Application of DEEP LEARNING to the processing of TERRESTRIAL LiDAR data for the evaluation of ARCHITECTURAL FEATURES and functioning of fruit trees

Juan Pablo Rojas Bustos

THÈSE POUR OBTENIR LE GRADE DE DOCTEUR DE L'UNIVERSITÉ DE MONTPELLIER

En Informatique

École doctorale Information Structures Systèmes - I2S

Unité de recherche AMÉLIORATION GÉNÉTIQUE ET ADAPTATION DES PLANTES MÉDITÉRRANÉENNES ET TROPICALES - AGAP

Application du DEEP LEARNING au traitement de données LIDAR TERRESTRE pour l'évaluation de TRAITS ARCHITECTURAUX et de fonctionnement d'arbres fruitiers

Présentée par Juan Pablo ROJAS BUSTOS

le [29 Novembre 2023]

Sous la direction de Christophe Fiorio
et Emmanuel Faure, Evelyne Costes et Frédéric Boudon

Devant le jury composé de

[David ROUSSEAU, Pr., CNRS]	[Rapporteur]
[Geraldine MORIN, Pr., ENSEEIHT]	[Rapporteur]
[Marie WEISS, Dr., INRAE]	[Examinateur]
[Eric DUCHÊNE, Ingénieur de recherche HDR., INRAE]	[Examinateur]
[Frederic BOUDON, Dr., CIRAD]	[Encadrant]
[Emmanuel FAURE, Dr., LIRMM]	[Encadrant]
[Evelyne COSTES, Dr., INRAE]	[Co-Dir. Thèse]
[Christophe FIORIO, Pr., LIRMM]	[Dir. de Thèse]


UNIVERSITÉ
DE MONTPELLIER

Résumé

Face aux effets du changement climatique, à l'augmentation de la population et à la nécessité d'assurer la sécurité alimentaire, la compréhension du développement des plantes est un élément essentiel des solutions. Cette étude vise à répondre à ces préoccupations urgentes. Pour entreprendre une telle étude, il est nécessaire d'explorer comment la variabilité génétique donne lieu à des ensembles spécifiques de caractéristiques architecturales. Dans le cas des arbres fruitiers qui seront le support de cette étude, la compréhension de ces variations architecturales facilite la régulation, la prédiction et l'assurance de la production. Elle permettra également de proposer des stratégies de gestion visant à maintenir la santé de l'arbre et à optimiser sa productivité.

Les arbres fruitiers se composent généralement de deux parties : la partie supérieure, comprenant le tronc, les branches et les autres organes aériens, et la partie souterraine comprenant l'ensemble du système racinaire. Dans les deux cas, l'étude de l'architecture soulève diverses questions concernant la géométrie de l'arbre, les gènes impliqués dans son développement et ses variations, ainsi que les états successifs au cours de son ontogénie. En répondant à ces questions, la compréhension de l'architecture des arbres fruitiers peut contribuer à l'amélioration des variétés et de leur production.

C'est pourquoi plusieurs techniques ont été développées pour mesurer les organes de l'arbre et modéliser leur architecture, incluant leur topologie et leur forme. Initialement, cette tâche nécessitait un groupe d'opérateurs humains pour mesurer physiquement ces objets directement sur le terrain. Cependant, cette approche prend beaucoup de temps et nécessite des observations approfondies. Au cours des dernières décennies, l'avènement de nouveaux capteurs tels que les caméras et le LiDAR (Light Detection and Ranging) a facilité l'acquisition de représentations numériques des arbres, ouvrant ainsi de nouvelles possibilités d'étude.

L'objectif de cette thèse est d'obtenir des descripteurs de l'architecture donnant une charactérisation globale de la géométrie de l'arbre, puis une approximation plus précise au niveau des organes. Ces descripteurs ont été basées sur le traitement de nuages de points

obtenus à partir de LiDAR terrestres et aériens dans un verger de pommiers contenant une collection de génotypes. Les nuages de points ont été traités en développant deux pipelines utilisant divers algorithmes de filtrage, d'apprentissage automatique et d'apprentissage profond.

Le premier chapitre présentera une comparaison des indices architecturaux obtenus à partir du LiDAR aérien et du LiDAR terrestre. Dans le deuxième chapitre, l'accent sera mis sur l'application de l'apprentissage profond pour la segmentation des organes (fruits, feuilles et branches). Enfin, le troisième chapitre présentera une approche du développement d'un modèle GAN (Generative Adversarial Network) dans le but de générer des pommiers synthétiques entièrement annotés, dans le but de faciliter l'entraînement plus efficace d'autres modèles d'apprentissage profond.

Abstract

With the effects of climate change, a growing human population, and the need to ensure food security, understanding the development of plants is a crucial part of the possible solutions. This study is considered to be highly significant in addressing these pressing concerns. To undertake such a study, it is necessary to explore how genetic variability gives rise to specific sets of architectural traits. Understanding these architectural variations facilitates the regulation, prediction and assurance of fruit production. It will also allow management strategies to be proposed to maintain the health of the tree and optimize its productivity.

Fruit trees typically consist of two components: the upper part, including the trunk, branches, and other organs above ground, and the underground part encompassing the entire root system. In both cases, studying the architecture raises various questions concerning tree geometry, the genes involved in tree development and variation, and the successive states of the tree. By addressing these questions, understanding the architecture of fruit trees can contribute to the improvement of varieties and their production.

Hence, several techniques have been developed to measure tree organs and model their architecture and shape. Initially, this task required a group of human operators to physically measure these objects directly in the field. However, this approach is time-consuming and demands extensive observation. Over the past few decades, the advent of new sensors such as cameras and LiDAR (Light Detection and Ranging) has facilitated the acquisition of digital representations of trees, opening up new possibilities for study.

The objective of this thesis is to obtain architectural metrics that provide an overall perspective of the tree's geometry, followed by a more precise approximation at the organ level. These metrics were based on the processing of point clouds obtained from terrestrial and aerial LiDAR in an apple orchard including a collection of genotypes. The point clouds were processed by developing two pipelines utilizing various filtering, machine learning, and deep learning algorithms.

The first chapter will present a comparison of architectural indices obtained from aerial LiDAR and terrestrial LiDAR. In the second chapter, the focus will be on the application of

deep learning for organ segmentation (fruit, leaves, and branches). Lastly, the third chapter will introduce an approach to develop a GAN (Generative Adversarial Network) model with the aim of generating fully annotated synthetic apple trees. This will facilitate more efficient training of other deep learning models.

Acknowledgement

I am particularly grateful to my supervisors for their guidance, patience and support throughout my doctoral journey.

I thank Ruben Hernandez, Carol Martinez, Gregory Beurier and Llorenç Cabrera, without whom I would not have taken the decision to start my PhD.

I would like to thank the AFEF and PHENOMEN teams for their support. I am particularly grateful to Guillaume PEREZ for his help in carrying out the field measurements, to Simon Arzet for his explanations, collaboration and guidance at the beginning of my thesis, to Amaury Branthomme for his collaboration and the discussions we had, which helped me a lot in solving various problems.

A very special thanks to my girlfriend Lika Mamutova for her support, effort and patience in every moment. "*It is mainly thanks to you that I was able to reach the end of this enterprise, without your support and your words I would have fainted long ago, thank you*".

I thank my best friend Alejandro Baron and my friend Claudia Huertas for their advice and support.

Sommaire

CHAPTER 1

Introduction

The study of genetic diversity in plants enables scientists to better understand how plants have evolved over time and adapted to various environmental conditions. Similarly, the study of plant architecture enables the identification and understanding of plant development patterns and adaptation to diverse growth environments.

Plant architecture studies make it possible to characterize the spatial distribution of organs, as well as the geometric characteristics of each one. Moreover, organ physiological characteristics provide information on plant adaptation to a given environment. Also, by understanding the architecture of plants, it has been possible to determine how different management practices allow adequate development and yield, given the environment, including pressure from pests and diseases.

Understanding how certain architectural features lead to specific plant behavior, makes it possible to use them in breeding programs to select genotypes with appropriate traits, resulting in plants with higher adaptation and regular production.

This highlights the importance of characterizing plant architecture and its relationship to plant genetic variability, leading to an important development in the research community. For this, a large number of technologies and methodologies have been developed to carry out more specific measurements and analyses. It should be noted, however, that carrying out such studies presents a number of difficulties, as each species or type of plant has very specific characteristics that make the creation of generic methodologies and algorithms extremely difficult.

An interesting way to analyze plant architecture is through 3D modeling. It consists in creating a virtual representation of the plant that attempts to replicate both the geometries and the internal functions of the plant. This type of approach has been of great interest in the last decade because, from this digital representation, it is possible to simulate how changing different features of the plant will affect its growth or production. In addition, this

type of approach allows analysis and sampling time to be reduced and allows the effects of environmental or structural and physiological changes on the plant to be analyzed in a more controlled manner.

Different protocols and technologies have been developed to create such 3D models of plants, starting from the manual measurement of each organ and growth pattern of the plant.

Nowadays, the use of sensors such as cameras and LiDAR to model plants in 3D is widely studied. This is because it is possible with these technologies to obtain the representation of a large number of individuals with a low investment of time, effort, and money. The representations of such sensors are geometrically more accurate, with additional information such as reflectance or temperature values. This allows models such as light interception and carbon distribution to be applied more accurately. However, this type of representation needs to be further studied and developed, as there is still a long way to go before we can take full advantage of it.

The 3D representations generated by the LiDAR sensor are highly valuable as they provide an accurate description of the crop or the targeted plant. The LiDAR sensor captures these targets in the form of point clouds, which not only offer a 3D representation of the plant but also include reflectance data associated with each scanned point. These point clouds solely represent the visible plant's surface, which results in varying degrees of occlusion. Some plant organs may not be fully described due to this limitation.

Additionally, when analyzing this type of point cloud, it's essential to consider that, apart from occlusion, other deformations may arise from the sensor's operation. The sensor's mode of operation introduces white noise in the position of the points describing the surface and outliers when one laser beam interacts with two or more surfaces, leading to a false estimation of some point positions. Furthermore, if the target undergoes movement during scanning, various deformations may occur in the scanned geometry.

Given the significance of plant architecture, its large within-species genetic variability in the case of fruit trees, and the capabilities presented by LiDAR sensors, this thesis will concentrate on extracting architectural traits from point clouds of fruit trees. This endeavor is crucial not only for sustaining fruit production but also for establishing methodologies that facilitate the accurate analysis and selection of genotypes resilient to forthcoming climatic changes.

To approach this goal, point clouds of an apple orchard consisting of different genotypes will be processed. The analysis of the point clouds will be carried out using machine learning and deep learning algorithms. The application of these algorithms aims to ensure the correct processing of the point clouds but also to develop algorithms that generalize the information coming from the data. Once the point clouds have been processed using these techniques, the architectural features are extracted.

In order to develop such framework, this thesis is structured as follows: Chapter 1 presents the state of the art of the topics of interest, Chapter 2 presents a first methodology that glob-

ally evaluates the architecture of apple trees from airborne LiDAR scans, Chapter 3 describes a method to identify organs, in this case, apples from terrestrial scans, and finally, Chapter 4 address the critical problem of generating annotated data required to parametrize DL/ML models. In particular, the use of synthetic data is explored.

CHAPTER 2

State of the art

To characterize precisely the growth and development of plants, it's essential to measure and model the geometrical and physiological changes that the plant goes through over time in a given environment. It is equally important to link this study to the genetic variability unique to each plant species. This information provides valuable insights on plant behavior and development, enabling the selection of genotypes that are resistant to specific environments while facilitating the development of new genotypes with improved yields.

Therefore, section 2.1 clearly explains what tree architecture is. Based on this description, the studies of architecture in apple trees are presented in section 2.2. In sections 2.3 and 2.4, I present the most used techniques to model plant architecture and how different sensors have been used to obtain measurements of tree architecture and improve plant modeling, respectively. From the measurement techniques, I pay special attention to LiDAR sensors and the processing of point clouds (section 2.5). Finally, in section 2.7, I focus on how machine learning has helped to process data in agriculture and, more precisely, in the analysis of tree architecture.

2.1 Plant architecture

Plant architecture considers the three-dimensional organization of the plant body. For the upper part of the plant, plant architecture considers variables such as branching patterns and size, the shape, positions, and orientation of leaves, flowers, fruits, and other organs (1). The plant architecture is the result of the relationship between endogenous growth processes and exogenous constraints given by the environment (2). Endogenous processes refer to all the internal processes that a plant does to consume nutrients, grow, and develop each of its organs. The exogenous constraints refer to the characteristics of the environment in which

the plant grow. Plant architecture is a point of reference and strong influence in agronomy, due to its effect on the sustainability of plant cultivation, the improvement of yield and efficiency with which a plant can be harvested (3; 4).

Plant architecture representations an be broadly divided depending on the scale of representation. In the review made by (1), three main approaches are described. *The first approach defines the plant as a whole*. In this approach, all organs of the same type tend to contribute similarly to plant functioning (water uptake, photosynthesis, etc.). Consequently, the architecture of the plant is modeled by one or more compartments. Two types of representations can be used to construct the model: geometric representations and compartment based representations. In the geometric representation at a global scale, the plants are modeled using a parametric geometric model. For example, spheres and ellipses can be used to model the light interception in the crowns (5); cylinders and cones have been used to understand mechanical properties of the plants (6).

The compartment based representations model the exchange of substances that takes place in the whole body of the plant. To do this, the plant is represented by two or more compartments. A compartment represent the source or interfaces for the exchange of substances. For example, a compartment can represents the pools of leaves or fruits as a whole, with no difference or distance considered between independent organs. Each compartment is defined as equation and, in the whole model, the equations of the N compartments interact between each other.

This approach and its different representations help in designing parsimonious models, but they may struggle to represent more complex behaviors, such as internal competition for nutrients, due to the lack of details in the model.

The second approach represents the tree architecture as a modular system. This approach is grounded in the concept that, there are several recurring units along the plant body, from which multiple mathematical representations can be formulated and combined to generate a more intricate representation. In this case, a module represents an organ or a physiological function performed by the plant. To implement this concept, the plant is decomposed into modular representations. Different types of representation also exists for this approach. In the first type of representations, the plant is modeled considering only the 3D spatial distribution of the plant. In this representation, the spatial distribution of the plant is represented using a 3D spatial grid. In such a representation, only the voxels of the grid that contain organs are considered. Each voxel is assigned with different attributes (leaf densities, reflectances, temperatures, etc.) depending on the elements that it encapsulates. The second approach consider an organ-based decomposition. In this approach, each plant organ is represented as a module. The organ-based decomposition has been approximated from two different representations. The first one is based on the plant geometry and the second one is based on the plant topology. In the plant geometry-based representation, each of the independent organs is defined as a geometric module. Each of the modules is defined by the shape of the organ and its spatial localisation and orientation. In this representation,

the organs do not interact with each other and only the organs of interest need to be considered, such as leaves for light interception analysis. The topological organ based decomposition defines the organs as modules and represent their organisation using a tree graph. This allows to takes into account their interaction, in particular to simulate ecophysiological processes. In general, these approaches provides a more accurate representation of plant behavior. However, it should be noted that this type of model are more computationally demanding. Moreover, they requires precise information on the structure of the plant that may difficult to acquire.

The third approach for modeling architecture is based on multi-scale representations. This approach considers multiple levels of detail, capturing and correlating plant behavior across scales. The architecture and diverse patterns observed in trees have fractal-like characteristics confined within a specific range of scales. Example of such description are the tree branch structures, Figure 1 shows an example of such a representation.

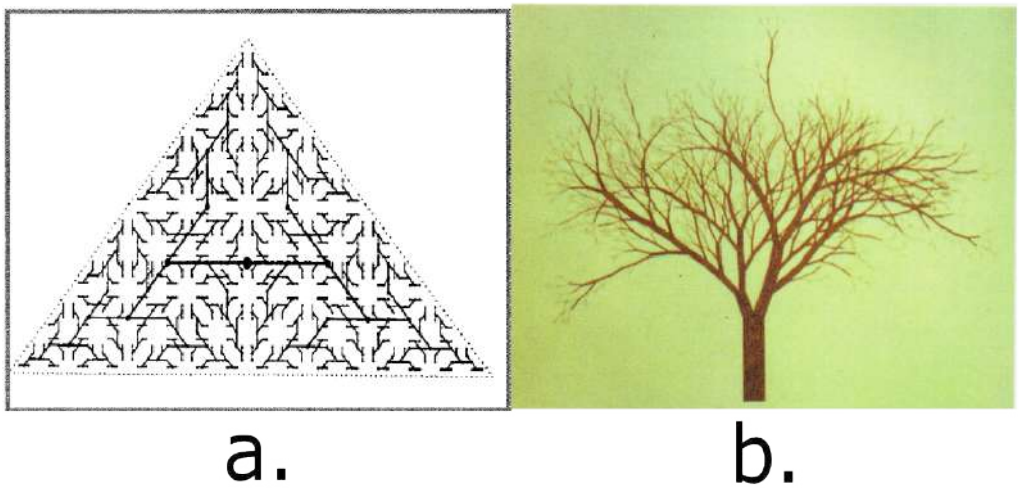


Figure 1 – **a.** Self-similar formal tree resulting from a fractal process. The theoretical output is a fractal object, i.e. it has details at every scale and has an infinite length (7) **b.** Virtual and realistic tree obtained from a fractal process (e.g. (8), Photo by Viennot). Figures and caption from (1; 7; 8)

In such an approach, different types of representation can also be built: spatial representations, geometric representations, and topological representations. In the spatial representation, the tree is partitioned by a multi-scale grid. For the first one, the voxel size of a grid based representation is adjusted based on the irregularity of the geometry. Greater irregularity leads to finer voxel granularity. In the multi-scale geometric representation, the tree geometry is divided into smaller and independent shapes until the desired accuracy is achieved. In the multi-scale topological representation, the plant organization is defined at multiple scales, each scale is represented by a tree graph. The relationship between graphs at different scales is defined by a quotient operator that maps a group of modules at a lower scale to a module at a macroscopic scale.

Plant architecture is primarily influenced by two factors, the genetics of the plant and the environment in which it grows. These factors impact processes such as photosynthesis,

water, and nutrient uptake, and the robustness of the plant to different stress levels.

Environmental factors like sunlight, water, and wind significantly influence fruit development and tree growth. Light can alter bud burst, plant shoot geometry, stomatal function, and chlorophyll content (9; 10; 11). Water stress causes developmental changes by triggering molecular adaptations that conserve water, impacting the plant growth and, process such as photosynthesis (12; 13). For instance, with Granny Smith apple trees, alterations in the transition time between vegetative and reproductive stages have been observed (14). Wind alters a tree's physical structure, affecting crown size, trunk diameter, root development, and the plant's mechanical properties such as flexibility and strength (15; 16).

Plant architecture has also been explored from a physiological perspective, revealing the influence of specific genes on biosynthetic processes responsible for the production of hormones such as auxins, cytokinins, and abscisic acid, which ultimately impact plant development (17; 18). For instance, in the apple cultivar *McIntosh Wijcik*, the gene $MdDOX - Co$ was found to increase the biosynthesis of bioactive gibberellins, elevate strigolactone levels, and enhance abscisic acid production, leading to the development of dwarf trees (19).

Specific genes have been observed to influence specific architectural traits. Much attention has been paid to dwarfing mutants in the context of food production, which has played a fundamental role in the green revolution, which began with the creation of dwarf varieties of rice and wheat(20). In fruit trees, these studies have been also approached. The columnar growth of apple trees is associated with a dominant allele located at locus 18.51 on chromosome 10. The identification and study of this gene could pave the way for the development of new cultivars not only in apples but also potentially in other species such as pear and peach (21). Also, the genes responsible for initiating the flowering process are conserved in all flowering plants even though there may be variations associated with floral architecture (22). In the apple tree, the $MdTFL1$ gene has also been identified as playing a role in juvenility and flowering, and it has been shown it could be utilized to manipulate the flowering process (23).

From the above, we can conclude that understanding the relationship between plant architecture, genes, and the environment can help to enhance the plant's ability to resist various levels of environmental stress, improve its productivity, and contribute to achieving food security.

2.2 Apple trees

Apples are the third most produced fruit in the world (24). On a global scale, apples accounted for 10% of fruit production between 2000 and 2021 (25). In Europe, production is projected to increase by 1% by 2023 (26). Consequently, this thesis will focus on this economically important species, with an emphasis on studying the architectural changes resulting from genetic variability.

2.2.1 Tree architecture

Apple trees exhibit great variability in their growth habits and topology, as their stature can range from dwarf to tall (27; 28). Figure 2 presents the range of size resulting from the interaction between a given genotype and different rootstocks (used for root system) from dwarfing to standard (29). In general, trees grafted on dwarf and semi-dwarf rootstocks will bear their first fruit 2 or 3 years after planting. Moreover, the environment, including factors such as water and nutrient availability, as well as temperature fluctuations (30; 31) can modulate tree size.

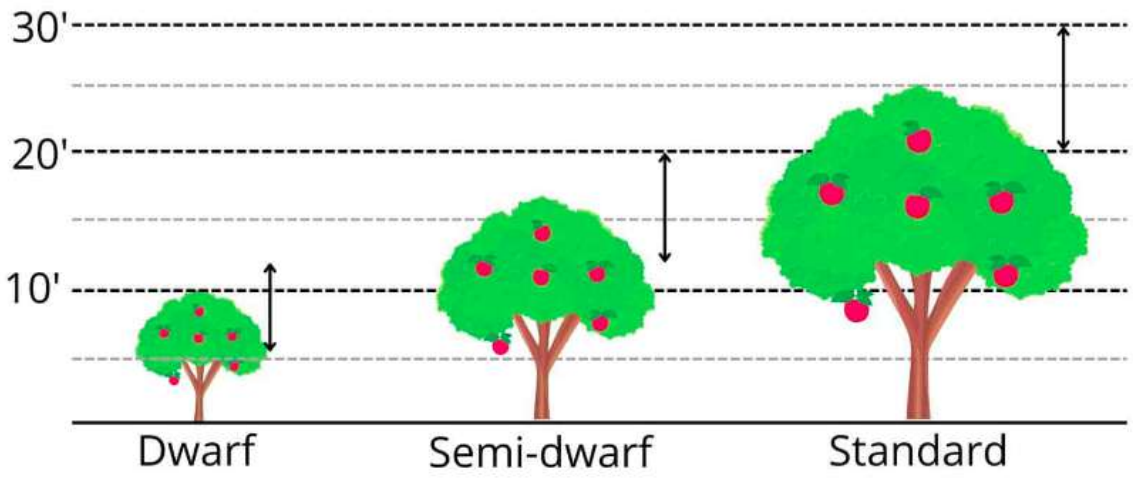


Figure 2 – Representation of the range for the different sizes of apple trees. Figure modified from (29)

Apple trees are allogamous, meaning that they require another variety to ensure the pollination process (32). The growth cycles of these trees are affected by seasonal changes. In winter, due to low temperatures, the shoots become dormant, and they start to grow again when temperatures rise in spring (33). Like many other perennial fruit trees, the apple tree may have irregular or biennial fruit production. In the case of biennial bearing, a year with high production but small fruits are followed by a year of low production with large fruits (34). *ON* years are characterized by a high density of flowering and minimal fruit dropping, whereas *OFF* years are characterized by a low density of flowering or high levels of fruit dropping (35).

Apple tree architecture is characterized by the presence of different shoot types, namely long, medium, and short shoots (3). The longest shoots develop during the initial years of an apple tree's life. Thereafter appears polycyclism, which is the capacity of plants to stop growing and resume new growth from the same shoots within a single growing season. This phenomenon is particularly observed during the adolescent phase of apple tree life and diminishes as the tree matures (37; 38). With aging, the growth process decreases while flowering appears more frequently but also more or less regularly depending on the cultivar. Consequently, with a tree distinct shoot types emerge, as depicted in Figure 3. Despite this

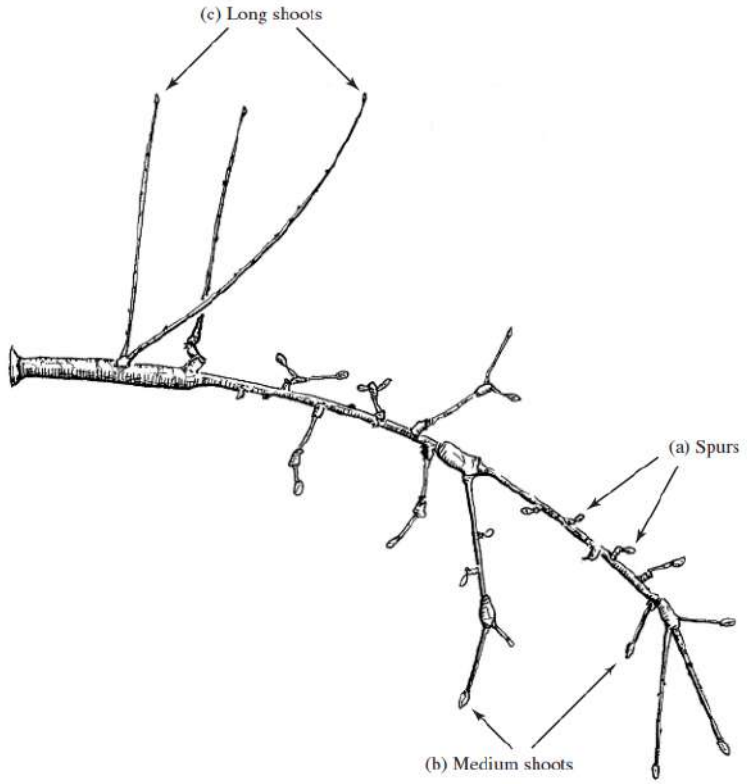


Figure 3 – Representation of shoot and bud types along an apple tree branch. Figure from (36)

scheme being observed whatever the cultivar, every genotype exhibits a typical variation in its ontogenetic development, based on both the time of year and tree's age (38).

The architecture of apple cultivars has been classified into four categories based on several variables, including the angle at which branches connect to the main trunk, branch distribution, and the arrangement of fruit on different types of branches (32; 39). Type 1 is characterized by a columnar geometry with short internodal distances and an abundance of short axillary shoots that emerge from the main trunk. These branches have the potential to bear flowers. Type 2 and 3 are defined as medium-sized trees in terms of their architecture. Type 2 is characterized by a significant number of spurs capable of bearing fruit. On the other hand, Type 3 exhibits a combination of short and long branches along the main trunk, with fruits typically borne at the ends of these branches. Type 4 is described as a vigorously growing tree, with fruits borne at the ends of its branches, causing its growth to extend towards the ground. These distinct characteristics classify Type 1 and Type 2 as alternate crop varieties, while Type 3 and Type 4 are considered regular production types. An alternate cultivar refers to a variety that exhibits high yields in one period and low yields in the next. Regular crops, on the other hand, consistently maintain a constant range of yields over the years. The types of apple tree architecture can be seen in Figure 4.

Based on this qualitative classification and on visual evaluation of traits, more in-depth genetic studies have been performed to better understand the genetic determinism of archi-

tectural traits (40; 41)

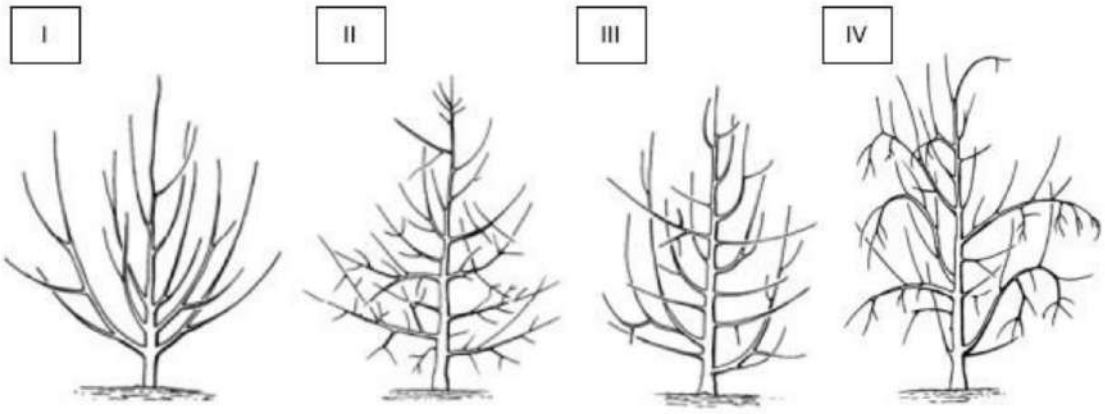


Figure 4 – Representation of the 4 types of apple tree architecture. Figure from (42)

Given the complexity of apple tree architectures and their characteristics, it is evident that the study and quantitative characterizations of a large population of trees that are mandatory to genetic studies are extensive and meticulous (43). However, considering the focus of this thesis, primary attention will be given to the genetic variability and architecture of apple trees.

2.2.2 Genetic variability

Over the last few decades, significant efforts have been made to sequence the genome of apple trees in order to gain a better understanding of their history, physiological development(44), and architectural characteristics (40; 41). In terms of architecture, the objective is to decipher the genetic determinisms of growth, branching (43) and flowering process (45; 46) as these elements play a crucial role in ensuring the productivity of apple trees.

Maintaining and increasing fruit production is essential for fruit growers. To ensure this, it is necessary to reduce the time that fruit trees remain in the juvenile stage, i.e. the tree must reach the adult stage and be able to regulate the production of vegetative and reproductive buds efficiently (47). This has been approached by studies that link these characteristics to specific genes (45). It has been demonstrated that flowering genes are not well related to biennial bearing in apple trees, and that such behavior is more related to the genes that control the tree hormones (45).

Dwarf apple trees are characterized by producing more buds, flowers, and fruit (48; 49). However, they are also more prone to biennial bearing (50; 51). Research has linked the overexpression or modification of certain genes to dwarfing in apple trees (52). It has also been shown how the activation of certain genes is related to the flowering process (53; 46). Gene *MdTFL1* is related to the time in its juvenile phase and controlling its expression can cause early flowering (23). Studies have been carried out to predict irregularity. In addition, these variables were used to predict tree behavior and detect QTL (54).

On either biparental or multiparent populations these studies have demonstrated the polygenitc nature of many architectural and flowering traits.

2.3 Modeling Plant Architecture

The field of plant modeling involves the development of mathematical models and software that aid in describing and understanding the functioning and growth of plants (55). Various approaches have been proposed in plant modeling. The first approach called process-based models (PBM), focuses on describing the physiological processes of plants independently of their architecture and morphology. Alternatively structure-based models (SPM), specifically consider the architectural features of plants. In recent years, there has been a growing trend toward integrating these two approaches, resulting in the development of functional-structural plant models (FSPM). FSPMs aim to create a virtual representation that closely resembles the actual plant of interest, encompassing both its physiological processes and architectural characteristics (56). In this section, I will focus on Functional-Structural Plant Models (FSPM) as a tool to simulate the 3D architecture of fruit trees.

2.3.1 What is FSPM?

FSPM models aim to understand plant growth by integrating the physiological processes and architecture development of the plant. FSPM models place particular emphasis on the spatial distribution of plant organs and commonly generate 3D models to represent the plant's architecture in their output (56). The development of these models requires collaboration across various disciplines, including mathematics, biology, and computer science.

FSPM models have found widespread application in the modeling of diverse fruit trees such as apple (57), mango (58; 59), peach (60) and avocado (61). These models have been developed to deepen our understanding of fruit tree growth and to explore the impact of architectural features and environmental dynamics on plant growth and fruit production. These models make it possible to study various ecophysiological processes, such as light interception and carbon distribution in the different organs of the plant (58). They also help to reduce the time needed to carry out field measurements by enabling large numbers of trees to be produced synthetically (59).

2.3.2 MAppleT

MAppleT is a stochastic model based on L-systems (62) that integrates the topology and geometry of apple trees to simulate the evolution of their architecture through the interaction of various processes. This model incorporates Hidden Markov Chains to simulate the growth sequences. The biomechanics of the plant are used to simulate the geometry of the GU (57). Figure 5 illustrates the growth of an apple tree simulated by the MAppleT model.

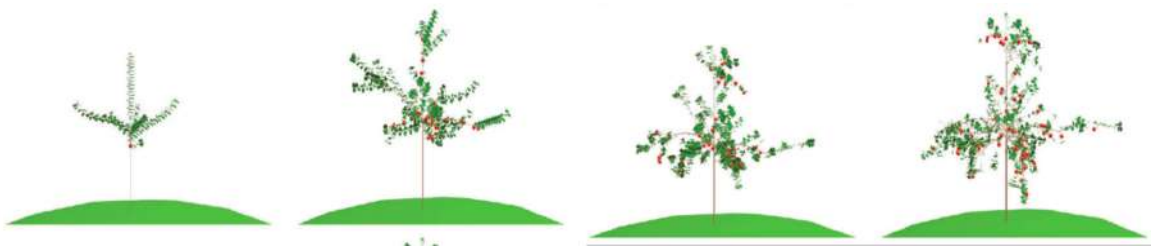


Figure 5 – Year-to-year changes of a typical Fuji apple tree simulated using default values of model parameters. The tree is visualized each year from the second to the fifth year of growth, just before harvest time. Figure and caption from (57)

The development of MAppleT is based on the analysis of field measurements of an apple crop. The measurements began in the fourth year after planting. In the analysis, two scales of the organization are precisely defined: growth units (GU), and metamers. A grow unit is defined as the stem elongation during the expansion phase and the metamers are defined as the segment where the shoot will grow. Additionally, the GUs are further classified into four categories: long GU, medium GU, short GU, and floral GU. The succession of GUs along the axes and the branching patterns was modeled as a two-stage stochastic process using a hierarchical hidden Markov model. The development of growth units was modeled using state Markov chains. Figure 6 depicts the representation of this model.

2.4 Phenotyping

In order to understand the development and behavior of the plants and to be able to create plant models, it is necessary to develop various measurements that describe the plant attributes.

Plant phenotyping can be defined as the art or field of science that is in charge of describing, and characterizing the anatomical, physiological, biochemical, and architectural behavior of a plant in a quantitative manner. This is to provide solid support for the decisions that farmers, plant breeders must make to ensure crop health, to select the best genotypes, and in general to understand the relationship between the plant and the environment (63; 64).

Over the last decades due to the development of new technologies (RGB cameras, multispectral, LiDAR, IoT sensors, SPAD) the protocols, and methods for data analysis have evolved drastically. This set of changes and the current need to understand more precisely the behavior of the plant due to the increase in the human population and the reduction of available resources, has led to the evolution of how phenotyping is done.

Traditional phenotyping methods are constrained by the time, cost, and throughput needed to carry them out (65). However, the emergence of new technologies has made it possible to conduct measurements and analyses more efficiently, with reduced time and cost, and increased throughput. These advancements enable comprehensive analysis of entire crops or individual plants, depending on the specific analysis requirements (66).

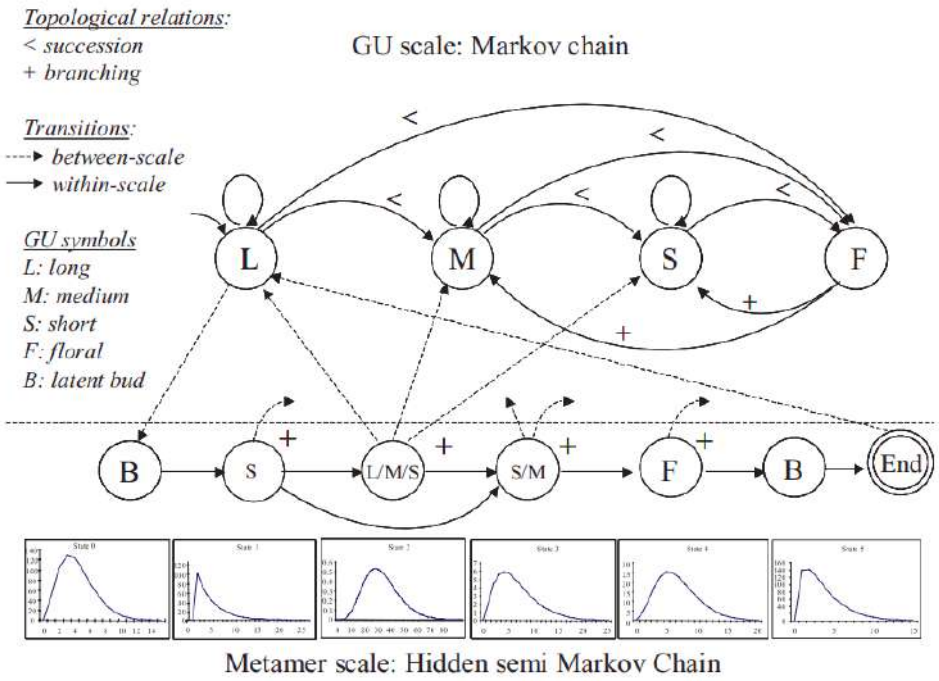


Figure 6 – Hierarchical stochastic model representing tree topology. Successions and branching between entities are represented by $<$ and $+$, respectively. At the growth unit (GU) scale, the succession of GUs along an axis is modeled by a 4-state Markov chain. The four 'macro-states' are long (L), medium (M), short (S), and flowering (F) GU. The long and medium macro-states activate HSMCs with between-scale transitions (dotted arrows). The figure shows only the activation from the long GU macro-state. HSMCs model the GU branching structure at the metamer scale, as a succession of zones with a specific composition of axillary GUs (for instance a mixture of long, medium, and short axillary GUs is observed in state 2). The HSMC ends with an artificial final state which gives control back to the pending macro-state. Likewise, for each axillary position, between-scale transitions give the control back to the macro-state model corresponding to the type of axillary GU generated by the HSMC (the figure illustrates only these transitions from HSMC state 2). Figure and caption from (57)

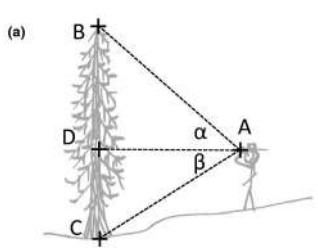
In the following subsections, the different types of technologies will be presented in detail.

2.4.1 Manual Digitizing

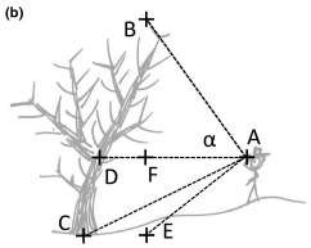
The manual digitization process of a plant consists of measuring and counting each of the plant's organs at different stages of growth in order to evaluate its development using computer algorithms or by generating 3D models of the plant from the measurements taken in the field. This process is carried out in different ways depending on the type, size of the plant, and objective of the study. In our case, we will focus on some important features for modeling a tree. These are the height of the tree, the radius of the crown, and the radius of the trunk at breast height.

For example, to measure the height of a tree, it is possible to start establishing a geo-

metrical relationship between the tree and an object of known size. Figures 7a, 7b, show the process proposed to measure the height of trees with a different topology.



(a) Manual measurement of the height of a vertical tree. Figure from (67)



(b) Manual measurement of the height of a leaning tree. Figure from (67)

Another example can be seen by measuring the crown radius of a tree. For this, a rope is taken, one person stands close to the trunk of the tree and another at the furthest point where the crown is visualized to reach as was shown by (68). This measurement is made in a specific direction ($N - S - E - W$).

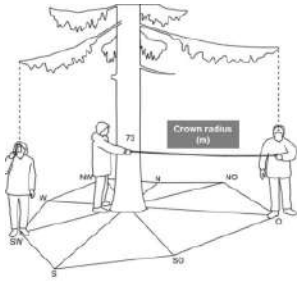
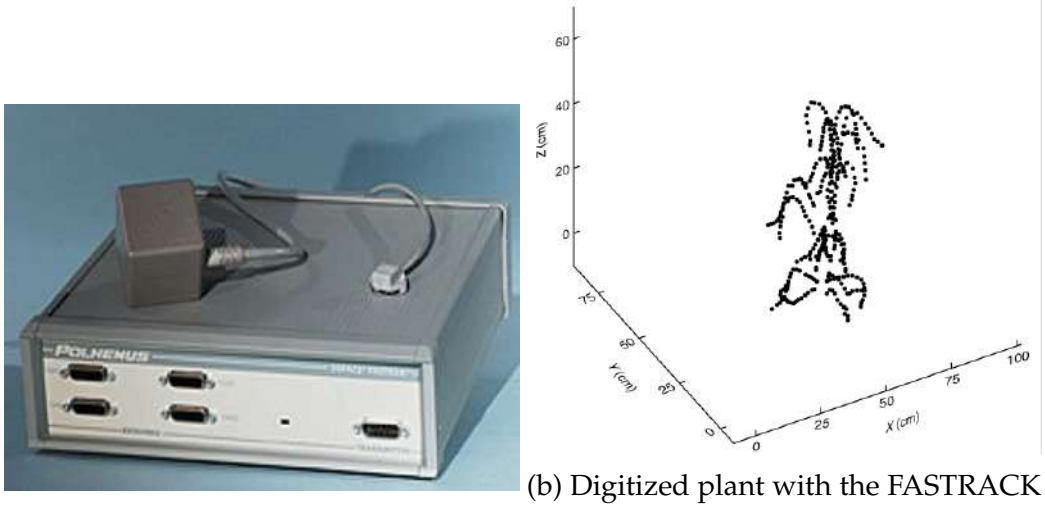


Figure 8 – Manual measurement of the crown of a tree. Figure from (68)

To measure the diameter of the trunk, a meter is placed along the trunk, and the measurement is taken (69). Another factor taken into account is the number of shoots along each branch of the tree. To count the shoots, the branch hierarchy and the number of shoots on each branch are considered. For measuring the size of leaves, a ruler is used to measure both the width and height of the leaf, as explained by (70). A faster and more precise approach to this task is using a planimeter. This method involves placing the leaf between a glass plate and a light-sensitive paper parallel to them. The planimeter captures the geometry, which can then be analyzed (71).

A magnetic sensor that can be used to obtain a 3D representation of the plant is the motion tracking sensor such as the *FASTRACK* (Polhemus, Colchester, VT, USA). The work presented by (72; 73) shows how this technology has enabled 3D digitization of the plant structure. To achieve this 3D representation of the plant, the system creates a magnetic field around the plant and then uses the sensor's pointer, to measure the coordinates of interest. Figure 9a shows the sensor, and Figure 9b shows an example of the digitized plant.

The FASTRACK sensor is very useful for obtaining a 3D representation of the plant topology and capturing the geometric intricacies of the plant. However, the measurement process



(a) FASTRACK sensor. Figure from (74)

(b) Digitized plant with the FASTRACK sensor. Figure from (73)

Figure 9 – FASTRACK sensor and an example of its usage to digitize a plant

using this technology is slow and can require several hours of work to complete the reconstruction of the targets.

Observing the way in which these processes are carried out, the time and effort required to digitize a single tree or plant is important. Resulting in important time and manpower consumption and a low number of individuals digitized.

2.4.2 Plant digitizing MRI-based

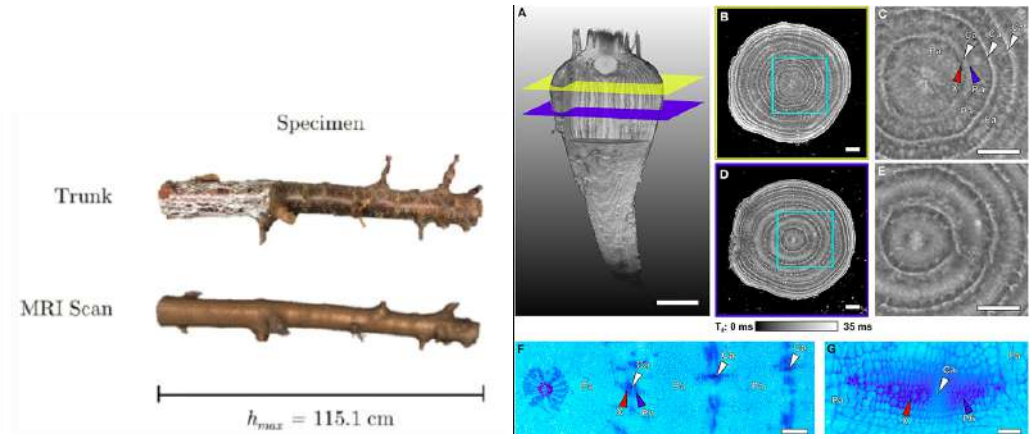
Another technology that enables the digitization and creation of 3D models of plants is MRI (Magnetic Resonance Imaging) sensors. MRI enables the acquisition of 3D images depicting the anatomy of the object of interest. In medicine, this technology serves as a valuable tool for disease detection, diagnosis, and treatment monitoring. Similarly, in plant science, MRI is utilized to obtain 3D models of the plant's internal and external geometry without causing harm to the plant. This capability enhances our understanding of the plant's architecture and physiology, leading to deeper insights (75).

MRI utilizes powerful magnets to generate a magnetic field that causes protons within the body to align with it. Subsequently, a radiofrequency pulse is emitted, which excites the protons and their spins. This generates a counterforce against the magnetic field. Once the radiofrequency pulse ceases, the sensor detects the alterations in the magnetic field resulting from the repositioning of the protons and spins, as well as the changes in magnetic field strength (76). The magnetic field variations also depend on the properties of the surroundings and the target being observed.

MRI technology has been employed in studying plants to characterize phyllotactic patterns in cherries (77). To detect the embolism formation in stems and their process of refilling (78). It has also been utilized to model the root systems of plants (79). It is important to highlight that in this specific case, MRI serves as a valuable tool that provides new perspectives,

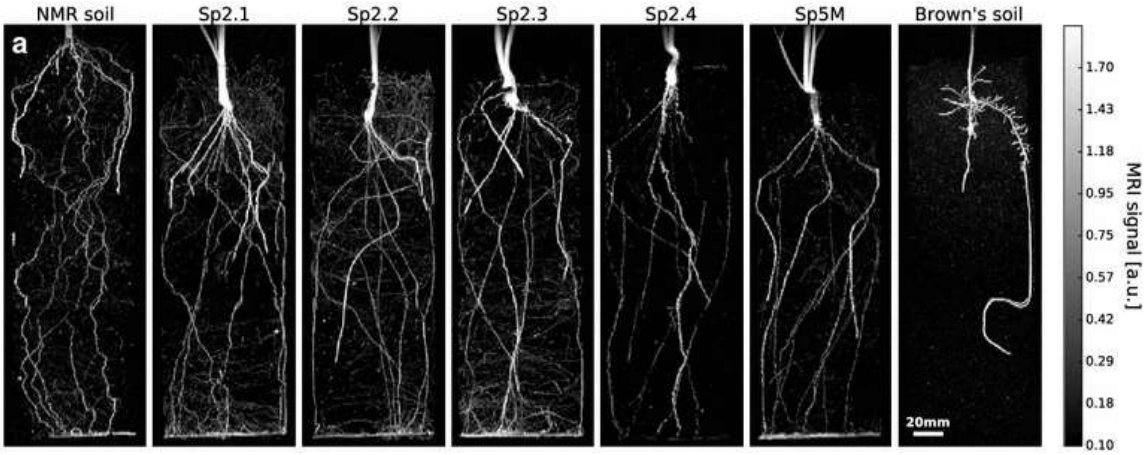
as it enables the analysis of both the root architecture, topology, and morphology (80) and the presence of specific components within the roots such as water content (81). Examples of the results obtained with these technologies can be seen in Figure 10.

The MRI technologies enable the acquisition of high-quality 3D representations that consider both external geometry and internal properties. However, they do have certain limitations, such as the high cost of the sensors, the time required for sampling, and the complexity of the software needed for image processing and reconstruction.



(a) Cherry's trunk reconstruction using MRI technologies. Figure from (77)

(b) Internal structure of the sugar beet. Figure from (80)



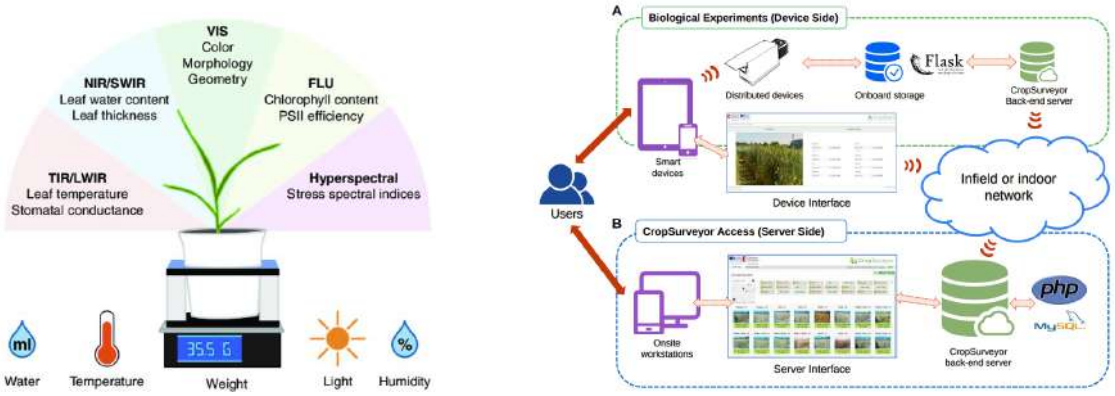
(c) Roots over different soil types. Figure from (79)

Figure 10 – Example of the Plant MRI images

2.4.3 IoT techologies

IoT stands for the acronym "Internet of Things." This technology encompasses all the devices and machines that are continuously connected to the Internet, transmitting information without requiring interaction. Therefore, sensors such as temperature, humidity, weight, light, and others, which are connected to the Internet and follow this working pattern, can be classified as *IoT* technologies. In Figure 11a, a general view is presented on how phenotyping is conducted in a controlled environment using *IoT*. Groups of sensors (organ tem-

perature, humidity, light, conductance, etc.) are attached to several plants to monitor their changes over time. Each embedded sensor measures specific environmental variables (humidity, light levels) or plant variables (temperature, branch or leaf elongation, conductance) and sends the data to a central database located in a local or global network, where it will be further processed and analyzed.



(a) General schema for the phenotyping in a platform. The figure was taken from (82)

(b) Field phenotyping with a IoT system. The figure was taken from (83)

Figure 11 – Example of an acquisition system and architecture of an IoT model for phenotyping.

Another example of sensors used in plant phenotyping are *stain sensors* are used to measure the growth rate of leaves and fruits. These sensors will be attached to the leaves or fruits and based on their growth, different levels of resistance will be monitored (84; 85). The *gas pressure sensor and the potometer* are used to measure the transpiration of leaves. The purpose of using these sensors is to estimate the rate of water loss from the plant due to evaporation (86; 87; 88). *Temperature sensors* are used in plants and crops to detect potential plant health problems and also to optimize the irrigation process. This variable can be used in conjunction with other measurements to select more resistant varieties. Examples of other sensors used to measure crop and plant temperature include thermocouples, infrared (IR) sensors, and thermal cameras (89; 90). *Light sensors* are used to measure the quality and quantity of light, these measurements are important because plants behave differently depending on them (91; 92). For example, in greenhouses, light measurements are made using two types of light sensors. The first is known as a global radiation sensor or pyrometer, and the second is known as a photosynthetic active radiation (PAR) sensor.

Figure 11b shows how the different technologies (electronic devices, databases, and communication protocols) interact with each other and what is the data flow. In this scheme, models, and technologies are specially developed for the capture and transmission of data, their storage, processing, and visualization.

Currently, this type of system has great value for the industry, agriculture, and the day-to-day life of people, since based on all the data captured, decision-making is facilitated and different processes are automated.

These types of sensors can provide highly accurate measurements of plant development over time and are therefore valuable for understanding plant behavior. However, it should be noted that, compared to other sensors such as cameras and LiDAR, the measurements of these sensors are limited to a single individual and their large-scale implementation involves different costs and difficulties, whereas cameras and LiDAR sensors allow high-quality representations of multiple individuals and the surrounding environment. Multiple metrics can be obtained from their measurements, allowing plants to be evaluated from different perspectives.

2.4.4 RGB, multispectral and hyperspectral cameras

The use of cameras for extracting plant characteristics has been extensively developed largely due to their capability to capture both visible and non-visible wavelengths (Figure 12), as well as complex scenes. Furthermore, processing the captured images allows to approximate segmentation and counting of various plant organs (such as fruits (93; 94; 95), leaves (96; 97), and branches (98; 99)), extraction of vegetation indices (100; 101; 102), and creation of 2D and 3D maps (103; 104; 105) for the desired crops or plant species of interest. It is important to note that each of these applications presents different challenges and limitations.

In the case of organ segmentation or detection, it is important to consider that, regardless of the protocol, due to the structure of the plants as well as the characteristics of the sensor, different levels of occlusion and other types of noise will be present in the image, making the task difficult to approach. On the other hand, it must be taken into account that the reflectance captured by the sensor will vary depending on the environment, so developing a model that takes into account all possible reflectance variations is a highly complex task. Consequently, new image representations (e.g. Vegetation indices, 3D point clouds) are employed to facilitate this task. However, even with these representations, the models do not fully generalize the segmentation and detection tasks.

Vegetative indices have been extensively utilized to complement segmentation and estimation tasks related to crop health status. Nonetheless, it has been observed that a combination of these indices is often necessary to approximate different tasks. Thus, finding a robust solution applicable to all crops across various scenarios provided by different terrains can be challenging.

In addition, both 2D and 3D maps can be generated from RGB and multispectral imagery. However, specific protocols must be followed to obtain these maps and their resolution or level of detail is limited by sensor resolution, protocol, and various environmental factors.

RGB cameras are those that capture the visible wavelengths where the full spectrum of the visible colors red, green, and blue are defined. Vegetative indices have been developed using these bands. These indices facilitate or highlight certain characteristics of the plant, or help to separate the plant of interest from the background in a good way. Examples of these indices can be found in the table 2.1.

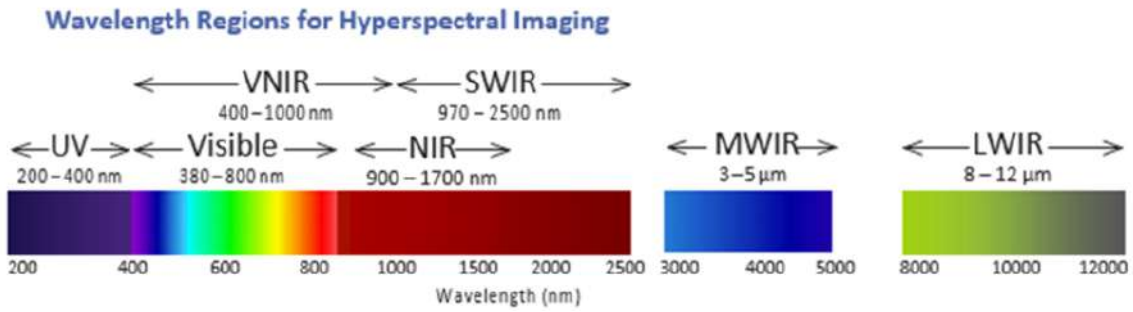


Figure 12 – Visible and non-visible wavelengths. Figure from (106)

Name	Formula	Reference
GLI	$\frac{2G-R-B}{2G+R+B}$	(107)
VARI	$\frac{G-R}{G+R-B}$	(108)
NGRDI	$\frac{G-R}{G+R}$	(109)

Table 2.1 – Vegetation indices created from the visible wavelength. The table was modified from the work of (110). Note that R is for red, G is for Green, and B is for Blue

The Green Leaf Vegetation Index (**GLI**) is used to measure the density of a green area. The working range of this index is $[-1, 1]$, negative values represent lifeless soil, and positive values represent vegetation. Initially, this index was developed to assess wheat covers (111; 107). The Visible Atmospherically Resistant Index (**VARI**) is used to measure different levels of vegetation in working areas that are not very sensitive to atmospheric effects. It has also been found to be sensitive to changes in chlorophyll and can be used to detect changes in biomass (108; 112). The Normalized green-red difference index (**NGRDI**) has been used to estimate biomass, nitrogen levels, and changes in different crops such as corn, soybean, and alfalfa (109; 113).

On the other hand, multispectral cameras allow to observe some non-visible wavelengths such as the NIR (Near-infrared light) and the red edge. These two wavelengths of light allow to characterize other physiological characteristics of the plant. And it is noteworthy that from these two non-visible wavelengths and in combination with the visible wavelengths, vegetative indices have been created with the purpose of discriminating the plant's health. Also, these indices allow the early detection of a particular stress, making it possible to take decisions to reduce the impact of the stress to be reduced. Examples of these indices can be seen in table 2.2

Normalized Difference Vegetation Index (**NDVI**) is an index used to evaluate the health status of the vegetation based on the relationship between **NIR** and red wavelengths. Its saturation depends on the soil type of the working area and the amount of vegetation. This index works in the range of $[-1, 1]$, where 1 indicates a high density of vegetation, 0 for areas with low or no vegetation, and -1 indicates that a water source, sand, grasslands is being targeted (119). The Soil Adjusted Vegetation Index (**SAVI**) is used to reduce soil brightness in soils with little vegetation. This index depends of a variable L. This variable is used as an

Name	Formula	Reference
NDVI	$\frac{NIR-R}{NIR+R}$	(114)
SAVI	$\frac{NIR-R}{NIR+R+L} + (1+L)$	(115)
SR	$\frac{NIR}{R}$	(116)
TVI	$\sqrt{\frac{NIR-R}{NIT+R}} + 0.5$	(117)

Table 2.2 – Vegetation Indices that use a combination of visible and non-visible wavelengths. The variables R, G, and B are related to the visible wavelengths of the colors Red, Green, and Blue. And the NIR is related to the non-visible values of the Near Infrared wavelength. A more complete list of vegetation indices derived from these wavelengths can be found in (118).

indicator of the amount of green in the image. For areas with little or no vegetation $L = 1$, in areas with moderate vegetation $L = 0.5$, and in areas with high vegetation $L = 0$. This index can be interpreted as a normalization of the **NDVI** (120; 121; 115). The Simple Ratio index (**SR**) is described as an indicator of the amount of vegetation in the area. Its working range is between [0 and 30]. Higher values indicate high vegetation density and lower values little or no vegetation (122; 116). The **TVI** is a modification of **NDVI** that avoids most of the negative values.

Hyperspectral cameras are capable of capturing a large number of wavelengths compared to RGB and multispectral cameras. With such a large variety of wavelengths, it is possible to estimate or relate different plant indices or reflectances to different plant behavior or organs. Figure 13 shows the comparison between these three types of technologies.

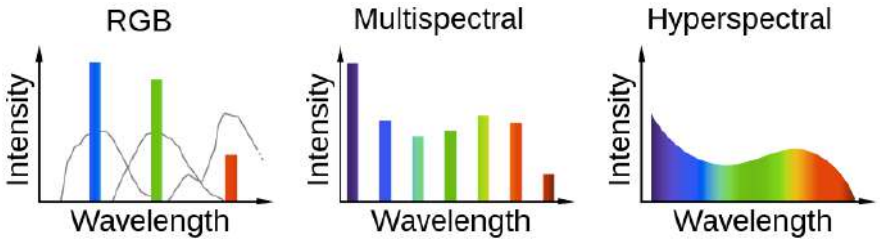


Figure 13 – Comparison between the different wavelength sampling done by a RGB camera, Multi-spectral camera, and a Hyper-spectral Camera. Figure from (123)

Vegetative reflectance is a distinctive characteristic of vegetation. Reflectance values and their distribution along the image highlight the health status of the plant and its geometry. Several studies have been carried out to take advantage of these characteristics. The developed studies mainly focus on relating the reflectance or the vegetation indices with variables like crop biomass (124), nitrogen (125), and chlorophyll levels (126). The used images could be aerial, terrestrial, or satellite imagery, depending on the spatial and temporal resolution needed for the study. It should be noted that the studies of these variables are mainly focused on finding which indices are closely related to specific field measurements. The process followed to achieve this comparison consists of a few generic steps. First, eliminate the soil and other undesired elements from the image. Second, associate selected groups of pixels

with a measurement in the field. Finally, the generation of a regression model, to find the relationship between the input indices and the desired crop variable.

In addition to the estimates that can be made from reflectance, it is also possible to perform a geometrical analysis from the plant image. This analysis includes the estimation of geometrical indices such as area, convexhull, convexity, and height, among others (127; 128); and more precise tasks as the segmentation and detection (129; 130), counting (131), and hierarchy estimation (132) of the different plant's organs.

To perform the previous analyses, when starting from RGB images, various color spaces such as HSV, HSL, YSV, etc., can be utilized to enhance the elements of interest. Subsequently, a threshold can be applied to one or a combination of channels to obtain a binary image (133; 134). Following this, filtering based on morphological operations (dilation or erosion) can be applied, and finally, a contour analysis can be performed to count or analyze the geometry of the segmented element (135).

From an artificial intelligence perspective, numerous deep learning methods have been utilized or developed for fruit counting and yield estimation (136; 137), as well as for segmenting or analyzing other tree organs, enabling architecture analysis (138; 139; 140; 99). Deep learning models are commonly employed to segment or detect these organs, while the estimation of expected variables or indices is typically performed through post-processing of the masks or alternatively generated representations.

Based on the information above, it can be concluded that RGB, multispectral, and hyperspectral images enable the detection of tree organs such as fruits and crowns. They also allow for the measurement of organ development. However, it is important to consider that the accuracy of these methods varies depending on environmental factors, scene complexity, and the specific task at hand.

As previously mentioned, relying solely on 2D representations may not be sufficient to achieve the desired objectives. Therefore, new representations are created from the 2D images to enhance the analysis. In many cases, 3D representations are utilized to complement the analysis of 2D images, for instance, in segmenting branches.

Furthermore, plant reflectance provides valuable information for measuring different crop variables and detecting various types of stress. It is important to note that each plant species will have a more accurate representation of its status through different vegetation indices. Additionally, challenges arise due to environmental factors such as sunlight and humidity, which can impact the robustness of vegetation indices. Both classical image processing techniques and artificial intelligence methods can be employed to detect elements of interest in the images

2.4.4.1 Photogrametry

Photogrammetry is defined as the science of obtaining, processing and interpreting terrestrial, aerial, and satellite images to estimate 3D information. In order to obtain accurate mea-

surements of 3D objects that are located on a given terrain or work area (141). This science is based on the notion of triangulation. This means that 2 or more images of the same object with a percentage of view overlap are used to estimate a 3D coordinate. To estimate these 3D coordinates sight-lines are used as illustrated in figure 14. The sight-lines are defined as the intersection of n different light beams from which a 3D coordinate can be estimated (142). The matrix of intrinsic parameters **K** must be first estimated in order to ensure the correct relationship between the real plane and the camera plane, see equation 2.1. This equation is defined by the focal length (f_x,f_y), the principal point of the camera (x_0, y_0), and the distortion in the projection of the image s. This equation allows us to correct and approximate the 3D points that are common between the **n** images (143).

$$\mathbf{K} = \begin{bmatrix} f_x & s & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.1}$$

The focal length (f_x,f_y) is defined as the distance between the focus and the camera sensor, the principal point of the camera (x_0, y_0) indicates where the center of the sensor is located, and s is the distortion in the projection of the image.

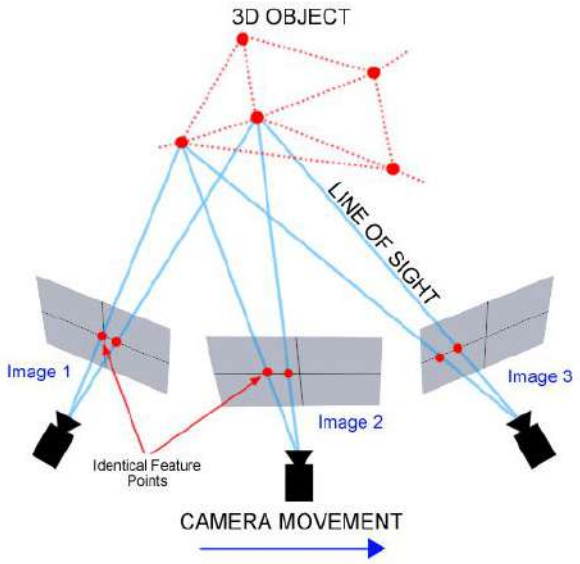
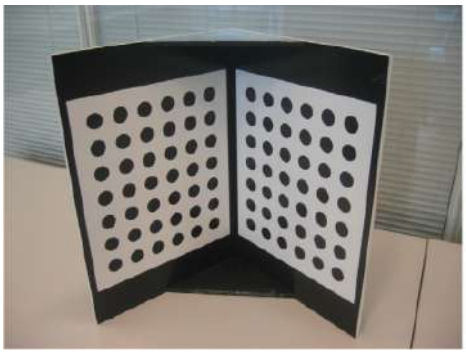


Figure 14 – Geometric relation between several images to estimate a 3D point of space and the representation of the sigh-lines. The images was taken from (144)

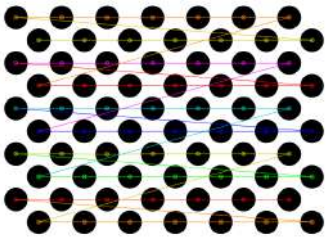
To estimate the intrinsic camera parameter matrix (**K**), predefined visual patterns are used, these patterns can be 3D, 2D, or 1D. In the case of the 3D pattern (Figure 15a), the 3D coordinates of the white edges of the tiles are known relative to the top left edge. By establishing the correspondence between the 3D coordinates and the 2D points of the pattern, the parameter matrix **K** can be estimated (145). For the case of the 2D pattern, usually, the checkerboard pattern (Figure 15b) and the dots pattern (Figure 15c) are used. The protocol consists of fixing the camera in a specific position and taking several photos of the selected

pattern in different positions and orientations. For the chessboard pattern at least 2 photos are expected and for the dots patterns at least 10 photos are expected (146). In the case of the chessboard pattern, the points where the edges of two or more squares coincide are detected. In the case of the 2D dots pattern, the center of each circumference is detected. Once the points of interest of each image are detected, the orientation of each checkerboard is estimated and it is proceeded to perform the geometric analysis of the shots to estimate the values of the intrinsic camera matrix. For the 1D case, a series of aligned points at fixed and known distances are used (Figure 15d). The camera is fixed in a specific position and several pictures are taken while moving and changing the orientation of the pattern. The points are then detected and the corresponding geometric relationships are made to estimate the desired values.

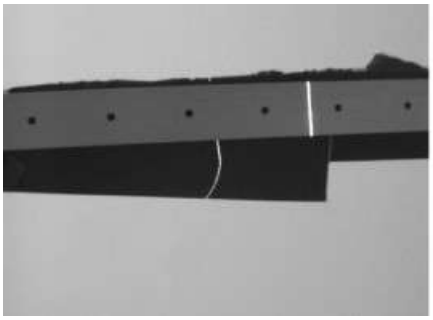


(a) 3D calibration pattern. The figure was taken from (145)

(b) 2D Calibration chessboard pattern. The figure was taken from (146)



(c) 2D Calibration dot pattern. The figure was taken from (146)



(d) 1D Calibration pattern. The figure was taken from (147)

Figure 15 – Camera calibration patterns

Knowing the **K** matrix, ensuring a percentage of overlap between images, it is possible to perform various tasks in the analysis of 3D geometries. Some applications in this field can be seen in archaeology, forensic sciences, forestry, and cartography, among others. Photogrammetry allows studying with precision the different objects found in the scene or how an area of interest changes over time. Different representations of the scene can be obtained from the images such as 2D maps or orthomosaic, point clouds, and elevation models. All these projections of the scene can be under an absolute or relative coordinate system depending on the application and the measurement protocol used. For example, in the cases

of archaeology, mapping, or precision agriculture, aerial images taken by UAVs or other manned aircraft are used. Before doing the flights, ground control points (GCPs) must be set in the work area and then a series of projections are made between the pixels and the GPS coordinates of each of the GCPs. This will allow to accurately measure the working area and the objects of interest. These projections are generated using interpolations constructed using polynomials of various orders..

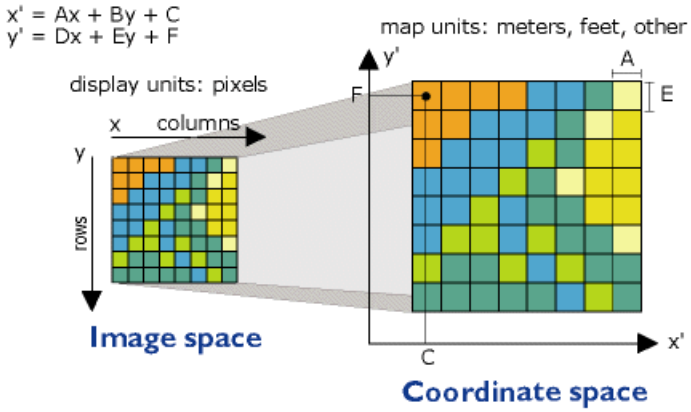


Figure 16 – Ratio between the actual working area and the pixels of the image that represents it. In this case, x and y represent the coordinates of the pixels in the image; x' and y' are the actual coordinates (e.g.: GPS). A and E represent the real width and height that represent each pixel in meters (m) or centimeters cm. B and D are rotation terms, C and F represent the center of the initial pixel. The image was taken from (148)

Depending on the size and complexity of the working area, polynomials of different order must be used. In figure 16 a polynomial of order 1 is shown. This polynomial can be useful for working areas smaller than $1Km$ because it is considered that at this distance the curvature of the earth is not visible and therefore it is assumed that such a linear relationship can be made. Polynomials of orders 2 and 3 are used to estimate areas of greater complexity (mountainous areas, areas larger than 1 km where the curvature of the earth or variations in its geometry are present). It is noted that for a polynomial of degree 1, at least 3 GCPs are needed, for a polynomial of degree 2 at least 6 GCPs are needed and for a polynomial of degree 3, at least 10 GCPs are needed.

Another interesting application is presented by (149). In his work, it is shown how from aerial images it is possible to make a 3D reconstruction of the crop. From this 3D representation of the crop, the height H, the width W, and the volume V are estimated. To make an evaluation of the canopy and to ensure the management and sustainability of the orchard. The pipeline defined in this work can be seen in figure 17. The drone *Parrot Bluegrass* and the *Parrot sequoia* camera were used to capture the images. The *Pix4Dcapture* software was used to delimit the flight area and the trajectory of the drone, i.e. speed, altitude, and control points were defined. After image capture, the *Pix4Dmapper* was used to obtain the characteristics of the images, calculate the coincidence between the images, and estimate the mosaic and the point cloud.

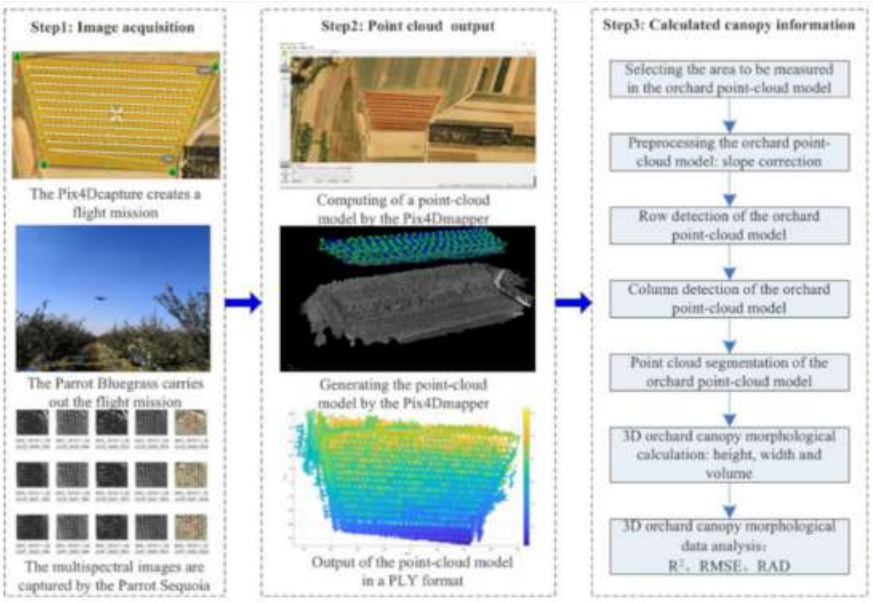


Figure 17 – Photogrammetry system for the evaluation of an apple orchard. The image was taken from (149)

To estimate the indices from the point clouds the equations of table 2.3 were used. In the equations, the estimations were done by operating directly on the points.

Index Name	Formula
Height	$H = Z_{max} - Z_{min}$
Width	$W = \max(\sum_{i=1-m}^{j=i+1-m}(\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}))$
Canopy projected area	$S_{xoy} = \frac{1}{2}\sum(x_i y_{i+1} - y_i x_{i+1})$
Radius	$r = \sqrt{S_{xoy}/\pi}$
Volume	$V = \frac{4}{3}\pi\frac{H_0.6}{2}r^2$

Table 2.3 – Definition of the volumetric indices made by (149)
all

To conclude photogrammetry is an indispensable tool for resource management and for estimating the evolution of an area or object of interest. It is important to see that there are currently a large number of tools to obtain the desired information or models, but these are restricted to laboratories and companies due to their costs. An important part of this process is to clearly define the measurement protocol, the desired quality, and the tools available for the processing of these photographs. Furthermore, it should be noted that the density and quality of the generated point clouds depend on the resolution and overlap of the individual images. This can lead to complications in the measurement and definition of the protocol.

2.4.5 LiDAR

The *LiDAR* sensor (*Light detection and ranging*) is a sensor that emits a light pulse from a laser at a specific wavelength. This sensor measures the time of flight of this pulse and allows to

obtain the 3D position of a point in space. Figure 18 shows the main schema of *LiDAR*'s working principle. By emitting thousands of these light pulses in different directions, this sensor makes it possible to get a point-based 3D representation of the environment or object of interest. This is achieved because the sensor system determines the orientation of the emitting and receiving diode at each moment. Consequently, during each sampling, it becomes possible to estimate the relative position of each point by considering the orientation at the time the laser beam is emitted. The resolution of the reconstructed object also depends on the sampling rate of the sensor, the field of view, and the measurement protocol used.

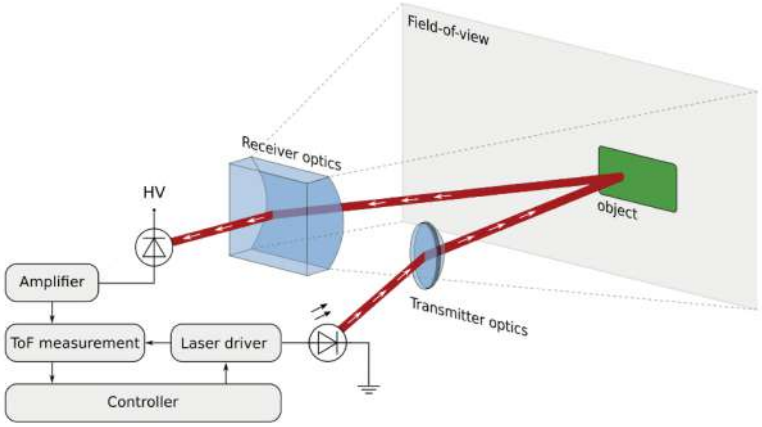


Figure 18 – Simplified illustration of the LiDAR sensor operational principle. Image was taken from (150)

LiDAR's are mainly divided into two main working categories based on their working principle, multi-beam *LiDAR*, and solid-state *LiDAR* (151). The solid-state *LiDAR* is a sensor that usually has a single laser beam and at the time of flight, a 3D point cloud starts to be generated with the different points captured, this sensor is characterized by having few or no moving parts and a reduced field of view. The multi-beam *LiDAR* sensor or *LiDAR* scanner can be categorized into two capture modes. The first capture mode could be described as a single laser beam directed in different directions on the same XY plane. An example of such a design is shown in Figure 19. In this diagram, the laser beam is directed at a mirror rotated at an angle of 45° relative to the laser frame, and this mirror is attached to the axis of a motor.

The second capture mode is defined by many laser beams in a vertical matrix pointing at different angles and mounted in a 360° rotating platform. This sensor is characterized by its wide viewing range (360°) and the need for constant maintenance (153). Figure 20 shows the described schema.

Depending on the number of wavelength bands used, *LiDAR* can be classified as single-band or multispectral. Single-band LiDAR generates point clouds using a single wavelength, while multispectral LiDAR generates point clouds using different laser beams at different wavelengths.

Depending on the wavelength and application, LiDAR can be classified as topographic

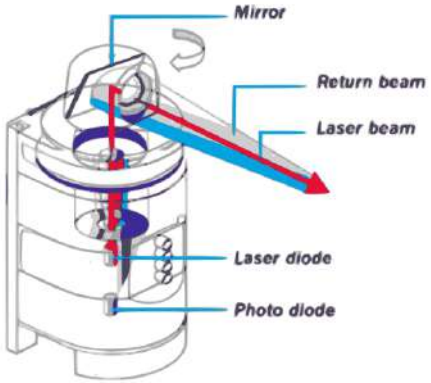


Figure 19 – 2D multi-bean LiDAR mechanical schema. Figure was taken from (152)

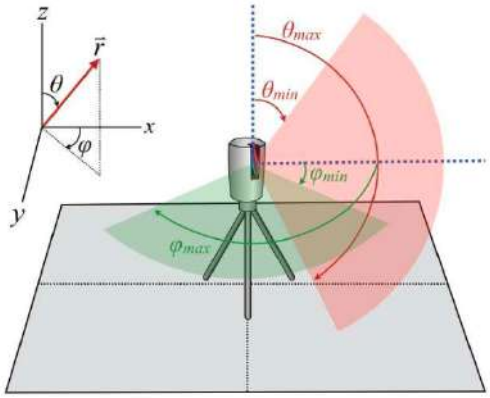


Figure 20 – Scaning pattern in spherical coordinates. Zenith and azimuth angle representation. Figure was taken from (154)

or bathymetric. Topography involves creating graphical representations of land surfaces, and topographic LiDARs specialize in mapping land surfaces. These LiDARs typically utilize a laser beam in the infrared wavelength to achieve their objective. Bathymetry, on the other hand, focuses on studying the soils of water bodies such as seas, lakes, and rivers. A LiDAR classified as bathymetric is capable of mapping the surfaces of water bodies. To achieve this, bathymetric LiDARs generally employ laser beams with wavelengths in the green spectrum, allowing them to penetrate through various water columns (155).

LiDAR sensors are highly dependent on their intrinsic and extrinsic parameters to provide accurate measurements (156). However, these parameters are only valid at the beginning of the sensor's lifetime. Over time, the mechanical and electronic components undergo changes that invalidate these parameters and cause the sensor's accuracy to degrade. To ensure accurate measurements throughout the life of the sensor, it is necessary to perform a sensor calibration. Calibrating a LiDAR sensor begins modeling its physical characteristics, the sensor is then adjusted or the measurements are corrected based on the estimated physical model (153; 157). The modeling process varies depending on the system used by each specific LiDAR sensor.

As demonstrated above, LiDAR technology offers a wide range of measurement options

depending on the type of environment and task. Additionally, the availability of highly accurate 3D representations has made LiDAR a valuable tool across various fields, including architecture, engineering, archaeology, plant science, among others.

In my particular case, I am going to focus on some of the applications in the agricultural and plant science fields. The LiDAR sensor is used in plant science to measure and estimate various plant properties (158). Examples of these properties are the canopy structure (159), canopy height (160), plant growth (161), fruit yield estimation (162), and others.

In order to make LiDAR measurements of a plant or crop, it is generally necessary to make different measurements over different views of the target. Different protocols can be proposed for this, such as the one proposed by (159; 163). The point clouds are then aligned to obtain a complete view of the target or scene of interest. Figure 21 shows the diagram of the general process followed to scan an area or object of interest with a LiDAR scan. Typically, once the point clouds have been aligned, the target geometry is approximated to perform the appropriate analysis.

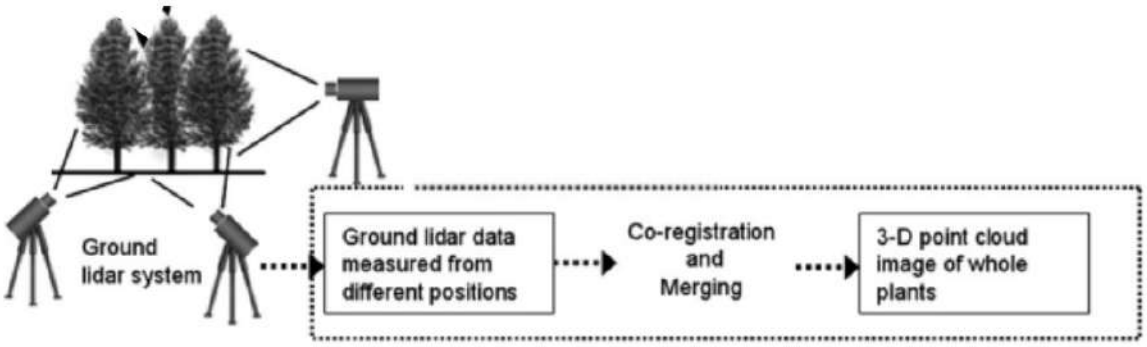


Figure 21 – Example of a measuring protocol to capture the 3D geometry of an area of interest. Figure was modified from (158)

The LiDAR sensor has been used for instance to segment and analyze sorghum panicles (164) and for the detection and geometrical analysis of apple fruits (165). In these two cases, the segmentation and detection algorithms use point density, geometry, and color as detection features. LiDAR point clouds have been also used for the architectural analysis of forests (166) and orchards (e.g. apple trees (167)). In these cases, algorithms re-express the point cloud of trees in simpler geometries as their geometrical skeleton or re-express the point cloud as a graph. The attributes of these representations are then used to find the desired features. LiDAR has also been used to evaluate the growth of various crops such as maize (168), cotton (161), and apple trees (169). For this analysis, 3D models are used to obtain variables such as Plant Area Index (PAI), Leaf Area Projection (PLA) and plant height.

LiDAR is an essential tool for efficient and precise phenotyping in both crop and forest applications. This is attributed to its ability to generate precise 3D geometries with high resolution (in millimeters). Compared to RGB images or point clouds obtained through photogrammetry, LiDAR enables an improvement in the quality of geometric representation by directly providing dense point clouds. Another major difference is that to create a point

cloud from LiDAR, only a single capture is needed, whereas photogrammetry requires a variety of views of the same target and a degree of overlap between them. However, it is important to note that the development of new algorithms to enhance the analysis of point clouds generated by this sensor is still ongoing.

2.5 Point Processing

To process point clouds obtained from LiDAR sensors, it is essential to transform the point clouds into structures that facilitate their processing. Additionally, extracting features at both the point and geometry levels is required. Definitions and applications that facilitate point cloud processing are presented below.

Point clouds can be defined as a multidimensional arrangement of points that describe a scene or an object of interest. Each element of the point cloud is typically composed of XYZ coordinates and may contain additional information such as color, intensity, reflectance, and other physical properties of the light beam.

A point cloud can be represented as a matrix ($\mathbf{A}$) of size ($M \times N$), where M is the number of sampled points and N is the number of features describing each point $\mathbf{a} \in \mathbf{A}$. The matrix $\mathbf{A}$ is variable in size, which means that the number of points describing a given scene will not always be the same and there is no order in the measured points. A schematic of such a representation can be seen in figure 22.

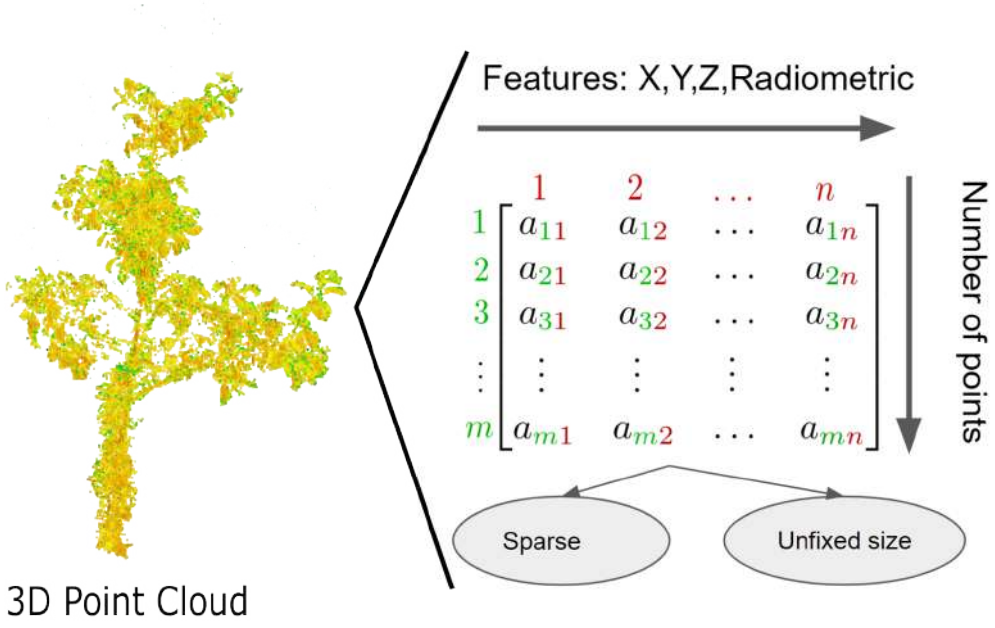


Figure 22 – Point cloud visualization and matrix representation. The different colors in this point cloud represent different levels of the captured reflectance

In the following sections, I present an overview of the tasks and processing techniques used to process and analyze point clouds.

2.5.1 Point Cloud Processing: Exploring Effective Data Structures

Point clouds are 3D or 2D representations of physical space that require significant storage capacity, ranging from megabytes to gigabytes or even terabytes, depending on the scan resolution and working area. They are also sparse and disordered. To handle these representations efficiently, special data structures such as (*Kd-trees* (170) or *octrees* (171)) are used to store and process them. These data structures allows to efficiently query (such as insert, delete and search) the point cloud (172).

2.5.1.1 KD-Trees

A k-dimensional tree, or k-d tree, is a data structure that partitions the k-dimensional space. This structure is widely used for performing tasks such as nearest neighbor searches and range queries, among others (170; 173). The creation of a *k-d tree* can be seen in Figure 23.

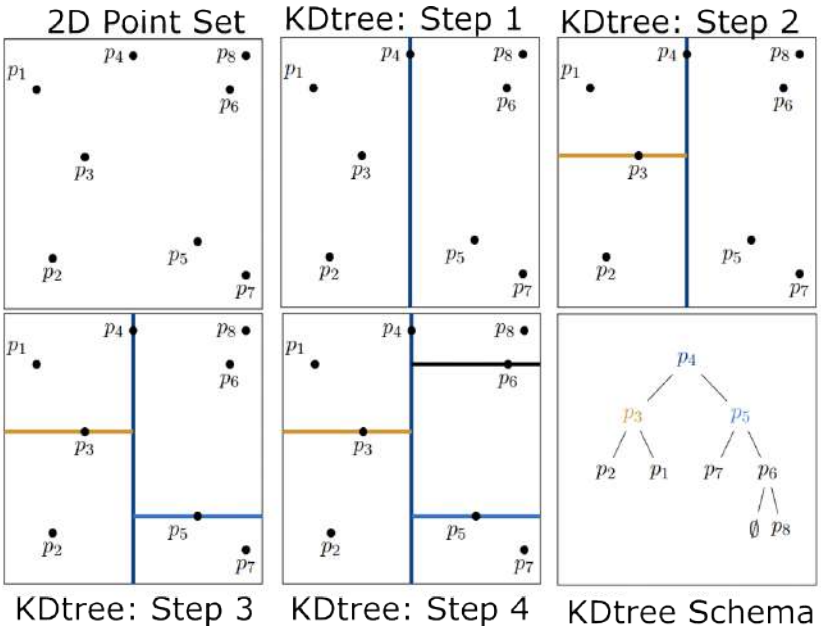


Figure 23 – A k-d tree is applied to a two-dimensional set of eight points. This demonstrates the step-by-step creation of the k-d tree structure. Figure was taken from (173)

In point cloud applications, *k-d tree* are often used to have an efficient way to measure distances between points or to find the **n** nearest neighbors of a point of interest. These tasks are essential for point feature extraction, point cloud filtering, geometry smoothing, point sampling, and point upsampling, among others.

As explained by (173), a *k-d tree* is defined for any finite set of points $\mathbf{P} = \{p_1, p_2, ..., p_n\}$, where $p_i = (p_i^1, p_i^2, ..., p_i^d)^T \in \mathbb{R}^s$ and i ranges from 1 to n. The k-d tree is empty if the data set $\mathbf{P}$ is empty. If the set contains only one point, the k-d tree will consist of a single node. For all $\mathbf{P}$ with a larger number of elements, the k-d tree is constructed by first selecting the dimension (d') with the highest dispersion.

In order to choose such a dimension, two points p_i and p_j which satisfy $|p_i^{d'} - p_j^{d'}| \geq |p_l^{d''} - q_m^{d''}|$ for all $d'' \in [d]$ and $l, m \in [n]$. After choosing the dimension d', the median of

p_i along d' is determined and the hyperplane $H = \{x \in \mathbb{R}^d | x^{d'} = q^{d'}\}$ is estimated. This hyperplane makes it possible to divide the data set into two subsets $P_1 = \{p_{i_1}, ..., p_{i_{n/2}}\}$ and $P_2 = \{p_{i_{(n/2)+1}}, ..., p_{i_n}\}$. The steps described above are applied recursively to each subset P_{i_n} to create a k-d tree. From the structure created, different queries can be executed to search and select points. After identifying one candidate, testing the distance to the hyperplane allows discarding rapid subsets of the point clouds.

2.5.1.2 Octrees

Octrees are a data structure used for spatial partitioning(174). An octree can be thought of as a collection of interconnected nodes. This structure is characterized by the fact that each node has eight children for 3D space. Within this structure, terminal nodes are called leaf nodes, which form the final component of the data structure(175). Each node contains a subdivision of a particular part space containing a group of points. The leaf node represents the final level of subdivision within the space. The process of subdividing the space can include as many sublevels as necessary. This data structure is created by an algorithm that evaluates the following steps.

1. Creation of the root node. This node represents the entire space containing the target geometry.

2. Divide the node into 8 nodes or octans. Mark as leaf node if the section of space contains 1 or 0 points within it.

3. Repeat the previous step until the desired space subdivision is achieved.

The graphical representation of the above steps can be found in figure 24. This type of structure is allows efficient processing (176; 177), visualization (178), and compression (179) of point clouds. The search for points in this type of data structure is a recursive search for the octant that encapsulates the point of interest at the deepest node of the structure. Efficient point cloud processing refers to the fact that this data structure allows efficient memory management, fast query handling, and simplicity of data structure (180). When talking about point cloud visualization, the goal is to display the target geometry in a virtual environment with the highest quality and lowest resource consumption (178).

Although this data structure is efficient for handling different types of data (e.g. point clouds) and performing different data-intensive tasks (e.g. particle-based fluid simulation), due to the large increase in data as well as the complexity of the problems, improvements and new applications for this algorithm are constantly being developed.

Improvements to this algorithm aim to reduce response time and resource consumption. To achieve this, algorithms using GPUs have been proposed (182). The octree creation algorithm itself has also been optimized and designed for other processing devices such as FPGAs (183).

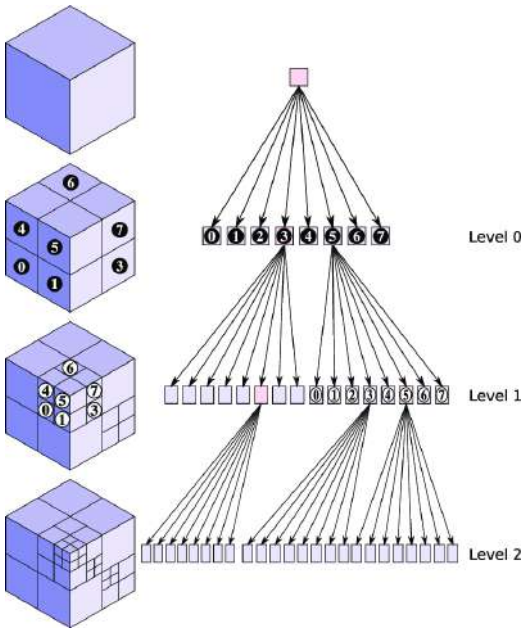


Figure 24 – Graphical Representation of Octree Creation. Figure from (181)

2.5.2 Geometrical processing

2.5.2.1 Skeletonization

Skeletonization is a fundamental technique that allows the reconstruction of biological and structural representations of trees (184). Skeletonization or thinning was initially defined by (185) as the medial loci of a geometry in the N dimensional space. This technique is applied with purpose of simplifying the representation of a target geometry and to facilitate tasks such as matching, registration, recognition and compression (186). The skeletons describe in a simplified way 3 properties of a shape, the geometry, the topology and the symmetry (187). Initially skeletons were applied to 2D signals with the aim of reducing the complexity of the data (185). Skeletonization have been well studied in 2D, several propositions have been developed. As explained by (188) the skeletonization algorithms can be divided into two main approaches, iterative and non-iterative algorithms. The iterative algorithms are divided into sequential and parallel, the main difference between these two approaches is that parallel algorithms consume less time and remove the undesired elements after identification, while in sequential algorithms the time of processing is higher and in general the algorithm identifies and removes the undesired elements of the studied signal in a specific order. The non-iterative algorithms don't study the signal and the skeleton is produced directly, Voronoi diagrams (189; 190) are an example of such techniques.

Skeletonization of 3D data is more complex than skeletonization of 2D data due to the richer information provided by the 3D representation. To obtain the skeletons of the 3D geometries, four critical problems must be addressed as described by (191). First, handling single points can become a complex task because it is difficult to define what is an end point and what is an unwanted point. Second, handling thin surfaces is a difficult task.

Sequential algorithms return one of the edges as a possible skeleton, not the central element. Parallel algorithms tend to stop after finding thin surfaces. Third, defining the hole in the geometry. Given the variety and characteristics of 3D geometry, it is difficult to establish a rule or set of rules that considers this definition in a generic way, and on the other hand, the handling of different levels of occlusion adds more complexity to the definition of the problem. Fourth, the definition of the end points can change the desired result, so depending on the application, defining such a parameter can be a difficult task.

Depending on the task, different skeleton definition can be approaches, example of such representation are surface skeletons (192), curve skeletons (193; 194) and centerlines. Surface skeletons refer to the 2D representation of the middle region of a 3D object (195). Curve skeletons refer to 1D representations within the core of a 3D object (193). A centerline is 1D Object represented by a set of curves embedded in a 3D space. The centerlines represent the symmetry of the axes (196).

In point clouds, the skeleton is defined by (197) as a graph containing the geometric information of the point cloud, including its vertices and edges, where the connection of three vertices represents a change in the topology of the object, and the edge represents a connection between them.

Several algorithms have been proposed for thinning or skeletonizing 3D geometries. In (198) the authors proposed a parallel algorithm of skeletonization based on the evaluation of 4 base templates (A,B,C,D), the templates are represented in Figure 25.

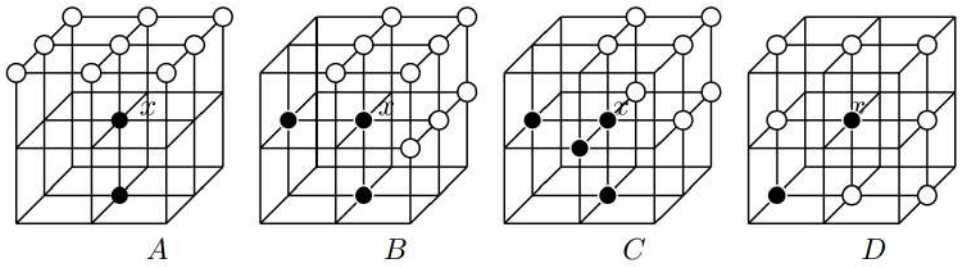


Figure 25 – Example of the templates proposed by (198)

From these templates, a group of deleting templates is created. The deleting templates are created by several rotations along the main axes of the templates. The evaluation of such templates consists in the iterative execution of a set of logical rules. The proposed rules evaluate the geometric relationship between points. The goal of such rules is to define whether a point should be deleted or not, the algorithm will stop when there is no more point to delete. (199) observed that the algorithm proposed by (198) doesn't preserve the connectivity between the elements of the geometry. To solve this problem 24 new deleting-templates were added to the list of the D base template. An example of the obtained results can be seen in Figure 26.

Lohou et al., (200) noticed that changes in the topology of the geometry are made by the proposed algorithm of (198) and even by the improvement made by (199). The topology

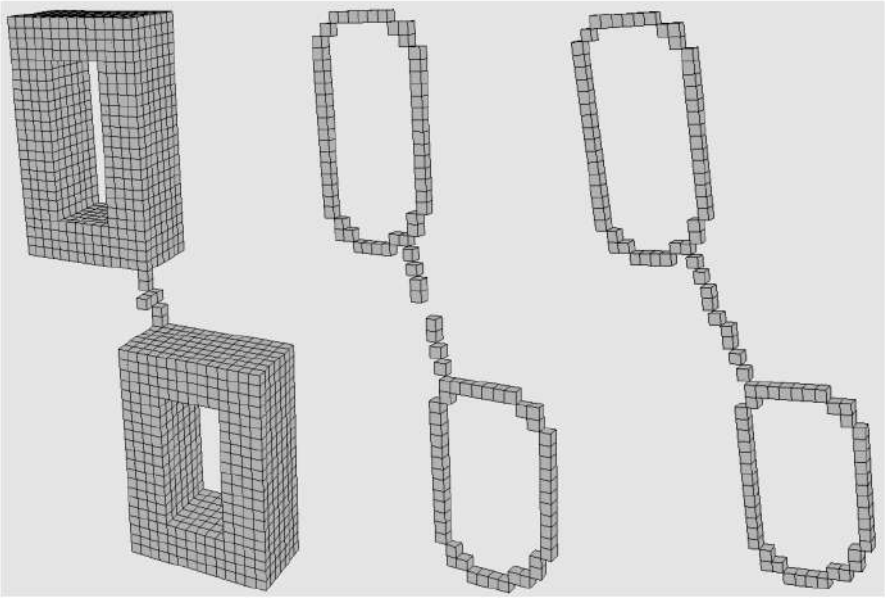


Figure 26 – a) Input geometry, b) Ma and Soka (198), c) Wang and Basu (199) Solution. Figure taken from (199)

modification came from the definition of the deletion templates from the D base template. To solve this problem, the algorithm *P-simple points* (201) was implemented. This algorithm in general will evaluate how much is deformed a geometry if a given point is deleted from a 3D window of size N. The improvements obtained after the implementations of such algorithm can be seen in Figure 27.

A sequential skeletonization algorithm was proposed by (197), the algorithm is called *CAMPINO* (Collapsing And Merging Procedures In Octrees-graphs). This algorithm is divided into 4 steps: First, an octree representing the point cloud is generated; Second, a graph is generated from the points representing each of the cells of the octrees; Third, the cycles are removed from the graph using a rule-based merging of the M-graph vertices and, finally, the M-new vertices and edges are obtained, these ones represent the union of all the octrees cells.

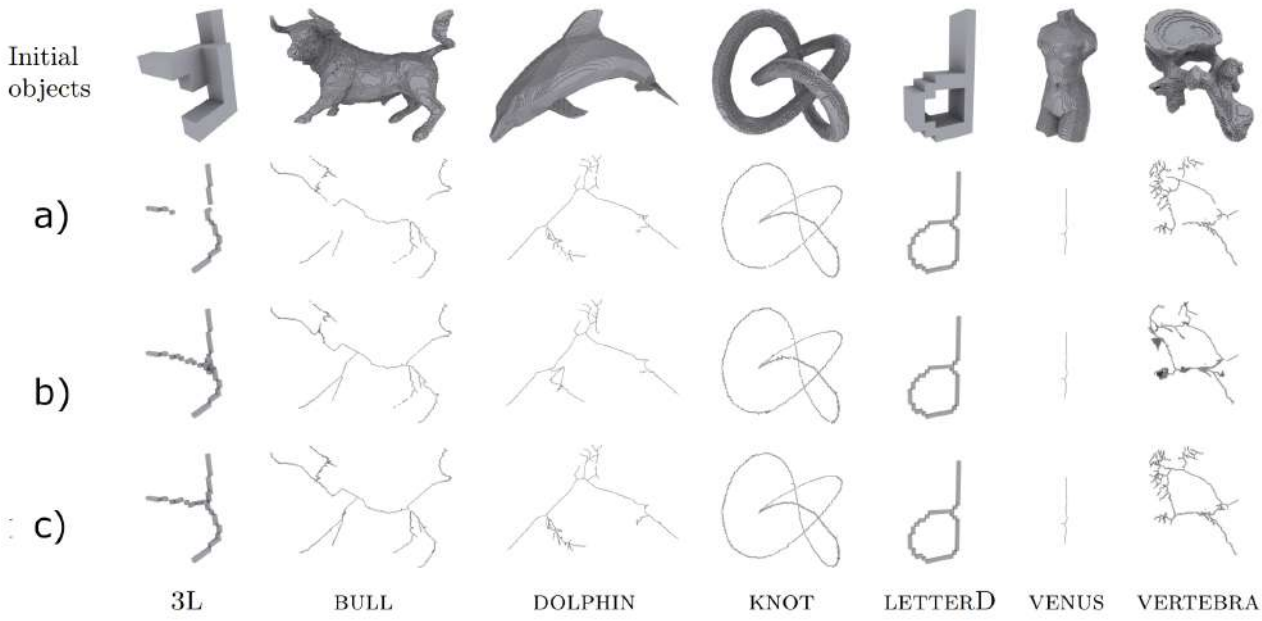
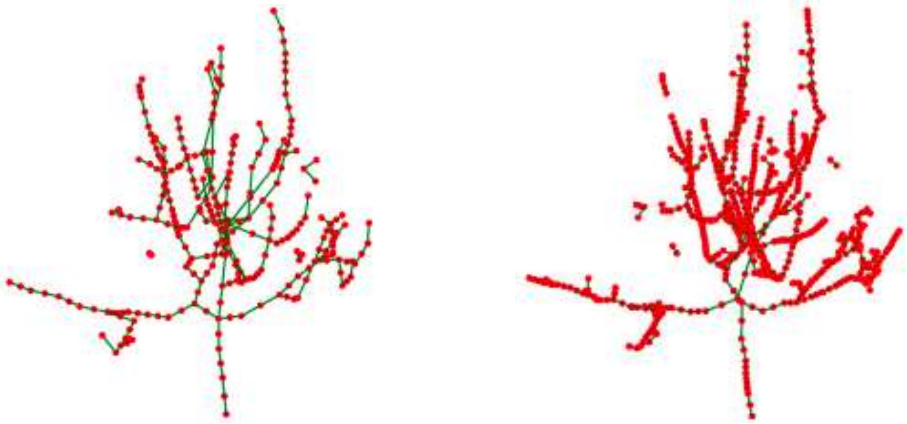


Figure 27 – a) Ma and Soka's (198), b) Wang and Basu (199), c) Lohou and Dehos (200). Figure taken from (200)



(a) Octree greed size 20cm. (b) Octree greed size 10cm.

Figure 28 – Example of skeletons generated by CAMPINO using different grid sizes to define the octrees. The figures were taken from (197)

Another way to estimate the skeleton of a point cloud is proposed by (202). In their research, they use the L_1 mean distance to estimate the arbitrary center of a set of points. An example of the application of this algorithm can be seen in figure 29.

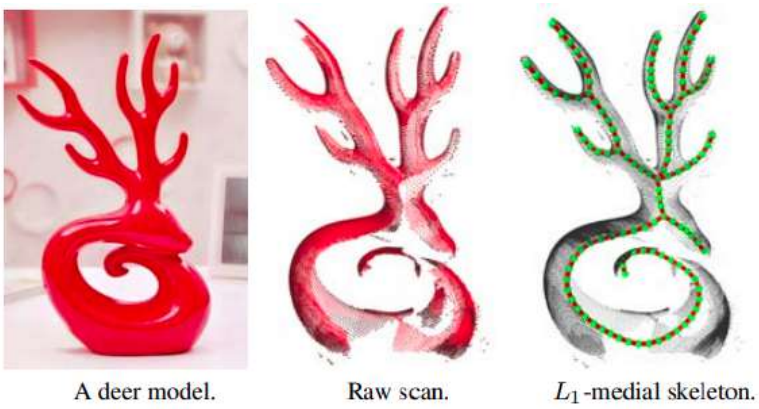


Figure 29 – Example of the application of the L1 medial skeleton algorithm to a point cloud. Figure from (202)

Skeletonization algorithms are very useful for the geometric analysis of point clouds. Furthermore, such algorithms can be classified according to the method they use to estimate the skeleton. Examples of such methods can be distance estimation, graph definition, and deep learning based. It is also an indispensable tool for the analysis of silverpoint clouds.

2.6 Machine Learning and Deep Learning for plant analysis

The development of new sensors and computerized systems has made it possible to enhance the quality and quantity of genotypic (such as SNPs, gene expression levels, and mitochondrial DNA, etc.) and phenotypic (such as leaf size, plant architecture, chlorophyll levels, etc) variables that can be measured throughout the growth of a plant or crop. However, as the amount of information increases, the analysis and processing tasks become more complex, thereby making it even more challenging to derive valuable insights from the data. At this point, machine learning and deep learning algorithms have become necessary tools for processing such large amounts of information (203; 204; 205; 206; 207).

Machine learning algorithms can be divided into supervised and unsupervised algorithms. Supervised learning algorithms are defined as all those algorithms that require the given features in a dataset to be annotated, examples of supervised learning algorithms are Support Vector Machines (SVMs), K-Nearest Neighbours (KNN), and Random Forest (RF). Unsupervised learning algorithms are all those that do not require the dataset to be annotated and seek to automatically discriminate the information, usually, these algorithms seek to develop clustering or feature extraction tasks. Examples of unsupervised learning algorithms are Principal Component Analysis (PCA), K-means, self-organized maps, etc. (208).

Support Vector Machines (SVM) is a machine learning model used for linear and nonlinear classification and regression problems. This model is focused on finding the best decision boundary, taking into account the largest possible classification margin. The classification margin is the distance between the decision boundary and the closest data points of each class, the closest points are also called support vectors. A graphical representation of

the decision boundary and the classification margin for the linear classification of two different classes can be seen in figure 30. When designing an SVM, a balance between keeping the classification margins as long as possible and their flexibility is needed in order to ensure a robust model (209).

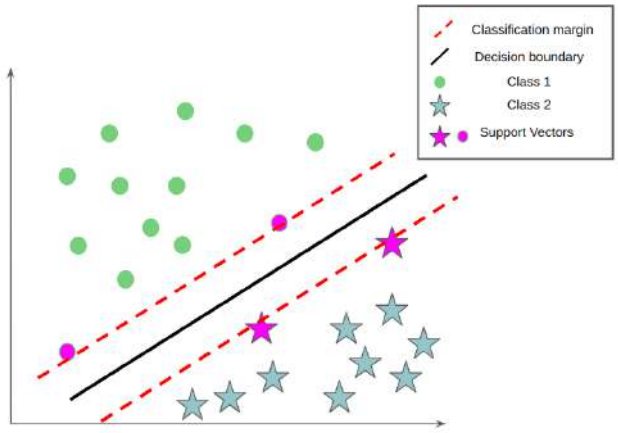


Figure 30 – Illustration of the decision boundary in feature space for the classification task

As explained by (210) in SVMs, the decision boundary is defined as $w_0 + \mathbf{w}^T\mathbf{x} = c$. From this equation, we have that $\mathbf{x}$ are the points representing the support vectors, $\mathbf{w}$ is a vector leading to the decision boundary, w_0 is an offset and c is the distance between $\mathbf{w}$ and the decision boundary. From this definition, it can be seen that every point $w_0 + \mathbf{w}^T\mathbf{x} > c$ will belong to one class and every point $w_0 + \mathbf{w}^T\mathbf{x} < c$ will belong to another class. The hyperplanes defining the classification margins are given by $w_0 + \mathbf{w}^T\mathbf{x_{pos}} = 1$ and $w_0 + \mathbf{w}^T\mathbf{x_{neg}} = -1$. To find the classification margin we subtract the hyperplanes that define it giving as a result $\mathbf{w}^T(\mathbf{x_{pos}} - \mathbf{x_{neg}}) = 2$. To normalize the above equation we will use the length of the vector $\mathbf{w}$ which is defined by $||\mathbf{w}|| = \sqrt{\sum_{j=1}^{m} w_j}$. Thus equation 2.2 will define the classification margin distance bounded by the hyperplanes.

$$\frac{\mathbf{w}^T(\mathbf{x}_{pos} - \mathbf{x}_{neg})}{||\mathbf{w}||} = \frac{2}{||\mathbf{w}||} \tag{2.2}$$

This is the distance to be maximized. It should be noted, however, that when optimizing this distance, one must take into account the constraint of correct element classification, which is defined by the 2.3 equation.

$$\begin{aligned} w_0 + \mathbf{w}^T\mathbf{x}^i \geq 1 \text{ if } y^i = 1 \\ w_0 + \mathbf{w}^T\mathbf{x}^i \leq -1 \text{ if } y^i = -1 \end{aligned} \tag{2.3}$$

Different methods extend the theory of SVMs, such as the implementation of kernels to work with non-linear spaces. A kernel in the case of SVMs is defined as a function that is responsible for transforming the low-dimensional space into a higher-dimensional space. A deep explanation of this topic can be found in (210; 209)

Another interesting model to classify data is Random Forest. Random Forest is a machine

learning model that consists of a set of N decision trees, This model is characterized by the fact that no preprocessing of the feature set is required, e.g. the features do not need to be scaled or centered (209). A decision tree is a hierarchical model that evaluates a set of logical rules at each of its nodes. An example of a decision tree is shown in Figure 31.

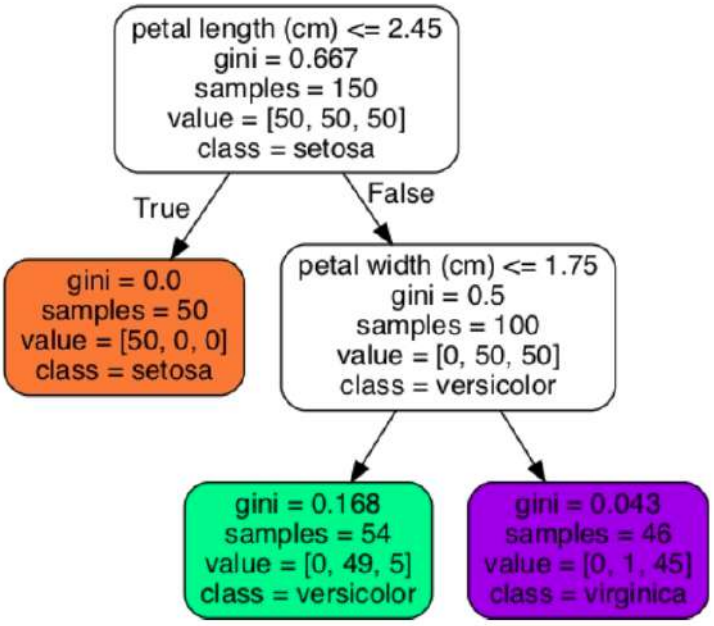


Figure 31 – Example of a decision tree generated from the Iris Dataset. Figure taken from (209).

Each node of the decision tree takes into account how many instances of a class it will contain and is evaluated by the Gini index. This index measures how homogeneous a node is, or in other words, how many different instances of each class the node contains. Gini is defined by equation 2.4. In this equation, $p_{i,k}$ is the ratio of the class of the k instances in the node i^{th}

$$G_i = 1 - \sum_{k=1}^{n} p_{i,k}^2 \tag{2.4}$$

Random forest is usually trained using the bagging technique (209). This methodology unites different models dedicated to the same task in order to create a more accurate system. The idea of such integration is to reduce the variance in the prediction and reduce overfitting, which is achieved by taking the predictions from each model and using them to make the final classification. To merge all the predictions into a single decision, majority voting is followed (210).

From unsupervised learning algorithms, it is important to highlight K-Means and DBSCAN because they are widely used in several fields to approach different problems (211; 212; 213; 214). K-Means is an algorithm that processes unlabelled data, this algorithm is responsible for clustering the data taking into account the expected N clusters. In general, this algorithm is responsible for finding the possible centers of each of the data blobs and then

assigning a label to the elements taking into account which center they are closest to. As presented by (210) this algorithm can be described in the following 4 steps:

1. Pick K points at random and set them as initial clusters.
2. Assign the points close to the proximate centroids μ^j, $j \in 1, ..., k$.
3. Move the centroids to the center of the assigned samples.
4. Repeat steps 2 and 3 until there is no displacement of the centroids.

The distance between the points and the center of the blob is given by the Squared Euclidean Distance. See equation 2.5. It follows from this equation that **x** and **y** will be two different points in N-dimensional space. j represents the j^{th} dimension.

$$d(\mathbf{x}, \mathbf{y})^2 = \sum_{j=1}^{m} (x_j - y_j) \mathbf{w}^{i,j} ||\mathbf{x} - \mathbf{y}||_2^2 \tag{2.5}$$

In the end, K-means is an optimization problem that seeks to minimize the sum of squared errors given by the equation 2.6 (210). In this equation, μ^j represents the centroid of the cluster j. **w** will be 1 if the actual point is in the cluster j otherwise it will be 0.

$$SSE = \sum_{i=1}^{n} \sum_{k}^{j=1} w^{i,j} ||\mathbf{x}^i - \mu^j||_2^2 \tag{2.6}$$

As an illustration of the use of such methods for plant science, machine learning algorithms have made it possible, for plant genomics, to predict the activity and function of different genes (215). The general structure for applying machine learning algorithms to plant genomics is shown in the figure 32.

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is a well-known clustering algorithm. The clusters of this algorithm are created then using the density of the points as the main reference. First, the density around each point (I) is estimted as the the number of N points at a defined lower than a radius (r). This points are marked as seed of the clusters. The seeds are extended with point lower than a distance to the initial point. Cluster are propagated recursivelly on the point set (216). From the above, it can be said that this algorithm will be able to separate the clusters when there is a low density of points in a given section of the point cloud. The operation of this algorithm will be defined by two hyperparameters mainly, the minimum number of points *MinPts* and the distance ε. An example of how the distance ε affects the clustering process can be seen in figure 33.

As described by (209) the steps describing the operation of this algorithm will be:

1. For each point, check how many neighbors have the same distance ε or lower. The region covered by such distance is called ε-neighborhood.

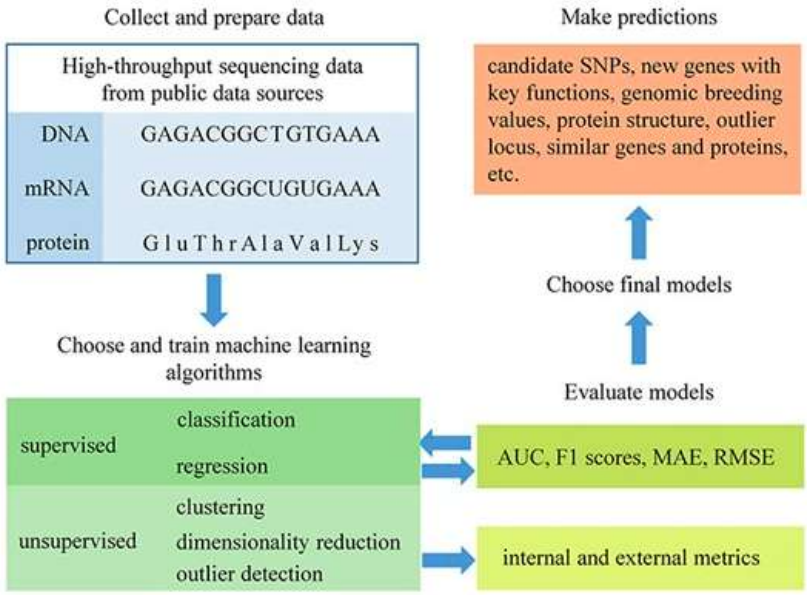


Figure 32 – General pipeline for the application of machine learning algorithms to plant genomics. Figure from (215)

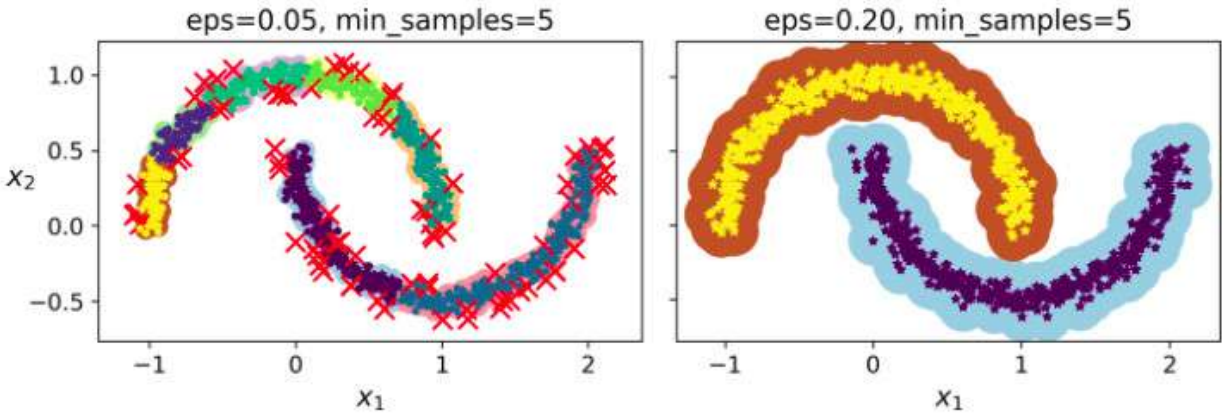


Figure 33 – Example of DBSCAN clustering changing the hyperparameter ε

2. If a number greater than or equal to *MinPts* is found within the evaluated distance ε, that point is considered to be one of the core points.

3. All points considered to be core points belong to a cluster. This means that areas of high density are considered as clusters.

4. Points that are not cores and have no cores in their neighborhood are considered anomalies.

Following a similar scheme as the one presented in Figure 32, SVMs have been used for example, for the efficient identification of DNase I hypersensitive (DHS). In detecting this element, SVMs achieved an accuracy of 87% and 85.79% respectively during the analysis of Arabidopsis thaliana and rice genes (217). Random forest (RF) has been widely used to approximate the processing of plant genomic data. The processing of such information is re-

lated to the tasks of prediction, classification, selection of variants, and genetic association, among others (218). In general, it has been observed that these models (SVMs, RF, Stochastic Gradient Boosting) allow finding and modeling more complex relationships between genomic information and provide an efficient approach for these studies (219).

Another interesting application of these algorithms is plant phenotyping using images and point clouds. When working with machine learning algorithms such as Random Forest, SVM, K-Means, there are 3 general steps that need to be followed, first the protocol and method of capturing the signal to be processed, for example, images or point clouds, then the characteristics of these signals need to be extracted, in the case of images they can be projected into different color spaces, or indices such as vegetation need to be estimated. In the case of point clouds, different types of point features can be used, such as Fast Point Feature Histograms (FPFH). Once such features are available, the model needs to be trained on such data. The training process must take into account the tuning of the hyperparameters and the evaluation of the model with the best possible data distribution, for which methods such as grid-search and K-Folds can be used (209; 210; 220). Figure 34 shows a clear diagram of the processes to be followed when implementing a machine learning model.

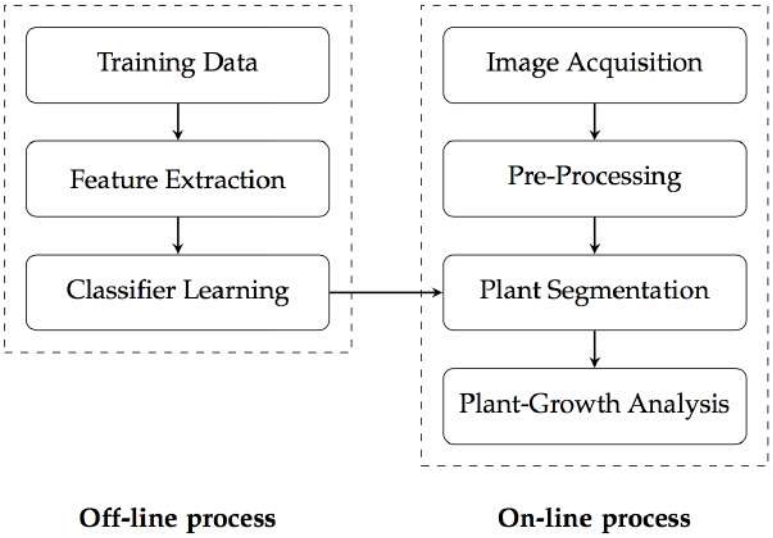


Figure 34 – Example of a pipeline for image segmentation with a machine learning model. Taken from (221)

Various machine learning algorithms such as SVM, Probabilistic Neural Networks, KNN, Random Forest, Fitness-Scaled Chaotic Artificial Bee Colony-based, etc. have been used for tasks such as fruit and vegetable identification and classification using both RGB and multispectral images (222; 223). These algorithms have made it possible to reduce the time and resources consumed in the different agricultural areas, broadly described as pre-harvest, harvest, and post-harvest (224). The main activities considered in each of these agricultural stages can be seen in the Figure 35. This type of algorithm has also allowed to development of the task of food quality and authenticity verification (225).

As we have seen, machine learning algorithms have made it possible to address, in a

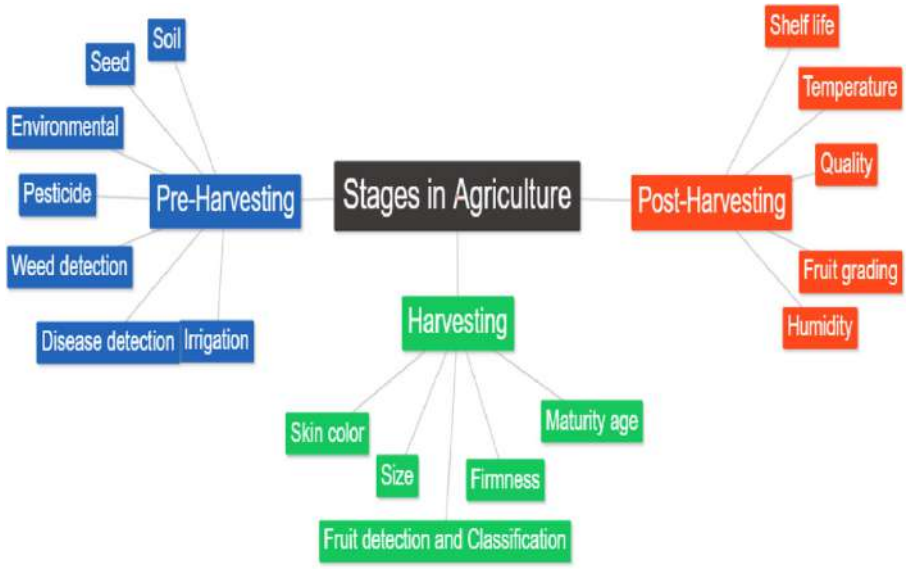


Figure 35 – Important parameters considered in each stage of farming. Figure and caption taken from (224)

certain way, a large number of problems related to the different stages of agriculture, and food production among others, but it should be noted that these algorithms are applicable to fixed cases, so it is not possible to say that they have generalized these tasks in their entirety. In the last decade, these algorithms have been surpassed by deep learning algorithms. First, Deep learning models are more flexible, meaning that their architecture can be easily adaptable to several problems. Second, their ability to generalize the problems is better. But in exchange for these skills, the models need a large amounts of data and hardware resources to be trained (226).

Deep learning is one of the sub-fields of artificial intelligence that focuses on the development of neural networks. Such models have the ability to make accurate decisions in high-complexity problems (226). These models are able to solve complex tasks as they are somewhat focused on abstracting and optimizing knowledge from the available datasets (images, 3D point clouds, audio, etc.).

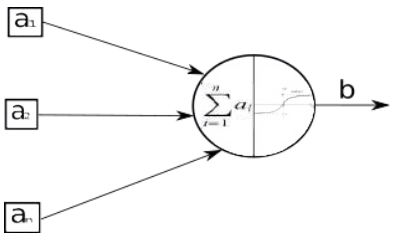


Figure 36 – Representation of the simplest synthetic neuron

In general, these models are composed of neurons interconnected with each other following different architectures. A neuron can be described in the simplest way, as a function that has **a** input parameters and an output value (b). The simplest schema of a neuron is shown in Figure 36. This neuron will perform two main operations inside of its definition.

First, it will operate the input values (applying the sum, max, or other operation to input arguments) and second, it will apply an activation function to the result of the output of the previous mathematical operation. The output of these operations is known as the activation value. The activation functions are important because they give the non-linearity factor to the neuron and define whether the neuron should be activated or not. There are three types of activation functions. First, *Linear activation or identity function*, this function is proportional to the inputs and can be seen as an addition function since it returns the sum of the input values. This function is given by the equation 2.7. It is important to note that because its derivative is constant, see equation 2.8, it is not possible to apply back-propagation and thus estimate the best learning weights for the solution of the problem. This type of activation is used in regression problems and it turns out that complex patterns cannot be modeled with it; Second, The *binary activation function* defines that the neuron should be activated if the input values exceed a specified threshold value and generally describes the binary classification tasks. The definition of this behavior can be seen in the equation 2.9;

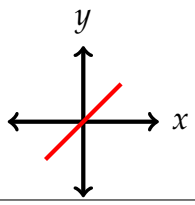
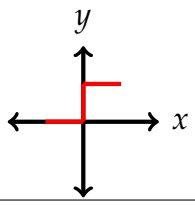
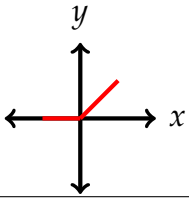
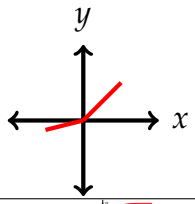
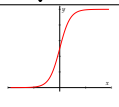
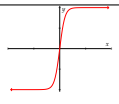
Name	Equation	Derivative	Range	Graphic
Linear	$f(x) = x$ (2.7)	$f'(x) = 1$ (2.8)	$-\infty, +\infty$	
Binary	$f(x) = \begin{cases} 0 \text{ for } x < 0 \\ 1 \text{ for } x \geq 0 \end{cases}$ (2.9)	$f'(x) = 0$	$-\infty, +\infty$	
Non-Linear				
ReLu (Rectified Linear Unit)	$f(x) = \begin{cases} 0 \text{ for } x < 0 \\ x \text{ for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 \text{ for } x < 0 \\ 1 \text{ for } x \geq 0 \end{cases}$	$0, +\infty$	
Leaky ReLu	$f(x) = \begin{cases} \alpha x \text{ for } x < 0 \\ x \text{ for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha \text{ for } x < 0 \\ 1 \text{ for } \geq 0 \end{cases}$	$-\infty, +\infty$	
Sigmoid	$f(x) = \frac{1}{1+\exp^{-x}}$	$f'(x) = f(x)(1 - f(x))$	$0, 1$	
tanh	$f(x) = \frac{2}{1+\exp^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$	$-1, 1$	

Table 2.4 – Types of activation functions

Finally, we have the *non-linear functions*. This group is composed of a wide variety of functions that have been developed or implemented in neural networks over the last few decades. Among the most used non-linear activation functions we can find the following:

- *ReLu*: The rectified linear unit (ReLU) is one of the most widely used activation functions in deep learning today. It is piecewise linear, meaning that it is composed of two linear parts. In general, a ReLU returns the input value if it is positive, and 0 otherwise (227).

- *Leaky Relu*: This function is presented as an improvement of ReLu, which tries to avoid the zero gradients for input x values less than zero. This allows all neurons to remain alive throughout the learning optimization process. It should be noted that this function contains the positive aspects of the ReLu function mentioned above (228).

- *Sigmoid*: This function is used in classification problems since it takes any integer number and returns a value between 0 and 1. It is clarified that the value returned by this function corresponds to the probability that the input data belongs to a given class (227).

- *Tanh*: The tanh function is a sigmoid function with a range of [-1, 1]. It is similar to the sigmoid function in that it saturates at the extremes, but it is more centered around zero. This makes it a good choice for activation functions in neural networks, as it can help to prevent the vanishing gradient problem (227; 209).

The architectures and definition of neurons have changed dramatically over the last few decades. Deep learning models started with the creation of convolutional networks, initially designed for image classification, as shown in the research of (229; 230). Following a close timeline, Recurrent Neural Networks (RNN) were developed to process sequential data [speech recognition, text summarisation, among others], this type of neural network was mainly characterized by maintaining a short-term memory. This means that this architecture allows the neural network to remember what was learned in a previous iteration. Examples of these models could be seen in the research of (231; 232). After this model, the LSTM (Long short-term memory) architecture was developed as an improvement on the RNN. And in recent years, we have seen how attention-based neurons, or better known as transformers, have improved the results obtained by LSTMs in natural language processing (NLP) tasks, and how they have provided the basis for large language models (LLMs) and for creating more efficient and robust models for image and point cloud processing (233; 234).

Convolutional Neural Networks (CNNs) are a specialized class of artificial neural networks designed to efficiently process structured arrays of data, particularly images. This architectural design is particularly well suited to handling the inherent lattice-like structure

of image data. In a typical CNN architecture, layers of neurons are organized in a hierarchical fashion, and each neuron performs the convolution operation on the input data using a set of N learnable filters or kernels. These filters act as local feature detectors, scanning different regions of the input data to extract meaningful information. What's remarkable about CNNs is their ability to capture features at multiple scales. As the depth of the network increases, lower layers tend to capture finer details such as edges and small textures, while deeper layers focus on more complex patterns and higher-level features. This hierarchical feature extraction process enables CNNs to recognize not only simple components of an image, but also complex, abstract concepts, making them incredibly powerful tools for tasks such as image classification, object detection, and face recognition (235; 236).

2.7 Deep learning for point clouds

As explained earlier, point clouds don't have a fixed size, and the points in the point cloud matrix don't have a specific geometric order. Therefore, the application of deep learning techniques used in image processing is not feasible. Then, to apply deep learning to point clouds on tasks such as object classification, tracking, and segmentation. It is necessary to create or apply techniques that handle these intrinsic behaviors of point clouds. A general view of the tasks and the approaches used to solve them can be seen in Figure 37.

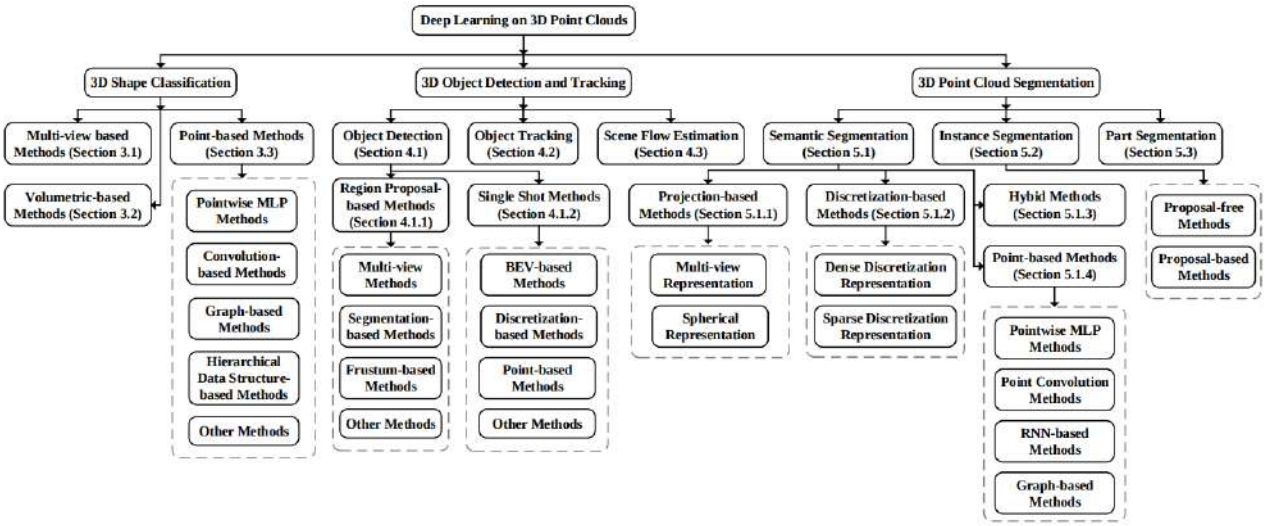


Figure 37 – Deep learning for the processing of 3D geometries. Figure from (237)

From the diagram shown in figure 37 I will focus on the point-based techniques. When we talk about methods based on direct point processing, we are talking about methods that are able to consume raw point clouds and create features from the relationship or distribution that exists between the points. Based on the technique used to make the feature extraction the methods can be mainly divided in pointwise MLP, convolution-based, and graph-based (237).

Pointwise MLP-based methods focus on using the neurons of the Multi-Layer Perceptron (MLP) model to perform feature extraction from the point cloud. This feature extraction consists of taking a set of point clouds and projecting them into a feature space using the MLP. These types of techniques attempt to describe the desired geometry by taking both local elements and opposite points in the geometry, for which the KDtrees and Farthest point sampling techniques are used, the most representative Deep learning models are (238; 239; 237).

Methods that focus on the use of convolution layers for feature extraction and task development approximate the solution by encapsulating the point cloud into voxels, thus creating a fixed-size matrix on which the convolution operation can be applied. Such models produce a feature matrix where each window of segments represents a specific portion of the geometries (240; 241; 237).

Graph-oriented methods are a sophisticated approach to handling point clouds. Instead of treating each point individually, these methods re-present the point cloud as a graph, where the points are nodes, and relationships between points are captured as edges. The underlying idea is to harness the inherent structural information within the point cloud and exploit it for various tasks. Once the point cloud has been transformed into a graph representation, a range of operations can be applied to this graph. One of the most common operations is graph convolutions. Much like their counterparts in image-based convolutional neural networks (CNNs), graph convolutions are employed to process and extract meaningful features from the graph (237).

2.7.1 GAN: Generative Adversarial Networks

Generative Adversarial Networks (GANs) are a powerful class of deep neural networks. GANs mainly consist of two main components: a *generator* and a *discriminator*. The generator generates new representations using the learned data distributions, while the discriminator classifies between real and fake samples. The GANs are trained in an adversarial way, which means that the generator and discriminator fight against each other. The generator tries to produce data that is indistinguishable from real data, and the discriminator tries to get better at distinguishing real from fake. This competitive training process leads to the improvement of both networks (227; 242).

In the context of point cloud generation and processing, several GAN models have been developed to approach tasks such as reconstruction(243; 244), visualization (245), and domain-adaptation (246). GAN models applied to 3D data can be divided into 2 main groups depending on how the networks consumes the 3D geometry. The first group considers the models that represent the point clouds as graphs. This representation has been widely used because it allows to overcome the unordered set of points(247), that is a intrinsic characteristic of the point clouds. To obtain the features representation of the graphs, the Graph-Convolution is used (247; 248; 249). The main difference between these models lies in the

structures that interlock the graph or graphs that represent a given geometry, approaches based on graph hierarchies (250; 251) and graph trees (252; 249) have been used. An example of the tree graph architecture can be seen in Figure 38.

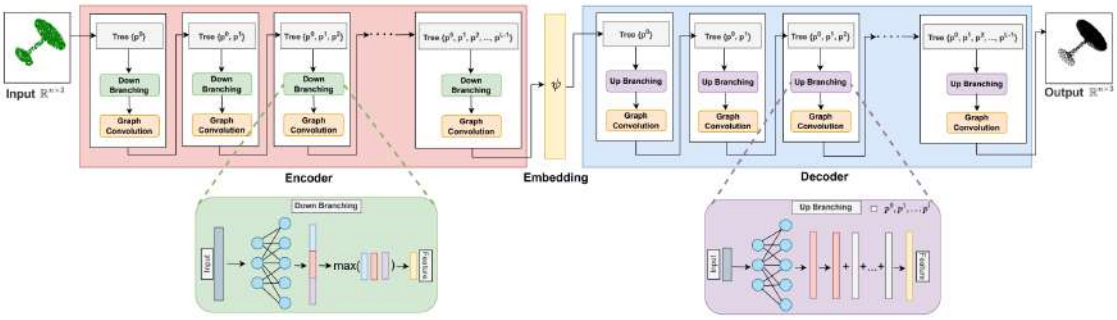


Figure 38 – Tree graph architecture. Figure taken from (249)

The second group contemplate the models known as point-GANs, these models are characterized for being able to consume or use raw point clouds. This type of GANs uses models like PointNET (238), PointNET++ (239) or similar to consume the point clouds. After obtaining the initial feature maps that represent the geometry, other architecture or layers are proposed to optimize the initial feature representation of the geometry (243; 253; 244; 245).

Examples of such architectures are the autoencoders. Autoencoders have also been used to develop GAN models (254). An autoencoder is a model capable of learning dense or latent representations of the input signal. These models usually consist of an element that encodes the input signal and another element that decodes it. The autoencoders are capable of performing tasks such as feature selection, dimensional reduction, and data generation (209). However, in the generation task, the autoencoders alone generate blurry data (227). Improvements on the autoencoders models have been achieved using the GAN architecture (255; 244). Figure 39 show the example of a GAN model that use a the T-NET from PointNET to consume the point cloud and get the initial feature abstraction, and later a encoder architecture is use to get a better feature map.

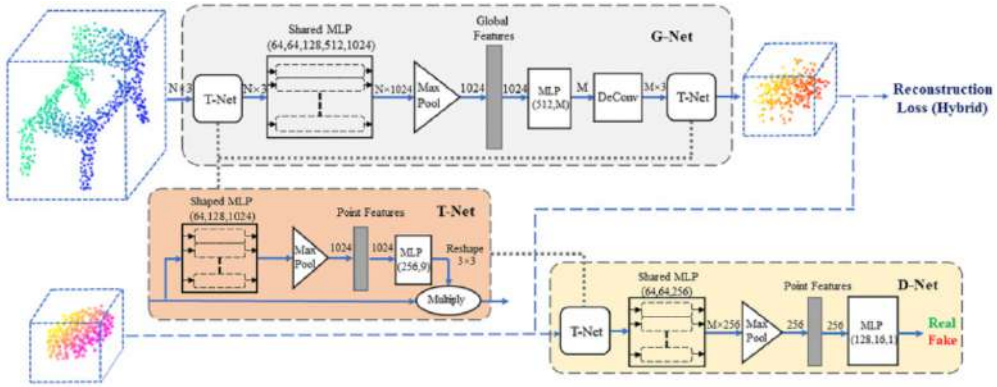


Figure 39 – Point Encoder GAN architecture (244)

GAN models have been mainly used for the tasks of reconstruction, generation, improve-

ment and visualization of geometries. But none of the models seen return annotated point clouds. In addition, there has been an interest from the community in developing models that directly consume the point clouds and do not approximate other types of geometry representations due to the resource consumption that such steps entail.

CHAPTER 3

Volumetric Indices for the characterization of fruit trees point clouds

In the context of climate change, the selection of fruit trees with appropriate shapes and sizes and with regular fruiting behavior is key to reduce the time and money dedicated to training and chemical thinning. It is important to consider architectural traits, such as plant structure and foliage distribution, to take into account the inherent production potential of different varieties (genotypes), their interactions with the environment (light, rain, insects, etc.), and ease of management.

In recent years, there has been considerable interest in using remote sensing techniques for high-throughput phenotyping of fruit tree traits (256; 257; 258). Remote sensing allows the analysis of large numbers of individuals and the extraction of many phenotypic variables with high precision. 3D acquisition systems based on Light Detection and Ranging (LiDAR) technology allow the rapid capture of tree shapes with exceptional accuracy. However, the characterization of architectural features from LiDAR data acquired in the field is dependent on wind conditions, tree morphology, and the scanning protocol used.

To accurately assess such traits in high throughput, we explored LiDAR technology on two types of vectors: terrestrial and airborne LiDAR. Our case study focuses on a non-commercial orchard comprising a large collection of French apple varieties. To analyze this orchard, airborne LiDAR measurements were performed using three different protocols and compared with terrestrial LiDAR measurements. This comparison aims to demonstrate which of the airborne protocols provides comparable quality to the terrestrial ones in estimating various architectural indices. It also highlights the advantages and disadvantages of these protocols for the estimated indices. The work described above is a continuation of the research carried out by (259). The text and results of this chapter were presented at the IHC2022 conference and published in the associated proceedings ((260)).

3.1 Material and methods

3.1.1 Plant material

This study was carried out on an Apple core collection of 241 genotypes. Each genotype was represented by 4 individuals. All genotypes were grafted on *M9 Pajam 2* rootstocks before being planted in February 2014 at the INRAE DiaScope experimental unit in Montpellier, France ($43°36N, 03°58E$). The trees were planted at $5 \times 2m$ spacing, irrigated by microsprayers between the trees, and left unpruned. The orchard ($1.2ha$) consists of ten rows of 100 trees, with two replications of two trees per genotype randomly distributed in the field. Due to technical constraints (autonomy of the UAV, presence of variety replicates), only 36 trees of 17 different varieties were scanned with the 3 different aerial and terrestrial protocols.

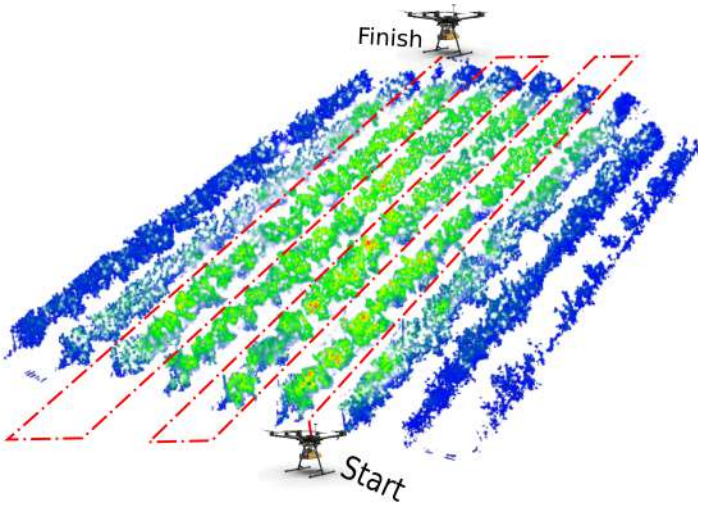
3.1.2 The UAV LiDAR Acquisition

In September 2020, the collection was scanned using a Yellowscan Surveyor LiDAR scanner mounted on a UAV. The UAV had a flight autonomy of 20 minutes. Three protocols were tested, each characterized by a height (H) and a flight speed (V) (Table 3.1). The orchard was scanned in the direction of the tree rows and perpendicularly with a scan line spacing of D. The airborne LiDAR at a given position scans a line up to $10m$ and $8m60$ laterally for heights of $H = 7m$ and $H = 6m$ respectively. Therefore, each position of interest in the orchard is scanned from at least 4 to 6 different LiDAR positions. The flight pattern is shown in Figure 40a and a comparison of the protocols is shown in Figure 40b. In Figure 40b it can be seen that each of the different protocols allows us to obtain different representations of the same geometry. The differences are mainly in the point density and therefore in the level of occlusion. In the case of the aerial scans ($F1, F2, F3$) it is possible to see the blurring of the geometry of the lower part of the tree and the inside of the tree. In the terrestrial scan, these two parts are well-defined.

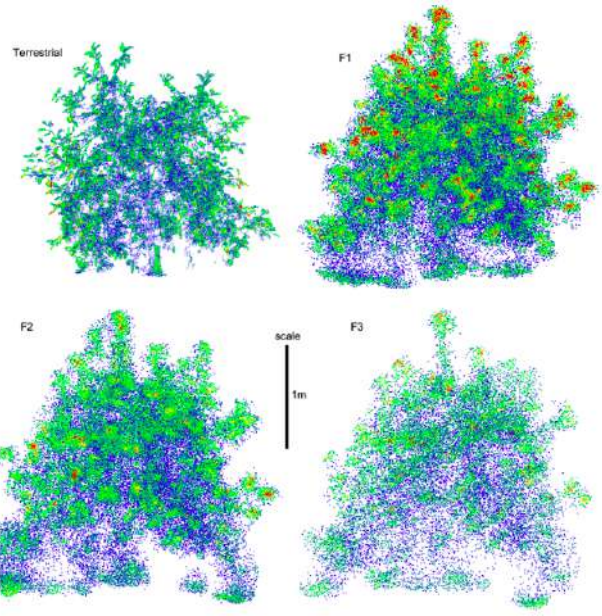
Protocol	$H(m)$	$V(\frac{m}{s})$	$D(m)$	$Pointdensity(\frac{pts}{m^2})$
F1	7	1	5	3140
F2	6	2	5	1842
F3	6	4	2.5	921

Table 3.1 – Characterization of the different scanning protocols based on UAV LiDAR. Height H and speed V of the drone during flight, interval distance D between lines of scans, and resulting point density are given for each protocol.

There are several advantages of using the above flight protocols. Firstly, they ensure an overlap area between the scans, allowing multiple scenes to be aligned, resulting in a comprehensive 3D map of the entire crop. Secondly, these protocols allow us to obtain different representations of the same orchard and make possible their comparison.



(a) An example of the flight path followed by the UAV is shown by the red line. Additionally, the figure displays the point cloud representation of the target apple orchard.



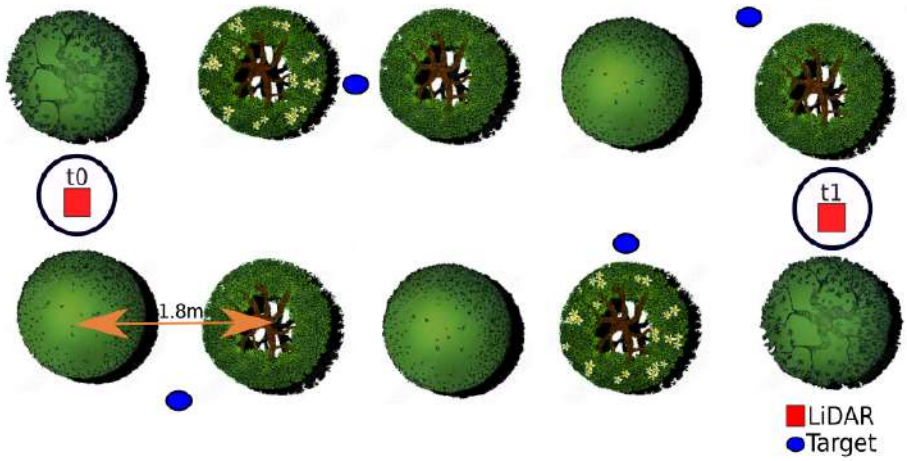
(b) Representation of the point clouds obtained with each protocol. The differences lie in the point density and the quality of the geometry representation.

Figure 40 – Example of the trajectory followed by the UAV and the point clouds of the apple trees obtained with each of the flight protocols. The color gradient seen in the trees represents the density of the points, with blue representing the minimum point density and red representing the maximum point density. The point density was calculated using a sphere of radius $0.1m$.

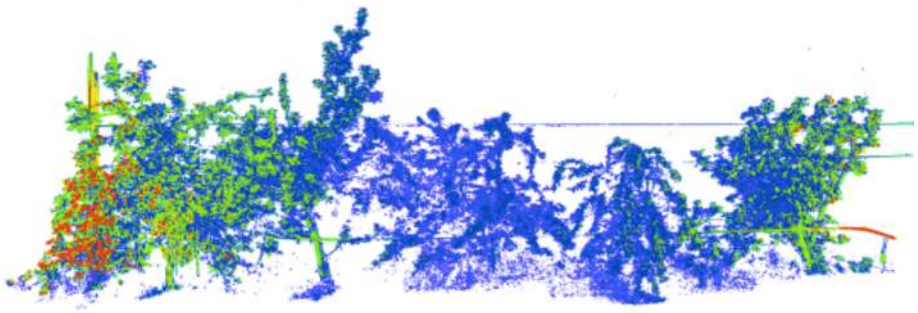
Figure 40a shows that there are 5 lines with a higher density of points. These lines are the most important as they include the trees previously scanned by terrestrial LiDAR.

3.1.3 The terrestrial LiDAR acquisition

Terrestrial scans were acquired in September 2018 using a RIEGL VZ400 terrestrial LiDAR (RIEGL Laser Measurement Systems GmbH, Horn, Austria). The acquisition protocols, as described in (259), consisted of taking a scan in the middle of the inter-row of every 5 tree (approximately $10m$), in the different inter-rows of the orchard (see figure 41a). The scans were performed with a view of $360°$ and an angular resolution of $0.06°$. The scans were then co-registered in a common coordinate system using Riegl software (RiSCAN PRO v2.0.1). This software recognizes the different targets, finds the transformation matrix for each of the point clouds, and applies it to the set of point clouds to create the working scene. In our particular case, the targets are white circular boards with a specific reflectance that allows their easy detection. The scans contain the coordinates x, y, and z of each measured point of the tree surface. The density pattern obtained for a row of trees can be seen in figure 41b.



(a) Representation of the terrestrial LiDAR measurement protocol. The red squares represent the LiDAR position at the time of measurement, the blue dots represent the reference targets used to link the scenes. t_1 and t_1 refer to the time intervals at which the LiDAR is repositioned and the scan is started.



(b) Example of a row scanned using the terrestrial protocol. The colors blue and red indicate the lowest and highest density of points in the scene. The density was obtained using a radius of $0.1m$.

Figure 41 – Terrestrial scan protocol illustration and example of the point cloud obtained

3.2 Estimation of the architectural traits

To process the 3D data produced by the different protocols, a semi-automatic workflow was designed. It is composed of a series of steps including the identification and removal of elements of the environment (soil, pole, etc.), the segmentation of individual trees, and the characterization of architectural traits. For this, a series of software and scripts were gathered.

3.2.1 Data preparation

The first step in the pipeline is to pre-process the point clouds to remove environmental elements and outliers using the CloudCompare software (261). The SOR (*Statistical Outlier Removal*) filter is used to remove the outliers. This filter calculates the mean distance (μ) between all the *k-nearest neighbors* of each point and then adds the mean (μ) to the standard deviation (S) multiplied by a value of n. The result of this operation is used as a threshold to remove all points with larger distance values. Note that the value of n is an input argument to the filter.

Next, the points representing the soil of the orchard are removed using the $CANUPO$ plugin (262). In this tool, the points are characterized by local neighborhoods of different sizes, and the dimensionality is associated with these neighborhoods through a PCA (*Principal Component Analysis*). From these indices, a classifier was trained and used to distinguish flat surfaces representing soil from vegetation. In our particular case, CloudCompare was used to annotate the point clouds. Two labels were defined. The label 1 was used to identify the points representing the tree geometry and label 0 was assigned to all points referring to the ground.

For the terrestrial scans, the poles and wires of the orchards were segmented by RandLA-NET (263), and then removed. Any remaining parts of these elements, if present, were removed manually. RandLA-NET was trained on 10 trees, and from this training set, 80% was used for training and 20% for testing. The accuracy presented in the test set was 0.80 for the pole class and 0.73 for the wire class.

The second step involved segmenting individual trees. For this purpose, we used the label spreading algorithm with a new Forward-Backward-Forward (FBF) implementation. Initially, we performed label spreading using the implementation proposed in scikit-learn (216; 264). The seeds used to initiate the different labels were determined from the measured GPS positions of the trees.Next, an alpha shape (265) was estimated for each group of labeled points representing each tree to determine their boundaries. It is assumed that points close to the boundaries may be misclassified, while those in the center of the tree are necessarily in the correct class (i.e., tree). To refine these boundaries, points close to the alpha shapes were unlabelled. The central parts of each tree were defined as new seed regions. Consequently, a second clustering was performed starting from the extended seed regions. As a result,

individual point sets were defined for each tree.

3.2.2 Scan Characterisation

The scans of the individual trees are characterized by local point densities. The local point density for a given point p_i of the scan is estimated as $d_i = \frac{n_i}{r^2}$ with n_i representing the number of neighbor points of p_i at a distance lower than r. A value of r of $0.1m$ was used to characterize the density of an entire scan of a tree, We estimated the mean density value for all the points of the scan.

3.3 Shape features

Different features related to the dimension and the shape of the tree were estimated and are described in this section.

3.3.1 Plant volume

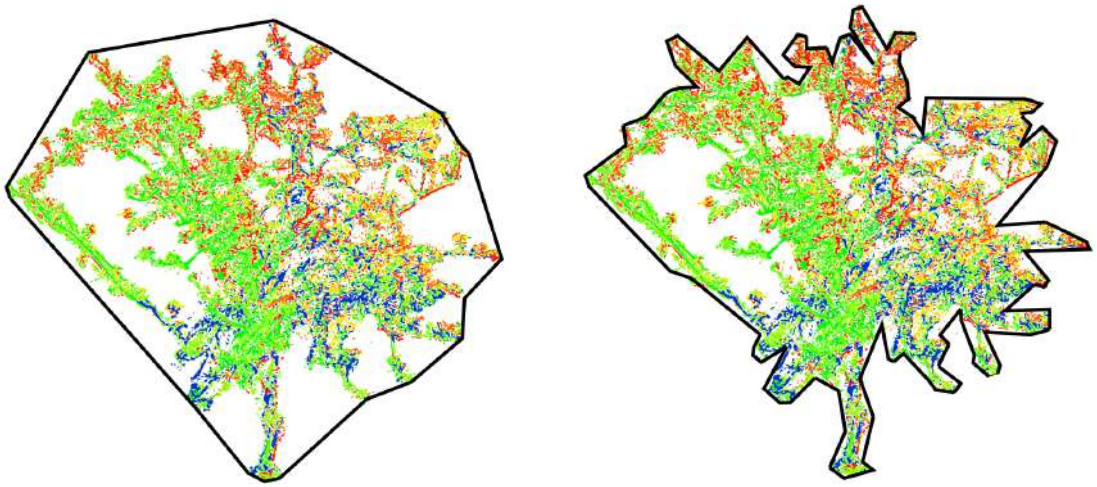
The convex hulls (c_{volume}) and alpha hull volumes of the tree point clouds were computed using the alphashape Python library (266). The c_{volume} reflects the minimal convex volume encompassing the point cloud representing the tree (Figure 42a), while the a_{volume} is an extension of the convex hull that allows the creation of concave envelopes around the point cloud, better reflecting the space occupied by the trees (Figure 42b). The creation of concavities in the alpha hull depends on a parameter (α). To estimate these concavities, a ball of radius $1/\alpha$ is created. This ball is used as the threshold of minimum concavity to be considered between N neighboring points. If $1/\alpha$ is positive, the radius of the disc is $1/\alpha$, and if $1/\alpha$ is negative, it is defined as $-1/\alpha$. The chosen α value (0.20) was the one that maximized the correlation between a_{volume} and the measured TLA on a randomly selected subset of trees (259). A convexity index (ci) was computed as the ratio of a_{volume} to c_{volume} to evaluate the convexity of the shape.

3.3.2 Silhouette to leaf area ratio (STAR)

To estimate this index, the silhouette is first estimated by projecting the pointset onto a plane perpendicular to a given light direction (267). A 2D alpha-shape is computed from the pointset and its corresponding area is computed. For the total leaf area, it is estimated using the relationship with the a_{volume} given in (259), equation 3.1

$$TLA = 1.6342 * a_{valume} - 0.1534 \tag{3.1}$$

This method estimates a directional $STAR$. To be representative of the different light directions perceived by the tree, we used 46 light directions taken from the turtle sky dis-



(a) Convexhull representation (b) Alphashape representation

Figure 42 – Graphical representation of the covexhull and alphashape applied to the LiDAR point cloud of an apple tree.

cretization method (268) and summed the corresponding directional STARs. This index provides hints at the efficiency of the light interceptions of the leaves and the branches.

3.3.3 Canopy height, maximum radius, and eccentricity

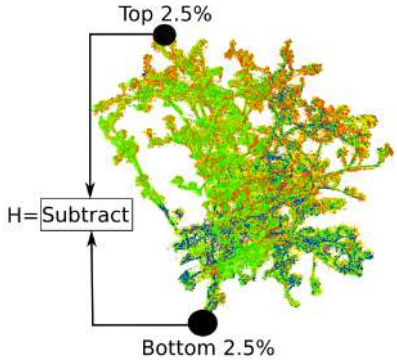
Simple geometric descriptors were also computed. The canopy height was computed by taking the mean height of the 2.5% highest and lowest points and subtracting them together, see equation 3.2 and Figure 43a.

$$Height = \mu(Toppoints) - \mu(bottompoints) \tag{3.2}$$

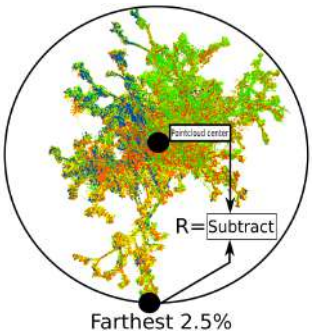
To take into account non-symmetric and/or bending crown shape, the radius and eccentricity of the crown were calculated using a layering principle. For this, the point cloud was divided into different layers of equal height along the vertical axis (we used 10 layers). The barycentre of each layer was calculated, and, for each layer, a radius was determined as the average of 2.5% points of the layer at maximum distance from the barycenter. The maximum radius of the tree was estimated as the maximum layer radius. The canopy eccentricity was estimated as the maximum distance between the barycenters of the layers and the barycentre of the entire point cloud.

3.4 Statistical analysis

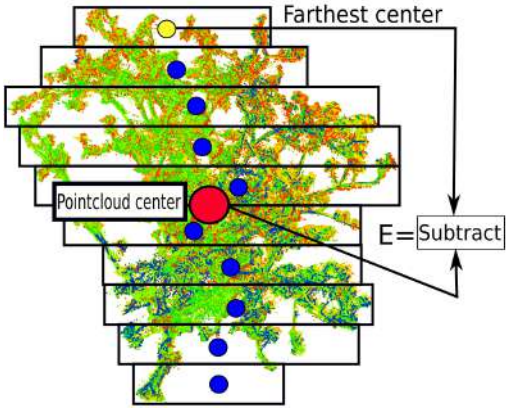
We compared the results obtained with the different protocols for the different architectural traits. Analyses were performed using R software (269). For each trait, the mean and standard deviation were assessed and compared. Correlations between feature values for the different protocols were assessed using ggcorrplot from package ggplot2 (270).



(a) Canopy Height - Side view of the tree



(b) Maximum radius - Top view of the tree



(c) Eccentricity. The layers are represented by black squares. The center of each layer is defined by the blue circles. The center of the point cloud is defined by the red circle. And the farthest layer is defined by the yellow circle. - Side view of the tree

Figure 43 – Graphical representation of how the volumetric canopy height, maximum radius and eccentricity indices are calculated.

We also calculated the heritability value (H^2) and compared them among protocols. For this, linear regression models were built. The phenotypic value of a given trait, P, was decomposed as the sum of the estimated genotypic effect, accounting for genotype replicates, G, and the set of non-genetic and uncontrolled causes of variation, e, and with μ being the mean of the trait in the study population, P is given by equation 3.3. Variance components were then used to estimate the broad-sense heritability (H^2), which is the ratio between genetic variance ($VarG$) and the phenotypic variance estimated as $H^2 = (\frac{VarG}{VarG+\frac{e}{n}})$ with n the number of replicates per genotype.

$$P = \mu + G + e \tag{3.3}$$

Heritability allows for the identification of how much of the variation in a group of indices can be attributed to genetic factors.

3.5 Results

Table 3.2 reports the average and standard deviation values of the different shape features characterizing the tree architecture presented in this manuscript. Scan resolution in the different protocols can be characterized by the average density of individual point clouds. We can observe that F3 has the lowest density. F2 corresponds to scans with densities approximately 2 times bigger than F3; and F1 to scans 1.5 times denser than F2. This seems consistent with the theoretical point density value estimated for the different protocols. In comparison, the terrestrial protocol generates scans 5 times denser than F1.

	Height (m)	Radius (m)	Eccentricity (m)	Convexhull(m^3)	Alpha shape (m^3)	Convexity	STAR	Density ($\frac{pts}{m^2}$)
				F1				
Mean	2.77	1.70	0.66	8.58	3.23	0.37	1.61	23 112.29
Std	0.40	0.38	0.30	3.39	1.18	0.09	0.32	12 808.39
				F2				
Mean	2.65	1.67	0.65	7.73	2.74	0.36	1.76	14 701.81
Std	0.43	0.37	0.22	3.12	1.00	0.06	0.31	7 888.22
				F3				
Mean	2.57	1.62	0.66	7.13	2.66	0.39	1.71	6 614.16
Std	0.46	0.39	0.26	3.39	0.91	0.07	0.29	3 616.94
				Terrestrial				
Mean	2.57	1.62	0.66	7.13	2.66	0.39	1.71	6 614.16
Std	0.46	0.39	0.26	3.39	0.91	0.07	0.29	3 616.94

Table 3.2 – Mean and standard deviation for the different shape features computed on the 36 tree scans from three different acquisition protocols. Different letters in a column indicate significant differences in feature distributions according to a Tukey test ($p < 0.05$)

The estimated mean values of the different shape features seem overall consistent between protocols. For most of them, the values of the airborne protocols are lower compared to the terrestrial ones and could be underestimated except for eccentricity, alpha shape volume and convexity which are higher and could be overestimated. Considering the difference between airborne protocols, we can observe that, except for ratio indices (Convexity and STAR) and eccentricity, the magnitude of the mean values of the different indices are ordered with the biggest value for protocol F1, intermediate value for F2 and lower one for F3.

3.5.1 Comparison between shape feature values estimated with the different acquisition protocols

To go deeper into the comparison between values obtained with the different protocols, correlation indices were computed between the different shape features (Figure 44). Correlation coefficients between airborne protocols are high for Height, Radius, Convex hull and alpha shape volumes, and STAR. Convexity has an intermediate level of correlation while eccentricity has a low or negative correlation between airborne protocols. When comparing airborne values with terrestrial ones, we observe that volumetric indices, such as convex hull and alpha shape volumes, or STAR and height are highly correlated. In contrast, radius

and convexity indices show low correlation while eccentricity is negatively correlated between terrestrial and airborne protocols, indicating certainly a strong effect of noise on the measures of such indices.

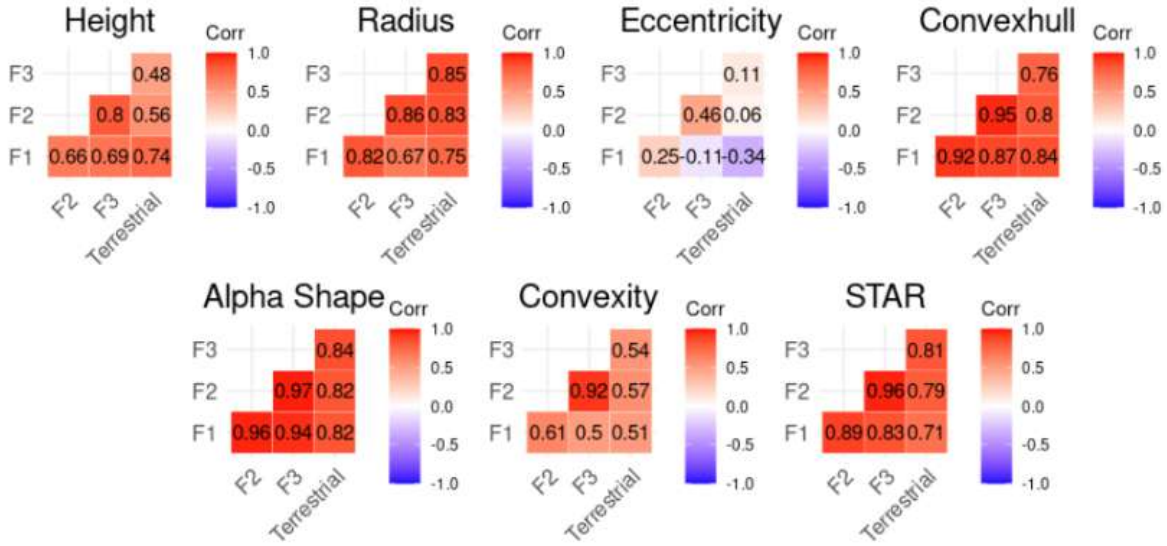


Figure 44 – Correlation plots between the different acquisition protocols for the different shape features.

3.5.2 Comparison between heritability of the shape features estimated with the different acquisition protocols

Finally, the heritabilities of the shape features were estimated for the different protocols (Table 3.3). For the terrestrial protocol, heritability values of the first four indices are above 0.7 and close to each other, in good agreement with values reported by Coupel et al. (2019). Eccentricity and radius are also in the same range. Tree height is less heritable (0.47), which may be due to a higher sensitivity of the estimation to outlier points contained in the scans. The heritabilities obtained with airborne protocols appear sensitive to the point density in the protocol. The highest heritabilities were obtained with the less dense protocols such as F3, which achieved similar results to terrestrial ones. Heritability values were found with F2 and F1 protocols with the lowest values (< 0.25) for the eccentricity. This seems to indicate a higher variability due to the protocol for this feature.

3.6 Conclusions

A semi-automatic pipeline to process LiDAR scans of apple trees and extract different architectural indices is presented in this manuscript. The indices computed from airborne LiDAR gave similar estimates than the one from terrestrial LiDAR even if the point of view and the resolution used for the scans changes. The variability observed on indices with the different

	F1		F2		F3		Terrestrial	
	VarG	H^2	VarG	H^2	VarG	H^2	VarG	H^2
Alphashape	0.83	0.73	0.647	0.76	0.615	0.83	0.389	0.69
Convexhull	5.24	0.63	6.020	0.77	7.7	0.84	6.49	0.73
Convexity	0.001	0.34	0.002	0.67	0.003	0.71	0.006	0.79
STAR	0.042	0.57	0.043	0.60	0.06	0.79	0.161	0.71
Height	0.032	0.33	0.078	0.58	0.068	0.48	0.052	0.47
Eccentricity	0.000	0.00	0.006	0.22	0.029	0.58	0.014	0.69
Radius	0.075	0.66	0.112	0.88	0.087	0.71	0.065	0.73

Table 3.3 – Genetic variability ($VarG$) and Heritability (H^2) estimated for each shape feature and for each acquisition protocol

protocols appears coherent except for the eccentricity. Eccentricity is computed using the mean position of each layer of the tree. The number of points produced for each layer is different with the different protocols and is not linear with depth because of occlusion. As such, sampling of each layer is strongly dependent on the resolution and may produce incoherent results for different sampling. This applied less for the other indices as they focus more on points on the periphery of the scans for which sampling is less disturbed by the occlusion. Similar sensitivity of tree indices estimations to protocols, especially when comparing terrestrial and UAV LiDAR scans was underlined in previous studies (e.g., (160; 271)). Those authors had suggested that UAV LiDAR provide better estimation of canopy height, while the Terrestrial LiDAR is better for the estimation of diameter at breast height. Also, both types of LiDAR were used to obtain stem diameters and tree heights in a mountain forest (272). The heritabilities obtained with the terrestrial protocol is around 0.7 for all indices except for tree height. Heritabilities for airborne scans are dependent on the protocols. Highest heritabilities, which are similar to the ones obtained from terrestrial protocol, are achieved with protocols producing less dense scans. This seems to indicate that the signal-to-noise ratio is low in case of UAV scans and additional scans of the same region add more noise than information. It should be noted that other sensors than scanners can be used in high throughput phenotyping for estimating tree indices, as shown by (273), in a comparison between RGB, Multispectral, UAV and terrestrial LiDAR and WV-3 stereo, for height estimation in two fruit crops, mango and avocado. In conclusion, our study confirms that airborne LiDAR are a valuable alternative for high throughput phenotyping of architectural traits, in a context of genetic studies. Still the protocol used is important to ensure a certain quality in the tree geometry. Here we showed that volumetric indices such as convexhull and alpha-shape are the most robust regardless of the protocol used.

CHAPTER 4

Organ segmentation on fruit tree point clouds

4.1 Introduction

To ensure food security and sustainability, various plans have been proposed to increase food production and reduce the use of chemicals and practices that negatively affect the environment (274; 275; 276). To achieve these goals, several fields of research are considered, such as agronomic management, pest and disease control, harvesting methods, and remote sensing (277; 278; 279; 280). These areas of research share the common goal of understanding and improving plant growth and behavior, with the aim of making plants more resistant to various stresses and increasing their production.

Remote sensing can be defined as the process of monitoring the physical characteristics of a target at distance for instance by observing the reflection of light beams by an object of interest at different wavelength. In recent decades, new technologies and remote sensing techniques have been applied to develop various studies on the growth, production, and behavior of fruit trees, as demonstrated by the work of (281; 257; 258). However, these technologies are not yet mature enough and need to be tuned for each specific crop and environment. Further research and development are needed to fully exploit the value of the signals provided by these sensors.

3D acquisition systems based on Light Detection and Ranging (LiDAR) technology can capture tree shapes with high accuracy and high throughput. In practice, the quality of canopy reconstruction from field data is highly dependent on weather conditions, the shape of the trees, and the scanning protocol, i.e. the position and number of scans collected. Additionally, extracting information from highly noisy 3D point clouds at the organ level remains challenging, and to our knowledge, no robust and easy-to-implement solution has yet emerged for this general purpose.

Recently, after outperforming the state of the art in 2D image analysis (282; 283; 284) and natural language processing (285; 286), deep learning has also shown promising results in 3D point cloud processing (237) and appears to be a suitable candidate for improving plant phenotyping (287; 288; 289; 290). Indeed, deep learning methods offer an interesting solution as they can identify different types of object representations by learning directly from training data.

As a first case study, our goal was to automatically detect apples from scans of an apple tree core collection. Scans were made using two protocols with different resolution. To detect the apples from the point clouds, we designed two automated software pipelines based on machine learning and deep learning methods. One originality of our method is to use synthetic data generated from realistic apple tree mockups, built using the MappleT FSPM, to test the robustness and sensitivity of our methods.

A presentation of this work has been made in FSPM2020 conference (291). I was associated to this work as I contributed to the finalization and extension of the pipeline and preparation of the simulated data sets. However, the abstract was submitted before my PhD started. My name is thus only on the presentation and not on the abstract of the proceedings. A journal article is planned on which I will be a main co-author.

4.2 Materials

4.2.1 Field experiments

This case study was conducted on a non-industrial apple orchard with significant genetic diversity (241 genotypes). The apple trees were not pruned and were very slightly thinned throughout their growth. The orchard received the classical water and nutrient supply. The apple trees, aged of 3 and 4 years-old, were scanned in 2018 and 2019 using a RIEGL VZ400 terrestrial LiDAR (RIEGL Laser Measurement Systems GmbH, Horn, Austria). Two specific acquisition protocols were employed, as illustrated in Figure 45.

The first protocol called *HiRes*, was the most precise and involved scanning a selection of trees from both sides. 31 trees were considered. This protocol was evaluated too time consuming and an alternative protocol was considered. The second protocol, named *LowRes* and described in (259) and in chapter 3, involved scanning each row every 5 trees (approximately ten meters) across the different rows of the orchard. In 2018, the scans were conducted with a 360° view and an angular resolution of 0.06°, while in 2019, the angular resolution was increased to 0.04°. In total, 250 trees with apples were scanned using this protocol in two weeks. With this acquisition protocol, different resolutions of scan were produced. 3 types of resolution can be distinguished. Scans of trees closer to the scanning position, were named of type 1 and show comparable resolution with the *HiRes* protocol. Scan of trees planted next to a tree with scan of resolution of type 1 were considered of type 2 and show a decreased resolution. Finally, scans of trees positioned in the middle of two scanner positions are con-

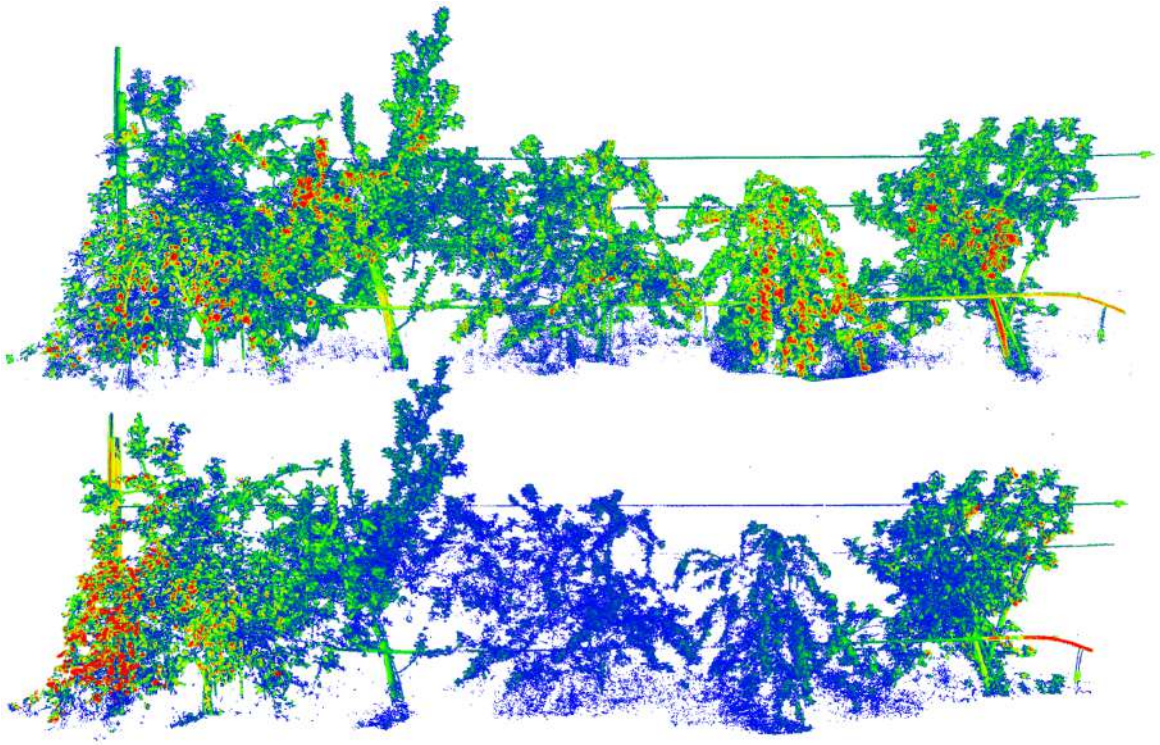


Figure 45 – HiRes (top) and LowRes (bottom) acquisition protocols. A comparison of the local point densities for the two protocols. The color of each point indicates the local point density, ranging from blue to red for low to high density, respectively. The local density of each point is measured as the number of points in its neighborhood (at a distance of 0.1m). In the HiRes protocol, the difference in density between the periphery and the central part of the tree can be observed. In the LowRes protocol, the density also depends on the distance to the scanning positions (near the first and last trees). Central trees in this case have a homogeneous low resolution.

sidered of type 3 and show the lowest resolution.

Scans of all trees were registered in a common coordinate system using Riegl software (RiSCAN PRO v2.0.1). The produced scans contain the X, Y, and Z positions of points sampled on the apple tree's surface. The RIEGL VZ400 also returned additional radiometric information such as the reflectance, the amplitude, and the deviation of the returned signal. The amplitude of a target echo, which is defined as the ratio of the actual detected optical amplitude of the echo pulse versus a detection threshold. As amplitude highly depend on distance, the laser scanner provides also a so-called reflectivity value for each target echo. This value gives the ratio of the actual optical amplitude versus the optical amplitude of a diffuse white target at the same range (assuming that the white target is larger than the laser footprint, 100% reflecting, flat, and its surface normal points toward the laser scanner). This reflectivity value allows for easy estimation of the target reflectivity. Reflectivity values above 0 indicate that the target gives an optical echo amplitude larger than those of a a diffuse white target, i.e., the target is partially retro-reflecting. Finally, for a given laser beam, multiple return echos can be measured by the scanner. To account for this, a standard deviation measure of the distance measured with the different echos is also provided.

This 3 radiometric feature provides complementary information on the measurements made during the scan. As a first test, we build the correlation matrix between these information (see Figure 46). As one can see, the amplitude and reflectance are highly correlated. This can be explained by the limited range of reflectivity of the organs of the apple trees.

Figure 46 – Correlation matrix between the features of the LiDAR scan

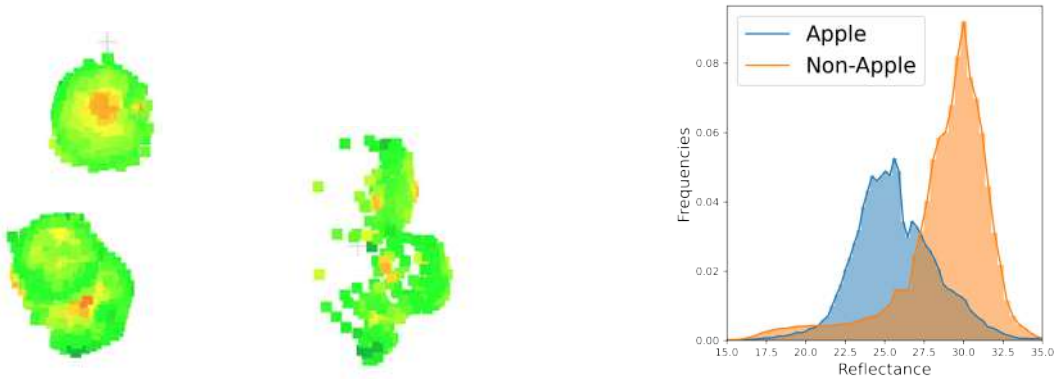
In these scans, some difference between the radiometric characteristics of the apple fruit and the other organs can be observed, as illustrated in figure 47c. Distribution of the reflectances of points of scans of apples and others organs only partially overlap. Such difference was already reported by (165; 169) in their research. Still the domains of the 2 distributions are the same. Additionally, the geometries of the apples have specific deformations with point density varying along their geometry, as illustrated in figures 47a and 47b. Additionally, outliers, considered as noise, are estimated from mixture of different hits of the geometry of the tree by the same LiDAR ray, creating unrealistic points between several geometries. One can note that their shape is not spherical but planar due to measurement errors, adding difficulties in their detection.

For validation purposes, the mean and total weight of the apples from each tree were measured, enabling the estimation of the number of apples produced per each tree.

4.2.2 Ground truth data

To train machine and deep learning models, annotated data are required. A dataset of annotated scans of 10 apple trees was constituted, with 9 of these point clouds made with the *LowRes* protocol, and 1 is from the *HighRes* protocol. For the *LowRes* protocol, 3 scans per type (1, 2, and 3) were built.

The annotations were made by an expert using the Cloud Compare software (261). For each scan, groups of points representing every apple were individually selected using the



(a) Front view of a scan of apples (b) Side view of a scan of apples (c) Comparison of reflectance distribution of scan of apples and the rest of the tree.

Figure 47 – An example of the apple geometry deformations and comparison of distribution of apple and non-apple point reflectance in the scans.

selection tools and saved in individual files. Reflectance information was used to color the point clouds and help the identification of the apple. Individual apples from apple clusters on the tree were sometimes difficult to distinguish and were left grouped in the same point set. Unique IDs were then generated for each apple (or group of apples). The rest of the point set was also saved into a separate file. Using these files, annotated scans were created, by adding new information corresponding to the ID of the shape each point represents. IDs corresponding to each apple were used and the ID 0 for points that do not correspond to apple.

4.2.3 Synthetic data

Point clouds have several characteristics that make them difficult to annotate, such as the geometric deformation of outliers. Annotating point clouds is, therefore, a time-consuming task, and in many cases, it is not possible to annotate with 100% accuracy. As an alternative to expert annotation, virtual scans of synthetic 3D mockups allow the generation of realistic point clouds with automatic annotation. They also allow different levels of noise to be added to test the sensitivity of a reconstruction method.

We developed thus a pipeline to generate synthetic point clouds. For this, 3D representations of apple trees were generated using the *MAppleT* FSP model (57). This model simulates the vegetative development, the flowering, and fruiting of apple trees. Architecture of the tree is defined at the metamer scale. Branching patterns are modeled using Hiden Semi-Markov Chain from observed data. Geometry of the branches are determined using a bio-mechanical approach. The tree growth simulation was stopped when the trees were 2-years-old. At this stage, the simulated trees have fruit in their structures. Some examples of the trees generated are shown in figure 48.

The Z-buffer algorithm implemented in the PlantGL framework (292) was used to simulate the LiDAR scans and parameterize to approximate the behavior of the real LiDAR

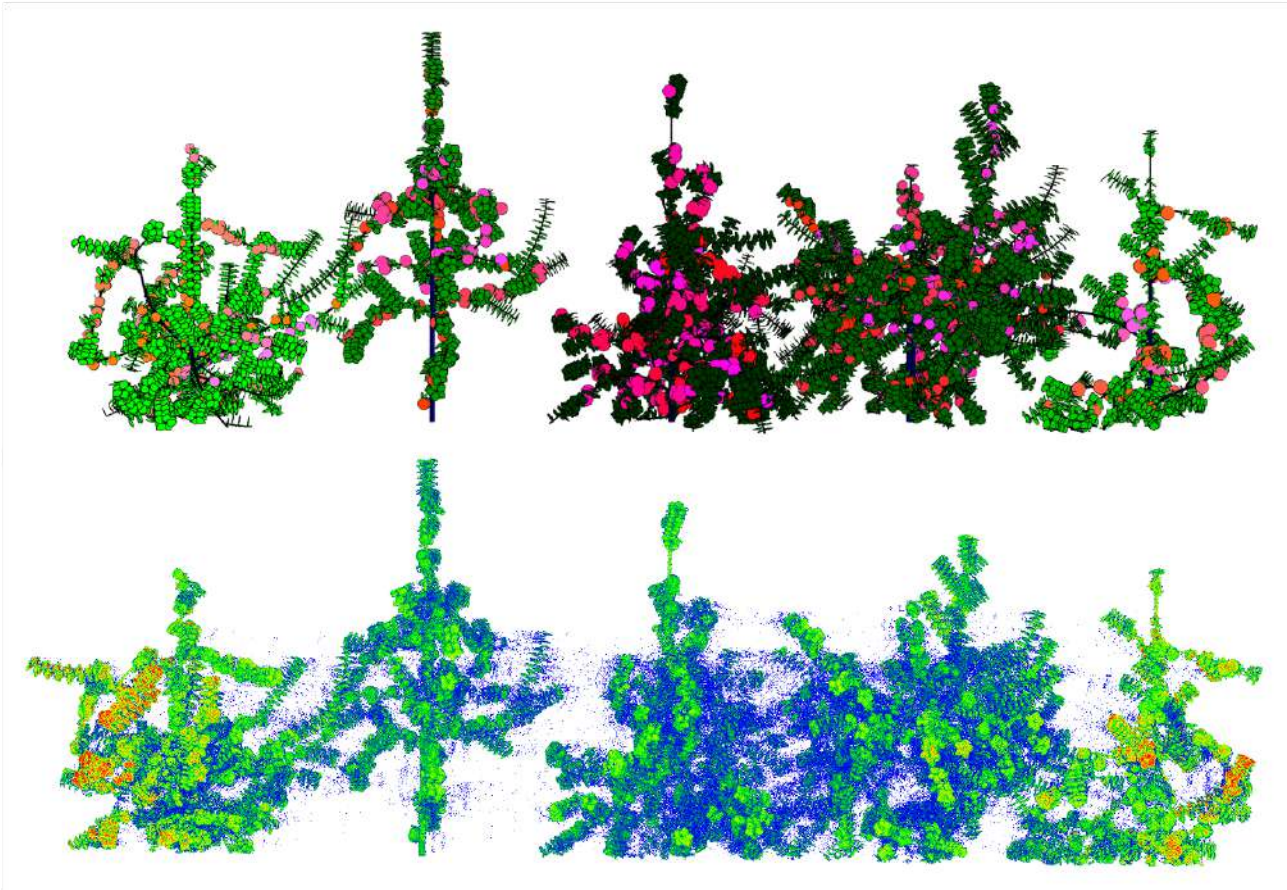


Figure 48 – Synthetic apple trees generated by the MAppleT model. Example of the type of apple trees found in the synthetic dataset generated. To know the position of each tree in the simulated rows, different values were assigned in the blue and green channels of the simulation, the red channel was used to annotate the different organs of the tree.

sensor used to obtain the field measurements. The aim was to approximate the resolution used by the real LiDAR sensor which is given by the angle between 2 rays (0.04° in scans from 2019). To achieve this, the camera field of view (FOV) and the image size (I_{size}) of the z-buffer algorithm were set so that the FOV of each pixel, defined as the ratio between the image FOV and the image size, corresponds to the given scan resolution. To simulate the noise present in LiDAR point clouds due to their intrinsic parameters (section 2.4.5), different sources of noise can be parameterized in the PlantGL Z-buffer based virtual scan tools. White noise can be introduced in LiDAR sensor point clouds and parameterized using the jitter parameter. This noise is inherent in all measurements made by the sensor and results from mechanical and electronic delays within the sensor's internal system. To also simulate the effect of the width of the ray beam of the LiDAR that makes it possible to hit several objects in the scene, a second parameter, called raywidth, can be used. In such a case, the depth measured by a given pixel is estimated as a weighted average depth of all pixels included within the ray width. This makes it possible to simulate outliers, occurring at the intersection of several objects with different depths in the scene.

The lowRes and highRes scanning protocols (section 4.1) were implemented in the vir-

tual environment. This was accomplished by first taking the simulated trees and dividing the dataset into groups of five trees. Then, the position of each of the five trees was estimated, based on the descriptions provided in the field experiment (section 4.1). Once the positions of the trees were determined, the relative positions of the simulated sensor were estimated for each protocol. For the lowRes protocol, the sensor position was estimated while considering the sensor's field of view (FOV), ensuring that the FOV could capture all trees within a bounding box. In the case of high resolution, the sensor was simply positioned perpendicular to each tree. In order to annotate the apples, leaves, and trunk of the synthetic trees, the base geometry type (PlantGL) of each of these organs was checked, and then an integer value was assigned to each of them using the ID attribute of each of the PlantGL shapes.

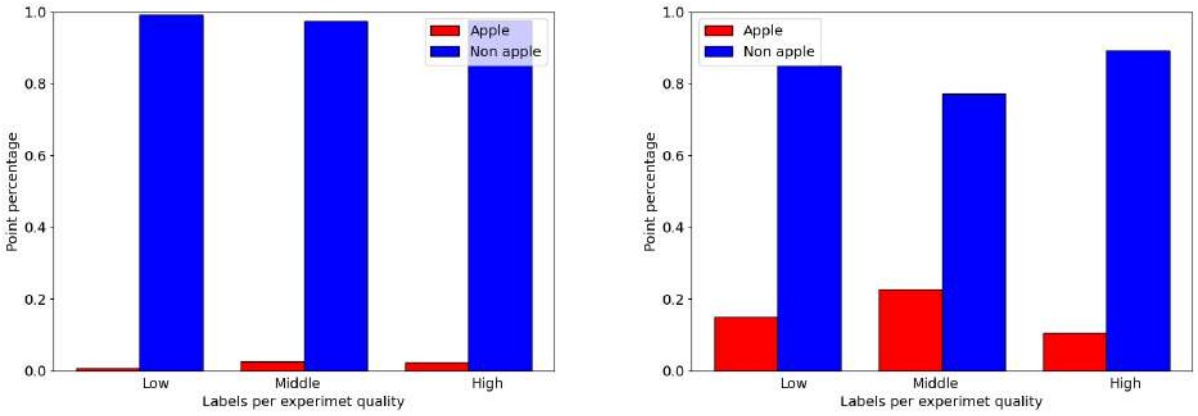
250 simulations were generated to produce training synthetic data. However, to have a comparable point set with scans from field experiments, only the 188 point sets with adequate number of points were considered.

The trees were combined in a group of 5 trees and used to generate the *LowRes* and *HighRes* protocols. Each group of trees was scanned with different simulated configurations of jitter and ray width. In total, each group was scanned 11 times. Eight different values of jitter were used ($0.0m$ to $0.1m$), along with three different values of raywidth ($0.04°$to$0.14°$).

4.2.4 Training data set

To train the models, 80% of the point clouds were used for training, and the remaining 20% were used for testing. The real and synthetic scans were considered separately in the experiment.

The resulting distribution of points within classes is highly unbalanced with only 2.2% of points representing apples for real scans and 16.4% for synthetic scans (see Figure 49).



(a) Real scans

(b) Synthetic scans

Figure 49 – Examples of the imbalance between the labels representing apples (red) and those representing other organs (blue) on the real and synthetic data. Overall, only 2.20% of the dataset represents apple points for scans from field experiments and 16.38% for synthetic scans.

4.3 Methods

4.3.1 Segmentation methods

To detect the different instances of apples within the scans, we designed a software pipeline, called *FruitHunter* (291). This pipeline, based on supervised learning, has two versions. The first one is based on standard machine learning methods, the second one on a state-of-the-art deep learning method. Our intention here is to compare these two types of approaches.

The general pipeline, depicted in Figure 50, consists of three steps. In the first step, some features are estimated for each point to locally characterize the geometry of the scanned object. Based on these features, the second step consists in building a classification model that labels each point depending on the fact that they correspond to the surface of an apple or something else (i.e., non-apple). In the third step, points corresponding to apples are clustered to identify individual apples.

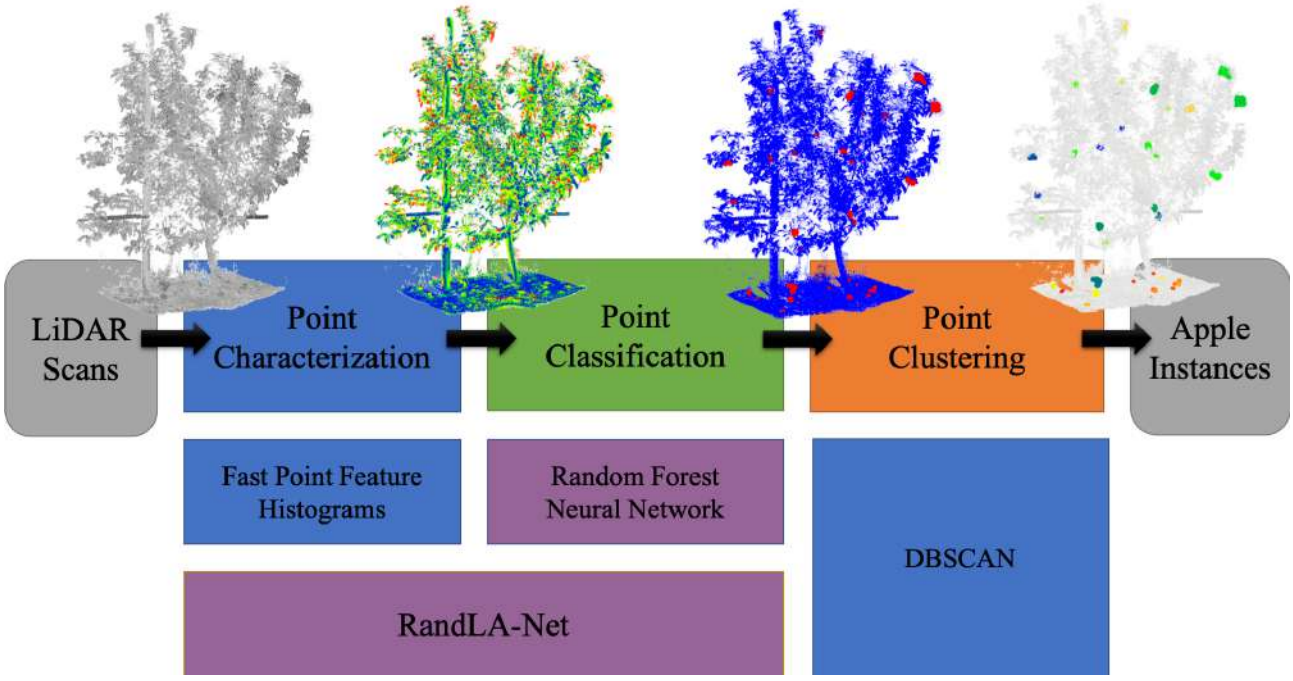


Figure 50 – Fruithunter: a software pipeline for apple segmentation in scans of non-industrial apple orchards. The general schematic description of the pipeline, showing the three main steps to classify points and identify apple instances, is given in the first row. The second and third rows give the different algorithms used for the two versions of the pipeline.

In a first version of the pipeline, the Fast Point Feature Histograms (FPFH) (293) is used to locally characterize each point. This algorithm encodes the geometric features of a point in space based on local neighborhood of varying size. Using these features and the radiometric information from the LiDAR, either a random forest model or a fully connected neural network is trained to predict the class of each point.

In the second version of the pipeline, the first two steps are combined in the RandLA-Net network (294). RandLA-Net is specifically designed to segment high-density point clouds

(containing millions of points) with high efficiency. It incorporates three types of specially designed layers. The first layer is called Local Spatial Encoding, and enables the extraction of features from group of points. The second layer is called Attentive-Pooling, allowing the model to determine which points should be sampled and which should be skipped based on their significance in describing the desired geometry. The last layer is the Dilated Residual Block, which combines the two previous layers and facilitates the extraction of both local and global features from the point cloud. In the original work presenting this network, a comparative test shows the efficiency of random sampling to select key points.

In the third step, similar for the two versions of the pipeline, points are clustered using the DBSCAN algorithm. This clustering algorithm uses the density of points to create groups of points corresponding to the different geometries. It uses a propagation algorithm to extend the initial groups with their neighborhood. The DBSCAN algorithm relies on two main parameters: the first one defines the minimum number of points k required to initiate a cluster within a specified radius. The second one is the distance ϵ to add a point to a cluster. More precisely, if a point p is at a distance below ϵ to a point of a given cluster c, p will be added to c. In this application, the Euclidean distance metric was used. After an optimization on a test set, we use a value of 0.01 for ϵ and 30 for k.

4.3.2 Evaluation

The evaluation of the pipeline was carried out in two steps. Firstly, the classifiers were assessed using the various classification metrics presented in Table 4.1, namely Precision, Recall, F1-score, Intersection Over Union (IoU), Balanced Accuracy and Matthews Correlation Coefficient (MCC). Each of these metrics provides insights into the performance of the models for the apple segmentation task.

Metric	Equation
Precision	$\frac{TP}{TP+TN}$
Recall	$\frac{TP}{TP+FN}$
F1-Score	$2*\frac{Precision \times Recall}{Precision+Recall}$
IoU	$\frac{TP}{TP+FP+FN}$
Bal. Accuracy	$\frac{\frac{TP}{TP+FN}+\frac{TN}{TN+FP}}{2}$
MCC	$\frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$

Table 4.1 – Evaluation metrics for the point segmentation. In the above equations TP=True Positive, TN=True Negative, FP=False Positive and FN=False Negative.

Precision represents the fraction of correctly predicted apple points (TP) among the predicted ones (TP+TN), while Recall (also called sensitivity) represents the fraction of correctly predicted apple points among the actual ones. These two indices can be combined into an harmonic mean, called *F1-Score*. The IoU index, also called Jaccard index, gives a measure

of the similarity of the predicted and ground truth values and is defined as the size of the intersection divided by the size of the union of the two sets.

Another classical index is the Accuracy which is defined as the ratio between correctly predicted values (TP+TN) over the total number of points. However, due the largely unbalanced data set that we consider in this work, we choose to use the *Balanced Accuracy*. Additionally, the Matthews Correlation Coefficient, which also do not gives emphasis on the Positive labels, was also employed. All the indices range between 0 and 1 except for the last index range from -1 to 1 to represent negative correlation between predicted and ground truth data.

These metrics collectively provide a comprehensive evaluation of how well the classifiers are performing in accurately segmenting apples from the point clouds. We use the implementation proposed in (216).

Secondly, the clustering process performed by DBSCAN was evaluated using Homogeneity, Completeness, V-Measure and Adjusted Rand Score (ARI) presented in Table 4.2. The three first one are derived from the Shannon's entropy (H). Assuming two label assignments (of the same N points), $C = \{c_i | i = 1, ..., n\}$ and $K = \{k_j | j = 1, ..., m\}$ (C representing classes of the ground truth labeling and K the clustering of the model), a contingency table A can produced such as $A = a_{ij}$ where a_{ij} is the number of points that are members of class c_i and elements of cluster k_j. Their entropy (295), i.e. the amount of uncertainty for a partition set, can be defined as $H(C) = -\sum_{c=1}^{|C|} \frac{\sum_{k=1}^{|K|} a_{ck}}{n} log(\frac{\sum_{k=1}^{|K|} a_{ck}}{n})$. The conditional entropy can be defined as $H(C|K) = -\sum_{k=1}^{|K|} \sum_{c=1}^{|C|} \frac{a_{ck}}{N} log(\frac{a_{ck}}{\sum_{c=1}^{|C|} a_{ck}})$.

The metrics used for evaluation are as follows:

1. Homogeneity: To satisfy homogeneity criteria, the clustering should assign only points of a single class into a single cluster. To evaluate this, the conditional entropy $H(C|K)$ is used and normalized using the entropy $H(C)$.

2. Completeness: This metric is symmetrical to homogeneity. It evaluates if a clustering assigns all points of a single class into a single cluster. To evaluate this, the conditional entropy $H(K|C)$ is used and normalized using the entropy $H(K)$.

3. V-Measure: The V-Measure is an harmonic mean of homogeneity and completeness.

Finally, the clustering are evaluated more globally using the actual number of apples of the trees. A regression is made to estimate the correlation with the ground truth data. The coefficient of determination R^2 of the regression is used to measure the accuracy of the clustering while the slope coefficient is used to estimate a bias coefficient between the ground truth and the estimated number of clusters.

These steps will ensure that the classifiers and clustering work correctly from the different perspectives given by each metric.

Metric	Equation
Homogeneity	$1 - (\frac{H(C\|K)}{H(C)})$
Completeness	$1 - (\frac{H(K\|C)}{H(K)})$
V-Measure	$2 * \frac{hc}{h+c}$

Table 4.2 – Evaluation metrics for clustering.

4.4 Results

4.4.1 Classifier comparison

As a first experiment, we performed trainings on the different annotated scans from field experiments and synthetically created. The different versions of the pipeline were trained and compared with the different metrics and results are given in Table 4.3. For scans from field experiments, training was made with or without radiometric information (R: Reflectance, A:Amplitude, D:Deviation) given by the LiDAR scanners. For synthetic scans, different levels of noise and resolutions were tested. For random forest and neural networks used in the first version of the pipeline, input of the models were features computed using the Fast Point Feature Histogram (FPFH) while XYZ coordinates of the points are directly given for the RandLA-Net model.

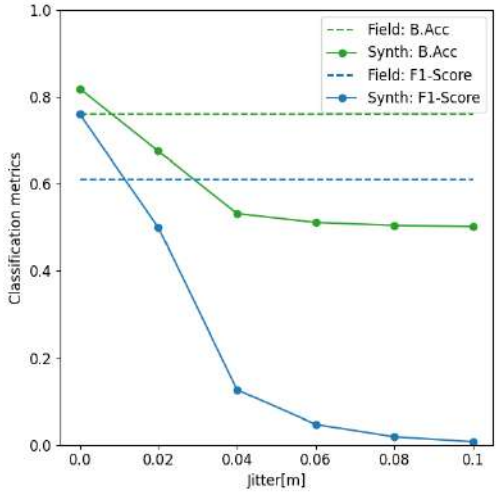
Model	Data	Feature	F1-score	Balanced Accuracy	IoU	MCC
Random Forest	Field	FPFH	0.39	0.58	0.61	0.41
		FPFH+RAD	0.40	0.70	0.58	0.43
	Synthetic	FPFH	0.75	0.81	0.76	0.73
Neural Network	Field	FPFH	0.09	0.74	0.43	0.14
		FPFH+RAD	0.15	0.85	0.48	0.24
	Synthetic	FPFH	0.45	0.69	0.46	0.31
RandLA-NET	Field	XYZ	0.24	0.57	0.56	0.34
		XYZ+RAD	0.90	0.95	0.91	0.90
	Synthetic	XYZ	0.87	0.95	0.86	0.85

Table 4.3 – Comparison of the different classification models.

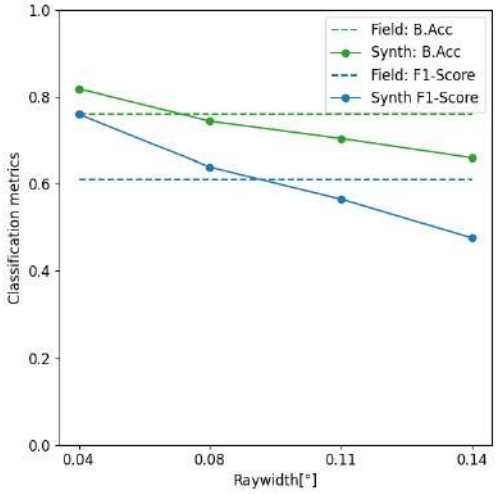
Considering the Balanced Accuracy, the best performance are achieved with the RandLA-Net model, with an accuracy up to 95%. This is confirmed by the IoU and MCC indices, still showing better results for scans from field experiments. This seems consistent with results of RandLA-Net on standard databases (94.8) (294). For field experiment, best results are achieved using the radiometric features. Such features seems to play an important role in the discrimination as achieved accuracy is only 57% without them. Neural Network and Random Forest gives lower accuracy (above 80%) with better result on field data for Neural Network and on synthetic data for Random Forest. Again the importance of the radiometric feature, even if less important, is confirmed with these versions of the pipeline (58% vs 70% without or with radiometric features for Random Forest and 74% vs 85% for Neural Net-

works). Using the F1-score, we can test the specific ability of the different methods to label correctly the points corresponding to apples. For this aspect, only RandLA-Net gives adequate results (90% for field experiments and 87% for synthetic data) while results of Neural Networks and Random Forest are poor.

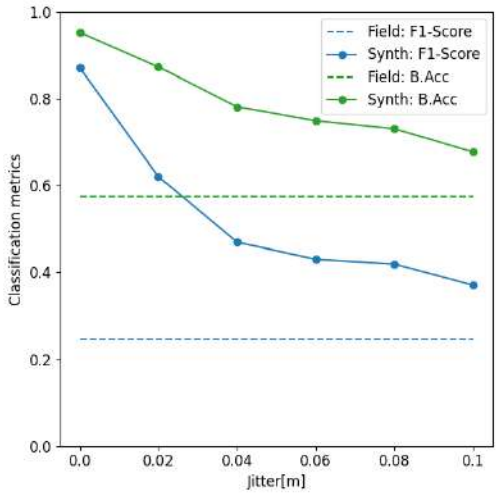
4.4.2 Sensitivity to noise



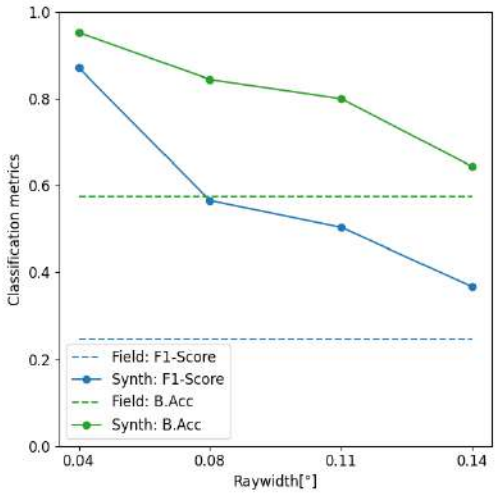
(a) Random forest over different Jitter values



(b) Random forest over different raywidth values



(c) Randla-NET over different Jitter values



(d) Randla-NET over different raywidth values

Figure 51 – Evaluation of the sensitivity of Random Forest and RandLA-NET on white noise and geometric deformation for synthetic apple segmentation. The dotted lines represent the performance of each of the models trained and test over real data. The point in the continuous lines represents the evaluated noise set (jitter or ray width). The evaluated metrics were the Balanced Accuracy and F1-score.

To evaluate the sensitivity of RandLA-Net to noise and deformation, we tested it with synthetic scans created with different level of noise. Range of white noise (*Jitter*) and ray width

where tested independently. With the first set of tests, sensitivity to the imprecision of the measure is tested (Figure 51c) while sensitivity to deformation and outliers is tested with the second set of tests 51d. For the white noise, scans with a range of magnitudes of noise from 0 to 10 cm were generated, while angles ranging from 0.04° to 0.14° were used for the ray angular resolution.

For RandLA-Net, the decreases of accuracy is linearly correlated with the white noise amplitude, with balanced accuracy going from 95% to 70%. The F1-score in this case, decreases rapidly and tend to a plateau of 40% showing a bigger sensitivity of the detection of apple to the noise (Figure 51c). Sensitivity to ray angular resolution exhibit similar behaviour (Figure 51d). Decrease of F1-score seems more linear but stand on the same range of values. Similar results were achieved for Random forest (Figures 51a, 51b). A rapid decreases of performance can be observed for white noise (Figure 51a). For magnitudes of noise bigger than 6 cm, result indicates a balanced accuracy close to 50% and F1-score less than 5%, indicating a particularly poor identification of apple points.

In general, RandLA-Net seems more sensitive to noise and geometric deformation than Random Forest. Comparing with results obtained with data from field experiment, results obtained with Random Forest seems to correspond to low level of noise (approximately 1 cm) and deformation (approximately 0.06°, which is consistent with scanner measurement precision (0.3 cm). Experiments of sensitivity of RandLA-Net seems to outperform results from field experiments, indicating specific difficulties for RandLA-Net to process XYZ information only on this type of data.

4.4.3 Use of radiometric information

To understand the impact of the different radiometric information provided by the LiDAR, RandLA-Net was trained and tested with different mixtures of radiometric features (Reflectance, Amplitude, Deviation). These tests shows the influence of each individual and combination of information have on the accuracy of the trained model (Table 4.4).

Interestingly, radiometric information alone (RAD) seems to have by default similar results than 3D coordinates alone (XYZ) with a balanced accuracy around 60% and an F1-score of approximately 30%. Still they provide the poorest result. A combination of XYZ coordinates with Reflectance or Amplitude improves greatly the accuracy of the model, with a balanced accuracy of 85%. On the opposite, the combination of XYZ with Deviation does not improve the accuracy, in comparison with the test with XYZ only. Combining 2 radiometric features with the XYZ coordinates increase the performance of the model in general, the mixtures of the XYZ-Deviation-Amplitude or Reflectance gives a better performance than the one with the mixture XYZ-Amplitude-Reflectance. Finally, the combination of the XYZ coordinates with the 3 radiometric features gives the best results with a balanced accuracy of 95%.

These seems to indicated the importance of Reflectance and Amplitude in the accuracy

of the classification. These two feature provide still similar information, as was observed in the correlation table given in Figure 46.

Model	Field Data	F1-score	Balanced Accuracy	IoU	MCC
RandLA-Net	XYZ	0.24	0.57	0.56	0.34
	XYZ+R	0.82	0.91	0.84	0.82
	XYZ+A	0.83	0.90	0.85	0.83
	XYZ+D	0.40	0.67	0.64	0.46
	XYZ+RA	0.78	0.88	0.82	0.79
	XYZ+RD	0.76	0.86	0.80	0.77
	XYZ+AD	0.88	0.94	0.89	0.88
	XYZ+RAD	0.90	0.95	0.91	0.90
	RAD	0.32	0.67	0.58	0.31

Table 4.4 – Evaluation of RandLA-Net with different radiometric features combination (XYZ)

4.4.4 Transfer learning from synthetic data

Robustness of deep learning models comes with an important amount of annotated data. In the case of 3D point cloud segmentation, annotation is fastidious and limit thus the amount of data that can be produced. To address this type of issue, the concept of transfer learning has been proposed, which consists in reusing knowledge learn for one task onto another task. Boosting of performance has been observed on diverse tasks (227; 296) For this, we propose to reuse model parameterization on synthetic data to initiate learning on field experiment data. Figure 52 gives an accuracy of the resulting models. In a first step, the model is first trained on synthetic data giving a balanced accuracy of 95%. When testing such model directly on field data, an accuracy of 50% was achieved. In a second step, the training of the model is fine-tuned on field data, leading to a balanced accuracy of 78%. This outperforms a direct training on field data with a balanced accuracy of 57%.

4.4.5 Clustering

Finally, the DBSCAN algorithm is used to cluster labeled points into groups representing apples and estimate thus the number of apples per tree. To validate the clustering, the number of clusters was then compared to the manually counted apples as shown in Figures 53 and 54.

As a first series of tests using synthetic data (Figure 53), perfectly labeled scans were first used as input of the DBSCAN algorithm. The resulting regression shows a r^2 of 0.92 and a coefficient of correlation of 0.91, showing a good ability of this approach to detect apples, still with a small underestimation. Labeling from RandLA-Net on synthetic scans was then used as input of the clustering process. In this case, the correlation coefficient is 0.82 with a

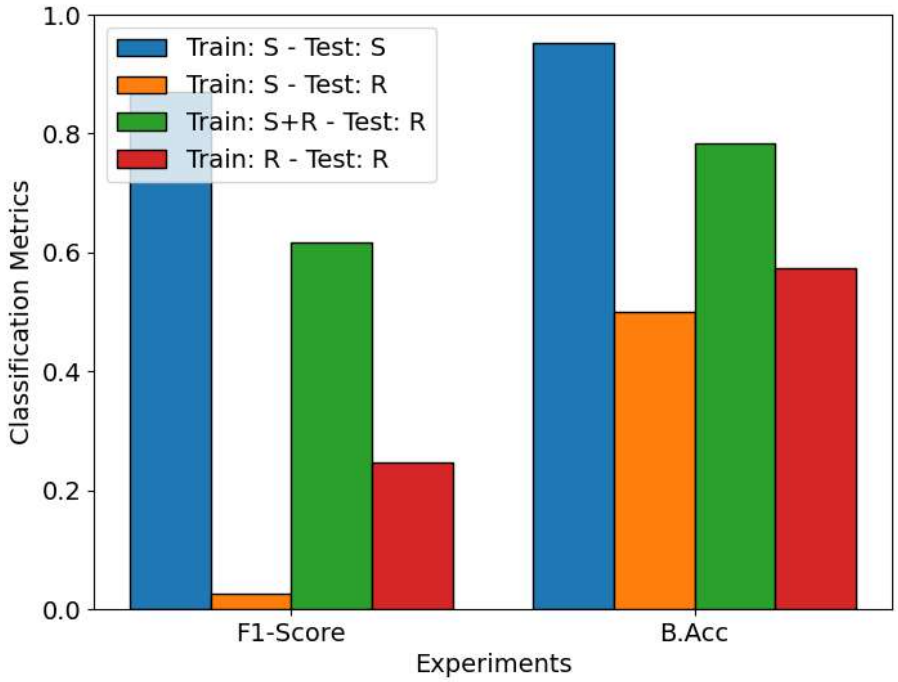


Figure 52 – RandLA-NET performance over synthetic and real pointclouds

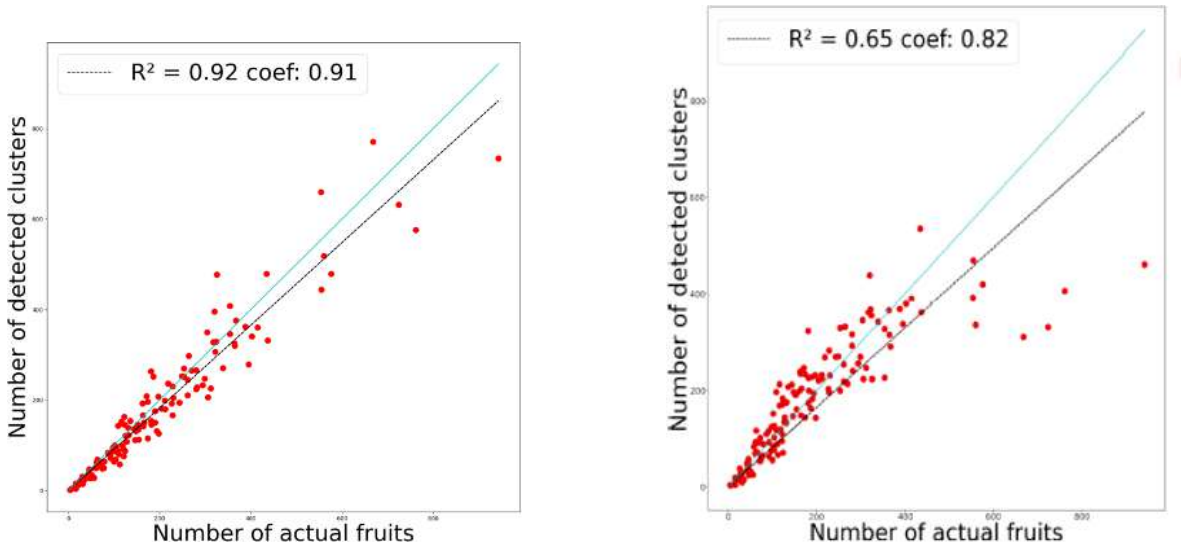
r^2 of 0.65. We can note that the discrepancy between the two enumeration of apples seems to come from trees with large numbers of apples which are largely underestimated.

In a second series of tests (Figure 54), synthetic scans were tested and estimated clusters are compared to manually estimated production. In general, the resulting regression shows a r^2 of 0.35, showing limited results. The correlation coefficient is 0.62, exhibiting an underestimation of the number of apples. In this case, case of underestimation of the number of apples can be found for trees with low to high number of apples. More in details, number of apples seems to be better estimated on scans of type 1 (correlation coefficient 0.74) than on scans of type 2 and 3 (coefficient 0.56), with less error of estimation on trees with low number of apples. This shows the importance of the resolution and can be potentially explained by a bigger level of occlusion.

Experiment/Metric	Homogeneity	Completness	V-Measure
Real	0.166	0.661	0.263
Synthetic	0.038	0.115	0.048

Table 4.5 – Clustering evaluation

Testing the achieved clustering with homogeneity and completeness measures gives additional information on the accuracy of the process. On our experiment, homogeneity gives better values than completeness. For real scans, homogeneity value of 0.166 are achieved in average while a value of 0.661 are obtained for completeness (Table 4.5). This can be explained by a lower number of cluster is obtained than in the ground truth. In such case,



(a) Perfectly labeled point sets.

(b) Point sets labeled using RandLA-Net.

Figure 53 – Comparison between number of fruits and estimated clusters from synthetic data.

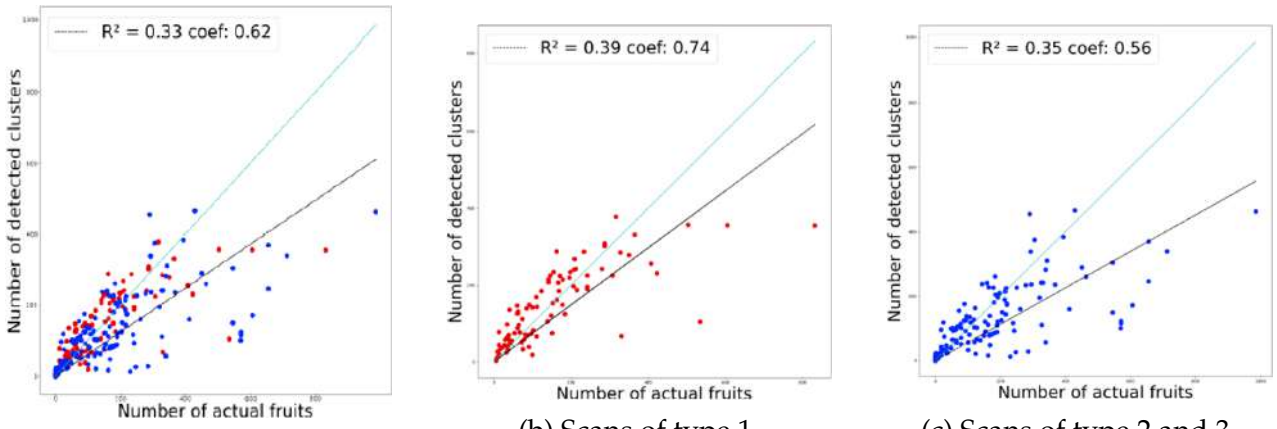
homogeneity is low as points of a cluster can be map to several classes representing apples. In opposite, a majority of points of a single cluster are mapped to the same class showing a better value of completeness. Similar tendency can be observed for synthetic dataset, with even lower values.

4.5 Discussion and conclusion

In this work, we designed and compared two versions of a pipeline based on state-of-the-art machine and deep learning approaches to classify points to identify apple geometry within orchard LiDAR scans. The considered orchard is a core-collection with large genetic variability. The process of identifying apples was decomposed into a point classification step and an instance identification step. Both of them were carefully evaluated. To test more in detail the sensitivity of the methods to noise in the scans, we used virtual scans of digital mockups.

Our results suggest that a trade-off between scan resolution and organ detection accuracy needs to be considered in future protocols. This trade-off may depend on tree age and training systems, as large trees seem with high production to be more sensitive to errors of estimation.

Since ground truth data are difficult to build on specific 3D data such as LiDAR scans, our experience confirm the interest of pre-training from synthetic data. Still, results can certainly be improved with more realistic geometric models and scan noise. One possibility for this is to build deep learning methods to generate realistic scans of apple trees, using for instance Generative Adversarial Neural Networks (GAN). In particular, the deformation of existing



(a) All scans. (b) Scans of type 1. (c) Scans of type 2 and 3.

Figure 54 – Comparison between number of fruits and estimated clusters from field experiments.

synthetic scans with realistic noise can be considered. This approach, based on LiDAR data, shows interesting but more limited results compared to approaches based on RGB images (297; 298) in particular on commercial orchards. A combination of both sensor types could help a better estimation and volumetric reconstruction of the organs.

Funding

The authors declare that they have no competing interests. This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement $N°817970$. This work was also supported by the French National Research Agency under the Investments for the Future Program, referred to as $ANR-16-CONV-0004$.

CHAPTER 5

A proposal for the generation of realistic fruit tree point clouds

Several deep learning models can be used to detect or segment the organs of a fruit tree in a point cloud obtained from LiDAR, such as Point-NET(238), Point-NET++(239), RandLA-NET (294), and VOTE-NET (299), among others (300; 301). These models have shown promising results in both our research and in research conducted by other authors (302).

However, point cloud annotation is a highly time-consuming task, and depends on the complexity of the scanned object and scene. With 3D points which are difficult to navigate within, the task can be hard to achieve with accuracy. Specifically, when it comes to processing point clouds of fruit trees, it is challenging to annotate the data to identify fruit, leaf, and trunk (for instance or semantic segmentation). The annotation of the above-mentioned organs is difficult, first of all because of their great number, secondly because of the difficulty in detecting the points corresponding to each specific geometry, due to the different levels of deformation present in the point cloud. Finally, the visualization of these organs and the handling of this type of point clouds is complex.

GAN (Generative Adversarial Networks) models are in theory able to generate synthetic data with a high degree of realism, as these models learn the statistical distributions of a given dataset to later generate elements with similar distributions. GAN-based point cloud models have been used to generate point clouds of cities for feeding models in autonomous vehicle navigation, creating virtual reality maps, and more. A better description of GAN models can be found in Sections 2.7.1.

Therefore, we propose to develop a deep learning GAN model that can generate realistic tree point clouds from the previously generated synthetic data using MAppleT and PlantGL. The proposed methods to improve the synthesized point clouds include the approximation of the reference geometries, the adjustment of the point density, and the mitigation of the possible noise caused by the environment and the sensor itself. More specifically, the development of this model aims to answer two fundamental questions, firstly *is it possible to*

generate the natural variability of the geometry of fruit trees? and secondly *is it possible to reproduce the characteristics of LiDAR-derived point clouds?*.

In the following sections, I will discuss why the tree topology leads to complex point clouds that are difficult to replicate, and the differences between real apple tree LiDAR scans and synthetic scans generated using MAppleT and PlantGL (Section 5.1). I will also explore how the shape transformation between point clouds can be approached using deep learning, and I will present a proposal for a GAN model to approximate apple tree LiDAR point clouds (Section 5.2). Finally, I will present the current state of point cloud processing, and the current approaches using GAN, and discuss how these new methods can be applied to improve and finalize the proposed model (Section 5.3). A detailed description of the hypothesis, the experiments conducted, as well as the results and conclusions, will be presented in the corresponding sections.

5.1 Differences between synthesized tree point clouds and those taken by LiDAR

Fruit trees display complex topologies and varying canopy densities influenced by factors like tree age, season, and species. As an apple tree grows, it undergoes changes in its branch count and leaf density, leading to an overall increase in its topology complexity over time. Moreover, the tree's growth patterns are influenced by seasonal temperature variations, resulting in the emergence and loss of leaves, the appearance of new buds, and the growth of additional branches, flowers, and fruits. Tracking the tree's development and growth becomes challenging due to the complex nature of representing such data and the intricate geometry of the tree itself. It is important to consider that each fruit tree species exhibits unique architectural characteristics, and different genotypes within each species possess distinct topological and physiological traits, further complicating the task of developing an analytical model for their study.

When using LiDAR for field measurements to capture a 3D representation of fruit trees, it's important to recognize that these measurements possess distinct characteristics resulting from the trees' unique attributes and the influence of environmental variables such as wind and sunlight. These characteristics primarily pertain to the interrelationships among the points that describe a specific geometry. They will be described in the following section.

During the generation of a synthesized point cloud, it is essential to acknowledge that the resulting tree topology may deviate significantly from that of a real tree. Additionally, the point density and noise present in the synthesized point cloud are likely to differ from reality, as it can be challenging to replicate the environmental variables that impact them.

5.1.1 Capturing LiDAR Point Clouds of Fruit Trees: Challenges and Considerations

Outdoor LiDAR point cloud acquisition poses various challenges. Firstly, the environment undergoes continuous changes, and the targets experience displacement, leading to outliers, geometric deformation, and ghosting in the point clouds. Figure 55 illustrates the different types of noise identified. The severity of these issues can vary based on the sensor's sampling rate and the extent of geometry displacement with each measurement. A point cloud with outliers is characterized by an uneven point density and a significant number of scattered points not associated with any recognizable geometry.

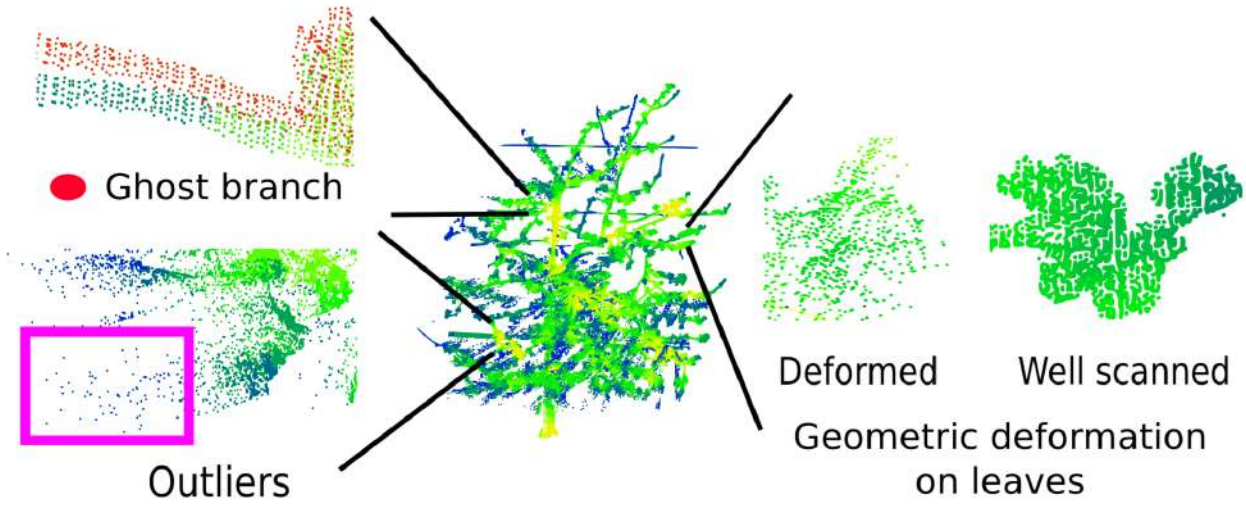


Figure 55 – The center of the figure is a representation of an apple tree scanned with a LiDAR sensor in an external and dynamic environment, exhibiting different types of noise. In the lower left side, highlighted in the purple box, the outliers are shown, which are commonly found in scans obtained with the LiDAR sensor. On the upper left side, ghosting is presented; this noise typically occurs when two point clouds are aligned, but the scanned object is displaced, causing misalignment. On the right side, an example of a deformed geometry is shown alongside one without deformation. Deformations usually occur when the geometry moves during the scanning process and the sensor frequency is low.

An outlier point is identified as such if it exceeds a certain distance from its nearest neighbors, indicating a significant deviation from the expected density distribution (303). The geometric deformations observed in the point cloud can be attributed to two main factors. Firstly, the object being scanned may experience slight displacements during the sampling process, leading to variations in its shape and position. Secondly, if the sampling rate of the sensor is low, it may accurately capture these small displacements, resulting in the object's position being mixed within the point cloud at different time intervals.

Ghosting refers to the phenomenon where geometry is duplicated or repeated in a scene, with each instance separated by some distance offset (304). This behavior typically occurs when attempting to align two or more point clouds and there is a displacement of the object of interest in each of the acquired measurements. As was explained in section 2.4.5 the pro-

cess of alignment made reference to the action of merging different views of the same object with the purpose of ensuring the correct 3D representation of the target. In addition, merging different point clouds can increase the number of artifacts and alter the original point distribution, which can complicate data annotation and machine learning tasks. An example of a point cloud created by combining multiple views of the same geometry, as well as a point cloud of a single geometry, is shown in Figure 56. This figure shows the variation in point density in each part of the tree.

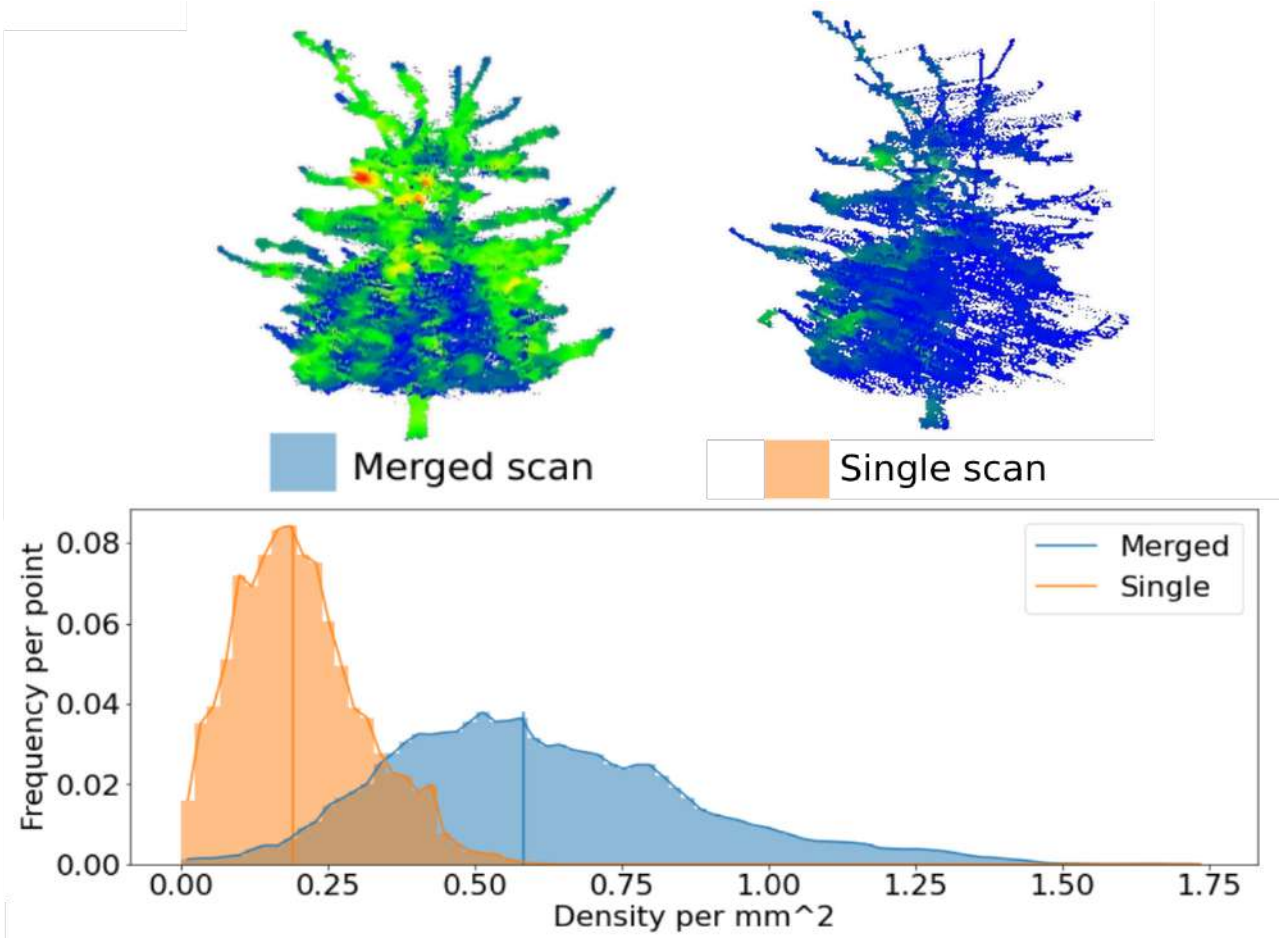


Figure 56 – Differences between the point distribution in a single scan and a group of merged scans. At the top of the figure are the point clouds related to the histograms at the bottom. The colors present in the point clouds relate to their density, where red means high density and blue means low density. It is clarified that the two point clouds are framed on the same color scale in order to show the differences in density and how it affects the integration of different views to represent a single geometry.

Secondly, when scanning a target, it is common to encounter varying levels of occlusion within the geometry. The first level of occlusion occurs because only a part of the geometry or scene is seen from a specific position, given the limited field of view of the sensor. Therefore, it is necessary to scan the same geometry or scene from different viewpoints and align or merge them into a single element to obtain a complete description. The second level of occlusion arises when multiple objects within the scene or target obstruct each other, making it challenging to capture a fully detailed view of the geometry. In the case of trees, the foliage

elements and a portion of the trunk can be well described, but there may be varying degrees of occlusion in the inner part of the captured tree. It's important to note that the density of the foliage can affect the extent to which the inner part of the tree is captured. In situations where the foliage density is high, there is a possibility that the inner part of the tree may not be adequately scanned or captured. Please refer to Figure 57 for visual reference.

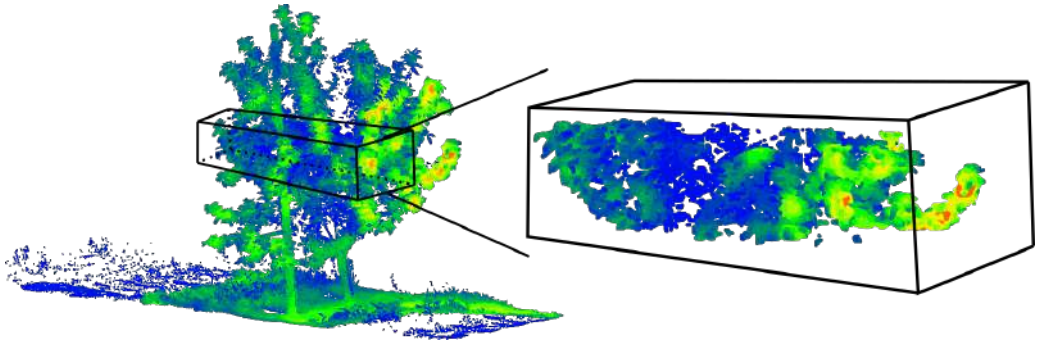


Figure 57 – Density difference from the inner part of the tree to the external part. This image is part of a point cloud capturing a 4-year-old apple tree. It reveals a low leaf density, and the crown appears sparsely covered with leaves. Within this image, a cube has been extracted from the inner region, and the density has been estimated for each point. The points with the highest density are depicted in red at the ends, while towards the center of the tree, variations in density and the quality of the tree's geometrical representation can be observed.

Finally, the target geometries within a point cloud may exhibit varying density values across different regions. When multiple scans are merged together, the resulting point cloud may exhibit an even greater gradient of point density between each scan. This means that the density of points in the merged point cloud can vary significantly across different regions, leading to irregular point distributions throughout the entire dataset. An example of the variation in density and geometry of the same organ is shown in Figure 58.

5.1.2 Analyzing the Discrepancies Between Synthetic and Real Scans: A Comparative Study

As explained in Chapter Two, the MAppleT model was used to generate several 3D representations of apple trees. The point clouds of these trees were obtained using the ZBuffer algorithm implemented in PlantGL.

The 3D models generated by MAppleT exhibit specific characteristics. Firstly, the leaves and fruits are represented by a single geometry for each at a specific time (t). Secondly, the model does not consider plant reflectance characteristics, resulting in RGB colors assigned to each geometry at specific time intervals. Lastly, the appearance and developmental patterns over time are determined based on field measurements for the desired genotype.

These characteristics produce 3D trees with topologies resembling real trees but with limited variability in the geometries representing certain organs. In real trees, variations in

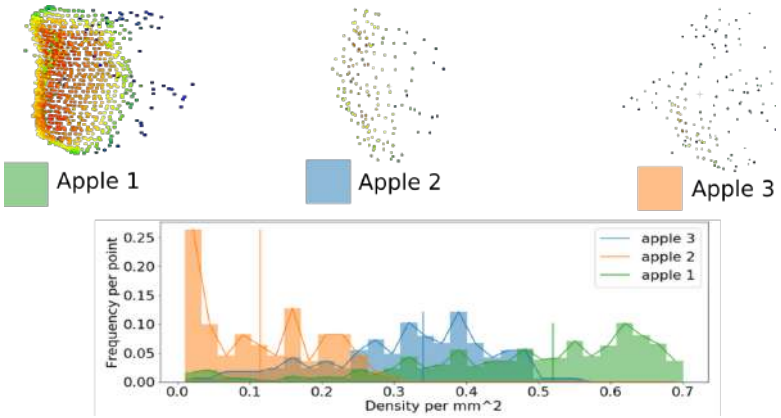


Figure 58 – The shapes shown correspond to the apples of one of the scans made in the field with the LiDAR sensor. The difference between the apples lies in the quality of the geometric description and their point density. Apple 1 has a high point density and part of its geometry is well reconstructed. This apple was at a closer distance than the other two example apples. Apple 2 has a lower point density and its geometry is not well described due to its greater distance from the sensor. Apple 3 not only has a lower point density than the other two example apples but also shows deformations. The histogram at the bottom of the figure shows the point density distribution of each of the apples.

size, shape, and reflectance of the leaves, fruits, trunk, and branches can be observed. Figure 59 illustrates these differences.

When analyzing both virtual and real LiDAR sensors, differences can be observed in their intrinsic parameters and the environment where the measurements are taken. Regarding the intrinsic characteristics of LiDAR, it is observed that due to its mechanical and digital systems, intrinsic white noise is always present in their point clouds. The definition of white noise can be seen in Equation 5.1. In this equation, $\mathbf{X}$ represents the point cloud generated from the ZBuffer algorithm, $\hat{\mathbf{X}}$ represents the point cloud with the added noise, and $\mathbf{Z}$ represents the noise $Z \sim N(\mu, \sigma)$, which is defined as a Gaussian distribution with mean (μ) of 0 and a standard deviation (σ) that defines the range of the noise.

$$\hat{\mathbf{X}}_\mathbf{i} = X_i + Z_i \tag{5.1}$$

In the case of simulated LiDAR, the intrinsic noise is not explicitly defined and must be approximated using the jitter variable, as in our case. Figure 60a illustrates how the geometry deforms as the jitter values increase. As explained in section 4.2.3, jitter is a parameter of the ZBuffer algorithm that introduces white noise to the position of each point in a point cloud.

Another factor to consider when comparing the virtual and real sensors is the perception or definition of outliers. As discussed in the section 2.4.5, the LiDAR laser beam has specific characteristics, which are determined by the diode type used for triggering. These characteristics, combined with the method used to estimate the position of the impact point, result in the estimation of an average position when the laser beam hits multiple objects at different distances. This leads to the creation of outlier points that do not correspond to

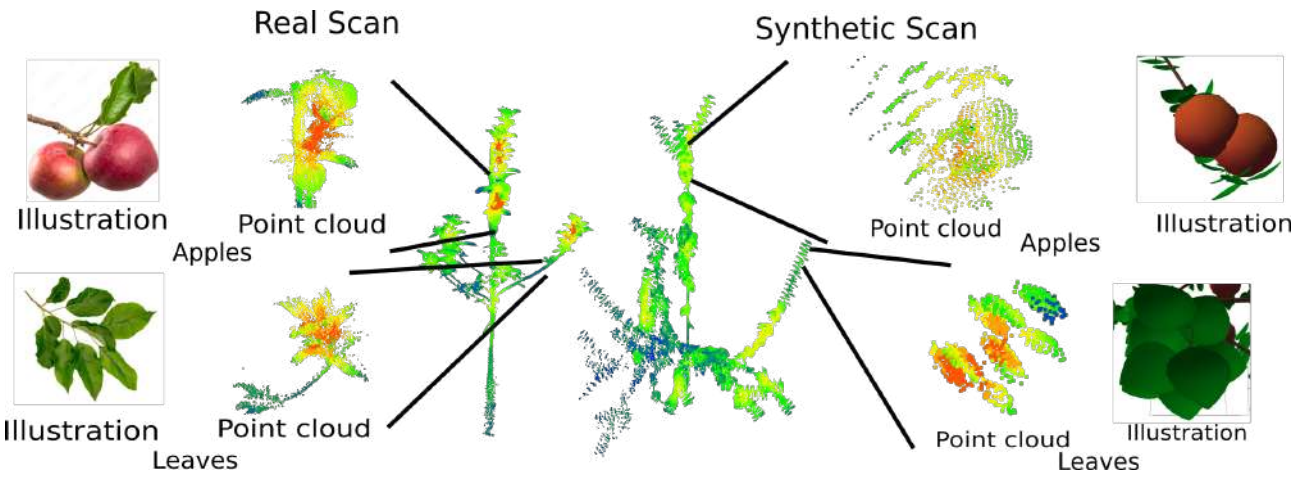
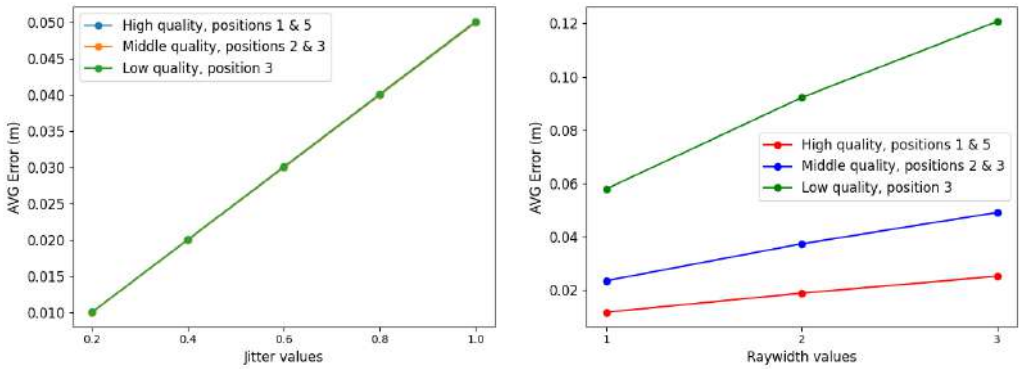


Figure 59 – The figure on the left shows the point cloud of a real apple tree. The top-left corner shows an illustration of the fruit and its corresponding point cloud visualization. The bottom-left corner shows an illustration of the leaves of the apple tree and their corresponding point cloud representation. In both cases, the variation of the geometries is evident. The image on the right shows a synthetic apple tree. The top-right corner shows an illustration of its fruit and the type of point cloud obtained by scanning it. The bottom-right corner shows an illustration of the synthetic leaves and an example of the type of point cloud obtained by scanning this type of geometry. The colors in the point clouds represent the gradient of point density, with red representing high density and blue representing low density.

real geometries. As explained in section 4.2.3, for the virtual sensor this behavior is determined by the raywidth argument of the ZBuffer algorithm. This parameter synthetically defines the radius of the laser beam by considering the set of pixels in the depth image that fall within that radius. It then estimates the average distance covered by each pixel within the radius. This approach allows for the generation of an intermediate point when multiple pixels point to different targets at varying distances. Conversely, if a group of pixels points to the same target, the resulting point will accurately represent the distance to that target. Figure 61 depicts an example of a scene that leads to the generation of an outlier.

5.2 GAN-driven Approach for Simulating LiDAR Point Clouds of Trees: A Proposal

As presented in section 2.7.1, GAN models are models that learn the statistical distribution of a given dataset and then generate new data based on this learned distribution (306). This means that these models do not evaluate the data itself. Instead, they try to find a generalization of the defining characteristics of the data. To accomplish this, these models primarily consist of two neural networks that compete between each other. The first model, known as the *generator*, aims to generate data that closely resembles real data to confuse the second model. The second model, known as the *discriminator*, is responsible for discriminating between real and fake input data. By the end of the training process, the discriminator is



(a) Point position displacement as jitter increases

(b) Change in point position as raywidth increases

Figure 60 – Illustration of the deformation of the geometries varying the jitter and raywidth values on the low-resolution protocol

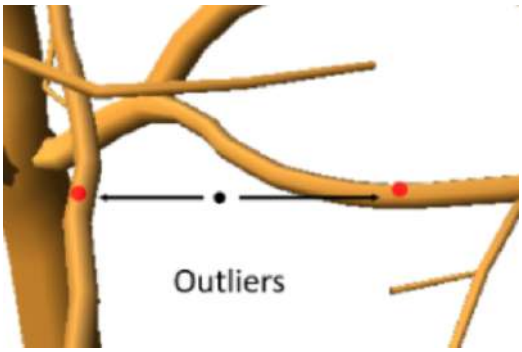


Figure 61 – Illustration of the creation of outliers during scanning. Red points represent partial hits made the same laser beam. The black point is the resulting outlier. Figure from (305)

expected to exhibit confusion by achieving a precision of 0.5, indicating that it is unable to differentiate between real and synthetic data. This suggests that the generator has successfully learned to replicate real data. Consequently, GAN models are typically classified as implicit density models (307).

In order to simulate the generation of synthetic fruit trees, a study was conducted on the geometric transformation of point clouds using deep learning and GAN models. Based on this research, we propose to use P2P-NET as the generator and Point-NET++ as the discriminator. This choice is based on the fact that, at the time of this research, P2P-NET was the only model specifically designed for shape transformation, while Point-NET++ served as a reference for point cloud classification and segmentation tasks. Several experiments and tests were conducted, and the process is described in the following sections.

5.2.1 Datasets description

To validate the shape transformation proposed by (308), the shape transformation between different objects, and to validate that P2P-NET can approximate both the synthesized point

clouds of apple trees and the real point clouds of apple trees. 3 datasets were used. The first dataset is the one presented in (308) to validate the performance of P2PNET. Figure 62, shows an example of the geometries or classes considered in this dataset. This dataset has point clouds of 4 main geometries (beds, sofa, chair, plane), each geometry is described in turn by two types of points, the first being a surface and the second being an abstraction representing the main axes of the geometry (skeletons or crosses). Each point cloud has 2048 points, and each dataset contains 330 elements.

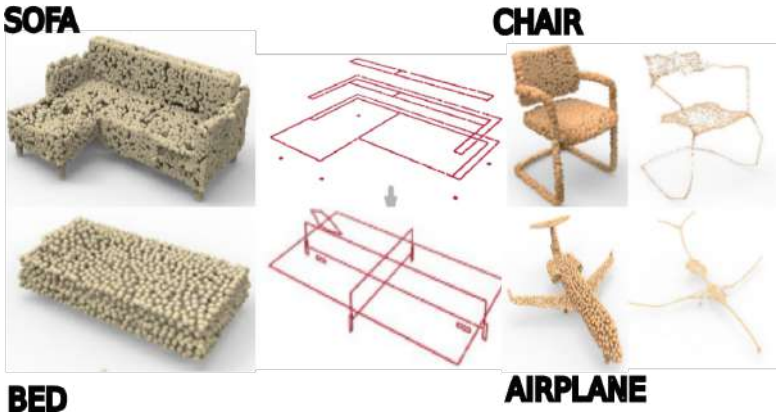


Figure 62 – Example of the geometries given in the dataset created by (308). Figure modified from (308)

The second dataset consists of synthetic trees. The point clouds of these trees are annotated. They have three defined classes (trunk, leaves, apples), were created with a resolution of 0.15°, and the original point range varies from thousands to millions of points. The protocol used is the *LowRes* protocol (see section 4.1). For initial testing, the annotations of this dataset were removed and the point clouds were sampled down to 2048 points using the farthest point sampling technique (see section 2.5.2). An example of the point clouds can be seen in figure 63

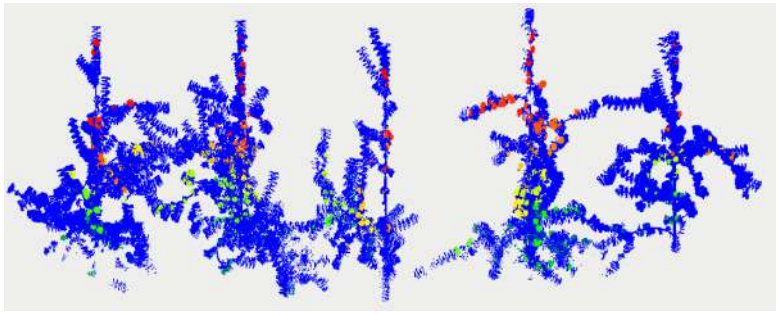


Figure 63 – Example of a row of five synthetic apple trees scanned with the simulated LiDAR sensor. Leaves and trunk are shown in blue. Individual apple clusters are represented by colors ranging from green to red, with green representing cluster 1 and red representing the Nth cluster.

The third dataset consists of the terrestrial scans described in section 3.1.3.

The above data sets will allow to evaluate the level of abstraction as well as the constraints of P2P-NET. On the other hand, they will give the guidelines for the modeling and propose the model that will allow the creation of realistic synthetic trees.

5.2.2 Approximating the fruit trees point clouds using shape transformer

PP2P-NET tackles tasks such as transforming a car shape into an airplane shape, or estimating the skeleton of a given geometry and vice versa. P2P-NET is known for its bi-directional analysis of input and predicted geometries. It not only estimates the loss function of the predicted geometry, but also has the ability to convert the predicted geometry back to its original form. To prevent overfitting, the model incorporates a regularisation function that evaluates the relationship between predictions and limits the number of point projections. The overall architecture of this neural network is shown in Figure 64.

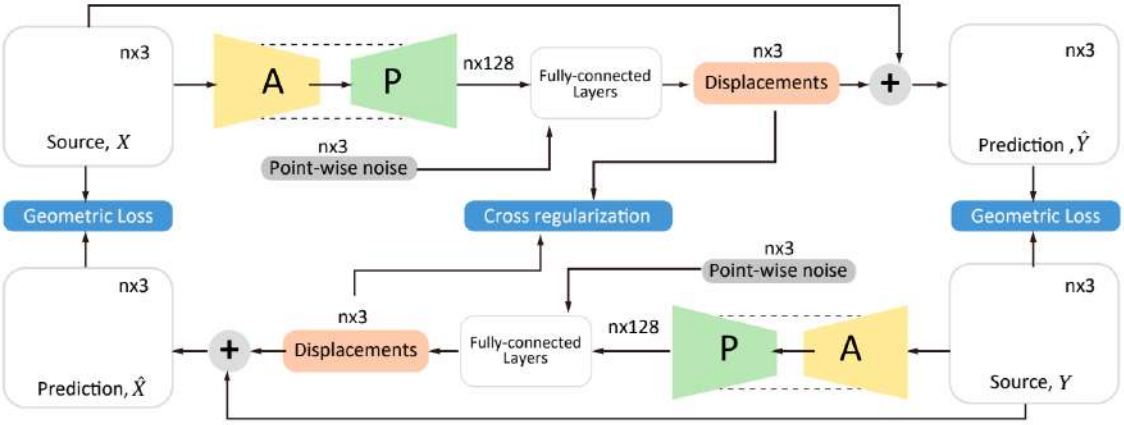


Figure 64 – P2P-NET architecture. Figure taken from (308)

Figure 64 illustrates two modules, A and P, from the original implementations of PointNet++. Module A represents the Abstraction layer, which consists of a group of neurons responsible for extracting characteristics from the point clouds. Module P, on the other hand, corresponds to the propagation operation that applies a hierarchical operation to the extracted features in order to restore the original size of the point cloud.

The Abstraction layer A is composed of three operations distributed across different layers. The first operation is *point sampling*, which utilizes the farthest point sampling (FPS) algorithm (see Algorithm 1). In this algorithm, a point is selected (*sampleIndex*) from the point cloud, then, distances between the selected point and the remaining points are estimated, and the point with the maximum distance is chosen to be the next reference. This process is repeated until the desired number of points (*npoint*) to be sampled is reached. The FPS algorithm enables the extraction of the most representative points of the geometry, regardless of potential noise and sparsity in the point cloud. This capability is particularly beneficial when describing and generalizing a geometry.

The grouping procedure is accomplished using the ball query algorithm, which focuses

Algorithm 1 Farthest Point Sampling. From (309)

$\mathbf{P} \leftarrow PointCloud_{NxM}$ $\mathcal{F}$ Point cloud of length N and with M feature dimensions
$npoint \leftarrow 5000$ $\mathcal{F}$ Number of points to get sampled
$\mathbf{D}[N] \leftarrow [1e10]$ $\mathcal{F}$ Point distance initialization
$sampleIndex \leftarrow randomInteger(0, N-1)$
$\mathbf{S}[\mathbf{0}] \leftarrow [sampleIndex]$ $\mathcal{F}$ Index vector initialization
$dist \leftarrow 0$ $\mathcal{F}$ Temporal variable do keep the calculate point distance
for $i \leftarrow 1$ to $npoint - 1$ **do** $\mathcal{F}$ Get the number of desires subsample points
  for $point2compare_{index} \leftarrow N - 1$ **do** $\mathcal{F}$ Get the distances
    $dist \leftarrow calculateDistance(P_j, P_{sampleIndex}, D)$
    if $dist \leq D[j]$ **then**
      $D[j] \leftarrow dist$
    end if
  end for
  $sampleIndex \leftarrow argmax(\mathbf{D})$
  $\mathbf{S}[i] = sampleIndex$
end for

on space partitioning and enables the identification of all the points within a specified radius. It is employed because it allows the definition of a consistent neighborhood scale, facilitating a more effective abstraction of the given geometry and creating a receptive field that depends on the defined geometries (239; 310). As explained by (239), this layer receives as input the representative points from the sampling layer, which is of size $N \times (d + C)$ where N is the length of the point cloud, d represents the n dimensions of the centroids, and C represents the features of the related centroids. The output of this layer will be K times the size of the input, due to the point grouping being done along each of the estimated selected points.

Point-Net (238) layers are then applied to obtain the feature map representing the geometry. In this process, Point-NET utilizes a multilayer perceptron model to obtain a set of activation functions that characterize the target geometry. Subsequently, a max-pooling operation is performed to extract the most significant features. The described procedure can be represented by Equation 5.2. In this equation, h and γ represent the application of the multilayer perceptron to each point in the point cloud, MAX denotes the max-pooling operation applied to the original feature map and $\mathbf{X}$ represents the point cloud of size $N \times M$.

$$f(\mathbf{X}) = \gamma(MAX(h(\mathbf{X}))) \tag{5.2}$$

It is important to note that each group $\mathbf{K}$ of points is transformed to the plane of each centroid and then re-expressed as a value in the feature map. Therefore the output will be $N \times (d + C)$ in size. A general representation of the described process for the feature extraction can be seen in Figure 65.

The feature propagation layer P aims to restore the size of the input point cloud after the feature extraction and abstraction process. To accomplish this, (239) proposed a model

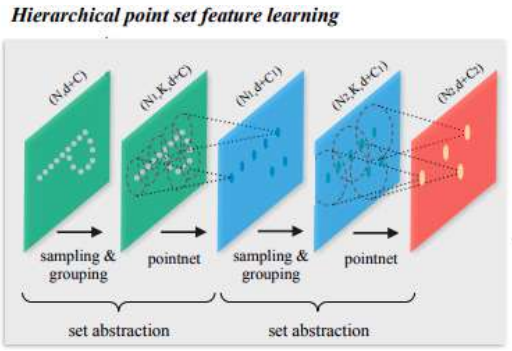


Figure 65 – Point-NET++ feature extractror. Figure was modified from (239)

based on hierarchical feature map interpolation as shown in Figure 66.

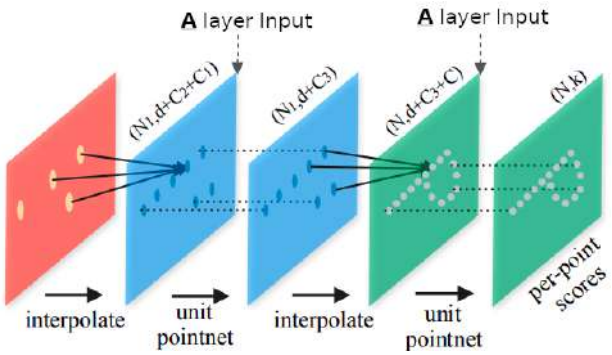


Figure 66 – Propagation layer of Point-NET++. Figure modified from (239)

This approach makes it possible to improve the descriptors of a given point of interest by exploiting its relationship to other elements of the surrounding geometry or scene. Equation 5.3 describes how the interpolation is done along the feature space.

$$f^{(j)}(\mathbf{x}) = \frac{\sum_{i=1}^{k} w_i(\mathbf{x}) * f_i^{(j)}}{\sum_{i=1}^{k} w_i(x)} \tag{5.3}$$

The equation 5.3 shows how the k neighbors of a point ($\mathbf{x}$) are re-expressed in the j layer. Note that $w_i(x)$ is a re-expression of the distance between the $\mathbf{x_i}$ point neighbors and the reference point $\mathbf{x}$. See equation 5.4.

$$w_i = \frac{1}{d(x, x_i)p} \tag{5.4}$$

Once the geometries have been represented as a feature vector, a white noise matrix $\mathbf{B}$ of size $M \times 3$ is generated and added to the feature vector. This is done to reduce the overfitting of the model and to ensure variability in the estimation of point transformations. The resulting vector is then fed into a fully connected neural network (ξ). The task of this neural network is to learn the relationship between the set of features and how they relate to the vector of transformations that will be applied to the input point cloud to approximate the new geometry. This operation can be represented in equation 5.5.

$$\hat{\mathbf{P}} = \xi(f(\mathbf{X}) + \mathbf{B}) \tag{5.5}$$

Once the new geometry is obtained, the process is repeated, but in this case, from the generated geometry back to the original base geometry. During this iterative process, the loss function (G_{loss}) is computed, and a regularization term ($r(x)$) is applied to ensure smoothness and coherence in the reconstructed geometry. The loss function (G_{loss}) is calculated by adding two separate loss functions. First, the shape loss function (L_{Shape}), see equation 5.6. This loss function is utilized to measure the shape dissimilarity between two point clouds.

$$L_{shape}(\hat{\mathbf{Y}}, \mathbf{Y}) = \sum_{p \epsilon \mathbf{Y}} \sum_{q \epsilon \hat{\mathbf{Y}}} \min d(p, q) + \sum_{q \epsilon \hat{\mathbf{Y}}} \sum_{p \epsilon \mathbf{Y}} \min d(q, p) \tag{5.6}$$

Equation 5.6 compares all the points p of the transformed geometry ($\hat{\mathbf{Y}}$) with all the q points of the target geometry ($\mathbf{Y}$) and vice-verse, by identifying the points with the smallest Euclidean distance (d) between the two point clouds.

Secondly, the density loss function (L_{density}) enables a closer approximation of the target point cloud by ensuring that the density of the points is properly represented. This loss function is defined by equation 5.7.

$$L_{density}(\hat{\mathbf{Y}}, \mathbf{Y}) = \frac{1}{k} \sum_{p \epsilon \mathbf{Y}} \sum_{i=1}^{k} |d(\mathbf{p}, N_i(\mathbf{Y}, \mathbf{p}) - d(\mathbf{p}, N_i(\hat{\mathbf{Y}}, \mathbf{p}))| \tag{5.7}$$

In the equation 5.7, the density is defined as the error in the distance between the closest points that exist between the target ($\mathbf{Y}$) and the approximation ($\hat{\mathbf{Y}}$) divided by the k points of most interest. To find such points, defined as $N_i(\mathbf{A}, \mathbf{p})$, a $k - Dtree$ was used which allows searching for the closest point of both $\mathbf{Y}$ and $\hat{\mathbf{Y}}$ to a point p.

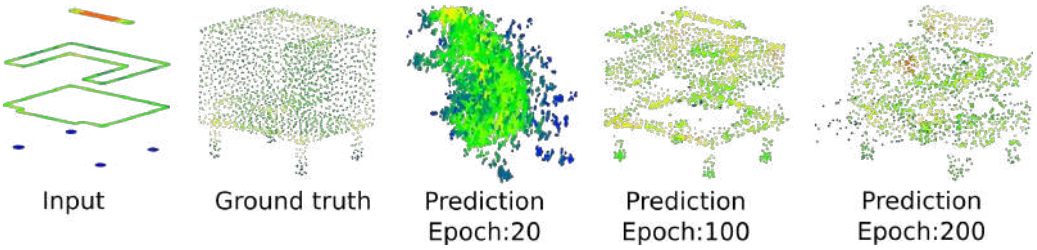
Finally, the loss function is given by equation 5.8. It represents the difference in both shape and density of two point clouds from a given dataset (D).

$$L_{\mathbf{X} \rightarrow \mathbf{Y}}(D) = \sum_{(X,Y) \epsilon D} (L_{shape}(\hat{\mathbf{X}}, \mathbf{X}) + \lambda L_{density}(\hat{\mathbf{Y}}, \mathbf{Y}) \tag{5.8}$$

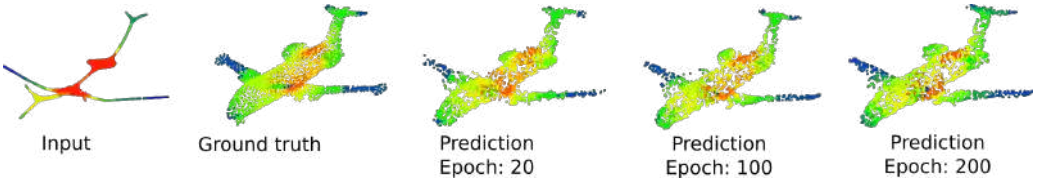
5.2.2.1 From a given shape, can we approximate a different shape?

To validate that P2P-NET can approximate point clouds that have no geometric relationship and do not belong to the same class, and to verify that it is possible to approximate the shape of trees from other shapes. Several experiments were carried out.

The **first experiment** consists on replicated the experiments carried out by the authors who developed P2P-NET. In this test, the aim is to approximate the shape of an object from a different representation of it, such as approximating the surface of a chair from its geometric skeleton. For this experiment, the default hyperparameter values of P2P-NET were used. And the point clouds used correspond to those proposed by the researchers who developed



(a) **Experiment 1:** Approximation of the Sofa cross skeleton to its surface.



(b) **Experiment 1:** Approximation of the airplane skeleton to its surface.

Figure 67 – Example of approximating the surface of two different shapes from two different skeletons.

P2P-NET. As an example, we tried to approximate the skeleton of a sofa to the surface of the sofa (see figure 67a). In this experiment, it was observed how P2P-NET learns the target geometry. First deforming the initial geometry in a certain range and from this deformation, the geometries start to approximate each other little by little. From the tests carried out, it was observed that P2P-NET is able to learn the transformation of geometries. In our case, we were not able to accurately approximate all the results presented by the P2P-NET authors. As shown in figure 67a, the geometries show deformations or surfaces with holes in them. However, in the case of Figure 67b, a better representation of the target geometry is observed.

The **second experiment** is to validate P2P-NET's ability to learn the transformations necessary to approximate two shapes that have no geometric relationship and belong to different classes. In this test, the standard P2P-NET parameters are used and the data set developed by the creators of P2P-NET is used. For example, in this test, we tried to approximate the geometry of a sofa to the geometry of a chair (See Figure 68). This test confirmed that P2P-NET is able to learn the necessary transformations to approximate different geometries. However, it was observed that it is necessary to modify the hyperparameters of the model in order to obtain a desirable solution.

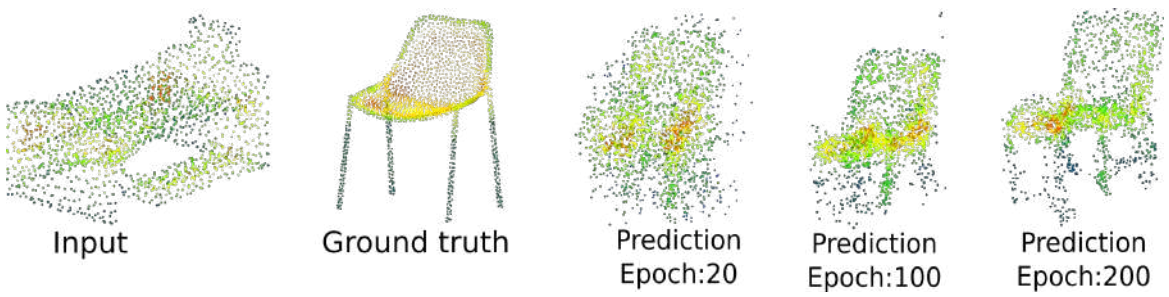


Figure 68 – **Experiment 2:** Approximation between two different geometries, they are centered in the same coordinates and in the same scale. The transformation from bed to chair.

In the **third experiment**, we aimed to verify that P2P-NET is able to approximate the shape of synthetic apple trees starting from a different shape. This experiment is required to validate the P2P-NET's ability to approximate topologies and geometries of great complexity, such as those of apple trees. For this experiment, a dataset of 132 shapes was used, consisting of 66 points clouds in the shape of different types of airplanes and 66 points clouds of different apple trees. In this experiment, both the point clouds of the planes and the point clouds of the apple trees have geometries described by 2048 points. The point clouds of the apple trees were not aligned with the point clouds of the airplanes, meaning that there is an offset between the center of the geometries. In the original dataset, each of the geometries is centered in the same coordinates. In this experiment, the model was not able to approximate the target geometry, the predictions made by the model have an offset with respect to the target geometry. It was also observed that the displacement of the points with respect to the predicted geometry is encapsulated in a specific volume that depends on the hyperparameter max range. From this experiment, we concluded that it is necessary to align all the geometries in the same coordinate, that the max range has to be adjusted depending on the relationship between the geometries, and that it is necessary to use objects with similar scales. An example of the results can be seen in Figure 69

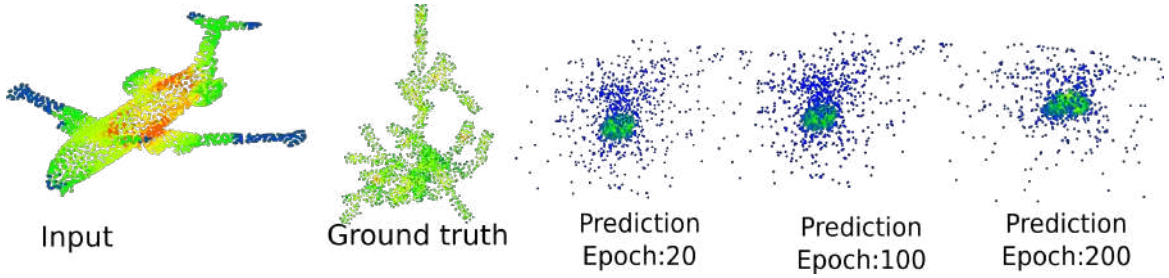


Figure 69 – **Experiment 3:** Approximating the geometry of a tree from the geometry of an aircraft. The trees have not been centered, nor scaled with the aircraft dataset.

The **fourth experiment**, like Experiment 3, tests whether it is possible to approximate the shape of the apple tree from another shape. For example, the shape of an airplane. The difference with experiment 3 is that in this case the center of the apple tree point clouds is aligned with the center of the airplane point clouds. The hyperparameters of P2P-NET are the default, each point cloud consists of 2048 points. It was observed that the model does not approximate the geometry of the trees. The predictions were centered on the same axis as the input geometries. The model gradually adjusts the volume of the predicted clouds in each iteration until a point cloud is obtained with a volume similar to that of the geometries to be approximated. An example of the predicted geometries can be seen in Figure 70.

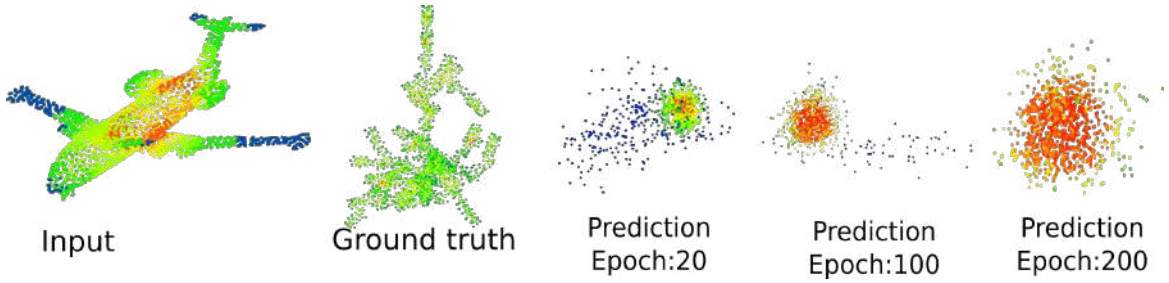
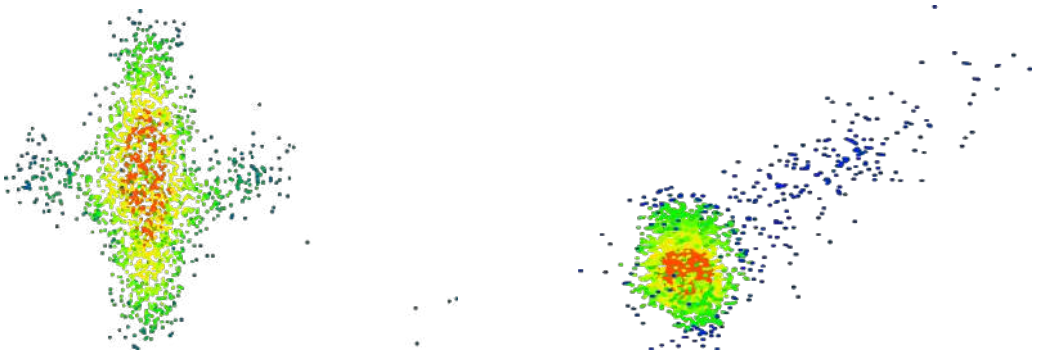


Figure 70 – **Experiment 4:** Approximating the geometry of a tree from the geometry of an aircraft. The trees have been centered and scaled with the aircraft dataset. Default parameters of P2P-NET

In the **fifth experiment**, the aim is to understand the role of the hyperparameters in the performance and ability of the model to approximate the desired geometries. In this case, the same data set is used as in Experiment 4. The tuning procedure involved systematically varying each parameter individually and observing the behavior of the model during the training process and its predictions after every few epochs. During this process, I observed the impact of each parameter on the efficiency and stability of the model when learning and predicting a specific geometry. The effect of each of the hyperparameters according to the experiment can be described as follows.

1. *Max range:* This parameter sets the maximum distance at which points can be moved. The effect of this parameter can be observed mainly at the beginning of model training when the input geometries are deformed. And its effect is also observed in the ability of the model to find the transformations to cover a target of larger dimensions than those of the input point cloud. See figure 71.



(a) Max range set to 3 (b) Max range set to 10

Figure 71 – Example of how the Max Range hyperparameter deforms the input geometries.

2. *Decay epoch:* This parameter defines how many epochs the learning rate should be reduced. This parameter is important to ensure some stability of the model. Depending on the value set, the model may start to have several instability peaks in advanced phases of training. Or it may be that the value is too small and the model stops learning after a certain number of epochs because the learning factor is too close to zero at an early stage of training. An example of the described pattern can be seen in figure 72.

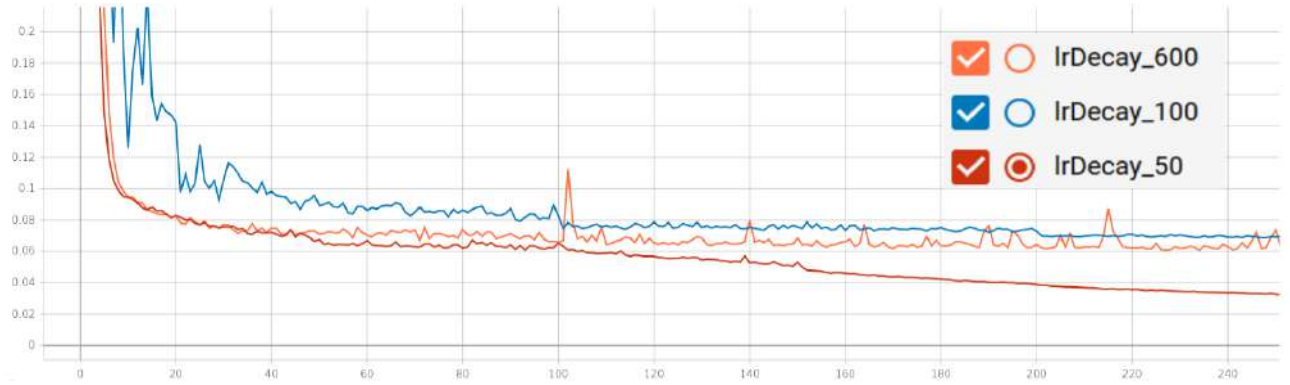


Figure 72 – Effects of different decay epochs on the process of training.

3. *Density weight:* This parameter is a regularisation factor that determines how much weight is given to the density loss function. It is important because the point density can be approximated but the geometry is not taken into account.

4. *regularization weight:* This parameter was implemented to prevent model over-fitting and to maintain control between the models being trained.

5. *NNK (K-nearest neighbors):* This parameter defines the number of points to be taken into account when extracting the characteristics of the different parts of the point cloud.

6. *Radius scale:* This parameter sets the radius in feature space that pointNET++ should take into account when abstracting features. This parameter can be thought of as an element that controls how accurate the feature map representing a given geometry will be.

7. *Noise length:* Is the size of the white noise vector added to the features extracted by Point.

We concluded that P2P-NET is able to approximate the point clouds of 2 different objects. P2P-NET tends to approximate the general characteristics of the target dataset. The hyperparameters of P2P-NET have been modified taking into account the geometric characteristics of the input and target datasets and this tuning helps to approximate the shape of the apple tree point clouds. However, when the geometries are approximated, unrealistic geometrical deformations are observed in the predictions, which deviate the point clouds from their realism.

Based on the results obtained and the lessons learned from the different experiments in which the model was applied. I can conclude that P2P-NET is a model that can serve as a generator in the GAN (Generative Adversarial Network) model to be developed. The model is able to partially approximate apple trees, and it is hypothesized that the model will improve its performance with the help of the discriminator.

5.2.3 Approximating classification between synthesized and real point clouds using a deep learning model

As explained in Section (2.7.1), a GAN model typically consists of two models (a generator and a discriminator) that compete with each other. This competition allows the models to improve their performance in their respective tasks (generating realistic data and discriminating between synthetically generated and real data). In Section (5.2.2), P2P-NET was presented as the model that will perform the generator task. In this section, we will present the model selected to perform the discriminator task. The discriminator is the model responsible for determining whether the input data is real or generated. Considering the differences described in Section 5.1.2 regarding synthetic and real point clouds of apple trees, along with the need for a classification model for the discriminator of the GAN, and the assumption that a deep learning model can accurately distinguish between real and synthetic trees, we selected the Point-NET++ classifier for this task. This model was chosen because it can process raw point clouds and was still considered a reference in classification and segmentation tasks at the time of this research. Additionally, as Point-NET++ is the basis for P2P-NET, making their integration is relatively straightforward.

To build a classification model based on the Point-NET++ feature extractor, we need to apply a fully connected neural network (FNN) at the end of the Point-NET++ feature extractor, as described by (239) (Figure 73). It is important to note that the output layer of this neural network must have a size of $C + 1$, where C is the number of known classes and the additional class allows the neural network to classify unknown geometries.

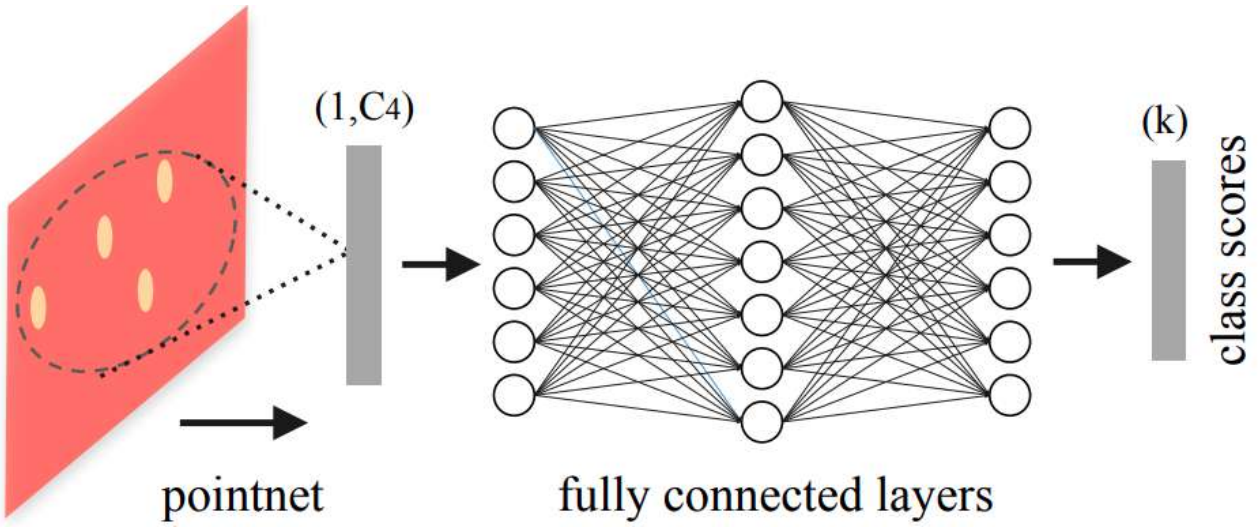


Figure 73 – Classifier applied in the Point-NET++ model. The figure was modified from (239). the red square with yellow dots represents the final feature map after the feature extraction process, made by each of the point-NET++ layers. And the fully connected neural network represents the classifier.

In order to evaluate the ability of the classifier based on Point-NET++ to discriminate between real and synthetic data, we carried out 4 experiments. The **First experiment** consists

in replicating the experiments carried out by (239). The experiment consists in training the model to discriminate well between 40 different classes given by the modelnet40 dataset. The modelnet40 dataset consists of 40 classes, the total size of the dataset is 12311 and it is divided into 80% for training and 20% for testing. Figure 74 shows the results obtained in this experiment. This figure shows that the model is able to classify the 40 classes of the dataset without any problems. However, the results obtained are not equal to those obtained by the authors of the models in all classes. The accuracy value for all classes is greater than 0.60.

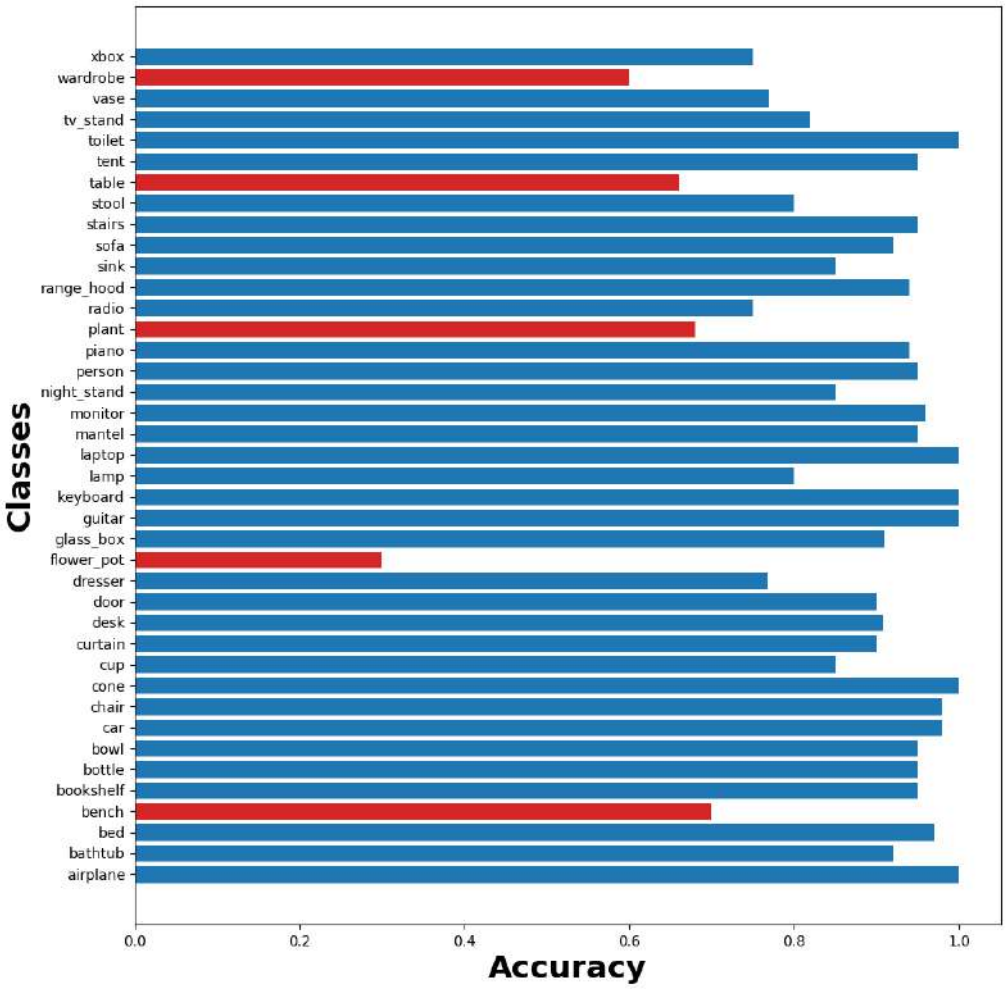


Figure 74 – **First experiment:** Barplot of the classification accuracy of Point-NET++ applied to the modelnet40 dataset. The red bars represent the classes for which the model has a precision of less than or equal to 0.7 and the blue bars represent the classes for which the model has a precision greater than 0.7.

In the **second experiment**, we aimed to validate if PointNet++ is capable of discriminating point clouds of apple trees. For this test, we replaced the *flower pot* class of the ModelNet40 dataset with the real apple tree point clouds dataset. The dataset of apple trees consists of 330 point clouds. PointNet++ was trained to discriminate between the 39 standard classes of ModelNet40 and the new apple tree class. The model successfully discriminated between the 39 default classes with an accuracy greater than 0.6, and it achieved an accu-

racy greater than 0.90 for the new apple tree class. Refer to Figure 75 for a visual comparison of predictions. Based on the results obtained, it is concluded that PointNet++ is capable of discriminating LiDAR sensor point clouds of apple trees. This result was expected since the point clouds of apple trees exhibit specific geometric characteristics.

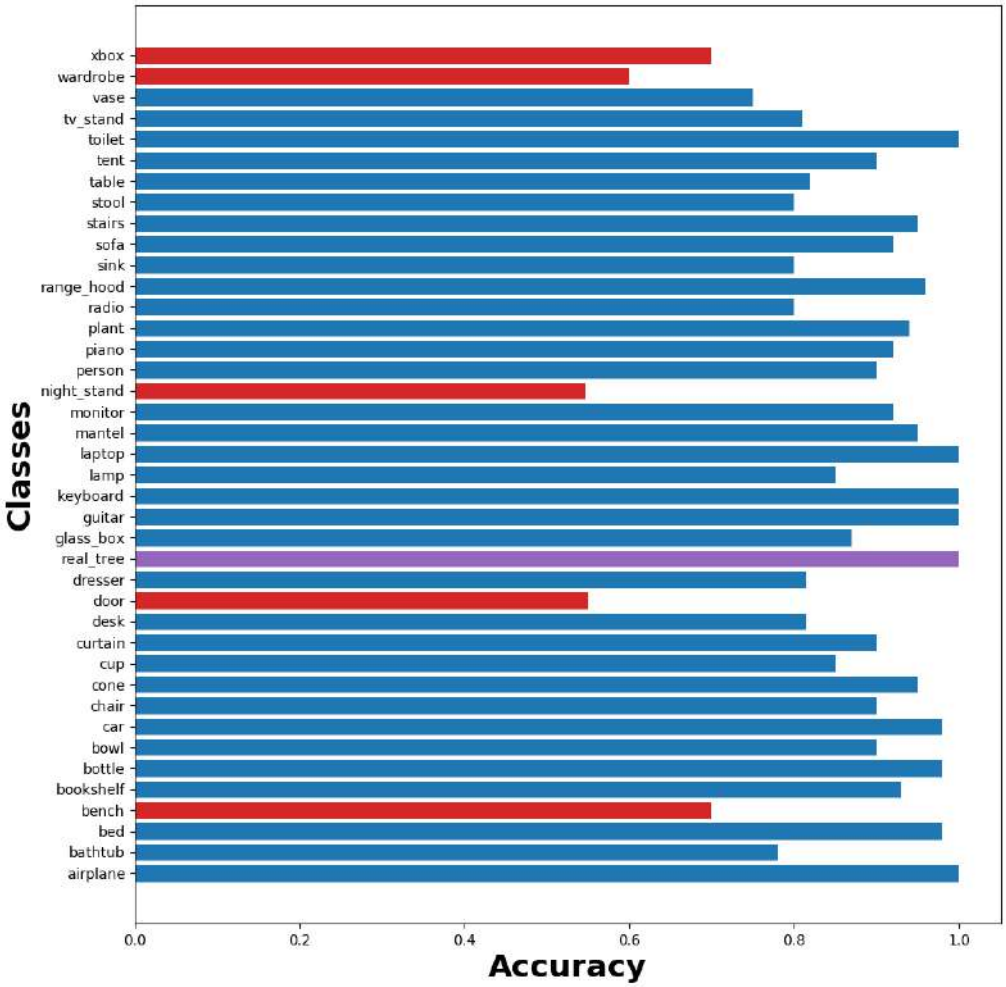


Figure 75 – **Second experiment:** Each of the bars represents the accuracy of the model in classifying different shapes. The red bars show the shapes with an accuracy of less than 0.70, the blue bars show the shapes with accuracy values greater than 0.70. The purple bar shows the accuracy of the model for real tree shapes.

In the **third Experiment** we want to validate whether Point-NET++ is able to discriminate the point clouds of synthetic apple trees from the other modelNET40 classes. The synthetic tree dataset is composed of 330 tree point clouds. To perform this test, one of the standard classes (flower pot) of the dataset was replaced by the synthetic tree dataset. We then proceeded to train and make the corresponding predictions. In this test, it was observed that the model discriminated all classes of the modelnet40 with an accuracy greater than 0.6. and for the new class tree, with an accuracy greater than 0.93. See figure 76. It is concluded that the model is able to discriminate synthetic apple trees. On the other hand, it was observed that the accuracy value of the other classes of the model net40 varied compared to

the previous experiments.

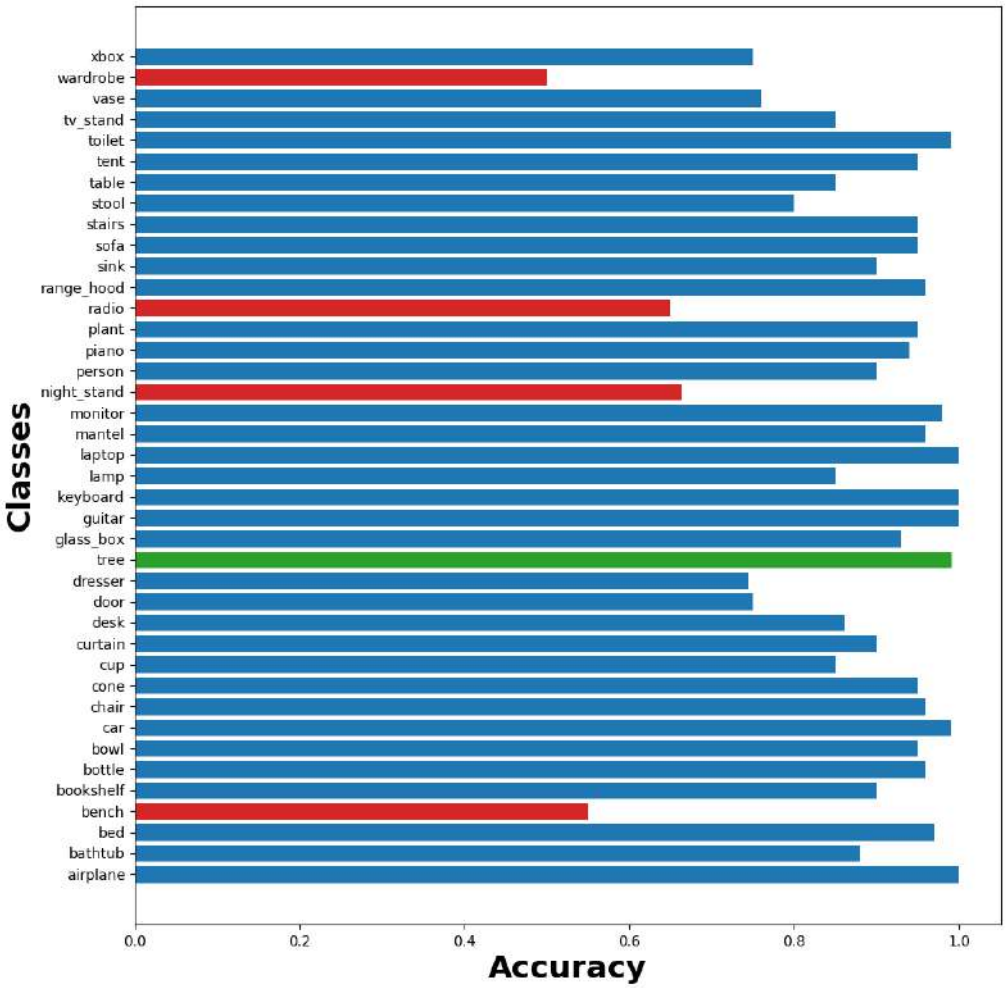


Figure 76 – **Third experiment:** Each of the bars represents the accuracy of the model in classifying different shapes. The red bars show the shapes with an accuracy of less than 0.70, the blue bars show the shapes with accuracy values greater than 0.70. The green bar shows the accuracy of the model for synthetic tree shapes.

Finally, in the **fourth experiment**, we propose to validate whether PointNet++ is capable of discriminating between synthetic and real tree point clouds, we used a dataset consisting of 330 real trees and 330 synthetic trees. The dataset was split, with 80% of the trees used for training the model and the remaining 20% used for testing. As shown in figure 77 the model demonstrated an accuracy superior to the 90% in classifying the two types of geometries. Therefore, we conclude that the model is indeed capable of discriminating between synthetic and real tree point clouds.

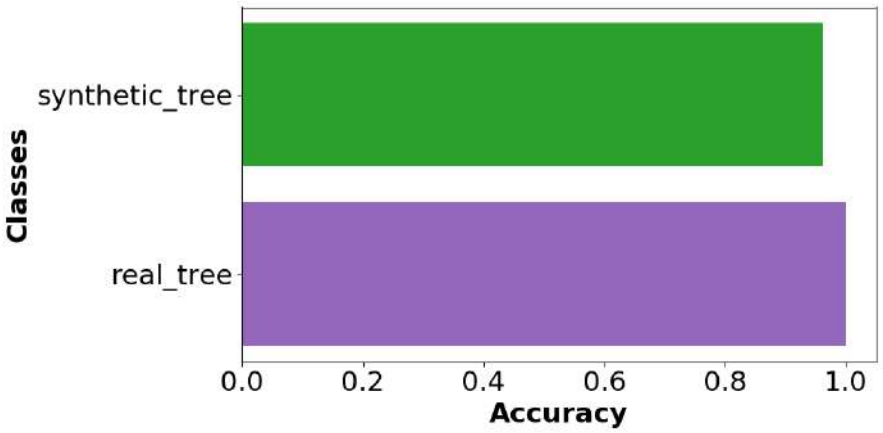


Figure 77 – **Fourth experiment:** bar chart of the accuracy of the model when trained and tested only to classify synthetic and real trees.

From the results obtained above, we concluded that the Point-NET++ model is able to correctly discriminate both synthetic and real point clouds. Furthermore, we validated that this model will serve as a discriminator for the GAN model to be developed.

5.2.4 Enhancing Synthetic Trees through Generative Adversarial Networks (GAN)

In order to develop a model capable of replicating the architecture of apple trees and simultaneously reproducing the distribution of LiDAR sensor points, it was proposed to create a GAN model. This GAN model aims to take the synthesized point clouds of apple trees and transform them into more realistic point clouds. The model consists of two neural networks: the generator and the discriminator. For this case, the generator's focus should be on shape transformation, as it is expected to reproject the synthesized point clouds into new, more realistic point clouds. On the other hand, the discriminator must be capable of detecting differences between real and synthetic point clouds based on their distinct point distributions. Figure 78 shows the general schema of the proposed model.

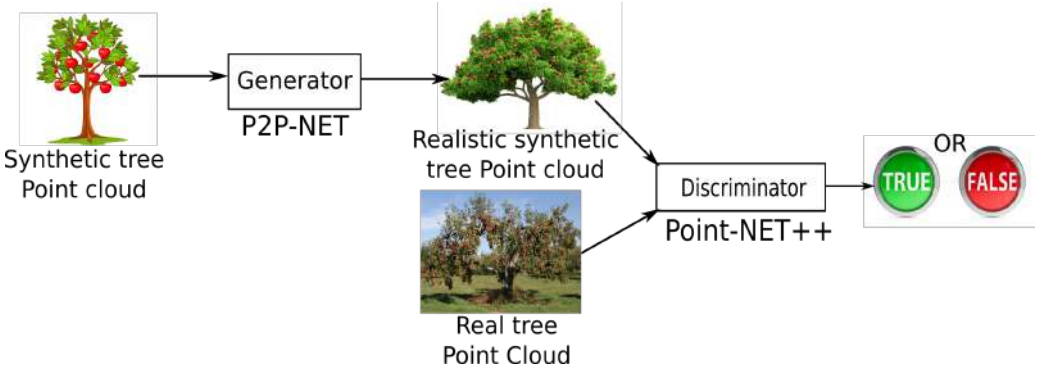


Figure 78 – The Base Architecture of a GAN Neural Network. A synthetic tree point cloud refers to the synthetic tree point cloud obtained using the MAppleT model and virtual LiDAR scan (Section 5.1.2). A real tree point cloud refers to the point cloud captured by the LiDAR sensor in an outdoor environment (Section 5.1.1). The realistic synthetic tree point cloud refers to the generated point cloud by the generator

More specifically, the proposed model aims to produce realistic fruit tree point clouds from the point clouds generated by MAppleT and PlantGL. This new point cloud is intended to approximate both the architecture and topology of the apple trees that were scanned using the LiDAR sensor. It's important to emphasize that this task is extremely challenging due to the distinct attributes found in synthetic and real point clouds. These attributes include the geometric variations of each organ, the diversity in the tree's topology, changes in density, and the various representations of geometries in the point cloud. These representations are influenced by factors such as tree genotype, environmental noise, measurement protocol, and the sensor used (as discussed in section 5.1.2). Furthermore, the limited number of individuals for each genotype in the dataset to be approximated, along with the geometric differences among these genotypes, further contribute to the complexity of this task.

In order to address the problem described in a controlled manner, several experiments were carried out to guide the development of the proposed model. These experiments are presented in the following sections.

5.2.5 Exploring Synthesized Point Clouds for Improved GAN Response

To validate the effectiveness of the proposed model, we aimed to approximate the geometry of a tree in summer based on the geometry of the same tree in winter. The goal of this experiment is to first test the ability of the model to generate organs for a given structure, and then to move on to more complex tasks such as annotation or generation of new topologies. For this purpose, we used a dataset containing 330 trees captured in both summer and winter. The dataset was split into 80% for training and 20% for testing.

The model's objective was to learn the underlying topology of the trees directly and then use this representation to generate new organs, such as fruits and leaves. An example of the result obtained from this process is shown in Figure 79.

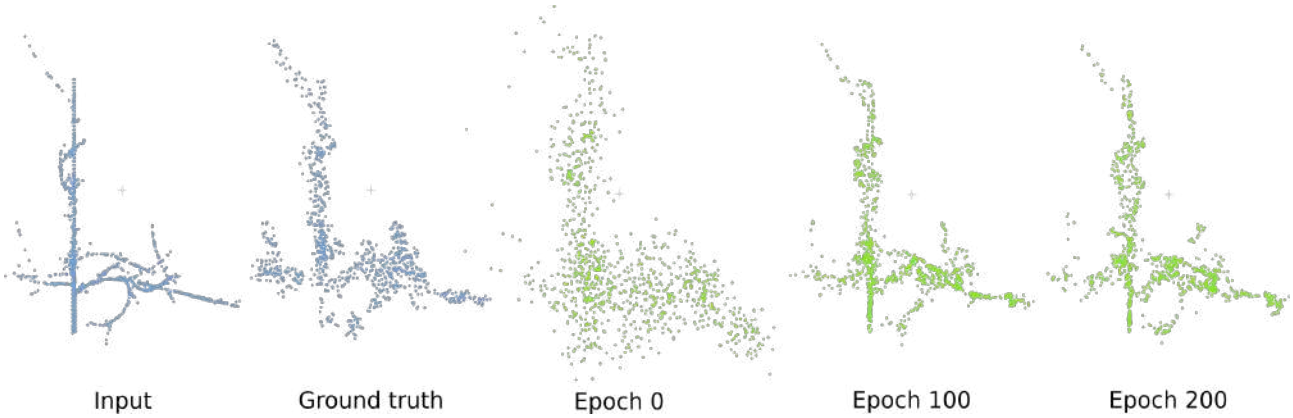
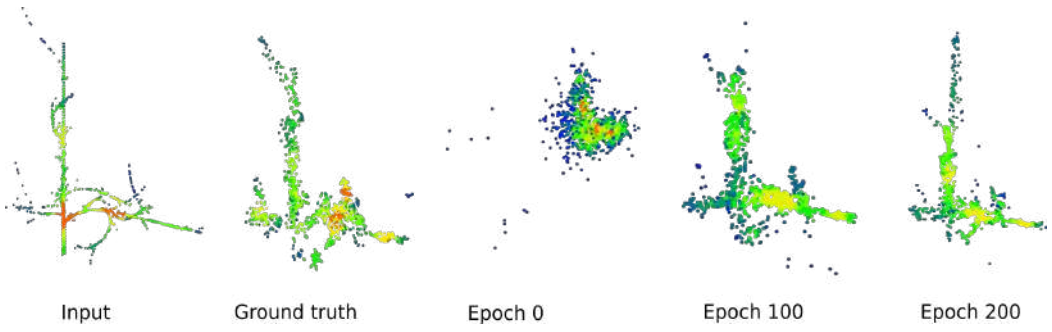


Figure 79 – Visual representation of the input and predictions made with the first proposed architecture and using low-resolution point clouds. The input corresponds to the winter tree and the ground truth is the same tree in summer.

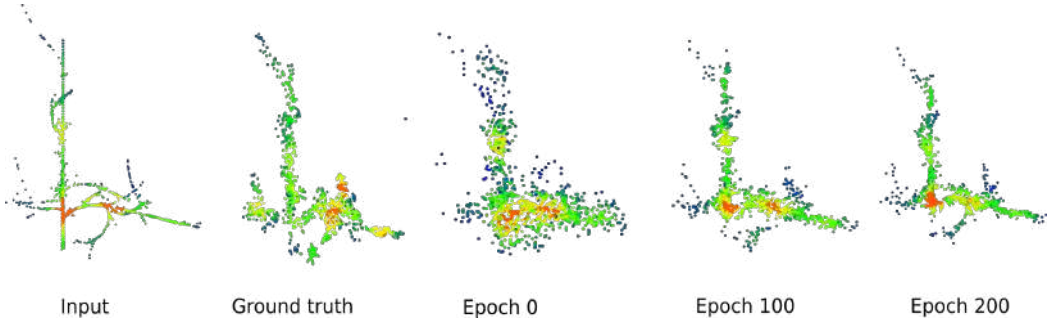
Figure 79 shows the input (a winter tree without leaves and fruit), the ground truth (an

apple tree in summer with leaves, trunk, and fruit), it is important to note that the summer tree point cloud has been sampled to have the same number of points as the input point cloud, which affects its resolution, and there are also shown two examples of the predicted geometry in different epochs of training. In epoch 0, the input geometry is expected to be completely deformed. In epoch 200 it can be seen that the model has already learned how to project the points to ensure the shape of the tree and the density. However, there are some errors, misplaced points are observed, the distribution of points in the trunk of the central part of the geometry is denser and the branches of the upper part are not well defined. From the above, we can conclude that the model is able to learn the topology of apple trees and is able to locate points close to the position where organs such as leaves and fruits should be, but does not approximate the details of these geometries.

In order to validate the effect of the different loss functions and how the new classification loss function affects the prediction of the geometries, it was proposed to train and optimize the model taking into account the sum of the different loss functions. To this end, the model was trained using the following operations *Density loss function + Classification loss function*, *Shape loss function + Classification loss function* , The description of the loss functions can be found in section 5.2.2. A dataset containing 330 synthetic trees from both the summer and winter sessions was used for training. From this set, 80% was used for testing and 20% for validation. Overall, it was observed that the model is able to approximate the geometry of a point cloud using the shape loss function plus the discriminator loss function. When the generator is optimized using the *Density loss* + the *Classifier loss* deformed geometries are generated. Figure 80 shows the described behaviours.



(a) Model weights updated using density loss plus classification loss



(b) Model weights updated using shape loss plus classification loss

Figure 80 – Effects of adding different loss functions when generating geometries

Based on the above results, I conclude that defining both the density loss function and the shape loss function is necessary to ensure that the approximated point cloud retains its geometric properties and point distribution. It's worth noting that while it is possible to approximate a geometry using only the shape loss function, the resulting shape may exhibit irregular point distribution patterns. Conversely, attempting to approximate a shape using only the density loss function is not viable. In addition, the learning rate emerges as a critical hyperparameter to consider when approximating a geometry. Depending on its value, the model may either fail to converge or approximate geometries with significant noise variations.

5.2.6 A GAN-Based Approach for Generating Annotated Synthesized Point Clouds

Having been able to approximate the point clouds of the apple tree in summer from the point clouds of the apple tree in winter, the next question was whether it would be possible to predict the class of each point in the summer apple tree's point cloud. The aim was to ensure that each approximate point cloud would have a label representing each of the tree's organs (e.g. leaves, trunk, apples). To achieve this, it was proposed to modify the P2P-NET input and output layers. This modification initially consisted in defining as input and output a tensor of size $NxMx4$, where N represents the number of point clouds of each batch, M indicates the number of points that each point cloud represents, and the definition of 4 is given because it is expected that 3 features related to the X, Y, Z coordinates of each point will be introduced, and 1 extra column that will represent the label of each point.

To validate the above hypothesis, I proposed an experiment consisting of approximating a fully annotated synthetic summer apple tree point cloud from the same tree in winter. The point cloud of the summer tree contains 3 annotations (1 for trunk, 2 for leaves and 3 for apples) and the winter point cloud contains only 1 annotation 1 for trunk. The dataset for this experiment consisted of 660 elements, 330 for winter trees, 330 for summer trees, of which 70% was used to train the model, 10% for the validation process during training and updating the loss function, and 20% for testing. When training and testing the model with this approximation, I observed that the model did not correctly learn to generalize the task of approximating the point clouds of summer apple trees, and when the prediction process was carried out, point clouds were obtained with no shape in an interval of coordinates.

Based on the described results of the previous experiment, a new question was posed: *is it possible to get better results using a different label encoding?*. For the above, it is proposed to re-express the labels using *hot-encoding*, with this encoding the labels are re-expressed as a vector of three positions where $[0, 0, 1]$ is the trunk, $[0, 1, 0]$ are the leaves and apples are given by $[1, 0, 0]$.

Therefore, the P2P-NET input and output parameters were modified again, the model now has an input size of $NxMx6$, where N is the batch size, M is the size of the point cloud and 6 represents the input features, which are the X, Y, Z coordinates and the labels

represented using hot encoding.

During the learning phase, we observe that the discriminator learns to distinguish the differences between the real point clusters and those created by the generator without any problem.

However, it was observed that the generator was not able to approximate the geometries at any time and was also changing the scale of the point clouds in each epoch. At this point, we asked if it was possible that one of the loss functions was causing this scaling effect?. To verify this, it was suggested that the loss functions be evaluated independently and the evolution of each part of the model observed. It was noticed that the model tended to reduce the scale of the input point cloud without taking into account the used loss function and that it tended to cluster the points of the different classes. See figure 81

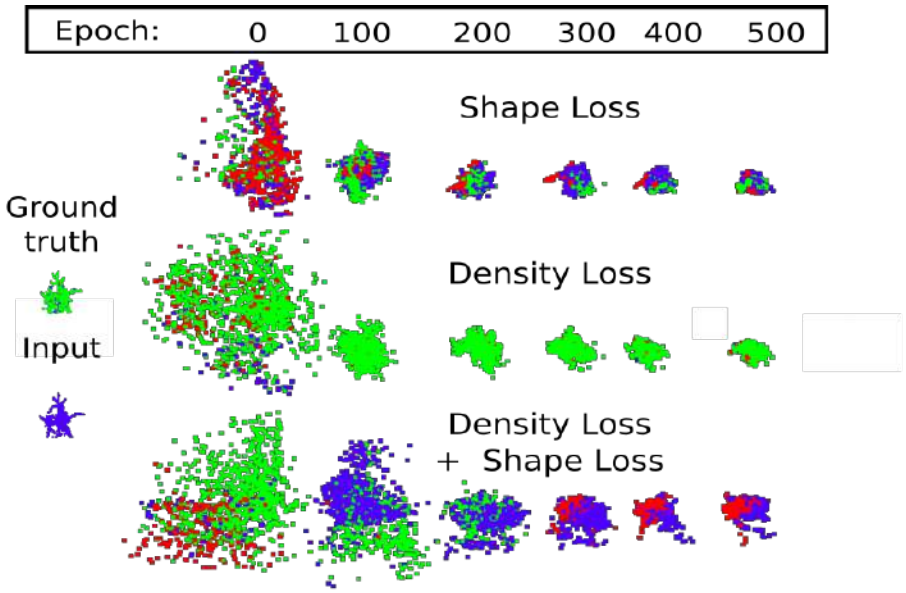


Figure 81 – Evaluation of loss function over annotated data

Observing the above behavior, we hypothesized that the dataset should be in unit space to facilitate the task of approximating the geometries and to avoid the constant change of scale at each epoch. To achieve this, the dataset used in the previous experiment was taken and each point cloud was divided by the maximum height of each geometry, which allowed the scale of the point clouds to be reduced. This was then used to train the model. In this particular case, it was observed that the model did not attempt to scale the point cloud, but failed to approximate the geometries. When this was observed, it was suggested that the hyperparameters be changed, but the only one that was observed to change was the learning rate. we observed that the higher the learning rate, the faster the model has to create the desired geometry, but if the learning rate is too low, the model will never converge. The results are shown in figure 82.

Figure 82 – Aproximation of geometries when the hot-encoded labels

Observing the previous behavior and knowing that the model was able to generate the desired geometries when it did not have the annotations as an input parameter, it is proposed to set all the annotations of the classes to 0 in order to validate if this behavior is given by the annotations. To achieve this, we took the dataset from the previous experiment and set all values in columns 3 to 5 to 0. It was observed that the model is able to generate the geometries and that very small values of the learning rate can cause the model not to converge. See figure 83

Figure 83 – Approximation of geometries when the hot-encoded labels on 0

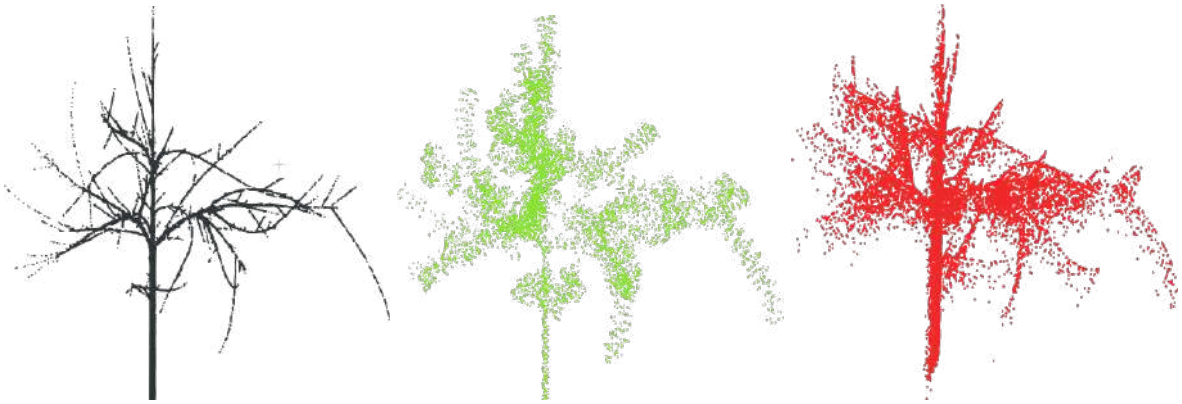
After observing the model's behavior in the previous experiments, it is proposed to integrate and regularize the loss functions of the model while keeping the labels in hot-encoding format. This aims to verify that the model updates correctly and enforces the generation of a target geometry. The new loss function is defined by Equation 5.9.

$$L_{\mathbf{X}\rightarrow\mathbf{Y}} = \sum_{(X,Y)inD} (\theta L_{Shape} + \beta L_{Density} + \alpha L_{Classification}) \tag{5.9}$$

For this experiment, the same dataset from the two previous experiments was used. The updated loss function assigns more weight to the geometry classification loss, while reducing the loss weights for density and shape. This adjustment is made to partially maintain the learning of the GAN models.

When training and prediction are performed with this updated loss function and the proposed architecture, the model is observed to approximate topologies reasonably well.

However, it struggles to approximate precise details of the apple trees, such as leaves and fruits. An example of such results can be seen in Figure 84.



(a) Input (b) Ground truth (c) Prediction

Figure 84 – Approximation of geometries with a high-density point cloud

As a general conclusion, the developed model serves as a good basis for approximating the general geometry of apple trees but faces challenges in accurately capturing specific details. Further work is required to approximate the desired results.

5.3 Conclusions and Discussion

A state-of-the-art review was conducted to provide a comprehensive understanding of current developments in Generative Adversarial Networks (GANs) for point clouds and how deep learning can be applied to the task of transforming point cloud shapes. In the state of the art, it was observed that GAN architectures typically retain the generator and discriminator structure. In some cases, there are models considered GANs, but they focus on simple structures resembling auto-encoders.

Moreover, based on my knowledge at the time of the state-of-the-art review, it was noted that these models can generate point clouds (311; 312; 313; 314). However, these point clouds lack annotations, and represent entire scenes, but exhibit limited geometric detail (312). None of these models were specifically designed for generating trees or improving upon previously available geometries.

Furthermore, the review revealed that shape transformations represent a novel approach. They primarily concentrate on reinterpreting geometry in alternative models or attempting to correct errors in the input geometry, as seen in tasks such as auto-completion and skeletonization.

Given the state of the art, it was considered to approximate a GAN model starting from a model focused on shape transformation. In this case, P2P-NET (308) was selected for the generator task, and Point-NET++ (239) was chosen for the development of the discriminator task. P2P-NET was chosen because it is a model completely dedicated to point cloud shape

transformation, while Point-NET++ was chosen because it was a reference in point cloud classification. Moreover, the integration of the two models could be done in a direct way, since P2P-NET is based on the feature extractor of Point-NET++.

In order to validate that the P2P-NET model is able to approximate the geometry of synthetic trees, different experiments with this model were proposed, ranging from replicating the experiments proposed by the authors of P2P-NET to approximating apple trees from shapes belonging to other objects. From these experiments, the importance of the model's hyperparameters was understood. It was observed that if the input geometry is too small compared to the geometry to be approximated, and the *maxrange* hyperparameter does not allow the points to be distributed at the desired distances, the model will return strange geometries far from the center of the desired geometry. It was observed that the *learning rate* hyperparameter allows the model to come out of a possible local minimum. Furthermore, the higher the parameter *nnk*, the better the approximation of the desired geometry. It was confirmed that the *noise vector* reduction helps to reduce the overfitting of the model. And that different *radios* affect the approximation of the model. On the other hand, it was observed that the model is able to approximate the shape of an object class from another object, but that this representation has different types of outliers and certain deformations. It was observed that it is possible to approximate tree geometries. The model has less difficulty in approximating low-resolution geometries.

After the validation of P2P-NET, we proceeded to the validation of Point-NET++. To validate this model, different experiments were proposed to understand and validate its behavior. Starting by trying to replicate the model sharing using the reference dataset, we were able to partially replicate the authors' results. The same accuracy values were not obtained for all classes. Furthermore, it was found that the model is able to discriminate between the point clouds of synthetic and real trees, and between these and other types of classes present in the original dataset. In general, the point clouds of synthetic trees and real trees were discriminated with an accuracy above 80%.

A GAN architecture based on P2P-NET and Point-NET++ has been proposed, several experiments have been developed and it has been shown that it is possible to generate the geometry of the apple trees. The point clouds generated are of low quality and contain various types of noise. It was observed that it is possible to add different loss functions to optimize the model. A modification of P2P-NET was proposed to try to predict the classes of each point in the point cloud, but the desired results were not achieved. Different representations of the labels of each class were tried. When the labels were represented as integers, the model did not converge. When trying to represent the labels as hot-encoded, it was observed that the model tended to change the scale of the point clouds.

A possible hypothesis is proposed to solve the problems observed when trying to generate annotated point clouds. Is it possible that the number of points defining the geometry is not sufficient to generate a representative feature map for the desired task?

In order to solve the above question, it is proposed to change the way the model is fed

with point clouds. The change is to feed the point cloud with voxels from each point cloud, which will allow the quality of the geometries to be maintained. See figure 85.

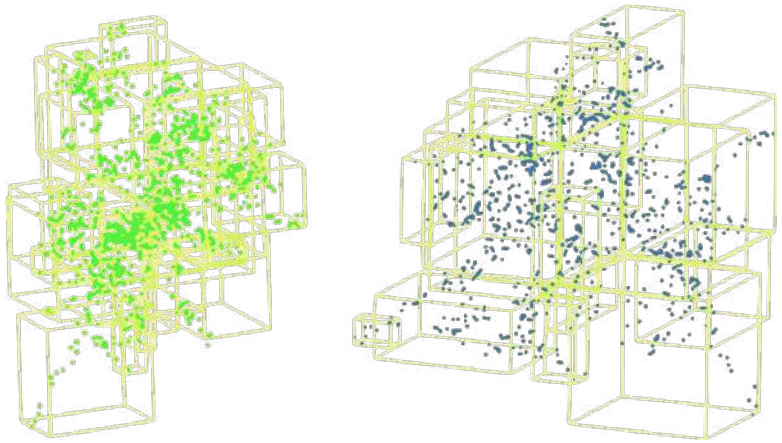


Figure 85 – Voxelisation of the point cloud of a fruit tree defined by 2048 points

A second proposal would be to feed the model with a number N of points from each part of the geometry to ensure the correct relationship between points and classes. Examples of the two previous proposals are shown in the figure below. See figure 86

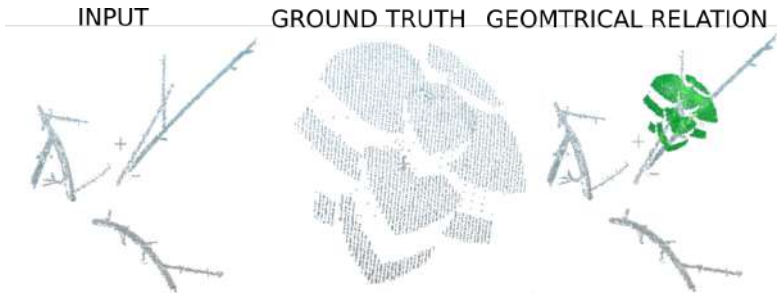


Figure 86 – Example of the selection of geometries to train the P2P-NET model based on the number of points.

CHAPTER 6

Conclusions and perspectives

During this thesis, I developed two pipelines for the processing and analysis of LiDAR point clouds of apple trees based on machine learning and deep learning techniques. As a third contribution, I also proposed the development of a GAN model to approximate the distribution of points of real LiDAR scans of apple tree from synthetic data with the aim of creating annotated dataset for the development of deep learning models for tree architecture development analysis. This model gives only for now promising results.

The **first pipeline** analyse and compare both ground-based LiDAR and airborne LiDAR point clouds of a non-industrial apple orchard for the task of architectural traits estimation. This processing includes filtering the two types of point clouds, removing unwanted elements, separating each tree independently, and extracting 6 volumetric architectural indices from the 3D representations of these trees.

From the indices extracted with this pipeline, a comparison was made between three aerial protocols and the indices extracted from the point clouds acquired with the terrestrial LiDAR were used as a reference. From this analysis, the aerial protocols with the lowest densities of point clouds provide the more accurate volumetric indices compared to terrestrial LiDAR measurements. This behavior is partly determined by the level of noise which seems less important than scans with lowest speed.

Compared to other research works such as those of (315; 316; 317) this pipeline focused on the analysis of aerial and ground-based LiDAR point clouds of apple orchards, and presents a semi-automatic scheme for the extraction of volumetric architectural indices. This differs from other works (e.g. (315)) by comparing two types of point clouds from two LiDAR sensors.

The **second pipeline** focuses on the segmentation and fruit counting of a non-industrialized apple orchard. This pipeline offers two methods for segmentation. The first one, is based on machine learning, with steps of feature extraction are made using FFPH and classification

uses Random Forest. The second one is based on deep learning, where RandLA-NET is applied and a training methodology is presented to improve its accuracy based on the optimization of learning from synthesized data. At the end of the pipeline, a clustering method is applied to count the segmented fruits. Starting from this pipeline, a series of tests have been performed to understand the sensitivity of the models to different levels of noise related to possible deformations of the geometries. And the sensitivity of RandLA-NET to different radiometric characteristics is tested. These tests showed that RandLA-NET is less sensitive than Random Forest to the noise present in the point clouds. This can be explained by the fact that RandLA-NET optimizes its feature extraction, while Random Forest uses precomputed features. RandLA-NET has also a higher accuracy than Random Forest in segmenting both synthesized and real data.

To perform the clustering process, the DBSCAN algorithm has been used, this model has to be tuned depending on the characteristics (geometry size, expected point density) of the geometry to be processed. An optimization of the hyperparameters of the DBSCAN algorithm is necessary in order to approximate the number of fruits in both real and synthetic LiDAR point clouds. The main difference is in the epsilon hyperparameter related to the distance to be evaluated. This indicates that there are still differences between the distribution of the generated synthetic data and the real data.

As in the work done by (165), we observed that the reflectance of apple tree fruit is different from the reflectance of other organs of the tree and this information can be used to classify efficiently the point clouds. Compared to the work of (165), our pipeline uses both machine learning and deep learning models that allow us to further extend the robustness of the pipeline in the segmentation of apples. Furthermore, it is important to highlight that our pipeline processes point clouds from a non-industrialized orchard with a large variety of genotypes.

Finally, **a GAN model** was proposed for the generation of realistic LiDAR point clouds of apple trees from synthetic data. To the best of my knowledge at the time of proposing this model, GANs for generating point clouds existed, but none of these models were specialized for fruit trees and none for the creation of annotated point clouds.

For the development of this model, Point-NET++ and P2P-NET were chosen as base models for the discriminator and generator respectively. Several tests were made to approximate the desired result, but along these tests, different difficulties were encountered related to how the information should be represented. It was observed that in the case of P2P-NET, the model reacts in different ways depending on the scale of each of the characteristics of the input point clouds. It was also observed that with the proposed model it is possible to approximate the geometry of point clouds with low density, but the model starts to have difficulties with increasing density in the representation of the geometry.

Finally, a model that approximates the topology of the tree was created. The model does not preserve the details that define the leaves or fruits of the tree. In addition, the model does not makes it possible to generate annotations due to the deformations that occur in the

different organs during the generation.

In order to improve such results, I proposed to train the model with different segments of the tree to approximate the generation of simpler geometries. To go beyond these results, attention mechanisms (318; 319) can offer new interesting possibilities, as they could help to increase the performance of the model, since they can be included in different layers to improve the feature selection and, in general, the learning weights. From this perspective, it will be possible to create a new layer architecture optimised for 3D point cloud processing. Also in line with this idea, the implementation of transformers (320) could bring to the model a better feature abstraction and then more details in the generated geometries (e.g.: (321; 322)). It is seen that the integration of the spectral domain (312; 323) and the transformers (320) is a process that could take GANs and other types of 3D point cloud processing networks to a new level of performance, as this representation can allow learning about deeper characteristics of the dataset being processed or used as a reference for generating new data. Another interesting approach may be the implementation of diffusion (324) techniques for generating and annotating point clouds (e.g. (325)), as these techniques help to preserve structure that can be used for these tasks.

Bibliography

[1] C. Godin, "Representing and encoding plant architecture: a review," *Annals of forest science*, vol. 57, no. 5, pp. 413–438, 2000.

[2] D. Barthélémy and Y. Caraglio, "Plant architecture: a dynamic, multilevel and comprehensive approach to plant form, structure and ontogeny," *Annals of botany*, vol. 99, no. 3, pp. 375–407, 2007.

[3] P. De Reffye, F. Blaise, and F. Houllier, "Modelling plant growth and architecture: some recent advances and applications to agronomy and forestry," *II Modelling Plant Growth, Environmental Control and Farm Management in Protected Cultivation 456*, pp. 105–116, 1997.

[4] T. Fourcaud, X. Zhang, A. Stokes, H. Lambers, and C. Körner, "Plant growth modelling and applications: the increasing importance of plant architecture in growth models," *Annals of Botany*, vol. 101, no. 8, pp. 1053–1063, 2008.

[5] J. Norman and J. Welles, "Radiative transfer in an array of canopies 1," *Agronomy Journal*, vol. 75, no. 3, pp. 481–488, 1983.

[6] C. Baker, "The development of a theoretical model for the windthrow of plants," *Journal of Theoretical Biology*, vol. 175, no. 3, pp. 355–372, 1995.

[7] W. Newman, D. Turcotte, and A. Gabrielov, "Fractal trees with side branching," *Fractals*, vol. 5, no. 04, pp. 603–614, 1997.

[8] X. Viennot, G. Eyrolles, N. Janey, and D. Arques, "Combinatorial analysis of ramified patterns and computer imagery of trees," in *Proceedings of the 16th annual conference on Computer graphics and interactive techniques*, 1989, pp. 31–40.

[9] R. M. Bastías and L. Corelli-Grappadelli, "Light quality management in fruit orchards: physiological and technological aspects," *Chilean Journal of Agricultural Research*, vol. 72, no. 4, p. 574, 2012.

[10] Y. Sharma, H. Singh, and S. Singh, "Effect of light interception and penetration at different levels of fruit tree canopy on quality of peach," *Current Science*, vol. 115, no. 8, pp. 1562–1566, 2018. [Online]. Available: https://www.jstor.org/stable/26978450

[11] J. Zhang, S. Serra, R. S. Leisso, and S. Musacchi, "Effect of light microclimate on the quality of 'd'anjou'pears in mature open-centre tree architecture," *biosystems engineering*, vol. 141, pp. 1–11, 2016.

[12] Y. Sun, C. Wang, H. Y. Chen, and H. Ruan, "Response of plants to water stress: a meta-analysis," *Frontiers in plant science*, vol. 11, p. 978, 2020.

[13] Y. Osakabe, K. Osakabe, K. Shinozaki, and L.-S. P. Tran, "Response of plants to water stress," *Frontiers in plant science*, vol. 5, p. 86, 2014.

[14] W. Yang, B. Pallas, J.-B. Durand, S. Martinez, M. Han, and E. Costes, "The impact of long-term water stress on tree architecture and production is related to changes in transitions between vegetative and reproductive growth in the 'granny smith'apple cultivar," *Tree Physiology*, vol. 36, no. 11, pp. 1369–1381, 2016.

[15] F. Brüchert and B. Gardiner, "The effect of wind exposure on the tree aerial architecture and biomechanics of sitka spruce (picea sitchensis, pinaceae)," *American journal of botany*, vol. 93, no. 10, pp. 1512–1521, 2006.

[16] A. Stokes, A. Fitter, and M. Courts, "Responses of young trees to wind and shading: effects on root architecture," *Journal of Experimental Botany*, vol. 46, no. 9, pp. 1139–1146, 1995.

[17] V. Hooijdonk, D. Woolley, I. Warrington, and D. Tustin, "Initial alteration of scion architecture by dwarfing apple rootstocks may involve shoot-root-shoot signalling by auxin, gibberellin, and cytokinin," *The Journal of Horticultural Science and Biotechnology*, vol. 85, no. 1, pp. 59–65, 2010.

[18] N. C. Cook, D. U. Bellstedt, and G. Jacobs, "Endogenous cytokinin distribution patterns at budburst in granny smith and braeburn apple shoots in relation to bud growth," *Scientia Horticulturae*, vol. 87, no. 1-2, pp. 53–63, 2001.

[19] K. Okada and C. Honda, "Molecular Mechanisms Regulating the Columnar Tree Architecture in Apple," *Forests*, vol. 13, no. 7, pp. 1–11, 2022.

[20] G. S. Khush, "Green revolution: the way forward," *Nature reviews genetics*, vol. 2, no. 10, pp. 815–822, 2001.

[21] R. Petersen and C. Krost, "Tracing a key player in the regulation of plant architecture: the columnar growth habit of apple trees (malus× domestica)," *Planta*, vol. 238, pp. 1–22, 2013.

[22] R. Benlloch, A. Berbel, A. Serrano-Mislata, and F. Madueño, "Floral initiation and inflorescence architecture: a comparative view," *Annals of botany*, vol. 100, no. 3, pp. 659–676, 2007.

[23] N. Kotoda, H. Iwanami, S. Takahashi, and K. Abe, "Antisense expression of mdtfl1, a tfl1-like gene, reduces the juvenile phase in apple," *Journal of the American Society for Horticultural Science*, vol. 131, no. 1, pp. 74–81, 2006.

[24] "Fao(2021) faostat database," https://www.fao.org/faostat/en/#data/QCL, accessed: 2023-08-29.

[25] F. A. Brief, "Agricultural production statistics 2000–2021," *Agricultural production statistics 2000–2021*, 2022.

[26] U. S. D. of Agriculture, "Fresh apples, grapes, and pears: World markets and trade," *Foreign Agricultural Service*, pp. 1–10, 2019.

[27] T. Tworkoski and S. Miller, "Rootstock effect on growth of apple scions with different growth habits," *Scientia horticulturae*, vol. 111, no. 4, pp. 335–343, 2007.

[28] Richard P. Martini, "Training and pruning apple trees," pp. 1–32, 2006. [Online]. Available: RM-7861applehighdensitymanagement.pdf

[29] M. J. DUFORDJANUARY, "Growing apples in the home garden," 2023. [Online]. Available: https://www.homefortheharvest.com/how-tall-do-apple-trees-grow/

[30] J. Tromp and B. Boertjes, "The effect of air temperature in successive periods of the growing season on sylleptic shoot formation in young apple trees," *Plant growth regulation*, vol. 19, pp. 177–182, 1996.

[31] A. Gur, B. Bravdo, and J. Hepner, "The influence of root temperature on apple trees. iii. the effect on photosynthesis and water balance," *Journal of Horticultural Science*, vol. 51, no. 2, pp. 203–210, 1976.

[32] Y. Lespinasse *et al.*, "Breeding apple tree: aims and methods." in *Proceedings of the Joint Conference of the EAPR Breeding & Varietal Assessment Section and the EUCARPIA Potato Section, Landerneau, France, 12-17 January 1992.* INRA, 1992, pp. 103–110.

[33] D. P. Horvath, J. V. Anderson, W. S. Chao, and M. E. Foley, "Knowing when to grow: signals regulating bud dormancy," *Trends in plant science*, vol. 8, no. 11, pp. 534–540, 2003.

[34] S. Monselise and E. Goldschmidt, "Alternate bearing in fruit trees," *Horticultural reviews*, vol. 4, no. 1, pp. 128–173, 1982.

[35] H. Jonkers, "Biennial bearing in apple and pear: a literature survey," *Scientia horticulturae*, vol. 11, no. 4, pp. 303–317, 1979.

[36] E. Costes, P. Lauri, and J.-L. Regnard, "Analyzing fruit tree architecture: implications for tree management and fruit production," *Horticultural reviews*, vol. 32, pp. 1–61, 2006.

[37] E. Costes, H. Sinoquet, J.-J. Kelner, and C. Godin, "Exploring within-tree architectural development of two apple tree cultivars over 6 years," *Annals of Botany*, vol. 91, no. 1, pp. 91–104, 2003.

[38] E. Costes and Y. Guédon, "Deciphering the ontogeny of a sympodial tree," *Trees*, vol. 26, pp. 865–879, 2012.

[39] P.-E. Lauri, E. Térouanne, J.-M. Lespinasse, J.-L. Regnard, and J.-J. Kelner, "Genotypic differences in the axillary bud growth and fruiting pattern of apple fruiting branches over several years—an approach to regulation of fruit bearing," *Scientia Horticulturae*, vol. 64, no. 4, pp. 265–281, 1995.

[40] V. Segura, C. Cilas, and E. Costes, "Dissecting apple tree architecture into genetic, ontogenetic and environmental effects: mixed linear modelling of repeated spatial and temporal measures," *New Phytologist*, vol. 178, no. 2, pp. 302–314, 2008.

[41] V. Segura, C.-E. Durel, and E. Costes, "Dissecting apple tree architecture into genetic, ontogenetic and environmental effects: Qtl mapping," *Tree genetics & genomes*, vol. 5, no. 1, pp. 165–179, 2009.

[42] J. Lespinasse and J. Delort, "Apple tree management in vertical axis: appraisal after ten years of experiments," in *III International Symposium on Research and Development on Orchard and Plantation Systems 160*, 1984, pp. 139–156.

[43] V. Segura, C. Cilas, F. Laurens, and E. Costes, "Phenotyping progenies for complex architectural traits: a strategy for 1-year-old apple trees (malus x domestica borkh.)," *Tree Genetics & Genomes*, vol. 2, pp. 140–151, 2006.

[44] C. P. Peace, L. Bianco, M. Troggio, E. Van de Weg, N. P. Howard, A. Cornille, C.-E. Durel, S. Myles, Z. Migicovsky, R. J. Schaffer *et al.*, "Apple whole genome sequences: recent advances and new prospects," *Horticulture Research*, vol. 6, 2019.

[45] B. Guitton, J.-J. Kelner, R. Velasco, S. E. Gardiner, D. Chagne, and E. Costes, "Genetic control of biennial bearing in apple," *Journal of experimental botany*, vol. 63, no. 1, pp. 131–149, 2012.

[46] B. Guitton, J.-J. Kelner, J.-M. Celton, X. Sabau, J.-P. Renou, D. Chagné, and E. Costes, "Analysis of transcripts differentially expressed between fruited and deflowered 'gala'adult trees: a contribution to biennial bearing understanding in apple," *BMC Plant Biology*, vol. 16, no. 1, pp. 1–22, 2016.

[47] M.-V. Hanke, H. Flachowsky, A. Peil, and C. Hättasch, "No flower no fruit-genetic potentials to trigger flowering in fruit trees," *Genes Genomes Genomics*, vol. 1, no. 1, pp. 1–20, 2007.

[48] J. L. Hill Jr and C. A. Hollender, "Branching out: new insights into the genetic regulation of shoot architecture in trees," *Current Opinion in Plant Biology*, vol. 47, pp. 73–80, 2019.

[49] L. C. Donadio, I. E. Lederman, S. R. Roberto, and E. S. Stucchi, "Dwarfing-canopy and rootstock cultivars for fruit trees," *Revista Brasileira de Fruticultura*, vol. 41, pp. e–997, 2019.

[50] N. Looney and W. Lane, "Spur-type growth mutants of mcintosh apple: a review of their genetics, physiology and field performance," in *International Workshop on Controlling Vigor in Fruit Trees 146*, 1983, pp. 31–46.

[51] P.-E. Lauri and E. Costes, "Progress in whole-tree architectural studies for apple cultivar characterization at inra, france-contribution to the ideotype approach," in *XI Eucarpia Symposium on Fruit Breeding and Genetics 663*, 2003, pp. 357–362.

[52] X. Zheng, Y. Zhao, D. Shan, K. Shi, L. Wang, Q. Li, N. Wang, J. Zhou, J. Yao, Y. Xue *et al.*, "Md wrky 9 overexpression confers intensive dwarfing in the m26 rootstock of apple by directly inhibiting brassinosteroid synthetase md dwf 4 expression," *New Phytologist*, vol. 217, no. 3, pp. 1086–1098, 2018.

[53] C. Hättasch, H. Flachowsky, D. Kapturska, and M.-V. Hanke, "Isolation of flowering genes and seasonal changes in their transcript levels related to flower induction and initiation in apple (malus domestica)," *Tree physiology*, vol. 28, no. 10, pp. 1459–1466, 2008.

[54] J.-B. Durand, A. Allard, B. Guitton, E. Van de Weg, M. C. Bink, and E. Costes, "Predicting flowering behavior and exploring its genetic determinism in an apple multi-family population based on statistical indices and simplified phenotyping," *Frontiers in Plant Science*, vol. 8, p. 858, 2017.

[55] C. Godin and H. Sinoquet, "Functional-structural plant modelling," *New Phytologist*, 2005.

[56] J. Vos, L. Marcelis, and J. Evers, "Functional-structural plant modelling in crop production: adding a dimension," *Frontis*, pp. 1–12, 2007.

[57] E. Costes, C. Smith, M. Renton, Y. Guédon, P. Prusinkiewicz, and C. Godin, "Mappet: simulation of apple tree development using mixed stochastic and biomechanical models," *Functional Plant Biology*, vol. 35, no. 10, pp. 936–950, 2008.

[58] F. Boudon, S. Persello, A. Jestin, A.-S. Briand, P. Fernique, M. Léchaudel, I. Grechi, F. Normand *et al.*, "An fspm approach for modeling fruit yield and quality in mango trees," in *IEEE International Conference on Functional-Structural Plant Growth Modeling, Simulation, Visualization and Applications (FSPMA 2016)*, 2016.

[59] F. Boudon, S. Persello, A. Jestin, A.-S. Briand, I. Grechi, P. Fernique, Y. Guédon, M. Léchaudel, P.-É. Lauri, and F. Normand, "V-mango: a functional–structural model of mango tree growth, development and fruit production," *Annals of Botany*, vol. 126, no. 4, pp. 745–763, 2020.

[60] D. Da Silva, R. Favreau, S. Tombesi, and T. DeJong, "Modeling size-controlling rootstock effects on peach tree growth and development using l-peach-h," in *IX International Symposium on Modelling in Fruit Research and Orchard Management 1068*, 2011, pp. 227–233.

[61] M. Wang, N. White, V. Grimm, H. Hofman, D. Doley, G. Thorp, B. Cribb, E. Wherritt, L. Han, and J. Wilkie, "Patternoriented modelling as a novel way to verify and validate functional-structural plant models: a demonstration with the annual growth module of avocado," *Annals of botany*, vol. 121, pp. 941–959, 2018.

[62] P. Prusinkiewicz, A. Lindenmayer, P. Prusinkiewicz, and A. Lindenmayer, "Graphical modeling using l-systems," *The Algorithmic Beauty of Plants*, pp. 1–50, 1990.

[63] A. Walter, F. Liebisch, and A. Hund, "Plant phenotyping: from bean weighing to image analysis," *Plant methods*, vol. 11, no. 1, pp. 1–11, 2015.

[64] R. Pieruschka and U. Schurr, "Plant phenotyping: past, present, and future," *Plant Phenomics*, vol. 2019, 2019.

[65] Z. Li, R. Guo, M. Li, Y. Chen, and G. Li, "A review of computer vision technologies for plant phenotyping," *Computers and Electronics in Agriculture*, vol. 176, p. 105672, 2020.

[66] L. Li, Q. Zhang, and D. Huang, "A review of imaging techniques for plant phenotyping," *Sensors*, vol. 14, no. 11, pp. 20 078–20 111, 2014.

[67] M. Larjavaara and H. C. Muller-Landau, "Measuring tree height: a quantitative comparison of two common field methods in a moist tropical forest," *Methods in Ecology and Evolution*, vol. 4, no. 9, pp. 793–801, 2013.

[68] G. J. L. Panzou and T. R. Feldpausch, "Measuring crown dimensions for tropical forest trees a field manual," *NERC: Exeter, UK*, 2020.

[69] R. Leverett and D. Bertolette, "American forests champion trees measuring guidelines handbook," *American Forests*, 2014.

[70] M. S. C. Mat, J. M. Diah, M. A. M. Din, and A. M. Samad, "Data acquisition and representation of leaves using digital close range photogrammetry for species identification," in *2014 IEEE 5th Control and System Graduate Research Colloquium*. IEEE, 2014, pp. 108–113.

[71] H. Jenkins, "An airflow planimeter for measuring the area of detached leaves," *Plant physiology*, vol. 34, no. 5, p. 532, 1959.

[72] H. Sinoquet and P. Rivet, "Measurement and visualization of the architecture of an adult tree based on a three-dimensional digitising device," *Trees*, vol. 11, no. 5, pp. 265–270, 1997.

[73] J. B. Evers, J. Vos, C. Fournier, B. Andrieu, M. Chelle, and P. C. Struik, "Towards a generic architectural model of tillering in gramineae, as exemplified by spring wheat (triticum aestivum)," *New Phytologist*, vol. 166, no. 3, pp. 801–812, 2005.

[74] V. Realities, "Fastrack," 2023. [Online]. Available: https://www.vrealities.com/

[75] H. Van As, T. Scheenen, and F. J. Vergeldt, "Mri of intact plants," *Photosynthesis Research*, vol. 102, pp. 213–222, 2009.

[76] N. institute of biomedical imaging and bioengineering, "Magnetic resonance imaging (mri)," 2023. [Online]. Available: https://shorturl.at/iuFHW

[77] M. Eithun, J. Larson, G. Lang, D. H. Chitwood, and E. Munch, "Isolating phyllotactic patterns embedded in the secondary growth of sweet cherry (prunus avium l.) using magnetic resonance imaging," *Plant methods*, vol. 15, pp. 1–12, 2019.

[78] M. A. Zwieniecki, P. J. Melcher, and E. T. Ahrens, "Analysis of spatial and temporal dynamics of xylem refilling in acer rubrum l. using magnetic resonance imaging," *Frontiers in Plant Science*, vol. 4, p. 265, 2013.

[79] D. Pflugfelder, R. Metzner, D. van Dusschoten, R. Reichel, S. Jahnke, and R. Koller, "Non-invasive imaging of plant roots in different soils using magnetic resonance imaging (mri)," *Plant Methods*, vol. 13, no. 1, pp. 1–9, 2017.

[80] R. Metzner, D. van Dusschoten, J. Bühler, U. Schurr, and S. Jahnke, "Belowground plant development measured with magnetic resonance imaging (mri): exploiting the potential for non-invasive trait quantification using sugar beet as a proxy," *Frontiers in plant science*, vol. 5, p. 469, 2014.

[81] H. Schulz, J. A. Postma, D. van Dusschoten, H. Scharr, S. Behnke, G. Csurka, and J. Braz, "3d reconstruction of plant roots from mri images." in *VISAPP (2)*, 2012, pp. 24–33.

[82] N. Fahlgren, M. A. Gehan, and I. Baxter, "Lights, camera, action: high-throughput plant phenotyping is ready for a close-up," *Current opinion in plant biology*, vol. 24, pp. 93–99, 2015.

[83] D. Reynolds, J. Ball, A. Bauer, S. Griffiths, and J. Zhou, "Cropsurveyor: a scalable open-source experiment management system for distributed plant phenotyping and iot-based crop management," *bioRxiv*, p. 451120, 2018.

[84] J. M. Nassar, S. M. Khan, D. R. Villalva, M. M. Nour, A. S. Almuslem, and M. M. Hussain, "Compliant plant wearables for localized microclimate and plant growth monitoring," *npj Flexible Electronics*, vol. 2, no. 1, p. 24, 2018.

[85] W. Tang, T. Yan, J. Ping, J. Wu, and Y. Ying, "Rapid fabrication of flexible and stretchable strain sensor by chitosan-based water ink for plants growth monitoring," *Advanced Materials Technologies*, vol. 2, no. 7, p. 1700021, 2017.

[86] P. Ache, H. Bauer, H. Kollist, K. A. Al-Rasheid, S. Lautner, W. Hartung, and R. Hedrich, "Stomatal action directly feeds back on leaf turgor: new insights into the regulation of the plant water status from non-invasive pressure probe measurements," *The Plant Journal*, vol. 62, no. 6, pp. 1072–1082, 2010.

[87] T. De Rocher and R. Tausch, "Predicting potential transpiration of singleleaf pinyon: an adaptation of the potometer method," *Forest ecology and management*, vol. 63, no. 2-3, pp. 169–180, 1994.

[88] D. Smith and K. Buchholtz, "Modification of plant transpiration rate with chemicals," *Plant Physiology*, vol. 39, no. 4, p. 572, 1964.

[89] L. Yu, W. Wang, X. Zhang, and W. Zheng, "A review on leaf temperature sensor: Measurement methods and application," in *Computer and Computing Technologies in Agriculture IX: 9th IFIP WG 5.14 International Conference, CCTA 2015, Beijing, China, September 27-30, 2015, Revised Selected Papers, Part I 9*. Springer, 2016, pp. 216–230.

[90] H. Jones and P. Schofield, "Thermal and other remote sensing of plant stress," *General and Applied Plant Physiology*, vol. 34, no. 1-2, pp. 19–32, 2008.

[91] D. R. Hershey, "Plant light measurement & calculations," *The American Biology Teacher*, vol. 53, no. 6, pp. 351–353, 1991.

[92] B.-S. Wu, A.-S. Rufyikiri, V. Orsat, and M. G. Lefsrud, "Re-interpreting the photosynthetically action radiation (par) curve in plants," *Plant Science*, vol. 289, p. 110272, 2019.

[93] Y. Song, C. Glasbey, G. Horgan, G. Polder, J. Dieleman, and G. Van der Heijden, "Automatic fruit recognition and counting from multiple images," *Biosystems Engineering*, vol. 118, pp. 203–215, 2014.

[94] U.-O. Dorj, M. Lee, and S.-s. Yun, "An yield estimation in citrus orchards via fruit detection and counting using image processing," *Computers and electronics in agriculture*, vol. 140, pp. 103–112, 2017.

[95] P. K. Sethy, S. Panda, S. K. Behera, and A. K. Rath, "On tree detection, counting & post-harvest grading of fruits based on image processing and machine learning approach-a review," *International Journal of Engineering & Technology*, vol. 9, no. 2, pp. 649–663, 2017.

[96] L. Gao and X. Lin, "A method for accurately segmenting images of medicinal plant leaves with complex backgrounds," *Computers and Electronics in Agriculture*, vol. 155, pp. 426–445, 2018.

[97] D. Al Bashish, M. Braik, and S. Bani-Ahmad, "A framework for detection and classification of plant leaf and stem diseases," in *2010 international conference on signal and image processing*. IEEE, 2010, pp. 113–118.

[98] W. Ji, Z. Qian, B. Xu, Y. Tao, D. Zhao, and S. Ding, "Apple tree branch segmentation from images with small gray-level difference for agricultural harvesting robot," *Optik*, vol. 127, no. 23, pp. 11 173–11 182, 2016.

[99] A. You, C. Grimm, and J. R. Davidson, "Optical flow-based branch segmentation for complex orchard environments," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 9180–9186.

[100] R. Raj, J. P. Walker, R. Pingale, R. Nandan, B. Naik, and A. Jagarlapudi, "Leaf area index estimation using top-of-canopy airborne rgb images," *International Journal of Applied Earth Observation and Geoinformation*, vol. 96, p. 102282, 2021.

[101] J. Xue and B. Su, "Significant remote sensing vegetation indices: A review of developments and applications," *Journal of sensors*, vol. 2017, 2017.

[102] Y. Huang, Z. Ren, D. Li, and X. Liu, "Phenotypic techniques and applications in fruit trees: a review," *Plant Methods*, vol. 16, no. 1, pp. 1–22, 2020.

[103] H. Huang, J. Deng, Y. Lan, A. Yang, X. Deng, S. Wen, H. Zhang, and Y. Zhang, "Accurate weed mapping and prescription map generation based on fully convolutional networks using uav imagery," *Sensors*, vol. 18, no. 10, p. 3299, 2018.

[104] J. Zhao, X. Zhang, C. Gao, X. Qiu, Y. Tian, Y. Zhu, and W. Cao, "Rapid mosaicking of unmanned aerial vehicle (uav) images for crop growth monitoring using the sift algorithm," *Remote Sensing*, vol. 11, no. 10, p. 1226, 2019.

[105] R. Sinha, J. J. Quirós, S. Sankaran, and L. R. Khot, "High resolution aerial photogrammetry based 3d mapping of fruit crop canopies for precision inputs management," *Information Processing in Agriculture*, vol. 9, no. 1, pp. 11–23, 2022.

[106] M. S. Vision, "What is hyperspectral imaging?" 2022. [Online]. Available: https://www.middletonspectral.com/resources/what-is-hyperspectral-imaging/

[107] M. Louhaichi, M. M. Borman, and D. E. Johnson, "Spatially located platform and aerial photography for documentation of grazing impacts on wheat," *Geocarto International*, vol. 16, no. 1, pp. 65–70, 2001.

[108] A. A. Gitelson, Y. J. Kaufman, R. Stark, and D. Rundquist, "Novel algorithms for remote estimation of vegetation fraction," *Remote sensing of Environment*, vol. 80, no. 1, pp. 76–87, 2002.

[109] J. W. Rouse Jr, R. H. Haas, J. Schell, and D. Deering, "Monitoring the vernal advancement and retrogradation (green wave effect) of natural vegetation," NASA, Tech. Rep., 1973.

[110] U. Lussem, A. Bolten, M. Gnyp, J. Jasper, and G. Bareth, "Evaluation of rgb-based vegetation indices from uav imagery to estimate forage yield in grassland," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 42, no. 3, p. 1215, 2018.

[111] L. G. documentation center, "Broadband greenness," 2023. [Online]. Available: https://www.l3harrisgeospatial.com/docs/broadbandgreenness.html#Green6

[112] L. S. Eng, R. Ismail, W. Hashim, and A. Baharum, "The use of vari, gli, and vigreen formulas in detecting vegetation in aerial images," *International Journal of Technology*, vol. 10, no. 7, pp. 1385–1394, 2019.

[113] E. R. Hunt, M. Cavigelli, C. S. Daughtry, J. E. Mcmurtrey, and C. L. Walthall, "Evaluation of digital photography from model aircraft for remote sensing of crop biomass and nitrogen status," *Precision Agriculture*, vol. 6, pp. 359–378, 2005.

[114] J. Rouse Jr, R. H. Haas, D. Deering, J. Schell, and J. C. Harlan, "Monitoring the vernal advancement and retrogradation (green wave effect) of natural vegetation," NASA, Tech. Rep., 1974.

[115] A. R. Huete, "A soil-adjusted vegetation index (savi)," *Remote sensing of environment*, vol. 25, no. 3, pp. 295–309, 1988.

[116] G. S. Birth and G. R. McVey, "Measuring the color of growing turf with a reflectance spectrophotometer 1," *Agronomy Journal*, vol. 60, no. 6, pp. 640–643, 1968.

[117] D. Deering and R. Haas, "Using landsat digital data for estimating green biomass," in *Ann. Meeting of the Soc. for Range Management Symp. on Rangeland Remote Sensing*, ser. TM-80727, no. E80-10328, 1980.

[118] N. G. Silleos, T. K. Alexandridis, I. Z. Gitas, and K. Perakis, "Vegetation indices: advances made in biomass estimation and vegetation monitoring in the last 30 years," *Geocarto International*, vol. 21, no. 4, pp. 21–28, 2006.

[119] T. U. S. G. Survey, "Weekly ndvi," 2023. [Online]. Available: https://www.usgs.gov/fire-danger-forecast/weekly-ndvi

[120] Z. Zhen, S. Chen, T. Yin, E. Chavanon, N. Lauret, J. Guilleux, M. Henke, W. Qin, L. Cao, J. Li *et al.*, "Using the negative soil adjustment factor of soil adjusted vegetation index (savi) to resist saturation effects and estimate leaf area index (lai) in dense vegetation areas," *Sensors*, vol. 21, no. 6, p. 2115, 2021.

[121] L. M. The United States Geological Survey, "Landsat soil adjusted vegetation index," 2023. [Online]. Available: https://www.usgs.gov/landsat-missions/landsat-soil-adjusted-vegetation-index

[122] C. S. P. U. Humboldt, "Vegetation indices," 2023. [Online]. Available: shorturl.at/iy128

[123] Lucasbosch, "Comparison of spectral sampling in rgb, multispectral and hyperspectral imaging," 2021. [Online]. Available: https://commons.wikimedia.org/wiki/File:Spectral_sampling_RGB_multispectral_hyperspectral_imaging.svg

[124] C. A. Devia, J. P. Rojas, E. Petro, C. Martinez, I. F. Mondragon, D. Patiño, M. C. Rebolledo, and J. Colorado, "High-throughput biomass estimation in rice crops using uav multispectral imagery," *Journal of Intelligent & Robotic Systems*, vol. 96, no. 3, pp. 573–589, 2019.

[125] V. Lebourgeois, A. Bégué, S. Labbé, M. Houles, and J.-F. Martiné, "A light-weight multi-spectral aerial imaging system for nitrogen crop monitoring," *Precision Agriculture*, vol. 13, no. 5, pp. 525–541, 2012.

[126] H. Qi, Z. Wu, L. Zhang, J. Li, J. Zhou, Z. Jun, and B. Zhu, "Monitoring of peanut leaves chlorophyll content based on drone-based multispectral image feature extraction," *Computers and Electronics in Agriculture*, vol. 187, p. 106292, 2021.

[127] A. Gazda and K. Kędra, "Tree architecture description using a single-image photogrammetric method," *Dendrobiology*, vol. 78, no. 78, pp. 124–135, 2017.

[128] K. Kędra, I. Barbeito, M. Dassot, P. Vallet, and A. Gazda, "Single-image photogrammetry for deriving tree architectural traits in mature forest stands: a comparison with terrestrial laser scanning," *Annals of forest science*, vol. 76, no. 1, pp. 1–13, 2019.

[129] B. G. Weinstein, S. Marconi, S. Bohlman, A. Zare, and E. White, "Individual tree-crown detection in rgb imagery using semi-supervised deep learning neural networks," *Remote Sensing*, vol. 11, no. 11, p. 1309, 2019.

[130] B. G. Weinstein, S. Marconi, M. Aubry-Kientz, G. Vincent, H. Senyondo, and E. P. White, "Deepforest: A python package for rgb deep learning tree crown delineation," *Methods in Ecology and Evolution*, vol. 11, no. 12, pp. 1743–1751, 2020.

[131] L. Fu, F. Gao, J. Wu, R. Li, M. Karkee, and Q. Zhang, "Application of consumer rgb-d cameras for fruit detection and localization in field: A critical review," *Computers and Electronics in Agriculture*, vol. 177, p. 105687, 2020.

[132] J. Xie, Y. Zhao, Y. Liu, P. Su, Y. Zhao, J. Cheng, Y. Zheng, and J. Liu, "Topology reconstruction of tree-like structure in images via structural similarity measure and dominant set clustering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8505–8513.

[133] P. Fan, G. Lang, B. Yan, X. Lei, P. Guo, Z. Liu, and F. Yang, "A method of segmenting apples based on gray-centered rgb color space," *Remote Sensing*, vol. 13, no. 6, p. 1211, 2021.

[134] M. Teixidó, D. Font, T. Pallejà, M. Tresanchez, M. Nogués, and J. Palacín, "Definition of linear color models in the rgb vector color space to detect red peaches in orchard images taken under natural illumination," *Sensors*, vol. 12, no. 6, pp. 7701–7718, 2012.

[135] H. N. Patel, R. K. Jain, and M. V. Joshi, "Automatic segmentation and yield measurement of fruit using shape analysis," *International Journal of Computer Applications*, vol. 45, no. 7, pp. 19–24, 2012.

[136] M. Rahnemoonfar and C. Sheppard, "Deep count: fruit counting based on deep simulated learning," *Sensors*, vol. 17, no. 4, p. 905, 2017.

[137] M. Fukuda, T. Okuno, and S. Yuki, "Central object segmentation by deep learning to continuously monitor fruit growth through rgb images," *Sensors*, vol. 21, no. 21, p. 6999, 2021.

[138] Y. Majeed, J. Zhang, X. Zhang, L. Fu, M. Karkee, Q. Zhang, and M. D. Whiting, "Deep learning based segmentation for automated training of apple trees on trellis wires," *Computers and Electronics in Agriculture*, vol. 170, p. 105277, 2020.

[139] A. Safonova, E. Guirado, Y. Maglinets, D. Alcaraz-Segura, and S. Tabik, "Olive tree biovolume from uav multi-resolution image segmentation with mask r-cnn," *Sensors*, vol. 21, no. 5, p. 1617, 2021.

[140] Z. Chen, D. Ting, R. Newbury, and C. Chen, "Semantic segmentation for partially occluded apple trees based on deep learning," *Computers and Electronics in Agriculture*, vol. 181, p. 105952, 2021.

[141] J. S. Aber, I. Marzolff, and J. Ries, *Small-format aerial photography: Principles, techniques and geoscience applications*. Elsevier, 2010.

[142] J. A. Siegel and P. J. Saukko, *Encyclopedia of forensic sciences*. Academic Press, 2012.

[143] R. Szeliski, *Computer vision: algorithms and applications*. Springer Nature, 2022.

[144] A. Mason, "Making 3d models with photogrammetry," 2022. [Online]. Available: https://thehaskinssociety.wildapricot.org/photogrammetry

[145] V. Lepetit, P. Fua *et al.*, "Monocular model-based 3d tracking of rigid objects: A survey," *Foundations and Trends® in Computer Graphics and Vision*, vol. 1, no. 1, pp. 1–89, 2005.

[146] OpenCV, "Camera calibration with opencv," 2023. [Online]. Available: https://docs.opencv.org/3.4/d4/d94/tutorial_camera_calibration.html

[147] Z. Wei, L. Cao, and G. Zhang, "A novel 1d target-based calibration method with unknown orientation for structured light vision sensor," *Optics & Laser Technology*, vol. 42, no. 4, pp. 570–574, 2010.

[148] ARCGIS, "Overview of georeferencing," 2022. [Online]. Available: https://pro.arcgis.com/en/pro-app/latest/help/data/imagery/overview-of-georeferencing.htm

[149] G. Sun, X. Wang, Y. Ding, W. Lu, and Y. Sun, "Remote measurement of apple orchard canopy information using unmanned aerial vehicle photogrammetry," *Agronomy*, vol. 9, no. 11, p. 774, 2019.

[150] S. Muckenhuber, H. Holzer, and Z. Bockaj, "Automotive lidar modeling approach based on material properties and lidar capabilities," *Sensors*, vol. 20, no. 11, p. 3309, 2020.

[151] B. Alsadik, "Multibeam lidar for mobile mapping systems," *GIM International*, pp. 1–3, 2020.

[152] T. Raj, F. Hanim Hashim, A. Baseri Huddin, M. F. Ibrahim, and A. Hussain, "A survey on lidar scanning mechanisms," *Electronics*, vol. 9, no. 5, p. 741, 2020.

[153] N. Muhammad and S. Lacroix, "Calibration of a rotating multi-beam Lidar," *IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings*, pp. 5648–5653, 2010.

[154] B. N. Bailey and M. H. Ochoa, "Semi-direct tree reconstruction using terrestrial lidar point cloud data," *Remote Sensing of Environment*, vol. 208, pp. 133–144, 2018.

[155] S. Pan, H. Guan, Y. Chen, Y. Yu, W. N. Gonçalves, J. M. Junior, and J. Li, "Land-cover classification of multispectral lidar data using cnn with optimized hyper-parameters," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 166, pp. 241–254, 2020.

[156] J.-K. Huang, C. Feng, M. Achar, M. Ghaffari, and J. W. Grizzle, "Global unifying intrinsic calibration for spinning and solid-state lidars," *arXiv preprint arXiv:2012.03321*, 2020.

[157] N. Sun, Q. Qiu, Z. Fan, T. Li, C. Ji, Q. Feng, and C. Zhao, "Intrinsic Calibration of Multi-Beam LiDARs for Agricultural Robots," *Remote Sensing*, vol. 14, no. 19, 2022.

[158] K. Omasa, F. Hosoi, and A. Konishi, "3d lidar imaging for detecting and understanding plant responses and canopy structure," *Journal of experimental botany*, vol. 58, no. 4, pp. 881–898, 2007.

[159] J.-F. Côté, R. A. Fournier, G. W. Frazer, and K. O. Niemann, "A fine-scale architectural model of trees to enhance lidar-derived measurements of forest canopy structure," *Agricultural and forest meteorology*, vol. 166, pp. 72–85, 2012.

[160] B. Brede, A. Lau, H. M. Bartholomeus, and L. Kooistra, "Comparing riegl ricopter uav lidar derived canopy height and dbh with terrestrial lidar," *Sensors*, vol. 17, no. 10, p. 2371, 2017.

[161] S. Sun, C. Li, A. H. Paterson, Y. Jiang, R. Xu, J. S. Robertson, J. L. Snider, and P. W. Chee, "In-field high throughput phenotyping and cotton plant growth analysis using lidar," *Frontiers in Plant Science*, vol. 9, p. 16, 2018.

[162] J. P. Underwood, C. Hung, B. Whelan, and S. Sukkarieh, "Mapping almond orchard canopy volume, flowers, fruit and yield using lidar and vision sensors," *Computers and Electronics in Agriculture*, vol. 130, pp. 83–96, 2016.

[163] A. Coupel-Ledru, B. Pallas, M. Delalande, V. Segura, B. Guitton, H. Muranty, C.-E. Durel, J.-l. Regnard, and E. Costes, "Tree architecture, light interception and water-use related traits are controlled by different genomic regions in an apple tree core collection," *New Phytologist*, vol. 234, no. 1, pp. 209–226, 2022.

[164] L. Malambo, S. Popescu, D. Horne, N. Pugh, and W. Rooney, "Automated detection and measurement of individual sorghum panicles using density-based clustering of terrestrial lidar data," *ISPRS journal of photogrammetry and remote sensing*, vol. 149, pp. 1–13, 2019.

[165] J. Gené-Mola, E. Gregorio, J. Guevara, F. Auat, R. Sanz-Cortiella, A. Escolà, J. Llorens, J.-R. Morros, J. Ruiz-Hidalgo, V. Vilaplana *et al.*, "Fruit detection in an apple orchard using a mobile terrestrial laser scanner," *Biosystems engineering*, vol. 187, pp. 171–184, 2019.

[166] J.-F. Côté, R. A. Fournier, and R. Egli, "An architectural model of trees to estimate forest structural attributes using terrestrial lidar," *Environmental Modelling & Software*, vol. 26, no. 6, pp. 761–777, 2011.

[167] B. Pallas, S. Martinez, O. Simler, E. CarriÃ, E. Costes, and F. Boudon, "Assessing t-lidar technology for high throughput phenotyping apple tree topological and architectural traits," in *XXX International Horticultural Congress IHC2018: International Symposium on Cultivars, Rootstocks and Management Systems of 1281*, 2018, pp. 625–632.

[168] Y. Su, F. Wu, Z. Ao, S. Jin, F. Qin, B. Liu, S. Pang, L. Liu, and Q. Guo, "Evaluating maize phenotype dynamics under drought stress using terrestrial lidar," *Plant methods*, vol. 15, pp. 1–16, 2019.

[169] N. Tsoulias, D. S. Paraforos, G. Xanthopoulos, and M. Zude-Sasse, "Apple shape detection based on geometric and radiometric features using a lidar laser scanner," *Remote Sensing*, vol. 12, no. 15, p. 2481, 2020.

[170] R. Pinkham, S. Zeng, and Z. Zhang, "Quicknn: Memory and performance optimization of kd tree based nearest neighbor search for 3d point clouds," in *2020 IEEE International symposium on high performance computer architecture (HPCA)*. IEEE, 2020, pp. 180–192.

[171] J. Elseberg, D. Borrmann, and A. Nüchter, "Efficient processing of large 3d point clouds," in *2011 XXIII International Symposium on Information, Communication and Automation Technologies*. IEEE, 2011, pp. 1–7.

[172] H. Roodaki and M. N. Bojnordi, "Compressed geometric arrays for point cloud processing," *IEEE Transactions on Multimedia*, 2022.

[173] M. Skrodzki, "The kd tree data structure and a proof for neighborhood computation in expected logarithmic time," *arXiv preprint arXiv:1903.04936*, 2019.

[174] Y.-L. Chen, B.-Y. Chen, S.-H. Lai, and T. Nishita, "Binary orientation trees for volume and surface reconstruction from unoriented point clouds," in *Computer Graphics Forum*, vol. 29, no. 7. Wiley Online Library, 2010, pp. 2011–2019.

[175] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3D: A modern library for 3D data processing," *arXiv:1801.09847*, 2018.

[176] E. Camuffo, D. Mari, and S. Milani, "Recent advancements in learning algorithms for point clouds: An updated overview," *Sensors*, vol. 22, no. 4, p. 1357, 2022.

[177] Y. Xu and U. Stilla, "Toward building and civil infrastructure reconstruction from point clouds: A review on data and key techniques," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 14, pp. 2857–2885, 2021.

[178] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous robots*, vol. 34, pp. 189–206, 2013.

[179] Y. Xu, X. Tong, and U. Stilla, "Voxel-based representation of 3d point clouds: Methods, applications, and its potential use in the construction industry," *Automation in Construction*, vol. 126, p. 103675, 2021.

[180] S.-H. Han, S.-J. Lee, S.-P. Kim, C.-J. Kim, J. Heo, and H.-B. Lee, "A comparison of 3d r-tree and octree to index large point clouds from a 3d terrestrial laser scanner," *Journal of the Korean Society of Surveying, Geodesy, Photogrammetry and Cartography*, vol. 29, no. 1, pp. 39–46, 2011.

[181] V. Laurmaa, M. Picasso, and G. Steiner, "An octree-based adaptive semi-lagrangian vof approach for simulating the displacement of free surfaces," *Computers & Fluids*, vol. 131, pp. 190–204, 2016.

[182] D. Madeira, A. Montenegro, E. Clua, and T. Lewiner, "Gpu octrees and optimized search," in *Proceedings of VIII Brazilian Symposium on Games and Digital Entertainment*, 2009, pp. 73–76.

[183] F. Chen, R. Ying, J. Xue, F. Wen, and P. Liu, "Parallelnn: A parallel octree-based nearest neighbor search accelerator for 3d point clouds," in *2023 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. IEEE, 2023, pp. 403–414.

[184] L. Fu, J. Liu, J. Zhou, M. Zhang, and Y. Lin, "Tree skeletonization for raw point cloud exploiting cylindrical shape prior," *Ieee Access*, vol. 8, pp. 27 327–27 341, 2020.

[185] H. Blum, "A Transformation for Extracting New Descriptors of Shape," in *Models for the Perception of Speech and Visual Form*, W. Wathen-Dunn, Ed. Cambridge: MIT Press, 1967, pp. 362–380.

[186] P. K. Saha, G. Borgefors, and G. S. di Baja, "A survey on skeletonization algorithms and their applications," *Pattern recognition letters*, vol. 76, pp. 3–12, 2016.

[187] A. Tagliasacchi, T. Delame, M. Spagnuolo, N. Amenta, and A. Telea, "3d skeletons: A state-of-the-art report," in *Computer Graphics Forum*, vol. 35, no. 2. Wiley Online Library, 2016, pp. 573–597.

[188] W. Abu-Ain, S. N. H. S. Abdullah, B. Bataineh, T. Abu-Ain, and K. Omar, "Skeletonization algorithm for binary images," *Procedia Technology*, vol. 11, pp. 704–709, 2013.

[189] R. L. Ogniewicz and M. Ilg, "Voronoi skeletons: theory and applications." in *CVPR*, vol. 92, 1992, pp. 63–69.

[190] A. Beristain, M. Graña, and A. I. Gonzalez, "A pruning algorithm for stable voronoi skeletons," *Journal of Mathematical Imaging and Vision*, vol. 42, pp. 225–237, 2012.

[191] P. C. Kwok and V. Ranjan, "A survey of 3-d thinning algorithms," in *Vision Interface'91*, vol. 3, 1991.

[192] M. Berger, A. Tagliasacchi, L. M. Seversky, P. Alliez, G. Guennebaud, J. A. Levine, A. Sharf, and C. T. Silva, "A survey of surface reconstruction from point clouds," in *Computer graphics forum*, vol. 36, no. 1. Wiley Online Library, 2017, pp. 301–329.

[193] N. D. Cornea, D. Silver, and P. Min, "Curve-skeleton applications," in *VIS 05. IEEE Visualization, 2005.* IEEE, 2005, pp. 95–102.

[194] N. D. Cornea, *Curve-skeletons: properties, computation and applications*. Rutgers University New Jersey, USA, 2007.

[195] G. Borgefors, I. Nyström, and G. S. Di Baja, "Computing skeletons in three dimensions," *Pattern recognition*, vol. 32, no. 7, pp. 1225–1236, 1999.

[196] A. Telea and A. Vilanova, "A robust level-set algorithm for centerline extraction," in *EPRINTS-BOOK-TITLE*. University of Groningen, Johann Bernoulli Institute for Mathematics and ..., 2003.

[197] A. Bucksch and R. Lindenbergh, "Campino—a skeletonization method for point cloud processing," *ISPRS journal of photogrammetry and remote sensing*, vol. 63, no. 1, pp. 115–127, 2008.

[198] C. M. Ma and M. Sonka, "A fully parallel 3d thinning algorithm and its applications," *Computer vision and image understanding*, vol. 64, no. 3, pp. 420–433, 1996.

[199] T. Wang and A. Basu, "A note on 'a fully parallel 3d thinning algorithm and its applications'," *Pattern Recognition Letters*, vol. 28, no. 4, pp. 501–506, 2007.

[200] C. Lohou and J. Dehos, "Automatic correction of ma and sonka's thinning algorithm using p-simple points," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 6, pp. 1148–1152, 2010.

[201] G. Bertrand, "On p-simple points," *Comptes rendus de l'Académie des sciences. Série I, Mathématique*, vol. 1, no. 321, pp. 1077–1084, 1995.

[202] H. Huang, S. Wu, D. Cohen-Or, M. Gong, H. Zhang, G. Li, and B. Chen, "L1-medial skeleton of point cloud." *ACM Trans. Graph.*, vol. 32, no. 4, pp. 65–1, 2013.

[203] H. Wang, E. Cimen, N. Singh, and E. Buckler, "Deep learning for plant genomics and crop improvement," *Current opinion in plant biology*, vol. 54, pp. 34–41, 2020.

[204] S. Orozco-Arias, G. Isaza, and R. Guyot, "Retrotransposons in plant genomes: structure, identification, and classification through bioinformatics and machine learning," *International journal of molecular sciences*, vol. 20, no. 15, p. 3837, 2019.

[205] V. Demidchik, A. Y. Shashko, U. Bandarenka, G. Smolikova, D. Przhevalskaya, M. Charnysh, G. Pozhvanov, A. Barkosvkyi, I. Smolich, A. Sokolik *et al.*, "Plant phenomics: fundamental bases, software and hardware platforms, and machine learning," *Russian journal of plant physiology*, vol. 67, pp. 397–412, 2020.

[206] S. Kolhar and J. Jagtap, "Plant trait estimation and classification studies in plant phenotyping using machine vision–a review," *Information Processing in Agriculture*, vol. 10, no. 1, pp. 114–135, 2023.

[207] A. D. J. van Dijk, G. Kootstra, W. Kruijer, and D. de Ridder, "Machine learning in plant science and plant breeding," *Iscience*, vol. 24, no. 1, 2021.

[208] E. H. Mahood, L. H. Kruse, and G. D. Moghe, "Machine learning: a powerful tool for gene function prediction in plants," *Applications in Plant Sciences*, vol. 8, no. 7, p. e11376, 2020.

[209] A. Géron, *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. " O'Reilly Media, Inc.", 2022.

[210] S. Raschka, *Python machine learning*. Packt publishing ltd, 2015.

[211] A. M. Ikotun, A. E. Ezugwu, L. Abualigah, B. Abuhaija, and J. Heming, "K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data," *Information Sciences*, 2022.

[212] M. Ahmed, R. Seraj, and S. M. S. Islam, "The k-means algorithm: A comprehensive survey and performance evaluation," *Electronics*, vol. 9, no. 8, p. 1295, 2020.

[213] H. Patel and D. Patel, "A brief survey of data mining techniques applied to agricultural data," *International Journal of Computer Applications*, vol. 95, no. 9, 2014.

[214] K. Khan, S. U. Rehman, K. Aziz, S. Fong, and S. Sarasvady, "Dbscan: Past, present and future," in *The fifth international conference on the applications of digital information and web technologies (ICADIWT 2014)*. IEEE, 2014, pp. 232–238.

[215] S. Sun, C. Wang, H. Ding, and Q. Zou, "Machine learning and its applications in plant molecular studies," *Briefings in functional genomics*, vol. 19, no. 1, pp. 40–48, 2020.

[216] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," pp. 2825–2830, 2011.

[217] S. Zhang, Z. Zhou, X. Chen, Y. Hu, and L. Yang, "pdhs-svm: a prediction method for plant dnase i hypersensitive sites based on support vector machine," *Journal of Theoretical Biology*, vol. 426, pp. 126–133, 2017.

[218] X. Chen and H. Ishwaran, "Random forests for genomic data analysis," *Genomics*, vol. 99, no. 6, pp. 323–329, 2012.

[219] J. O. Ogutu, H.-P. Piepho, and T. Schulz-Streeck, "A comparison of random forests, boosting and support vector machines for genomic selection," in *BMC proceedings*, vol. 5, no. 3. BioMed Central, 2011, pp. 1–5.

[220] C. Huyen, *Designing machine learning systems*. " O'Reilly Media, Inc.", 2022.

[221] U. Lee, S. Chang, G. A. Putra, H. Kim, and D. H. Kim, "An automated, high-throughput plant phenotyping system using machine learning-based plant segmentation and image analysis," *PloS one*, vol. 13, no. 4, p. e0196615, 2018.

[222] S. K. Behera, A. K. Rath, A. Mahapatra, and P. K. Sethy, "Identification, classification & grading of fruits using machine learning & computer intelligence: a review," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–11, 2020.

[223] K. Hameed, D. Chai, and A. Rassau, "A comprehensive review of fruit and vegetable classification techniques," *Image and Vision Computing*, vol. 80, pp. 24–44, 2018.

[224] V. Meshram, K. Patil, V. Meshram, D. Hanchate, and S. Ramkteke, "Machine learning in agriculture domain: A state-of-art survey," *Artificial Intelligence in the Life Sciences*, vol. 1, p. 100010, 2021.

[225] A. M. Jiménez-Carvelo, A. González-Casado, M. G. Bagur-González, and L. Cuadros-Rodríguez, "Alternative data mining/machine learning methods for the analytical evaluation of food quality and authenticity–a review," *Food research international*, vol. 122, pp. 25–39, 2019.

[226] J. D. Kelleher, *Deep learning*. MIT press, 2019.

[227] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[228] J. Brownlee, *Better deep learning: train faster, reduce overfitting, and make better predictions*. Machine Learning Mastery, 2018.

[229] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[230] A. Krizhevsky, "One weird trick for parallelizing convolutional neural networks," *arXiv preprint arXiv:1404.5997*, 2014.

[231] D. T. Pham and X. Liu, "Training of elman networks and dynamic system modelling," *International Journal of Systems Science*, vol. 27, no. 2, pp. 221–226, 1996.

[232] L. Medsker and L. C. Jain, *Recurrent neural networks: design and applications*. CRC press, 1999.

[233] D. Rothman, *Transformers for Natural Language Processing: Build innovative deep neural network architectures for NLP with Python, PyTorch, TensorFlow, BERT, RoBERTa, and more*. Packt Publishing Ltd, 2021.

[234] T. Lin, Y. Wang, X. Liu, and X. Qiu, "A survey of transformers," *AI Open*, 2022.

[235] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A survey of convolutional neural networks: analysis, applications, and prospects," *IEEE transactions on neural networks and learning systems*, 2021.

[236] W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: A comprehensive review," *Neural computation*, vol. 29, no. 9, pp. 2352–2449, 2017.

[237] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep learning for 3d point clouds: A survey," *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[238] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.

[239] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," *Advances in neural information processing systems*, vol. 30, 2017.

[240] B.-S. Hua, M.-K. Tran, and S.-K. Yeung, "Pointwise convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 984–993.

[241] M. Engelcke, D. Rao, D. Z. Wang, C. H. Tong, and I. Posner, "Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 1355–1361.

[242] A. N. Wu, R. Stouffs, and F. Biljecki, "Generative adversarial networks in the built environment: A comprehensive review of the application of gans across data types and scales," *Building and Environment*, p. 109477, 2022.

[243] L. Jiang, S. Shi, X. Qi, and J. Jia, "Gal: Geometric adversarial loss for single-view 3d-object reconstruction," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 802–816.

[244] Y. Yu, Z. Huang, F. Li, H. Zhang, and X. Le, "Point encoder gan: A deep learning model for 3d point cloud inpainting," *Neurocomputing*, vol. 384, pp. 192–199, 2020.

[245] T. Shinohara, H. Xiu, and M. Matsuoka, "Point2color: 3d point cloud colorization using a conditional generative network and differentiable rendering for airborne lidar," in *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, 2021, pp. 1062–1071.

[246] J. Huang, J. Yuan, and C. Qiao, "Generation for adaption: A gan-based approach for 3d domain adaption with point cloud data," *arXiv preprint arXiv:2102.07373*, 2021.

[247] D. Valsesia, G. Fracastoro, and E. Magli, "Learning localized generative models for 3d point clouds via graph convolution," in *International conference on learning representations*, 2018.

[248] D. W. Shu, S. W. Park, and J. Kwon, "3d point cloud generative adversarial network based on tree structured graph convolutions," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 3859–3868.

[249] P. Singh, A. Tiwari, K. Sadekar, and S. Raman, "Treegcn-ed: A tree-structured graph-based autoencoder framework for point cloud processing," *Pacific Graphics Short Papers and Posters*, 2023.

[250] K. Mo, P. Guerrero, L. Yi, H. Su, P. Wonka, N. Mitra, and L. J. Guibas, "Structurenet: Hierarchical graph networks for 3d shape generation," *arXiv preprint arXiv:1908.00575*, 2019.

[251] Y. Li and G. Baciu, "Hsgan: Hierarchical graph learning for point cloud generation," *IEEE Transactions on Image Processing*, vol. 30, pp. 4540–4554, 2021.

[252] C. Chen, D. Liu, C. Xu, and T.-K. Truong, "Genecgan: A conditional generative adversarial network based on genetic tree for point cloud reconstruction," *Neurocomputing*, vol. 462, pp. 46–58, 2021.

[253] B. Wang, J. Lan, and J. Gao, "Msg-point-gan: Multi-scale gradient point gan for point cloud generation," *Symmetry*, vol. 15, no. 3, p. 730, 2023.

[254] B. Ghojogh, A. Ghodsi, F. Karray, and M. Crowley, "Generative adversarial networks and adversarial autoencoders: Tutorial and survey," *arXiv preprint arXiv:2111.13282*, 2021.

[255] M. Zamorski, M. Zięba, P. Klukowski, R. Nowak, K. Kurach, W. Stokowiec, and T. Trzciński, "Adversarial autoencoders for compact representations of 3d point clouds," *Computer Vision and Image Understanding*, vol. 193, p. 102921, 2020.

[256] D. Bulanon and T. Kataoka, "Fruit detection system and an end effector for robotic harvesting of fuji apples," *Agricultural Engineering International: CIGR Journal*, vol. 12, no. 1, 2010.

[257] F. J. Y. Narváez, J. S. del Pedregal, P. A. Prieto, M. Torres-Torriti, and F. A. A. Cheein, "Lidar and thermal images fusion for ground-based 3d characterisation of fruit trees," *Biosystems Engineering*, vol. 151, pp. 479–494, 2016.

[258] J. Feng, L. Zeng, and L. He, "Apple fruit recognition algorithm based on multi-spectral dynamic image analysis," *Sensors*, vol. 19, no. 4, p. 949, 2019.

[259] A. Coupel-Ledru, B. Pallas, M. Delalande, F. Boudon, E. Carrié, S. Martinez, J.-L. Regnard, and E. Costes, "Multi-scale high-throughput phenotyping of apple architectural and functional traits in orchard reveals genotypic variability under contrasted watering regimes," *Horticulture research*, vol. 6, 2019.

[260] J.-P. Rojas-Bustos, A. Branthomme, E. Costes, and F. Boudon, "Comparison between uav and terrestrial lidar scans for high throughput phenotyping of architectural traits of a core collection of apple trees," in *XXXI International Horticultural Congress (IHC2022): III International Symposium on Mechanization, Precision Horticulture, and 1360*, 2022, pp. 15–22.

[261] D. Girardeau-Montaut, "Cloudcompare," 2023. [Online]. Available: https://www.danielgm.net/cc/

[262] N. Brodu and D. Lague, "3d terrestrial lidar data classification of complex natural scenes using a multi-scale dimensionality criterion: Applications in geomorphology," *ISPRS journal of photogrammetry and remote sensing*, vol. 68, pp. 121–134, 2012.

[263] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham, "Learning semantic segmentation of large-scale point clouds with random sampling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 11, pp. 8338–8354, 2021.

[264] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux, "API design for machine learning software: experiences from the scikit-learn project," in *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 2013, pp. 108–122.

[265] H. Edelsbrunner, D. Kirkpatrick, and R. Seidel, "On the shape of a set of points in the plane," *IEEE Transactions on information theory*, vol. 29, no. 4, pp. 551–559, 1983.

[266] K. E. Bellock, "alphashape: Toolbox for generating alpha shapes." 2019. [Online]. Available: https://alphashape.readthedocs.io/en/latest/index.html

[267] C. Massonnet, J.-L. Regnard, P. Lauri, E. Costes, and H. Sinoquet, "Contributions of foliage distribution and leaf functions to light interception, transpiration and photosynthetic capacities in two apple cultivars at branch and tree scales," *Tree physiology*, vol. 28, no. 5, pp. 665–678, 2008.

[268] J. A. den Dulk, *The interpretation of remote sensing: a feasibility study*. Wageningen University and Research, 1989.

[269] R. Team, *RStudio: Integrated Development Environment for R*, RStudio, PBC., Boston, MA, 2020. [Online]. Available: http://www.rstudio.com/

[270] H. Wickham, *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016. [Online]. Available: https://ggplot2.tidyverse.org

[271] B. Brede, K. Calders, A. Lau, P. Raumonen, H. M. Bartholomeus, M. Herold, and L. Kooistra, "Non-destructive tree volume estimation through quantitative structure modelling: Comparing uav laser scanning with terrestrial lidar," *Remote Sensing of Environment*, vol. 233, p. 111355, 2019.

[272] M. Kruuvcek, K. Král, K. C. Cushman, A. Missarov, and J. R. Kellner, "Supervised segmentation of ultra-high-density drone lidar for large-area mapping of individual trees," *Remote Sensing*, vol. 12, no. 19, p. 3260, 2020.

[273] D. Wu, K. Johansen, S. Phinn, A. Robson, and Y.-H. Tu, "Inter-comparison of remote sensing platforms for height estimation of mango and avocado tree crowns," *International Journal of Applied Earth Observation and Geoinformation*, vol. 89, p. 102091, 2020.

[274] Food Agriculture Organization; World Health Organization, "Food and Agriculture Organization Assuring Food Safety and Quality," *Food and Nutrition Paper*, vol. 76, 2003. [Online]. Available: ftp://ftp.fao.org/docrep/fao/006/y8705e/y8705e00.pdf

[275] T. I. Year, *Fruit and vegetables – your dietary essentials*. FAO, 2020.

[276] I. Keller and C. Tukuitonga, "The who/fao fruit and vegetable promotion initiative," in *I International Symposium on Human Health Effects of Fruits and Vegetables 744*, 2005, pp. 27–37.

[277] M. Brown and T. Tworkoski, "Pest management benefits of compost mulch in apple orchards," *Agriculture, ecosystems & environment*, vol. 103, no. 3, pp. 465–472, 2004.

[278] JICA, "Vegetable farming techniques manual vegetable farming techniques manual," Japan International Cooperation Agency, Tech. Rep., 2016.

[279] A. Chawade, J. Van Ham, H. Blomquist, O. Bagge, E. Alexandersson, and R. Ortiz, "High-throughput field-phenotyping tools for plant breeding and precision agriculture," *Agronomy*, vol. 9, no. 5, pp. 1–18, 2019.

[280] Y. Huang, Z. Ren, D. Li, and X. Liu, "Phenotypic techniques and applications in fruit trees: A review," *Plant Methods*, vol. 16, no. 1, pp. 1–22, 2020. [Online]. Available: https://doi.org/10.1186/s13007-020-00649-7

[281] D. M. Bulanon, T. F. Burks, and V. Alchanatis, "A multispectral imaging analysis for enhancing citrus fruit detection," *Environmental Control in Biology*, vol. 48, no. 2, pp. 81–91, 2010.

[282] S. Minaee, Y. Y. Boykov, F. Porikli, A. J. Plaza, N. Kehtarnavaz, and D. Terzopoulos, "Image segmentation using deep learning: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[283] P. Druzhkov and V. Kustikova, "A survey of deep learning methods and software tools for image classification and object detection," *Pattern Recognition and Image Analysis*, vol. 26, no. 1, pp. 9–15, 2016.

[284] W. Wang, Q. Lai, H. Fu, J. Shen, H. Ling, and R. Yang, "Salient object detection in the deep learning era: An in-depth survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[285] A. Torfi, R. A. Shirvani, Y. Keneshloo, N. Tavaf, and E. A. Fox, "Natural language processing advancements by deep learning: A survey," *arXiv preprint arXiv:2003.01200*, 2020.

[286] D. W. Otter, J. R. Medina, and J. K. Kalita, "A survey of the usages of deep learning for natural language processing," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 2, pp. 604–624, 2020.

[287] K. Turgut, H. Dutagaci, G. Galopin, and D. Rousseau, "Segmentation of structural parts of rosebush plants with 3d point-based deep learning methods," *arXiv preprint arXiv:2012.11489*, 2020.

[288] R. Jayakumari, R. R. Nidamanuri, and A. M. Ramiya, "Object-level classification of vegetable crops in 3d lidar point cloud using deep learning convolutional neural networks," *Precision Agriculture*, pp. 1–17, 2021.

[289] M. Ghahremani, K. Williams, F. M. Corke, B. Tiddeman, Y. Liu, and J. H. Doonan, "Deep segmentation of point clouds of wheat," *Frontiers in Plant Science*, vol. 12, p. 429, 2021.

[290] X. Chen, K. Jiang, Y. Zhu, X. Wang, and T. Yun, "Individual tree crown segmentation directly from uav-borne lidar data using the pointnet of deep learning," *Forests*, vol. 12, no. 2, p. 131, 2021.

[291] S. Artzet, J. P. Rojas, B. Pallas, E. Costes, and F. Boudon, "Machine and deep learning based identification of organs within lidar scans of tree canopies: Application to the estimation of apple production," in *FSPM2020*. Institute of Horticultural Production Systems, 2020.

[292] C. Pradal, F. Boudon, C. Nouguier, J. Chopard, and C. Godin, "Plantgl: a python-based geometric library for 3d plant modelling at different scales," *Graphical models*, vol. 71, no. 1, pp. 1–21, 2009.

[293] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *2009 IEEE international conference on robotics and automation*. IEEE, 2009, pp. 3212–3217.

[294] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham, "Randla-net: Efficient semantic segmentation of large-scale point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 108–11 117.

[295] A. Rosenberg and J. Hirschberg, "V-measure: A conditional entropy-based external cluster evaluation measure," in *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*, 2007, pp. 410–420.

[296] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *Journal of Big data*, vol. 3, no. 1, pp. 1–40, 2016.

[297] 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.

[298] 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.

[299] C. R. Qi, O. Litany, K. He, and L. J. Guibas, "Deep hough voting for 3d object detection in point clouds," in *proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 9277–9286.

[300] D. Li, G. Shi, J. Li, Y. Chen, S. Zhang, S. Xiang, and S. Jin, "Plantnet: A dual-function point cloud segmentation network for multiple plant species," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 184, pp. 243–263, 2022.

[301] D. Li, J. Li, S. Xiang, and A. Pan, "Psegnet: simultaneous semantic and instance segmentation for point clouds of plants," *Plant Phenomics*, vol. 2022, 2022.

[302] K. Turgut, H. Dutagaci, G. Galopin, and D. Rousseau, "Segmentation of structural parts of rosebush plants with 3d point-based deep learning methods," *Plant Methods*, vol. 18, no. 1, p. 20, 2022.

[303] H. Balta, J. Velagic, W. Bosschaerts, G. De Cubber, and B. Siciliano, "Fast statistical outlier removal based method for large 3d point clouds of outdoor environments," *IFAC-PapersOnLine*, vol. 51, no. 22, pp. 348–353, 2018.

[304] M. Vlaminck, L. Diels, W. Philips, W. Maes, R. Heim, B. D. Wit, and H. Luong, "A multisensor uav payload and processing pipeline for generating multispectral point clouds," *Remote Sensing*, vol. 15, no. 6, p. 1524, 2023.

[305] F. Boudon, "Course notes," Class Slides, 2020, accessed: 2020–11-06.

[306] S. Y. Cheong, *Hands-On Image Generation with TensorFlow: A practical guide to generating images and videos using deep learning*. Packt Publishing Ltd, 2020.

[307] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, "Generative Adversarial Networks: An Overview," *IEEE Signal Processing Magazine*, vol. 35, no. 1, pp. 53–65, 2018.

[308] K. Yin, H. Huang, D. Cohen-Or, and H. Zhang, "P2p-net: Bidirectional point displacement net for shape transform," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, pp. 1–13, 2018.

[309] J. Li, J. Zhou, Y. Xiong, X. Chen, and C. Chakrabarti, "An adjustable farthest point sampling method for approximately-sorted point cloud data," in *2022 IEEE Workshop on Signal Processing Systems (SiPS)*. IEEE, 2022, pp. 1–6.

[310] N. Bhatia *et al.*, "Survey of nearest neighbor techniques," *arXiv preprint arXiv:1007.0085*, 2010.

[311] C.-L. Li, M. Zaheer, Y. Zhang, B. Poczos, and R. Salakhutdinov, "Point cloud gan," *arXiv preprint arXiv:1810.05795*, 2018.

[312] S. Ramasinghe, S. Khan, N. Barnes, and S. Gould, "Spectral-gans for high-resolution 3d point-cloud generation," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 8169–8176.

[313] C. Wen, B. Yu, and D. Tao, "Learning progressive point embeddings for 3d point cloud generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10 266–10 275.

[314] V. Zyrianov, X. Zhu, and S. Wang, "Learning to generate realistic lidar point clouds," in *European Conference on Computer Vision*. Springer, 2022, pp. 17–35.

[315] L. Cao, H. Liu, X. Fu, Z. Zhang, X. Shen, and H. Ruan, "Comparison of uav lidar and digital aerial photogrammetry point clouds for estimating forest structural attributes in subtropical planted forests," *Forests*, vol. 10, no. 2, p. 145, 2019.

[316] M. G. Raman, E. F. Carlos, and S. Sankaran, "Optimization and evaluation of sensor angles for precise assessment of architectural traits in peach trees," *Sensors*, vol. 22, no. 12, p. 4619, 2022.

[317] Y. Jiang, C. Li, F. Takeda, E. A. Kramer, H. Ashrafi, and J. Hunter, "3d point cloud data to quantitatively characterize size and shape of shrub crops," *Horticulture research*, vol. 6, 2019.

[318] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[319] X. Wang, Y. Jin, Y. Cen, T. Wang, and Y. Li, "Attention models for point clouds in deep learning: a survey," *arXiv preprint arXiv:2102.10788*, 2021.

[320] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[321] D. Lu, Q. Xie, M. Wei, K. Gao, L. Xu, and J. Li, "Transformers in 3d point clouds: A survey," *arXiv preprint arXiv:2205.07417*, 2022.

[322] R. Xu, L. Hui, Y. Han, J. Qian, and J. Xie, "Transformer-based point cloud generation network," in *Proceedings of the 31st ACM International Conference on Multimedia*, 2023, pp. 4169–4177.

[323] C. Wen, J. Long, B. Yu, and D. Tao, "Pointwavelet: Learning in spectral domain for 3d point cloud analysis," *arXiv preprint arXiv:2302.05201*, 2023.

[324] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.

[325] T. Li, Y. Fu, X. Han, H. Liang, J. J. Zhang, and J. Chang, "Diffusionpointlabel: Annotated point cloud generation with diffusion model," in *Computer Graphics Forum*, vol. 41, no. 7. Wiley Online Library, 2022, pp. 131–139.