteers) to perform the task correctly; i.e. draw a rectangle around each visible apple "in the foreground of images". Figure 2 shows the distribution of annotated data in training, validation, and test sets.

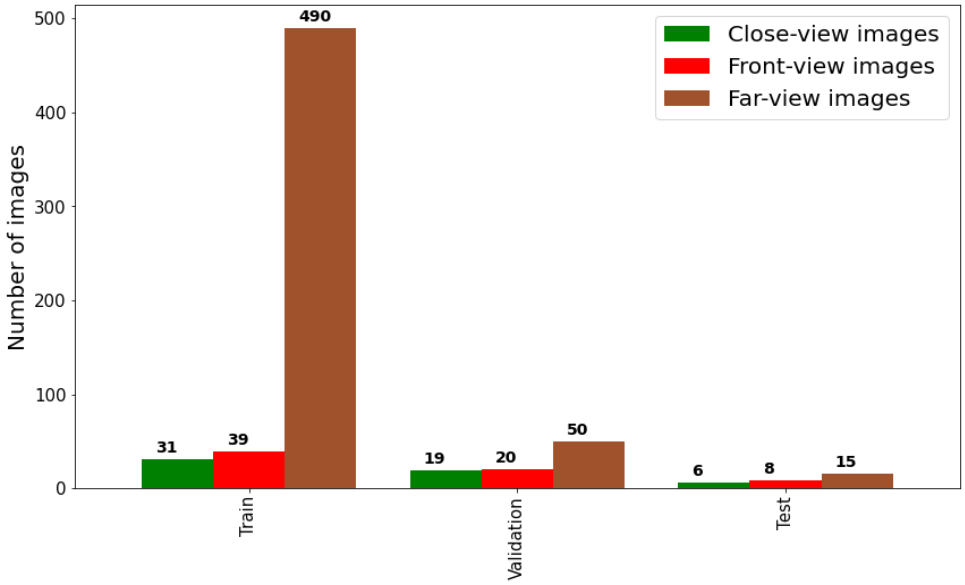


Figure 2: Distribution of multi-view images in training, validation and test data.

Data challenge creation

Machine learning competitions offer one, very effective, platform for the practical application of data science techniques. Machine learning competition platforms allow users to find and publish data sets, explore and build models in a web-based data-science environment, work with other data scientists and machine learning engineers, and enter competitions to solve data science challenges. There are several websites that host machine learning competitions. We cite, for example, DrivenData, Innocentive, Kaggle, Codalab, Topcoder, AIcrowd, etc. The majority of these platforms have a similar format. This typically includes the availability of a wide variety of com-

petitions ranging in skill level, and both no prize practice problems and competitions with financial rewards attached. In the following, we will focus on hosting a data challenge on *Codalab* platform. It is an open-source web-based platform that enables researchers, developers and data scientists to collaborate, with the goal of advancing research fields where machine learning and advanced computation is used. *CodaLab* is centered on the idea of bundles, which can store data. You can upload a file / folder to *CodaLab* to create a bundle, and then you can download the entire bundle or parts of the bundle. Each competition is freely created on the platform by uploading a bundle to the platform. A competition bundle is simply a zip file containing the competition yaml which defines different aspects and attributes of your competition, the logo, the html pages documenting your competition, and the data associated with your competition. A competition is composed of a phase or many phases defining the active times of the competition, along with some other settings such as execution time limit. For each phase, they have one or more tasks attached. A task is the problem, the submission should be solving, therefore submissions that solve a task can be thought of as a solution. A task consists of reference data, input data, a scoring program, and an ingestion program.

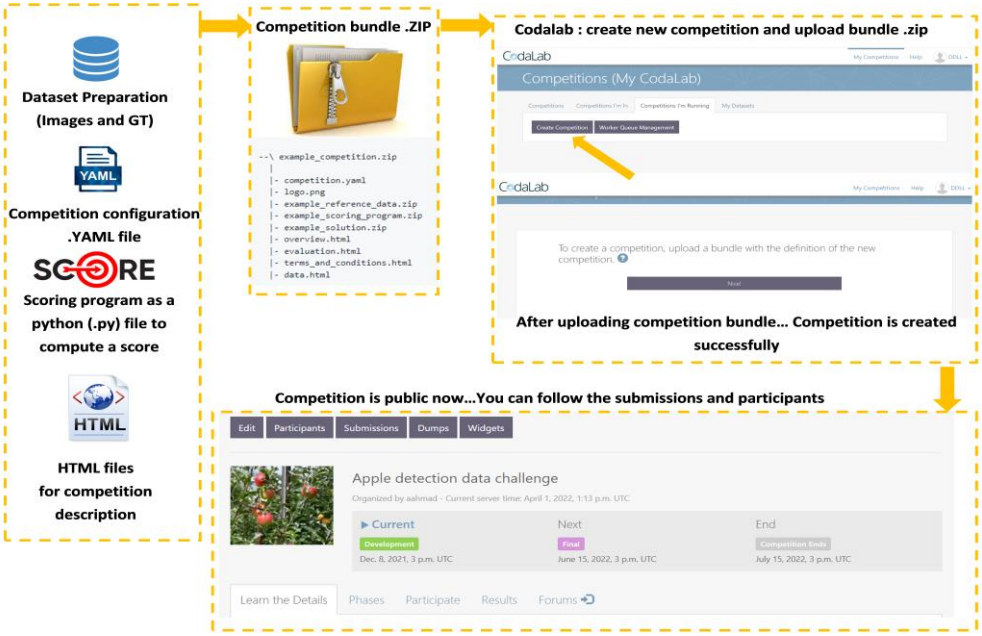


Figure 3: Main steps for the creation of a data challenge in *Codalab*.

Proposal of a code for baseline solution

In the proposed data challenge, we care about object detection. Several standard deep learning architectures for object detection can be found in the literature [22]. We selected the YOLO architecture to serve as the baseline code provided to the competitor of the data challenge. YOLO (You Only Look Once) is a state-of-the-art, real-time object detection architecture [23]. We opt for a YOLO model because it requires only a small dataset to achieve good results and to allow other participants to train their models without the need for powerful resources as YOLO is extremely fast, more than 1000x faster than R-CNN and 100x faster than Fast R-CNN [24].

YOLO reframes object detection as a single regression problem, straight from image pixels to bounding box coordinates and class probabilities. A single convolutional network simultaneously

predicts multiple bounding boxes and class probabilities for those boxes. YOLO trains on full images and directly optimizes detection performance [23]. The baseline model used in this competition is YOLOV4 Tiny. The improved architecture is explained carefully in [24]. While the full structure remains the same as the classical YOLO, YOLOV4 Tiny is an optimized and efficient object detection model. The architecture of the baseline model is the same one explained in the paper [24]. We offered the possibility to train the baseline model in the virtual environment *Collaboratory*. *Google Colaboratory* is a research tool for machine learning education developed by Google. It is a Jupyter notebook environment that requires no setup to use. Hence, choosing it in this competition as it allows every participant to have the required tools to compete in the data challenge. We conducted several training strategies. Table 2 presents the hyperparameters chosen and the mean-average precision metric on test data. Figure 4 shows the associate learning curve for strategy 3, on the validation data.

	Hyperparameters				
Training Strategies	**Batch size**	**Subdivision**	**Image resolution**	**Epochs**	**Mean-Average Precision (mAP)**
Strategy 1	**64**	**16**	**640x640**	**6000**	**30%**
Strategy 2	**32**	**8**	**640x640**	**6000**	**45.35%**
Strategy 3	**16**	**1**	**640x640**	**6000**	**50%**

Table 2: hyperparameters chosen in different training strategies.

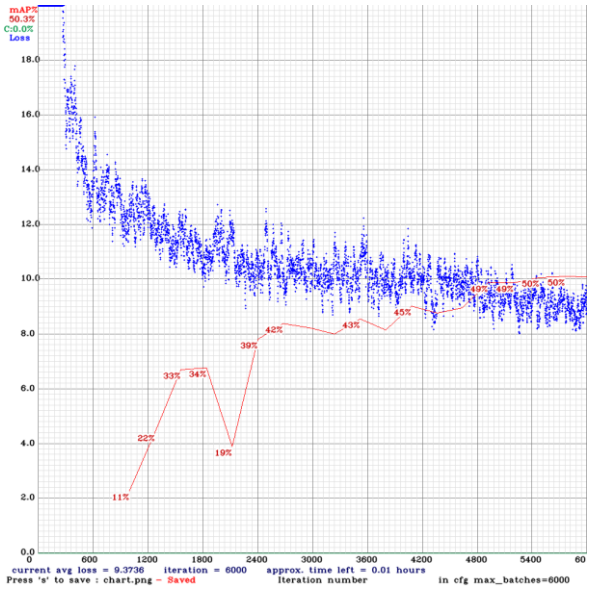


Figure 4: Learning curve of strategy 3, detailed in table 2.

We observe from table 2, that when we reduce the batch size, the mAP metric increases. Such result is trivial as the number of iterations for model parameters adjustment, increases, making the model more generalizable. Image resolution wasn't modified because the input images in YOLOV4 Tiny are compressed to the same resolution. The subdivision parameter is more related to memory allocation than model performance, as it allows instances in the batch to be processed at the same time and not simultaneously.

Besides the metric value on test data (see table 2), we conducted a visual inspection of the 29 images in the test set (using the model trained following strategy 3). Figure 5 shows examples of apple trees images after prediction using strategy 3. The model successfully detects apples in far-view, front view, and close-view images. In addition, the model shows robustness toward variant

acquisition conditions (as shown in figure 5 (b)) because these variations were included synthetical through data augmentation. In some cases, the model fails to detect all the apples (as shown in figure 5 (c)). Among the reasons, is the choice of the prediction threshold. If the threshold is higher than the apples confidence scores, they will not be detected. To demonstrate the impact of the threshold on the results, we repeated the prediction while reducing the threshold from 0.4 to 0.2. Figure 6 shows the image in figure 5(c) with the new prediction threshold. We observe more detected apples. In addition, we observe redundant bounding boxes around apples. This problem can be overcome with postprocessing operations such as Non-maximum Suppression [25].

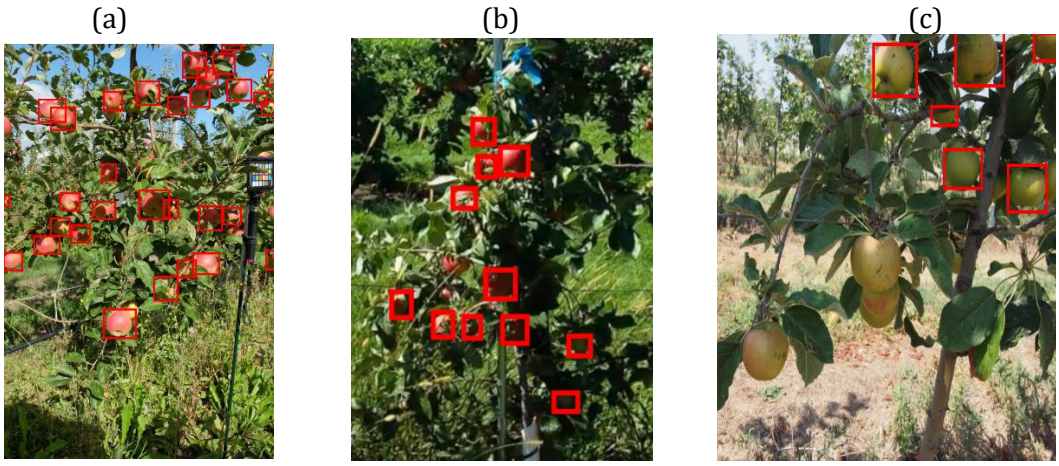


Figure 5: Illustrative examples of test data after prediction using threshold =0.4.

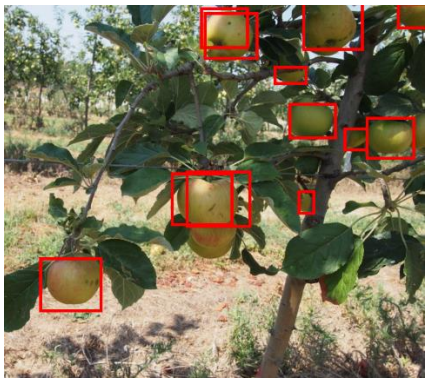


Figure 6: Figure 5 (c) using a threshold =0.2

Conclusion

In this communication, we provided a full pipeline to detect apple including an annotated data set, a trained algorithm, and a data challenge public website open to further investigation. We detailed all the steps of the creation of this ensemble of public information including the crucial annotation steps which was also performed on a publicly available website. We believe this knowhow would be highly valuable for the precision horticulture community to promote reproducible and open science practices and help to enhance the spreading of the numerical expertise throughout the community. This is specifically valuable for computer vision problem for which solutions already exists but do not include any shared annotated data nor codes. The result of the data challenge will be presented during the conference. To encourage more initiative of this kind in horticulture we also share a new data set dedicated to flowering (an important phenological stage) and an associated trained model.

References

[1] D.-L. Wulfsohn, F. Zamora, C. Tellez, I. Lagos, and M. Garcia-Finana, "Multilevel systematic sampling to estimate total fruit number for yield forecasts," *Precision Agriculture*, vol. 13, no. 2, pp. 256–275, 2012.

[2] O.E.Apolo-Apolo,M.Pérez-Ruiz,J.Mart´ınez-Guanter,and J.Valente, "A cloud-based environment for generating yield estimation maps from apple orchards using UAV imagery and a deep learning technique," *Frontiers in Plant Science*, vol. 11, p. 1086, 2020. [Online]. Available: https://www.frontiersin.org/article/10.3389/fpls.2020.01086

[3] S. Bargoti and J. Underwood, "Deep fruit detection in orchards," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 3626–3633.

[4] K. Bresilla, G. D. Perulli, A. Boini, B. Morandi, L. Corelli Grappadelli, and L. Manfrini, "Single-shot convolution neural networks for real-time fruit detection within the tree," *Frontiers in Plant Science*, vol. 10, p. 611, 2019. [Online]. Available: https://www.frontiersin.org/article/10.3389/fpls.2019.00611

[5] S. W. Chen, S. S. Shivakumar, S. Dcunha, J. Das, E. Okon, C. Qu, C. J. Taylor, and V. Kumar, "Counting apples and oranges with deep learning: A data-driven approach," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 781–788, 2017.

[6] N. Häni, P. Roy, and V. Isler, "Apple counting using convolutional neural networks," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 2559–2565.

[7] N.Häni,P.Roy,and V.Isler,"A comparative study of fruit detection and counting methods for yield mapping in apple orchards," *Journal of Field Robotics*, vol. 37, no. 2, pp. 263–282, 2020. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21902

[8] Y. Tian, G. Yang, Z. Wang, H. Wang, E. Li, and Z. Liang, "Apple detection during different growth stages in orchards using the improved YOLO-V3 model," *Computers and Electronics in Agriculture*, vol. 157, pp. 417 – 426, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S016816991831528X

[9] J. Gené-Mola, V. Vilaplana, J. R. Rosell-Polo, J.-R. Morros, J. Ruiz-Hidalgo, and E. Gregorio, "Multi-modal deep learning for Fuji apple detection using RGB-D cameras and their radiometric capabilities," *Computers and Electronics in Agriculture*, vol. 162, pp. 689 – 698, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0168169919301413

[10] X. Liu, S. W. Chen, S. Aditya, N. Sivakumar, S. Dcunha, C. Qu, C. J. Taylor, J. Das, and V. Kumar, "Robust fruit counting: Combining deep learning, tracking, and structure from motion," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1045–1052, 2018.

[11] S. Tu, Y. Xue, C. Zheng, Y. Qi, H. Wan, and L. Mao, "Detection of passion fruits and maturity classification using Red-Green-Blue Depth images," *Biosystems Engineering*, vol. 175, pp. 156 – 167, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1537511017309844

[12] H. A. Williams, M. H. Jones, M. Nejati, M. J. Seabright, J. Bell, N. D. Penhall, J. J. Barnett, M. D. Duke, A. J. Scarfe, H. S. Ahn, J. Lim, and B. A. MacDonald, "Robotic kiwifruit harvesting using machine vision, convolutional neural networks, and robotic arms," *Biosystems Engineering*, vol. 181, pp. 140 – 156, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S153751101830638X

[13] L. Fu, Y. Majeed, X. Zhang, M. Karkee, and Q. Zhang, "Faster R–CNN–based apple detection in dense-foliage fruiting-wall trees using RGB and depth features for robotic harvesting," *Biosystems Engineering*, vol. 197, pp. 245 – 256, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1537511020302002

[14] H. Gan, W. S. Lee, V. Alchanatis, and A. Abd-Elrahman, "Active thermal imaging for immature citrus fruit detection," *Biosystems Engineering*, vol. 198, pp. 291 – 303, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1537511020302348

[15] J. Xiong, Z. Liu, S. Chen, B. Liu, Z. Zheng, Z. Zhong, Z. Yang, and H. Peng, "Visual detection of green mangoes by an unmanned aerial vehicle in orchards based on a deep learning method," *Biosystems Engineering*, vol. 194, pp. 261 – 272, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1537511020300970.

[16] S. Bargoti and J. P. Underwood, “Image segmentation for fruit detection and yield estimation in apple orchards,” *Journal of Field Robotics*, vol. 34, no. 6, pp. 1039–1060, 2017.

[17] J.Gené-Mola, R.Sanz-Cortiella, J.R.Rosell-Polo, J.-R.Morros, J. Ruiz-Hidalgo, V. Vilaplana, and E. Gregorio, “Fruit detection and 3D location using instance segmentation neural networks and structure-from-motion photogrammetry,” *Computers and Electronics in Agriculture*, vol. 169, p. 105165, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0168169919321507

[18] A. Gongal, A. Silwal, S. Amatya, M. Karkee, Q. Zhang, and K. Lewis, “Apple crop-load estimation with over-the-row machine vision system,” *Computers and Electronics in Agriculture*, vol. 120, pp. 26 – 35, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S016816991500335X

[19] N. Häni, P. Roy, and V. Isler, “Minneapple: A benchmark dataset for apple detection and segmentation.”

[20] M. Zine-El-Abidine, H. Dutagaci, G. Galopin, and D. Rousseau, “Assigning apples to individual trees in dense orchards using 3D colour point clouds,” *Biosystems Engineering*, vol. 209, pp. 30–52, 2021.

[21] M. Jung, M. Roth, M. J. Aranzana, A. Auwerkerken, M. Bink, C. Denancé, C. Dujak, C.-E. Durel, C. F. i Forcada, C. M. Cantin *et al.*, “The apple refpop—a reference population for genomics-assisted breeding in apple,” *Horticulture research*, vol. 7, no. 1, pp. 1–16, 2020.

[22] Zaidi, Syed Sahil Abbas, et al. "A survey of modern deep learning based object detection models." Digital Signal Processing (2022): 103514.

[23] Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[24] Bochkovskiy, Alexey, Chien-Yao Wang, and Hong-Yuan Mark Liao. "Yolov4: Optimal speed and accuracy of object detection." arXiv preprint arXiv:2004.10934 (2020).

[25] N. Bodla, B. Singh, R. Chellappa and L. S. Davis, "Soft-NMS — Improving Object Detection with One Line of Code," 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 5562-5570, doi: 10.1109/ICCV.2017.593

Chapter 13

Annex H: *Ordinalysis*

Code Metadata

Nr.	Code metadata description	Please fill in this column
C1	Current code version	v2022.07.10
C2	Permanent link to code/repository used for this code version	`https://uabox.univ-angers.fr/index.php/s/OrgEdoXh57hPKRE`
C3	Code Ocean compute capsule	None
C4	Legal Code License	GNU GPL
C5	Code versioning system used	None
C6	Software code languages, tools, and services used	MATLAB(R2021a)
C7	Compilation requirements, operating environments & dependencies	Windows
C8	If available Link to developer documentation/	None
C9	Support email for questions	zinemouadaix@hotmail.fr

Table 13.1 – Code metadata

13.1 Motivation and significance

In recent years, machine learning algorithms have been exploited tremendously in all fields of data science due to their efficiency in exploring the data and taking the decision in feature spaces [169, 170]. A common criticism against the use of machine learning algorithms is their possible black box aspect. To go beyond their efficiency, it is therefore important to head toward interpretability of machine learning algorithms [123].

Interpretability concerns revealing the internal logic of the models, i.e. opening the black boxes.

In this work, we care about the interpretability of ordinal data, i.e. where an order is respected between a set of definite classes. Ordinal data exist in several fields such as pathology in medical imaging, diseases studies in plant sciences, and recommendation systems. They constitute specific types of data, distinct from nominal and quantitative data. While ordinal data are frequent, generic tools to analyze and visualize them are few to be found. In this paper, we present a standalone software named *Ordinalysis*, which provides support for the analysis and visualization of ordinality in latent spaces.

Ordinalysis embeds the dimension reduction technique BVP that we specially designed to project ordinal data in the direction where the ordinality is the most visible [142]. *Ordinalysis* also incorporates new metrics that we have developed to quantify the ordinality of ordinal data sets [143]. We provide, through *Ordinalysis*, the access to these new methodological tools with an ergonomic interface. We allow processing synthetic data sets for didactic purposes and uploading high dimensional latent space and image datasets, by the users.

13.2 Software description

13.2.1 Software architecture

The application can process three types of input data. First, to serve as an educational tool, the user can create in *Ordinalysis*, synthetic ordinal data and observe how the parameters (noise, number of instances, number of classes, the order of the classes) impact the ordinality in the feature space. The second and third type of data refer respectively to, real data of ordinal feature space in the CSV format or images of an ordinal dataset, saved in a folder with their corresponding ordinal labels (see Fig. 13.1). For these three types of input data, similar post-processing steps are provided. This includes dimension reduction, visualization and ordinality quantification. Last, a log window is available for the user to have a recap of the experiments conducted and the numerical values chosen during the configuration of ordinal data.

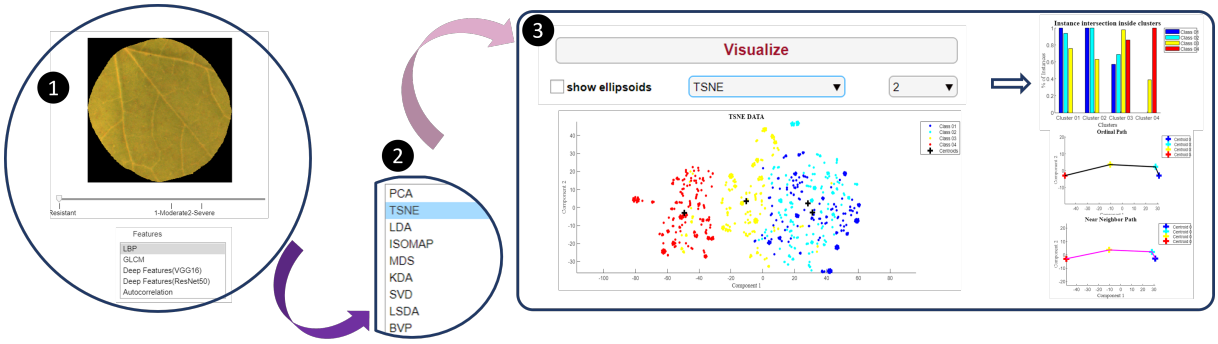


Figure 13.1 – Illustrative schema to use *Ordinalysis*, to process image ordinal data. (1): create the feature space by selecting the image dataset and the type of features. (2): select the dimension reduction (3): visualize ordinality and the ordinal metrics.

13.2.2 Programming environment

Ordinalysis is written in the MATLAB programming language (R2021a). The software can be used in conjunction with MathWorks' MATLAB commercial application, or distributed for free as a compiled standalone desktop application running on top of the MATLAB runtime compiler [171].

13.2.3 Software functionality

Data generation

Ordinal synthetic feature space: the user can adjust several parameters to create a synthetic ordinal feature space. This includes the following parameters: Shape of data (Swiss roll, aligned and curved data); Dimension (2D or 3D feature space); Number of classes (from 4 to 10 classes); The number of instances in each class (data created is balanced); Noise factor (variance σ in the Gaussian distribution); Pi factor (scalar value in the equation of the Swiss roll data); State of ordinality (allows permuting the order of the classes to either respect or break the ordinality).

Ordinal real feature space: an existing ordinal feature space can be imported by clicking on the button "load your feature space". Currently, two formats of the input are accepted. The first one is the MATLAB format, where the mat file contains two variables: Features and Labels. The second format is the CSV file, where the first column contains

the labels and the rest contains the features. After loading the feature space, its spatial dimension is displayed.

Ordinal real images space: users can load image datasets with associated ordinal labels. The software extracts features on the image ordinal dataset. The feature space created can be exported as a CSV file, where the first column contains the labels and the rest contains the features. Users can choose among the following features: Local binary patterns (LBP) [172] and Deep features [173] of the architecture VGG16 and the architecture ResNet50.

Dimension reduction techniques

Ordinalysis allows the user to visualize ordinal data after different dimension reduction techniques. Currently, the available dimension reduction techniques in the software are: PCA, TSNE, LDA, ISOMAP, MDS, LSDA and BVP [142]. The projected feature spaces can be exported by clicking on "Save feature space after projection".

Quantifying ordinality in latent space

In the reduced latent space, ordinality can be quantified using the metrics, *Inter-Class intersection* and *Deviation from ordinality*, described in [143]. The output values can be exported by clicking on "Save ordinal metrics". Two types of visualizations are available for the ordinality metrics. First, a histogram of the percentages of instances of all classes in each cluster computed via the *Inter-Class intersection* metric is provided. Second, a curve passing through the centers of classes in the correct order and in the order following the minimal distance between classes is provided. If ordinality is not well projected, then the curves will follow different paths.

13.3 Applications

In this section, we provide an example of the use of *Ordinalysis* on real ordinal data.

13.3.1 Selection of the best dimension reduction technique

In [142], we introduced a novel dimension reduction technique called BVP (best-view point), suitable for ordinal classification problems. We highlighted the cases where it

outperforms the existing dimension reduction techniques, on real ordinal classification datasets in [174, 175]. The figure shows the output visualizations of the dimension reduction techniques: TSNE, LDA, LSDA and BVP, applied on the ordinal feature space, "car", presented in [174, 175]. The ordinality is better projected on the side of BVP and LSDA with a slight improvement for BVP. This is a simple demonstration of the utility of the software at observing all results and switching between visualizations quickly. In Fig. 13.2, we show only the feature space, but the histogram and both paths are also updated while the user switches to different dimension reduction techniques.

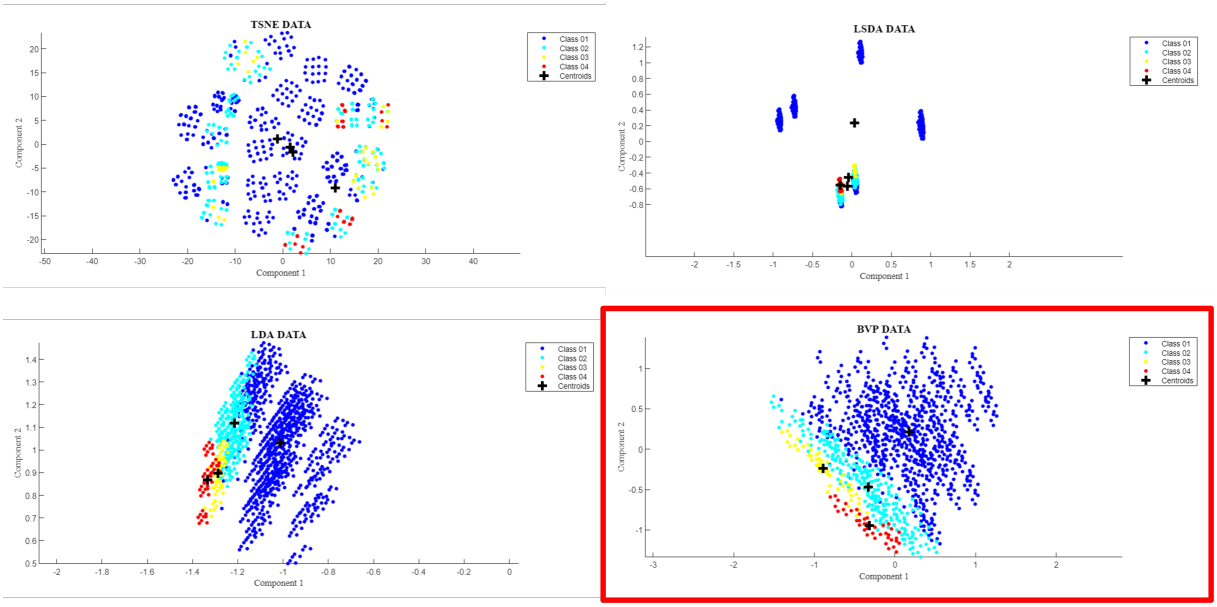


Figure 13.2 – Visualizing the dimension reduction techniques: TSNE, LDA, LSDA and BVP, applied on the ordinal feature space: car, presented in [174, 175].

13.3.2 Ordinality in images

In this work, we demonstrate the utility of *Ordinalysis* at analyzing ordinal data in VCU tests of variety testing. The test concerns measurements of the resistance of melon varieties to powdery mildew (see Fig. 13.3). Details about the manual inspection protocol, the automatic computer vision pipeline proposed and the results can be found in [176].

Using *Ordinalysis*, we aim to demonstrate that the classification results are coherent with the quality of ordinality in the feature space. To do that, we uploaded the annotated dataset of melon foliar disks, taken in front-lighting as described in [176] and applied the

following features: LBP, Deep features (VGG16), and Deep features (ResNet50). After, we projected the feature spaces created from N-D to 2D using the dimension reduction techniques: PCA, TSNE, and BVP. Figure 13.4 shows the projected feature space.

First, we observe that for all the dimension reduction techniques assessed, the Resnet50 and VGG16 deep features are better than LBP features, because the ordinality is better projected. We can also remark that BVP is slightly better than the other dimension reduction techniques as the classes are better distanced, while in TSNE and PCA, the point cloud of each class is sparse, leading to an intersection between instances of the other classes. To approve our analysis, we can also rely on the quantification metrics, especially *Inter-Class intersection* (ICC) presented in Tab. 13.2. The mean value representing the intersection between classes reaches its lowest value in the combination: deep features using ResNet50 and dimension reduction technique BVP. This result is in accordance with the best machine learning strategy found in [176].

Figure 13.3 – Illustration figure of the associate images to the scale used by the VCU examiners to measure the propagation of powdery mildew in melon foliar disks.

Table 13.2 – *Inter-Class intersection* values extracted from LBP, Deep features of VGG16 and deep features of ResNet50 latent spaces generated after applying the dimension reduction techniques: PCA, TSNE and BVP.

Features	Dimension Reduction	Cluster 1	Cluster 2	Cluster 3	Mean Clusters
LBP	PCA	0.948	0.364	0.98	0.764
	TSNE	1	0.4	1	0.8
	BVP	0.97	0.382	0.962	0.771333
Deep Features VGG16	PCA	0.498	0.4	0.996	0.631333
	TSNE	0.622	0.39	1	0.670667
	BVP	0.496	0.4	0.978	0.624667
Deep Features ResNet50	PCA	0.59	0.374	0.984	0.649333
	TSNE	0.702	0.352	1	0.684667
	BVP	0.496	0.348	1	0.614667

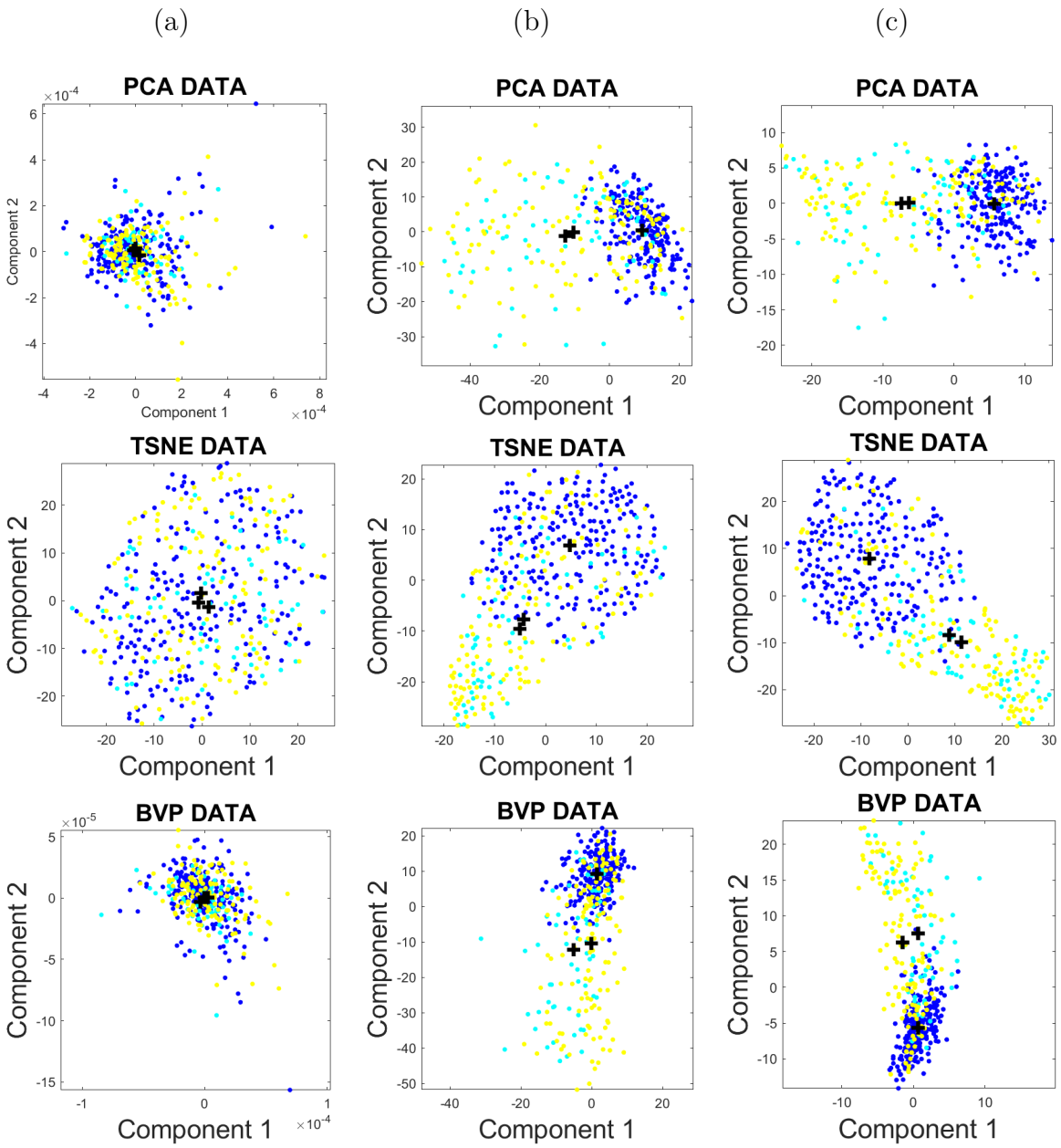


Figure 13.4 – Visualizing the dimension reduction techniques: PCA, TSNE, and BVP, applied on (a): LBP features, (b): deep features (VGG16) and (c): deep features (ResNet50) of the powdery mildew datasets (blue: resistant, cyan: moderate, yellow: severe).

13.4 Impact

Ordinalysis possibly interests several scientific communities, including: (i) computer scientists interested in ordinal data analysis in medical imaging, plant science and other

fields using ordinal data (ii) end-users with moderate knowledge of machine learning but with few experience of coding. The methodological elements of *Ordinalysis* were presented in signal processing conference [142, 143]. The proposed application was also tested on end-users composed of international experts in plant variety testing through the European project INVITE (https://www.h2020-invite.eu/). This community is very commonly using ordinal data for rating traits, and received *Ordinalysis* very positively.

13.4.1 Tutorial

A tutorial video for *Ordinalysis* was recorded. It is available at this link.

13.5 Conclusion

In this article, we presented *Ordinalysis* a unique software specially dedicated to the analysis of ordinal data. The software was successfully applied to real high dimensional ordinal data and images labeled with ordinal data and is now available for end-users concerned by the interpretability of their ordinal data sets.

BIBLIOGRAPHY

[1] Quan Qiu et al., « Field-based high-throughput phenotyping for maize plant using 3D LiDAR point cloud generated with a “Phenomobile” », *in*: *Frontiers in plant science* 10 (2019), p. 554.

[2] *DUS and VCUS Testing*, `https://www.geves.fr/about-us/variety-study-department/dus-vcus-testing/`, Accessed: 2022-05-30.

[3] *Phenoarch*, `https://www.english.arvalisinstitutduvegetal.fr/high-throughput-phenotyping-tools-at-arvalis-institut-du-vegetal-@/view-2158-arvstatiques.html`, Accessed: 2022-05-30.

[4] *Autonomous Electric Tractor - Le futur de l'agriculture | John Deere FR*, `https://www.youtube.com/watch?v=6KRvADZSBeY&ab_channel=JohnDeereFrance`, Accessed: 2022-05-30.

[5] Arti Singh et al., « Machine learning for high-throughput stress phenotyping in plants », *in*: *Trends in plant science* 21.*2* (2016), pp. 110–124.

[6] Aalt Dirk Jan van Dijk et al., « Machine learning in plant science and plant breeding », *in*: *Iscience* 24.*1* (2021), p. 101890.

[7] Abhinav Sharma et al., « Machine learning applications for precision agriculture: A comprehensive review », *in*: *IEEE Access* 9 (2020), pp. 4843–4873.

[8] Biyun Yang and Yong Xu, « Applications of deep-learning approaches in horticultural research: a review », *in*: *Horticulture Research* 8 (2021).

[9] Anne-Katrin Mahlein, « Plant disease detection by imaging sensors–parallels and specific demands for precision agriculture and plant phenotyping », *in*: *Plant disease* 100.*2* (2016), pp. 241–251.

[10] Qin Zhang, *Precision agriculture technology for crop farming*, Taylor & Francis, 2016.

[11] Michaela Jung et al., « The apple REFPOP—a reference population for genomics-assisted breeding in apple », *in*: *Horticulture research* 7.*1* (2020), pp. 1–16.

[12] Mouad Zine-El-Abidine et al., « Assigning apples to individual trees in dense orchards using 3D colour point clouds », *in*: *Biosystems Engineering* 209 (2021), pp. 30–52.

[13] Zhen Zhen, Lindi Quackenbush, and Lianjun Zhang, « Trends in Automatic Individual Tree Crown Detection and Delineation—Evolution of LiDAR Data », *in*: *Remote Sensing* 8 (Apr. 2016), p. 333.

[14] Tomas Brandtberg et al., « Detection and analysis of individual leaf-off tree crowns in small footprint, high sampling density lidar data from the eastern deciduous forest in North America », *in*: *Remote Sensing of Environment* 85.*3* (2003), pp. 290–303.

[15] Xingcheng Lu et al., « A bottom-up approach to segment individual deciduous trees using leaf-off LiDAR point cloud data », *in*: *ISPRS Journal of Photogrammetry and Remote Sensing* 94 (Aug. 2014), pp. 1–12.

[16] A. Tabb and H. Medeiros, « A robotic vision system to measure tree traits », *in*: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 6005–6012.

[17] Jose T. Colmenero-Martinez et al., « An automatic trunk-detection system for intensive olive harvesting with trunk shaker », *in*: *Biosystems Engineering* 172 (2018), pp. 92–101.

[18] Henry Medeiros et al., « Modeling Dormant Fruit Trees for Agricultural Automation », *in*: *Journal of Field Robotics* 34.*7* (2017), pp. 1203–1224, eprint: `https://onlinelibrary.wiley.com/doi/pdf/10.1002/rob.21679`.

[19] Jing Zhang et al., « Branch Detection with Apple Trees Trained in Fruiting Wall Architecture using Stereo Vision and Regions-Convolutional Neural Network(R-CNN) », *in*: *2017 ASABE Annual International Meeting*, Jan. 2017.

[20] Lihua Zeng, Juan Feng, and Long He, « Semantic segmentation of sparse 3D point cloud based on geometrical features for trellis-structured apple orchard », *in*: *Biosystems Engineering* 196 (2020), pp. 46–55.

[21] Michael Nielsen et al., « Orchard and tree mapping and description using stereo vision and lidar », *in*: *International Conference of Agricultural Engineering*, 2012.

[22] James P Underwood et al., « Lidar-based tree recognition and platform localization in orchards », *in*: *Journal of Field Robotics* 32.*8* (2015), pp. 1056–1074.

[23] Lishan Zhong et al., « Segmentation of individual trees from TLS and MLS Data », *in*: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 10.*2* (2016), pp. 774–787.

[24] Suchet Bargoti et al., « A pipeline for trunk detection in trellis structured apple orchards », *in*: *Journal of Field Robotics* 32.*8* (2015), pp. 1075–1094.

[25] Jordi Gené-Mola et al., « Multi-modal deep learning for Fuji apple detection using RGB-D cameras and their radiometric capabilities », *in*: *Computers and Electronics in Agriculture* 162 (2019), pp. 689–698.

[26] Tien Thanh Nguyen et al., « Detection of red and bicoloured apples on tree with an RGB-D camera », *in*: *Biosystems Engineering* 146 (2016), Special Issue: Advances in Robotic Agriculture for Crops, pp. 33–44.

[27] Yongting Tao and Jun Zhou, « Automatic apple recognition based on the fusion of color and 3D feature for robotic fruit picking », *in*: *Computers and Electronics in Agriculture* 142 (2017), pp. 388–396.

[28] Shuqin Tu et al., « Detection of passion fruits and maturity classification using Red-Green-Blue Depth images », *in*: *Biosystems Engineering* 175 (2018), pp. 156–167.

[29] Guichao Lin et al., « In-field citrus detection and localisation based on RGB-D image analysis », *in*: *Biosystems Engineering* 186 (2019), pp. 34–44.

[30] Longsheng Fu et al., « Faster R–CNN–based apple detection in dense-foliage fruiting-wall trees using RGB and depth features for robotic harvesting », *in*: *Biosystems Engineering* 197 (2020), pp. 245–256.

[31] O. Safren et al., « Detection of green apples in hyperspectral images of apple-tree foliage using machine vision », *in*: *Transactions of the ASABE* 50.*6* (2007), pp. 2303–2313.

[32] D. Stajnko, M. Lakota, and M. Hočevar, « Estimation of number and diameter of apple fruits in an orchard during the growing season by thermal imaging », *in*: *Computers and Electronics in Agriculture* 42.*1* (2004), pp. 31–42.

[33] D.M. Bulanon, T.F. Burks, and V. Alchanatis, « Study on temporal variation in citrus canopy using thermal imaging for citrus fruit detection », *in*: *Biosystems Engineering* 101.*2* (2008), pp. 161–171.

[34] D.M. Bulanon, T.F. Burks, and V. Alchanatis, « Image fusion of visible and thermal images for fruit detection », *in*: *Biosystems Engineering* 103.*1* (2009), pp. 12–22.

[35] Juan Pablo Wachs et al., « Low and high-level visual feature-based apple detection from multi-modal images », *in*: *Precision Agriculture* 11 (2010), pp. 717–735.

[36] Hao Gan et al., « Active thermal imaging for immature citrus fruit detection », *in*: *Biosystems Engineering* 198 (2020), pp. 291–303.

[37] Jordi Gené-Mola et al., « Fruit detection in an apple orchard using a mobile terrestrial laser scanner », *in*: *Biosystems Engineering* 187 (2019), pp. 171–184.

[38] Q.I. Wang et al., « Automated Crop Yield Estimation for Apple Orchards », *in*: *Experimental Robotics: The 13th International Symposium on Experimental Robotics*, vol. 88, June 2012, pp. 745–758.

[39] Subhajit Sengupta and Won Suk Lee, « Identification and determination of the number of immature green citrus fruit in a canopy under different ambient light conditions », *in*: *Biosystems Engineering* 117 (2014), Image Analysis in Agriculture, pp. 51–61.

[40] Sajad Sabzi et al., « Segmentation of Apples in Aerial Images under Sixteen Different Lighting Conditions Using Color and Texture for Optimal Irrigation », *in*: *Water* 10 (2018), p. 1634.

[41] A. Gongal et al., « Apple crop-load estimation with over-the-row machine vision system », *in*: *Computers and Electronics in Agriculture* 120 (2016), pp. 26–35.

[42] Pravakar Roy and Volkan Isler, « Vision-Based Apple Counting and Yield Estimation », *in*: *Kulić D., Nakamura Y., Khatib O., Venture G. (eds) 2016 International Symposium on Experimental Robotics. ISER 2016. Springer Proceedings in Advanced Robotics*, vol. 1, Oct. 2016, pp. 478–487.

[43] Suchet Bargoti and James P. Underwood, « Image Segmentation for Fruit Detection and Yield Estimation in Apple Orchards », *in*: *Journal of Field Robotics* 34.*6* (2017), pp. 1039–1060.

[44] Salma Samiei et al., « Toward Joint Acquisition-Annotation of Images with Egocentric Devices for a Lower-Cost Machine Learning Application to Apple Detection », *in*: *Sensors* 20.*15* (2020), p. 4173.

[45] Sashuang Sun et al., « Recognition of green apples in an orchard environment by combining the GrabCut model and Ncut algorithm », *in*: *Biosystems Engineering* 187 (2019), pp. 201–213.

[46] Aiping Gong et al., « Citrus yield estimation based on images processed by an Android mobile phone », *in*: *Biosystems Engineering* 115.*2* (2013), pp. 162–170.

[47] Jun Lu et al., « Immature citrus fruit detection based on local binary pattern feature and hierarchical contour analysis », *in*: *Biosystems Engineering* 171 (2018), pp. 78–90.

[48] Zhiliang He et al., « A method of green citrus detection based on a deep bounding box regression forest », *in*: *Biosystems Engineering* 193 (2020), pp. 206–215.

[49] Eliyahu (Efim) Kelman and Raphael Linker, « Vision-based localisation of mature apples in tree images using convexity », *in*: *Biosystems Engineering* 118 (2014), pp. 174–185.

[50] Raphael Linker, « Machine learning based analysis of night-time images for yield prediction in apple orchard », *in*: *Biosystems Engineering* 167 (2018), pp. 114–125.

[51] Gang Wu et al., « Automatic recognition of juicy peaches on trees based on 3D contour features and colour data », *in*: *Biosystems Engineering* 188 (2019), pp. 1–13.

[52] Orly Enrique Apolo-Apolo et al., « A Cloud-Based Environment for Generating Yield Estimation Maps From Apple Orchards Using UAV Imagery and a Deep Learning Technique », *in*: *Frontiers in Plant Science* 11 (2020), p. 1086.

[53] S. Bargoti and J. Underwood, « Deep fruit detection in orchards », *in*: *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 3626–3633.

[54] Kushtrim Bresilla et al., « Single-Shot Convolution Neural Networks for Real-Time Fruit Detection Within the Tree », *in*: *Frontiers in Plant Science* 10 (2019), p. 611.

[55] S. W. Chen et al., « Counting Apples and Oranges With Deep Learning: A Data-Driven Approach », *in*: *IEEE Robotics and Automation Letters* 2.*2* (2017), pp. 781–788.

[56] N. Häni, P. Roy, and V. Isler, « Apple Counting using Convolutional Neural Networks », *in*: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 2559–2565.

[57] Nicolai Häni, Pravakar Roy, and Volkan Isler, « A comparative study of fruit detection and counting methods for yield mapping in apple orchards », *in*: *Journal of Field Robotics* 37.*2* (2020), pp. 263–282, eprint: `https://onlinelibrary.wiley.com/doi/pdf/10.1002/rob.21902`.

[58] Yunong Tian et al., « Apple detection during different growth stages in orchards using the improved YOLO-V3 model », *in*: *Computers and Electronics in Agriculture* 157 (2019), pp. 417–426.

[59] Xu Liu et al., « Robust Fruit Counting: Combining Deep Learning, Tracking, and Structure from Motion », *in*: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2018), pp. 1045–1052.

[60] Henry A.M. Williams et al., « Robotic kiwifruit harvesting using machine vision, convolutional neural networks, and robotic arms », *in*: *Biosystems Engineering* 181 (2019), pp. 140–156.

[61] Juntao Xiong et al., « Visual detection of green mangoes by an unmanned aerial vehicle in orchards based on a deep learning method », *in*: *Biosystems Engineering* 194 (2020), pp. 261–272.

[62] Jordi Gené-Mola et al., « Fruit detection and 3D location using instance segmentation neural networks and structure-from-motion photogrammetry », *in*: *Computers and Electronics in Agriculture* 169 (2020), p. 105165.

[63] P. Roy, W. Dong, and V. Isler, « Registering Reconstructions of the Two Sides of Fruit Tree Rows », *in*: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 1–9.

[64] C.L. Scher, E. Griffoul, and C.H. Cannon, « Drone-based photogrammetry for the construction of high-resolution models of individual trees », *in*: *Trees* 33 (2019), pp. 1385–1397.

[65] C. Wu, « Towards Linear-Time Incremental Structure from Motion », *in*: *2013 International Conference on 3D Vision - 3DV 2013*, 2013, pp. 127–134.

[66] C. Wu et al., « Multicore bundle adjustment », *in*: *Conference on Computer Vision and Pattern Recognition 2011*, 2011, pp. 3057–3064.

[67] Y. Furukawa et al., « Towards Internet-scale multi-view stereo », *in*: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010, pp. 1434–1441.

[68] Y. Furukawa and J. Ponce, « Accurate, Dense, and Robust Multiview Stereopsis », *in*: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.*8* (2010), pp. 1362–1376.

[69] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Second, Cambridge University Press, ISBN: 0521540518, 2004.

[70] Pedro D. Marrero Fernández et al., « Fast and robust multiple ColorChecker detection using deep convolutional neural networks », *in*: *Image and Vision Computing* 81 (2019), pp. 15–24.

[71] T.C. Lee, R.L. Kashyap, and C.N. Chu, « Building Skeleton Models via 3-D Medial Surface Axis Thinning Algorithms », *in*: *CVGIP: Graphical Models and Image Processing* 56.*6* (1994), pp. 462–478.

[72] Richard O. Duda and Peter E. Hart, « Use of the Hough Transformation to Detect Lines and Curves in Pictures », *in*: *Commun. ACM* 15.*1* (Jan. 1972), pp. 11–15.

[73] P.H.S. Torr and A. Zisserman, « MLESAC: A New Robust Estimator with Application to Estimating Image Geometry », *in*: *Computer Vision and Image Understanding* 78.*1* (2000), pp. 138–156.

[74] Rafael C. Gonzalez and Richard E. Woods, *Digital Image Processing (3rd Edition)*, USA: Prentice-Hall, Inc., 2006.

[75] Thomas H. Cormen et al., *Introduction to Algorithms, Third Edition*, 3rd, The MIT Press, 2009.

[76] Shane Torbert, *Applied Computer Science*, 2nd, Springer Publishing Company, Incorporated, 2016.

[77] Yang Chen and Gérard G Medioni, « Object modeling by registration of multiple range images. », *in*: *Image and Vision Computing* 10.*3* (1992), pp. 145–155.

[78] UM Rao Mogili and BBVL Deepak, « Review on application of drone systems in precision agriculture », *in*: *Procedia Computer Science* 133 (2018), pp. 502–509.

[79] Mark Holden, « A review of geometric transformations for nonrigid body registration », *in*: *IEEE Transactions on Medical Imaging* 27.*1* (2007), pp. 111–128.

[80] Francis Hallé, *Architectures de Plantes*, 109 Avenue de Lodève, 34070 Montpellier, France: JPC Editions, 2004.

[81] Pravakar Roy et al., « Vision-based preharvest yield mapping for apple orchards », *in*: *Computers and Electronics in Agriculture* 164 (2019), p. 104897.

[82] Nicolai Häni, Pravakar Roy, and Volkan Isler, « Apple counting using convolutional neural networks », *in*: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2018, pp. 2559–2565.

[83] Nicolai Häni, Pravakar Roy, and Volkan Isler, « A comparative study of fruit detection and counting methods for yield mapping in apple orchards », *in*: *Journal of Field Robotics* 37.*2* (2020), pp. 263–282.

[84] Wenbo Dong, Pravakar Roy, and Volkan Isler, « Semantic mapping for orchard environments by merging two-sides reconstructions of tree rows », *in*: *Journal of Field Robotics* 37.*1* (2020), pp. 97–121, eprint: `https://onlinelibrary.wiley.com/doi/pdf/10.1002/rob.21876`.

[85] A. Gongal et al., « Sensors and systems for fruit detection and localization: A review », *in*: *Computers and Electronics in Agriculture* 116 (2015), pp. 8–19.

[86] Pierre-Eric Lauri, Eric Terouanne, and Jean-Marie Lespinasse, « Quantitative Analysis of Relationships between Inflorescence Size, Bearing-axis Size and Fruit-Set —An Apple Tree Case Study », *in*: *Annals of Botany* 77.*3* (Mar. 1996), pp. 277–286.

[87] P. E. Lauri et al., « Insights into secondary growth in perennial plants: its unequal spatial and temporal dynamics in the apple (Malus domestica) is driven by architectural position and fruit load », *in*: *Annals of Botany* 105.*4* (Mar. 2010), pp. 607–616.

[88] Geoffroy Couasnet et al., « Machine learning meets distinctness in variety testing », *in*: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 1303–1311.

[89] Mouad Zine-El-Abidine et al., « Apple Shape classification based on Drawings in Variety Testing catalogs », *in*: *Biosystems Engineering* (2022).

[90] Manuel Melgosa, Alain Trémeau, and Guihua Cui, « Colour difference evaluation », *in*: *Advanced color image processing and analysis*, Springer, 2013, pp. 59–79.

[91] Antoni Buades, Jose-Luis Lisani, and Jean-Michel Morel, « Dimensionality of color space in natural images », *in*: *JOSA A* 28.*2* (2011), pp. 203–209.

[92] Divya Srivastava, Rajesh Wadhvani, and Manasi Gyanchandani, « A review: color feature extraction methods for content based image retrieval », *in*: *International Journal of Computational Engineering & Management* 18.*3* (2015), pp. 9–13.

[93] Julien Chauveau et al., « Multifractal analysis of three-dimensional histogram from color images », *in*: *Chaos, Solitons & Fractals* 43.*1-12* (2010), pp. 57–67.

[94] Julien Chauveau, David Rousseau, and François Chapeau-Blondeau, « Fractal capacity dimension of three-dimensional histogram from color images », *in*: *Multidimensional Systems and Signal Processing* 21.*2* (2010), pp. 197–211.

[95] Jing Bai, Xiaohua Wang, and Licheng Jiao, « Image retrieval based on color features integrated with anisotropic directionality », *in*: *Journal of Systems Engineering and Electronics* 21.*1* (2010), pp. 127–133.

[96] Junding Sun et al., « Image retrieval based on color distribution entropy », *in*: *Pattern Recognition Letters* 27.*10* (2006), pp. 1122–1126.

[97] Surina Borjigin and Prasanna K Sahoo, « Color image segmentation based on multi-level Tsallis–Havrda–Charvát entropy and 2D histogram using PSO algorithms », *in*: *Pattern Recognition* 92 (2019), pp. 107–118.

[98] Gabriel Peyré, Marco Cuturi, et al., « Computational optimal transport: With applications to data science », *in*: *Foundations and Trends® in Machine Learning* 11.*5-6* (2019), pp. 355–607.

[99] Julien Rabin, Sira Ferradans, and Nicolas Papadakis, « Adaptive color transfer with relaxed optimal transport », *in*: *2014 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2014, pp. 4852–4856.

[100] Ju Han and Kai-Kuang Ma, « Fuzzy color histogram and its use in color image retrieval », *in*: *IEEE Transactions on image Processing* 11.*8* (2002), pp. 944–952.

[101] Zulfiqar Hasan Khan, Irene Yu-Hua Gu, and Andrew G Backhouse, « Robust visual object tracking using multi-mode anisotropic mean shift and particle filters », *in*: *IEEE transactions on circuits and systems for video technology* 21.*1* (2011), pp. 74–87.

[102] Wenhan Luo et al., « Multiple object tracking: A literature review », *in*: *Artificial Intelligence* (2020), p. 103448.

[103] Jana Wäldchen and Patrick Mäder, « Machine learning for image based species identification », *in*: *Methods in Ecology and Evolution* 9.*11* (2018), pp. 2216–2225.

[104] Sapan Naik and Bankim Patel, « Machine vision based fruit classification and grading-a review », *in*: *International Journal of Computer Applications* 170.*9* (2017), pp. 22–34.

[105] Rashmi Pandey, Sapan Naik, and Roma Marfatia, « Image processing and machine learning for automated fruit grading system: A technical review », *in*: *International Journal of Computer Applications* 81.*16* (2013), pp. 29–39.

[106] Manali R Satpute and Sumati M Jagdale, « Color, size, volume, shape and texture feature extraction techniques for fruits: a review », *in*: *Int. Res. J. Eng. Technol* 3 (2016), pp. 703–708.

[107] Dexter I Mercurio and Alexander A Hernandez, « Classification of sweet potato variety using convolutional neural network », *in*: *2019 IEEE 9th international conference on system engineering and technology (ICSET)*, IEEE, 2019, pp. 120–125.

[108] Xiaoling Yang et al., « Spectral and image integrated analysis of hyperspectral data for waxy corn seed variety classification », *in*: *Sensors* 15.*7* (2015), pp. 15578–15594.

[109] Michael F Barnsley, *Fractals everywhere*, Academic press, 2014.

[110] Thomas M Cover and Joy A Thomas, « Information theory and statistics », *in*: *Elements of Information Theory* 1.*1* (1991), pp. 279–335.

[111] Cédric Villani, *Optimal transport: old and new*, vol. 338, Springer, 2009.

[112] I. Bloch and Jamal Atif, « Deux approches pour la comparaison de relations spatiales floues. Transport optimal et morphologie mathématique », *in*: *Rev. d'Intelligence Artif.* 29 (2015), pp. 595–619.

[113] Chika Yinka-Banjo and Ogban-Asuquo Ugot, « A review of generative adversarial networks and its application in cybersecurity », *in*: *Artificial Intelligence Review* 53.*3* (2020), pp. 1721–1736.

[114] Xianlin Zhang et al., « A survey on freehand sketch recognition and retrieval », *in*: *Image and Vision Computing* 89 (2019), pp. 67–87.

[115] T Ishikawa et al., « CLASSIFICATION OF STRAWBERRY FRUIT SHAPE BY MACHINE LEARNING. », *in*: *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences* 42.*2* (2018).

[116] Susovan Jana and Ranjan Parekh, « Shape-based fruit recognition and classification », *in*: *International Conference on Computational Intelligence, Communications, and Business Analytics*, Springer, 2017, pp. 184–196.

[117] Kamran Kheiralipour and Abbas Pormah, « Introducing new shape features for classification of cucumber fruit based on image processing technique and artificial neural networks », *in*: *Journal of food process engineering* 40.*6* (2017), e12558.

[118] Conghui Hu et al., « Sketch-a-classifier: Sketch-based photo classifier generation », *in*: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9136–9144.

[119] Christian Dujak et al., « Genetic and morphological study of apple fruit shape », *in*: (2020).

[120] Frank P Kuhl and Charles R Giardina, « Elliptic Fourier features of a closed contour », *in*: *Computer graphics and image processing* 18.*3* (1982), pp. 236–258.

[121] Jaime S Cardoso and Ricardo Sousa, « Measuring the performance of ordinal classification », *in*: *International Journal of Pattern Recognition and Artificial Intelligence* 25.*08* (2011), pp. 1173–1195.

[122] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep learning*, MIT press, 2016.

[123] Amina Adadi and Mohammed Berrada, « Peeking inside the black-box: A survey on Explainable Artificial Intelligence (XAI) », *in*: *IEEE Access* 6 (2018), pp. 52138–52160.

[124] Jian-Xun Mi, An-Di Li, and Li-Fang Zhou, « Review Study of Interpretation Methods for Future Interpretable Machine Learning », *in*: *IEEE Access* 8 (2020), pp. 191969–191985.

[125] Sérgio Pereira et al., « Brain tumor segmentation using convolutional neural networks in MRI images », *in*: *IEEE transactions on medical imaging* 35.*5* (2016), pp. 1240–1251.

[126] *Deep learning for multi-task plant phenotyping*, 2017, pp. 2055–2063.

[127] Andrea Seveso et al., « Ordinal labels in machine learning: a user-centered approach to improve data validity in medical settings », *in*: *BMC Medical Informatics and Decision Making* 20.*5* (2020), pp. 1–14.

[128] Diogo V Carvalho, Eduardo M Pereira, and Jaime S Cardoso, « Machine learning interpretability: A survey on methods and metrics », *in*: *Electronics* 8.*8* (2019), p. 832.

[129] Eibe Frank and Mark Hall, « A Simple Approach to Ordinal Classification », *in*: *Proceedings of the 12th European Conference on Machine Learning*, Berlin, Heidelberg: Springer-Verlag, 2001, pp. 145–156.

[130] Xu Liu et al., « Robust Fruit Counting: Combining Deep Learning, Tracking, and Structure from Motion », *in*: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2018), pp. 1045–1052.

[131] JesÃºs Bobadilla et al., « Generalization of recommender systems: Collaborative filtering extended to groups of users and restricted to groups of items », *in*: *Expert Systems with Applications* 39.*1* (2012), pp. 172–186.

[132] J. Xie and C. Pun, « Deep and Ordinal Ensemble Learning for Human Age Estimation From Facial Images », *in*: *IEEE Transactions on Information Forensics and Security* 15 (2020), pp. 2361–2374.

[133] I. Borg and P.J.F. Groenen, *Modern Multidimensional Scaling: Theory and Applications*, Springer, 2005.

[134] Joshua B. Tenenbaum, Vin de Silva, and John C. Langford, « A Global Geometric Framework for Nonlinear Dimensionality Reduction », *in*: *Science* 290.*5500* (2000), pp. 2319–2323, eprint: `https://science.sciencemag.org/content/290/5500/2319.full.pdf`.

[135] Daniel D. Lee and H. Sebastian Seung, « Learning the parts of objects by non-negative matrix factorization », English (US), *in*: *Nature* 401.*6755* (Oct. 1999), pp. 788–791.

[136] Laurens van der Maaten and Geoffrey Hinton, « Visualizing Data using t-SNE », *in*: *Journal of Machine Learning Research* 9.*86* (2008), pp. 2579–2605.

[137] Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*, 2nd ed., New York: Wiley, 2001.

[138] Deng Cai et al., « Locality Sensitive Discriminant Analysis », *in*: *International Joint Conference on Artificial Intelligence (IJCAI'07)*, 2007.

[139] B. Sun et al., « Kernel Discriminant Learning for Ordinal Regression », *in*: *IEEE Transactions on Knowledge and Data Engineering* 22.*6* (2010), pp. 906–910.

[140] Liliana Forzani et al., « Supervised dimension reduction for ordinal predictors », *in*: *Computational Statistics & Data Analysis* 125 (2018), pp. 136–155.

[141] B. Kwon et al., « Optimal Camera Point Selection Toward the Most Preferable View of 3-D Human Pose », *in*: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* (2020), pp. 1–21.

[142] Mouad Zine-El-Abidine, Helin Dutagaci, and David Rousseau, « Dimensionality Reduction for Ordinal Classification », *in*: *2021 29th European Signal Processing Conference (EUSIPCO)*, IEEE, 2021, pp. 1531–1535.

[143] P. Roy, W. Dong, and V. Isler, « Vers une quantification de l'interprétabilité des espaces latents pour la classification ordinale », *in*: *Groupe de Recherche en Traitement du Signal et des Images,(GRETSI)*, 2022.

[144] Nicolas Boumal, P. A. Absil, and Coralia Cartis, « Global rates of convergence for nonconvex optimization on manifolds », English (US), *in*: *IMA Journal of Numerical Analysis* 39.*1* (Jan. 2019), pp. 1–33.

[145] Jonathan Barzilai and Jonathan M. Borwein, « Two-Point Step Size Gradient Methods », *in*: *IMA Journal of Numerical Analysis* 8.*1* (Jan. 1988), pp. 141–148, eprint: `https://academic.oup.com/imajna/article-pdf8/1/141/2402762/8-1-141.pdf`.

[146] Javier Sánchez-Monedero, Pedro A. Gutiérrez, and María Pérez-Ortiz, « ORCA: A Matlab/Octave Toolbox for Ordinal Regression », *in*: *Journal of Machine Learning Research* 20.*125* (2019), pp. 1–5.

[147] P. A. Gutiérrez et al., « Ordinal Regression Methods: Survey and Experimental Study », *in*: *IEEE Transactions on Knowledge and Data Engineering* 28.*1* (2016), pp. 127–146.

[148] *A simple approach to ordinal classification*, Springer, 2001, pp. 145–156.

[149] Jaime S Cardoso and Joaquim F Costa, « Learning to classify ordinal data: The data replication method », *in*: *Journal of Machine Learning Research* 8.*Jul* (2007), pp. 1393–1429.

[150] Jaime S Cardoso and Ricardo Sousa, « Measuring the performance of ordinal classification », *in*: *International Journal of Pattern Recognition and Artificial Intelligence* 25.*08* (2011), pp. 1173–1195.

[151] *Interpreting deep learning models for ordinal problems.* 2018.

[152] Javier Sánchez-Monedero et al., « Exploitation of pairwise class distances for ordinal classification », *in*: *Neural computation* 25.*9* (2013), pp. 2450–2485.

[153] Eric Sven Ristad and Peter N Yianilos, « Learning string-edit distance », *in*: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20.*5* (1998), pp. 522–532.

[154] Fuzhen Zhuang et al., « A comprehensive survey on transfer learning », *in*: *Proceedings of the IEEE* 109.*1* (2020), pp. 43–76.

[155] Zicong Jiang et al., « Real-time object detection method based on improved YOLOv4-tiny », *in*: *arXiv preprint arXiv:2011.04244* (2020).

[156] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, « U-net: Convolutional networks for biomedical image segmentation », *in*: *International Conference on Medical image computing and computer-assisted intervention*, Springer, 2015, pp. 234–241.

[157] William S Noble, « What is a support vector machine? », *in*: *Nature biotechnology* 24.*12* (2006), pp. 1565–1567.

[158] Neena Aloysius and M Geetha, « A review on deep convolutional neural networks », *in*: *2017 international conference on communication and signal processing (ICCSP)*, IEEE, 2017, pp. 0588–0592.

[159] K. Hirakawa, *ColorChecker Finder*, `http://iss.udayton.edu/software`, 2013.

[160] Yanyuan Ma and Liping Zhu, « A review on dimension reduction », *in*: *International Statistical Review* 81.*1* (2013), pp. 134–150.

[161] Oliver Kramer, « K-nearest neighbors », *in*: *Dimensionality reduction with unsupervised nearest neighbors*, Springer, 2013, pp. 13–23.

[162] Lior Rokach and Oded Maimon, « Decision trees », *in*: *Data mining and knowledge discovery handbook*, Springer, 2005, pp. 165–192.

[163] Irina Rish et al., « An empirical study of the naive Bayes classifier », *in*: *IJCAI 2001 workshop on empirical methods in artificial intelligence*, vol. 3, 22, 2001, pp. 41–46.

[164] Nikhil R Pal and Sankar K Pal, « A review on image segmentation techniques », *in*: *Pattern recognition* 26.*9* (1993), pp. 1277–1294.

[165] Zhengxia Zou et al., « Object detection in 20 years: A survey », *in*: *arXiv preprint arXiv:1905.05055* (2019).

[166] Wei Liu et al., « Ssd: Single shot multibox detector », *in*: *European conference on computer vision*, Springer, 2016, pp. 21–37.

[167] Joseph Redmon et al., « You only look once: Unified, real-time object detection », *in*: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[168] Ross Girshick et al., « Rich feature hierarchies for accurate object detection and semantic segmentation », *in*: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[169] Jafar Alzubi, Anand Nayyar, and Akshi Kumar, « Machine learning from theory to algorithms: an overview », *in*: *Journal of physics: conference series*, vol. 1142, 1, IOP Publishing, 2018, p. 012012.

[170] Longbing Cao, « Data science: a comprehensive overview », *in*: *ACM Computing Surveys (CSUR)* 50.*3* (2017), pp. 1–42.

[171] Matlab Compiler, « MathWorks », *in*: 2021.01.03.

[172] Timo Ahonen, Abdenour Hadid, and Matti Pietikäinen, « Face recognition with local binary patterns », *in*: *European conference on computer vision*, Springer, 2004, pp. 469–481.

[173] Bolei Zhou et al., « Learning deep features for discriminative localization », *in*: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2921–2929.

[174] Javier Sánchez-Monedero, Pedro Antonio Gutiérrez, and Maria Pérez-Ortiz, « ORCA: a matlab/octave toolbox for ordinal regression », *in*: *Journal of Machine Learning Research* 20 (2019).

[175] Pedro Antonio Gutiérrez et al., « Ordinal regression methods: survey and experimental study », *in*: *IEEE Transactions on Knowledge and Data Engineering* 28.*1* (2015), pp. 127–146.

[176] Mouad Zine El Abidine et al., « Machine learning-based classification of powdery mildew severity on melon leaves », *in*: *International Conference on Image and Signal Processing*, Springer, 2020, pp. 74–81.

[177] Zine El Abidine Mouad, David Rousseau, and H.Dutagaci, « Ordinalysis: Interpretability of Ordinal Data in Machine Learning. Application in Plant Science. », *in*: *SoftwareX* (2022).

[178] D Rousseau Zine El Abidine H Dutagaci, « Motion-based acquisition to speed up annotation. Application to Russeting classification », *in*: *International Plant Phenotyping Symposium (IPPS)*, 2022.

[179] D Rousseau Zine El Abidine H Dutagaci, « Reducing the complexity of annotation though transfer learning from indoor to outdoor. Application to Russeting classification », *in*: *International Horticultural Congress (IHC)*, 2022.

UNIVERSITE BRETAGNE LOIRE / MATHSTIC

université angers

Titre : Contributions à la vision par ordinateur et à l'apprentissage automatique pour les tests des variétés végétales

Mot clés : Vision par ordinateur, apprentissage automatique, reconstruction 3D, automatisation, tests des variétés végétales, pommiers, robotique, apprentissage profond.

Résumé : Cette thèse propose des contributions originales de vision par ordinateur et apprentissage automatique pour les tests des variétés végétales. L'imagerie pour les plantes s'est développée ces dernières années en direction du phénotypage pour des expérimentations en milieu contrôlé ou pour le domaine de l'agriculture. Le domaine des tests des variétés végétales consiste à réaliser des mesures pour valider la qualité et l'originalité de toute nouvelle variété avant d'autoriser sa commercialisation. Il a été jusqu'ici peu étudié au moyen d'outils numériques et les tests actuels sont le résultat d'inspections visuelles. Dans cette thèse, nous avons contribué à la vision par ordinateur et à l'apprentissage automatique appliqués aux tests de variétés de pommes, en particulier les tests de distinction pendant les périodes de pré-récolte (dans les vergers) et de post-récolte (dans les milieux contrôlés). Les méthodologies proposées ont une valeur générique.

Automatiser les mesures réalisées durant un test de distinction dans les vergers des tests des variétés représente un défi puisque chaque arbre appartient à une variété et que les arbres peuvent être proches les uns des autres. Nous avons développé une méthode originale pour séparer les arbres. Cette méthode a été appliquée au comptage des pommes en les associant à leurs variétés [12].

Nous avons démontré la possibilité d'évoluer vers des tests de distinction numérique dans des milieux contrôlés en utilisant l'apprentissage automatique de manière supervisée par le biais du transport optimal. Cette approche a été illustrée dans le test de distinction basé sur la couleur de pomme [88]. De même, nous avons proposé une méthode alternative basée sur l'apprentissage automatique non supervisé. Certains traits dans les tests des variétés sont mesurés par des examinateurs à l'aide des dessins issus du catalogue de l'union pour la protection des obtentions végétales (UPOV). Nous avons montré pour la première fois qu'il est possible d'utiliser ces dessins comme référence numérique pour permettre l'automatisation des mesures sans remettre en cause les pratiques actuelles [89].

Dans les tests des variétés, certains traits sont mesurés suivant une échelle ordinale. L'ordinalité peut être un critère à vérifier dans l'espace des caractéristiques pour valider une méthode d'apprentissage automatique. Nous avons introduit une nouvelle méthode de réduction de dimension spécifiquement adaptée à la visualisation de données sur des échelles ordinales et deux métriques pour quantifier l'ordinalité dans l'espace des caractéristiques [142, 143]. Ces techniques ont été illustrées sur des données synthétiques et des données réelles issues de tests de résistance des maladies foliaires [177, 176].

L'utilisation de l'apprentissage automatique supervisé nécessite l'annotation des données, ce qui peut prendre beaucoup de temps. Nous avons développé une approche d'acquisition d'images originale qui permet de réduire le temps d'annotation et envisager une transition numérique qui ne constitue pas une perte de temps pour les experts annotateurs [178]. Dans la même optique, nous avons démontré que les images acquises dans un en-

vironnement contrôlé pouvaient être ajoutées aux images acquises dans les vergers et ainsi "booster" l'apprentissage automatique via l'apprentissage par transfert [179].

Title: Contributions to computer vision and machine learning for plant variety testing

Keywords: Computer vision, machine learning, 3D reconstruction, automatisation, variety testing, apples, robotics, deep learning.

Abstract: This thesis proposes original contributions to computer vision and machine learning for plant variety testing. Plant variety testing is a set of tests performed by examiners to certify candidate varieties to be registered in the catalog of varieties for commercialization. Computer vision is widely used in precision agriculture and plant breeding, unlike in variety testing measurements, which are based on visual inspection. In this thesis, we contributed to the computer vision and machine learning applied to apple variety testing, particularly the distinctness tests during the preharvest and post-harvest periods. The methodologies proposed are generalizable and low-cost.

Automating measurements for the preharvest distinctness tests in variety testing orchards is challenging since each tree belongs to a variety and trees can be close to each other (intersecting). We developed a novel methodology to separate trees. This methodology was applied in the apple fruit counting and associated the counting to the correct tree [12].

We demonstrated the possibility to shift toward numerical distinctness tests during the post-harvest period using machine learning in a supervised manner through optimal transport. This approach was illustrated in the distinctness test based on color [88]. Likewise, we proposed an alternative method based on unsupervised machine learning. Some traits in variety testing are measured by examiners using drawings from the catalog of the international union for the protection of new varieties of plants (UPOV). The drawings serve as a reference for the human eye. We have shown for the first time that it is possible to use these drawings as a numerical reference to allow the automation of measurements without disrupting or challenging current practices [89].

In variety testing, some traits are measured on an ordinal scale. The ordinality can be a criterion to be checked in the feature space to validate a machine learning method. We introduced a new dimension reduction method specifically adapted to visualize data on ordinal scales and two metrics to quantify ordinality in feature space [142, 143]. These techniques have been illustrated on synthetic data and real data from disease resistance testing [177, 176].

The use of supervised machine learning requires the annotation of data which can be time-consuming. We developed a novel image acquisition approach that reduces annotation time and allows for a numerical transition of variety testing that is not a waste of time for expert annotators [178]. For the same purpose, we demonstrated that images acquired in a controlled environment could be used to perform data augmentation on the orchard datasets and thus "boost" the machine learning training via the transfer learning [179].