國立臺灣大學電機資訊學院電機工程學系

碩士論文

Department of Electrical Engineering
College of Electrical Engineering and Computer Science
National Taiwan University

Master Thesis

基於機器手臂和視覺反饋控制的表型分析系統

A Robotic and Visual Feedback Phenotyping System

杜晟道

Sheng-Dao Du

指導教授：連豐力 博士

Advisor: Feng-Li Lian, Ph.D.

中華民國 112 年 7 月

July 2023

doi:10.6342/NTU202303769

doi:10.6342/NTU202303769

誌謝

時光荏苒，白駒過隙，短暫而充實的研究生生涯要在這一刻，被努力與時間畫上一個句號。

初見**連豊力**教授的時候，他正準備騎車離開明達館，我上前詢問，您是不是連豊力教授，經過短暫的詢問后，連教授請我來聽下午的專題討論，我知道，我的碩士生涯就要開始了。教授是一個對學術很嚴厲的人，從他對論文的格式要求，從教導我們篩選優劣論文的能力，或是從相當嚴謹的數據表達的方式，體現地淋漓盡致。儘管教授表面上看起來不苟言笑，但他其實對學生們都非常地用心負責，每周一次的進度報告，給我們這一周以來，最快速與准確的反饋，輪流的英文期刊報告，提高我們用英文演講與聆聽的能力。正所謂授人與魚，不如授人與漁，他正是那個教會我們如何去做研究的那個人，在這裏，我鄭重地向**連豊力**教授表達我由衷的致意。

實驗室的夥伴們，也是我在這裏難得的回憶。謝謝**国郡**，跟我討論研究方向的確立，很多次向你取經一些基礎的知識，你總是很積極的回復我。謝謝**伯佑**和**詠隆**，我們都是在做機器人相關項目，所以很常能聊到盡興。謝謝**王捷**，騎單車的日子分外難忘。**世興**，當初寫論文的時候我們互相鼓勵，我會記得在電機二館討論室的日子。**仲宇**，我碩士中的好夥伴也是好朋友，一起經歷了很多，一起討論學術問題，一起騎車環臺北，一起吃飯、打趣，我不會忘記。**宇强**和**佳芸**，謝謝你們曾經給我學術上的幫助，也很開心跟你們有一次完美的露營生活。謝謝賴橋曾給我的研究建議。謝謝**高達**，跟我分享很多研究方法，在我低谷的時候給我幫助。**雁丞**我會記得，在 101 夜幕下的剪影。**晁維**，我們的生物鐘非常相似，每次都能在晚上一起聊天討論。還有**暄坐**，你是我的機器手臂的學弟，一起討論如何設計實驗，難忘。每次來實驗室也會跟**之蕙**聊天，開心且放鬆。**旅青**，一起討論農業機器人，一起吃飯、玩耍，是美好的回憶。**芳緯**，你是個值得信任的夥伴，在溫室實驗的日子有你很好。還有其他碩一的同學們，**芳源**、**昱彣**還有**昀誠**，謝謝你們，無論是作爲朋友，還是作爲一個研究夥伴。最後謝謝謝**昱翔**博士，在論文寫作和人生規劃上的幫忙，**宜儒**博士，在開發研華應用上的支援。以及**有田**學長對我的規劃上的指點與幫助。謝謝我遠在德國和美國的哥們**林森**和**瀚元**的陪伴，你們是我充電的地方，希望有一天與你們見面。

還有，**仁仁**同學，你的出現，是我的幸運，謝謝你曾經的鼓勵與陪伴。

謝謝在背後支持我的家人，**姑姑**、**爺爺**、**爸爸媽媽**，沒有你們的幫助，我不可能有今天的成果。

都說人與人的緣分是分段的，也許畢業後大家很難保持像這樣的聯絡，但曾經的種種，我的心裏，早已預留好了一塊地方，留著你們的笑與我們的那些年。

杜晟道 謹誌

中華民國一百一拾二年七月

doi:10.6342/NTU202303769

doi:10.6342/NTU202303769

基於機器手臂和視覺反饋控制的表型分析系統

研究生：杜晟道　　指導教授：連豊力 博士

國立臺灣大學　電機工程學系

摘要

種植一個具有高經濟價值的農作物往往需要投入大量的人力和物力，農夫需要對作物的生長資訊進行記錄以監控植物生長狀況與調整種植策略，需要對影響植物產量的因素進行干預，例如植物的病蟲害的檢測。儘管農夫可以很好地完成大部分的工作，但人類無法全天候地工作，開發一個機器人輔助植物檢測系統，不僅能夠節省農夫的時間，對相關問題進行標準化處理，還能夠在任何時間對植物進行檢測。

然而，任何高階任務都需要具有魯棒性的基礎信息來完成，例如，植物葉片病蟲害檢測首先需要拍攝到葉片的全貌，即需瞭解葉片在空間的位置，切除受病的葉片和摘取水果也需要事先知道葉柄和果實的位置，檢測植物的高度與節點數量、分割葉片葉柄需要事先對植物點雲進行重建。已知植物點雲將為上述高階任務提供操作對象的基礎。

doi:10.6342/NTU202303769

本文針對植物點雲重建，利用以標記為基礎的空間植物定位方法，提出了一個多階層植物點雲重建策略，包含對植物的快速掃描和主蔓遮擋重建算法。在流程內，開發了包含但不限於植物點雲濾波算法，多點雲疊合算法，主蔓提取算法，遮擋檢測算法和無遮擋視角優化器。由於主蔓連接這葉片和葉柄，所以主蔓的完整性影響著植物點雲分割的質量，故主蔓遮擋重建算法將對被遮擋的主蔓以最優視角進行重建。

實驗結果證明，本文開發的基於點云關係的主蔓提取算法具有較高的魯棒性，所提出的流程能夠完成定位、植物重建、最優視角重新規劃與植物點雲分割。

關鍵字：

機器人，植物模型重建，視覺反饋，農業機器人，植物表型，植物高度檢測，植物節點檢測，機器視覺，植物點雲重建，ICP，洋香瓜，葫蘆科

doi:10.6342/NTU202303769

A Robotic and Visual Feedback Phenotyping System

Student: Sheng-Dao Du Advisor: Dr. Feng-Li Lian

Department of Electrical Engineering

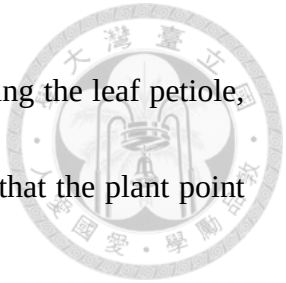
National Taiwan University

ABSTRACT

Planting a fruit with high economic value often requires a large amount of labor and material input. Farmers need to record information about the growth of the crop in order to monitor the growth status of the plants and adjust planting strategies, and need to intervene in factors that affect crop yield, such as the detection of plant diseases and pests. Although farmers can do most of the work well, humans are unable to work around the clock. Developing a robot assistance plant detection system not only saves the farmer's time, standardizes the handling of related issues, but also can inspect the plants at any time.

However, any high-level task requires robust basic information to be completed, such as, plant leaf pest detection first requires capturing the overall appearance of the leaf, that is, knowing the position of the leaf in 3D space. Cutting off the infected leaf and harvesting the fruit also requires knowing the position of the petiole and fruit beforehand.

doi:10.6342/NTU202303769



Detecting the height and number of nodes of the plant, and segmenting the leaf petiole, require reconstructing the plant point cloud in advance. It is known that the plant point cloud will provide the basis for the above high-level tasks to operate.

This paper proposes a multi-stage plant point cloud reconstruction strategy based on a spatial plant localization method based on fiducial marker, including a fast scan and main vine occlusion reconstruction algorithm for plants. In the process, a plant point cloud filtering algorithm, a pointcloud alignment algorithm, a main stem extraction algorithm, , an occlusion detection algorithm and the occluding-free viewpoint optimizer are developed, but are not limited to these.

The experimental results show that the main vine extraction algorithm based on point cloud relationships developed in this paper is highly robust, and the proposed process can complete positioning, plant reconstruction, optimal view angle re-planning and plant point cloud segmentation. For the segmented point cloud, the robot arm can plan a trajectory to touch and simulate subsequent tasks.

Keywords:

Robotic, Melon, Cucurbitaceae, Model Reconstruction, Phenotype, Agricultural Application, Height Measurement, Node Counting, Machine Vision.

doi:10.6342/NTU202303769

CONTENTS

doi:10.6342/NTU202303769

doi:10.6342/NTU202303769

LIST OF FIGURES

doi:10.6342/NTU202303769

doi:10.6342/NTU202303769

doi:10.6342/NTU202303769

doi:10.6342/NTU202303769

LIST OF TABLES

doi:10.6342/NTU202303769

doi:10.6342/NTU202303769

Chapter 1
Introduction

In this chapter, the motivation will be shown in Section 1.1. The motivation and problem formulation will be discussed in Section 1.2. The contribution and the organization of the rest of this thesis are provided in Section 1.3 and 1.4, respectively.

1.1 Motivation

Agriculture is the practice of cultivating plants and livestock [1 International Labour Organization 1999]. The importance of agriculture is self-evident because it provides basic living materials and contributes considerable commercial revenue. In the United States, from 2019 to 2020, for example, the contribution of agricultural output reached 133.0 billion dollars, which accounts for 0.6 % of total GDP [2 USDA 2021]. Also, in India, the proportion of GDP contribution is 2.06% until 2021 September [3 Trading Economics 2022].

The agricultural forms could be generally divided into traditional and precise agriculture with automation. A conventional skilled farmer should know many best properties of their planting condition like **crop situation, soil nutrients**, illumination condition, temperature, humidity and irrigation approaches.

doi:10.6342/NTU202303769

Figure 1.1 An Ethiopia Farmer is working in the field [76]

In Figure 1.1, an Ethiopia farmer is dealing with irrigation stuff in the field. In this country, over 70 percentage of population are farmers and they contribute more than 40 percent GDPs for their country. However, these farmers pay huge attention for low productivity and profit. In Figure 1.2, an Indian farmer is levering their livestock to plow the earth as a preparation of cultivating plants. This type of farming is called primitive subsistence farming, which the amount of harvested food only demands for their family. Similarly, this cultivating method is still with low efficiency and productivity.

doi:10.6342/NTU202303769

Figure 1.2 Some Indian still uses the livestock to plow the earth [77]

Generally, the experienced farmer adjusts the cultivating process with experimental knowledge, a small-scale, human-variant and costly method. Nevertheless, without the help of agricultural industrial, a limited number of farmers can not cover an enormous farming area. Although the farmer faces many challenges like scorching sun and fatigue, which are tedious, exhausted and carcinogenic, the farming inputs and yields are not proportionally-depended. Also, taking care of fruit or vegetable is time-consuming and expensive, especially in harvesting. In Figure 1.3, the farmer is harvesting the ripe rice in the yellow rice field. Working long time to keep this pose is tired and may also cause potential spine damage. According to the United States department of Agriculture report [4 Astill 2020], the total human labor cost (including harvesting) of lettuce peaked at

doi:10.6342/NTU202303769

almost 50 per cent. That of fresh tomatoes, spinach, and peaches ended up at 43%, 38% and 36%, respectively.

Figure 1.3 A farmer is harvesting the ripe rice by stooping her body

Therefore, the emergence of large-scale automation devices facilitates the development of commercial agriculture. The automation trucks could response multiple works like plowing the earth, weeding or harvesting.

By changing the specific end-effector of the truck, like a rotational plow, the truck turns to an automatic large-scale plowing machine. Figure 1.4 shows a truck loosening the soil in large area farming field. The length of plow machine up to dozens of meters which is much efficient and time-saving method for large-scale commercial field.

doi:10.6342/NTU202303769

Figure 1.4 The ploughing machine is loosening the soil in large area of field [79]

In addition, a flame throwing tractor in Figure 1.5 is ejecting the flammable gas to combust the unnecessary weeds:

Figure 1.5 The flame trucker is combusting the unwanted weed in preparation stage [80]

doi:10.6342/NTU202303769

Also, Figure 1.6 shows the large-scale agricultural truck working in the enormous field to harvest the ripe rice:

Figure 1.6 Large Scale Agricultural Industry [81]

In Taiwan, the food self-sufficiency rate dropped by 23.3% from 56% within 28 years [5 Merit Times 2013]. Ageing and shortage of agricultural labour force are the main reasons for this phenomenon. Under this scenario, several agricultural apparatuses have already been developed in the commercial market, coming into presence for the past two decades. These types of machinery provide several aspects of applications ranging from ploughing land [6 FarmBot 2022], seeding, watering, spraying [7 Yallappa et al. 2017], weeding [8 DINO 2022], to harvesting [9 Xiong et al. 2020] robots. Most of them are large-scale machinery like tractors, drones and multi-wheeled robotics, which take responsibility for a gigantic arable land.

doi:10.6342/NTU202303769

For a local cultivation example in Taiwan, high mountain tea is a feature that Taiwan want to present to the world, most of which are grown on undulating hillsides. Tea trees are prone to be infested by various pests, and biological control often does not work. So, they have to deploy many human resources to spray medicines for pest control. They carried heavy liquid insecticides working over the mountains shown in Figure 1.7. This is not cost-effective in the labor force and does not fulfil sustainable development requirements. A spraying robot can more effectively meet the needs of tea farmers. They only need to designate the area where the robot will fly, and through the path planning algorithm, the robot can optimally complete the task. Also, apple or peach farmers can copy this template to their fruit mountain.

Figure 1.7 A worker is spraying the pesticide in the high-mountain tea bushes [82]

doi:10.6342/NTU202303769

Japan, a highly developed country with social severe population ageing conditions (20% of the population are over 65 years old [10 Dong 2020]), also is encountering a significant labour force shortage, especially in the traditional industry. According to the current news [11 TVBS 2022], in the COVID-19 Pandemic period, most farms faced a dilemma that seldom workers were willing to participate in agricultural work. The local workers usually resign because they want to start their own business or are too senior enough to quit, while foreign workers can not apply for a job visa under this pandemic. Figure 1.8 shows a 70-years-old farmer working in the field.

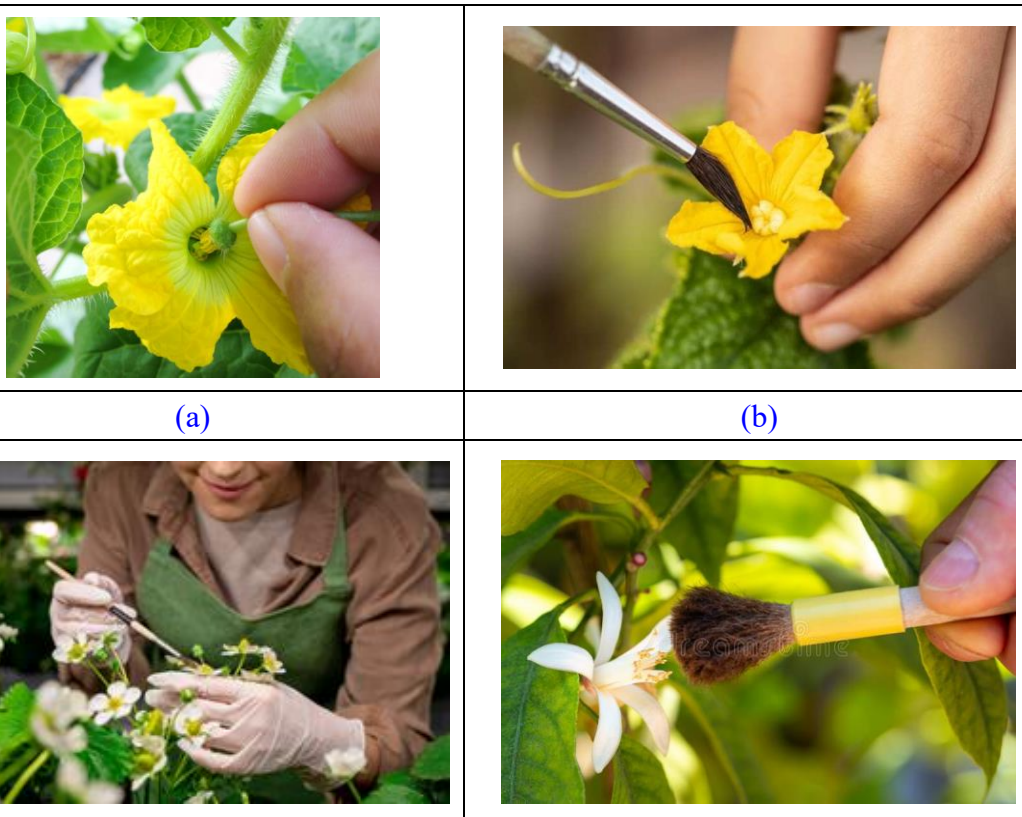
Figure 1.8 A senior farmer in the field [94]

The types of agriculture are varying, depended on the climate, temperature, culture. The cultivating environment could be open area or semi-open greenhouse.

doi:10.6342/NTU202303769

Considering the greenhouse is the main research environment in this paper. There are varieties of vegetables and fruits cultivated in the greenhouse because this environment is much warmer than open environment. The warm condition facilitates the growth of the plant. Without any automation tools in the greenhouse, the works of human could roughly range from nutrient solution preparation, seeding, watering, pollination, de-leafing, pest checking and removing, and fruit harvesting.

Pollination is an importance procedure to fertilize the plant. The fertilized female follower grows into fruit within several weeks. The following Figure shows the human pollination processing:



(a) (b)

(c) (d)

Figure 1.9 Human pollination [12, 13, 14, 15]

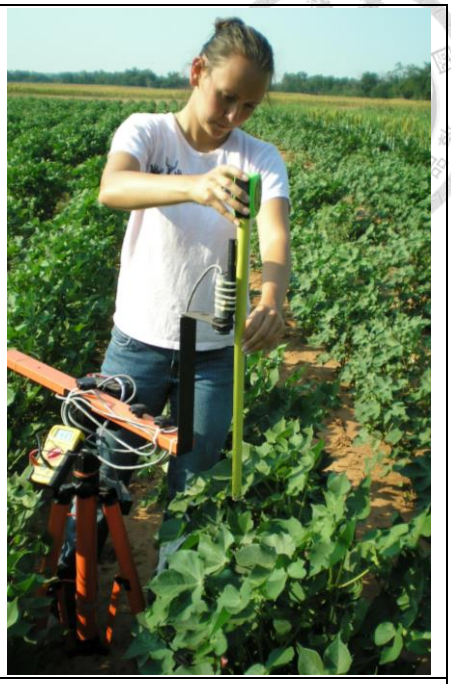
doi:10.6342/NTU202303769

The process of artificial pollination is roughly as follows: First, observe the flowering state of the female flower of the plant every day. If there is a flowering female flower, remove the male flower of the plant, and touch the stigma of the female flower with the corsage stamen in the limited time window in the morning to complete pollination. Generally, such a time window is very short, and the petals will close for a period of time after the female flower blooms, and pollination cannot be completed by then. Therefore, artificial pollination is a task with a high priority in time. In Figure 1.9, four case of artificial pollination are given to represent the previous mentioned procedure.

In order to measure the plant growth condition, the plant height is regarded as straightforward method without sophisticated analyzing like phytomass (dry mass). The measurement of the plant height is time-consuming with human-observing error. The greenhouse crew have to work around and measures the plant height sequentially. The commonly used tool is a roll ruler, which is hard to guarantee that the ruler from the apical meristem to the ground is perpendicular. If there has some subtle angle error, the measured plant height is not accurate. The following figure describes the human plant height measurement procedure in a potted plant greenhouse. In Figure 1.10, the human is measuring the plant height using the roll ruler, which is the most manifest information to measure the plant growing condition.

doi:10.6342/NTU202303769

(a) Measuring the plant height in a greenhouse [16]



(b) Measuring Plant Height in open filed [17]

Figure 1.10 A greenhouse crew is measuring the plant height

Also, other environment variables like soil condition and illuminance intensity are measured by human to track their cultivating condition. Further changes might be applied under the environment information. In Figure 1.11, the left figure is the EC (Electrical Conductivity) and PH value which represent the density of the nutrient solution and the concentrations of hydrogen ions or hydroxide ions in this solution. The right part of this figure shows a worker measuring the plant light condition.

doi:10.6342/NTU202303769

(a) measure the pH and EC

(b) measure the illuminance [18]

Figure 1.11 The measurement of plant growing environment

Furthermore, plants may be infested by various pests during the growth process, causing plants to suffer from diseases. Timely detection and removal of diseased leaves can largely prevent the spread of the disease. Under manual operation conditions, the plants need to be inspected regularly, and it will consume a lot of working time to detect each leaf. In addition, not every worker has the ability to distinguish different types of plant diseases. Therefore, an automatic disease identification equipment and automatic de-leaf equipment will greatly help farmers improve efficiency and stay health. Figure 1.12 show the scenario of identification and removal disease plant part by human:

(a) Fungal attack stock [19]

(b) Rose black spot [20]

Figure 1.12 Identification of disease leaf and removal by human

doi:10.6342/NTU202303769

In addition, the part that accounts for the largest labor cost is the harvest of fruits. First of all, the action of harvesting also takes a lot of time, and the quality of the fruits picked by people with different experiences is not nearly the same. In addition, judging the ripeness of fruit is also quite a work based on experience, and the judgment standards of different picking workers will have certain deviations. Also, the transportation of the harvested fruits by human is a strenuous work. Figure 1.13 shows the harvesting of tomato and strawberry by human beings.

(a) Gardener harvesting tomatoes in greenhouse [21]

(b) Strawberry harvest [22]

Figure 1.13 A greenhouse crew is harvesting the tomato

doi:10.6342/NTU202303769

Figure 1.14 describes the current situation of agriculture in recent decades. The market needs more grain and fruit, and farmers have to increase their yield to satisfy the objective within fixed planting areas. For traditional agriculture, the farmers must cultivate plants accurately, which requires various precise modern technologies to implement. Under this trend, the automatic agricultural devices appear on the commercial market one after another. It is essential to monitor plant growth conditions and environmental variables, accurately identify potential diseases, and precisely irrigate, fertilize, weed, insecticide, harvest, and fruit optimized selection to realize this kind of precision agriculture, from seeds to fruits.

Figure 1.14 A Japanese farmer is trying to mitigate workforce shortage using a robotic solution (TVBS News). [11 TVBS 2022]

The automatic or intelligent greenhouse is the heating research point in recent decades. The research topic ranges from detecting plant diseases, removing leaf to robotic

doi:10.6342/NTU202303769

harvesting. Figure 1.15 shows one example of the robotic harvesting procedure. The robotic system is running on a guided rail based on an UGV and a robotic manipulator with a scissors-like end-effector. After locating the root of fruit cluster, the end-effector is moved to the specific stem location to cut this cluster. And then the harvested fruit is transported to the storage bin for later packaging. This robot finished fruit localization, ripeness analyzing and trajectory planning. The motion that the robotic manipulator moves the end-effector to the root of cluster is challenging when the environment contains wind disturbance and the system is under open-loop control.

Figure 1.15 A robotic system is cutting the fruit cluster [23]

Similarly, there has other harvesting robotic system designed for different fruits. For example, in Figure 1.16, a strawberry harvest robotic system consisting of a UGV, robotic

doi:10.6342/NTU202303769

manipulator and a specialized end-effector. Ripeness detection is also important because different growing progress of fruits may exist in same observing region.

(a) Harvesting end-effector [24]

(b) Designed System [25]

Figure 1.16 A strawberry harvesting system

Furthermore, the redundant lower leaf will cause the potential disease after the fruit grown up. Removing these leaves is necessary to keep the quality of the fruit. The following video screenshot (Figure 1.17) shows one case that the robotic system deals with the leaf removing task. This system is an integrated system with a UGV and a robotic manipulator. The mobile system is roaming in the greenhouse to detect these target leaf. Also, the challenge in de-leaf task are divided into two parts: leaf petiole localization and a redundant leaf label / classification algorithm. In this case, the system does not know the entire information of the observed plant. This kind of solution is specific for a limited condition, for example the leaf ahead the fruit, without much sophisticated plant morphology analyzing. The more general de-leaf will deal with the disease or unwanted leaf which may be currently not implemented.

doi:10.6342/NTU202303769

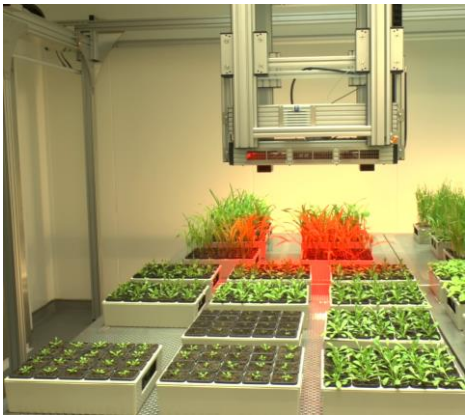
Figure 1.17 The specific robotic system is removing the redundant leaves [26]

In Figure 1.18, the leaf cutting robotic system for cucumber is developed. The edge of specialized end-effector is sticking a blade which cut the petiole region.

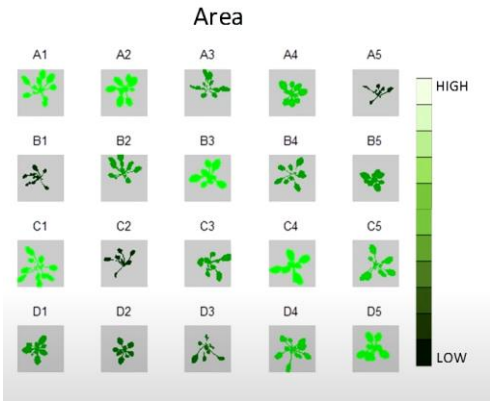
Figure 1.18 Cucumber Leaf Cutting Robotic System [27]

doi:10.6342/NTU202303769

The plant phonotype detection system is usually highly integrated with built-in software and optimized solution for academic or commercial organization. In Figure 1.19, an integrated plant phonotype scanning devices is trying to acquisition the area, bio-stress, dry mass or Chlorophyll content.



(a) Phonotype system



(b) Generated area

Figure 1.19 An Integrated Plant Phonotype System [28]

In addition, the robotic based phonotype detection system is developed to measure the height of the observed crop in Figure 1.20. The leaf size, angle, color and area could also be calculated by this system, which provides the rich information to farm crew about their cultivated plant.

doi:10.6342/NTU202303769

Figure 1.20 The mobile robotic based soybean and crop analyzing system [29]

Therefore, using the automatic system to help the farmer to precisely and efficiently cultivate plant, vegetables, fruits is meaningful with commercial value. In next section, several topics of automatic agriculture will be discussed. And the important part of both these topics is stressed in this paper.

doi:10.6342/NTU202303769

1.2 Problem Formulation

Although the traditional agricultural industrial is an indispensable form agriculture, the precisely automatic agriculture will significantly increase the productivity and reduce the labor cost. For a single plant from seedling to apoptosis stage, the more labor-intensive, repetitive, and non-standardized works, the more automated procedure could substitute and increase the working efficiency and standardize each procedure. However, whether the research object is herbaceous plants or woody plants, they all have common features, that is, there are organs such as leaves, petioles, fruits, male and female flowers, stems or branches. Therefore, the orientation of researches for automation to replace manual work is to study how to detect and operate the organs of these plants.

In real-world, the plant cultivating processing could be simplified as a cylinder-like or box-like topology in 2-Dimensional space. For the system reaches every corner of 2D space, the Unmanned Ground Vehicle is commonly used. Similarly, for a plant located in a specific fix region, a enough range of workspace of robotic manipulator could manipulate the plant within the reachable workspace. Therefore, under this circumstance, we designed some automatic scenario based on potential human interference to the cultivated plant.

The following table shows the basic human operating procedure of fruiting plants, including the operation name and short description:

doi:10.6342/NTU202303769

Table 1.1 Human Operation in Plant Life Cycle		
No.	Operation	Description
1	Seedling	Cultivate from the seed stage. Usually the staff need to soak the seeds, wait for germination, and put them into a constant temperature and humidity box at the same time, and illuminate them for a fixed time.
2	Transplant	Transplant well-grown seedlings (usually with multiple nodes) into a cultivating platform or into the ground, allowing them to grow by absorbing sunlight.
3	Growth Condition Detection	The herbaceous plants grow fast, and there will be changes in height and node every day. The farmer needs to measure the height and node of the observed plant to evaluate the growth condition. Also, the flowing condition is also monitored.
4	Pollination	Female flowers need to be pollinated before fruiting, a task usually done by farmers at certain times in the morning
5	Pests & diseases Detection	Since plants are often attacked by pests and diseases during the growth process, farmers must observe the plants from time to time to find out the area and type of pests in time and do follow-up treatment
6	De-leaf	It is not that the more leaves, the better. During the fruiting stage, the lower leaves do not have enough light time, which is more likely to cause diseases. In addition, removing diseased leaves can also prevent the spread of the disease.
7	Fruit Harvesting	Fruit picking is at the end of a plant's life cycle. Conventional picking needs to be done manually. Picking methods may vary for different fruits. Generally, picking means removing the tissue connecting the fruit and the branch.

The main works of traditional crew caring a plant is not changing too much compared to the mentioned table. Although the robotic system could plan and research the any topic in the table, not every task is suitable or worthy of investing large resources

doi:10.6342/NTU202303769

in research and development, especially when there are other faster solutions. For example, in the seedling stage, the plant from seed to seedling needs time, 5 days in common. The research of this topic is not difficult but the time cost of idle time of robotic manipulator does not meet commercial considerations. Another example is the work of pollination. At present, there are relatively mature worker bees pollinating plants. Regardless of the time of plant flowering, worker bee pollination can have a high success rate. The development of pollination automation is relatively difficult. Not only does it need to remove the anthers of the male flowers, but also precisely rub the pollen against the stigma of the female flowers. With better solutions, it becomes less necessary to develop such automated processes. In the same way, planting requires moving the seedlings to the plant growth platform, which has a large space span, so it is better to use manual treatment in limited automation environment.

In summary, in the process of plant_growth, growth condition detection, pest detection, leaf removal, and fruit harvesting can all be operated using the robot arm platform. The figure below shows the possible working scenarios of the robotic arm platform:

doi:10.6342/NTU202303769

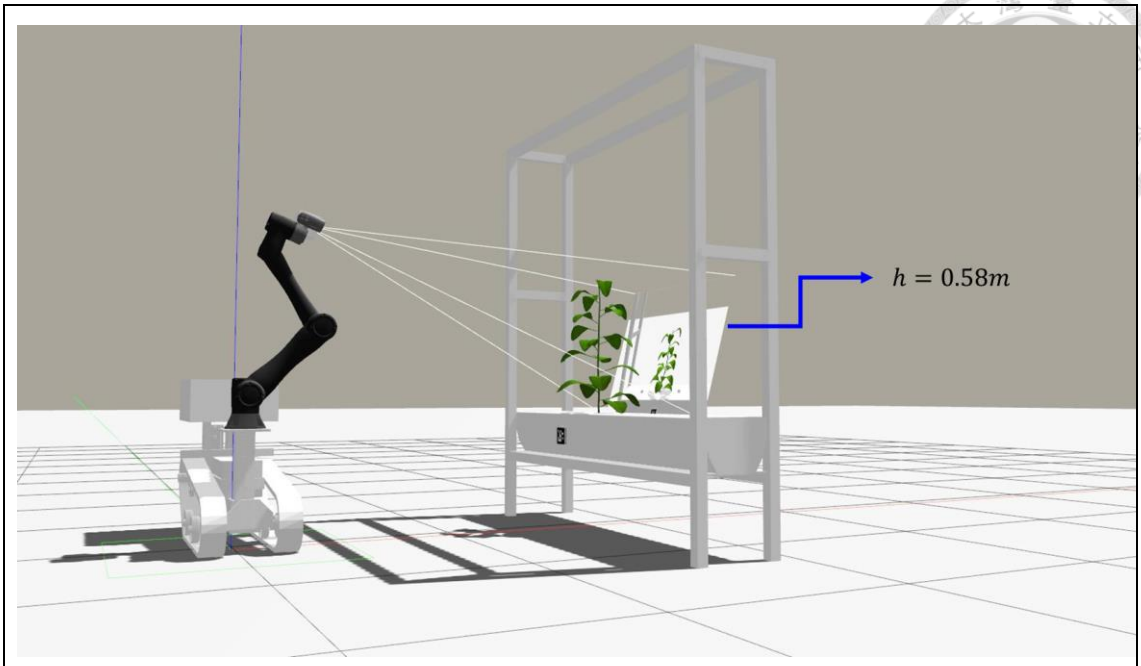
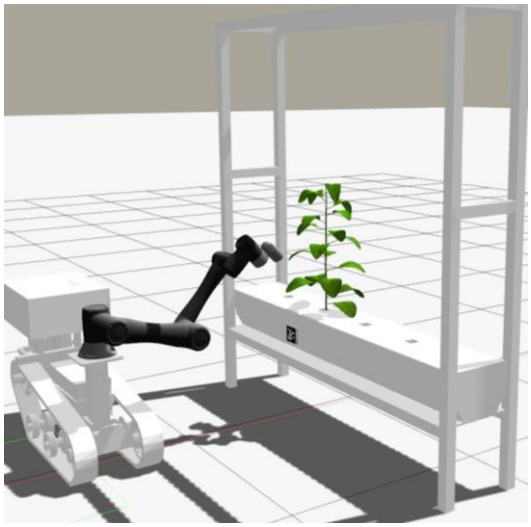
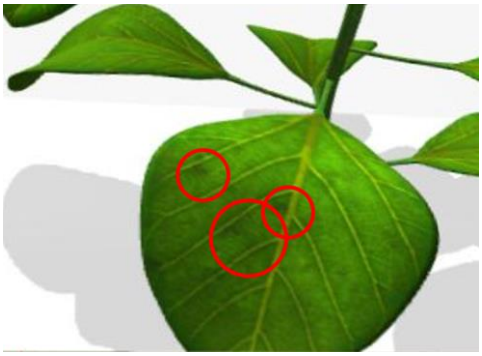


Figure 1.21 Scenario 1: Plant Growth Condition Measurement

In Figure 1.21, the robotic manipulator is trying to calculate the plant height which is a growing condition indicator.



(a) overview



(b) observed leaf with defect (red circle)

Figure 1.22 Scenario 2: Pest and Disease Detection

doi:10.6342/NTU202303769

In Figure 1.22, the robotic manipulator moves more closer to the leaf to evaluate the healthiness of the target leaf.

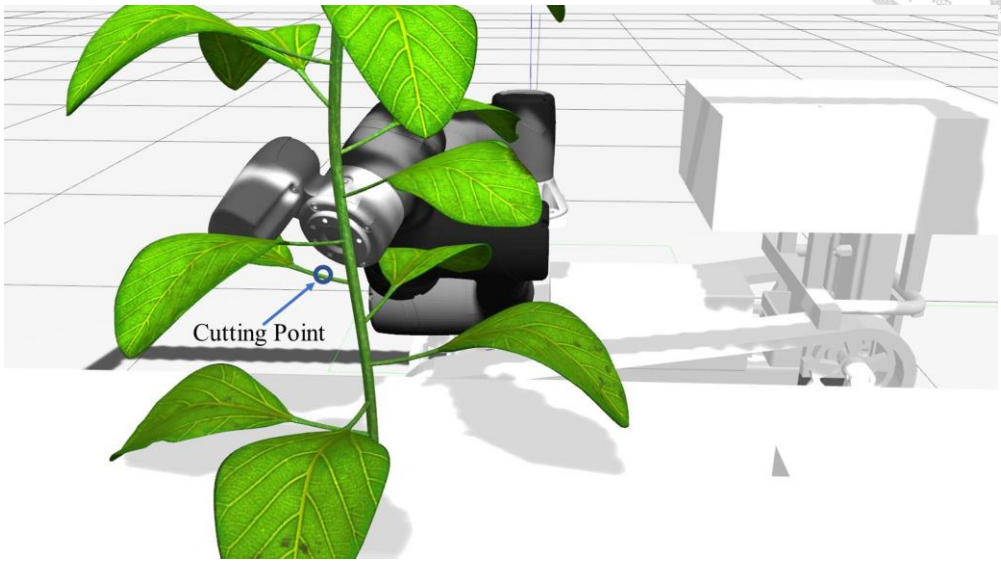


Figure 1.23 Scenario 3: De-Leaf

In Figure 1.23, the robot is trying to cut the petiole (the end-effector is not shown) to remove this leaf. Because this leaf is currently unwanted or in sick.

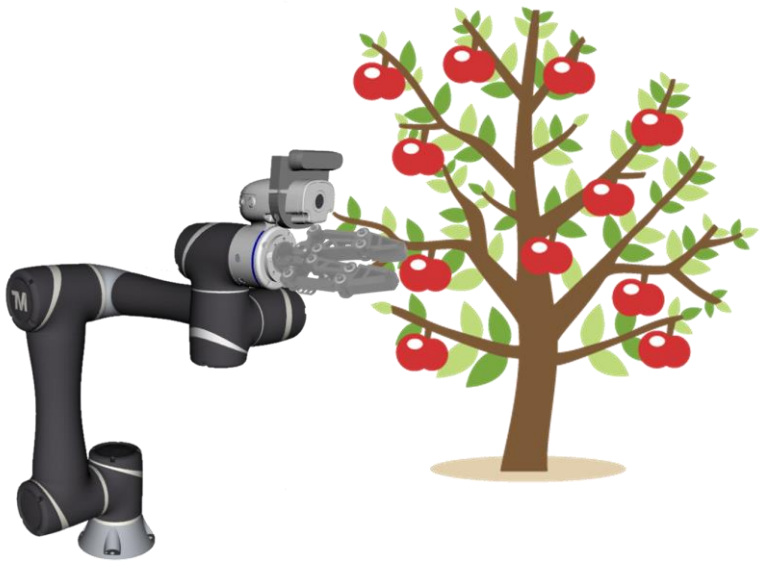


Figure 1.24 Scenario 4: Fruit Harvesting

doi:10.6342/NTU202303769

In Figure 1.24, this system is trying to harvest the fruit in the 3-dimensional space using visual clue and an end-effector.

From the previous mentioned table, monitoring the plant growth condition is essential because the farmer can adjust the cultivating strategies simultaneously. Also, the fruit harvesting and unwanted leaf removal tasks are executed based on sufficient plant geometric information. The leaf defect detection works is contained in the plant leaf removing procedure, because the system needs to evaluate whether cut it after inspecting the target leaf. The simplified task of these three facets is shown in Figure 1.25.

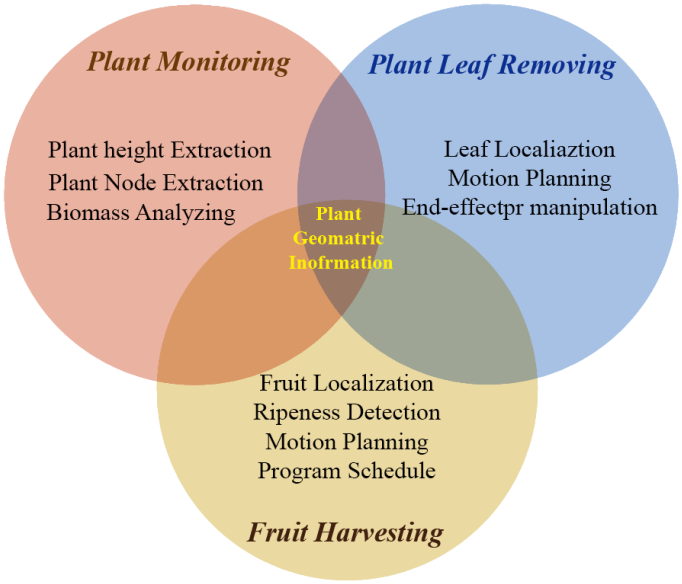


Figure 1.25 The three potential robotic tasks mentioned in Section 1.1

The three potential robotic tasks have a common requirement which is the plant geometric information, which the type of this information cloud is pointcloud. If the plant pointcloud is given, the plant height and petiole number cloud be extracted. Each leaf

doi:10.6342/NTU202303769

held by the petiole also could be segmented. This provides a foundation for the leaf-removing task. Furthermore, the fruits are also segmented when the plant info is given. In this scenario, seldom resources and previous research material could be re-implemented and used. Therefore, the essential function of reconstructing a plant is vital for these high-level applications.

The main research plant is Netted Melon [30 Wikipedia 2022] in this experiment, firstly imported from Japan. Unlike traditional cultivation, all the plants studied in this paper were hydroponically grown. The pH and temperature of the nutrient solution will be precisely controlled at 6.0 and 30 degrees Celsius, respectively. Moreover, we cultivate all melon in greenhouses, whereby the air temperature during the day can rise to nearly 40 degrees Celsius. The purpose of this configuration is that a more considerable temperature difference is conducive to the growth of plants.

Firstly, we transplanted the netted melon into the hydroponically nutrient solution at the seedling stage and used thin wires to assist the plant to grow vertically. Secondly, according to the empirical rule, the best location of fruit is node 10 to 13 because the fruit in this location can absorb nutrients from above and below efficiently and effective. After the 20th to 25th day, the plant will grow to about 160 cm, and the number of nodes will be about 25. After the plants grew to target height, the experimenters cut off the apical meristem (a growth tissue) to stop the growth. At the same time, female and male flowers

doi:10.6342/NTU202303769

will grow as the plant grows in the side vein, and the male flowers will also be pruned regularly to prepare for future artificial pollination. Finally, the fruit has grown and expanded. We use the hook to support the weight of the melon, and this is because we are intervening in the growth, and there is no supporting object around to help support the fruit. Figure 1.26 shows the seeding stage of melon plant and the matured fruit.

Figure 1.26 left: the netted melon, right: the seeding of netted melon.

Figure 1.27 represents the part of our greenhouse where the plant cultivating platform is deployed as a matrix form. The platform contains nutrient solution where the plant root is placed at. The position of each plant is not fixed; for the robotic manipulator, a highly-robustness plant growing center point localization is needed. The plants are cultivated using the “high-wire” mode shown in Figure 1.28, in which the plant grows by climbing the vertical wire. Each cultivating row contains a distance interval the UGV could pass through to it.

doi:10.6342/NTU202303769

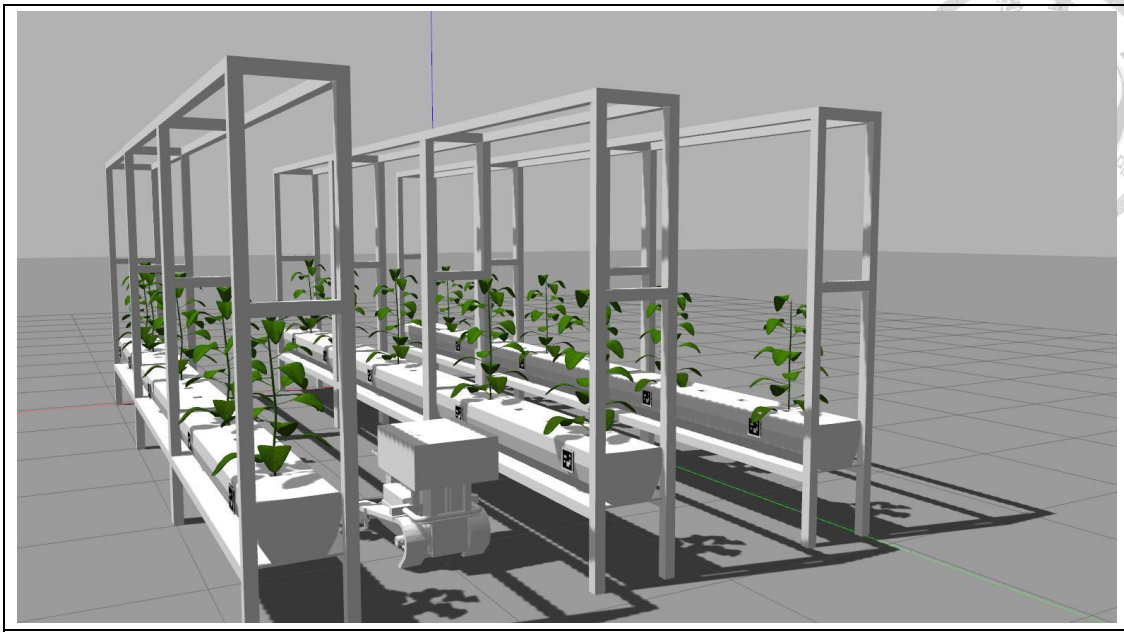


Figure 1.27 Greenhouse Operating Scenario

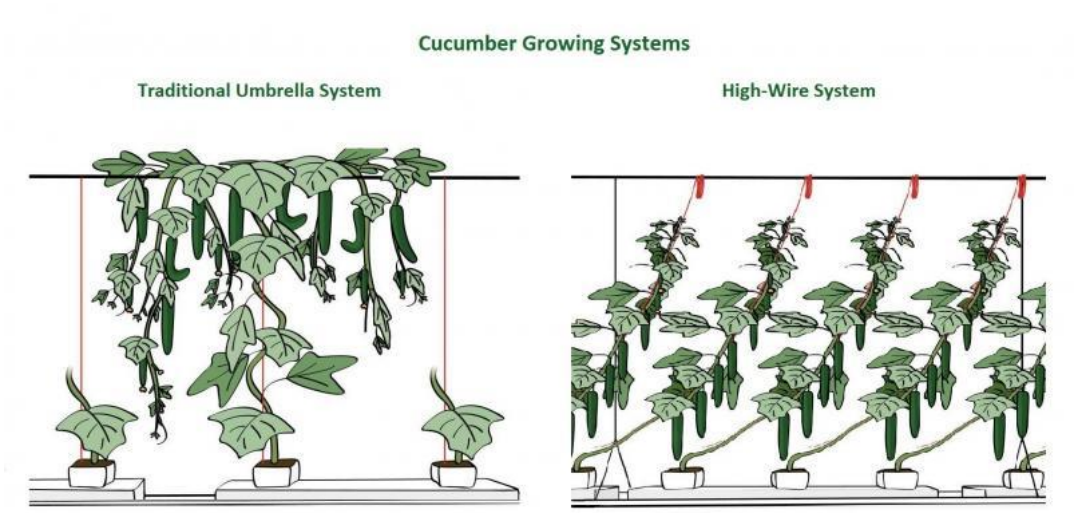


Figure 1.28 Traditional and High-wire system [31]

In the UGV, a robotic manipulator is placed using a connector which is considered as an environment sensor and interacting tool. The termination objective is to move the UGV along different rows of cultivating platform. The system is asked to stop in each plant according the given fiducial marker information as a localization media. The

doi:10.6342/NTU202303769

pointcloud of each plant will be reconstructed using different scale of control strategy, and then the pointcloud will be quantitively post-processed to analyze and segment the different plant organs. Once the plant leaves have been segmented, the further algorithm could analyze the properties of plant leaf, like the area and the health condition. Also, the high-level application like leaf-cutting and even fruit harvesting has a solid foundation. For leaf-cutting task, the petiole position is important because the cutting procedure is to cut the petiole. Similarly, the position and size of fruit cloud be determined for fruit harvesting procedure.

Therefore, under the limited researching background resources, the reconstruction of plant geometric information is the core task, which not only extract the plant growth information and provides a foundation for more sophisticated tasks. According to the given information, a schematic diagram shown in Figure 1.29 describes the entire procedure:

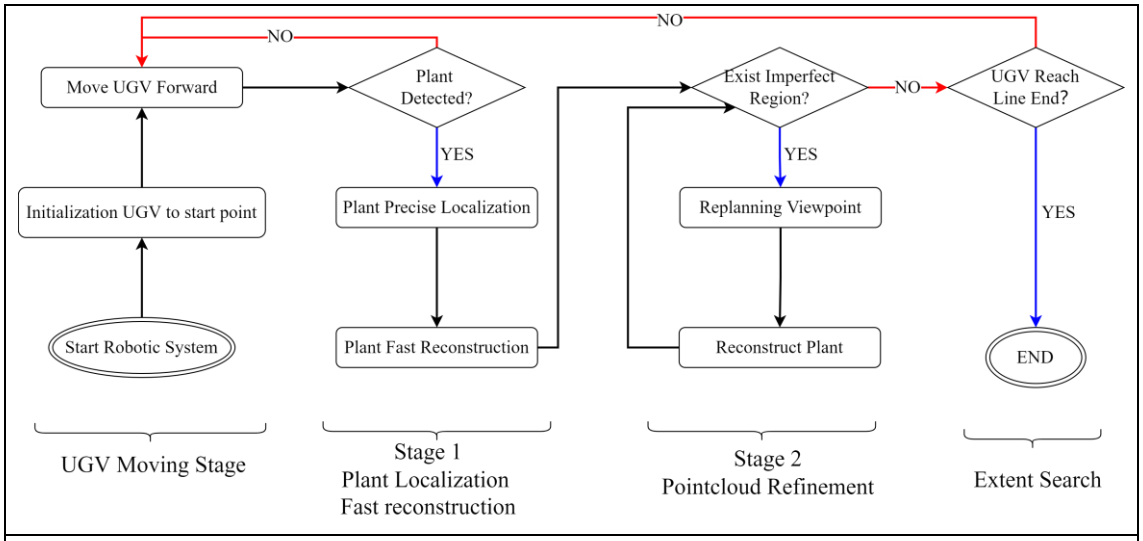
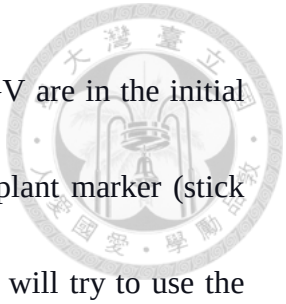


Figure 1.29 A overall solution to reconstruct a plant in 3-Dimensional space

doi:10.6342/NTU202303769

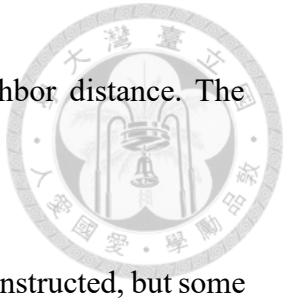


In Figure 1.29, the system firstly initialize itself, which the UGV are in the initial start point. And then, the UGV starts moving and trying to detect plant marker (stick below/beside). If the plant is found, the robotic manipulator system will try to use the given information to precisely locate the observed/target plant. After that, the robot plans several viewpoints to roughly reconstruct the target plant. The plant height will be extracted at this time. The current reconstructed pointcloud may be not perfect enough to execute further analyzing, like organ segmentation. The system then finds the imperfect region to re-plan several viewpoints to refine the pointcloud. If the quality of pointcloud is acceptable, the robotic system tells the UGV to move forward for next plant. Or, if the UGV is already in the end-of-line, the procedure stops here.

1.3 Contribution

This paper proposes a multi-stage procedure for reconstructing the plant point cloud. At the outset, the plant's location in 3D space is unknown. Therefore, a marker-based plant localization method is implemented. The depth camera generates the environment point cloud containing the observed plant and complex background. A marker-based HSV plant point cloud filter is proposed and implemented. The first stage of the multi-stage procedure involves locating the plant in space and attempting to quickly reconstruct the observed plant using a pre-defined viewpoint under plant coordination. In this step, point

doi:10.6342/NTU202303769



cloud alignment is also implemented using dynamic adjacent neighbor distance. The plant's height is calculated using the reconstructed point cloud.

In the second stage, the plant point cloud is assumed to be well-constructed, but some regions may be occluded. The paper also proposes a stem occlusion checking module based on the stem extraction algorithm to provide the missing stem region. The system then plans a series of viewpoints and scores these viewpoints using an occlusion-free viewpoint optimizer to select the best viewpoint for this missing region.

The procedure then segments the entire point cloud into stems, leaves, and petioles using a re-implemented segmentation method.

1.4 Organization of this thesis

In the Chapter 2, the survey of the agricultural history, application, and phonotype system has been delicately discussed. In the Chapter 3, the related basic principle and algorithm has been introduced including pinhole camera model, method of hand-eye calibration, K-D tree, ICP and PCA. In the Chapter 4, the system architecture, coordination system and related controller design is discussed.

The Chapter 5 contains all the experimental-oriented algorithm used in this paper including plant localization and filter, implementation of iterative closest point using dynamic adjacent distance, the stem extraction algorithm, the stem-based occlusion checking algorithm, the cylinder viewpoint set generation algorithm, the occluding-free

doi:10.6342/NTU202303769

viewpoint optimizer, and plant pointcloud segmentation algorithm. These algorithms are the basis of the following proposed procedure. In Chapter 6, a multi-stage plant reconstructs and analyzing procedure are proposed including the plant fast reconstruction, plant height extraction in the first stage, and stem extraction, stem-based missing region re-planning and plant segmentation in the second stage.

In the Chapter 7, the experiment has been conducted. The experimental setup and some preparation works are introduced. The algorithm verification stage in laboratory and real-world experiment (*) is executed.

In the Chapter 8, the conclusion and future work are listed.

doi:10.6342/NTU202303769

Chapter 2
Background and Literature Survey

In this chapter, the background and literature survey of the autonomous agricultural robotic system is investigated. The background of agrarian robotics and operating platforms are introduced in Section 2.2.1 and 2.1.2, respectively, including the different types of applications. The plant protection robotics development history review is presented in Section 2.2, containing several facets to be inspected and measured. Finally, the robotic vision system exclusively applied in the agricultural field has been discussed, including the types of sensors and image processing algorithms.

2.1 Agricultural Robotic Development History

Over the past fifty years, scientists and researchers have conducted extensive research in agricultural robotic automation. The type can be divided into **weeding**, **sowing**, **irrigation**, **spraying**, **fertilisation**, **artificial pollination**, **plant protection** and **harvesting robotics**.

Weeds shares illumination, nutrient and water with other crops, and the crop yield will be significantly impacted by weed without control—Korean team [32 Choi., et al. 2015] designed a screw-type wheel to remove the weed in the rice row, using a single

doi:10.6342/NTU202303769

camera with an infrared filter to track the rice row in morphology way. McCool and his team [33 McCool, et al. 2017] developed a lightweight deep neural network to reduce weed detection efficiency based on Inception-v3 architecture. The laser-based weeding techniques have also been applied to Echinochloacrus-Galli, Amaranthus retroflexus, targeting meristem, establishing the weed damage model. The result shows the treated plant leaf under computed dose of laser is no longer alive [34 Marx, et al. 2012]. A non-segmentation, traditional computer vision and feature detection method using contour or shape descriptor has been developed by [35 Haug, et al. 2014]. In the Netherlands, [36 Van Evert, et al. 2011] developed an integrated robotic system with a diesel engine for removing "Broad-leaved dock", and the robot has a specific cutting blade to destroy the weed by using Fast Fourier Transform to successfully detect the morphological position of the weed and a Real-Time Kinematic GPS to follow the predefined trajectory.

High precision **sowing** is based on the pre-computed path and confidence of motion control of the vehicle. A platform with these properties will accomplish tasks more than sowings, such as irrigation, fertilization and spraying. The accuracy of the path is the critical challenge of this task. An attempt has been conducted to design and fabricate a seedling robotic for corn and soya bean [37 Kumar, et al. 2021]. Hasson and him team [38 Hassan, et al. 2016] also developed a ground vehicle platform with a single seed selector and a planter to inject seed into the soil. Moreover, the [39 Katupitiya, et al. 2007]

doi:10.6342/NTU202303769

devised a force-controlled and self-propelled platform using dual differential GPS and IMU to put seeds into the soil, and no operator manoeuvres this system because of moduled part, adaptive control path following algorithms.

Irrigation is usually conducted several times by farmers in a single day, and this will cause nutrient loss and waste of water without precise control. David [40 David, et al. 2012] designed an irrigation control approach used in a conservatory. They monitor the soil humidity percentage and autonomous turn on the water pump to water the soil. This saves 75% water consumption and increases 50% plant growing speed.

Spraying is another essential application in agriculture. For example, a coconut palm tree will reach 60 to 100 feet [41 Better Home Gardens 2022], a farmer usually working at risk of spraying pesticide. Faical [42 Faiçal, et al. 2014] deployed multiple ground sensors for UAV navigation and feedback on the pesticide concentration of chemicals. All sensor nodes are distributed computed communication transponders w.r.t the UAV. The result shows that the system can endure the wind changes and adaptively control the trajectory. Jiandong MAO [43 MAO, et al. 2020] also implemented an uncrewed ground vehicle to spray and bury fertilizer into the soil within a specific depth. The robotic system could be deployed in an arid region, self-navigating using field image processing algorithms.

doi:10.6342/NTU202303769

In addition, Nicholas Ohi [44 Ohi, et al. 2018] introduced a bionic end-effector from Mason Bee, equipped a 3D LiDar for local map generation using SLAM. The specific end-effect is a flexible material. The flexible material brushes from the anthers（male reproductive organ) to the pistils (female organ) . The classification of flowers uses the Inception-v3 [45 Szegedy, et al. 2016] model powered by Google. Finally, the robot explored the whole greenhouse and discretized the environment to a Voronoi map. The motion planning is provided by Open Motion Planning Library and Flexible Collision Library.

Plant protection is crucial to maintain a stable quantity and quality of crop or fruit in preciseness agriculture. There are several aspects such as plant leaves **disease detection**, **leaf pruning**, **phenotype inspection** and **measurement**. Gavhale [46 Gavhale, et al. 2014] investigated bacterial, viral, fungal disease symptoms and tried to summarize the affected region's quantity, boundaries, and colour in leaves; several methodologies were introduced, detailly including image pre-processing, feature extraction and classifier. The greenhouse-cultivated tomatoes disease has also been discussed by [47 Durmus, et al. 2017]. The robot wanders in a greenhouse with the mounted camera on the flange. The neural network architecture AlexNet and SqueezeNet have been studied, and both models reached about 95% true positive(TP) rate. Similarly, Selvaraj [48 Selvaraj, et al. 2013] pre-processed using HSV filter and texture feature and

doi:10.6342/NTU202303769

feed data into an SVM classifier to distinguish bacterial, sunburn, early or late scorch and fungal diseases. As the plant grows, some leaves under the greenhouse are shaded by other leaves, making the plant more susceptible to disease infection, so for cultivated cucumbers, it is necessary to prune the leaves below after the maturity period. Based on this premise, Van Henten [49 Van Henten, et al. 2006] used near-infrared light to identify the stem and designed a set of heat-based cutting end effectors. Similarly, as plant leaves mature, their photosynthetic efficiency declines, and some leaves are cut off to maintain the overall fruit nutrient supply.

In the field of plant protection, plant **phenotyping** is a crucial area. In order to intuitively understand the plant growth state and make subsequent adjustments, it is necessary to establish a high-throughput field phenotyping (HTTP [50 Baidu_HK 2022]). Generally, an HTTP platform includes a visible light camera, near-infrared camera (NIR), infrared camera(IR), hyperspectral(HS), fluorescence camera and depth camera to detect plant growth height, width, leaf area, angle, colour and plant disease information. The phenotype can be divided into morphological and physiological [51 Yandun Narvaez, et al. 2017]. The morphological feature contains leaf area, foliage density, stem shape and size, and plant height, while the physiological feature includes nutrient content, water stress, biomass, fruit maturity, disease, and plagues symptoms.

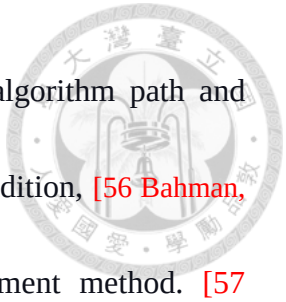
doi:10.6342/NTU202303769

A phenotyping platform containing 735 individual roses was developed [52 Tisné, et al. 2013]. This platform can rotate each studied plant to record and control its weight, soil moisture content, leaf growth process, watering and control of water-deficient plants and give the nutrient solution. In a review paper [53 Chen, et al. 2015], an Australian ACPFG Scanalyzer 3D platform has been introduced, deploying four fully controlled climates with conveying belts to transform plants. It is also analyses stem, leaf size and colour using a visual camera, leaf and soil water content using IR camera, chlorophyll and green fluorescent protein (GFP) fluorescence using UVs.

Similarly, Virlet [54 Virlet, et al. 2017] developed a vast gantry crane with a TIR camera, NDVI, visual camera and 3D laser scanner mounted on the top to analyze canopy height and wheat ears using reconstructed pointcloud and RGB image. Shafiekhani [55 Shafiekhani, et al. 2017] used a ground-based autonomous vehicle and mobile observation towers to collect individual plant data and comprehensive field information, respectively. The top and bottom binocular cameras were used to reconstruct the plant model, and then the plant height, leaf area index and photosynthetically active radiation were measured. The results showed that the height means square error (MSE) was 2.36 cm.

In addition to this, the use of aluminium blocks to calibrate RGB and thermal cameras is described in detail. [Chen-Ming Wang et al. 2021] also implemented a three

doi:10.6342/NTU202303769



robotic manipulator (UR-5) system, generating an RRT-Connect algorithm path and optimizing the next viewpoint using a deep learning algorithm. In addition, [56 Bahman, et al. 2019] proposed a point cloud-based plant height measurement method. [57 Constantino, et al. 2018] proposed a height measurement method based on pixel size, and [58 Wang and Chen 2020] also constructed an iterative closest point (ICP) based aligning method and a rotational platform. The point cloud is used to reconstruct the 3D model of the plant and obtain the height and leaf size.

In addition, scientists have invested much research on fruit harvesting robots in the past 30 years. Sa [59 Sa, et al. 2017] proposed an SVM algorithm based on point cloud FPFH to find the position of the peduncle. Hemming [60 Hemming, et al. 2014] used a 7-axis robotic arm and a linear guide to perform bell pepper harvesting using pneumatic grippers. [61 Mehta, et al. 2014] Citrus picking using Bayesian classification and depth estimation. [62 Han, et al. 2012] Blueberry picking using fixed threshold and HSV filter. Figure 2.1 shows the compilation of paper survey.

doi:10.6342/NTU202303769

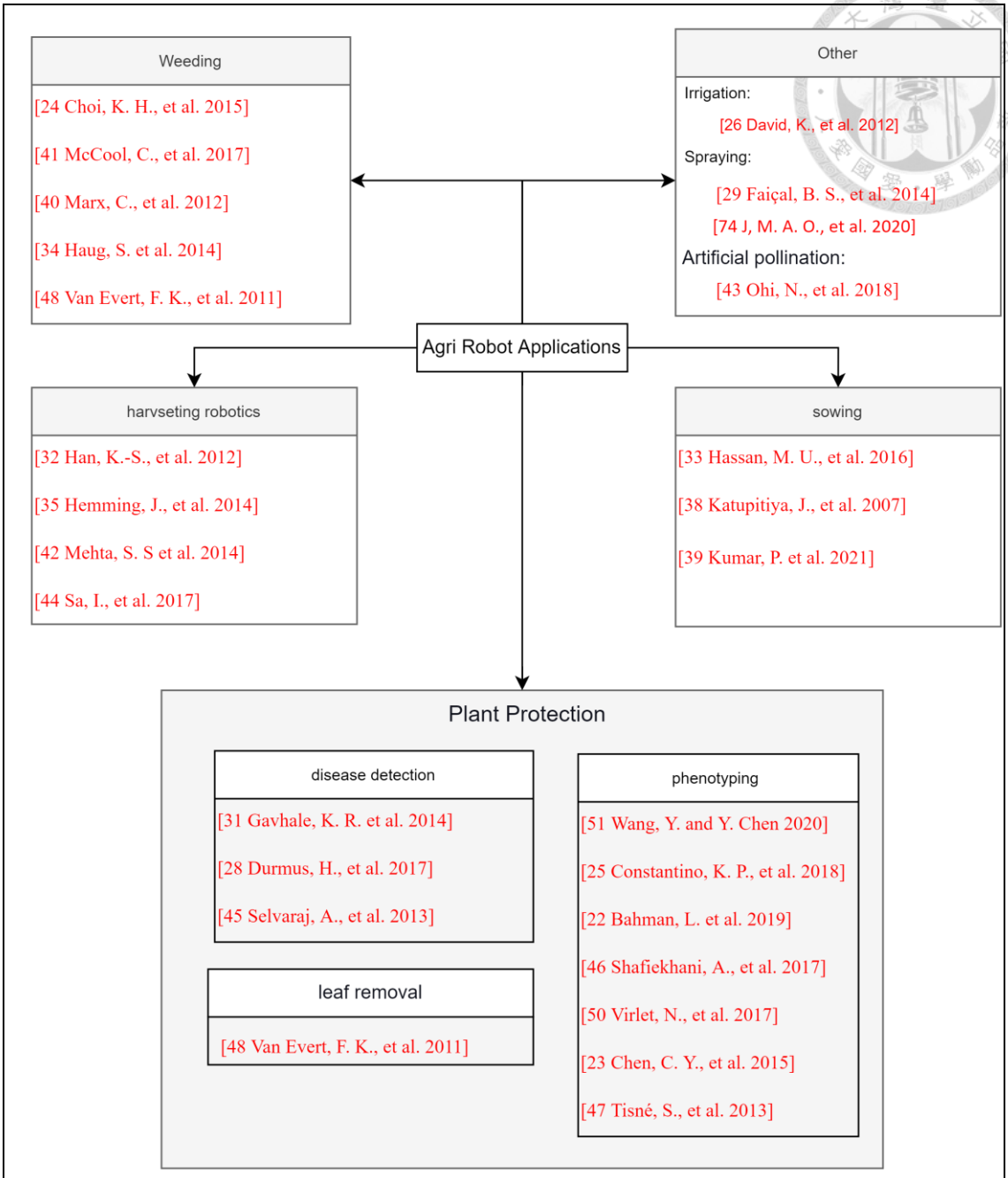


Figure 2.1 The Agricultural robotic background and literature survey

2.2 Agricultural Robotic System Platform

In most papers, agricultural robots are designed to work in greenhouses with manual intervention, such as the strawberry picking robot in [63 Xiong, et al. 2020], and the indoor grape picking in [64 Luo, et al. 2018]. Technology, for plant phenotyping, most

doi:10.6342/NTU202303769

studies are performed indoors [52 Tisné, et al. 2013] and [53 Chen., et al. 2015]. However, a small number of measurements are still performed outdoors, such as [65 Andrade-Sanchez, et al. 2014], [66 Barker, et al. 2016], and [55 Shafiekhani, et al. 2017].

Meanwhile, the plants analyzed and detected can be divided into single and whole [52 Tisné, et al. 2013] . Most outdoor robots use tractors for the equipment carrying platform, and [65 Andrade-Sanchez, et al. 2014] use human-driven tractors, and [66 Barker, et al. 2016] introduced a UGV that can plan a path, [67 Busemeyer, et al. 2013] also proposed a four-wheeled vehicle with a higher chassis, changing any wheel direction to adapt to different fields. Virlet [54 Virlet, et al. 2017] uses a Gantry crane to analyze plant conditions. However, the use of high-weight implements will cause soil compaction [68 Hamza, et al. 2005], so some lightweight platforms have also been born [52 Tisné, et al. 2013] proposed a small UGV-based robotic arm car [69 Wu, et al. 2019] developed an indoor three robotic manipulator system to inspect plant phenotype. In addition to ground vehicles, UAVs can maintain a better degree of flexibility in this task [70 Sugiura, et al. 2005] Using UAVs for Crop Monitoring and Mapping, Göktoğan [71 Göktoğan, et al. 2010] use UAVs to identify and manage weed in the aquatic environment. However, the endurance and carrying capacity of UAVs is greatly limited, and the flight area of UAVs is strictly controlled, so we propose a crawler UGV platform based on robotic arms, which integrates lightweight, heavy load and endurance UGV platform.

doi:10.6342/NTU202303769

2.3 Robotic Vision in plant phenotyping

In traditional operations, the measurement of plant phenotypes is often measured manually, which consumes much time and cannot standardize the accuracy of the measurement [55 Shafiekhani, et al. 2017]. The use of computers can significantly improve the speed and efficiency of inspections. The detection of phenotypes is similar to general agricultural robot vision, and this subsection will discuss sensors applied in robot vision, detection orientation, machine vision algorithms, and current machine vision challenges. This section will discuss different aspects of the robotic vision system in the following order: sensors, visual entry point, machine vision algorithms and challenges.

2.3.1 Sensor Types

Sensors for robot vision can be divided into monocular monochrome, colour, calibrated binocular vision cameras, depth cameras, (near) infrared, hyperspectral(HS), and LiDars.

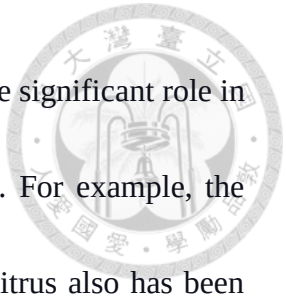
Generally speaking, monocular cameras have no depth information, small data, easy processing, and low cost. The binocular vision camera can synthesize depth information under the condition of consistent left and right camera intrinsic and extrinsic parameters and generate spatial point cloud [72 De-An, et al. 2011]. Relatively, the price ratio of brand binocular cameras is more expensive. Further, [73 Kang, et al. 2008] introduced the task of

doi:10.6342/NTU202303769

fruit picking using more than two cameras. In [55 Shafiekhani, et al. 2017], the authors use a BumbleBee XB3 trinocular camera to photograph plants to reconstruct plant models. In recent years, with the development of technology, small and lightweight depth cameras have come out, such as Intel's Realsense and Microsoft's Kinect series, where Kinect V2 uses ToF technology, and Intel D435 uses structured light technology to obtain the position of objects in space, for example, in In the paper [58 Wang and Chen 2020], the visual sensor used is Kinect-V2. At the same time, Vit and Shani [74 Vit and Shani 2018] compare depth cameras on the market in detail, including D435, Kinect v2, Orbbec Astra s and Intel sr300 in agricultural phenotyping. The results show that Intel Real Sense D435 @ 848 resolution is better than other RGBD cameras detecting corn stem width, object size estimation and object recognition.

The absorption rates of different wavelengths of light are different due to the different moisture content of leaves and stems. In the literature [49 Van Henten, et al. 2006], cucumber fruit and stem and leaves show completely different absorption rates at 850nm and 970nm, distinguishing target and noise. The difference between multiple spectral cameras and the hyperspectral camera is that the HS camera provides a smaller sensing wavelength gap, and the experimenter could analyze different components such as chlorophyll and other nutrients.

doi:10.6342/NTU202303769



Although IR cameras and hyperspectral cameras will play a more significant role in the fieldwork, the cost is higher than any other type of equipment. For example, the hyperspectral camera has been used in satellite and terrain. Green citrus also has been detected by a hyperspectral camera with 751 nm, 682 nm and 548 nm [75 Okamoto. and Lee 2009].

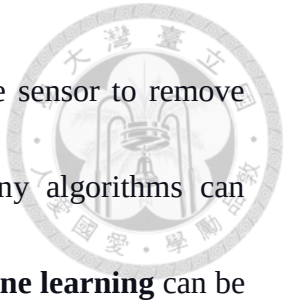
The Lidar is commonly used in the autonomous vehicle, serving in the agricultural environment. LiDar is a lossless technique to collect distance, including ToF LiDar and phase-shift LiDar. In Paper [76 Allouis, et al. 2013], LiDar is used to construct a giant pine tree and estimate its volume and biomass. [77 Wei, et al. 2012] designed and validated a multispectral LiDar and verified its performance.

2.3.2 Computer Vision Algorithms

Whatever the sensors are, the primary data type is the image. By referencing the review paper [78 Kapach, et al. 2012], we can divide visual entry points from colour, texture and shape. The colour is the basic information from an image. By analyzing the colour of all pixels, multiple peaks will exist representing different colours, like canopy or fruit. However, if the target's colour is the same as that of other regions, only applying the colour parameter is not enough [79 Hannan, et al. 2007].

Plant temperature is an indicator of plant water availability [80 Baluja, et al. 2012], the usage of thermal data of plants gives clues to irrigation control. In observing plant

doi:10.6342/NTU202303769



phenotypes, we may need to process the raw data generated by the sensor to remove unwanted information and highlight the subject observation. Many algorithms can generate similar results. Image **segmentation**, **clustering**, and **machine learning** can be performed. There are several methods in **segmentation** algorithms—the [9 Xiong et al. 2020] developed a strawberry harvesting robotic system using an adaptive threshold algorithm to change HSV filter parameters dynamically when illumination condition varies. Zhao and his team [81 Zhao, et al. 2005] use the edge detection fruit in an apple tree. [72 De-An, et al. 2011] also developed an apple harvesting robotic that uses double threshold and region growing algorithms to find the location of the apple.

Clustering is an unsupervised learning technique; the background and foreground(target) can be distinguished when multiple features are used. Wachs[82 Wachs, et al. 2010] developed a clustering algorithm using the LAB colour representation method. The result shows that apple, leaf and background can be clustered using the linear programming method, while the result also indicates that no evident clear division was found.

When a problem is difficult to describe with a mathematical model, machine learning may be another way to get results. For example, the peduncle is a stem-like region that connects the side vein and fruit, detection of the peduncle informs the robotic system

doi:10.6342/NTU202303769

where is the cut point. Vit and Shani [74 Vit and Shani 2018] shows that the Mask RCNN can be trained to generate a mask that contains a maze stem. The result can be further used to calculate maze stem width.

Although machine learning can achieve satisfactory results, the model's output is not explanatory, and the correctness of the results has not yet been proved mathematically by humans, but it is sufficient in terms of use. The robotic vision in plant phenotyping techniques has been listed in Figure 2.2.

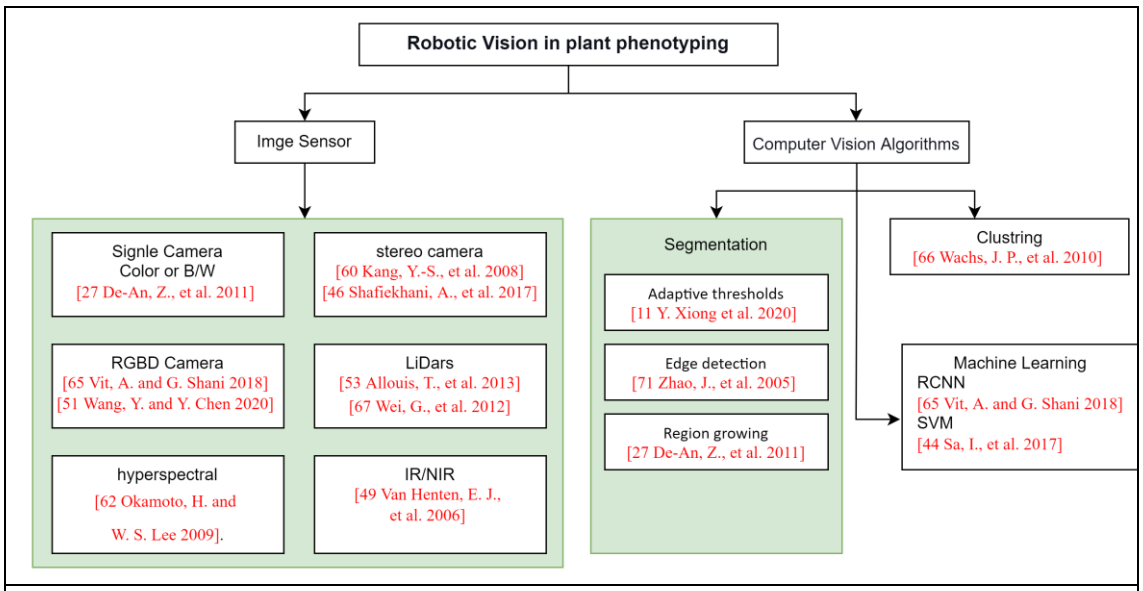


Figure 2.2 Classification of Robotic Vision in plant phenotyping

doi:10.6342/NTU202303769

Chapter 3
Related Algorithms

The third chapter will introduce some algorithms and theoretical basis used in this paper. This paper sets out to perform agricultural applications using robotic arms and depth imaging, and all experiments are based on the pinhole camera model explained in Section 3.1. Secondly, algorithms for hand-eye calibration will be discussed in Section 3.2, because the relationship between camera and robotic end-effector is unknown. In Section 3. the K-D Tree, a point cloud storage datatype, and nearest search algorithm is introduced. In Section 3.4, a registration method called iterative closest point purposed by [83 Besl, et.al 1992] will be briefly introduced, that sensor error will be cancelled. Finally, The principal Component analysis which compute the point cloud distribution direction is introduced, in Section 3.5. These algorithms are directly used without modification. Some modified or invoked algorithm will be described in the Chapter 5.

3.1 Pinhole Camera Model

The pinhole camera model is the most basic and fundamental camera representation model, without considering the aperture and focal length changes. The model defines the relationship between the object in the 3D space and the 2D image captured by electronic devices, like CMOS. A standard image capturing device consist of the object, an infinitely

doi:10.6342/NTU202303769

small pinhole, sensor plane and inverted image plane. Figure 3.1 shows the simplified pinhole model, the object under world frame W spread by projection rays through the pinhole O, the projection ray will be precepted by CMOS in sensor frame S. For computation simplicity, the inverted image frame I has been investigated in following explanation. According to this setting up, assuming that the camera rigid body center is the pinhole O, the homogeneous transformation matrix of the world frame with respect to (w.r.t.) camera rigid body center defined as

$${}^{O}_{W}\mathrm{T} = \begin{bmatrix} {}^{O}_{W}\mathrm{R} & {}^{O}_{W}\mathrm{t} \\ \mathbf{0} & 1 \end{bmatrix}$$	Equation 3.1

Where the ${}^{O}_{W}\mathrm{R}$ is a 3 by 3 rotational matrix and ${}^{O}_{W}\mathrm{t}$ is 1 by 3 translational matrix. The $\mathbf{u}$ and $\mathbf{v}$ represent the axis of inverted image frame, the origin of the inverted image frame is starting from $\mathbf{u}$ and $\mathbf{v}$. The u and v represent the projection ray projecting on the inverted image frame, which is the objective of pinhole model. The intersection between the line along Z axis and inverted image frame is called principal point.

doi:10.6342/NTU202303769

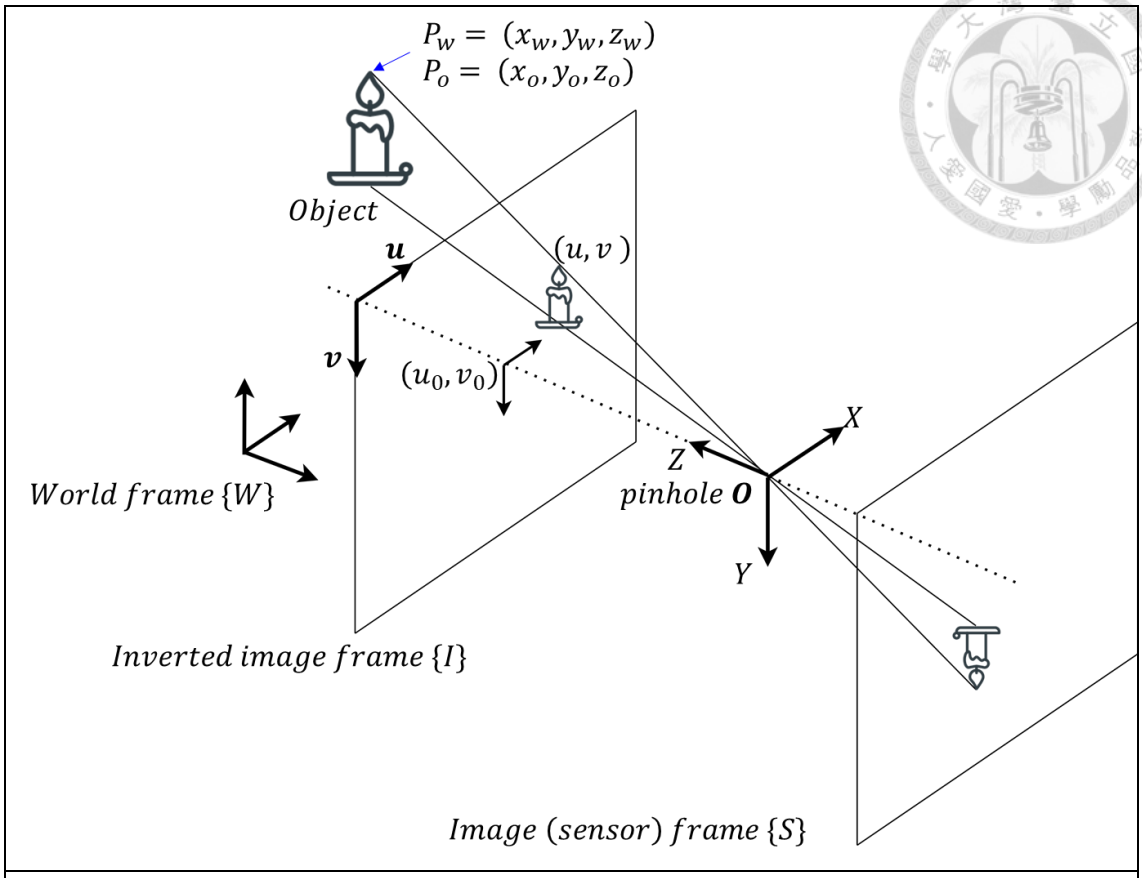


Figure 3.1 Pinhole camera model

Assuming that the model only interest about the top flare of the candle, where the same point under world frame and pinhole frame could be defined as $P_w = (x_w, y_w, z_w)$ and $P_o = (x_o, y_o, z_o)$, respectively. Merging Equation 3.1 and point definition, the relationship between point in pinhole frame and world frame has been deduced shown below:

$$\begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix} = \begin{bmatrix} {}^{O}_{W}\mathrm{R} & {}^{O}_{W}\mathrm{t} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} = {}^{O}_{W}T \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \quad \text{Equation 3.2}$$

doi:10.6342/NTU202303769

The reason why adds the 1 in the fourth element is to satisfy homogeneous matrix computation requirement.

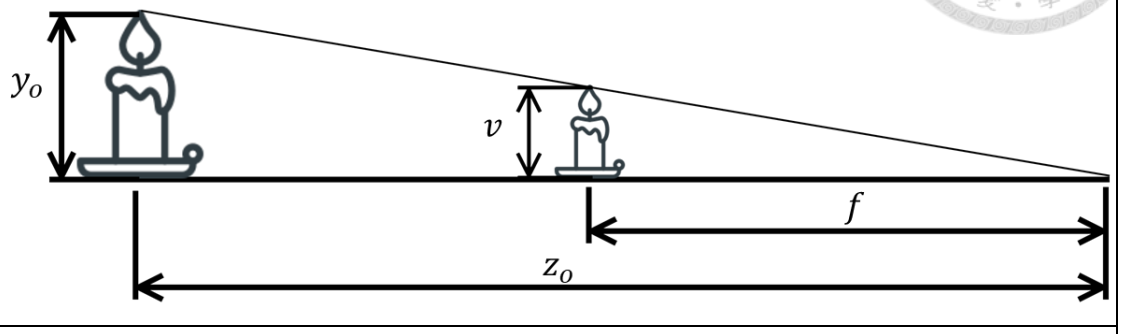


Figure 3.2 Side View of Camera Model

For an ideal sensor and pinhole, the focal length is a constant f. However, due to the imperfectness of image sensor, image anamorphic format, calibration error, the focal length of x axis and y axis can differ, where the pixel under different focal length is not a square. According to the perspective principle, the ratio of y_0 and v, x_0 and u, and z_0 and f are same shown in Figure 4. The u and v can be defined as follows:

$u = x_o * \frac{f_x}{z_o} + u_0$	Equation 3.3

$v = y_o * \frac{f_y}{z_o} + v_0$	Equation 3.4

doi:10.6342/NTU202303769

The intrinsic matrix which defines the camera internal physical properties has been introduced by [84 Hartley et al. 2003] shown below:

$$\mathrm{K} = \begin{bmatrix} f_x & 0^* & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \text{Equation 3.5}$$

Where, the f_x and f_y are focal length along x and y axis, the u_0 and v_0 are principal points, 0^* has been set as zero for simplicity instead of skew factor.

The position in pixel coordination will also be deduced using Equation 3.2 to 3.5, showing below:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0^* & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} [{}^{O}_{W}\mathrm{R} \quad {}^{O}_{W}\mathrm{t}] \begin{bmatrix} x_o \\ y_o \\ z_o \end{bmatrix} = K\, {}^{O}_{W}T \begin{bmatrix} x_o \\ y_o \\ z_o \end{bmatrix} \qquad \text{Equation 3.6}$$

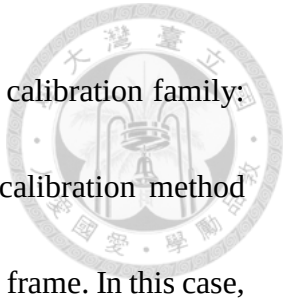
$$(4 * 1) = (3 * 3) * (3 * 4) * (3 * 1)$$

The dimension of first, second and third matrix are 3 by 3, 3 by 4 and 3 by 1, satisfying the matrix multiplication principle.

3.2 Hand-Eye Calibration

Whether it is a high-precision industrial robot arm or a collaborative robot arm with compliance control, as long as machine vision is equipped, the relative relationship between the robot arm coordinate system and the camera coordinate system needs to be clearly defined.

doi:10.6342/NTU202303769



Two approaches have been widely investigated in the hand-eye calibration family: eye-in-hand (EiH) and eye-to-hand (EtH). The eye-in-hand (EiH) calibration method finds the relationship between the robotic flange frame and the camera frame. In this case, the camera moves with the robotic arm's movement and is usually used to detect the object ahead to the end-effector. However, the eye-to-hand (EtH) method finds the relationship between the robotic base (world frame) to the camera frame. In this camera setting configuration, the robotic manipulator and the considerable size of workspace can be perceived from the outside to provide a comprehensive viewpoint.

In this paper, the depth camera is deployed on the last axis of the robotic manipulator, which is the flange surface. So, the configuration of setting up the EiH is the basis of the experiments in this paper. This section will focus on the theoretical basis of calibration of EiH. In Figure 3.3, there are four coordination frames, which are world frame $\{World\}$, flange frame $\{Flange\}$, camera frame $\{Camera\}$ and fiducial frame $\{Fiducial\}$.

doi:10.6342/NTU202303769

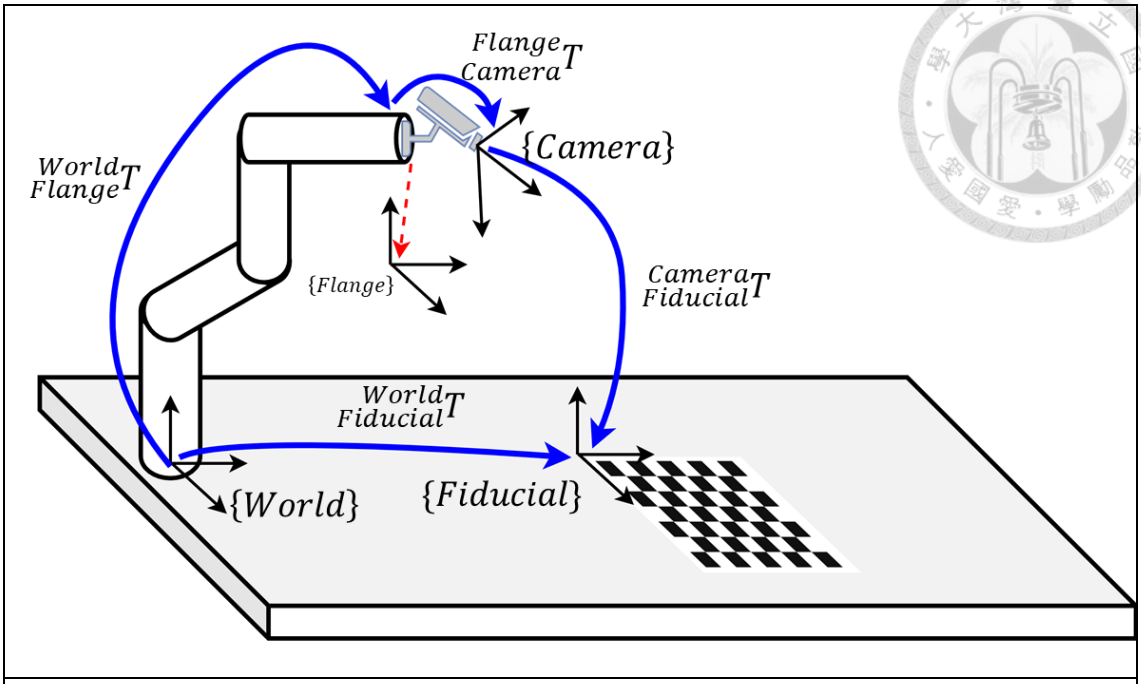


Figure 3.3 Eye-in-hand Calibration Configuration

There definition of transformation matrix shown in Table 3.1:

Table 3.1 Definition of transformation matrix in hand-eye calibration algorithm.

Transformations	Definition
$^{World}_{Flange}T$	The transformation matrix of Flange frame w.r.t World frame
$^{Flange}_{Camera}T$	The transformation matrix of Camera frame w.r.t Flange frame
$^{Camera}_{Fiducial}T$	The transformation matrix of Fiducial frame w.r.t Camera frame
$^{World}_{Fiducial}T$	The transformation matrix of Fiducial frame w.r.t World frame

In Figure 3.3, the four blue line represent the transformations. The $^{World}_{Flange}T$ can be obtained by using robotic forward kinematics, where the Denavit-Hartenberg table generates the robotic model, and the joint state (angle or displacement) is already known. The computer vision libraries like OpenCV or Halcon provide library to detect chessboard or Aruco-based chessboard (generally defined as fiducial), which returns the position and

doi:10.6342/NTU202303769

orientation of the fiducial, that is, ${}^{Camera}_{Fiducial}T$, also called extrinsic matrix of the camera. The value of transformation matrix ${}^{World}_{Fiducial}T$ is a don't care element because it can be eliminated when multipair movement is executed. For a general case, the ${}^{World}_{Fiducial}T$ can be rewritten as the combination of other transformation matrixes shows as follows:

${}^{World}_{Fiducial}T = {}^{World}_{Flange}T * {}^{Flange}_{Camera}T * {}^{Camera}_{Fiducial}T$	Equation 3.7

Considering the arbitrary two case according to Equation 3.7 shows below:

${}^{World}_{Fiducial}T = {}^{World}_{Flange_1}T * {}^{Flange_1}_{Camera_1}T * {}^{Camera_1}_{Fiducial}T$	Equation 3.8

${}^{World}_{Fiducial}T = {}^{World}_{Flange_2}T * {}^{Flange_2}_{Camera_2}T * {}^{Camera_2}_{Fiducial}T$	Equation 3.9

In the Equation 3.8 and 3.9, the term ${}^{World}_{Fiducial}T$ is fixed, combining two formula, the Equation 3.10 can be obtained:

${}^{World}_{Flange_2}T * {}^{Flange_2}_{Camera_2}T * {}^{Camera_2}_{Fiducial}T = {}^{World}_{Flange_1}T * {}^{Flange_1}_{Camera_1}T * {}^{Camera_1}_{Fiducial}T$	Equation 3.10

And then, by simply moving the matrix, the Equation 3.11 has been defined:

${}^{World}_{Flange_2}T * {}^{World}_{Flange_1}T^{-1} * {}^{Flange_2}_{Camera_2}T = {}^{Flange_1}_{Camera_1}T * {}^{Camera_2}_{Fiducial}T^{-1} * {}^{Camera_1}_{Fiducial}T$	Equation 3.11

doi:10.6342/NTU202303769

The Table 3.2 defines the AX = XB problem.

Table 3.2 AX = XB definition	
Notation	Formula
A	$A_2A_1^{-1} = {}^{World}_{Flange_2}T * {}^{World}_{Flange_1}T^{-1}$
B	$B_2^{-1}B_1 = {}^{Camera_2}_{Fiducial}T^{-1} * {}^{Camera_1}_{Fiducial}T$
X	${}^{Flange_1}_{Camera_1}T$ or ${}^{Flange_2}_{Camera_2}T$

The Equation 3.11 can be simplified as the problem of

$AX = XB$	Equation 3.12

In Equation 3.12, the A, B is given and shown in Table 3-2. The term A is also called extrinsic parameter of camera.

Since the 1980s, people have carried out extensive and in-depth research on "hand-eye calibration", among which Tasi [85 Tsai et al. 1989] proposed a widely used algorithm, but this algorithm is not effective for lens nonlinear distortion, Daniilidis [86 Daniilidis 1999] uses the double quaternion and singular value decomposition algorithm to obtain the hand-eye matrix. Next, this paper will give a mathematical explanation of the hand-eye calibration based on Lie Algebras. The Equation 3.12 can be expanded into matrix form as follows:

$\begin{bmatrix} R_A & t_A \\ \mathbf{0} & 1 \end{bmatrix}\begin{bmatrix} R_X & t_X \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} R_X & t_X \\ \mathbf{0} & 1 \end{bmatrix}\begin{bmatrix} R_B & t_B \\ \mathbf{0} & 1 \end{bmatrix}$	Equation 3.13

doi:10.6342/NTU202303769

After multiplying the matrices, the rotational part and translational prat has been divided, showing below in Equation 3.14 and 3.15, respectively:

$R_A R_x = R_x R_B$	Equation 3.14

$R_A t_X + t_A = R_A t_B + t_X$	Equation 3.15

In the first part of derivation, the rotational will be handled. The all element in Equation 3.14 are SO (3) and by moving element R_A to right hand side and apply logarithmic mapping, the Equation 3.15 has been rewritten as follows:

$\log(R_A) = \log(R_x R_B R_X^T)$	Equation 3.16

Define $\log(R_A) = [\alpha]\ and\ \log(R_B) = [\beta]$, the Equation 3.16 can be also rewritten as follows:

$\alpha = X\beta$	Equation 3.17

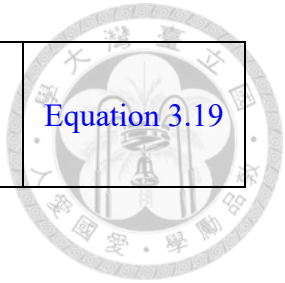
When we considering multiple set of equations, the problem is also called least square fitting problem, which defined as follows:

$\min \sum_{i=1}^{k} \|R_X \beta_i - \alpha_i\|^2$	Equation 3.18

The solution of Equation 3.18 is absolute orientation problem, which can be solved using following method:

doi:10.6342/NTU202303769

$\mathrm{R_X} = (M^T M)^{-0.5} M^T, where\ M = \sum_{i=1}^{k} \beta_i \alpha_i^T$	Equation 3.19



However, the requirement of M in Equation 3.19 is not a singular. When only two set of motion has been inputted, this method cannot solve this problem.

The translational part in this paper is same as [TSAI]'s method, rewrite the Equation 3.15, the following formula is derived:

$(I - R_A) * t_X = t_A - (R_X t_B)$	Equation 3.20

Multiplying $(I - R_A)^T$ of each side, t_X can be found shows as follows:

$t_X = ((I - R_A)^T * (I - R_A))^{-1} * \left((I - R_A)^T * \left(t_A - (R_X t_B)\right)\right)$	Equation 3.21

3.3 K-D Tree and Nearest Search Algorithm

Whether using RGBD cameras for indoor applications or LiDar for outdoor applications which less is affected by light, the raw data they output is a disordered point cloud. Usually, the output frequency of these devices is 30 to 60 Hz, and the resolution of a Realsense depth frame is 680*480, the number of point clouds in a single frame is 307200. Each point contains XYZ position information. If these data are stored out of order, corresponding algorithms such as nearest point search and deletion of elements

doi:10.6342/NTU202303769

will have considerable time complexity. So, a data structure for efficiently modelling point clouds needs to be used. In 1975, JLB invented K-Dimensional Tree to model high-dimensional data to achieve the time complexity of minimum O (log n) and maximum O (n) in search (closest point), insertion point and deletion point.

The KD Tree is also a binary tree. The points in the point cloud are represented as nodes in KD Tree. These nodes can be leaf nodes that can no longer be extended or non-page nodes that can continue to generate sub-binary trees. For a k-dimensional space, the root layer represents the 0th dimension, the first layer is 1-dimensional, the k-1th layer is k-dimensional, and the kth layer represents the 0th dimension. Taking the three-dimensional space XYZ as an example, the 0th, first, and second layers represent the divisions about the X, Y and Z axes, respectively. The traditional split method of balanced KD-Tree takes the midpoint of a series of points, the left half-plane is the value smaller than the split point of the axis, and the right half-plane is the opposite.

Considering the situation that the points p_i, axis number are known. An algorithm that describe these procedure shows as follows:

doi:10.6342/NTU202303769

Algorithm 1 K-Dimensional Tree Generation algorithm

Input:

The list of points, p_i;

The number of axis, n_{axis} ;

if $sizeof(p_i) == 0$ **then**

  return None;

end if

Dimension $d_i = n_{axis}/l_i$, l_i is current layer;

Ascendingly sort p_i, according dim d_i,

$median = int(sizeof(p_i)/2)$

append point $p_i[median]$ to graph G_{KD}

$p_{il} = p_i[start : median]$ and $p_{ir} = p_i[median + 1 : end]$

repeat

  Goto Step 1: Recursively generate left leaf using p_{il} and $l_i = l_i + 1$;

until return value is $None$

repeat

  Goto Step 1: Recursively generate left leaf using p_{ir} and $l_i = l_i + 1$;

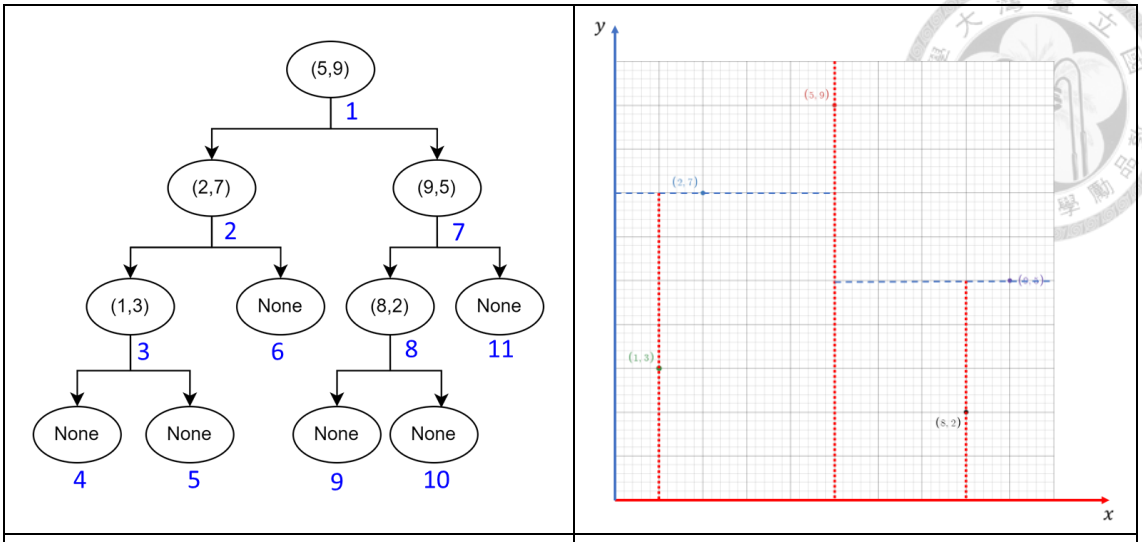
until return value is $None$

return G_{KD};

Algorithm 3.1 KD-Tree generation algorithm

For example, given the point sequence $p = (1,3), (2,7), (5,9), (8,2), (9,5)$, passing this point sequence into Algorithm 3.1, we first calculate the ascending point sets and the first node constructed is (5,9). And then, the algorithm searches the left part of point sets and right point sets. After finishing the procedure, the sequence order of each points generation shows as follows:

doi:10.6342/NTU202303769



(a) (b)

Figure 3.4 (a) Example of KD-Tree construction, (b) KD-Tree segmentation

where the blue numbers are search steps using Algorithm 3.1. The generated KD-Tree also shows in the right-hand side Figure.

After constructing a KD tree, the KD tree could be used to search for nearby points. The definition of the KD binary tree makes it clear that the left side of the limit is less than the right side of the limit, and the top is greater than the bottom, so when a point is known, a KD tree can quickly find the node which is close to the search point. After checking one side, search when the extreme closest distance is better than the current point according to the attributes of the other side; if not, skip this branch. After the algorithm re-reaches the root node, the nearest neighbor search task is completed.

doi:10.6342/NTU202303769

3.4 Iterative Closest Point registration

In most applications, the system perceives the environment according to the information from numerous sensors, like camera, depth image, LiDar. The point cloud is a well-known widespread 3-dimensional object representation technique, in which each point is characterized under the Cartesian coordinate system and colour information. A depth camera can generate a point cloud that is unorganized or organized, where points of an unorganized point cloud are disordered without any regular pattern, and each point in an organized point cloud corresponds to a pixel in an image.

Usually, there are two depth camera configuration types for a robotic system, fixed and eye-in-hand, mentioned in Section 2.2, and the EiH solution has been selected for this paper. A system that is designed to simply merge sequence point clouds of observing objects by changing the joint state of a manipulator without the consideration of calibration is usually not aligned well. The error comes from cameras and robotic kinematics. Different camera to object distance, angles and inaccuracy camera calibration parameters contributes errors into point cloud perception. Furthermore, the robotic forward kinematics is an ideal condition without considering the heat error, motor encoder error, robotic link deformation problem. The sum of the different types of errors contributes to the point cloud not being perfectly aligned. Therefore, for the elimination

doi:10.6342/NTU202303769

of errors, point cloud registration is necessary. The current point cloud registration algorithm includes iterative closest point, robust point matching, Kernel correlation , coherent point drift and sorting the correspondence space .Since the system uses the robotic arm as the camera motion carrier, the initial guess between each point cloud is known. Therefore, this paper uses ICP, which is relatively simple to implement and return rigid transformation matrix in point cloud registration.

The definition of ICP point cloud registration is that the finite size *target* point cloud $\{CL_{tar}\}$ with m points and the input *source* $\{CL_{src}\}$ point cloud with n points is known, and the source point cloud is registered to the reference point cloud. According to the, a general ICP algorithm consists of following steps, initialization rotational and translational matrix, matching two point cloud nearest inliers, transformation, and termination steps.

The purpose of ICP is to minimize the root mean square distance (RMSD) between target point cloud $\{CL_{tar}\}$ and source point cloud under the following definition:

$\min_{\mu:CL_{tar}\to CL_{src},t\in R^d,R\in SO(d)} \sqrt{\frac{1}{m}\sum_{a\in CL_{tar}} \|CL_{tar}-\mu(CL_{tar})\|^2}$	Equation 3.22

Where, d is the point cloud dimension, the SO(d) is the special orthogonal matrix.

The centroid of target and source point clouds can be defined as follows:

doi:10.6342/NTU202303769

$\overline{CL_{tar}} = \frac{1}{m}\sum_{i} CL_{tar}^{i}$	Equation 3.23

$\overline{CL_{src}} = \frac{1}{mn}\sum_{i} CL_{src}^{i}$	Equation 3.24

The difference between each point in source and target point cloud has been defined using the Equation 3.23 and 3.24 shown below:

$CL'_{tar} = CL_{tar}^{i} - \overline{CL_{tar}}$	Equation 3.25

$CL'_{src} = CL_{src}^{i} - \overline{CL_{src}}$	Equation 3.26

The minimization part in Equation 3.22 is equivalent to Equation 3.27:

$\sum_{i}^{m} \left\| R * CL_{tar}^{i} - t - CL_{src}^{i} \right\|^2 =$ $= \sum_{i}^{m} \left\| R * CL_{tar}^{i}{}' - CL_{src}^{i}{}' + (R\overline{CL\_tar} - \overline{CL_{src}} - t) \right\|^2$	Equation 3.27

The smallest value occurred when $(R\overline{CL\_tar} - \overline{CL_{src}} - t) = 0$, which $t = R\overline{CL\_tar} - \overline{CL_{src}}$. After eliminating second part in Equation 3.27, the algorithm then decomposes the rest part shows as follows:

doi:10.6342/NTU202303769

$$\sum_{i}^{m}\left\|R * CL_{tar}^{i}{}' - CL_{src}^{i}{}'\right\|^{2} = \sum_{i}^{m}\left\|CL_{tar}^{i}\right\|^{2} - 2trace\left(R\sum_{i}^{m} CL_{tar}^{i}{}' \, CL_{src}^{i}{}'^{T}\right) + \sum_{i}^{m}\left\|CL_{src}^{i}\right\|^{2}$$

Equation 3.28

The part $trace\left(R\sum_{i}^{m} CL_{tar}^{i}{}' \, CL_{src}^{i}{}'^{T}\right)$ can be obtained using singular value decomposition (SVD):

$$\sum_{i}^{m} CL_{tar}^{i}{}' \, CL_{src}^{i}{}'^{T} = U\sum V^{T}$$

Equation 3.29

The rotational matrix can be then solved using Equation 3.30:

$$R = VU\text{^}T$$

Equation 3.30

For the whole procedure, a pseudo algorithm shows as follows:

Algorithm 2 Iterative Closest Point Algorithm

Input:
Target Point Cloud: CL_{tar};
Source Point Cloud:CL_{src};
Output: Rotation Matrix **R** and transnational vector **t**;
1: Initialize **R** as identity and **t** as zeros
2: (Optional) Find nearest matching points of CL_{tar} and CL_{src}
3: Calculate point cloud centroid, $\overline{CL_{tar}}$ and $\overline{CL_{src}}$;
4: Obtain Transnational Vector $t = R * \overline{CL_{tar}} - \overline{CL_{src}}$
5: Obtain Rotational Matrix Using SVD: $R = VU^{T}$
6: **if** diff(RMSD) $\geq$ threshold **then**
7: Goto Step 2;
8: **end if**
9: Terminate Process;

Algorithm 3.2 Iterative Closest Point Algorithm

doi:10.6342/NTU202303769

3.5 Principal Component Analysis

The principal component analysis (abbr. PCA) is a universal tool to reduce the dimensions of the target data, widely used in machine learning, data analysis and pointcloud processing. Given a set of points in 2D or 3D space, the PCA is a method to find a new coordination to maximize the variance of projection of each point.

Assumed that the input points set is x, the output of PCA is z, and a function describing the PCA method as W, the following relationship has been established:

$z = Wx$	Equation 3.31

If the data is 2-dimensional, the first Principal Component (PC1) will reduce the dimension from 2 to 1. The random data following the normal distribution with scaling x axis and rotating $-30°$ shows in Figure 3.7.

Assumed that the size of points is N,in this case, the z is a scalar row, the Equation 3.31 is rewritten as follows:

$z_1 = w^1 \cdot x$	Equation 3.32

where w^1 is unit vector, or $\|w^1\|_2 = 1$

doi:10.6342/NTU202303769

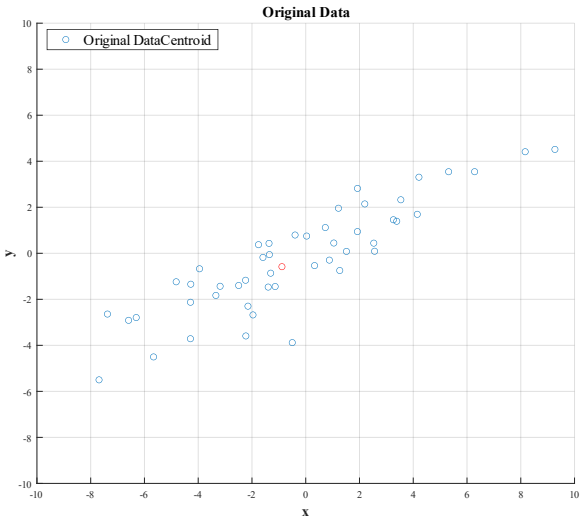


Figure 3.5 Original Data

The result of Equation 3.32 is the projection from vector x to vector w^1 shown as Figure 3.8, where this w^1 is not the optimized:

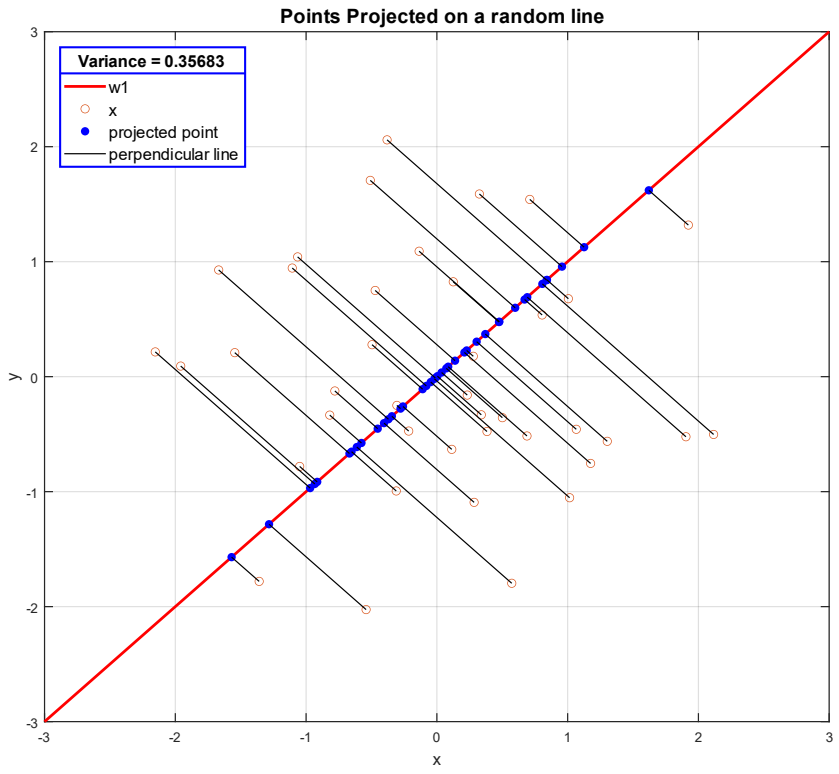


Figure 3.6 The projection from vector x to vector w^1

doi:10.6342/NTU202303769

The vector w^1 cloud be randomly distributed, the PCA is to minimize the summation of perpendicular distance between x and w^1 or maximize the variance of z_1. The Equation 3.33 represent the variance of z_1:

$$Var(z_1) = \sum_{z_1} (z_1 - \bar{z_1})^2$$	Equation 3.33

Similarly, for 3-dimensional points, the PCA reduces the 3D information to 2D plane, a new scalar row in second principal axis z_2 shows as follows:

$$z_2 = w^2 \cdot x$$	Equation 3.34

Also, the variance of z_2 is:

$$Var(z_2) = \sum_{z_2} (z_2 - \bar{z_2})^2$$	Equation3.35

Where $\bar{z_1}$ and $\bar{z_2}$ are mean value of projected point value of principal component 1 and 2, respectively. The combination of vector w^1 and w^2 consists the entire W in Equation 3.31, shown as follows:

$$W = \begin{bmatrix} (w^1)^T \\ (w^2)^T \\ \dots \end{bmatrix}$$	Equation3.36

The mean value of z_1 could be extracted from Equation 3.32:

$$\bar{z_1} = \frac{1}{N}\sum z_1 = \frac{1}{N}\sum w^1 \cdot x = w^1 \cdot \frac{1}{N}\sum x = w^1 \cdot \bar{x}$$	Equation 3.37

doi:10.6342/NTU202303769

Also, the derivation of equation 3.34 shows as follows:

$$Var(z_1) = \sum_{z_1} (z_1 - \bar{z_1})^2$$

$$= \sum_{x} (w^1 \cdot x - w^1 \cdot \bar{x})^2$$

$$= \sum_{x} (w^1 \cdot (x - \bar{x}))^2$$

$$= \sum_{x} (w^1)^T (x - \bar{x})(x - \bar{x})^T w^1$$

$$= (w^1)^T \left[\sum_{x} (x - \bar{x})(x - \bar{x})^T \right] w^1$$

$$= (w^1)^T cov(x,) w^1$$

Equation 3.38

And then, the covariance $cov(x)$ is symmetric and positive-semidefinite matrix as S. To maximize the Equation 3.38, $(w^1)^T S w^1$, the Lagrange multiplier is utilized, which finds the local limit value of multiple variable function when one or more variable is under some constraint. This method converts the optimization problem with n variable and k constraints to an equation set with $n + k$ variables. In this case, there are one constraint and N variable. The Equation 3.39 represent the Lagrange method used for Equation 3.38:

$$L(Var(z_1), \lambda) \equiv L(w^1, \alpha) = (w^1)^T S w^1 - \alpha((w^1)^T w^1 - 1)$$ Equation 3.39

where the second term is the constraint of w^1, $\|w^1\|_2 = 1$.

To solve the Equation 3.39, the LM method calculates the partial derivation of $L(w^1, \lambda)$, that is

doi:10.6342/NTU202303769

$\frac{\partial L(w^1, \alpha)}{\partial w_1^1} = 0$ $\frac{\partial L(w^1, \alpha)}{\partial w_2^1} = 0$ ……	Equation 3.40

The solution of Equation 3.40 is

$Sw^1 - \alpha w^1 = 0$ $Sw^1 = \alpha w^1$	Equation 3.41

Where w^1 is the eigenvector of S. And the algorithm needs to calculate all eigenvectors to maximize the $Var(z_1)$.

Rewriting Equation 3.38:

$(w^1)^T\, cov(x)\, w^1 = (w^1)^T\, S\, w^1 = \alpha (w^1)^T w^1 = \alpha$	Equation 3.42

The size of α is depended on $(w^1)^T\, S\, w^1$, which means the maximum α exists when w^1 is the eigenvector of the covariance matrix S corresponding to the **largest** eigenvalue λ_1.

The pointcloud information is stored with 3-dimensional points, consisting information of x, y and z axis. The extracted 1st principal component is the main orientation of these points, which projects 3-dimensional data into one axis. After determining the direction of first principal component, the second principal component

doi:10.6342/NTU202303769

cloud also be derived using similar principle with additional constraints for Equation 3.35 shown as follows:

1) $\|w^2\|_2 = 1$, and

2) $w^1 \cdot w^2 = 0$, that is w^1 and w^2 is orthogonal.

Like Equation 3.39, the Lagrange multiplier applied to covariance function of w_2 shown as follows:

$L(Var(z_2), \alpha, \beta) \equiv L(w^2, \alpha, \beta)$ $= (w^2)^T S\, w^2 - \alpha((w^2)^T w^2 - 1)$ $- \beta((w_2)^T w_1 - 0)$	Equation 3.43

The two terms scaled by α and β are two constraints mentioned before. To solve this optimization problem, the algorithm determines the partial derivation of $L(w^2, \alpha, \beta)$.

$\frac{\partial L(w^2, \alpha, \beta)}{\partial w_1^2} = 0$ $\frac{\partial L(w^2, \alpha, \beta)}{\partial w_2^2} = 0$	Equation 3.44

The solution of Equation 3.45 is:

$\mathrm{S}w^2 - \alpha w^2 - \beta w^1 = 0$	Equation 3.45

doi:10.6342/NTU202303769

Multiplying both size of Equation 3.45 by $(w^1)^T$, we get:

$(w^1)^T S w^2 - \alpha (w^1)^T w^2 - \beta (w^1)^T w^1 = 0$ $(w^1)^T S w^2 - \alpha * 0 - \beta * 1 = 0$ $[(w^1)^T S w^2]^T - \beta = 0$ $(w^2)^T S^T w^1 - \beta = 0$ $(w^2)^T S w^1 - \beta = 0$ $\alpha (w^2)^T w^1 - \beta = 0$ $LHS = \alpha (w^2)^T w^1 - \beta = \alpha * 0 - \beta = -\beta = RHS, if\ \beta = 0$	Equation 3.46

The equation then rewrites as follows:

$S w^2 - \alpha w^2 = 0$	Equation 3.47

$S w^2 = \alpha w^2$	Equation 3.48

The maximum covariance under w^2 direction is when w^2 is the eigenvector of the covariance matrix S corresponding to the **second largest** eigenvalue: λ_2. An example result of PCA shows in Figure 3.9, where the two blue line represent the original axes, the current x-axis and y-axis are PC1 and PC2, respectively. The red points are transformed data from Figure 3.7.

doi:10.6342/NTU202303769

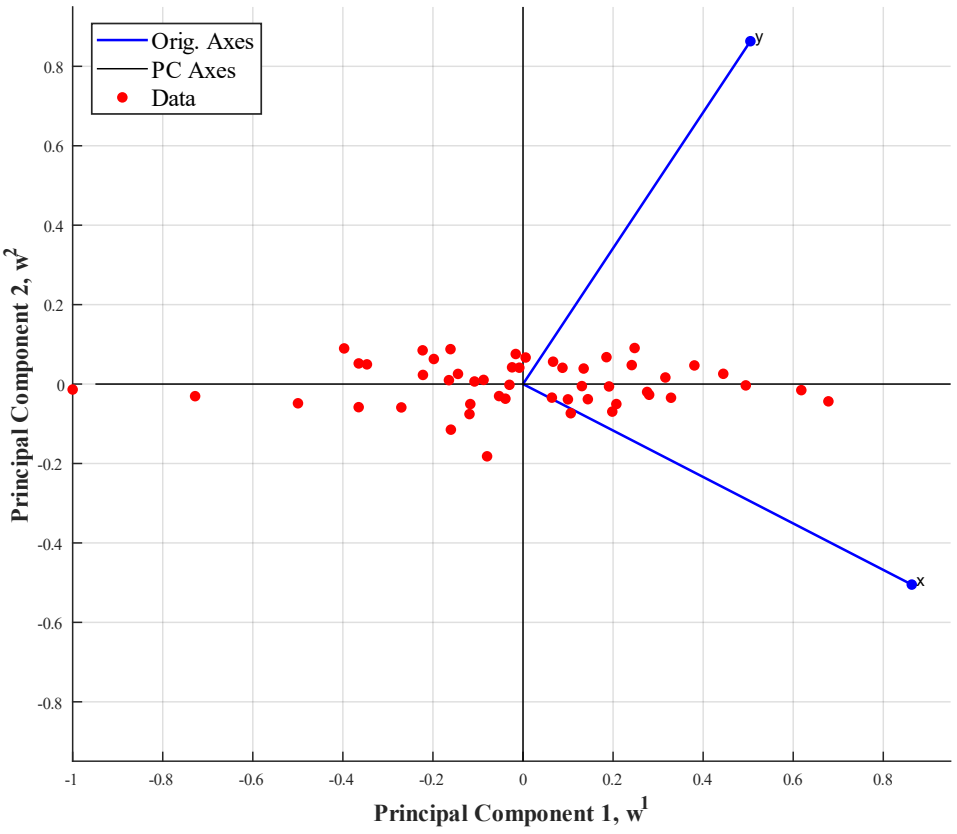


Figure 3.7 An example of PCA result

Finally, the Point Cloud Library (PCL) and MATLAB both provides methods to efficiently calculate the PCA.

doi:10.6342/NTU202303769

Chapter 4
System Overview

4.1 System Architecture

The main purpose of this thesis is to automatically locate the plant in 3D space, finish plant height generation, pointcloud reconstruction, organ segmentation and post-processing tasks. To meet the previous demand, an automatic manipulator vehicle is designed and constructed. In the hardware configuration, an unmanned ground track vehicle, a robotic manipulator, a depth camera and an artificial illuminance are merged together. The participation of UGV expands the workspace from local to part or entire of the greenhouse. A large range of robotic manipulator covers most area of the observed plant which helps for dedicated scanning and alignment. The depth camera provides the pointcloud converted from environment information based on two infrared camera and triangulation technique. The artificial illumination shines the observed plant in uniform way. For software of this system, the robotic operating system (ROS) is considered as an integrating platform. The robotic manipulator and unmanned ground vehicle have their own low-level controller. The official-provided ROS driver bridges the robotic manipulator and ROS node. Similarly, the currently used Realsense also enabled using official ROS package, which the pointcloud, image topics are provided. The artificial

doi:10.6342/NTU202303769

illumination system is made by two sunlight spectrum LEDs. The system architecture shows as follows:

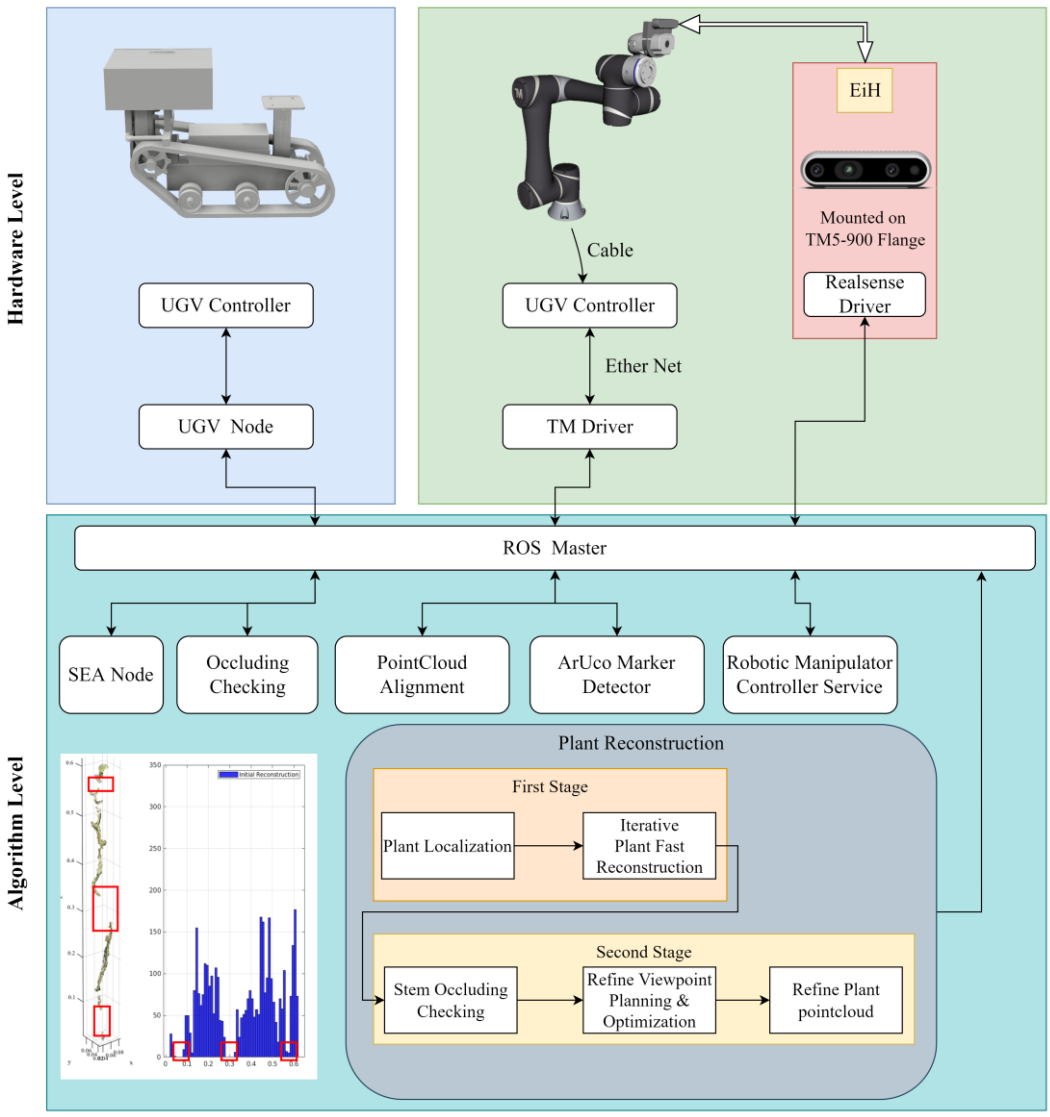


Figure 4.1 The system architecture

doi:10.6342/NTU202303769

4.2 Coordinate Systems

In this system, there are varieties of coordination system which is complicated. The purpose of different coordination is to simply the problem or represent the different physical environment. The definition of coordination system is a relative notation. For example, the center of the greenhouse could be zero or some latitude and longitude in the Earth. Also, there has many unused coordination systems for example the depth frame of the depth camera. The following graph illustrate the different used coordination system in the simulated Gazebo (same as the real-world) environment.

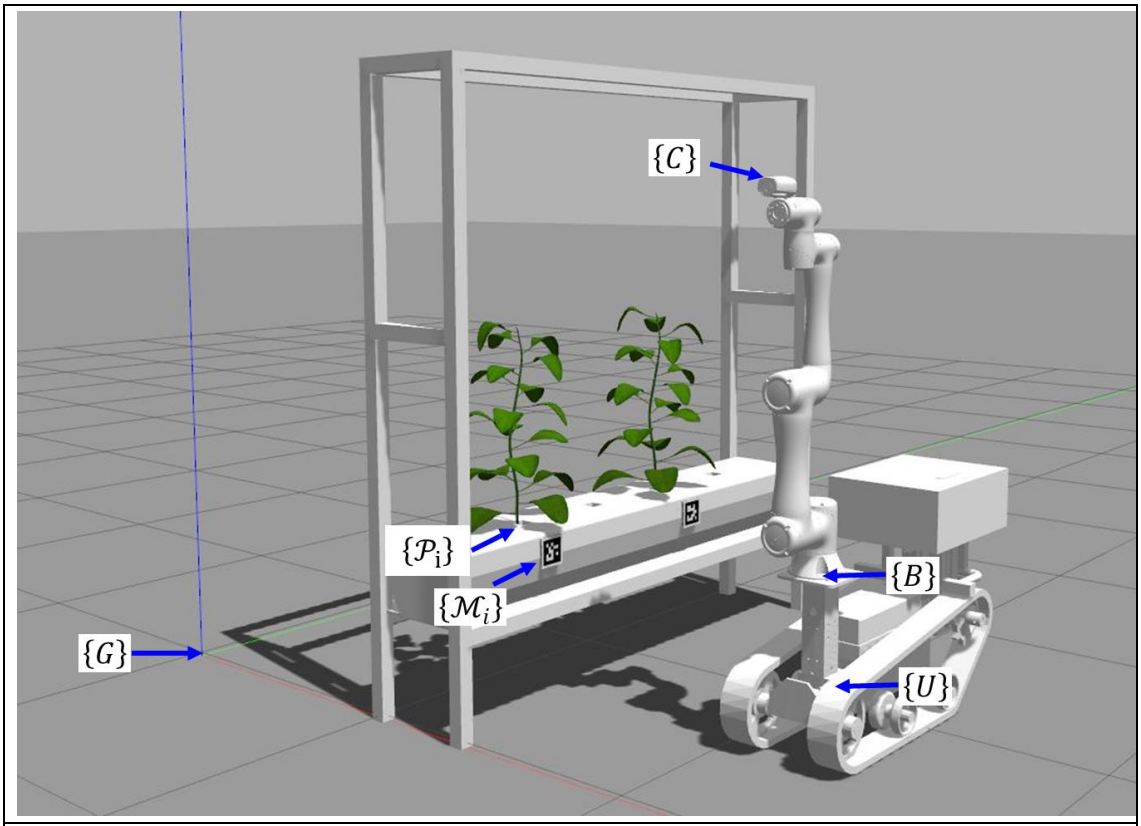


Figure 4.2 Coordination System

doi:10.6342/NTU202303769

The following table shows the detail of each coordination system:

Index	Symbol	Description
1	$\{\mathcal{P}_i\}$	The i-th plant in the greenhouse, the center of the $\{\mathcal{P}_i\}$ is considered at the root region.
2	$\{B\}$	The center of $\{B\}$ is located on the center of the robotic manipulator
3	$\{C\}$	The camera coordination, especially represent the optical frame of the Realsense, which the Realsense fixture not shown in Figure 4.1.
4	$\{\mathcal{M}_i\}$	The i-th localization fiducial marker in the greenhouse, the center of $\{\mathcal{M}_i\}$ is defined by user or originally located at the center of the marker
5	$\{G\}$	Greenhouse coordination, used for UGV navigation
6	$\{U\}$	The center of the UGV

4.3 Controller Design

To solve the plant pointcloud reconstruction, organ (leaf, stem, petiole) segmentation problem, a multi-stage feedback plant reconstruction system is designed. This system currently has two stages, plant localization, pointcloud fast reconstruction and plant height analysis in first stage, and stem-based occluding checking and refinement pointcloud procedure. After the system finishes these works, the system executes stem-based pointcloud organ segmentation algorithm to distinguish leaf, petiole and stem. These segmented leaves will be then post-processed.

In the first stage of the procedure, the system does not know where the plant is located. Dedicated time-consuming plant reconstruction is not cost-efficient. The design

doi:10.6342/NTU202303769

of this control stage is first to locate the plant location from placed fiducial marker by rotating the robotic manipulator itself. After the fiducial marker located, the system will plan the closer view to this marker $\{\mathcal{M}_i\}$ for getting more accurate fiducial marker readings. The system then calculates the transformation from $\{B\}$ to $\{\mathcal{P}_\mathrm{i}\}$ given the transformation from $\{\mathcal{M}_i\}$ to $\{\mathcal{P}_\mathrm{i}\}$. This is the first part of the first stage – plant localization. If the plant is located, the system planned several fixed vertical fixed viewpoints to roughly scans and reconstructs the observed plant by capturing each frame of pointcloud, filtering pointcloud and aligning. The height of each frame of reconstructed pointcloud cloud be calculated and the height increase rate is considered as a termination sign of the first stage of procedure. This is a trade-off method to minimize the spent time and maximize the efficiency.

In the start of second stage, the plant pointcloud is basically reconstructed without verification. The stem will be extracted by using SE Algorithm in Chapter 5, and the stem occluding region will be highlighted for future processing. The controller finds the missing stem part because the plant pointcloud segmentation algorithm will generate more reliable segmented plant organs if the critical region is dense. This system uses the detection-planning-optimization-alignment-detection loop to finish the close loop plant pointcloud enhancement task.

doi:10.6342/NTU202303769

In summary, the entire level of the controller is not limited on low-level control, such as the PID control of a motor or the pose control of the robotic manipulator but focus on the current pointcloud information. The low-level control is generally tuned or locked by manufactor for a highly-integrated system. In this paper, a series of task-specific pointcloud-based evaluation algorithms have been developed. The imperfect or defeat region or part is generated, and the robotic manipulator uses this information to re-planning and refining the pointcloud as a closed feedback loop. Meanwhile, in the first stage of this procedure, the controller planned the fixed pose with respect to the plant coordination frame to cover the maximum observed height. In the real-world experiment, the height of plant varies from plant to plant. The most efficient way is to calculate the current plant height and height changing rate. If the height changing rate lower than specific threshold, the first part of procedure stops.

doi:10.6342/NTU202303769

Chapter 5
Plant Analyzing Algorithms

A well-designed procedure to reconstruct plant and real-time pointcloud processing algorithm are both important. In this chapter, several plant algorithms used in plant reconstruction and post-processing will be detailly introduced. In Section 5.1, To merge plant pointcloud from different viewpoints with acceptable observation error, an enhanced pointcloud registration method is introduced. In Section 5.2, a geometric stem extraction algorithm (SEA) which generate stem pointcloud from original dense pointcloud is introduced. Sometimes, the stem information may be occluded from the captured pointcloud under specific viewpoint, a SEA-based stem occluding approach is discussed in Section 5.3. In the Section 5.4, a pos-processing stem-based plant pointcloud segmentation method is introduced.

doi:10.6342/NTU202303769

5.1 Marker-Based Plant Filter

A robust filter set has been deployed for original pointcloud illustrated in the following figure.

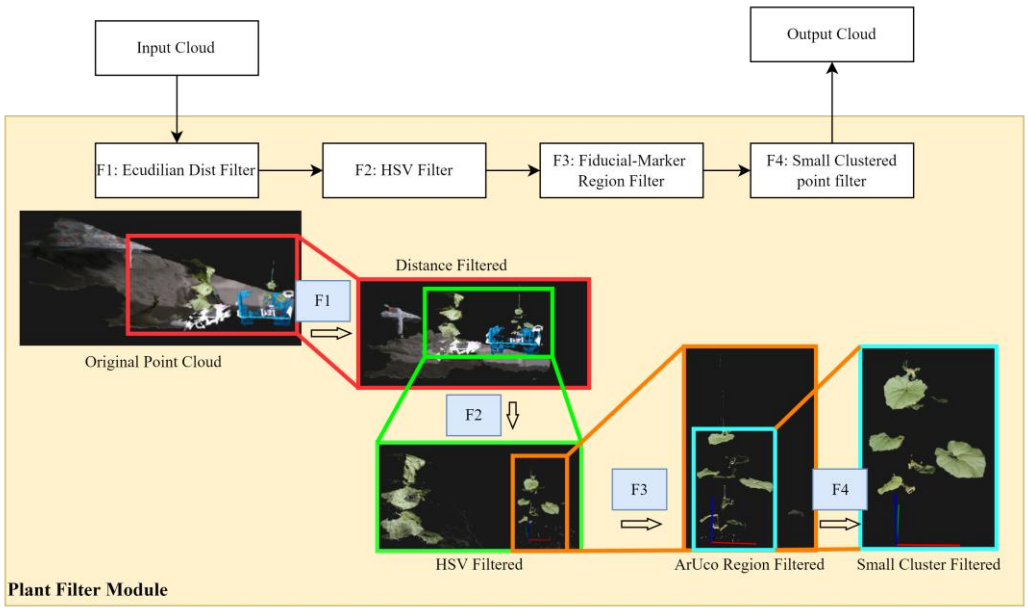


Figure 5.1 Plant Filter Module

Assume that the input pointcloud at time $t[i]$ under camera frame is ${}^{\{C\}}P_{t[i]}$. The transformation from camera frame to robotic base frame is given from forward kinematics. The ${}^{\{C\}}P_{t[i]}$ firstly is converted to robotic base frame:

$$ {}^{\{B\}}P_{t[i]} = T_B^C(q) * {}^{\{C\}}P_{t[i]} \quad \text{Equation 5.1} $$

doi:10.6342/NTU202303769

The first step of this plant filter is to transform the pointcloud from robotic base frame $\{B\}$ to plant frame $\{\mathcal{P}_i\}$:

$^{\{\mathcal{P}_i\}}P_{t[i]} = T^{B}_{\mathcal{P}_i} * {}^{\{B\}}P_{t[i]}$	Equation 5.2

The current pointcloud origin of $^{\{\mathcal{P}_i\}}P_{t[i]}$ is theoretical located on plant root. A "Euclidean Distance Filter (EDF)" is firstly applied to this original pointcloud $^{\{\mathcal{P}_i\}}P_{t[i]}$. The principle of EDF is calculating the distance from any point to the pointcloud origin, and add this point if:

$\left\| {}^{\{\mathcal{P}_i\}}P_{t[i]}[j] \right\|_2 < r_{th,EDF}$	Equation 5.3

Where the $^{\{\mathcal{P}_i\}}P_{t[i]}[j]$ is the j-th point in pointcloud $^{\{\mathcal{P}_i\}}P_{t[i]}$ and $r_{th,EDF}$ is the radius threshold of Euclidean Distance Filter. This step will increase the filtering speed in the run-time level.

Secondly, the algorithm applies the HSV filter to remove any point that has literally green color. The advantage of HSV is that this color representation method extracts color information into one channel (Hue) from 0 to $360°$. Other two channels are Saturation and Value, which describes the richness of the color and brightness of the color. The following graph shows the color map of hue channel:

doi:10.6342/NTU202303769

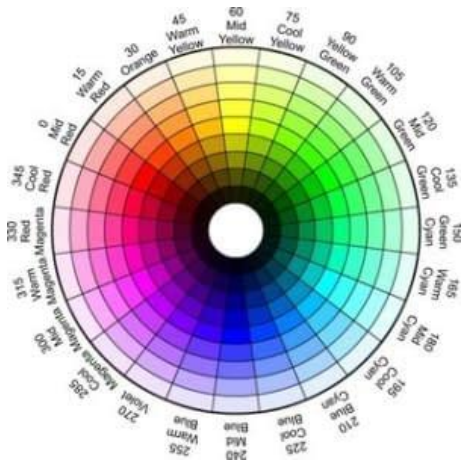


Figure 5.2 Hue wheel [87 Stack Overflow 2016]

Generally, three upper limitation and three lower boundaries is defined, which are $H_l, H_u, S_l, S_u, V_l, V_u$. Any color of point inside this region will be reserved. The function is simply defined as follows:

$P_{out}^{HSV} = HSV(P_{out}^{EDF}, H_l, H_u, S_l, S_u, V_l, V_u)$	Equation 5.4

Thirdly, the filter keeps the current plant using a 3-dimensional box described with three lower and upper bound, called Fiducial-Marker Based Region Filter (FMBRF). Because the current origin frame is considered as plant root.

$P_{out}^{MRF} = FMBRF(P_{out}^{HSV}, x_l, x_u, y_l, y_u, z_l, z_u)$	Equation 5.5

Assumed that a box consisted of six constraint $x_l, x_u, y_l, y_u, z_l, z_u$, any point inside this box will be kept or removed.

doi:10.6342/NTU202303769

Finally, to remove small cluster noise, Secondly, the algorithm finds all possible clusters in point cloud P_{out}^{MRF} using Euclidean Cluster Extraction (ECE) [88 Rusu 2010]. The size of cloud $\boldsymbol{p_{A,i,j}}$ defined as $N_{A,i,j}$. The ECE algorithm conducted under following steps:

1. Build a queue $\boldsymbol{Q}$ and cluster $\boldsymbol{C}$
2. For every $P_{out}^{MRF}[\boldsymbol{k}] \in P_{out}^{MRF}, \boldsymbol{k} \in [\boldsymbol{0}, size(P_{out}^{MRF})]$
    a. Add $P_{out}^{MRF}[\boldsymbol{k}]$ to **Q**
    b. For every $P_{out}^{MRF}[\boldsymbol{k}] \in \boldsymbol{Q}$
        i. Find nearest M points set $P_{out}^{MRF}[\boldsymbol{k}]_{\boldsymbol{m}}$ of $P_{out}^{MRF}[\boldsymbol{k}]$ with the Euclidean distance < d_{th}
        ii. If $P_{out}^{MRF}[\boldsymbol{k}]_{\boldsymbol{m}}$ is already added into C or Q, ignore it, or add it to $\boldsymbol{Q}$
    c. When all point in Q has been processed, add $\boldsymbol{Q}$ to $\boldsymbol{C}$
3. Terminate when k = $size(P_{out}^{MRF})$.

After that, any size of cluster is less than N_{noise} will be removed.

The result pointcloud of this entire plant filter processing is defined as ${}^{\{\mathcal{P}_i\}}P'_{t[i]}$.

5.2 Iterative Closest Point Implementation

After that, the procedure needs to execute iterative closest point method to register a series of pointcloud. The main principle of ICP has been introduced in Chapter 3. The following figure illustrates how this algorithm uses ICP approach.

doi:10.6342/NTU202303769

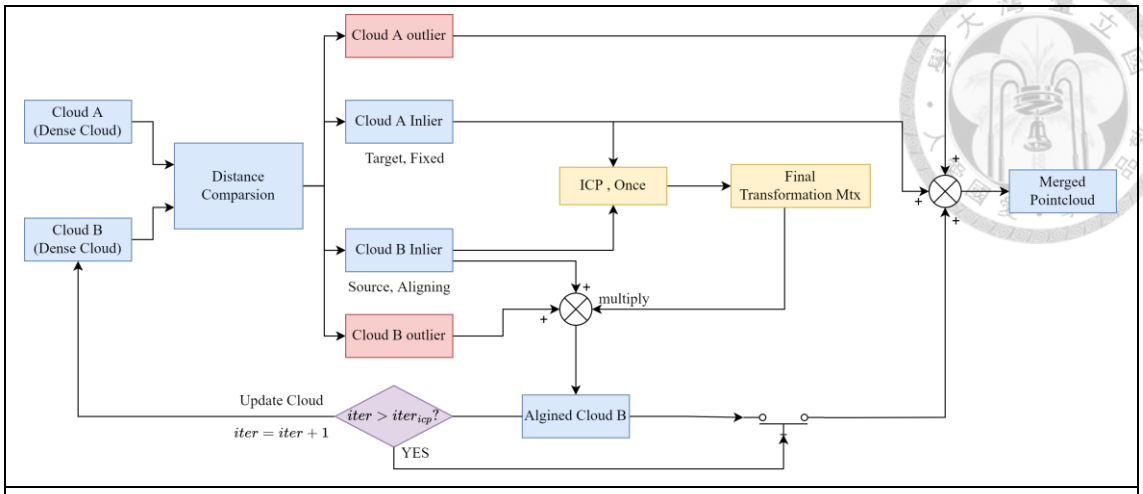


Figure 5.3 The Iterative closest point procedure

The ICP approach has two input pointcloud, defined as cloud A and B in Figure 5.3, which are also ${}^{\{\mathcal{P}_i\}}P'_{t[i]}$ and ${}^{\{\mathcal{P}_i\}}P'_{t[i+1]}$. The $t[i]$ and $t[i+1]$ literately describes these pointcloud are captured from different time or viewpoint. The distance comparison module compares the overlapping region of this two pointcloud. Assumed that, the point in ${}^{\{\mathcal{P}_i\}}P'_{t[i]}$ and ${}^{\{\mathcal{P}_i\}}P'_{t[i+1]}$ are indexed using character j and k. And the algorithm also needs a distance threshold imported as d_{icp}. The following algorithm mathematically represent this ICP procedure:

1. Input: ${}^{\{\mathcal{P}_i\}}P'_{t[i]}$, ${}^{\{\mathcal{P}_i\}}P'_{t[i+1]}$, d_{icp} and $iter_{icp}$
2. Initialize cloud container: $P^{inlier}_{traget}, P^{outlier}_{traget}, P^{inlier}_{source}, P^{outlier}_{source}$, current iteration: $iter$
3. For each point ${}^{\{\mathcal{P}_i\}}P'_{t[i]}[j] \in {}^{\{\mathcal{P}_i\}}P'_{t[i]}$:
    a. Find the nearest point ${}^{\{\mathcal{P}_i\}}P'_{t[i+1]}[k] \in {}^{\{\mathcal{P}_i\}}P'_{t[i+1]}$
    b. Calculate $d_{ij} = \left\| {}^{\{\mathcal{P}_i\}}P'_{t[i]}[j] - {}^{\{\mathcal{P}_i\}}P'_{t[i+1]}[k] \right\|$

doi:10.6342/NTU202303769

i. If $d_{ij} < d_{icp}$, $P_{traget}^{inlier} = P_{traget}^{inlier} \cup {}^{\{\mathcal{P}_i\}}P'_{t[i]}[j]$

ii. Else: $P_{traget}^{outlier} = P_{traget}^{inlier} \cup {}^{\{\mathcal{P}_i\}}P'_{t[i]}[j]$

4. For each point ${}^{\{\mathcal{P}_i\}}P'_{t[i+1]}[k] \in {}^{\{\mathcal{P}_i\}}P'_{t[i+1]}$

   a. Find the nearest point ${}^{\{\mathcal{P}_i\}}P'_{t[i]}[j] \in {}^{\{\mathcal{P}_i\}}P'_{t[i+1]}$

   b. Calculate $d_{ij} = \left\| {}^{\{\mathcal{P}_i\}}P'_{t[i+1]}[k] - {}^{\{\mathcal{P}_i\}}P'_{t[i]}[j] \right\|$

      i. If $d_{ij} < d_{icp}$, $P_{source}^{inlier} = P_{source}^{inlier} \cup {}^{\{\mathcal{P}_i\}}P'_{t[i+1]}[k]$

      ii. Else: $P_{source}^{outlier} = P_{source}^{outlier} \cup {}^{\{\mathcal{P}_i\}}P'_{t[i+1]}[k]$

5. Execute ICP, the transformation matrix defined as: T

6. $iter = iter + 1$

   a. If $iter < iter_{icp}$, ${}^{\{\mathcal{P}_i\}}P'_{t[i+1]} = T * {}^{\{\mathcal{P}_i\}}P'_{t[i+1]}$, go to step 1.

   b. Else: go to step 7

7. Output merged pointcloud $P_{merged}[i]$.

Where the $P_{merged}[i] = P_{traget}^{inlier} + P_{traget}^{outlier} + T * \left(P_{source}^{inlier} + P_{source}^{outlier}\right) =$

${}^{\{\mathcal{P}_i\}}P'_{t[i]} + T * {}^{\{\mathcal{P}_i\}}P'_{t[i+1]} = P_{global}$.

In our head-on viewpoint plant reconstruction procedure, the merged pointcloud $P_{merged}[i]$ is then assigned to P_{global}, which updates iteration by iteration. If, in the first head-on viewpoint iteration, the size of pointcloud P_{global} is zero. In this situation, the ICP algorithm is switched to a passthrough function block, which means the global pointcloud is direct equals to the first captured pointcloud ${}^{\{\mathcal{P}_i\}}P'_{t[i]}$.

doi:10.6342/NTU202303769

5.3 Stem Extraction Algorithm (SEA)

The stem extraction algorithm (SEA) provides the specific feature information of a given point cloud, which is the foundation of next viewpoint planning algorithm. The purposed SE Algorithm can automatically **Downsample the pointcloud**, designate **search start entry points**, **calculate gain function of each candidate pairs**, **filter results by principal component analysis and sizes**. Figure 6.X illustrates the whole procedure of the SE algorithm detailly.

First of all, the input point cloud of SE Algorithm is well-filtered point cloud which only contains the target plant shown as Figure 5.4.

(a)

(b)

Figure 5.4 Example of reconstructed plant from filter algorithm

doi:10.6342/NTU202303769

Usually, the plant point cloud consists of tens of thousands to hundreds of thousands of discrete points. The system needs to downsample the whole point cloud to process the point cloud faster and efficiently, although some information will be lost. Voxel is a proper term to describe the distance between points in three-dimensional space, just like the pixel we are familiar with - the smallest unit of an image. Under experimental data, the density of the pointcloud acquired by RealSense is about 10 to 20 points /(5mm^3) in executing plant reconstruction work. For the common algorithm complexity O(n) or O(n^2), the time will be reduced by 10~20 or 100~400 times, respectively. This will significantly speed up the processing of point clouds. The followed figure illustrates the difference between original point cloud and downsampled pointcloud.

The melon plant is cultivated based on high-wire method with human intervention, the growing direction is along the Z-axis with very large probability. The lower part of the plant is considered to be the first region to develop, from which the main stem will extend upwards.

doi:10.6342/NTU202303769

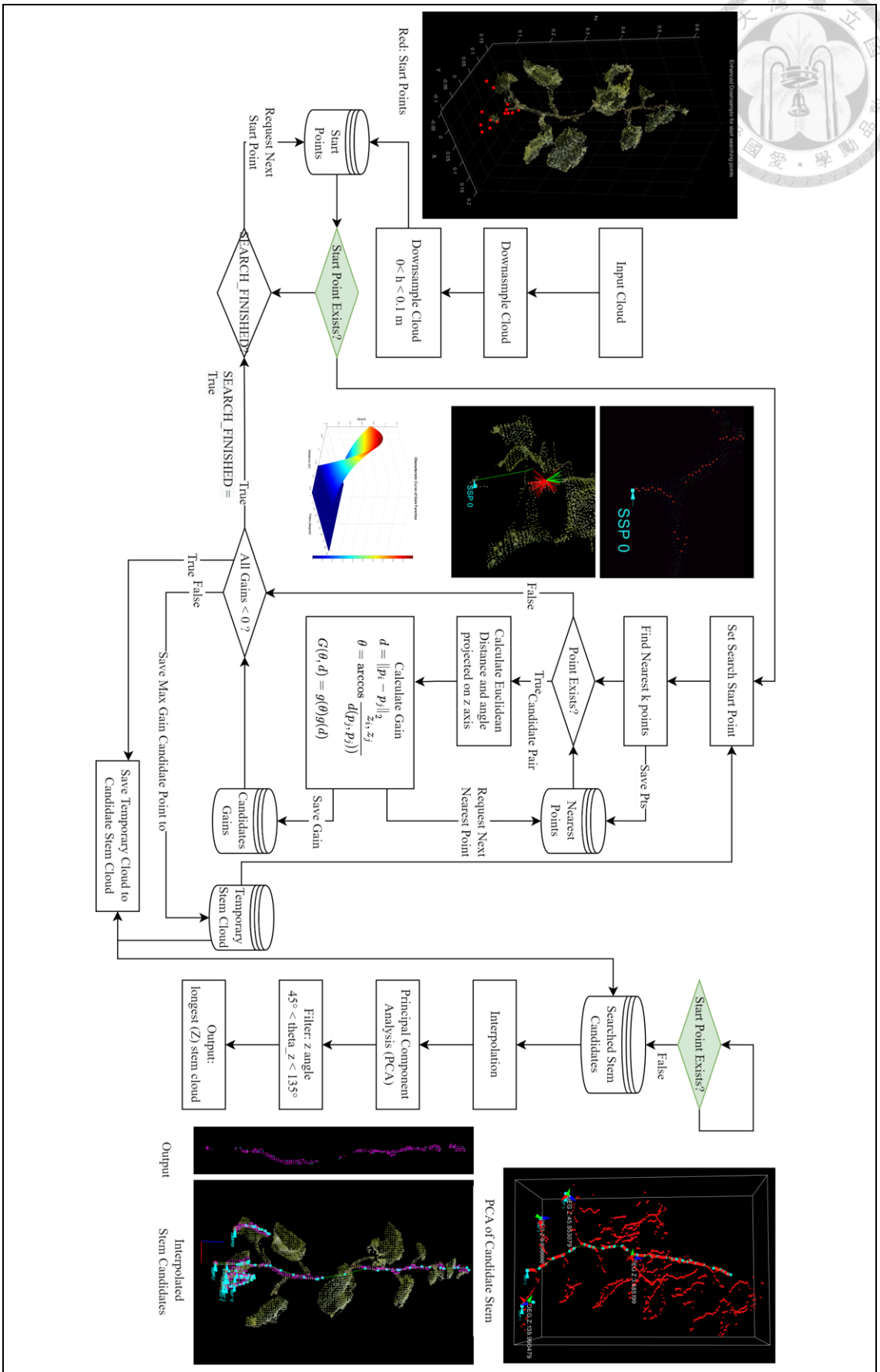


Figure 5.5 The structure of stem extraction algorithm

doi:10.6342/NTU202303769

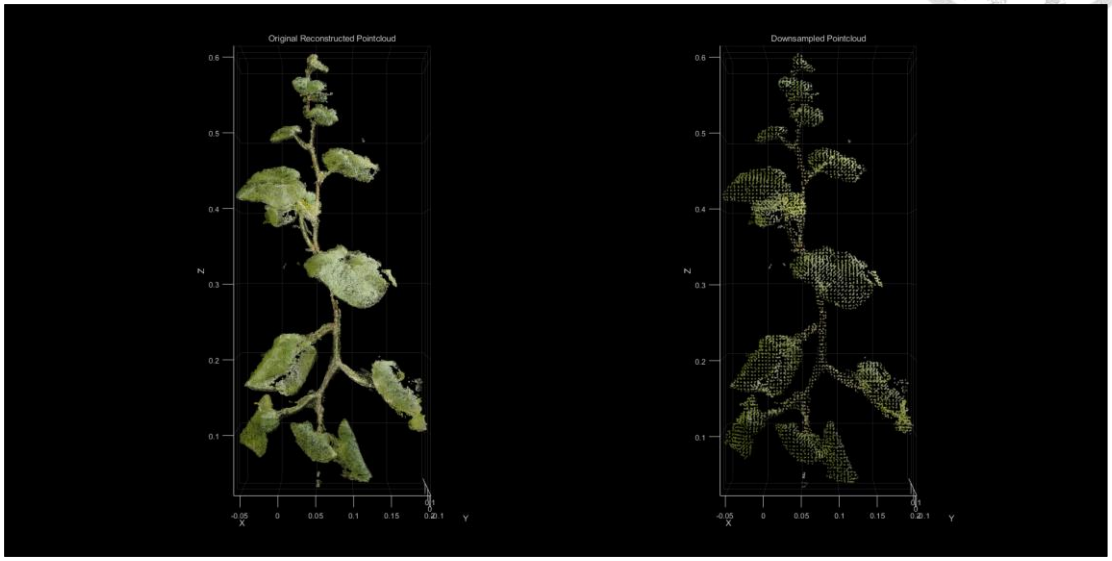


Figure 5.6 original point cloud and downsampled pointcloud

Secondly, the algorithm should determine which underneath points are the correct stem start point. Although the plant has been cultivated upright, the stem may not always locate in the center of x-y plane due to nature growth differences and observation error. Without the consideration of utilizing the pervious stem cloud to assist current stem extraction procedure, a proper start point is critical for whole stem extraction algorithm. If the system only selects lowest point of the cloud $P[\mathrm{n}]$, occasionally, leaf or side vein may be chosen, which are not we desired. To overcome this problem, downsampled pointcloud with large voxel size is a possible method to find start searching point.

doi:10.6342/NTU202303769

Therefore, assumed one input of plant pointcloud is $P[\mathrm{n}], \left\{z \in \left(0, \max_z P[\mathrm{n}]\right]\right\}$ and the global downsampled pointcloud from $P[\mathrm{n}]$ is $P_{d_1}[\mathrm{n}] \left\{z \in \left(0, \max_z P_{d_1}[\mathrm{n}]\right]\right\}$ with uniform voxel size d_1. After that, the lower part of $P[\mathrm{n}]$, especially for any points with the z component from 0 to h_{SSR}, will be downsample with uniform voxel size d_2, represent as $P_{d_2}[\mathrm{n}] \in \{(0, h_{SSR}]\}$, where the d_2 is greater than d_1. The input pointcloud for following stem extraction algorithm is defined as $P_d[\mathrm{n}] = P_{d_1}[\mathrm{n}]$ and the $\boldsymbol{SSP}$ set is P_{d_2}. The candidate starts searching point shows in Figure 5.7, and highlighted using red point.

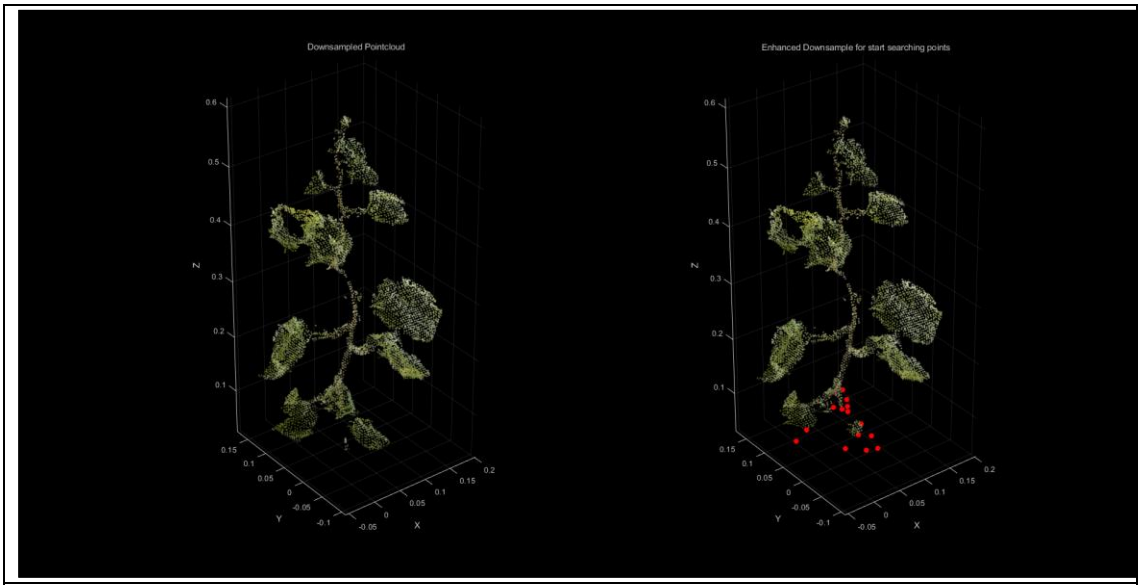


Figure 5.7 Varies Start Searching Point (SSP) found by downsample method

Once the candidate starts searching point has been found, the program will continue to execute the main function block – stem extraction. Figure 5.8, illustrates the initial

doi:10.6342/NTU202303769

stage of stem extraction function, where the red point represents the start searching point found by step 3.

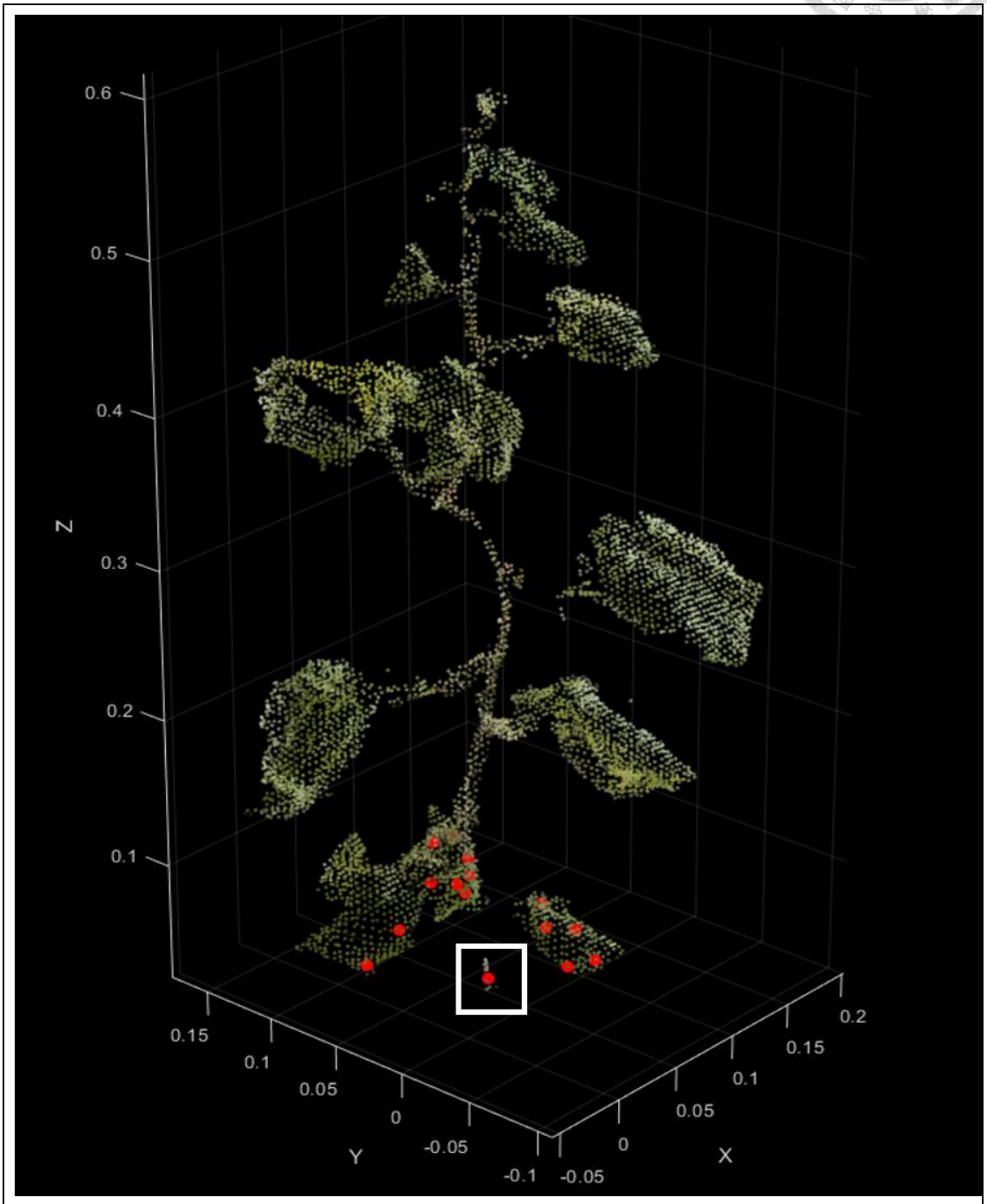


Figure 5.8 Close view of extracted start searching point which labelled as red

doi:10.6342/NTU202303769

In Figure 5.8, all points including SSP and plant point can be selected for the SE algorithm, where the white rectangle box represents the main-stem plant region labelled by hand. Other part of SSPs is leaves or petioles because this algorithm is guaranteed that all start searching point is located on stem. From the empirical evidence and [some reference], the petiole, connecting a leaf / leave and main stem, usually grown horizontally or have significant growing direction differences from main stem in Figure 5.9.

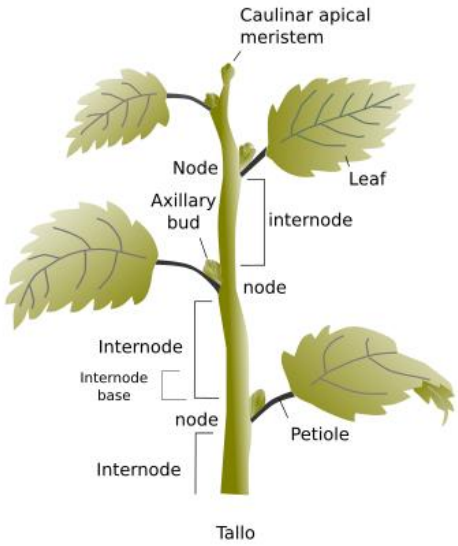


Figure 5.9 A common plant structure

The concept of stem extraction is to use a gain function to evaluate the acceptance extent. To avoid selecting a wrong point, ex: petiole, mentioned before, the angle between a vector $\overrightarrow{\boldsymbol{p}_{start}\boldsymbol{p}_{next}}$ and $\hat{\boldsymbol{z}}$ should be constrained, where the p_{start} and $\boldsymbol{p}_{next}$ are start point and next possible point respectively, as a candidate pair in SE Algorithm .

doi:10.6342/NTU202303769

Furthermore, the angle is not the only constraint of the SE function, because a long-distance vector $\overrightarrow{\boldsymbol{p}_{start}\boldsymbol{p}_{next}}$ is not reliable that omitting much information while that of short-distance costs large amount of computational resources. That is, the distance is an important variable we should concern additionally.

The offset angle is described how much vector $\overrightarrow{\boldsymbol{p}_{start}\boldsymbol{p}_{next}}$ deviates from unit vector of Z ($\hat{\boldsymbol{z}}$), defined as follows:

$\psi_{start}^{next} = \arccos\frac{abs\left(z_{\boldsymbol{p}_{next}} - z_{\boldsymbol{p}_{start}}\right)}{d_{start}^{next}}$	Equation 5.6

And the Euclidean distance between point p_{start} and p_{next} is given in following formula, Equation **5.7**:

$d_{start}^{next} = \|\boldsymbol{p}_{next} - \boldsymbol{p}_{start}\|_2$	Equation 5.7

Where, the absolute value of $z_{\boldsymbol{p}_{next}} - z_{\boldsymbol{p}_{start}}$ represent the distance which vector $\overrightarrow{\boldsymbol{p}_{start}\boldsymbol{p}_{next}}$ projected on z-axis, and the d_{start}^{next} is the distance between $\boldsymbol{p}_{next}$ and $\boldsymbol{p}_{start}$. To avoid the program selecting a point with either large ψ_{start}^{next} or small ψ_{start}^{next} with large d_{start}^{next}, a limitation of z-offset angle and distance has been purposed, which represent as ψ_{th} and r_{th}, respectively.

The following terms define the individual score of angles and distances respectively:

doi:10.6342/NTU202303769

$$g(\psi) = \begin{cases} \dfrac{e^{-\frac{\psi}{\psi_{th}}} - e^{-1}}{(1 - e^{-1})}, & 0 \le \psi < \psi_{th} \\ 0, & \psi > \psi_{th} \end{cases}$$

Equation 5.8

and

$$g(d) = \begin{cases} -100 * (d - d_{th})^2 + 1, & 0 \le d < d_{th} + 0.1 \\ 0, & d \ge d_{th} + 0.1 \end{cases}$$

Equation 5.9

where, the second term of molecular e^{-1} and denominator $(1 - e^{-1})$ normalize the gain function of offset angle to 1, and similarly, the $g(d)$ reach its peak when $d = d_{th}$. The figure X illustrates the function plotted in x-y space:

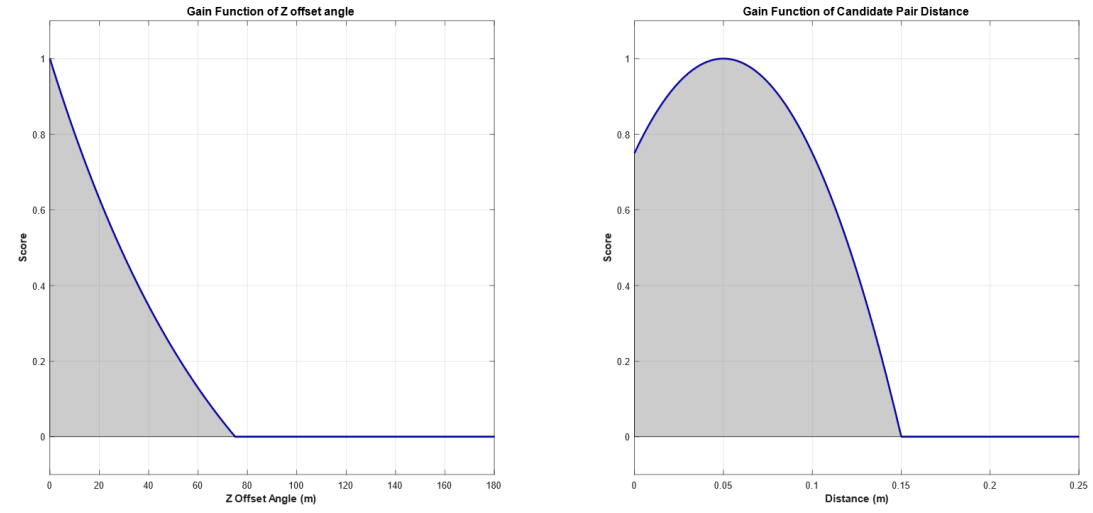


Figure 5.10 Gain function of the offset theta between p_k^i and p_{start}

The entire gain function cloud be described as the multiplication of function distance and angle, and the output of gain function is called fitness score, the detail formula shown as below:

doi:10.6342/NTU202303769

$$g(\psi, d) = \frac{e^{-\frac{\psi}{\psi_{th}}} - e^{-1}}{(1 - e^{-1})} * (-100 * (d - d_{th})^2 + 1), where \begin{cases} \psi \in [0, \psi_{th}) \\ d \in [0, d_{th} + 0.01) \end{cases}$$

Equation 5.10

The characteristic curve of the gain function shows in Figure 5.11:

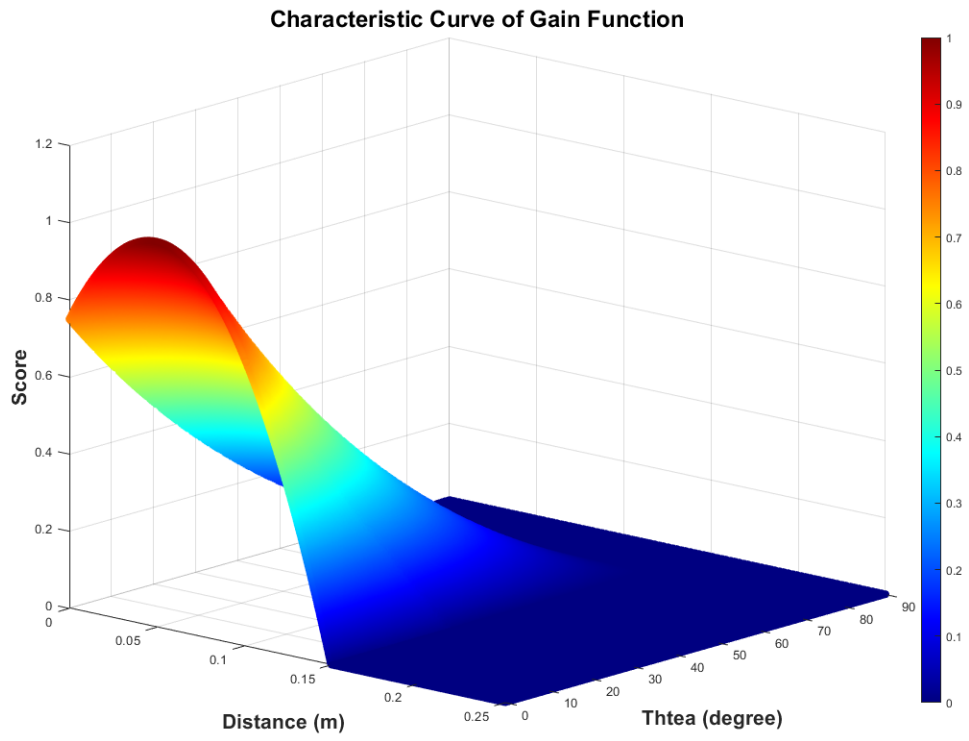


Figure 5.11 Characteristic Curve of Gain Function $G(\psi, d)$

Till now, the start searching point set $\boldsymbol{SSP}$ and the gain function $G(\psi, d)$ describing the confidence extent have been established and established.

doi:10.6342/NTU202303769

The entire pseudo-code is shown in following algorithm followed with detail introduction of the Algorithm 5.1.

Algorithm 5.1 Stem Extraction Algorithm

Input:
Reconstructed Plant Pointcloud: $P[n]$;

1: Define: ArUco Board B_i, where i $\subset$ [1,n]
2: Downsample Entire Plant $P[n]$ to $P_{d_1}[n]$
3: Downsample Lower Part of Plant $P[n]$ to $P_{d_2}[n]$, where $Z(P_{d_2}[n]) \in [0, h_{SSR}]$
4: **repeat**
5: Start Searching Point ${}^{j}p^{m}_{start} \leftarrow p^{m}_{start} \in P_{d_2}[n]$ where $j = 0$
6: Find K-Nearest Neighbour with distance $\leq d_{adj}$: $\mathbf{P}^{j}_{cand} \in P_{d_1}[n]$
7: **repeat**
8: **if** ${}^{l}P^{j}_{cand} <^{j} p^{m}_{start} \cdot \hat{z}$ **then**
9: Score ${}^{l}g_j = 0$, $\mathbf{G}^j = \mathbf{G}^j \cup^l g_j$
10: $l = l + 1$, return to line 7.
11: **end if**
12: Calculate Offset Angle ψ^l_j and Distance d^l_j
13: Compute $g(\psi, d) = g(\psi^l_j) * g(d^l_j)$
14: **if** $g(\psi, d) < g_{th}$ **then**
15: ${}^{l}g_j = 0$ $\mathbf{G}^j = \mathbf{G}^j \cup^l g_j$
16: $l = l + 1$, return to line 7.
17: **end if**
18: **until** No more ${}^{l}P^{j}_{cand}$
19: **if** $\mathbf{G}^j = \mathbf{0}$ **then**
20: No Stem Extracted, goto line XXX.
21: **end if**
22: Select ${}^{l}P^{j}_{cand}$ with maximum score ${}^{l}g_j$ and $j = j + 1$
23: **until** No More Start Searching Points p^{m}_{start}

Algorithm 5.1 Stem extraction pseudo algorithm

Assumed that the size of set $\boldsymbol{SSP}$ is M, and for any start searching point p^{m}_{start}, an iterative stem feature extraction approach is proposed as follows:

1. For each start searching point $p^{m}_{start} \in P_{d_2}[n]$ $where\ m = 0$, initialize stem feature pointcloud $\boldsymbol{P}^{m}_{stem}$. The search point defines as ${}^{j}p^{m}_{start} = p^{m}_{start}$, $where\ j$ is the iteration repeat number initialized as 0.

doi:10.6342/NTU202303769

2. Find the nearest k-th candidate points $\boldsymbol{P}_{cand}^{j} \in \boldsymbol{P}_{d}[\mathrm{n}]$ around p_{start}^{m} using K-D Tree with the maximum distance d_{adj} and initialize a container $\mathbf{G^j}$.

3. For each candidate stem point ${}^{l}\boldsymbol{p}_{cand}^{j} \in \boldsymbol{P}_{cand}^{j}, l \in N$, where l is the index of $\boldsymbol{P}_{cand}^{j}$:

   a. If ${}^{l}\boldsymbol{p}_{cand}^{j}\hat{z} < {}^{j}\boldsymbol{p}_{start}^{m}\ \hat{z}$

      i. Fitness Score ${}^{l}g_{j} = 0,\ \mathbf{G^j} = \mathbf{G^j} \cup \{\ {}^{l}g_{j}\ \}$

      ii. $l = l + 1$, and return to step 3

   b. Calculate offset angle ψ_j^l and Euclidean distance d_j^l

   c. Calculate each angle and distance gain: $g(\psi_j^l)$ and $g(d_j^l)$

   d. Compute $g(\psi, d)$

      i. If $g(\psi, d) < g_{th},\ g = 0$

      ii. $l = l + 1$, and save fitness score: $\mathbf{G^j} = \mathbf{G^j} \cup \{\ {}^{l}g_{j}\ \}$

4. Next stem feature point selection:

   a. If all value ${}^{l}g_{j} = 0,\ where\ (g_l^j \in \mathbf{G^j})$, go to step 5.

   b. Select the candidate point ${}^{l}\boldsymbol{p}_{cand}^{j}$ with maximum fitness score ${}^{l}g_{j}$ in $\mathbf{G^j}$ as ${}^{j+1}p_{start}^{m}$. Add ${}^{l}\boldsymbol{P}_{cand}^{j}$ to $\boldsymbol{P}_{stem}^{m},\ j = j + 1$, and return to step 2.

doi:10.6342/NTU202303769

5. End of sub-iteration, output stem cloud $\boldsymbol{P}_{stem}^{m}$ of m-th start point p_s^m, and $m = m + 1$. If $m < N_{SSP}$, return to step 1.

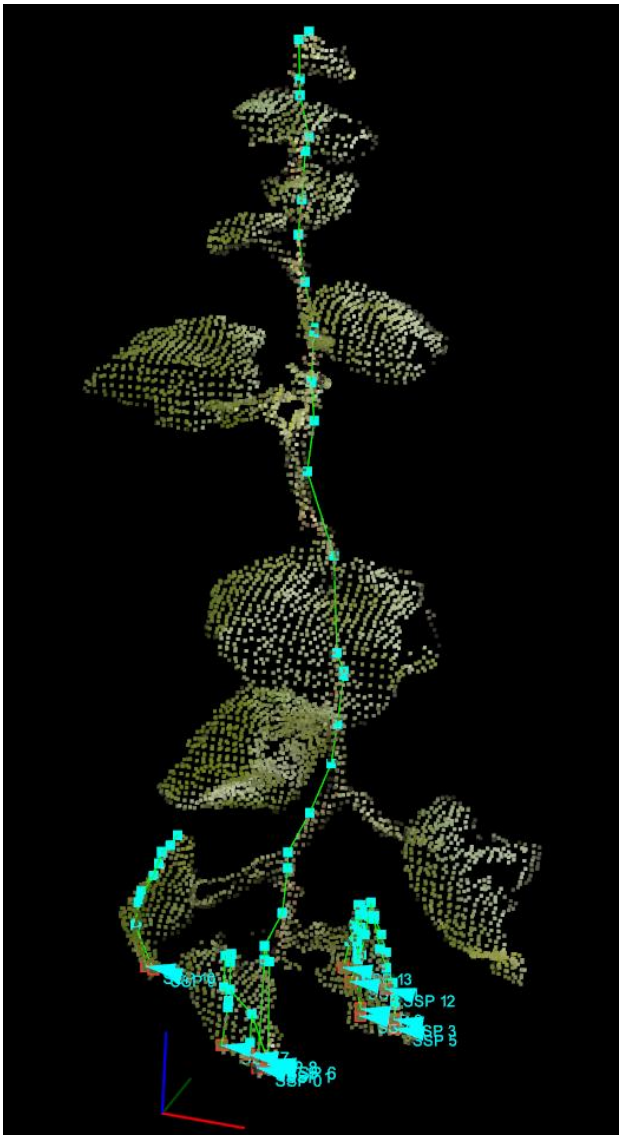


Figure 5.12 Computed stem candidate by SEA

doi:10.6342/NTU202303769

The extracted stem candidates $\boldsymbol{P}_{stem}^{cand}$ have been shown in Figure 5.12, for this case, thirteen possible stem candidates computed by iterative geometric stem extraction algorithm.

After achieving these stem candidates, the algorithm has to interpolate the cloud between each neighbor points. This is because the gain function does not usually select the nearest next point. Some features may be omitted by the previous SE algorithm shown as Figure 6.9. Therefore, a perpendicular distance-based interpolation approach has been applied as a post-processing technique to densify the outcome stem cloud.

Until now, the dense entire plant pointcloud $P_d[\mathrm{n}]$ and extracted stem pointcloud $\boldsymbol{P}_{stem}$ is given. For each two consecutive point pair, $\boldsymbol{p}_{stem}^{i}$ and $\boldsymbol{p}_{stem}^{j}$ belonging to $\boldsymbol{P}_{stem}$ and any point $\boldsymbol{p}_{d}^{k} \in P_d[\mathrm{n}]$, the distance D between $\boldsymbol{p}_{d}^{k}$ and the line passing through $\boldsymbol{p}_{stem}^{i}$ and $\boldsymbol{p}_{stem}^{j}$ represent as follows [89 Perpendicular Distance]:

$D = \dfrac{\left\| \overrightarrow{\boldsymbol{p}_{stem}^{i}\boldsymbol{p}_{d}^{k}} \times \vec{d} \right\|}{\left\| \vec{d} \right\|}$	Equation 5.11

$\overrightarrow{l_{ij}} = \overrightarrow{\boldsymbol{p}_{stem}^{i}\boldsymbol{p}_{stem}^{j}} = \begin{bmatrix} \hat{x}\,\boldsymbol{p}_{stem}^{i} - \hat{x}\,\boldsymbol{p}_{stem}^{j} \\ \hat{y}\,\boldsymbol{p}_{stem}^{i} - \hat{y}\,\boldsymbol{p}_{stem}^{j} \\ \hat{z}\,\boldsymbol{p}_{stem}^{i} - \hat{z}\,\boldsymbol{p}_{stem}^{j} \end{bmatrix}^{T}$	Equation 5.12

where $\vec{d}$ is the direction vector of the line.

Assumed that $\overrightarrow{\boldsymbol{p}_{stem}^{i}\boldsymbol{p}_{d}^{k}} = [a\ b\ c]$ and $\overrightarrow{l_{ij}} = [d\ e\ f]$, the cross product of $\overrightarrow{\boldsymbol{p}_{stem}^{i}\boldsymbol{p}_{d}^{k}}$ and $\vec{d}$ shows as follows:

doi:10.6342/NTU202303769

$$\overrightarrow{\boldsymbol{p}_{stem}^{i}\boldsymbol{p}_{d}^{k}} \times \overrightarrow{l_{ij}} = \begin{bmatrix} 0 & -c & b \\ c & 0 & -a \\ -b & a & 0 \end{bmatrix} * \begin{bmatrix} d \\ e \\ f \end{bmatrix} * \begin{bmatrix} \boldsymbol{i} \\ \boldsymbol{j} \\ \boldsymbol{k} \end{bmatrix}^{T}$$

$$= (bf - ce)\boldsymbol{i} + (cd - af)\boldsymbol{j} + (ae - bd)\boldsymbol{k} \qquad \text{Equation 5.13}$$

where $\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}$ are unit vector of $\boldsymbol{x}, \boldsymbol{y}, and\ \boldsymbol{z}$.

Hence, the distance between $\boldsymbol{p}_{stem}$ and 3-dimensional line clearly computes as follows [96] :

$$D = \frac{\left\| \overrightarrow{\boldsymbol{p}_{stem}^{i}\boldsymbol{p}_{d}^{k}} \times \overrightarrow{l_{ij}} \right\|}{\left\| \overrightarrow{l_{ij}} \right\|}$$

$$= \frac{\sqrt{(bf - ce)^2 + (cd - af)^2 + (ae - bd)^2}}{\sqrt{d^2 + e^2 + f^2}} \qquad \text{Equation 5.14}$$

The interpolation algorithm procedure shows as follows:

1. Initialize pointcloud $\boldsymbol{P}_{stem,dense}$
2. For each point pair $\boldsymbol{p}_{stem}^{i}, \boldsymbol{p}_{stem}^{j} \in \boldsymbol{P}_{stem}$
    a. For each point $\boldsymbol{p}_{stem}^{i} \in P_d$[n]
    b. Calculate perpendicular distance D
    c. If $D < d_{th}^{ip}$, $\boldsymbol{P}_{stem,dense} = \boldsymbol{P}_{stem,dense} \cup \boldsymbol{p}_{stem}^{i}$

The following Figure 5.13 shows the result of perpendicular distance-based interpolation which the information omitted by SE algorithm has been found, where the

doi:10.6342/NTU202303769

cyan-blue and magenta colors represent the result of solely SE algorithm and post-processed interpolation.

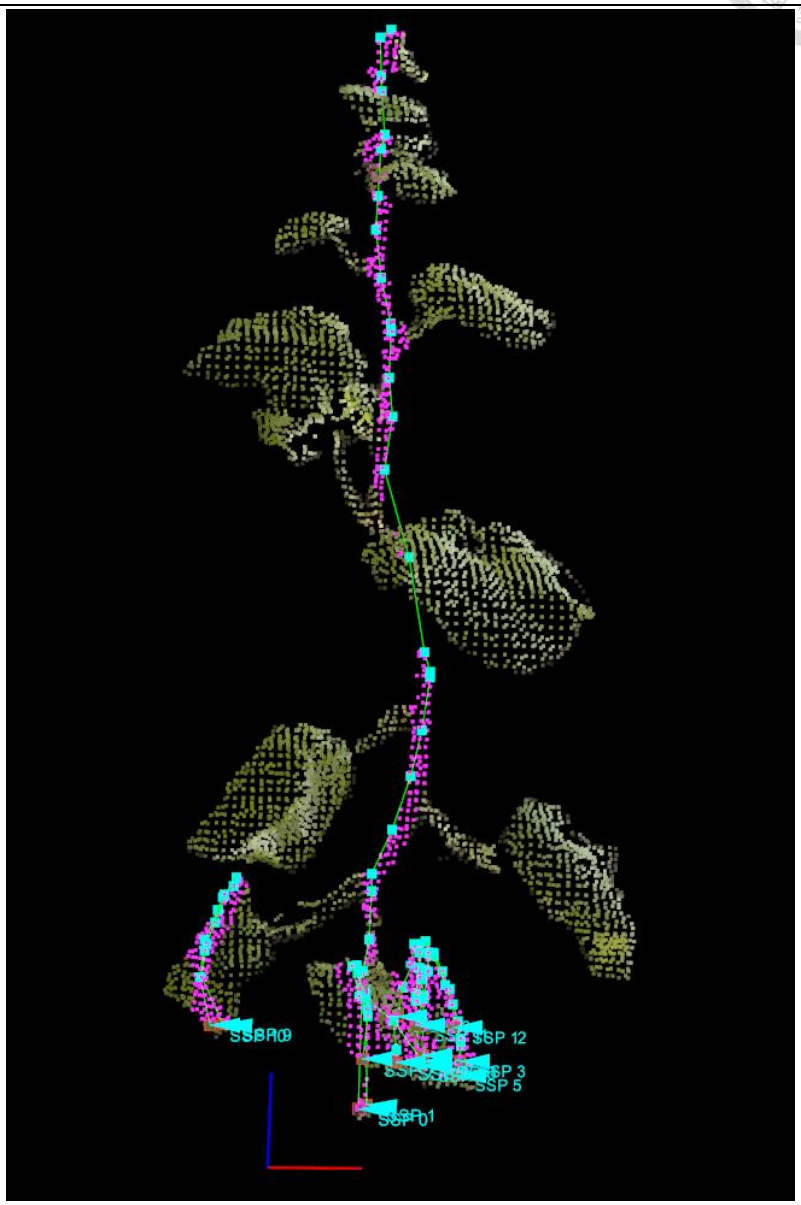


Figure 5.13 The result stem cloud after perpendicular distance interpolation

After the program gets these densifies stem candidates, it needs to automatically select which stem is the true plant stem and which stems are misjudgment results.

doi:10.6342/NTU202303769

Obviously, the SE algorithm cloud utilizes the geometric distribution and stem point cloud size to evaluate the trustworthiness of each stem candidate.

Empirically, the size of extracted stem candidates is significantly varying, namely, large deviation, selecting the candidate with the largest cloud size is theoretical acceptable. However, without loss of generality, the program also analyzes each candidate's growing distribution, namely stem cultivating direction, to exclude that of large angles which tilts from z axis. The reason why the stem candidates contain a large z tilt/ offset angle is because the SE algorithm only chooses the point with best score and the error will accumulate if the start searching point is wrong.

To solve the misjudgment problem, the stem growing direction should clearly upright under highly manual intervention. The principal component analysis (PCA) cloud carry through the 3-dimensional distribution extraction tasks. According the algorithm introduced in Section 3.5, the principal pointcloud distribution direction is computed. The eigenvector from PCA contains the projection information of extracted stem candidate with respect to standard reconstructed pointcloud frame. The eigenvector shows as follows:

$$ {}^{A}_{B}X = \begin{bmatrix} {}^{A}\hat{X}_B & {}^{A}\hat{Y}_B & {}^{A}\hat{Z}_B \end{bmatrix} = \begin{bmatrix} \hat{X}_B * \hat{X}_A & \hat{Y}_B * \hat{X}_A & \hat{Z}_B * \hat{X}_A \\ \hat{X}_B * \hat{Y}_A & \hat{Y}_B * \hat{Y}_A & \hat{Z}_B * \hat{Y}_A \\ \hat{X}_B * \hat{Z}_A & \hat{Y}_B * \hat{Z}_A & \hat{Z}_B * \hat{Z}_A \end{bmatrix} $$ Equation 5.15

where A represent the plant coordination frame and that of B is stem candidate pointcloud PCA frame.

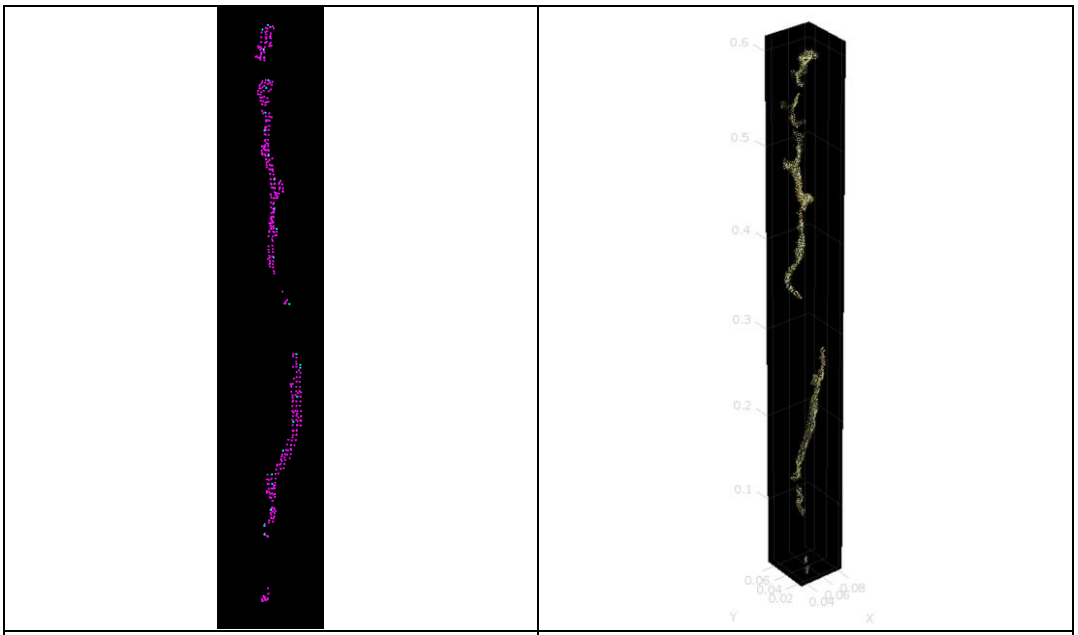
doi:10.6342/NTU202303769

One them that the algorithm only needs to concern is the third element of ${}^{A}\hat{Z}_B$, that is $\hat{Z}_B * \hat{Z}_A$. This term represents the projection of z axis of principal component of extracted candidate pointcloud to plant frame. After that, this program calculates the confidence score $\boldsymbol{S}_{stem}$ of each candidate stem pointcloud $\boldsymbol{P}_{stem}^{m} \in \boldsymbol{P}_{stem}^{cand}$, where the size of stem pointcloud is $\boldsymbol{N}_{stem}^{m}$:

$$\boldsymbol{S}_{stem}^{m} = \boldsymbol{N}_{stem}^{m} * \hat{Z}_B * \hat{Z}_A * \eta_{sp} \qquad \text{Equation 5.16}$$

where the η_{sp} is the stem plant ratio, the maximum value of the ratio is 1.

By Selecting the best stem $\boldsymbol{P}_{stem}$ with the largest confidence score, the misjudged results which are in wrong direction or not optimized in pointcloud size will be filtered. The followed Figure 5.14 shows the result after the program selects the best result.



(a) The downsampled raw result

(b) The post-processed extened stem pointcloud

Figure 5.14 Extracted stem cloud

doi:10.6342/NTU202303769

5.4 SEA-Based Occluding Checking Module

Although the reliable stem pointcloud was generated from SE Algorithm, due to the plant cultivating environment difference, solar illumination amount, and biological diversity, the distribution of leaves and stem may vary significantly. The design of the SEA is only considering the best condition; that is, the input pointcloud is intact, and the stem pointcloud is continuous. The system will encounter an occluding condition of the stem pointcloud because the leaves are in front of the observed stem or the stem is under a high density of brightness that no valid features are extracted by the RGBD Camera. One of the important purposes of this system is to segment different types of plant organs from a reconstructed pointcloud. Without consideration of sensor error and drawbacks, an algorithm that assigned a different size of d_{adj}, a parameter in the SE algorithm limiting the maximum distance of the candidate point pair, could compensatively find an optimized stem pointcloud. Also, according to the occluding region, a series of new viewpoints will be utilized to reconstruct the occluded part. The intact ideal plant pointcloud and stem-occluded pointcloud are shown in Figure 5.15. Also, the procedure of occluding checking is iteratively set different size of d_{adj}, execute SEA, checking stem plant height ratio. This module first initializes the input pointcloud; according to the empirical observation data, the maximum occluded stem distance caused by a single head-

doi:10.6342/NTU202303769

on leaf is not exceeded 10 centimeters. The result of SEA may not be a reference when fewer candidate point pairs are found with too large a distance. The initial d_{adj} is set as 2cm instead of a smaller value because under this condition, a large amount of candidate pairs is found if the pointcloud is intact.

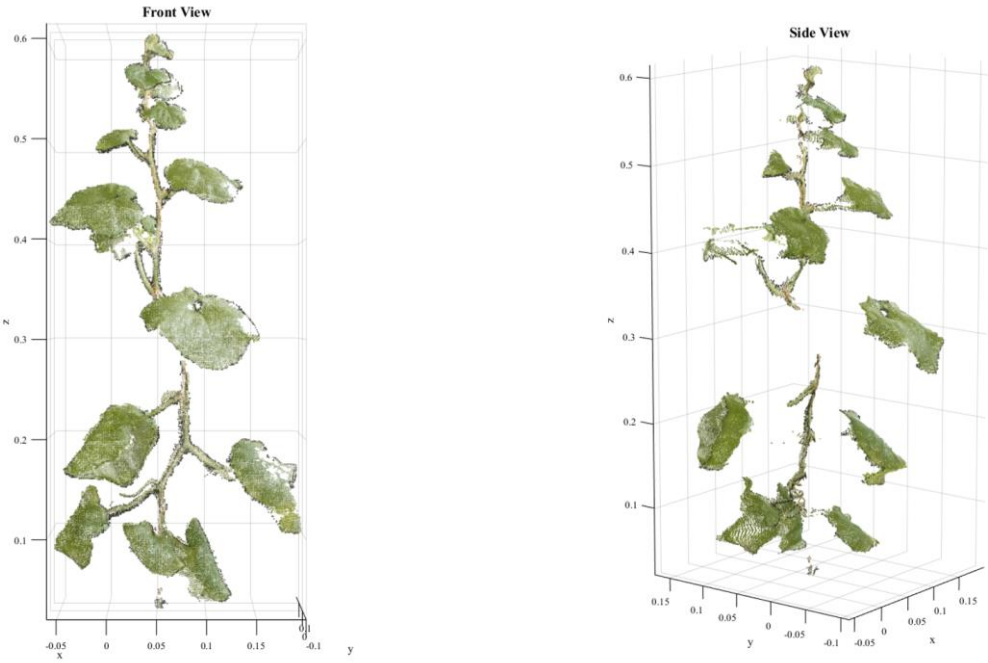


Figure 5.15 pointcloud with incomplete stem

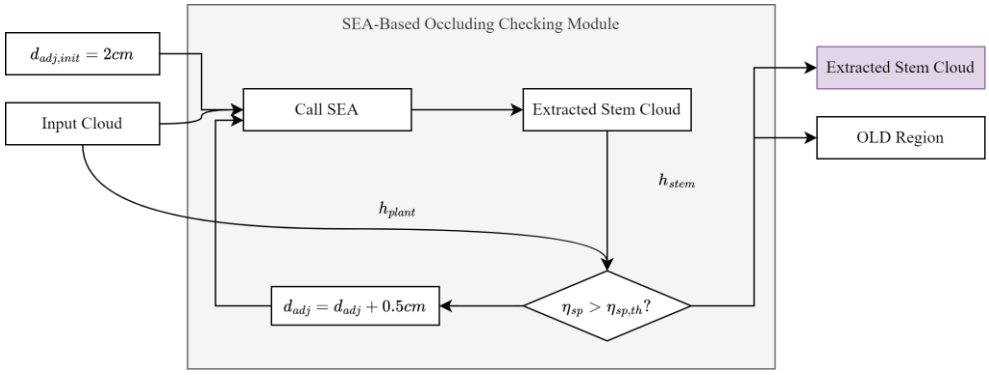


Figure 5.16 SEA-based occluding checking

doi:10.6342/NTU202303769

After the SEA generates the stem pointcloud P_{stem}, the occluding checking module first check the stem height and plant height ratio. The height of both pointcloud is calculated as follows:

$h_{plant} = \max_z P[n]$	Equation 5.17

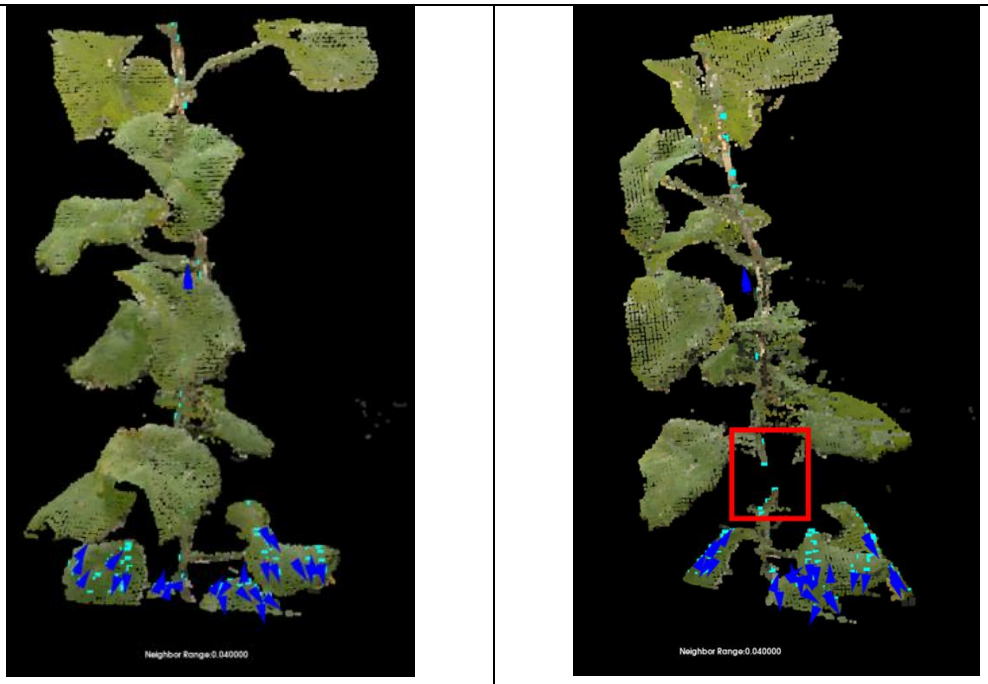
$h_{stem} = \max_z P_{stem}[n]$	Equation 5.18

The stem plant ratio is calculated using previous equation, and $\eta_{sp} = \frac{h_{stem}}{h_{plant}}$. If the ratio η_{sp} exceeds reference threshold ratio $\eta_{sp,th}$, then the stem result from SEA is considered as acceptable, or increase d_{adj} by 0.5 cm and execute SEA again. The $\eta_{sp,th}$ is set as 0.85, which is also an empirical value. The following figure represent the d_{adj} from 0.02m to 0.035m, it shows no valid stem has been found because of the front leaf.

Figure 5.17 SEA under different d_{adj} from 2cm to 3.5cm

doi:10.6342/NTU202303769

After increasing the search radius d_{adj}, the stem was then extracted in Figure 5.13. The cyan-blue color represents the extracted stem, and the blue arrows are results of principal component analysis. The highlighted red box represents the occluded region, namely the region does not contain stem pointcloud.



(a) (b)

Figure 5.18 The extracted stem after increasing d_{adj} to 4cm
a) Front view, b) Side view (occluded region highlighted)

If the stem result is valid, the occluding checking algorithm then calculate the occluding region using statistical analysis. The following figure shows an example of stem pointcloud with incomplete region. This algorithm abort x-axis and y-axis information and generate a histogram of stem pointcloud exclusively using z-axis data.

Specifically, the bin width w_b of histogram has been set as 0.01m. Assumed that the bin size is N_b and stem point size is $N_{P,stem}$, the average value of this histogram is:

doi:10.6342/NTU202303769

$\mu_{hist} = \frac{N_{P,stem}}{N_b}$	Equation 5.19

Also, the standard derivation is calculated, where n_b^i is height of i-th bin:

$\sigma_{hist} = \sqrt{\frac{1}{N_b}\sum_{i=0}^{N_b}\left(n_b^i - \mu_{hist}\right)^2}$	Equation 5.20

For any bin with height satisfying the following equation is labelled as occluding region:

$n_{occ} = \mu_{hist} - \sigma_{hist}$	Equation 5.21

An example of an imperfect stem with occluded region pass through occluding checking algorithm, the histogram of z-axis distribution and corresponding stem pointcloud are illustrated in Figure 5.19:

doi:10.6342/NTU202303769

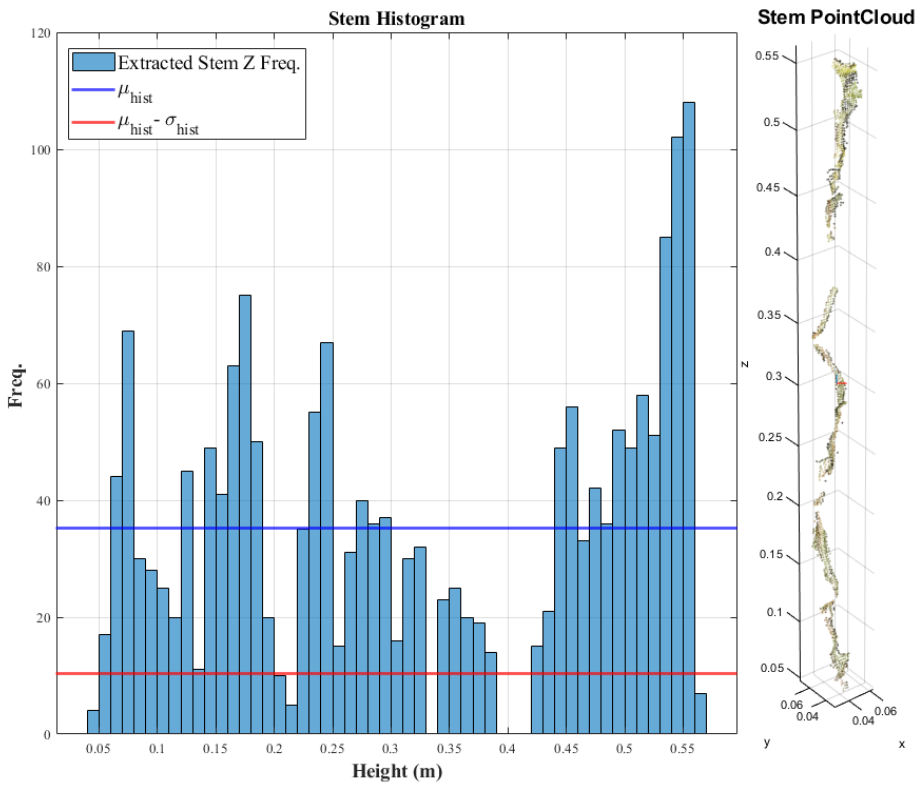


Figure 5.19 Imperfect stem pointcloud occluding analysis

doi:10.6342/NTU202303769

5.5 Cylinder viewpoint planning

This algorithm provides a set of candidate viewpoints given several variables. The input variables of this algorithm are shown as follows:

Table 5.1 Cylinder viewpoint generator input variables

Variable No.	Mathematically Symbol	Description
1	N_{layer}	The layer size
2	n_{layer}	The point size in one layer
3	r_{obv}	Observation radius
4	h_{start}	Start height of missing region
5	h_{end}	End height of missing region
6	θ_{start}	Start angle of planned viewpoints
7	θ_{end}	End angle of planned viewpoints

The value of h_{start} and h_{end} is assigned from z-axis value of the point p_o^k and p_u^k, respectively. This algorithm will plan a cylinder like viewpoints sets with one or more vertical layers N_{layer} and each layer has n_{layer} viewpoints. The observation radius is distance between each candidate and the z-axis of the plant coordination $\{\mathcal{P}_i\}$. The ψ_{start} and ψ_{end} represents the start and end angle of planned viewpoints around the z-axis. Merging the previous mentioned variables, the planned viewpoint is located on the surface described as follows:

doi:10.6342/NTU202303769

Table 5.2 Planned viewpoint in different representation coordination	
Cartesian Space	Polar Space
$x = r_{obv}\cos\theta$ $y = r_{obv}\sin\theta$ $z = h \in [h_{start}, h_{end}]$	$r_{obv} = \sqrt{x^2 + y^2}$ $\theta = tan^{-1}\left(\frac{y}{x}\right) \in [\theta_{start}, \theta_{end}]$ $z = h \in [h_{start}, h_{end}]$

In the left part of the table, the planned point in cartesian space is defined. Meanwhile, the angle θ is a set containing several viewpoint angles. The size of set $\boldsymbol{\theta}$ is $n_{layer} + 2$, which the extra two points locates at the boundaries of the cylinder. For example, if $n_{layer} = 1$, there will be only one viewpoint located on the angle $\theta_1 = \frac{(\theta_{end} - \theta_{start})}{2}$. Therefore, extremum of θ is calculated in the following algorithms:

$\theta_0 = \theta_{start}$	Equation 5.22

$\theta_{(n_{layer}+1)} = \theta_{end}$	Equation 5.23

Also, the angle interval of each candidate viewpoint θ_{gap} is defined as；

$\theta_{gap} = \frac{\lvert\theta_{end} - \theta_{start}\rvert}{n_{layer} + 1}$	Equation 5.24

The angle inside the upper and lower boundaries are shown as follows:

$\theta_l = l * \theta_{gap} + \theta_{start}, l \in [1, n_{layer} + 1]$	Equation 5.25

doi:10.6342/NTU202303769

Similarly, the height interval and the observed height are:

$$h_{gap} = \frac{(h_{end} - h_{start})}{N_{layer} + 1}$$	Equation 5.26

$$h_{m-1} = h_{start} + m * h_{gap}, m \in [1, N_{layer} + 1]$$	Equation 5.27

From these equations, the position of the candidate viewpoints is known:

$$p_{vp}^{i} = (r_{obv}\, cos\, \theta_l\, , r_{obv} sin\theta_l, h_{m-1})$$	Equation 5.28

Also, the orientation of the candidate viewpoints is generated from three rotational transformation part, which from plant coordination to camera coordination T_C^{Plant}, camera horizontal tilt and camera vertical tilt transformations, $T_{tilt,h}$ and $T_{tilt,v}$.

$$R_{Cylinder}^{\mathcal{P}_i}[i, \theta_{tilt}, \Phi_{\text{tilt}}] = R_x(-90°) * R_y(\theta_{tilt}) * R_x(\psi_{\text{tilt}})$$ $$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}$$ $$* \begin{bmatrix} \cos(\theta_{tilt}[i]) & 0 & \sin(\theta_{tilt}[i]) \, 0 \\ 0 & 0 & 1 \\ -\sin(\theta_{tilt}[i]) & -1 & \cos(\theta_{tilt}[i]) \end{bmatrix}$$ $$* \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\psi_{\text{tilt}}[i]) & -\sin(\psi_{\text{tilt}}[i]) \\ 0 & \sin(\psi_{\text{tilt}}[i]) & \cos(\psi_{\text{tilt}}[i]) \end{bmatrix}$$	Equation 5.29

The transformation of any viewpoint could represent as:

$$T_{Cylinder}^{\mathcal{P}_i} = \begin{bmatrix} R_C^{\mathcal{P}_i}[i] & p_{vp}^{i\ \prime} \\ 0 & 1 \end{bmatrix}$$	Equation 5.30

doi:10.6342/NTU202303769

If more than a layer viewpoint is desired, the determination of tilt angle $\Phi_{\text{tilt}}[i]$ is necessary. The algorithm is designed to plan the viewpoint which forces the camera to stare at the central point of the occluding region. The tilt angle defined as:

$$\psi_{\text{tilt}}[i] = -\arctan\frac{h_{m-1} - \frac{\left(h_l^k + h_u^k\right)}{2}}{r_{obv}}$$	Equation 5.31

The following Figure 5.20 shows the planned viewpoint using the real-world plant as reference.

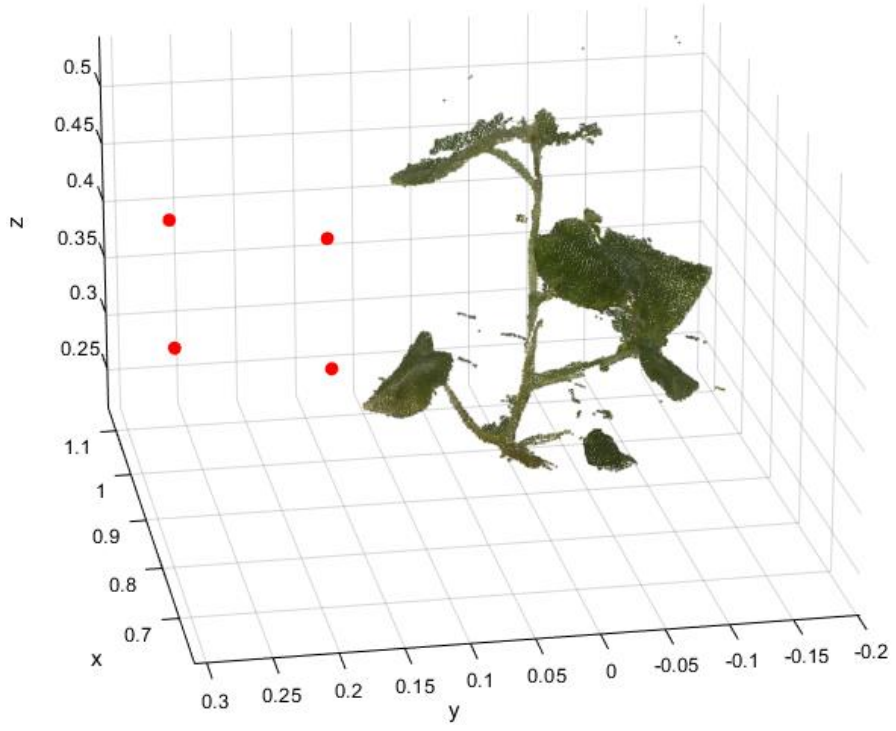


Figure 5.20 Selected circular viewpoints

doi:10.6342/NTU202303769

5.6 Occluding-Free Viewpoint optimizer

After generating these viewpoints, directly moving camera to all viewpoint is not time-efficient. The pointcloud captured from new moved viewpoint has possibility of occluding condition. Brute force moving camera in this case may work but wastes more time. A perpendicular-distance-based occluding-free viewpoint optimizer for planned viewpoint has been purposed. This algorithm uses the perpendicular distance between each points of plant to viewpoint line (a line from viewpoint to occluding region) to evaluate the blocking condition of new planned viewpoint.

Given the plant pointcloud P_{merged} reconstructed in stage 1, the stem pointcloud P_{stem} and occluded region $R_{occ}^{k} \in \mathbb{R}^{2\times1}$ extracted from SEA-Based occluding algorithm, and the planned transformation matrix set of new viewpoints $\boldsymbol{T}_{C}^{Plant}$, the occluding-free algorithm then analyze each viewpoint starting from extracting the lower and higher boundary points p_A and p_B, respectively. The lower boundary point p_A is the centroid of the regional pointcloud P_{stem}^{low}, where

$P_{stem}^{low} = P_{stem}, z \in \left[h_l^k - 0.01, h_l^k\ \right]$	Equation 5.32

Also, the lower boundary point is the centroid of P_{stem}^{low}:

$p_A = \frac{1}{\mathrm{N}\left(P_{stem}^{low}\right)} \sum p_{stem}^{low}$	Equation 5.33

doi:10.6342/NTU202303769

Similarly, the regional pointcloud of upper point and the centroid of that cloud is defined as:

$P_{stem}^{upper} = P_{stem}, z \in [h_u^k, h_u^k + 0.01\ ]$	Equation 5.34

And

$p_B = \frac{1}{\mathrm{N}(P_{stem}^{upper})} \sum p_{stem}^{upper}$	Equation 5.35

To increase the accuracy, the middle point of p_A and p_B defined as p_C is:

$p_C = \frac{p_A + p_B}{2}$	Equation 5.36

Meanwhile, the value of viewpoint p_D is assigned from the translation part of the transformation matrix T_C^{Plant}. The algorithm then establishes three 3-dimensional line using p_A, p_B, p_C and p_D. Three lines shows below:

$\overrightarrow{l_{AD}} = \overrightarrow{p_A p_D}$ $\overrightarrow{l_{BD}} = \overrightarrow{p_B p_D}$ $\overrightarrow{l_{CD}} = \overrightarrow{p_C p_D}$	Equation 5.37

For each point p_{merged}^i in pointcloud P_{merged}, if the perpendicular distance between p_{merged}^i and $\overrightarrow{l_{AD}}$, $\overrightarrow{l_{BD}}$ or $\overrightarrow{l_{CD}}$ is less than specific value, example 0.5cm, this

doi:10.6342/NTU202303769

point is labelled as a blocking point in this viewpoint. The following figure shows the how this algorithm works in real plant:

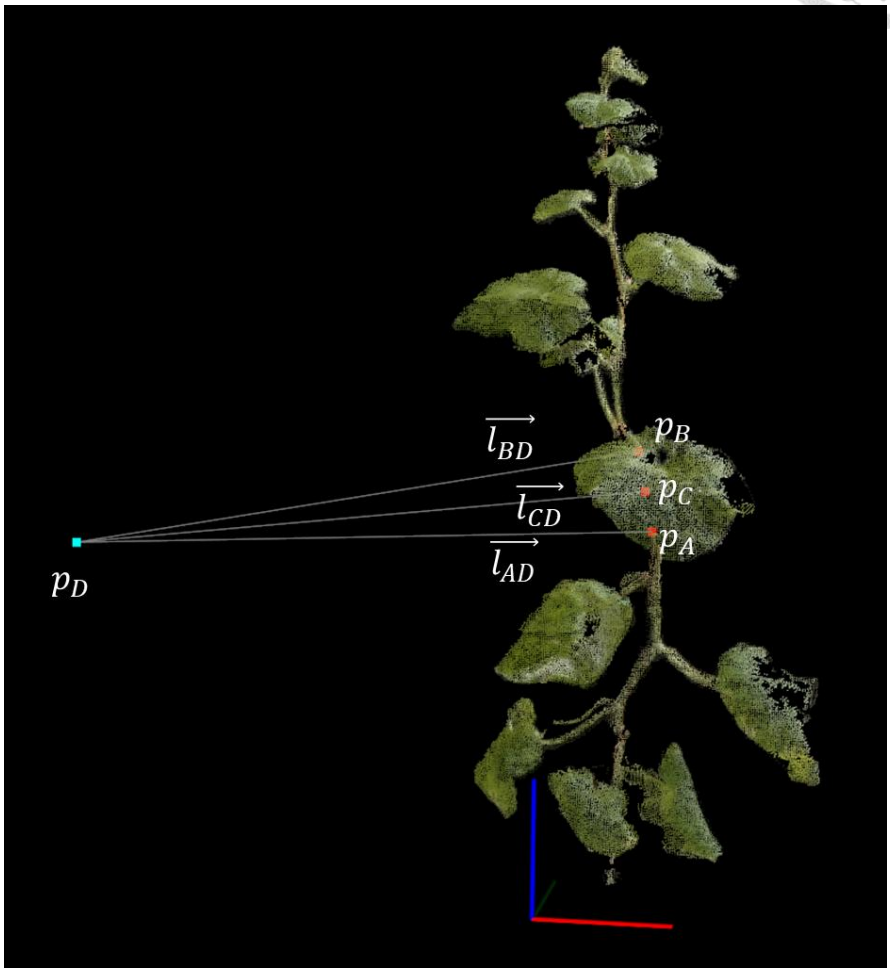


Figure 5.21 Perpendicular-based viewpoint optimization

The boundary point and middle point p_A, p_B, p_C rendered as red point. The blue point is the planned viewpoint. Three 3D line also drew in this figure. In this case, few points are labelled as blocked points, which means the camera under this viewpoint could capture more missing detail without leaf occluding. The following figure shows the

doi:10.6342/NTU202303769

opposite condition, the leaf is in front of the planned viewpoint and the intersection region also labelled as red.

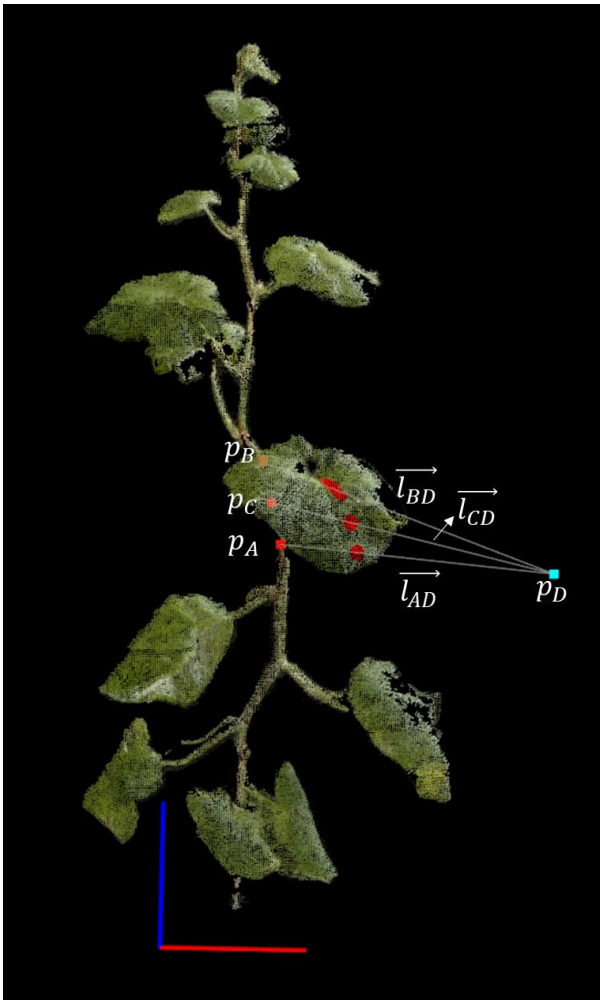


Figure 5.22 Perpendicular-based viewpoint optimization (With blocking)

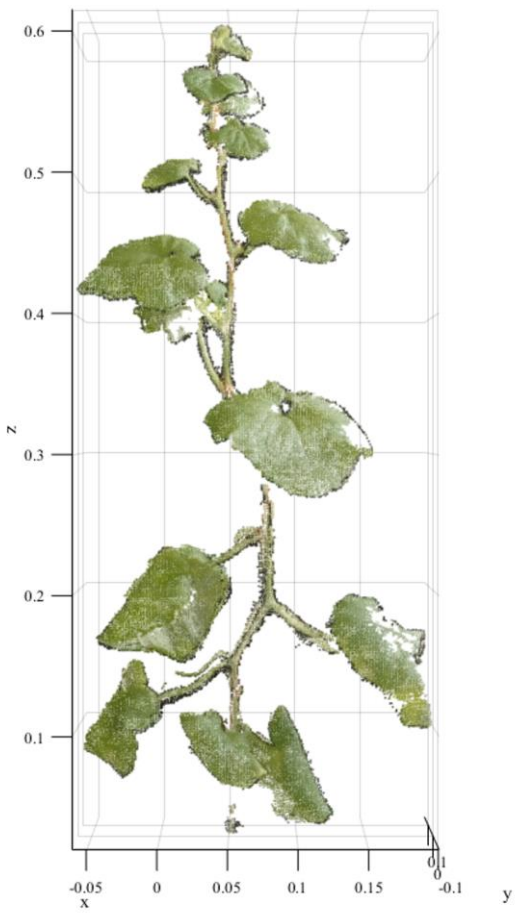
Iteratively running this algorithm for all candidate viewpoint, the blocking Index (BI) is introduced to describe how difficult the camera could capture correct missing part. Simply, the value of BI is the total number of points nearby the $\overrightarrow{l_{AD}}$, $\overrightarrow{l_{BD}}$ and $\overrightarrow{l_{CD}}$. And then, the candidate viewpoint with smaller BI is chosen for future pointcloud refinement procedure.

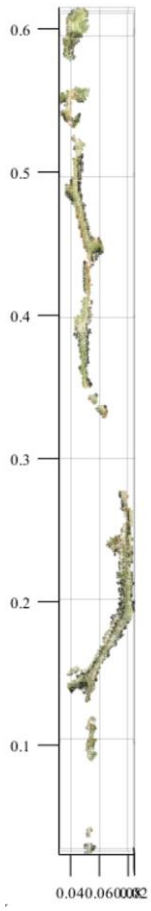
doi:10.6342/NTU202303769

5.7 Stem-Based Plant Pointcloud Segmentation

One purpose of this paper is to segment different part of plants. The paper [95 William 2017] shows the intuition and concept how to segment the plant. If the pointcloud is dedicatedly reconstructed, the node (connect the main stem and leaf) and leaves will be segmented using the extracted stem pointcloud.

The input pointcloud should have distinctive stem features, commonly shown in Figure 5.24 (a), and the stem should also give using SEA.





(a) (b)

Figure 5.23 Input pointcloud of plant segmentation algorithm

doi:10.6342/NTU202303769

Two radii $r_{pet,l}$ and $r_{pet,h}$ are defined which represents the distance petiole size. Also, assumed that the original pointcloud in Figure 5.16 (a) is P and (b) is P_{stem}, the following Algorithm 5.3 shows how segment procedure works. The segmented leaf pointcloud and petiole pointcloud shows in the following Algorithm 5.2.

Algorithm 5.3 Plant Organ Segmentation Algorithm

Input:

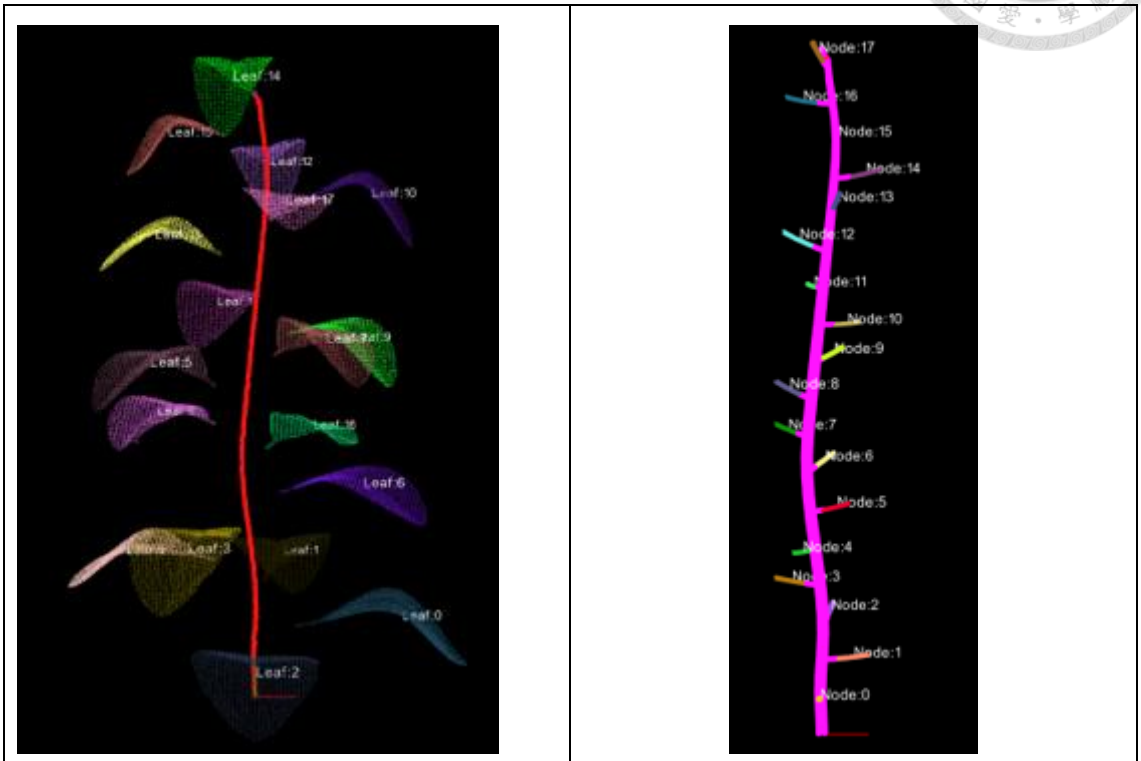
Reconstructed Plant Pointcloud: $P[n]$;
Stem Pointcloud: P_{stem}

Initialize Petiole and Leaf **NOT Repeating** Pointcloud Container: $P_{petiole}$ and P_{leaf}
Calculate the point cloud without stem $P_{ws} = P[n] - P_{stem}$
Initialize stem index: $i = 0$
repeat
    $p = P_{stem}[i]$, Find All Neighbours $P_{pet,i} \in P_{ws}$ with $r_{pet,l} \leq d \leq r_{pet,u}$
    $P_{petiole} = P_{petiole} \cup P_{pet,i}$
    $i = i + 1$
until $i < size(P_{stem})$
$P_{leaf} = P_{ws} - P_{petiole}$
Initialize leaf and petiole point cloud list: $\mathbf{P}_{leaf}$ and $\mathbf{P}_{petiole}$
Apply Euclidean Cluster to $P_{petiole}$ and P_{leaf}.
$\mathbf{P}_{leaf} = \mathbf{ECE}(P_{leaf})$
$\mathbf{P}_{petiole} = \mathbf{ECE}(P_{petiole})$

Algorithm 5.2 Plant organ segmentation algorithm

doi:10.6342/NTU202303769

Also, for ideal plant pointcloud, this algorithm also cloud extract leaves and petioles shows as follows:



(a) Leaf (b) Petiole

Figure 5.24 Segmented leaf and petiole pointcloud of an ideal plant

doi:10.6342/NTU202303769

Chapter 6
Visual-Based Plant Reconstruction (VBPR) Method

In this chapter, we purposed a procedure to localize target plant, roughly plant detection and reconstruction, stem-based plant viewpoints re-planning, plant parts segmentation. In the Section 6.1, the purposed procedure will be introduced generally, including system block diagram and structure. This algorithm has been divided into two main stages. The fiducial-based plant growth position and orientation localization and fast plant pointcloud reconstruction is introduced in Section 6.2, namely first observation stage. After achieving the roughly constructed pointcloud, the procedure then analyzes the intactness of stem using stem extraction algorithm (SEA). If there has occluding region in stem pointcloud, the robotic system will then move to these new planned viewpoints around the occluding region, introduced in Section 6.3.

doi:10.6342/NTU202303769

6.1 VBPR System Architecture

In this system, we purposed an integrated robust plant reconstruction procedure. Firstly, the robotic scans the environment by rotating itself to detect ArUco marker beside the plant, which provides the plant id and pre-defined transformation with respect to this plant. Secondly, the robotic system reconstructs plant pointcloud using multiple fixed head-on viewpoints, automatically detect the plant height and height growth rate, and stop first stage when this growth rate is lower than a specific value. The occluding checking module then operated to scan any occluding parts in stem cloud. If there has occluded region along the detected stem, the system will plan a circular viewpoint set to re-capture and re-align the target plant region. And then, the algorithm segments the entire plant pointcloud using extracted stem cloud. Individual leaves and petioles have been separated. The algorithm analyses the normal direction for each incomplete leaf, re-planning optimized viewpoint to refine and reconstruct the entire pointcloud. The following graph illustrates this purposed procedure:

doi:10.6342/NTU202303769

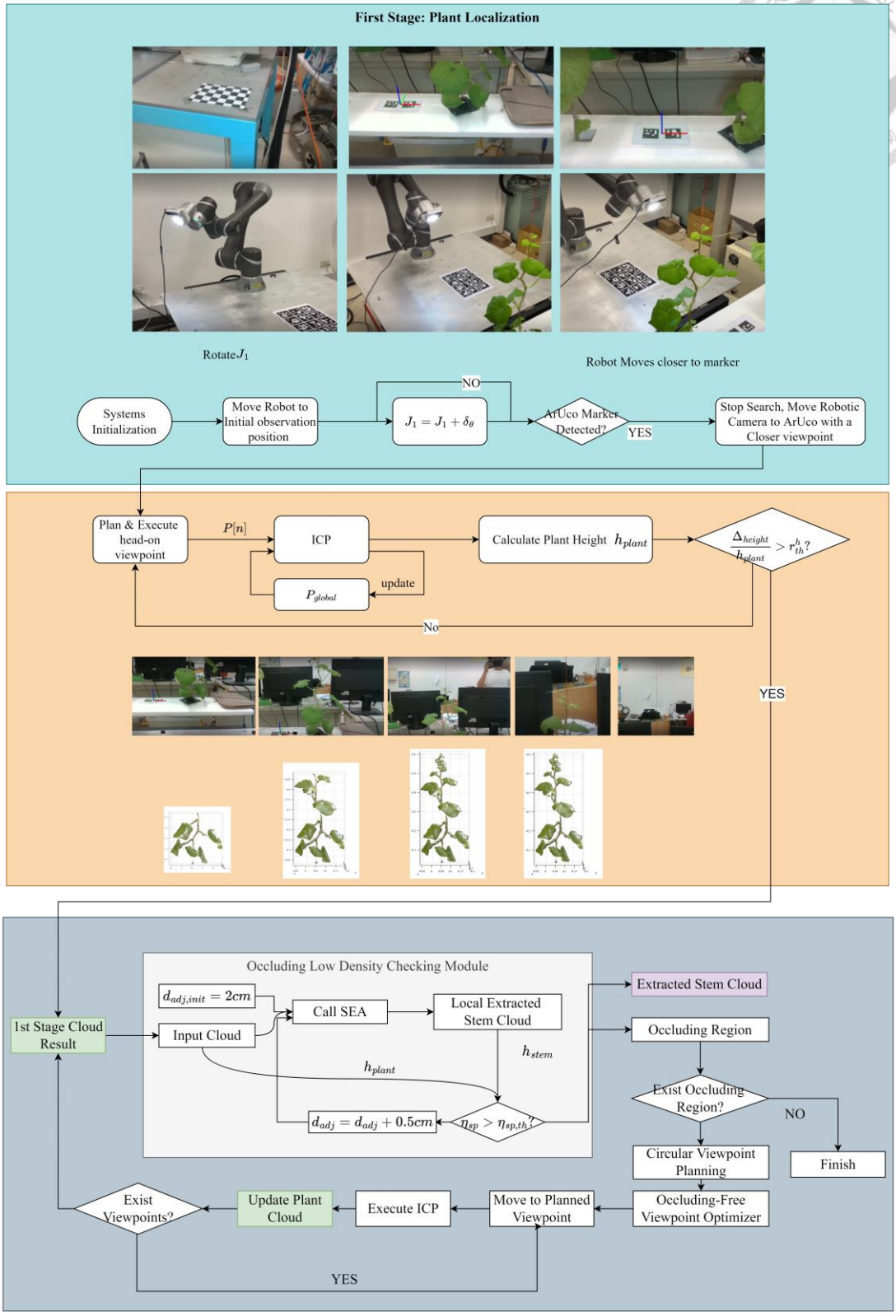


Figure 6.1 VBPR system architecture

doi:10.6342/NTU202303769

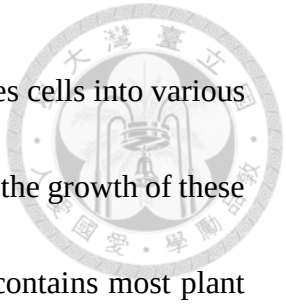
This Visual-Based Plant Reconstruction system briefly contains two main parts:

1. Marker-based plant automatic localization with rough plant cloud reconstruction
2. Plant Cloud fine reconstruct based on Stem Extraction Algorithm.

At the beginning stage of plant reconstruction, the robotic system needs to know the exact plant position with respect to the robotic base frame. Reasoning the plant's exact location from either image or pointcloud is quite sophisticated. Therefore, a fiducial marker attached to the top of the cultivating container is considered a bridge between the robotic manipulator and the center of the plant root. Meanwhile, the experimenter measured the transformation matrix from the fiducial marker and plant root in advance.

In the first stage of this process, the robotic manipulator moves from any random pose to a pre-defined pose, in which most of the objects in front of the camera, mounted on the end-effector, are visible, including the melon plant and ArUco Marker. By rotating the first joint of the manipulator, any registered ArUco marker appearing in this ring-like scanning region will call a stop sign to the robot arm controller. The transformation matrix between the robotic manipulator base and the specific ArUco-guided plant root is computed by programs. And then, the robotic manipulator plans several fixed vertical head-on viewpoints to capture the pointcloud of the observed plant, execute a series of pointcloud processing and align the pointcloud. Concurrently, the height of the observed plant is generated by computing the distribution of the aligned cloud.

doi:10.6342/NTU202303769



Similar to other plants, the apical meristem of melons differentiates cells into various organs with different functions, and the main stem is the beginning of the growth of these organs. Although the reconstructed point cloud from the first stage contains most plant parts like leaves, main stems, and petioles, there have possibilities that the reconstructed pointcloud has occluded by the leaf in front of the camera. This paper also proposes a stem extraction algorithm to segment stem pointcloud from the whole plant pointcloud used in second stage.

An occluding & low-density module uses SEA under different adjacent searching radii to adaptively overcome the interference from the undesired condition and generate occluding region and optimized-selected stem pointcloud.

In the second stage, the system uses the point cloud generated in the first stage as the basis, and through the occlusion interval generated by the OLD module, the occluding area is re-observed in a circular viewpoint, and the point cloud in the stem region is improved.

doi:10.6342/NTU202303769

6.2 First Stage: Fiducial-Based Plant Localization and fast reconstruction

In the real operating environment, knowing the exact location of plants in advance is not an easy task. Labelling the plant location in 3-dimensional space ahead of the experiment is time-consuming and the accuracy is not guaranteed when different people measure at different time. Also, global measured location needs one more transform from UGV to robotic base frame, increasing the uncertainty of the accumulation of the errors. Therefore, in this procedure, to simplify the complexity of the experiment, the location of each plant is indirectly detected from the location of a fiducial marker beside the plant. The result of this method is detected using image processing algorithm packages without extra transformation from the UGV to manipulator. Figure 6.2 illustrates the relationship between robotic system and observed plant.

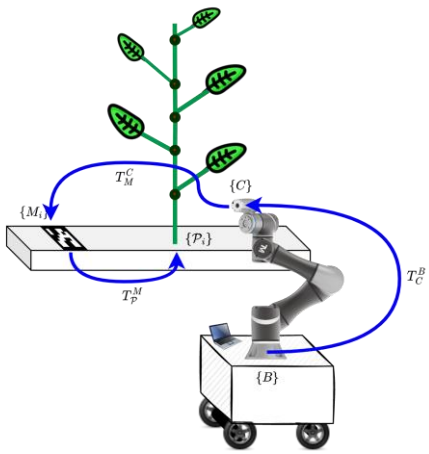


Figure 6.2 Relationship between Robotic system and observed plant

doi:10.6342/NTU202303769

Also, the Table 6.1 describes the general reference frames in space, and related transformation matrices.

Table 6.1 Definition of reference frames and transformation matrices

Symbols	Description
$\{B\}$	Base Frame
$\{C\}$	Camera Frame
$\{M_i\}$	i-th Marker Frame
$\{\mathcal{P}_i\}$	i-th Plant Frame
$\mathrm{T}_{\mathrm{C}}^{\mathrm{B}}$	Trans. Matrix from Base to Camera
T_M^C	Trans. Matrix from Camera to Marker
$\mathrm{T}_{\mathcal{P}}^{M}$	Trans. Matrix from Marker to Plant

At the beginning of reconstruction procedure, the joint states of the robotic system may be in anywhere. The robotic manipulator firstly moves to a fixed searching standby joint state $\boldsymbol{q}_{init}$, which defined as:

$$\boldsymbol{q}_{init} = [-90°, -30°, 135°, -75°, 90°, 0°\ ] \quad \text{Equation 6.1}$$

doi:10.6342/NTU202303769

The robotic manipulator under standby joint state is shown in Figure 6.3:

Figure 6.3 Initial searching standby joint state

Secondly, the algorithm requests the robotic manipulator moving first joint, from $\boldsymbol{q}_{init}^{0}$ to $\boldsymbol{q}_{end}^{0}$, where $\boldsymbol{q}_{end}^{0} = 90°$. The joint moving velocity factor has been set as 0.03, which the joint velocity is approximately $4.8\ °/s$. The following several figures shows this procedure.

(a) Start Searching

(b) In Search

doi:10.6342/NTU202303769

(c) In Search

(d) Fiducial Marker Found

(e) Detected fiducial marker by robotic mounted camera

Figure 6.4 Fiducial marker localization procedure

The identification (id_{M_i}) and pose $p_{M_i}[t_{mf}]$ of fiducial marker M_i is resolved using image processing library OpenCV, which the geometric information is based on camera frame $\{C\}$. The pose of ArUco is extracted by analyzing the position of pixel of black blocks. The jitter amplitude or detection error has positive correlation with marker distance from camera. Therefore, the system then needs to refine the accuracy of this marker, using the "marker found" pose $p_{M_i}[t_{mf}]$ to generate a closer viewpoint to this marker M_i. The marker position under the base frame of robotic manipulator $\{B\}$ is derived using transformation matrix shown as follows:

$$\mathrm{T}_{\mathrm{C}}^{\mathrm{B}}[t_{mf}] * T_{M_i}^{C}[t_{mf}] = \mathrm{T}_{M_i}^{\mathrm{B}}[t_{mf}] \quad \text{Equation 6.2}$$

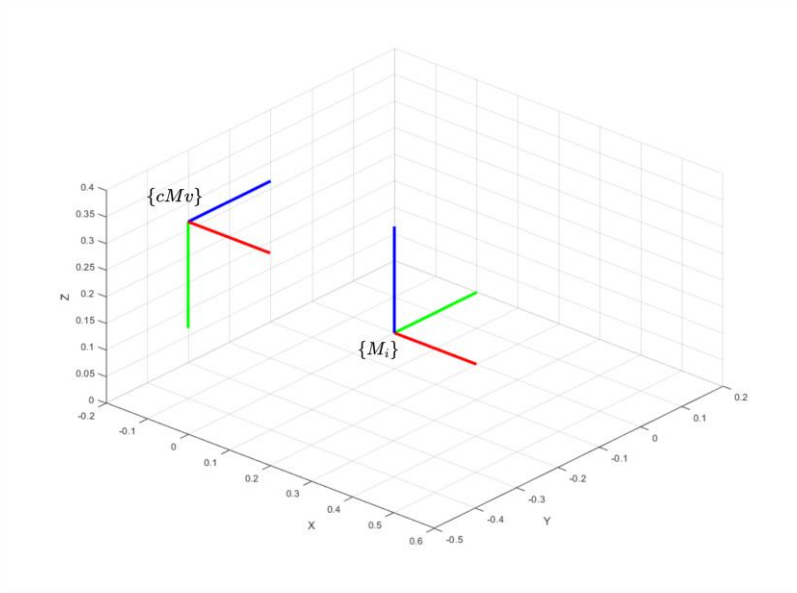
doi:10.6342/NTU202303769

This algorithm will plan a new closer viewpoint to marker M_i using the relationship $T^{B}_{M_i}[t_{mf}]$. The pose of new viewpoint under $\{M_i\}$ is defined as follows:

$$T^{M_i}_{cMv} = \begin{bmatrix} R_x(-90°) & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} * \begin{bmatrix} R_x(-30°) & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} * \begin{bmatrix} & r_{cMv} * cos(-270°) & \\ \mathbf{0} & r_{cMv} * sin(-270°) & \\ & h_{cMv} & \\ \mathbf{0} & 1 & \end{bmatrix}$$

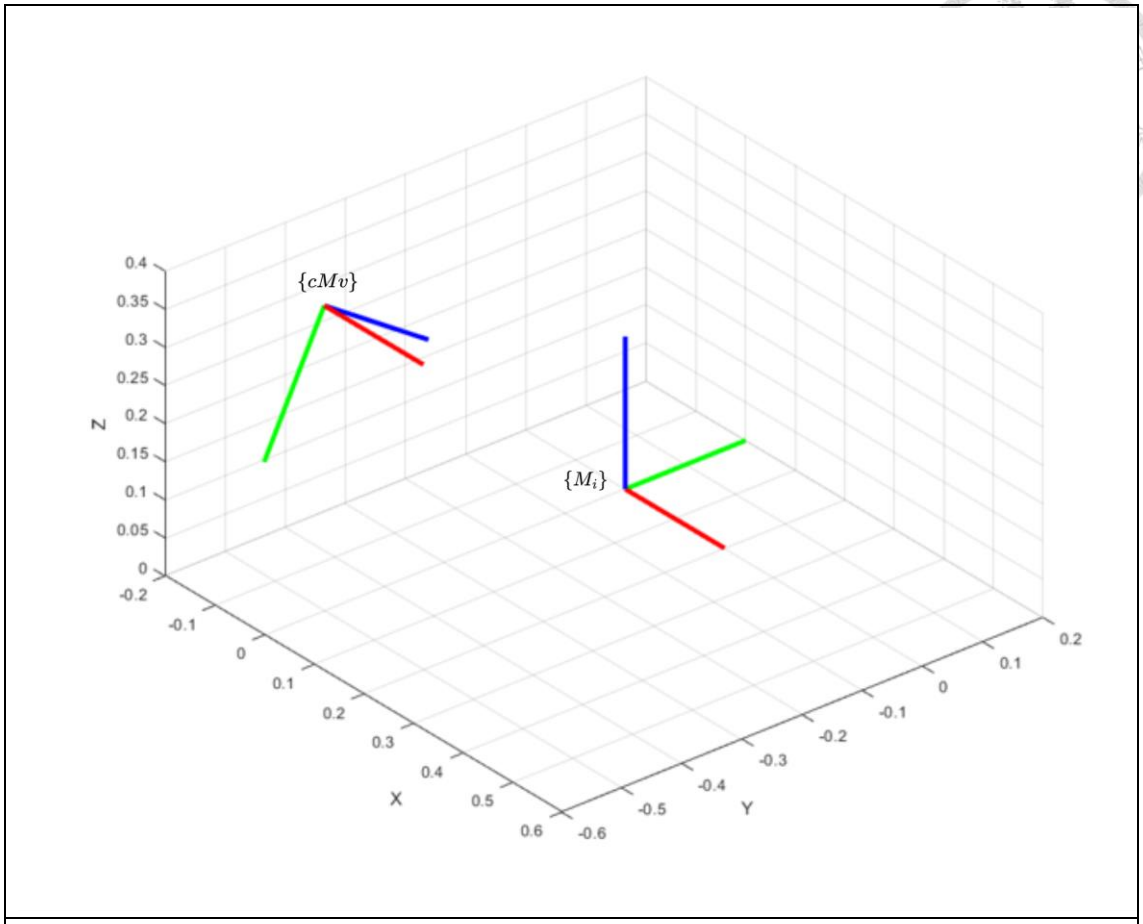
Equation 6.3

where, the $T^{M_i}_{cMv}$ is the transformation matrix of closer marker viewpoint (cMv) with respect to M_i. The first rotation term is rotating the frame along x-axis to fit the camera coordination matrix regulation, while the second term is to rotate camera imaging side. The r_{cMv} and h_{cMv} are designed radius and height or new planned viewpoint for closer marker view. Figure 6.? Illustrates how $T^{M_i}_{cMv}$ works:



(a) $T^{M_i}_{cMv}$ without rotates camera

doi:10.6342/NTU202303769



(b) $T_{cMv}^{M_i}$ with rotates camera for better marker view

Figure 6.5 Illustration of generating closer marker viewpoint

After knowing each transformation relations, the closer marker viewpoint under robotic base frame cloud be derived as：

$T_{cMv}^{B} = \mathrm{T}_{M_i}^{\mathrm{B}}[t_{mf}]\, T_{cMv}^{M_i}$	Equation 6.4

Calculating the inverse kinematics using the integrated move group services, a valid joint state T_{cMv}^{B} is executed. Figure 6.6 shows the camera is capturing the marker almost along the marker y-axis, which satisfies the excepted result.

doi:10.6342/NTU202303769

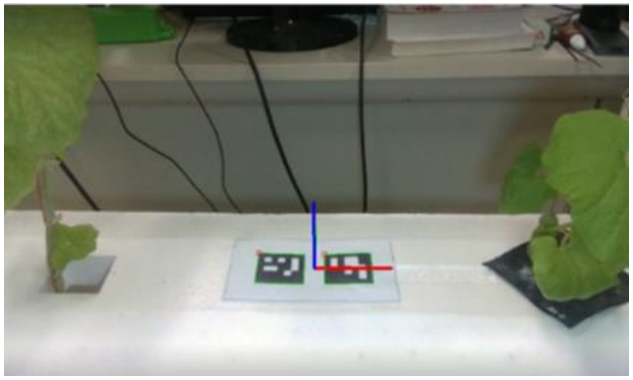


Figure 6.6 Robotic manipulator moves to closer marker view

To enhance the pose accuracy and cancel possible detection jitter, this algorithm takes the average of detected pose $p_{M_i}[t_{cMv} + N_{avg}]$ where n is the average sample size and robotic manipulator is **motionless**. Averaging of linear transformation is easier than that of quaternion. The relationship between robotic manipulator and observed fiducial marker is almost stationary. The direct quaternion averaging method cloud generate a fast and approximate solution when the quaternions not change a lot. The following equation shows how the system average the fiducial marker pose [90]:

$$p_{M_i}^{\text{avg}} = \frac{1}{N_{avg}} \sum_{i=0}^{N_{avg}-1} p_{M_i}[t_{cMv} + i] \qquad \text{Equation 6.5}$$

where the $p_{M_i}[t_{cMv} + i]$ is considered as a row vector with 7 items.

Also, the averaged pose with respect to robot base frame is described as:

doi:10.6342/NTU202303769

$\mathrm{T}_{\mathrm{M_i},avg}^{C} = \begin{bmatrix} R\left(p_{M_i}^{\mathrm{avg}}\right) & t\left(p_{M_i}^{\mathrm{avg}}\right) \\ 0 & 1 \end{bmatrix}$	Equation 6.6

$T_{M_i,avg}^{B} = \mathrm{T}_{\mathrm{C}}^{\mathrm{B}}[t_{cMv} + i] * \mathrm{T}_{\mathrm{M_i},avg}^{C}[t_{mf}]$	Equation 6.7

Usually, the fiducial marker is placed on a horizontal or sticks on vertical plant, the plant location with respect to robotic base frame cloud be derived by post-multiplying a transformation matrix $T_{\mathcal{P}_i}^{M_i}$, where $\mathcal{P}_\mathrm{i}$ and M_i represents the i-th plant and marker shown as follows.

$T_{\mathcal{P}_i}^{B} = T_{M_i,avg}^{B} * T_{\mathcal{P}_i}^{M_i}$	Equation 6.8

Until now, the robotic system already knows the exact location of the observed plant in the 3-dimensional space. Also, we know that the height of any observed plant won't exceed over 1.56 meters, that is, the upper side of the iron shelf and humans will trim the exceeding part. Although the system could generate a brute-force solution consisting of many viewpoints to reconstruct the observed plant, this is time-consuming, and the error or noise in pointcloud will be accumulated. Therefore, in the stage after plant localization, this algorithm moves the camera mounted on the robotic manipulator in front of the observed plant, plans fixed amounts of viewpoints based on plant frame $\{\mathcal{P}_i\}$, executes the fast plant pointcloud reconstruction procedure according to the captured pointcloud from each viewpoint, and calculates the plant height change rate to continue or stop the

doi:10.6342/NTU202303769

fast reconstruction. This algorithm will significantly increase the reconstruction efficiency and processing speed.

Different robotic manipulator has different size of valid dexterous workspace. In our scenario, the dexterous workspace radius is about 1.0 meter, which covers a large amount of the observed plant region. Assume that the value of z-axis of our robotic manipulator is moving from $h_{s,start}$ to $h_{s,end}$ with a gap $h_{s,gap}$, which means the start and end of the manipulator observing height and height increasing interval, respectively. For any height $h_s[i]$ satisfies:

$h[i] \in (h_{s,start}, h_{s,end}]$	Equation 6.9

For each planned "head-on" viewpoint contains a rotation matrix $R_h[i]$ and a translation vector $t_h[i]$, combining them to $T_{hv}^{\mathcal{P}_i}[i]$ where hv means "head-on viewpoint". For translation vector, the robotic manipulator could not work under any arbitrary input Cartesian position due to dexterous workspace limitation. Therefore, the notation of height threshold $h_{s,th}$ and equivalent observed height $h_{eqv}[i]$ is introduced. The equivalent observed height defines as:

$h_{eqv}[i] = \begin{cases} h[i], & h[i] < h_{s,th} \\ h_{s,th}, & else \end{cases}$	Equation 6.10

doi:10.6342/NTU202303769

Knowing the equivalent observed height, the head-on translation vector is described as follows:

$$t_h[i] = \left[r_{obv}\cos(270°), r_{obv}\sin 270°, h_{eqv}[i]\right] \quad \text{Equation 6.11}$$

Similarly, the plant coordination and the camera coordination have $90°$ difference on x-axis mentioned before. Furthermore, the scanning range is designed as a vertical sector to enlarge the coverage when the observed region is beyond the reachable workspace. The camera tilt angle θ_{tilt} which rotates the sensing device up or down defined as follows:

$$\theta_{tilt}[i] = \begin{cases} 37 * h[i]\,° + 22\,°, & h[i] \le 0.6 \\ -45 * h[i]\,° + 41\,°, & h[i] > h_{s,th} \end{cases} \quad \text{Equation 6.12}$$

The function of θ_{tilt} is an empirical solution to move camera, the curve of this function illustrates as Figure 6.7:

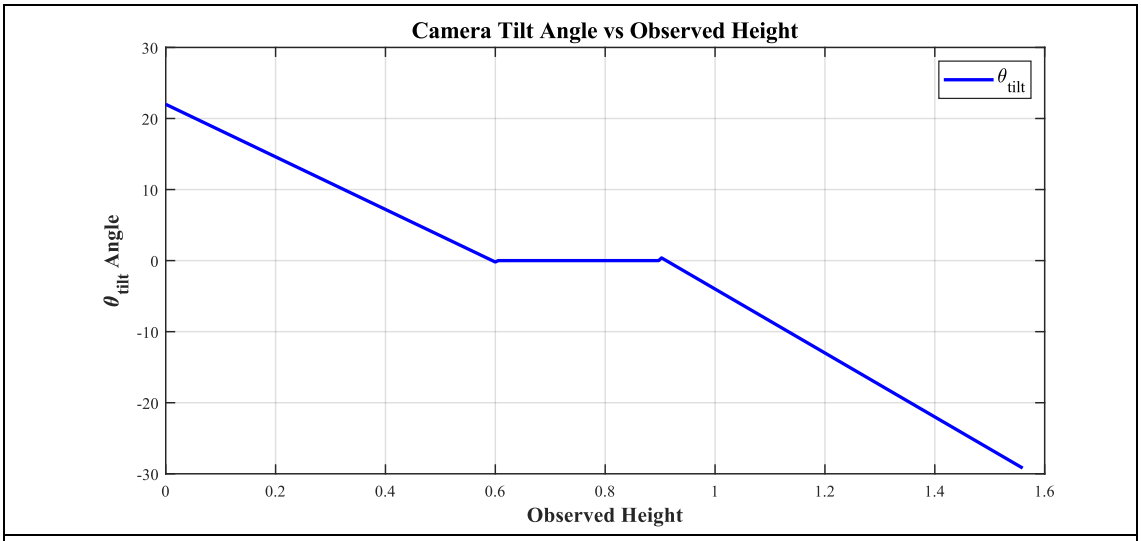


Figure 6.7 Camera tilt angle vs observed height

doi:10.6342/NTU202303769

According the camera definition, the positive angle rotates the camera downward, namely capturing towards to the ground, *vice versa*. The rotation matrix of head-on viewpoints is generated the three following transformations:

$$R_h[i] = R_x(-90°) * R_x(\theta_{tilt}) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_{tilt}[i]) & -\sin(\theta_{tilt}[i]) \\ 0 & \sin(\theta_{tilt}[i]) & \cos(\theta_{tilt}[i]) \end{bmatrix} \quad \text{Equation 6.13}$$

Therefore, the transformation matrix combines the rotation and translation part:

$$T_{hv}^{\mathcal{P}_i}[i] = \begin{bmatrix} R_h[i] & t_h[i] \\ \mathbf{0} & 1 \end{bmatrix} \quad \text{Equation 6.14}$$

This $T_{hv}^{\mathcal{P}_i}[i]$ is the information under $\mathcal{P}_\mathrm{i}$ coordination, the robotic system cloud uses it after transformation to robot base frame{B}:

$$T_{hv}^{B}[i] = T_{\mathcal{P}_i}^{B} * T_{hv}^{\mathcal{P}_i}[i] \quad \text{Equation 6.15}$$

And then, this procedure executes the plant reconstruction task. At the timestamp $t[i]$, the robotic manipulator is moved to the viewpoint $T_{hv}^{B}[i]$ to receive pointcloud ${}^{\{C\}}P_{t[i]}$ under camera frame {C}. The pointcloud under camera is meaningless because the robotic system cloud not resolve them. To transform the pointcloud from camera frame {C} to robotic base frame {B}, the system needs to know the current joint state $\boldsymbol{q}$

doi:10.6342/NTU202303769

for forward kinematic solver which describes the spatial information given angles. A typical six-joint robotic manipulator has following relationship:

$T_6^B(\boldsymbol{q}) = T_1^{0B} T_2^1 T_3^2 T_4^3 T_5^4 T_6^5$	Equation 6.16

Also, the camera is mounted on the end-effector, the transformation calculated using hand-eye calibration mentioned in Section 3.2:

$T_C^B(\boldsymbol{q}) = \mathrm{T}_6^0\, T_C^6$	Equation 6.17

Each transformation contains current axis joint state and fixed translation depended on robotic itself. After knowing the forward kinematic method, the pointcloud under camera frame is able to transform to robotic base frame:

${}^{\{B\}}P_{t[i]} = T_C^B(\boldsymbol{q})\,{}^{\{C\}}P_{t[i]}$	Equation 6.18

where each point in ${}^{\{C\}}P_{t[i]}$ is appended 1element for homogeneous calculation.

However, the transformed pointcloud ${}^{\{B\}}P_{t[i]}$ is not filtered, which non-related points exist in this original cloud, for example, the white foam cultivating platform, the iron shelf.

After the pointcloud filter mentioned in Section 5.1, the unrelated part of pointcloud will be eliminated. The output pointcloud defined as ${}^{\{\mathcal{P}_i\}}P'_{t[i]}$ which adds a prime sign.

By using the pointcloud alignment algorithm described in Section 5.2, the result aligned pointcloud $P_{merged}[i+1]$ will be defined as follows:

doi:10.6342/NTU202303769

$P_{merged}[i+1] = ICP\left({}^{\{\mathcal{P}_i\}}P'_{t[i]}, {}^{\{\mathcal{P}_i\}}P'_{t[i+1]}, d_{icp}\right)$	Equation 6.19

The ${}^{\{\mathcal{P}_i\}}P'_{t[i]}$ and ${}^{\{\mathcal{P}_i\}}P'_{t[i+1]}$ are target and source pointcloud from two camera viewpoints. The target pointcloud will be fixed and the ICP algorithm moves the ${}^{\{\mathcal{P}_i\}}P'_{t[i+1]}$ to find an optimized registration. Also, the d_{icp} is the radius to evaluate the correlation of source and target pointcloud at every iteration.

Furthermore, using the iterative merged pointcloud $P_{merged}[i+1]$, the height of the plant is also calculated by finding the extremum of the pointcloud. The geometric plant height is defined as follows:

$h_{\mathcal{P}_i}[i] = \min_z P_{merged}[i] - \min_z P_{merged}[i]$	Equation 6.20

Usually, the $\min_z P_{merged}[i]$ is considered as 0.

After getting at least two calculated height data, the plant height increase rate has been defined as follows:

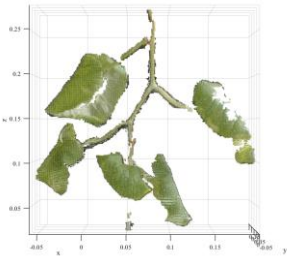
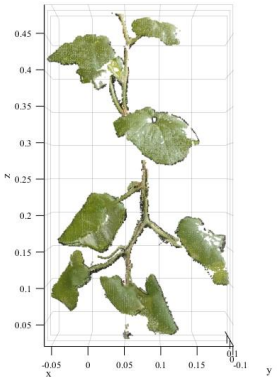
$\eta_h\,[i+1] = \frac{h_{\mathcal{P}_i}[i+1]}{h_{\mathcal{P}_i}[i]} * 100\%$	Equation 6.21

This rate represents the constructed height increasing speed, if the value $\eta_h\,[i+1] < \eta_{h,th}$, the system will stop the further head-on viewpoint planning and execution. Usually, the value of z-axis of each viewpoint are separately in several decimeter. The height increasing rate should be positive and gradually decrease.

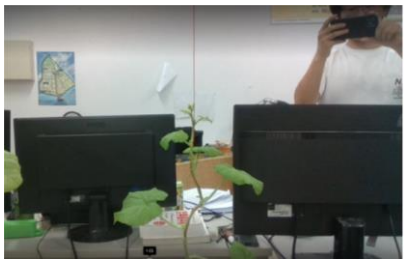
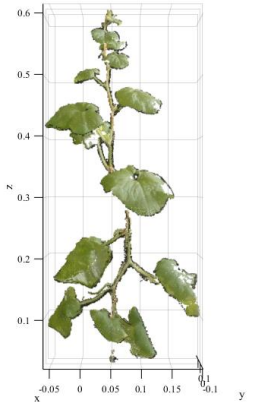
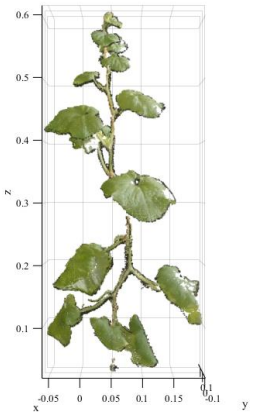
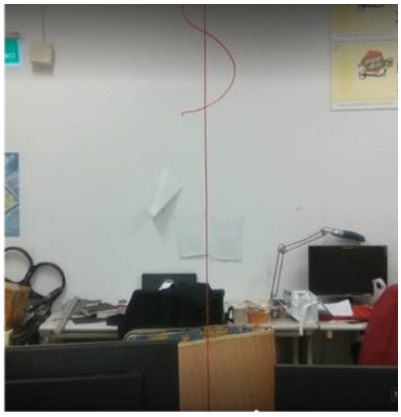
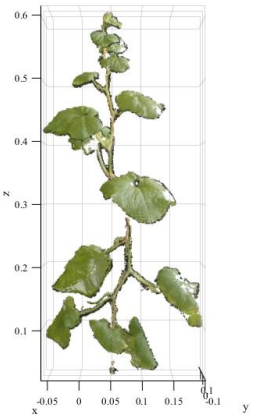
doi:10.6342/NTU202303769

Furthermore, another criterion to evaluate whether the system stops the head-on viewpoint plant reconstruction is the size of the filtered incoming pointcloud. If the size of filtered cloud ${}^{\{\mathcal{P}_i\}}P'_{t[i]}$ is zero or lower than a small threshold, this is also a termination signal of first-stage procedure.

The following figure sets shows the example of the observed plant under different "head-on" viewpoints. The image shows the procedure of movement of the viewpoints, where the plant appears in the camera region and then disappear.

Viewpoint	Robotic Camera	Reconstructed Pointcloud
1		
2		

doi:10.6342/NTU202303769

3		
4		
5		

doi:10.6342/NTU202303769

6.3 Second Stage: SEA-Based Occluding-Free Viewpoint Re-planning and Re-Alignment

The melon plant pointcloud has been basically reconstructed from several “head-on” viewpoints in the first stage. One purpose of this thesis is to segment different organs of this plant, and the extracted stem pointcloud is the critical information. To ensure the intactness and completeness of the segmented result, a procedure to evaluate and improve the input pointcloud is necessary. The previously mentioned stem occluding checking module is considered as a robust tool to extract stem pointcloud and also analyses the missing part / region.

This second stage procedure is recapped from the following figure:

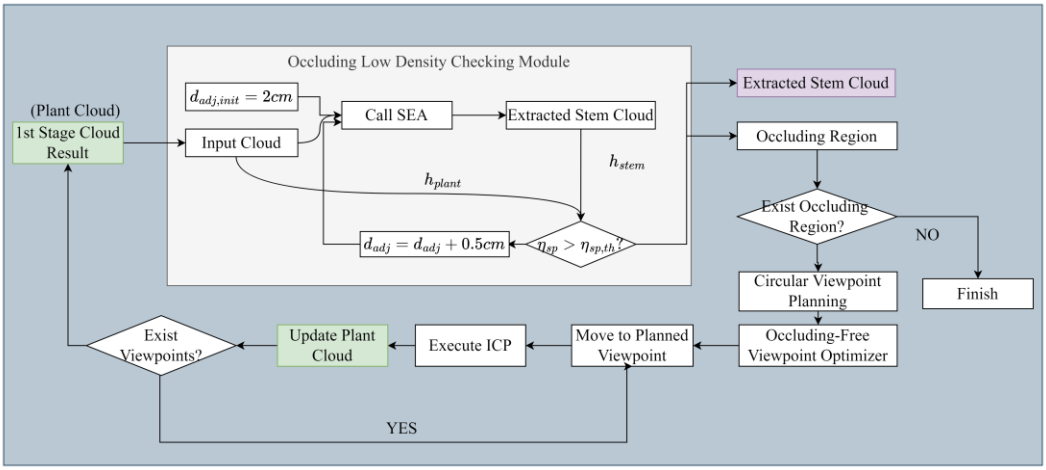


Figure 6.8 The second stage procedure

doi:10.6342/NTU202303769

Assumed that the occluding region set $\boldsymbol{R}_{occ}$ is defined as follows:

$$\boldsymbol{R}_{occ} = \left\{ \left(h_l^k, h_u^k\right) \in {}^{\{\mathcal{P}_i\}}P'_{t[i]} \right\} \quad \text{Equation 6.22}$$

Where the size of the set $\boldsymbol{R}_{occ}$ is N_{occ}, the h_l and h_u represents the minimum and maximum value of z component in one occluding region. And the k is the index of the analyzed occluding region.

For each occluding region $\boldsymbol{r}_{occ}^k \in \boldsymbol{R}_{occ}$, planning a series of occluding-free viewpoints is helpful to refine the pointcloud from the missing part. The viewpoint around the region $\boldsymbol{r}_{occ}^k$ is planned in a cylinder form. The planning algorithm is introduced in the following part.

By using the cylinder viewpoint planning algorithm mentioned in Section 5.5, a set of viewpoints in a cylinder surface is planned. The $\boldsymbol{T}_{Cylinder}^{\mathcal{P}_i}$ represent this set is defined. However, not all viewpoint $T_{Cylinder}^{\mathcal{P}_i}[\mathrm{i}] \in \boldsymbol{T}_{Cylinder}^{\mathcal{P}_i}$ is the best viewpoint under the given knowledge. Some planned viewpoint may still have occluding condition, which moves to this viewpoint is meaningless and time-consuming. Based on current known information, the current reconstructed pointcloud is given, which helps the system to reversely derivate the quality of this viewpoint.

In the Section 5.6, the occluding-free viewpoint optimizer gives a solution when the system knows the plant pointcloud P_{merged} and planned viewpoints $\boldsymbol{T}_{Cylinder}^{\mathcal{P}_i}$. After

doi:10.6342/NTU202303769

calculating the blocking score of each viewpoint, the system will select the viewpoint from lowest score ${}^{opt}T_{Cylinder}^{\mathcal{P}_i}[i]$:

${}^{opt}T_{Cylinder}^{\mathcal{P}_i}[i] \in {}^{opt}\boldsymbol{T}_{Cylinder}^{\mathcal{P}_i}, \mathrm{i} = 0, 1, 2 \ldots$	Equation 6.23

Similarly, the viewpoint T_C^{Plant} is under plant coordination, a transformation form plant to robotic base frame is necessary shown as follows:

$\mathrm{T}_{vp}^{B} = T_{\mathcal{P}_i}^{B} * T_C^{\mathcal{P}_i}$	Equation 6.24

The system then calculates the inverse kinematic solution of matrix T_{vp}^{B}. A reasonable translation and angular tolerance are applied to the IK solver and also, the planned viewpoint usually locates in the dexterous workspace of the manipulator. After moving the camera to the most occluding-free viewpoint, the new pointcloud captured from the depth camera passes through the plant filter mentioned in Section 6.2. And then, the procedure executes the alignment algorithm to register new filtered pointcloud to P_{merged}.

After moving one or more occluding-free viewpoint for each occluding region, the P_{merged} will be refined. The procedure then put the pointcloud P_{merged} to occluding checking module to evaluate the result of viewpoint re-planning.

doi:10.6342/NTU202303769

Once the current pointcloud P_{merged} passes the occluding checking module, the procedure of this step is finished. The following Figure 6.10 shows one example of stem pointcloud and its stem z-axis histogram before and after stage two procedure.

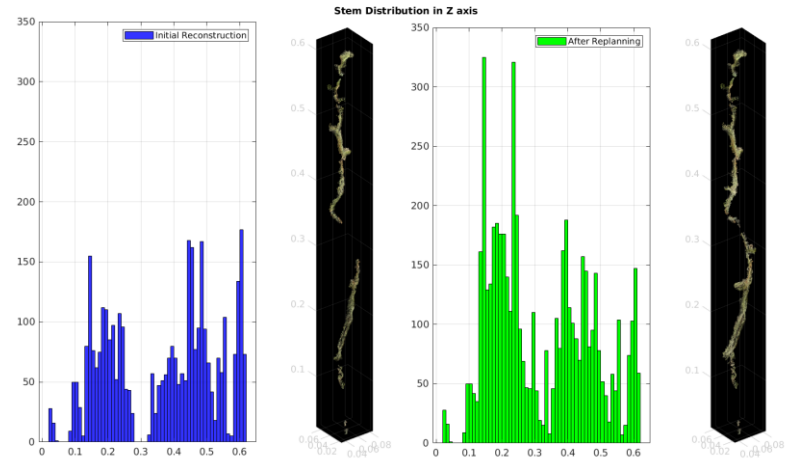


Figure 6.9 Stem pointcloud and histogram before and after refinement procedure

Also, in Figure 6.10, an example of comparison of two stem histogram has been shown to quantify the improvement of this procedure.

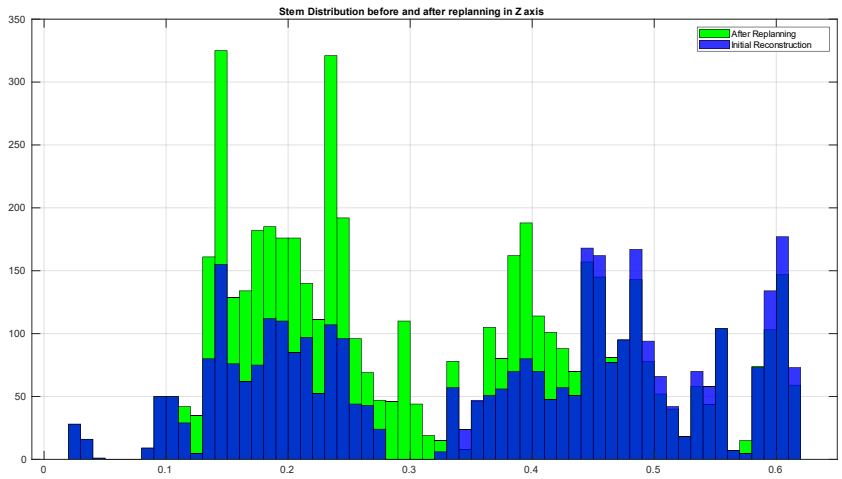


Figure 6.10 The comparison of stem distribution before and after refinement procedure

doi:10.6342/NTU202303769

Chapter 7
Experimental, Results and analysis

In Chapter 7, the experimental context, results, and analysis has been documented. The Section 7.1 introduces experimental equipment, greenhouse environment, laboratory experiment setting up, robotic manipulator hardware configuration and software deployment. In Section 7.2, the preparation works including camera calibration, hand-eye calibration and pointcloud filter have been introduced. In Section 7.3, we describe how our experiment conducted and results.

7.1 Experimental setup

In this thesis, the algorithms are tested on laboratory condition first because uncertain disturbance exists in the greenhouse environment which influences the program early development. In the Section 7.1, the experiment setup including the hardware configuration of laboratory and greenhouse environment and the field setup will be introduced.

7.1.1 General Operating Procedure

Whatever the location of this experiment is, the main algorithm of this operating procedure will not change too much. In this section, the procedure of laboratory will be

doi:10.6342/NTU202303769

introduced. And similarly, the greenhouse procedure which adds UGV operating will then described.

Before the experiment, the system has to finish the camera and hand-eye calibration. This is because the lens of camera is not intact and perfect and small distortions will lead a detection error of a fiducial marker. Also, the position of the end-effector-mounted camera is not precisely given. Although people could roughly measure the distance difference, a calculation of hand-eye matrix from multiple viewpoint is more reliable. The calibration content will be mentioned in the Section 7.2.

The following Figure shows the schematic diagram of the general plant reconstruction operating procedure.

There are four parts shown in Figure 7.5, which are initialization, Stage 1 & 2 and post-processing function blocks. The functions or services block shown in the initialization region are separately running in the computer.

doi:10.6342/NTU202303769

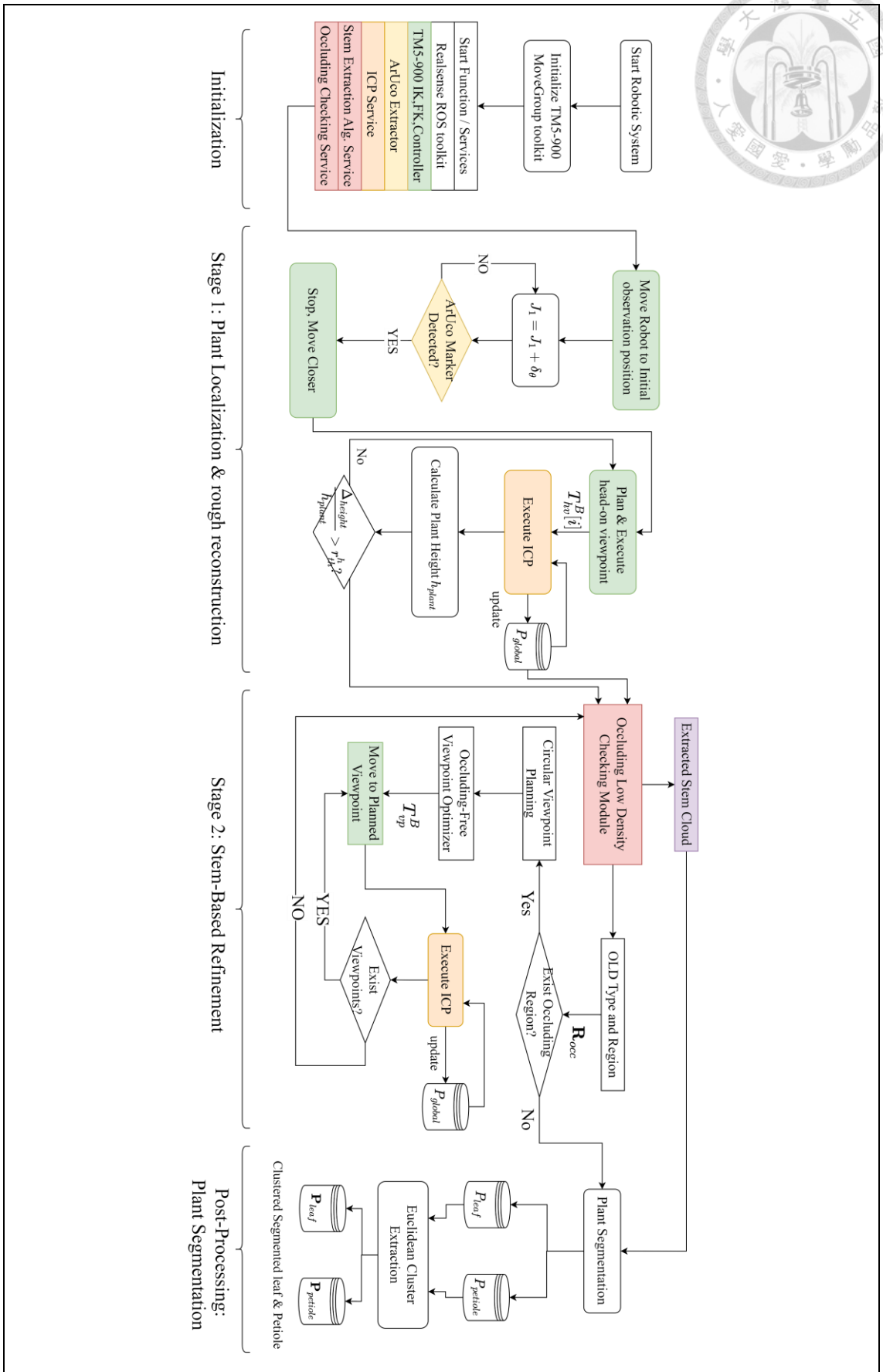
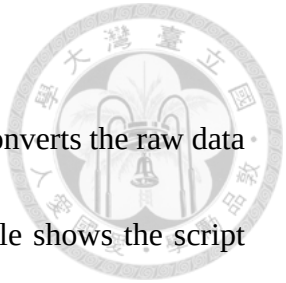


Figure 7.1 The program procedure in laboratory environment

doi:10.6342/NTU202303769



For example, the Realsense toolkit is a ROS launch file which converts the raw data from the Realsense to image or pointcloud topic. The following table shows the script running in the background as a service to help the system operating properly.

Table 7.1 Related scripts used in this procedure

Index	Function/Service/Script Name	Description
0	TM5-900 ROS Driver & Move Group	Running this script will generate the visualization interface and initialize the move group APIs. The built-in API provides the shortcut of some functions, like Inverse Kinematics and generate collision-free trajectory.
1	Realsense2_camera	An official package converts the raw camera information to image or pointcloud topic.
2	TM Controller	A controller with integrated services including executing joint state, calculating inverse kinematics, calculating forward kinematics and ArUco-Based functions
3	ArUco Extractor	A python script which uses OpenCV library to find pre-defined ArUco board or single marker. Return the pose array of recognized marker / board.
4	ICP Alignment	A service which align two dense pointcloud to a merged pointcloud. This service is an important algorithm in this system.
5	Stem Extraction Algorithm 3.0	The third version of stem extraction service. This service could analyses the input pointcloud and find the possible stem pointcloud.
6	Occluding Checking	This is a service which invokes the SEA service by changing different

doi:10.6342/NTU202303769

		adjacent searching radius. This service generates the occluding region by analyzing the distribution of extracted pointcloud. This script also could auto-termination function.
7	Main Procedure	The main script of this system. This script will finish the stage 1 and 2 function, and pointcloud post-processing (Segmentation). All these services and topics mentioned will be invoked by this script.

The main procedure script contains the first and second stage of reconstruction procedure and pointcloud organ segmentation algorithm. Therefore, in the laboratory environment, we will run this main procedure script before all the device is calibrated and the related services algorithm is online.

For greenhouse environment, the additional device is the UGV, which provides the extra workspace extension. The URDF () of the environment need some changes. The following figure shows the operating procedure in greenhouse condition. It is similar with the laboratory environment, the UGV moving between each cultivating row to find the possible plant fiducial marker. If a valid marker is found, the UGV system will ask the robotic system to analyze and reconstruct the target plant. Therefore, the output of the UGV system will be the input of the robotic system. After finishing same robotic procedure, the system will then return to the UGV side to verify whether the current UGV

doi:10.6342/NTU202303769

location is at the end of the cultivating platform. The UGV will move to the end to find all potential un-observed plant.

doi:10.6342/NTU202303769

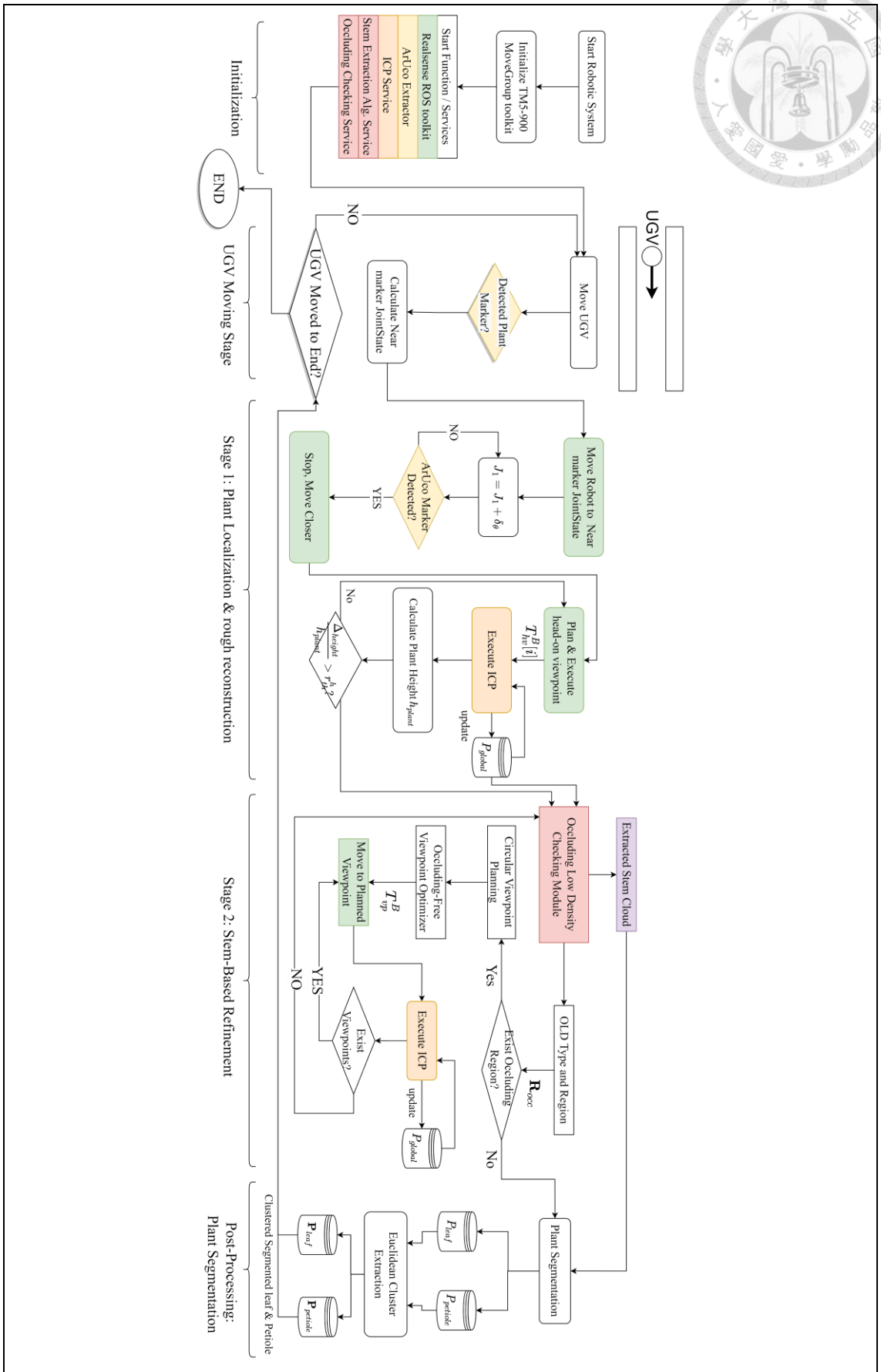


Figure 7.2 The program procedure in greenhouse environment

doi:10.6342/NTU202303769

7.1.2 Laboratory Environment

Although the greenhouse is the final applied environment, the complexity of real-world including plant cultivating interval, changing illumination condition, wind disturbance from cooling fans, will significantly challenge the robustness of the detection system. A laboratory environment is set to avoid these problems in algorithm development stage.

The hardware types and deployment vary between laboratory and greenhouse environment. The used hardware in the laboratory environment shows as the following schematic diagram:

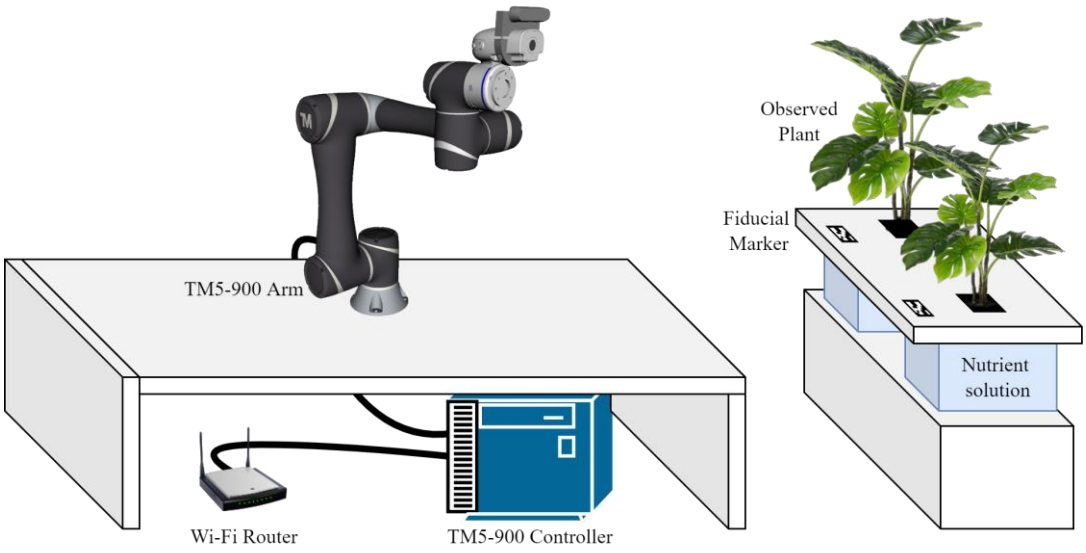
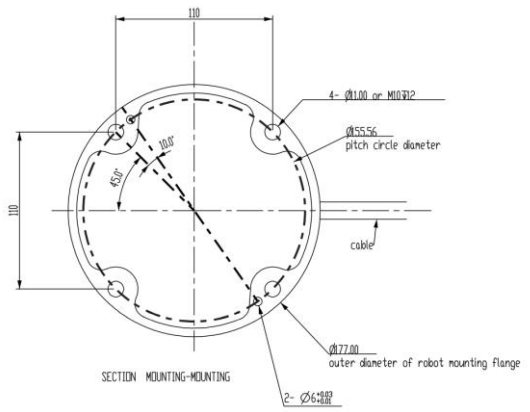


Figure 7.3 Laboratory environment schematic

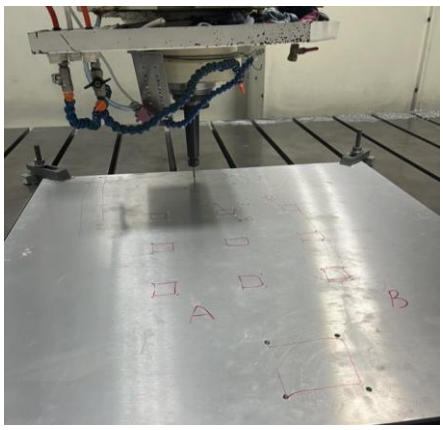
The TM5-900 robotic manipulator is main experiment tool, where an integrated controller drives the robotic system. The controller connected with the manipulator with

doi:10.6342/NTU202303769

a thick cable. Also, the controller communicates outside via the wired-internet to a Wi-Fi Router. The robotic manipulator TM5-900 is placed on the customized aluminum platform which also fixed on a lockable trolley. The bottom cross-section of robotic manipulator and related manufactured aluminum platform has been shown in Figure 7.4 (a) and (b).



(a) Robotic manipulator section



(b) Aluminum plate processing

Figure 7.4 Laboratory environment preparation

In front of the aluminum platform, a same white foam cultivating platform is deployed and the culture medium container is placed below that platform. The following figure shows the cultivating platform and culture medium container.

On the cultivating platform, an ArUco marker is used to localize the plant from known transformation defined in advance. The artificial illuminance LED is set up at left and right side to provide a constant light density and temperature. Due to the limited space of the laboratory, a virtual wall is added in the URDF of this system to avoid potential collision.

doi:10.6342/NTU202303769

In this scenario, the field operating figure shows how this system shown in Figure 7.5 works in real laboratory environment:

Figure 7.5 Laboratory environment configuration

7.1.3 Greenhouse Environment

The implementation site of this experiment is in an intelligent greenhouse attached to the farm of National Taiwan University, as shown in Figure A. The greenhouse is xx in length, xx in width, xx in height, and has a hydroponics platform with four columns and three rows. A single platform is built with an aluminum alloy frame, as shown in Figure B. The distance from the bottom to the top is 1.8m, the width is 35cm, and the length is 4m. Each cultivation platform is divided into left and right sides, with six to eight cultivation spaces on each side. The actual planting is interval planting. The greenhouse has four available channels, of which the fourth channel is blocked by debris, and the available channel is one, two and three. The maximum channel spacing is 1.9 m,

doi:10.6342/NTU202303769

and the minimum is 1.12 m. The overall greenhouse cultivating platform configuration shows as follows:

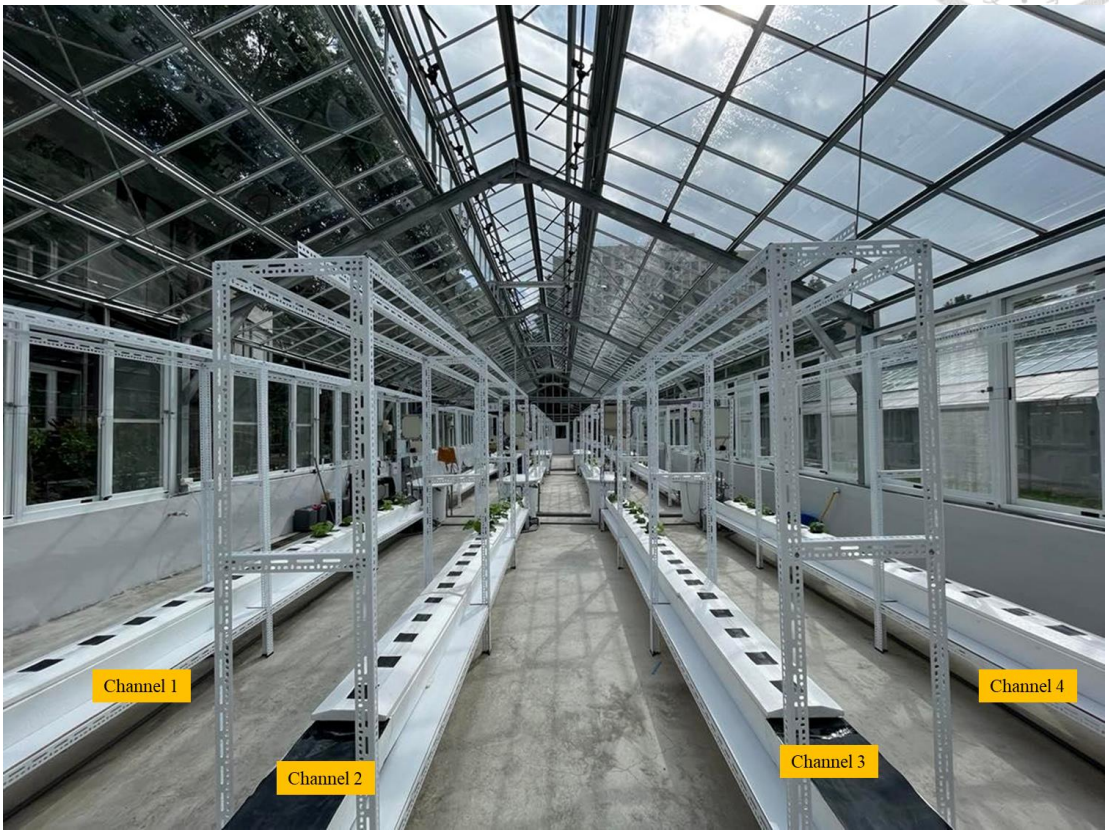


Figure 7.6 Greenhouse cultivating platform

The plant studied in this paper is Cantaloupe, cultivated in a specific cultivation laboratory in the early stage of seedling cultivation. After three to five weeks grow, it is transferred to a greenhouse hydroponic platform. After the plant has grown to the 9th section, the artificial growth starts, and the thin vertical line is used to make the plant grow vertically.

Compared with the laboratory environment, the unmanned ground vehicle is participated in this experiment to provide a workspace extension of the original robotic

doi:10.6342/NTU202303769

system. This hardware configuration in our greenhouse environment are illustrated in Figure 7.7:

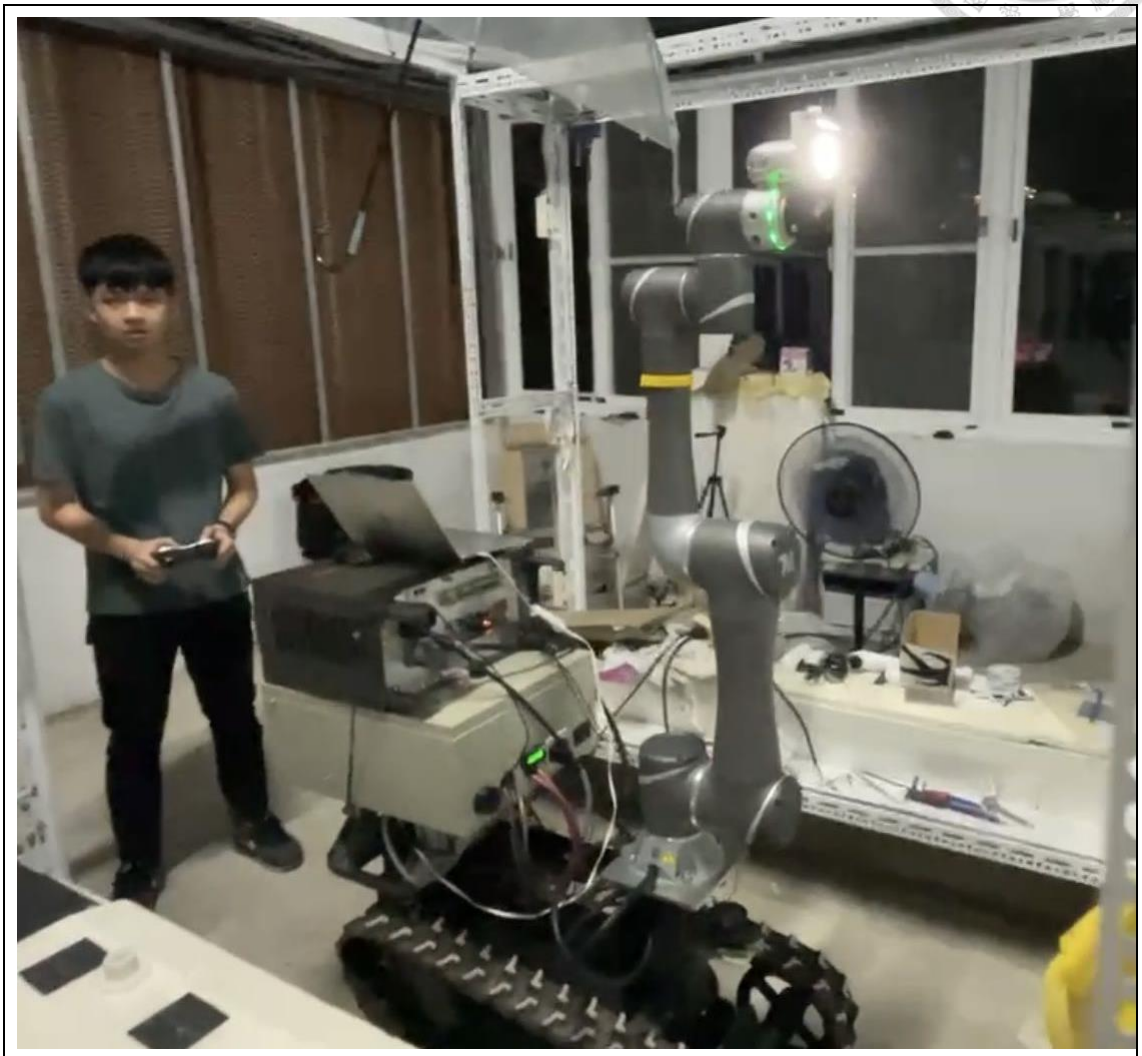


Figure 7.7 The greenhouse environment

7.1.4 Hardware specification

In this section, the detail of related hardware specification will be introduced.

Computer:

Over the all experiment in this thesis, the laptop with an Intel Core i7-10875H CPU @ 2.30GHz and 16.0 GB memory is used. The operating system for entire program is

doi:10.6342/NTU202303769

Ubuntu 20.04 which provides a good environment for program developer. A GPU card of RTX2060 with 6GB memory is accelerating limited function in our entire project.

Robotic Manipulator:

A robotic manipulator with large operating payload and large workspace radius, called TM5-900, provides more accurate pose achievability, lower error and large operating range up to 1.0 meters, while the price of the robotic manipulator expensive.

Table 7.2 TM5-900 Specification

Product Name	TM5-900		
Weight (kg)	13.5	Max payload (kg)	4
Reach (mm)	900	Typical Speed (m/s)	1.4
Joint Range			
Joint 1	±270°	Joint 3	±155°
Joint 2, 4, 5	±180°	Joint 6	±270°
Joint Speed			
Joint 1, 2, 3	±180°	Joint 4, 5, 6	±225°
Repeatability (mm)	±0.05	Operating Temperature (°C)	0~50

Realsense:

The main visual sensor of both robotic systems is the depth camera produced by Intel called Realsense D435, providing the integrated dual camera triangulation and pointcloud construct. This version of depth camera is most commonly used for robotic usage, which

doi:10.6342/NTU202303769

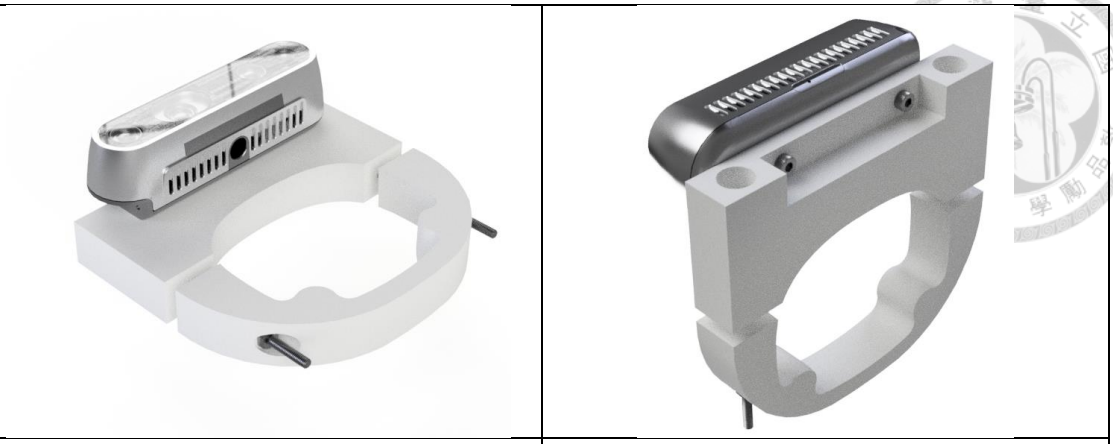
the performance in agricultural application is outstanding [CITE PAPER]. The specification of Realsense D435 is listed below:

Table 7.3 Realsense specifications

Dimensions		70.7 * 14 * 10.53 mm
Depth Resolution		1280x720
Color Resolution		1920x1080
Depth sensing range		0.2m ~ 3m (optimal)
Color Camera FOV		H: 69 °, V: 42 °, D: 77 °,
IR Camera FOV		H: 90 °, V: 63 °, D: 99 °,
Depth Camera FOV (16:9)		H: 87 °, V: 58 °, D: 95 °,
Depth Camera FOV (4:3)		H: 75 °, V: 62 °, D: 89 °,
Min Z Depth	1280x720	280 mm
	680x480	175 mm

Also, there is not hole to screw the fastener, a reverse model of robotic end-effector is designed and printed for fixing the Realsnese and robotic manipulator. The Realsense is fixed using two M2.5 screw shown in Figure 7.8 (b). Also, this fixture device has two part, which the upper and lower part connected using two long M3 screw with M3 nuts shown in Figure 7.8 (a). This model is printed and installed on the end-effector of the robotic manipulator. Due to the installation error, further hand-eye calibration to ensure the exact transformation is needed.

doi:10.6342/NTU202303769



(a) Top view | (b) Back view

Figure 7.8 TM5-900 flange and Realsense connector

Artificial Illuminance:

In addition, the intensity and direction of the sun light will vary over time, the weather changes. In this experiment, tracking the intensity and direction of sunlight and camera imaging affected by changing sunlight are not the focus of this paper. So, equipping a set of artificial illumination system is required to maintain the same luminance on the plant on each experiment and to simplify the experiment variables. In the nature, different ideal black radiation temperatures in varying time, also called color temperature. Figure 7.9 shows the sunlight temperature during a day:

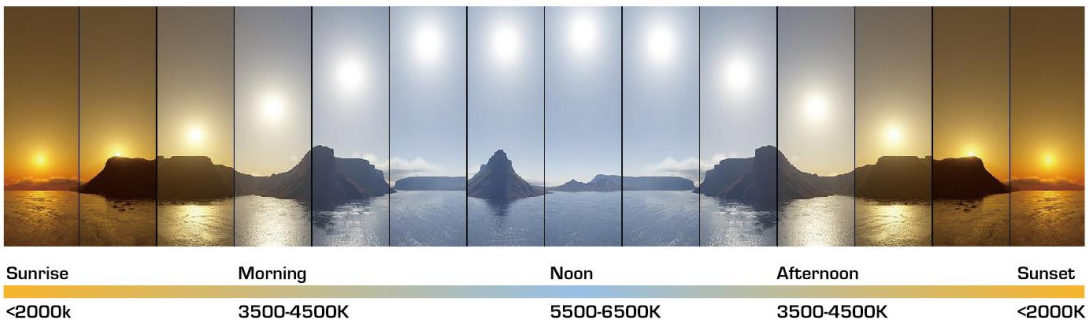


Figure 7.9 Sunlight Temperature in a day [91 Lumistrips]

doi:10.6342/NTU202303769

The temperature of sunlight in the daytime is ranging from 3500 K to 6500K. Therefore, we selected two 50-Watt chips on board (COB) sunlight LED device, which the Color Rendering Index (CRI) is greater than 95. Each LEDs are equipped with a lens to dense the light. The sunlight spectrum distribution and our LED excitation spectrum and curve shows as Figure 7.10 and 7.11:

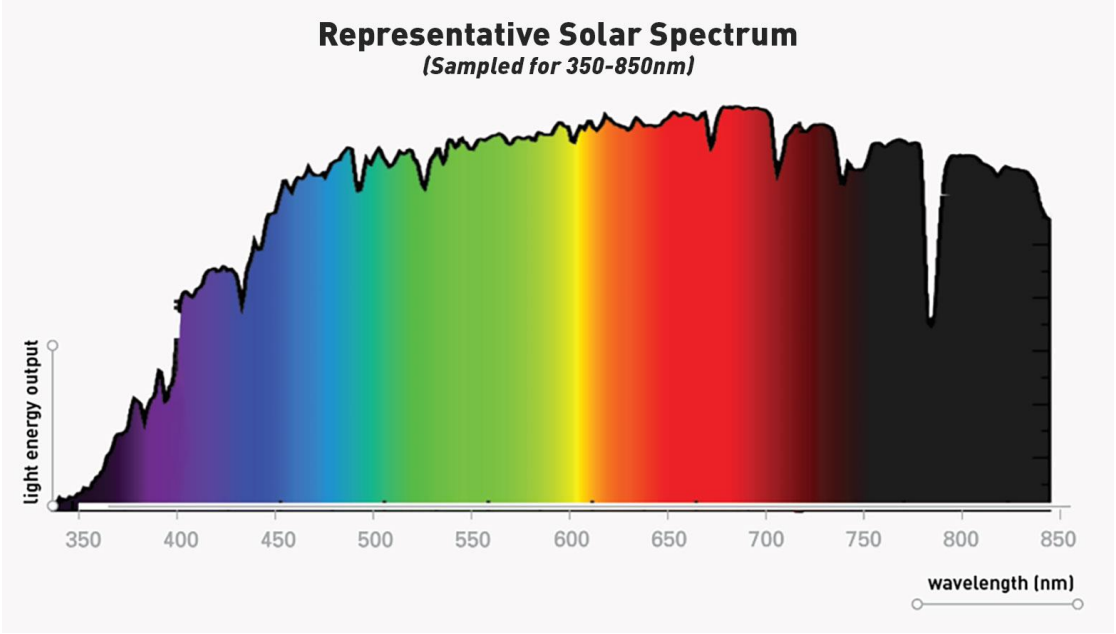


Figure 7.10 Sunlight spectral distribution curve [92 LumiGrow]

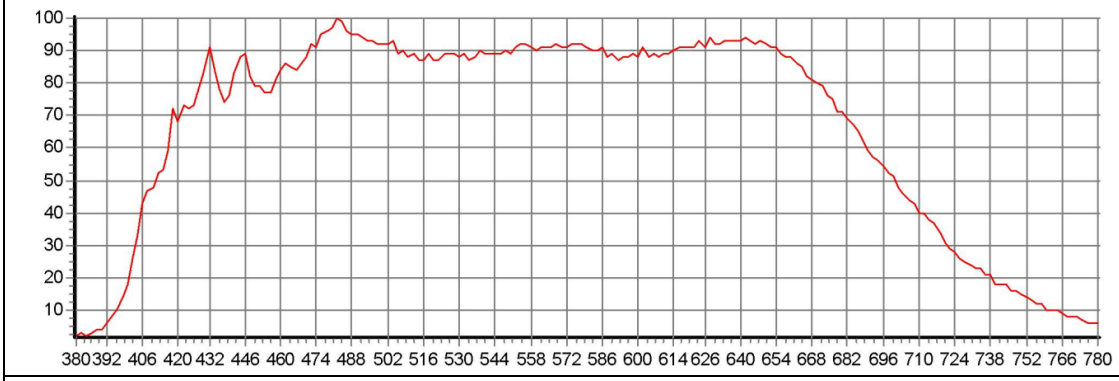


Figure 7.11 Our sunlight spectrum LED spectral distribution curve

doi:10.6342/NTU202303769

The LED equipment shows as following Figure 7.12:

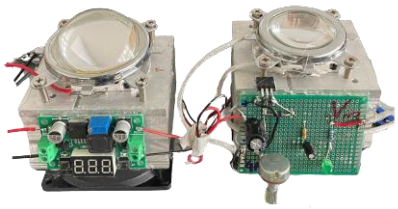


Figure 7.12 The Artificial Illumination Device

7.1.5 Software

All tasks are operating on the robotic operating system (ROS), an open-source robotic software integrating robotic control, motion planning, sensing, and communication. Each script in the ROS will initialize a node, which is a unique identification stamp for data communication in ROS. All data in ROS is encoded, published and subscribed using the topic mechanism.

On the ROS platform, many third-party libraries is utilized, including but not limited to Realsense SDK, PCL, Eigen Library, OpenCV, and OMPL.

The Realsense SDK communicates between the computer and the depth camera, publishes color images, depth images, infrared images, and point clouds to the local network and waits for other nodes to subscribe. The specific output topics and available resolutions are shown in the following table.

doi:10.6342/NTU202303769

OpenCV is an excellent open-source library for image processing, with basic image processing, color conversion, filtering, and ArUco marker recognition features throughout this article.

The point cloud related algorithms are also a critical step in data processing in this paper. In C++, Point Cloud Library (PCL) is an efficient open-source library for point cloud processing. Its relatively representative functions include, for example, KD-Tree for point cloud construction, color space conversion, nearest point search, Iterative Closest Point (ICP) reconstruction, the normal vector estimation, plane fitting, FPFH calculation. Also, the Eigen library helps the algorithm complete the coordinate transformation task under C++, which has built-in functions such as matrix multiplication and inverse matrix solution. In addition, the open motion planning library (OMPL) provides a variety of space search algorithms, such as RRT, A*., to provide solutions for collision-free trajectory planning of robotic arms in space.

7.2 Preparation works

7.2.1 Camera Calibration

RealSense contains one color camera module and two infrared camera modules. Color cameras can provide richer color information, while infrared cameras have higher

doi:10.6342/NTU202303769

FOV. Therefore, in practice, it is necessary to obtain the camera internal parameters of the two sets of cameras. Although Realsense's SDK provides internal parameters, based on the principle of experimentation, the camera's internal parameters will be re-solved here.

In this calibration, the calibration board used is a 7*9 checkerboard, as shown in the following Figure 7.13:

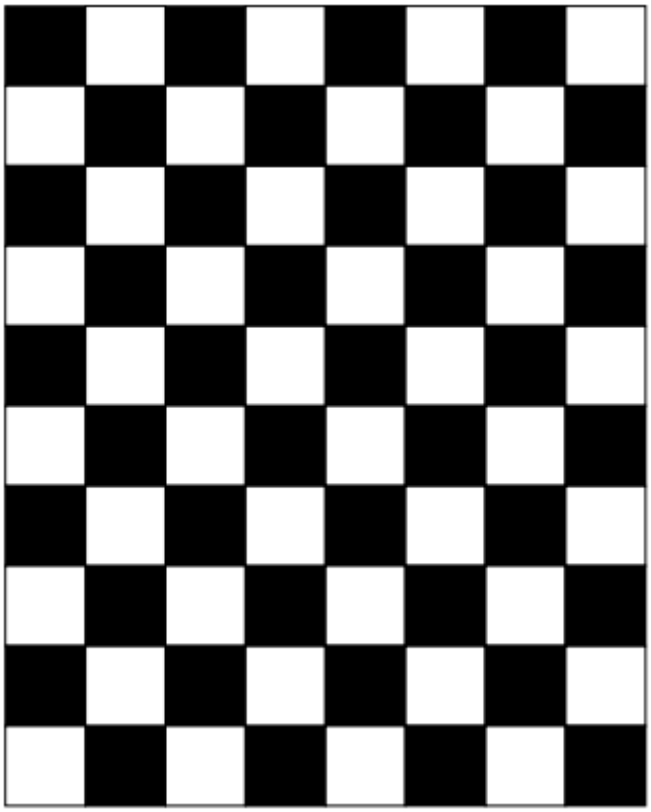


Figure 7.13 Camera intrinsic calibration board

First, we print the checkerboard at the original scale to confirm the resolution of the camera to be calibrated. In this calibration, we will calibrate the color camera and the depth camera.

doi:10.6342/NTU202303769

For the IR camera, as shown in Figure 7.14, we place the calibration board in the FOV of the camera and take about 20 pictures. Next, we use OpenCV's built-in API to detect checkerboards in the image (Figure 7.15) as the source data to calibrate the camera.

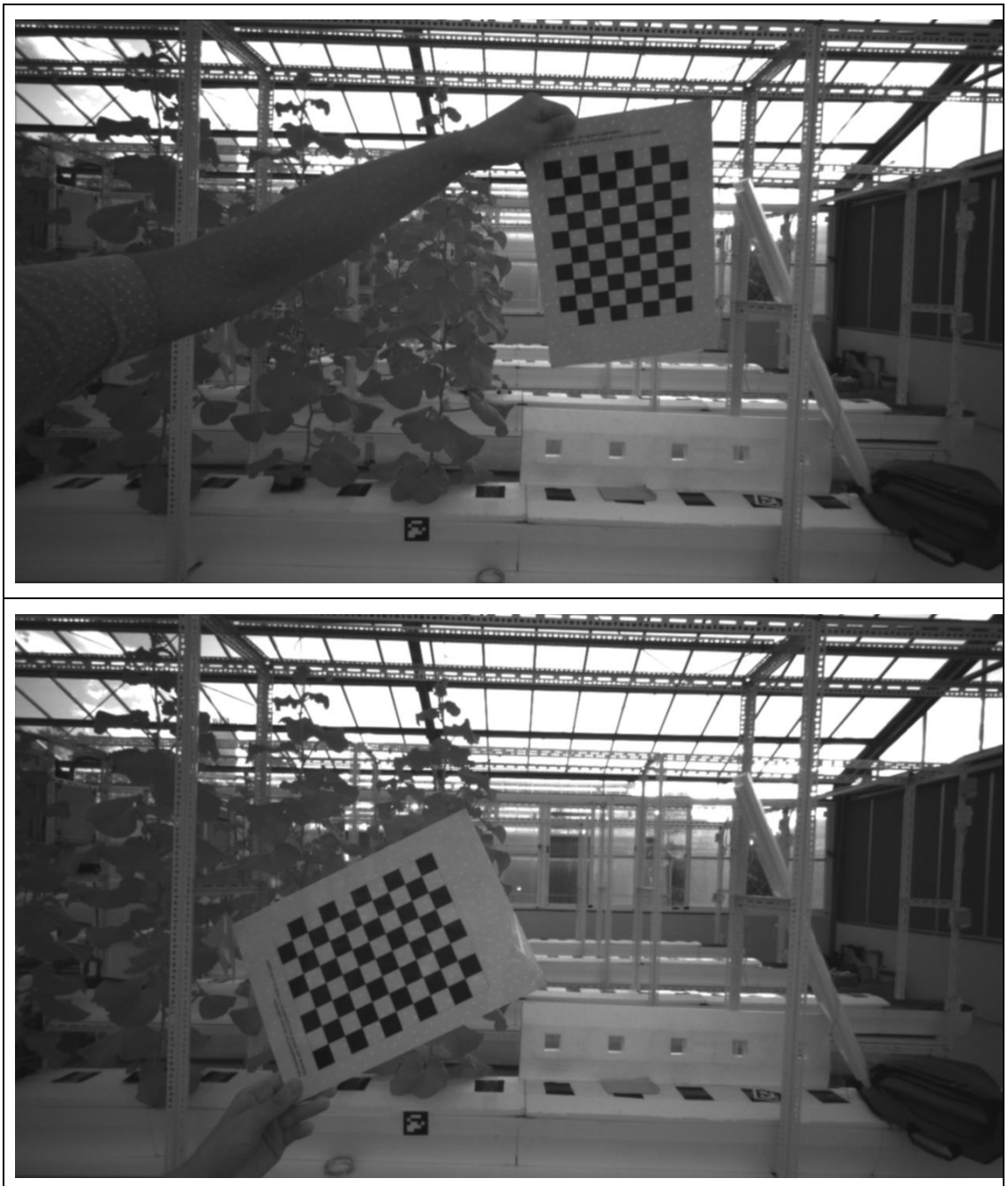


Figure 7.14 Original calibration chessboard in infread camera

doi:10.6342/NTU202303769

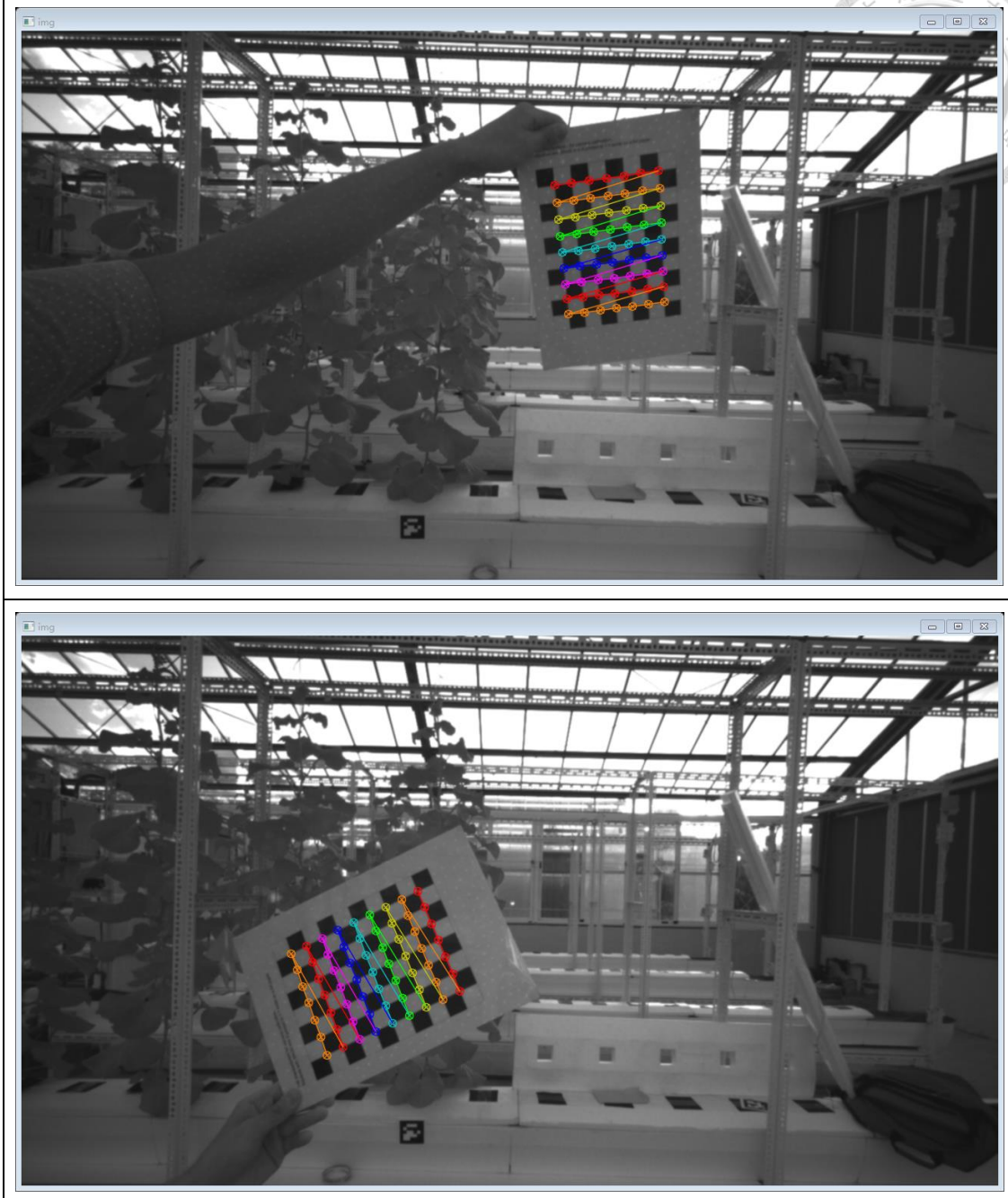


Figure 7.15 Detected chessboard in infread camera

After calculating the pixel relationship of each corner point of the checkerboard of multiple pictures, the system can obtain the internal parameter matrix and distortion matrix of the infrared camera through the API of OpenCV, as shown below:

doi:10.6342/NTU202303769

$I_{\text{infread}} = \begin{bmatrix} 618.96828422 & 0 & 665.64380579 \\ 0 & 616.90351154 & 359.47544161 \\ 0 & 0 & 1 \end{bmatrix}$	Equation 7.1

$dist_{\text{infread}} = \begin{bmatrix} -3.42 * 10e-2 \\ 2.167 * 10e-2 \\ -6.86 * 10e-5 \\ 7.06 * 10e-3 \\ -1.1 * 10e-2 \end{bmatrix}^T$	Equation 7.2

Similarly, for a color camera, the system first uses two resolutions of 680*480 and 1280*720, after taking multiple checkerboard photos (picture), uses API for corner detection (picture), and finally gets the color Internal parameters of the camera (formula):

7.2.2 Hand-Eye Calibration and result

Hand-eye calibration is to obtain the relationship between the camera imaging coordinate system and the end-effector of the robot arm. The second subsection of Chapter 3 describes the specific steps of calculating the hand-eye calibration. The experimental hand-eye calibration requires at least ten sets of the following conversion relationships:

- the conversion matrix from the robot arm base to the end effector (flange surface)
- the conversion matrix from the flange surface to the depth camera colour Frame
- the colour coordinate system to the transformation matrix of the calibration board

doi:10.6342/NTU202303769

Under the assumption that the Realsense is well fixed and each measurement is consistent, the ChArUco calibration plate is used in the experiment to obtain the conversion relationship between the camera and the calibration plate. As shown in Figure 7.16 below, this is a calibration board with several rows and columns, and each interval is (for reference) [93]:

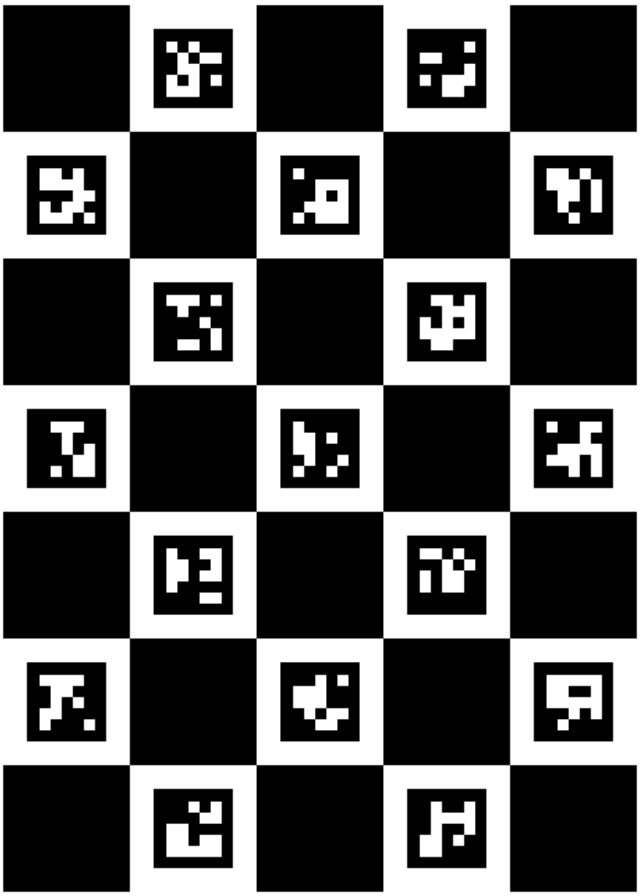
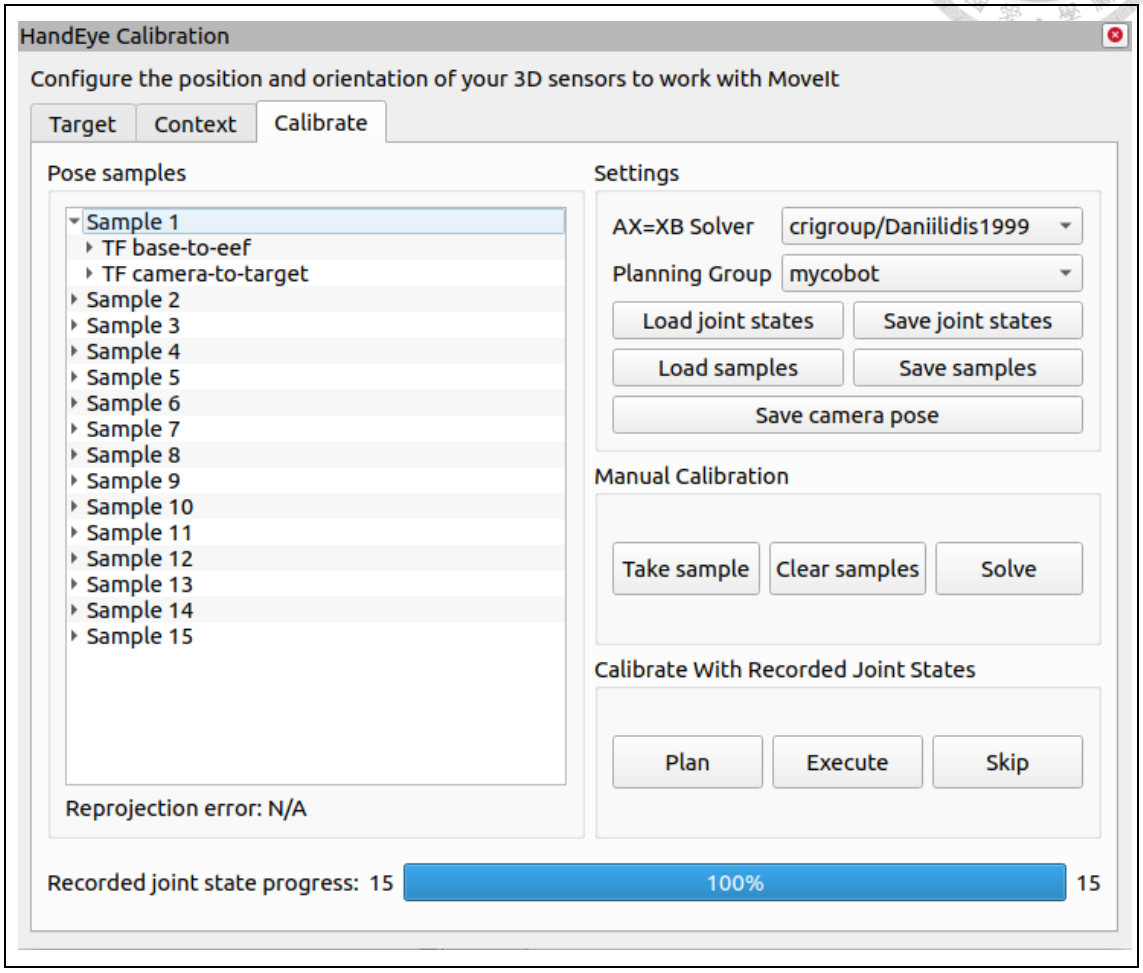


Figure 7.16 ChArUco calibration board

Rviz is a visualization tool in ROS and a package called “Hand Eye Calibration” provides an integrated GUI containing the functions of sampling robotic manipulator

doi:10.6342/NTU202303769

transformations matrix and ChArUco Markers' transformation matrix. The user interface shows in Figure 7.17:



doi:10.6342/NTU202303769

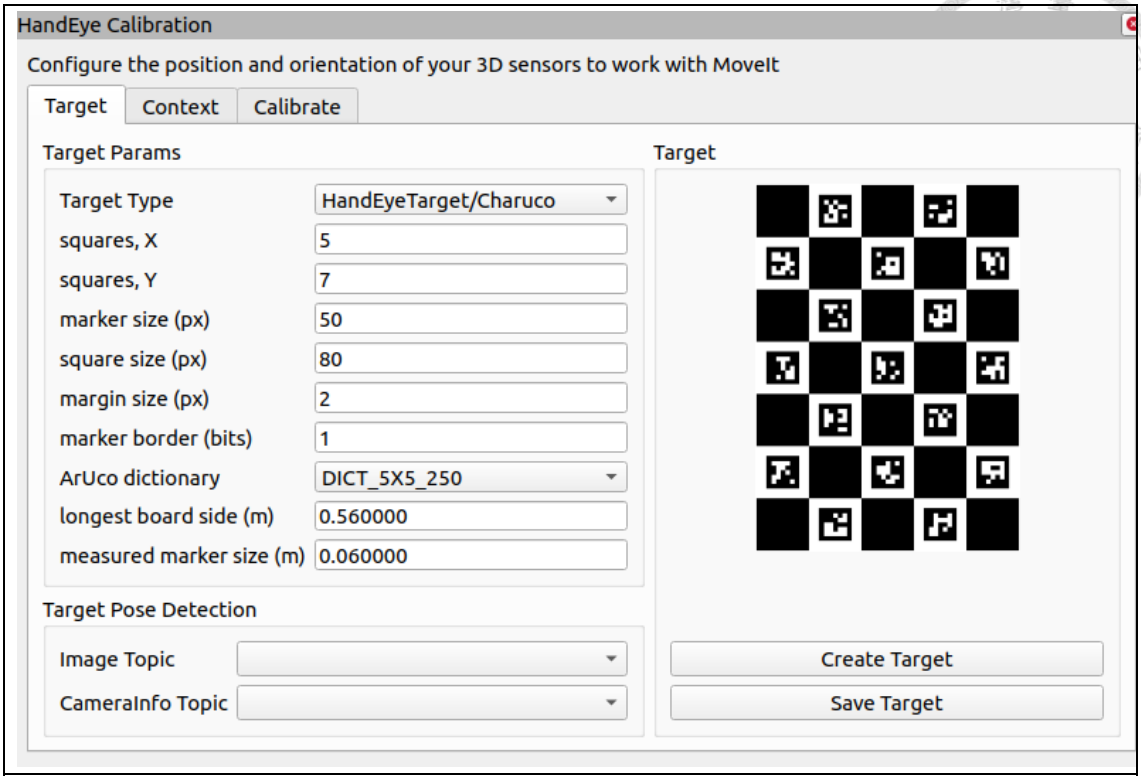


Figure 7.17 Hand-Eye calibration user interface in Rviz

By capturing at least 15 samples of Charuco board, the precise relationship between robotic end-effector and camera frame will be determined.

7.2.3 Plant Pointcloud Filter Experiment

In the chapter 6, the pointcloud filter principle was introduced. However, in the real-world experiment, different illumination conditions correspond to different HSV (Hue, Saturation, Value) range. We firstly observe the HSV of plant pictures to roughly get the range of HSV. The following Figure 7.18, 7.19 shows the HSV distribution of different pictures:

doi:10.6342/NTU202303769

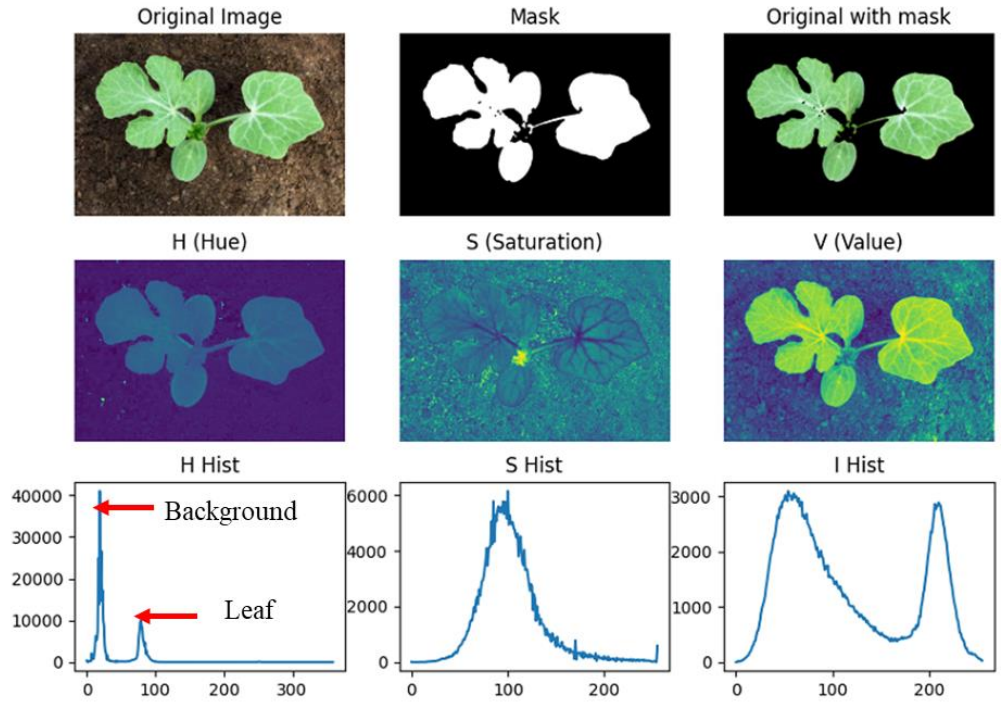


Figure 7.18 Example of HSV filter I (Image)

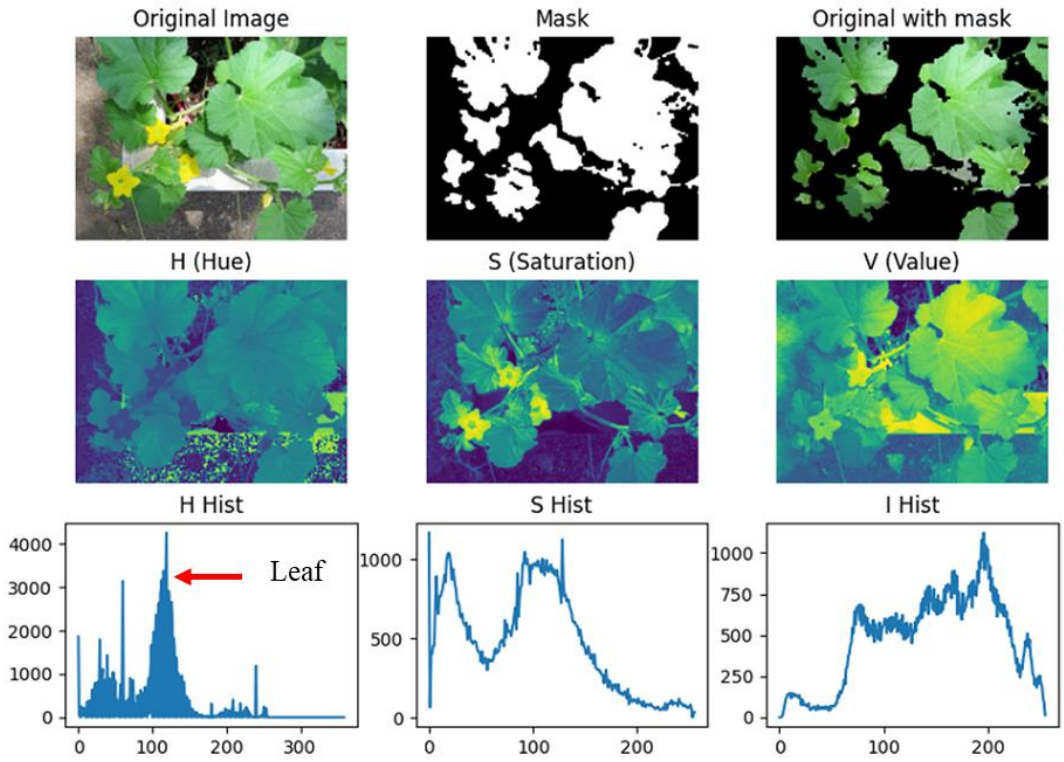


Figure 7.19 Example of HSV filter II (Image)

doi:10.6342/NTU202303769

The hue value of green leaf ranges from 90 to 150 where the maximum value is 255. In the PCL, the value of Hue, Saturation and Value set as follows:

Table 7.4 HSV ranges

	Lower Limit	Upper Limit	Maximum Value
Hue	30	90	360 Deg
Saturation	0.15	1.0	1.0
Value	0.15	1.0	1.0

The following figure shows the original pointcloud captured from depth camera where the non-plant and plant point are merged together.

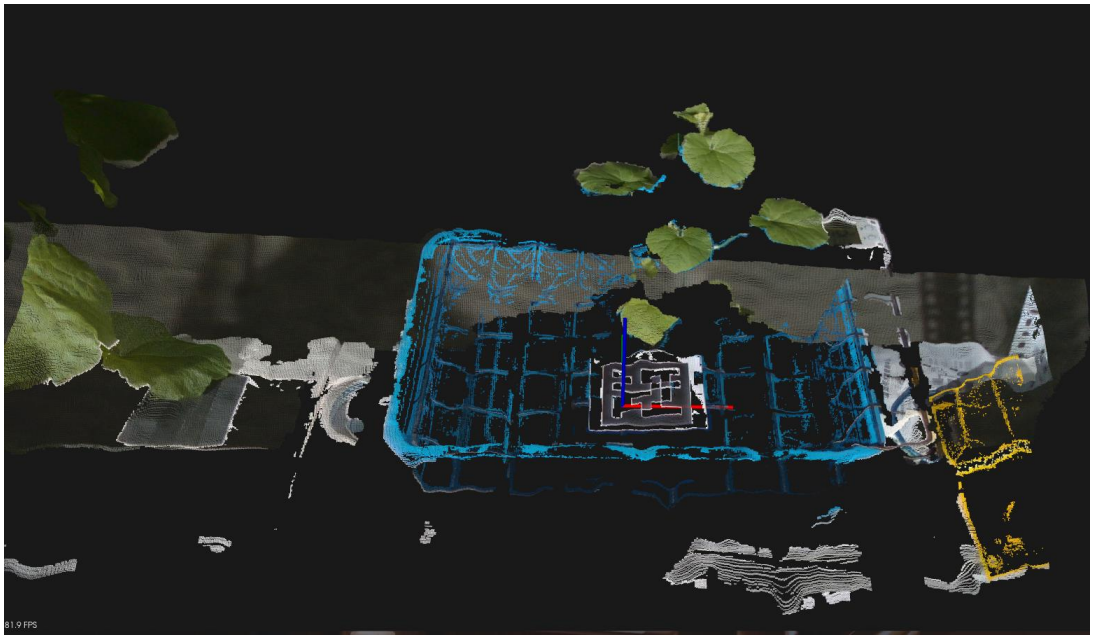


Figure 7.20 Original pointcloud

However, in the real-world scenario, the plants are cultivated side by side with fixed interval. To extract single plant from entire pointcloud file, the algorithm needs to know the location of each observed plant. This is the localization problem described Section

doi:10.6342/NTU202303769

5.1. Although the dimension of the plant is changing with plant growing, a fixed 3-dimensional ROI is deployed. The algorithm keeps $\pm 0.2\,m$ of the x and y axis. The interesting plant height is 1.56 meter which is determined by the height of the cultivating shelf. The following figure shows the plant exclusively.

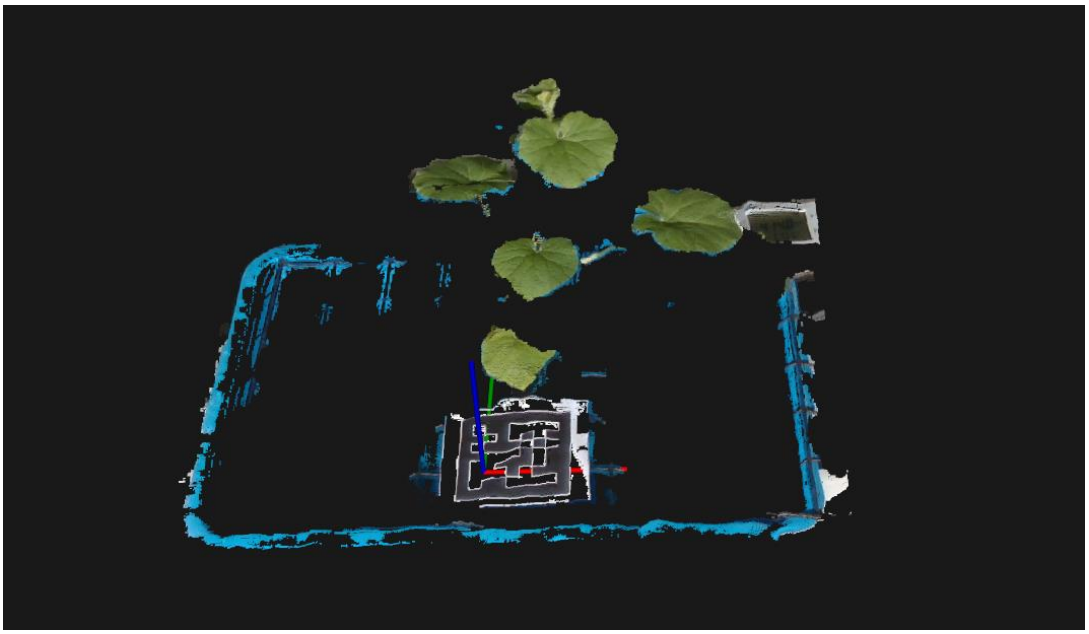


Figure 7.21 ArUco filtered plant pointcloud

doi:10.6342/NTU202303769

After apply the HSV filter, the green part of pointcloud is reserved in the processed pointcloud, shown in Figure 7.22:

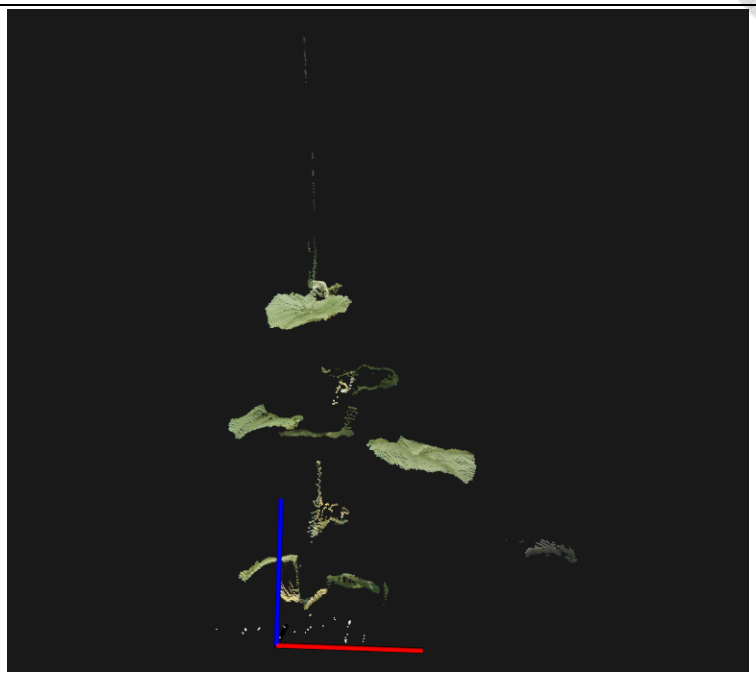
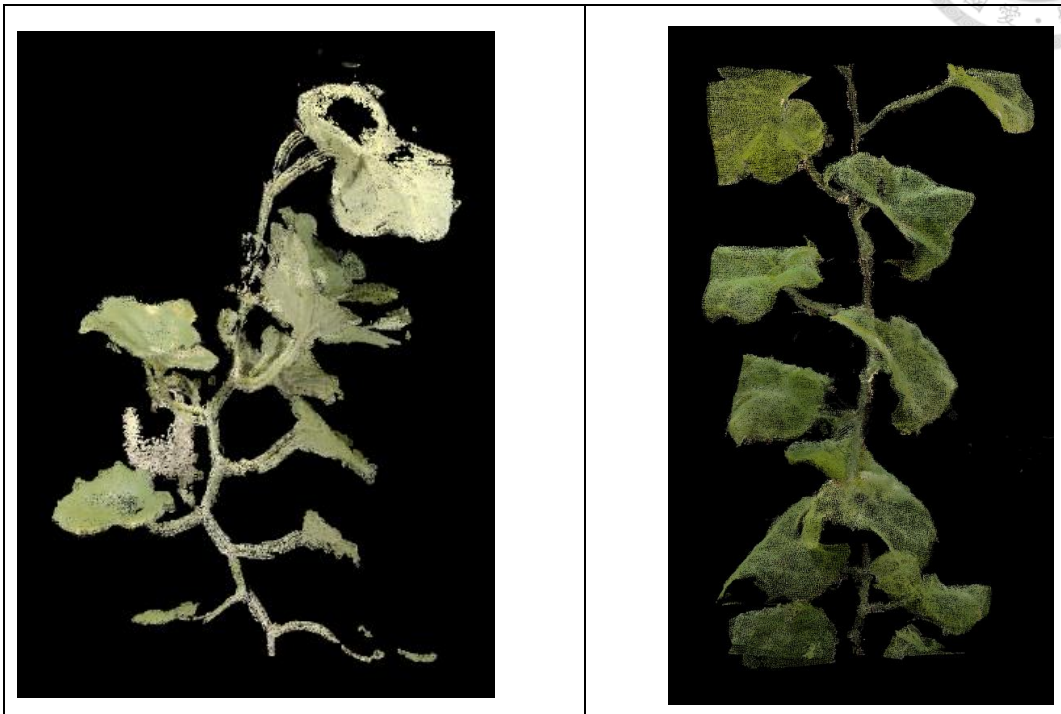


Figure 7.22 The filtered exclusive plant pointcloud

doi:10.6342/NTU202303769

The following Figure Set shows two reconstructed pointcloud which the plant filter plays an important role in the system.



(a) (b)

Figure 7.23 Example of reconstructed plant from filter algorithm

7.3 Laboratory Experiment

Although the ideal experiment procedure is already shown in Figure (), the preparation of the experiment and the related data recording needs more consideration. The experimental plant is cultivated by the NTU farm and the cultivating period is fixed. Usually we pick one to two plant from the greenhouse to the laboratory for **programming development purpose** and **algorithm performance verification**.

doi:10.6342/NTU202303769

We are selected five plant in the final laboratory procedure verification stage from the NTU farm. The plant will be placed on the same cultivating platform to keep the consistency between the laboratory and greenhouse.

Before the experiment, a fixed interval and size of the ArUco board is precisely plotted using the Adobe Illustrator and printed using the original scale. In this case, a horizontal symmetric ArUco marker array is used for plant localization, which the 2-dimensional marker matrix will significantly increase the marker detection precision. The ArUco marker array shows as follows:

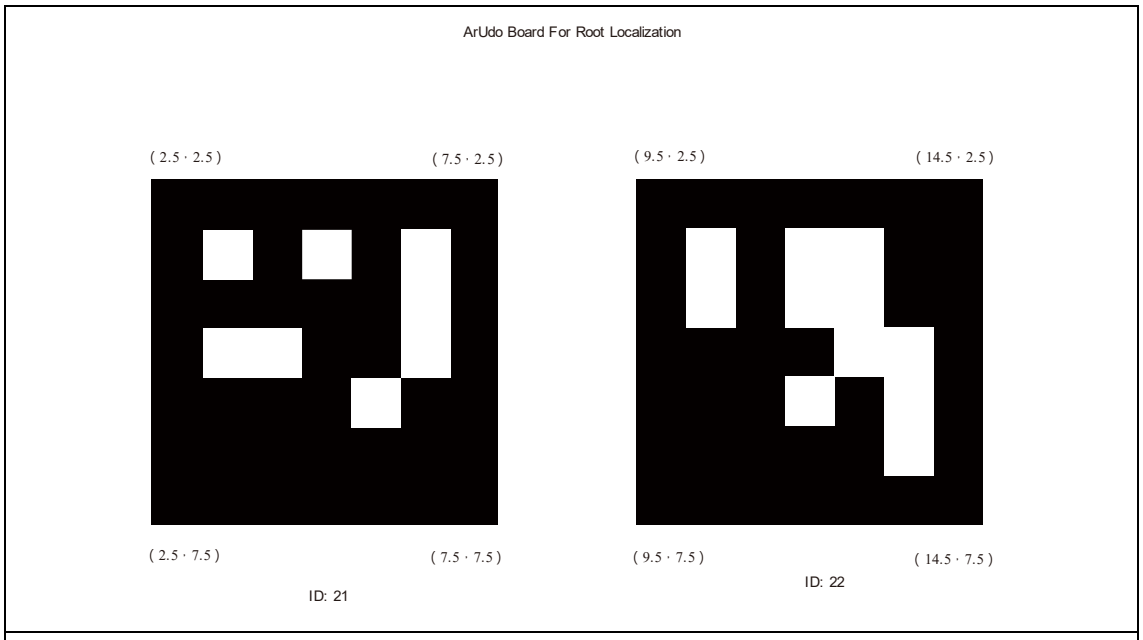


Figure 7.24 ArUco marker array board

After fixes the ArUco marker board behind the plant, we have to manually measure the distance between the marker and the plant root. This information will be used for plant localization procedure. To simplify the measurement difficulty, the coordination of plant root and ArUco board are

doi:10.6342/NTU202303769

Besides the mentioned preparation works, the related variables will be recorded. This not only provide experimental data but know the real procedure during this experiment. The following table shows all variables which will be recorded, including the variable name, type, frequency and related description. The variables are recorded using rosbag except the outside camera.

Table 7.5 Recorded variables

Variable Name	Type	Frequency	Description
JointState	sensor_msgs/jointstate	60 Hz +	The joint state message of the robotic manipulator. This message contains the timestamp and related joint position, velocity and effort.
Robot Pose	geometry_msgs/Pose	60 Hz +	The end-effector pose (xyzrpy) information vs. time.
Procedure State (PS)	Customized message	TBD	This message records the event that this program reaches.
Realsense Image	sensor_msgs/image	Discrete	This is the image message from the mounted camera in the robotic manipulator.
Outside Camera	Video Stream	30 Hz +	This video stream represents this procedure from a third-part viewpoint.
Constructed plant pointcloud	sensor_msgs/pointcloud2	Discrete	This message records the reconstructed plant pointcloud under different procedure time.

doi:10.6342/NTU202303769

Occluding Region	Customized message	Discrete	This message contains the low stem density region.
Stem Pointcloud	sensor_msgs/pointcloud2	Discrete	This message records the extracted stem pointcloud.
Extracted Height	Float	Discrete	This message only contains in Stage 1. The detected plant height changes when the model is reconstructed iteration by iteration.
ArUco Pose	Customized message	Discrete	This message contains the detected ArUco Marker / Board Pose and timestamp.

Consequently, we start the TM5-900 robotic manipulator system, ROS driver toolkit and a visualization tool called Rviz. In the Rviz, the move group toolkit is also loaded. This ROS-related platform / APIs provides necessary collision-free motion planning, forward and inverse kinematic solvers and other functions.

Procedure and Algorithm Verification

Figure 7.25 shows the joint states of the robotic manipulator with respect to time. The highlighted trajectory represents the movement from random position to standby position. There are two vertical lines which represent two critical finishing time: robot arm moving from random position to standby position and the marker found time. The blue line represents the readings from first joint value which applies a large line weight. The pure blue and red horizontal line represent the start and end of the first joint state. The robotic manipulator jogging first joint from -1.57 rad to 1.57 rad to find the marker.

doi:10.6342/NTU202303769

All of other joints except joint 1 keep the same value during the marker finding stage. At 29.76 seconds, the marker is found and the robot is asked to stop the searching movement.

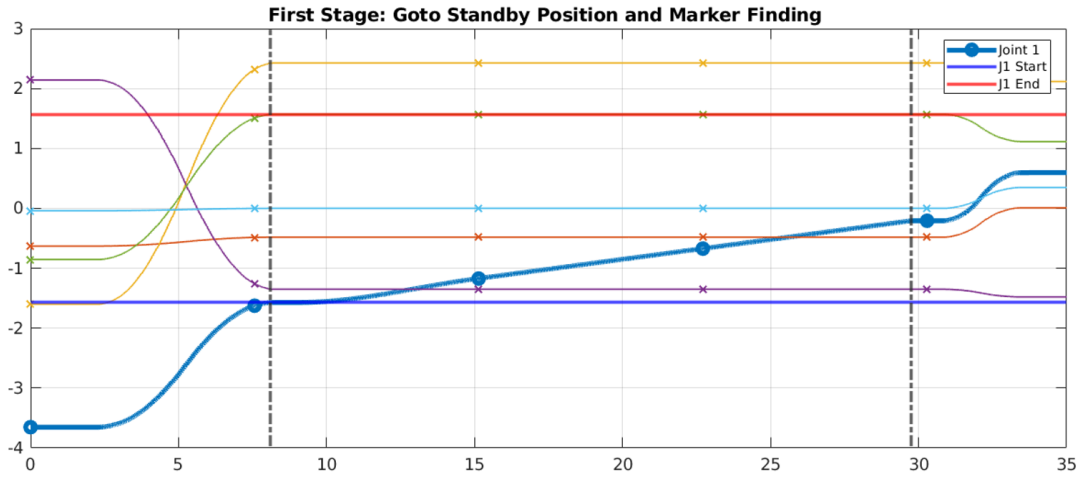


Figure 7.25 First Stage: From random position to standby position

The visualization of these two steps in 3d space shows in Figure 7.26, where the magenta points and blue points means the “Go to Standby Position” and “Marker Localization” steps.

doi:10.6342/NTU202303769

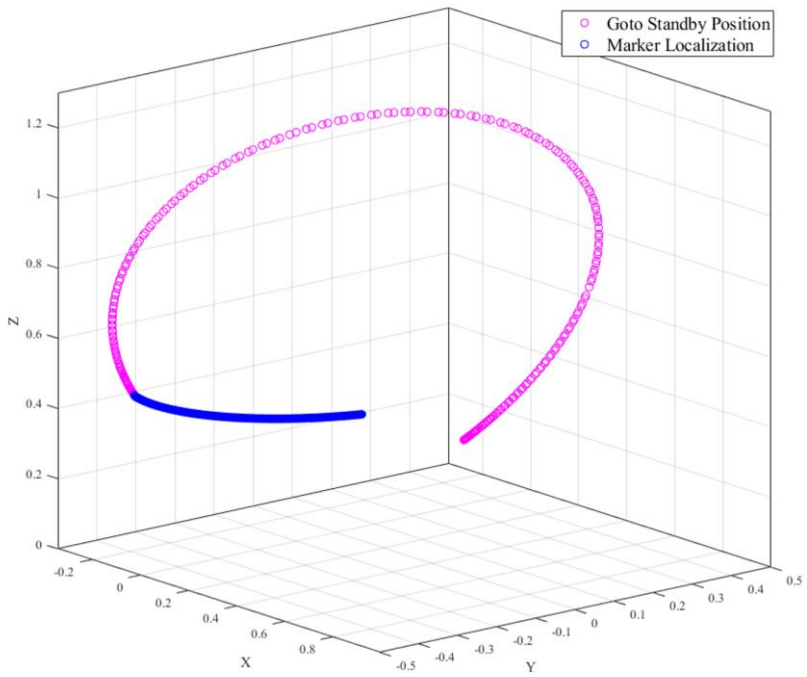


Figure 7.26 The 3D space illustration of two steps in first stage

Figure 7.27 shows the detected ArUco marker at about 29.76 sec.

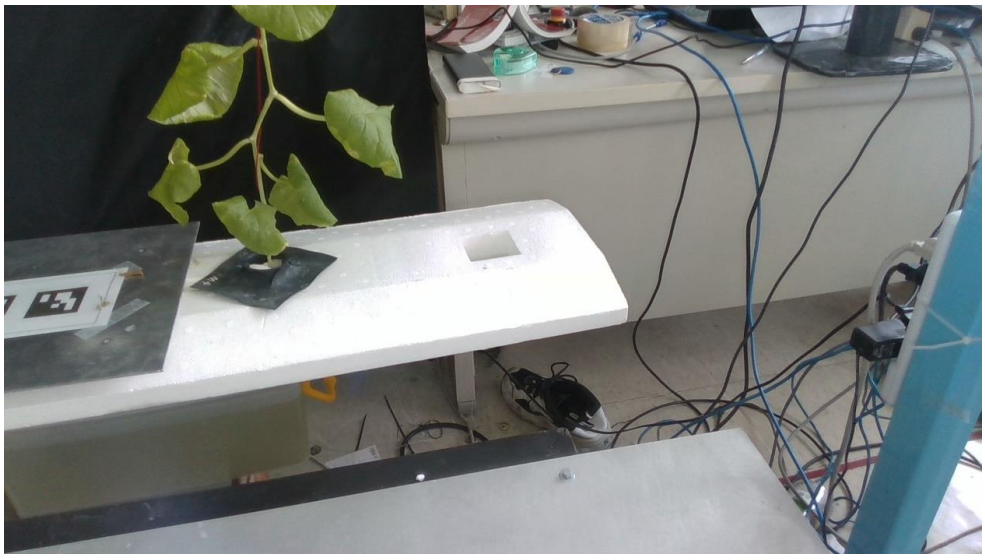


Figure 7.27 The system just detects the ArUco marker board

After the marker's position is found, the robotic manipulator will plan a closer view to re-scan the marker for accurate readings. Figure 7.28 describes the movement of getting

doi:10.6342/NTU202303769

closer marker viewpoint. The first and second figure in Figure 7.28 shows the movement of the camera from far viewpoint to a closer viewpoint. In the third subfigure in Figure 7.28, the ArUco marker was detected ahead 29.76 second, because the latency exits between application layer and arm controller level. The marker readings between 29.76 seconds to 30.65 seconds jitters because of the long distance between camera and ArUco marker. At 30.65 to 33.47 second, the robotic system is asked to move closer to the ArUco marker for better readings.

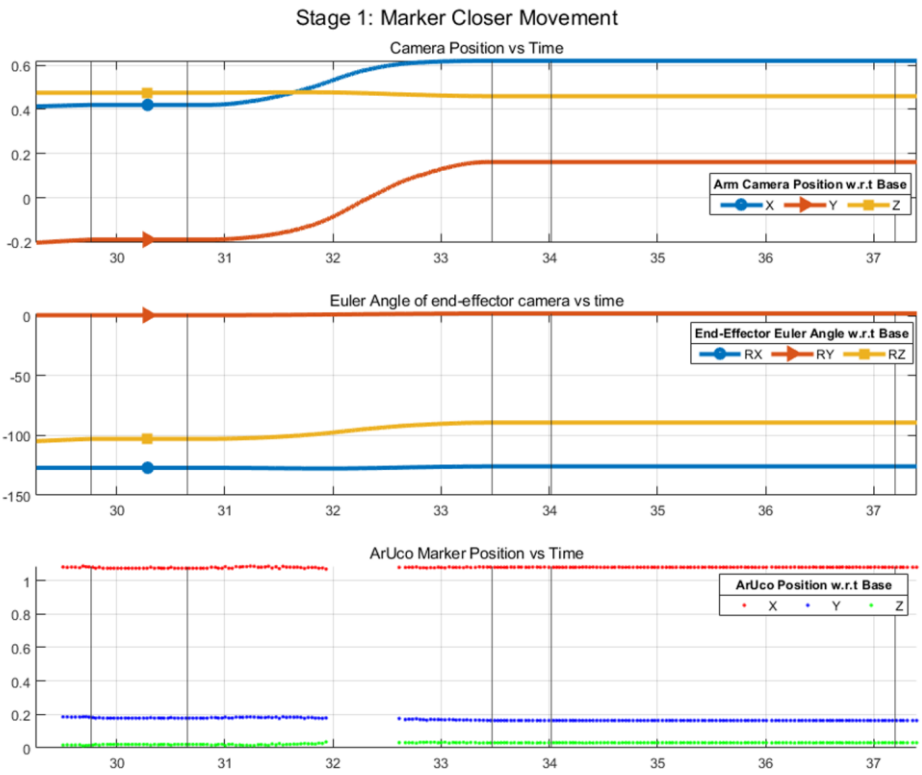


Figure 7.28 Marker closer movement

The Table 7.6 shows the description timestamp in Figure 7.28.

doi:10.6342/NTU202303769

Table 7.6 Time and event					
	29.76 (s)	30.65 (s)	33.47 (s)	34.01	37.19 (s)
Event	Marker Found	Move to Closer	Finished Moving	Start Averaging Readings	Finished Averaging Readings

The Table 7.7 represents the difference between the reading from far viewpoint and that from closer viewpoint. From Table 7.7, the closer viewpoint standard derivation readings did decrease compare that of far viewpoint.

Table 7.7 Statistical analysis of ArUco readings						
	Average Value			Standard Derivation		
Far	1.0746	0.1790	0.0201	0.0025	0.0015	0.0017
Closer	1.0791	0.1639	0.0310	**0.0004**	**0.0004**	**0.0004**

Figure 7.29 shows the “move to closer viewpoint” movement in 3D space where the green points represent this movement. The yellow point is the ArUco marker position under robotic base frame.

doi:10.6342/NTU202303769

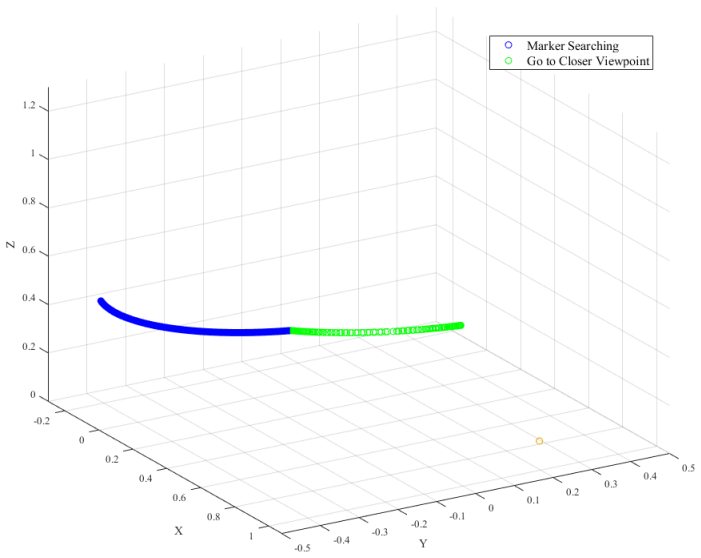


Figure 7.29 “move to closer viewpoint” movement in 3D space

Also, at 33.47 seconds, the image from camera shows as follows. The marker is strictly located in front of the camera.

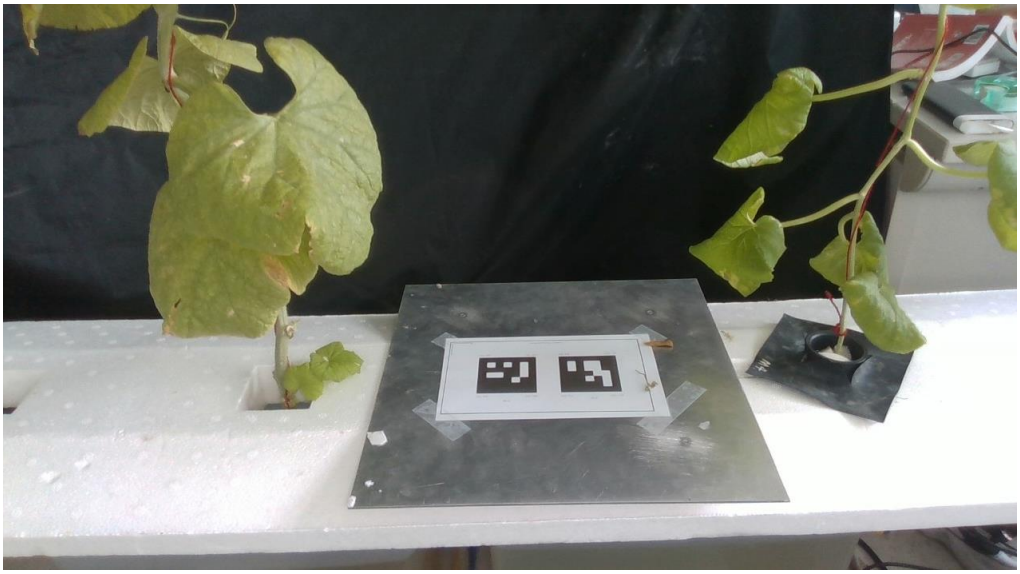


Figure 7.30 The system just detects the ArUco marker board

doi:10.6342/NTU202303769

After taking ten samples of ArUco pose, the robotic system plans several viewpoints just head-on the observed plant to reconstruct it.

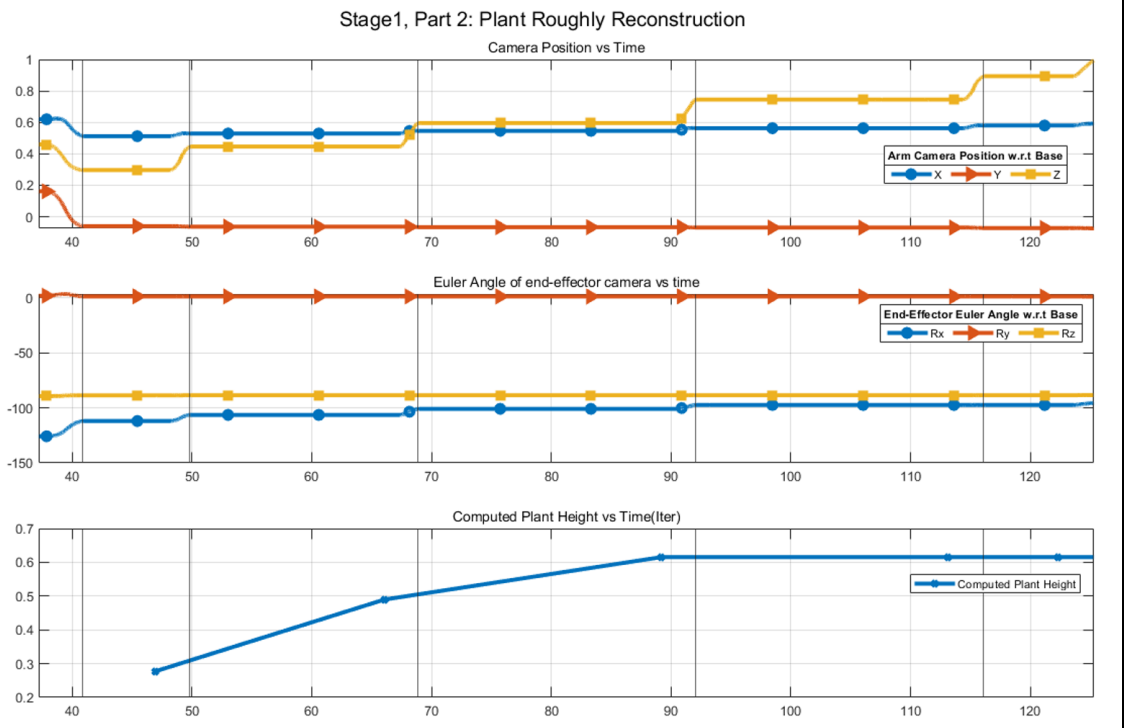


Figure 7.31 Stage 1: Plant roughly reconstruction period

In Figure 7.33, from the first subgraph, 5 different viewpoints have been executed, which are vp_1 to vp_6, at 40.84, 49.8, 68.8, 92.0, 116.1 and 125.3 seconds, respectively. The last viewpoint vp_6 does not contribute the plant reconstruction. Therefore, the plant height does not increase after this iteration. The computed plant height is also given in the third subgraph. The final reading of this plant height is 0.615 meter, which is close to the hand-measured height, 0.62 m.

The movement in 3D space shows in Figure 7.32:

doi:10.6342/NTU202303769

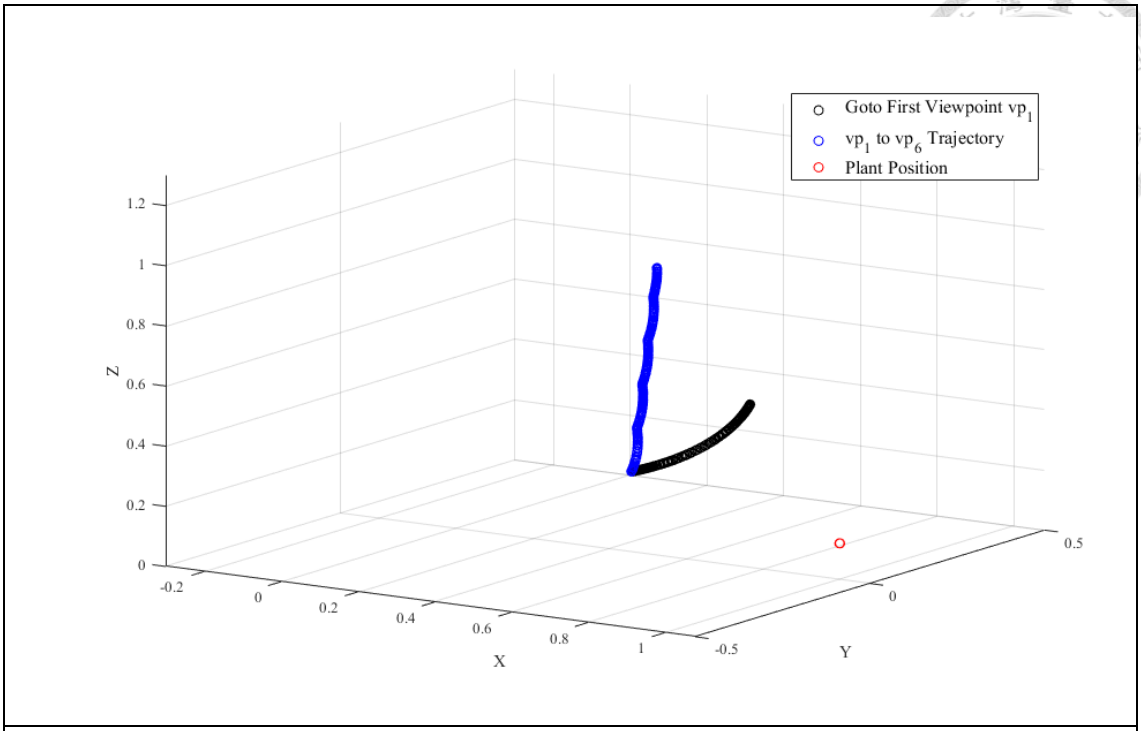


Figure 7.32 The trajectory of head-on viewpoint, and plant reconstruction

Figure 7.38 shows a set of images captured at different viewpoint. The observed plant is tied using red cable, and the left side plant is already been filtered.

(a) vp_1

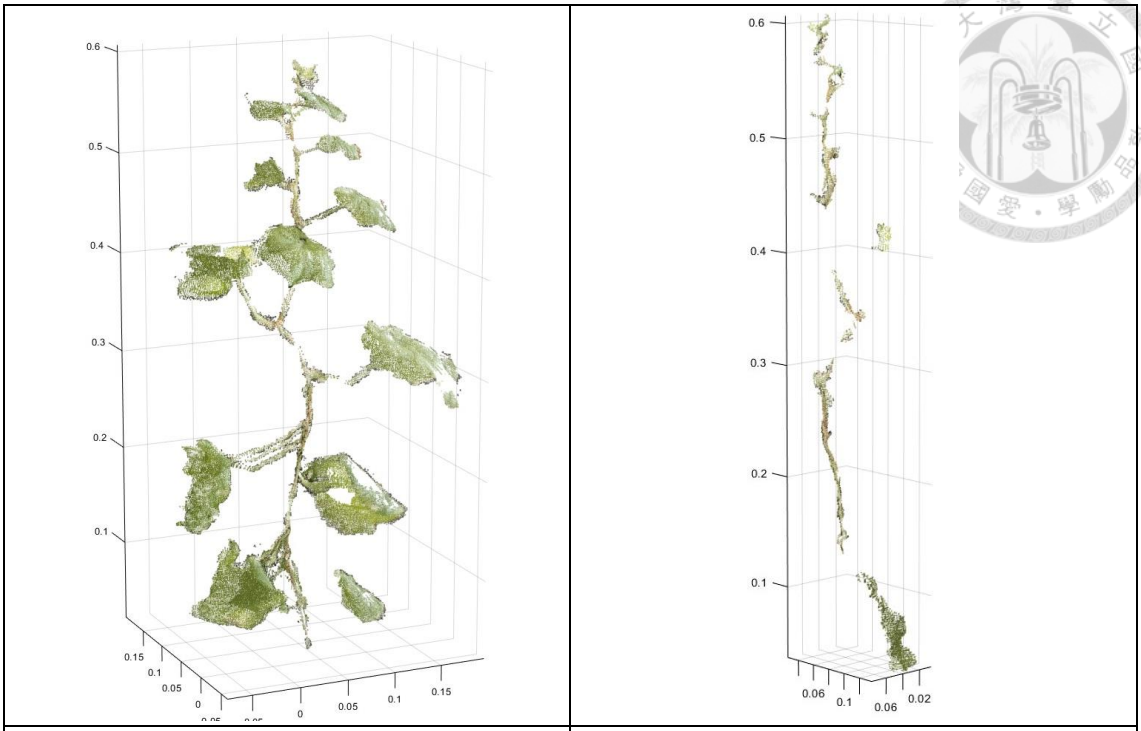
(b) vp_2

doi:10.6342/NTU202303769

(c) $vp3$	(d) $vp4$
	
(e) vp_5	(f) vp_6
Figure 7.33 Six different head-on viewpoint	

Until now, the first plant reconstruction procedure is finished. The plant pointcloud is reconstructed and the stem of this plant will be extracted using SEA. Figure 7.34 (a) and (b) shows the reconstructed plant and extracted stem pointcloud. In this case, the extracted stem contains the unrelated part: leaf. This is because from this start searching point, the result containing the leaf get a higher score. In another word, the density of stem needs to concentrate.

doi:10.6342/NTU202303769



(a) Reconstructed plant | (b) Extracted stem

Figure 7.34 Reconstructed plant and extracted stem pointcloud

From the Figure 7.39(b), the distribution of stem pointcloud are analyzed in Figure 7.40.

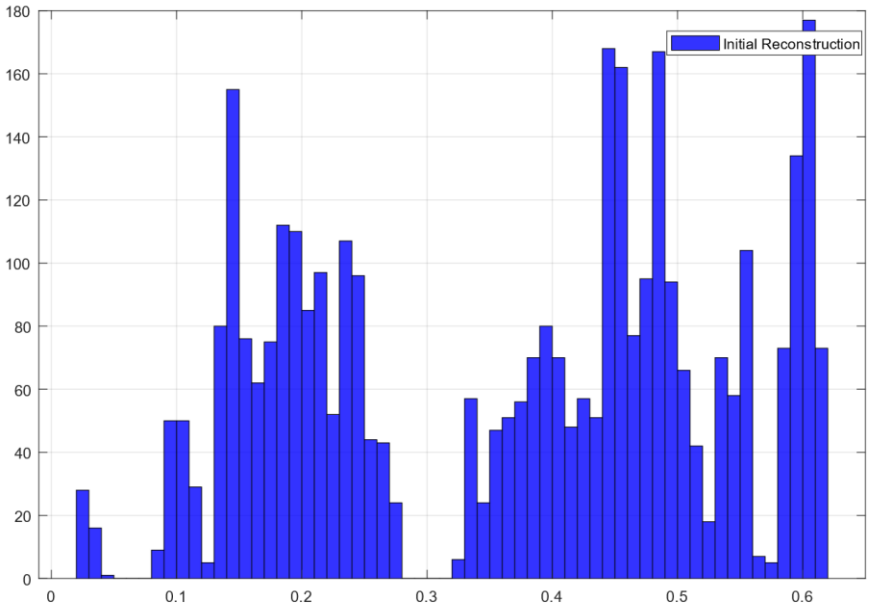


Figure 7.35 The histogram of initial reconstructed stem pointcloud

doi:10.6342/NTU202303769

Without consideration of lower and higher region of stem, the main missing stem is located at about 0.3 meter. The occluding analyzing service outputs four regions shown in the following table which the system only concerns the second region at this time. Because, the re-planning viewpoint to lower region is limited by robotic platform. The third and fourth are low density region and the length of the region is small. So, they are ignored.

Table 7.8 The result from occluding checking module

Region Number	From	To
1	0.043127220	0.083127216
2	0.28312722	0.32312721
3	0.52312720	0.53312725
4	0.56312722	0.57312721

The procedure then plans a circular viewpoint sets around the occluding region shown in Figure 7.36.

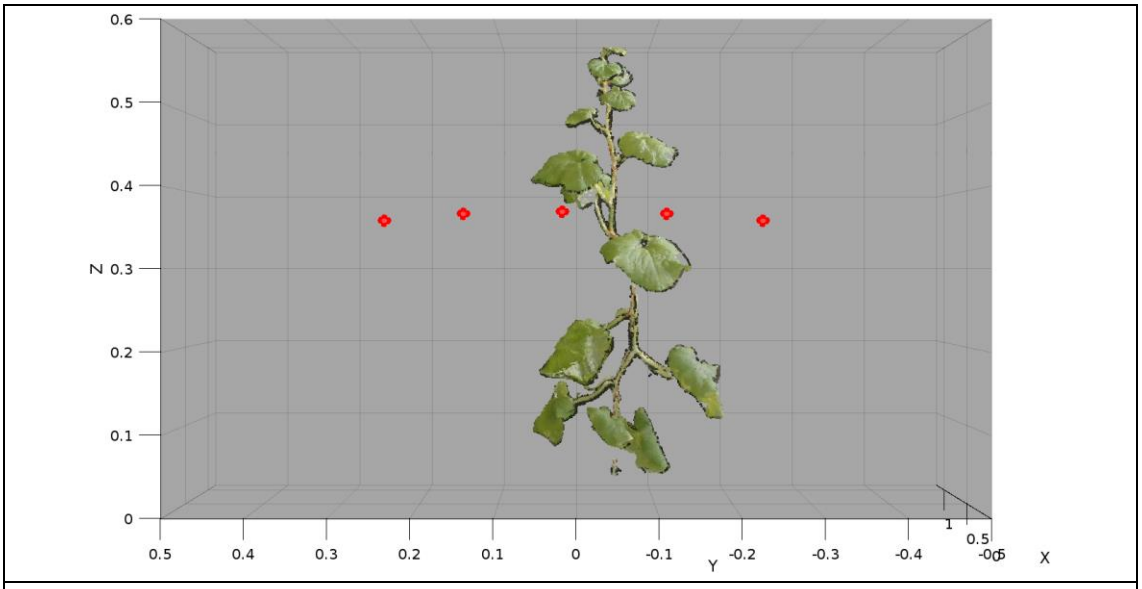


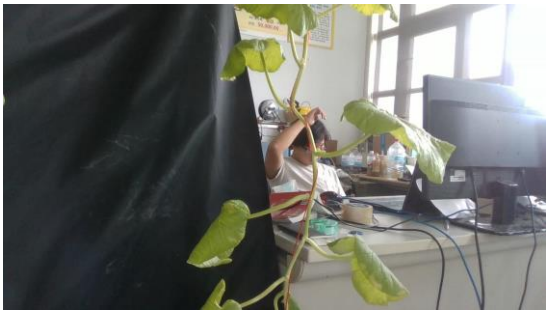
Figure 7.36 Planned circular viewpoint

doi:10.6342/NTU202303769

Note: At that time, the plant viewpoint optimizer is not developed. After the optimizer successfully developed, this experimental plant is gone.

Figure 7.37 shows 4 of 5 image captured from camera.

(a) Re-planned viewpoint 1 rvp_1



(b) Re-planned viewpoint 2 rvp_2

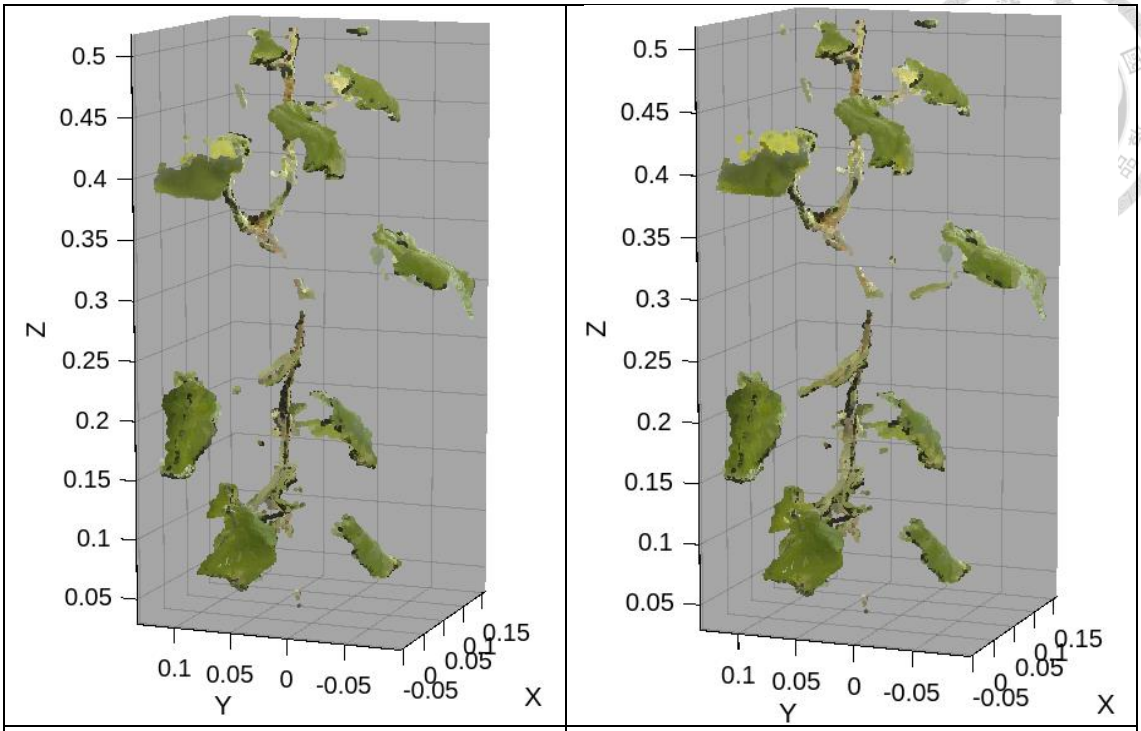
(c) Re-planned viewpoint 3 rvp_3

(d) Re-planned viewpoint 4 rvp_4

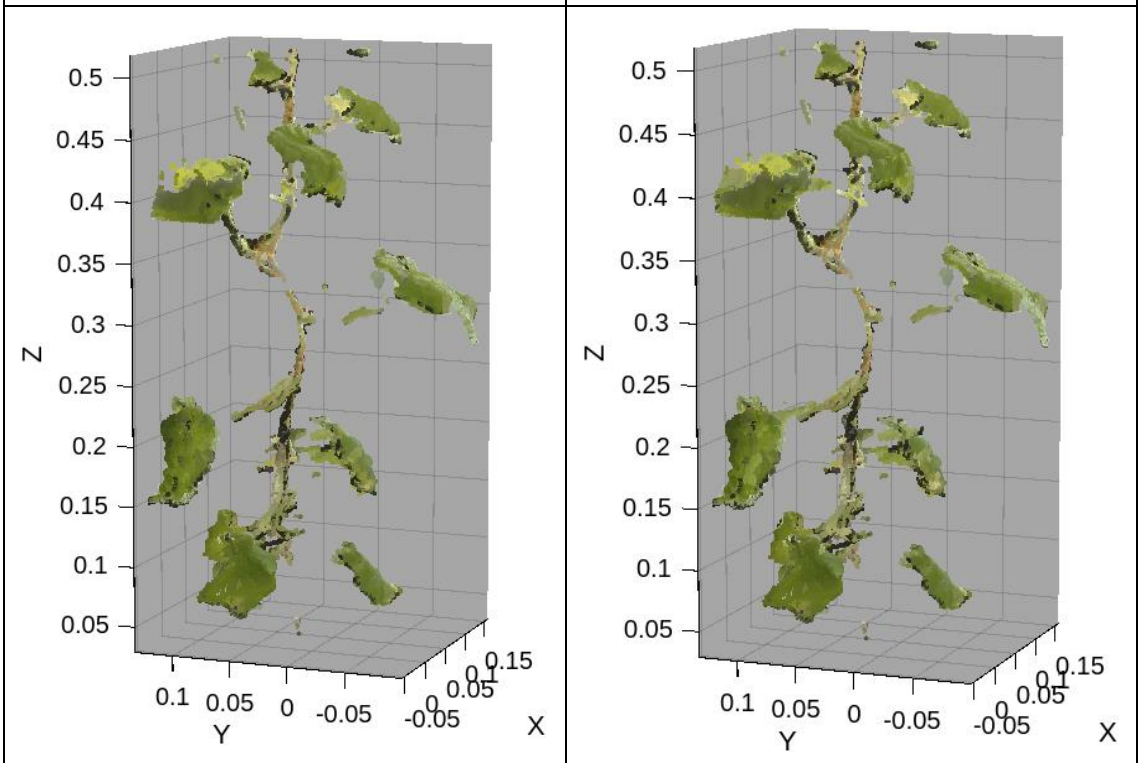
Figure 7.37 Four of five re-planned viewpoints to refine the pointcloud

Also, the refined plant pointcloud after the camera moves to the rvp_1 to rvp_4 shows in Figure 7.38.

doi:10.6342/NTU202303769



(a) Reconstructed plant after rvp_1 (b) Reconstructed plant after rvp_2



(c) Reconstructed plant after rvp_3 (d) Reconstructed plant after rvp_4

Figure 7.38 Four of Five Re-planned viewpoints to refine the pointcloud

doi:10.6342/NTU202303769

The trajectory of Stage 2 shows in Figure 7.39 which describes the five different viewpoints.

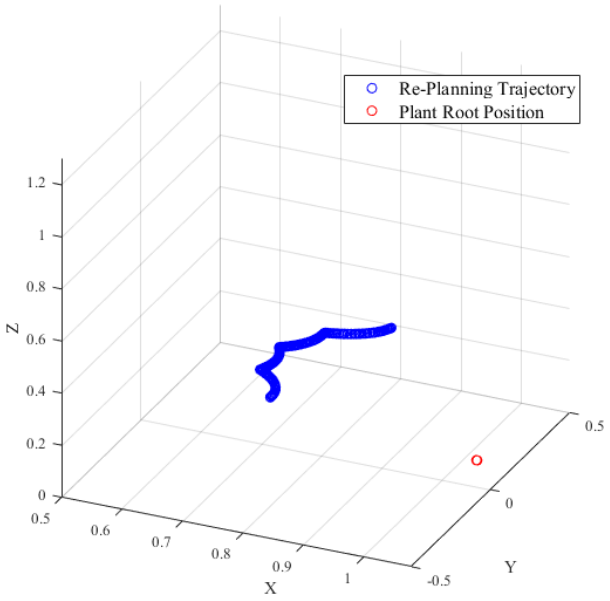
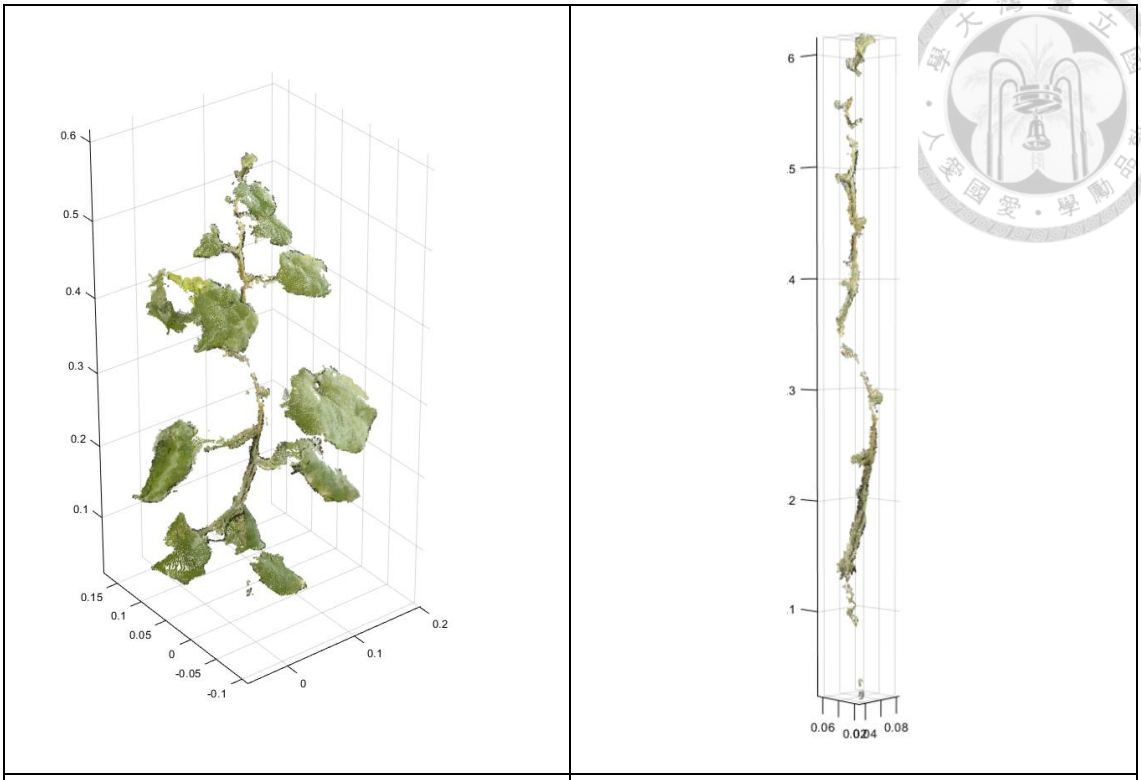


Figure 7.39 3D movement of Stage 2: Refinement

After the refinement, the reconstructed plant and stem pointcloud will be concentrated in exception. The entire plant pointcloud and extracted stem shows in Figure 7.40.

doi:10.6342/NTU202303769



(a) reconstructed plant | (b) extracted stem

Figure 7.40 **Refined** plant and extracted stem pointcloud

Figure 7.41 compares two histograms separately including two stem pointcloud.

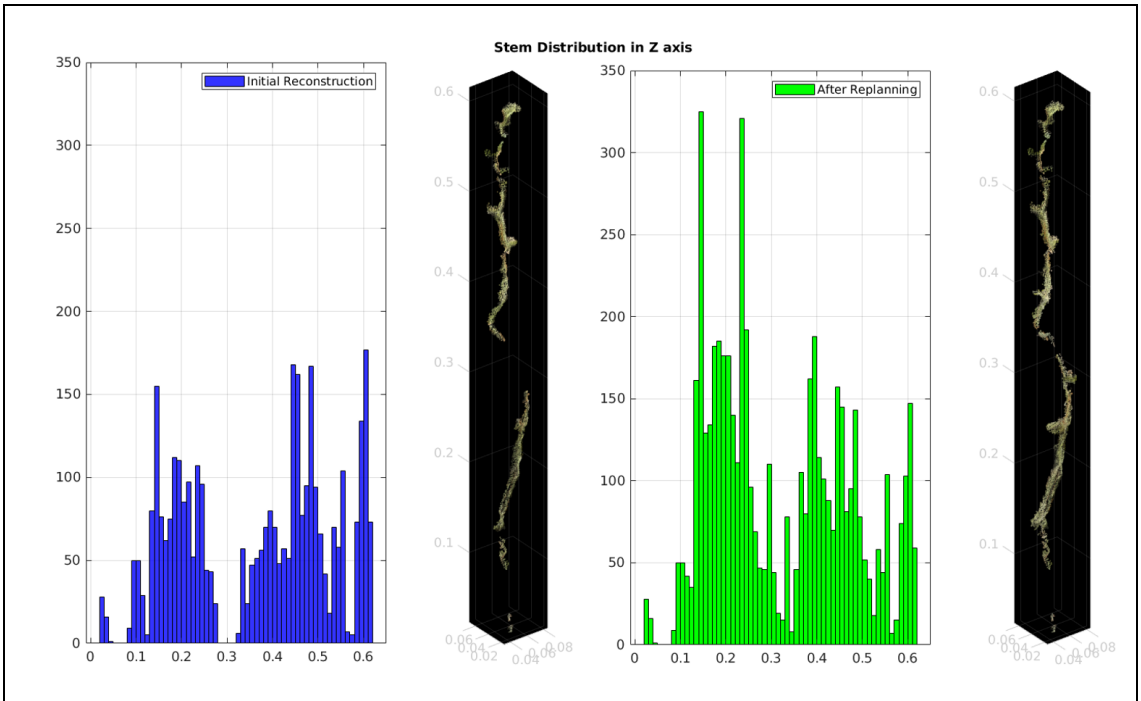


Figure 7.41 The comparison stem pointcloud histogram

doi:10.6342/NTU202303769

The comparison pointcloud histogram of before and after re-planning are put together, and significant difference appears in re-planned occluding region.

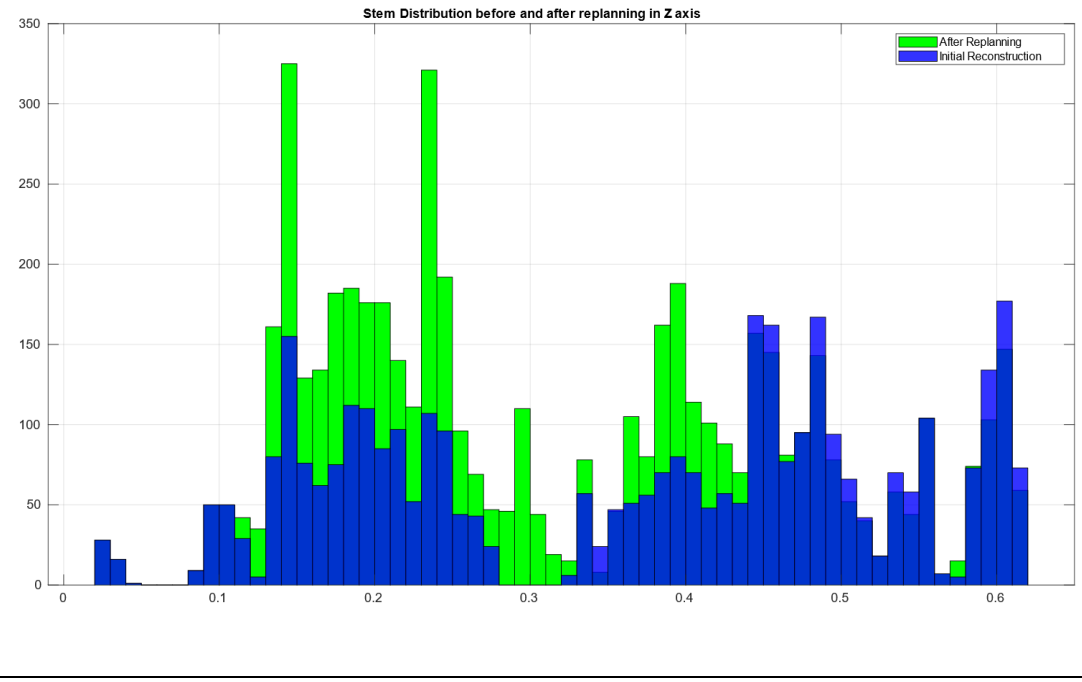


Figure 7.42 The comparison stem pointcloud histogram

Comparing the Figure 7.34 (b) and 7.40 (b), the occluding region are filled in Stage 2 procedure. The two extracted stem score are shows in the Table 7.9.

Table 7.9 Stem score

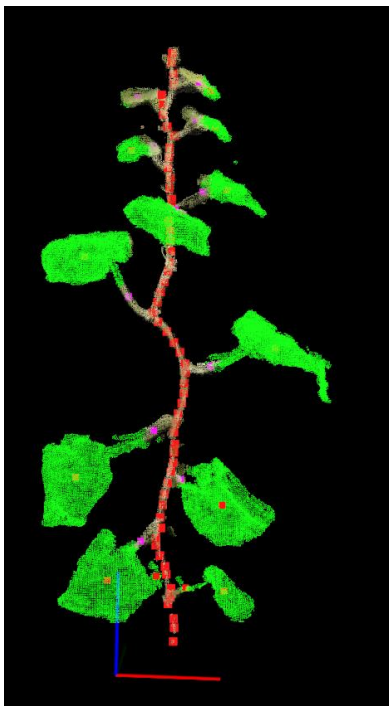
	Before Re-planning	After Re-planning
Score	47	**53**
Improvement	12%	

Plant Segmentation

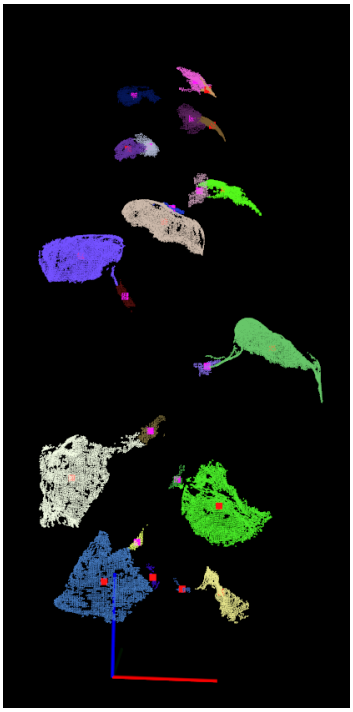
After the system reconstrued an acceptable plant pointcloud, the plant organs will be segmented using the re-implemented algorithm. By using two circles with different radii,

doi:10.6342/NTU202303769

the plant pointcloud could be segmented to stem, petiole and leaves. At this time, the stem is already been extracted and will be used in segmentation algorithm as an input data. Figure 7.43 shows the segmentation result. Figure 7.43 (a) illustrates the extracted stem pointcloud which labelled as red points, the pure green pointcloud is the leaf pointcloud and the original color pointcloud which are petioles. These pointcloud (leaf and petiole) are in an entire pointcloud. A clustering filter called ECE in PCL helps us to separate these pointcloud. Whereas the Figure 7.43 (b) shows the segmented leaves and petioles pointcloud without stem. And the red point represents the centroid of each individual leaf and the magenta point represents the centroid of petiole. Using the fixed circles' radii, the segmentation algorithm in upper region are misjudging the leaf as petiole.



(a) Pointcloud before clustering



(b) Entire pointcloud has been separated to individual parts

Figure 7.43 Post-processing: plant organ segmentation procedure

doi:10.6342/NTU202303769

Figure 7.44 and Figure 7.45 shows the segmented leaves and petioles.

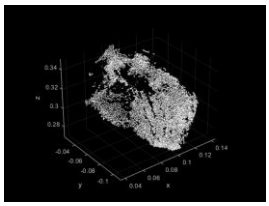
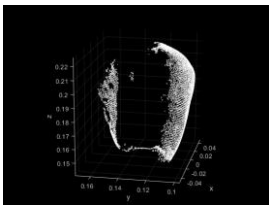
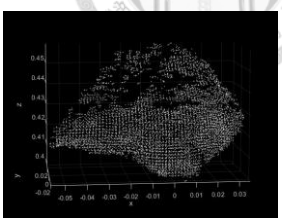
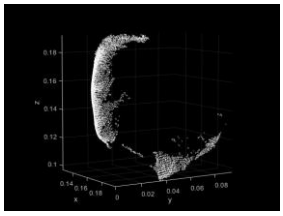
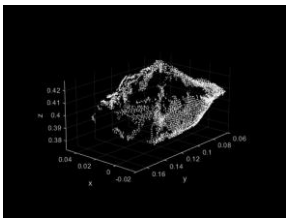
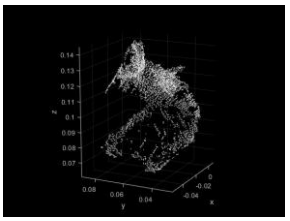
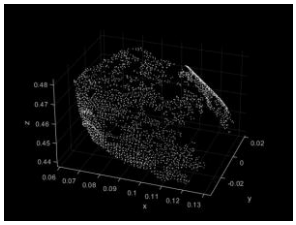
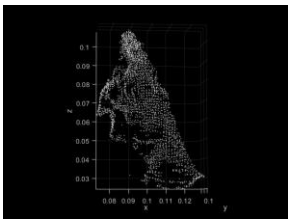
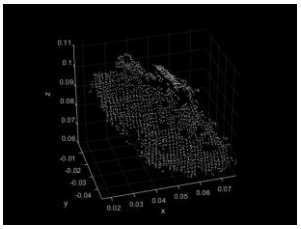
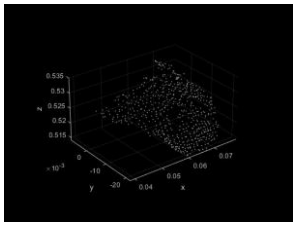
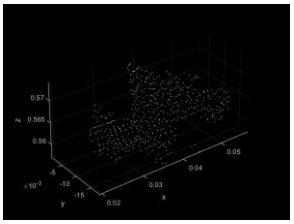
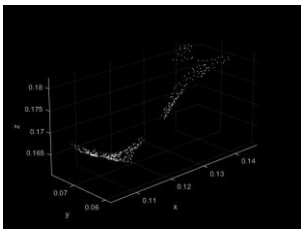
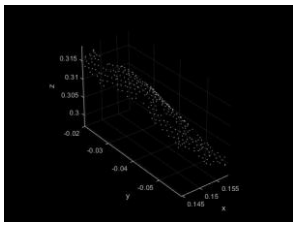
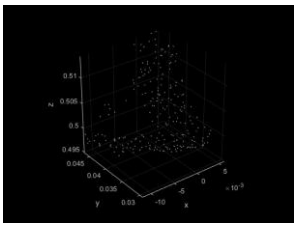
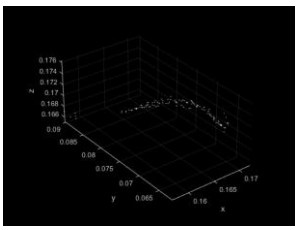
Index	0	1	2
Pointcloud			
Index	3	4	5
Pointcloud			
Index	6	7	8
Pointcloud			
Index	9	10	11
Pointcloud			
Index	12	13	14
Pointcloud			

Figure 7.44 Separated segmented leaf pointcloud

doi:10.6342/NTU202303769

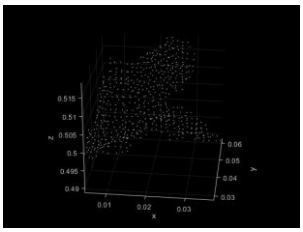
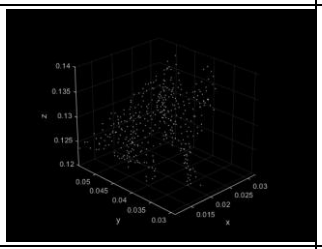
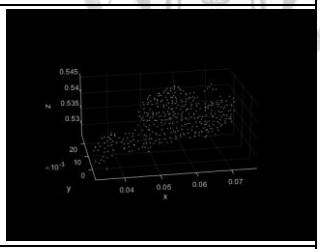
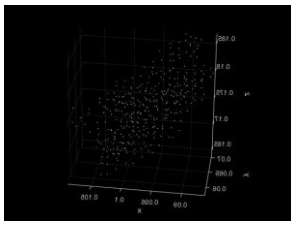
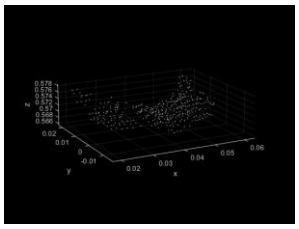
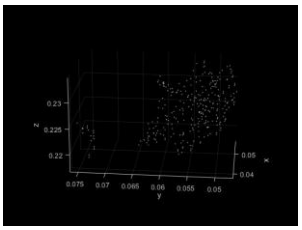
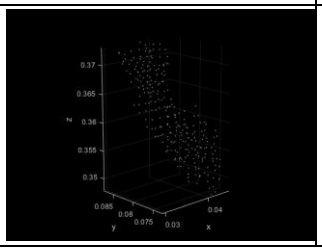
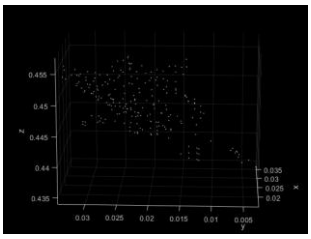
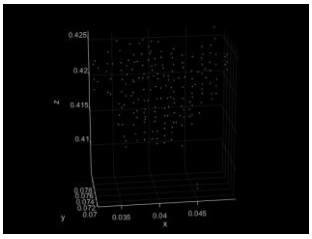
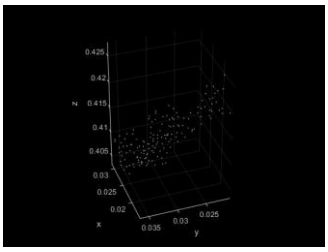
Index	0	1	2
Pointcloud			
Index	3	4	5
Pointcloud			
Index	6	7	8
Pointcloud			
Index	9		
Pointcloud			

Figure 7.45 Separated segmented petiole pointcloud

Without consideration of the top region of this plant pointcloud, the accuracy of the plant organ segmentation algorithm is

$$accuracy = \frac{8+7}{8+7+2+0} * 100\% = 88.2\ \%$$

doi:10.6342/NTU202303769

Table 7.10 Accuracy of organ segmentation algorithm

		Real Result	
		True	False
Reasoned Result	Leaf	8	2
	Petiole	7	0

doi:10.6342/NTU202303769

Chapter 8
Conclusion and future work

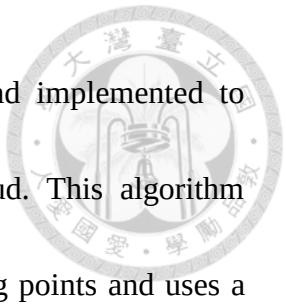
Based on the content of the above chapters, conclusions are made in this chapter on the proposed algorithm, the results achieved and future challenges.

8.1 Conclusion

This thesis presents a comprehensive approach involving various algorithms and procedures for the reconstruction and analysis of cantaloupe point clouds. The primary goal is to develop an efficient system to process and understand the captured point cloud data of cantaloupe plants. Several key contributions and outcomes of this research are discussed below.

Firstly, the paper introduces a novel procedure that incorporates a multi-scale control strategy to reconstruct the point cloud of the target cantaloupe plant. Subsequently, a structural analysis of the point cloud is performed to identify any defects or missing parts. By extracting the defect regions, the robotic system can re-plan a collision-free viewpoint and refine the incomplete areas of the point cloud. This methodology ensures a more accurate and complete representation of the cantaloupe plant's structure.

doi:10.6342/NTU202303769

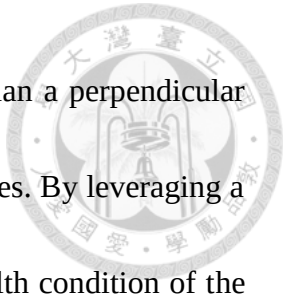


Secondly, a robust stem extraction algorithm is proposed and implemented to identify the stem point cloud from the plant's overall point cloud. This algorithm iteratively analyzes the geometric relationships between neighboring points and uses a non-linear gain function to find the most optimized point-pair. Through this process, the algorithm traverses the point cloud from the bottom to the top, effectively locating the potential stems. The stem candidate with the highest score, computed using Principal Component Analysis (PCA), is chosen as the stem point cloud. The algorithm exhibits high robustness and is well-suited for cantaloupe plant data.

Furthermore, the thesis explores the use of different adjacent radii of the Stem Extraction Algorithm (SEA) to find missing parts and further enhance the extracted stem point cloud. This approach serves as a verification method, providing insights into imperfect regions within the point cloud and enabling the system to improve the quality of the data. Additionally, the algorithm generates a normalized confidence score, which serves as an indicator of the stem point cloud's reliability and accuracy.

In the post-processing stage, the system performs segmentation of various plant organs, including leaves, petioles, and stems, utilizing the previously extracted stem point cloud. This segmentation step helps in understanding the individual components of the cantaloupe plant, facilitating further analysis and assessment.

doi:10.6342/NTU202303769



Moreover, the segmented leaf point cloud is employed to re-plan a perpendicular viewpoint, enabling dedicated scanning of the top surface of the leaves. By leveraging a pest detection algorithm proposed by another research team, the health condition of the leaves can be evaluated, which is crucial for agricultural applications.

Lastly, for diseased leaves, the system utilizes the provided segmented petiole position to plan the trajectory of the robotic manipulator's end-effector. This enables precise movement to the target position, allowing for the petiole to be cut and managed accordingly.

Notably, the stem extraction algorithm developed in this research is not limited to cantaloupe plants alone. It can also be adapted and applied to analyze the stem point clouds of climbing plants such as cucumber and kidney beans, broadening its potential applications in various agricultural contexts. Overall, the proposed algorithms and procedures contribute significantly to the field of plant point cloud analysis and have promising implications for future agricultural practices.

8.2 Future Works

In this paper, a robust plant reconstruction process and a stem extraction algorithm are proposed, aiming to achieve a relatively complete plant point cloud that facilitates deeper analysis based on the available point cloud information. With the successful

doi:10.6342/NTU202303769

implementation of these methods, a more detailed understanding of plant structures and growth patterns can be obtained.

However, due to time limitations, some aspects of the research are still in the planning stage, and future works are outlined below:

One potential future work involves utilizing the plant point cloud data to extract the positions and estimate the sizes of the fruits. By determining the specific nodes on which the fruits are growing, this information will provide valuable insights into the status of fruit growth. Researchers and cultivators can then adjust the cultivation program based on observed trends, leading to improved crop management.

Another area of interest for future research is developing algorithms to detect the ripeness of the plant's fruits using their known positions. By integrating this capability into the robotic system, the harvesting process can be optimized. A series of actions can be planned to pick the plant's fruits efficiently. Moreover, a multi-arm cooperation system could be employed, with each arm responsible for cutting the T-shaped tissue that supports the fruit. This would streamline the fruit picking process, making it more automated and productive.

Additionally, the paper discusses the possibility of leveraging the plant point cloud information to automate the process of winding the apical meristem around the vertical guide string. If the system detects that the plant's top has become detached from the

doi:10.6342/NTU202303769

traction rope, it can plan the path online and automatically rewind the plant tissue that has deviated from the traction rope. This automated intervention would reduce the need for manual labor, further enhancing the efficiency of plant growth and management.

By pursuing these potential future works and integrating advanced algorithms with robotics, this research aims to revolutionize agricultural practices. The ultimate goal is to contribute to increased efficiency, productivity, and sustainability in the cultivation of various plant species, positively impacting the agricultural industry and food production.

doi:10.6342/NTU202303769

doi:10.6342/NTU202303769

References

[1 International Labour Organization 1999]. Safety and health in agriculture. p. 77. ISBN 978-92-2-111517-5.

[2 USDA 2021] U.S. Department of Agriculture, Economic Research Service. Farm Sector Income & Finances: Farm Sector Income Forecast, December 1, 2021.

[3 Trading Economics 2022] India GDP From Agriculture | 2022 Data | 2023 Forecast | 2011-2021 Historical | Chart (tradingeconomics.com)

[4 Astill 2020] USDA ERS - Food Loss: Why Food Stays On the Farm or Off the Market

[5 Merit Times 2013] 糧食自給率降至 32.7% https://www.merit-times.com/NewsPage.aspx?unid=326957

[6 FarmBot 2022] FarmBot Genesis https://farm.bot/pages/genesis

[7 Yallappa et al. 2017] M. Veerangouda, D. Maski, V. Palled and M. Bheemanna, "Development and evaluation of drone mounted sprayer for pesticide applications to crops," 2017 IEEE Global Humanitarian Technology Conference (GHTC), 2017

[8 DINO 2022]

DINO vegetable weeding robot for large-scale vegetable farms [website]

[9 Xiong et al. 2020] "An autonomous strawberry-harvesting robot: Design, development, integration, and field evaluation." Journal of Field Robotics 37(2): 202-224.

[10 Dong 2020] Aging of the Agricultural Labor Force and its Solutions

doi:10.6342/NTU202303769

[11 TVBS 2022] 日研發採果機器人 解決農業勞動力短缺危機 | 日本 | TVBS 新聞網

[website]

[12] pollination image 1 3 Methods Of Hand Pollination - NoSoilSolutions

[13] pollination image 2 Increase Your Harvest Using Your Own Hands (agnetwest.com)

[14] pollination image 3 「"Artificial Pollination"」

[15] pollination image 4 Artificial Pollination REF

[16] Manually measure plant height 1 [website]

[17] Manually measure plant height 2 [website]

[18] How to measure light for plants [Website]

[19] Fungal attack stock [Website]

[20] Rose black spot [Website]

[21] Gardener harvesting tomatoes in greenhouse [Website]

[22] Strawberry harvest [Website]

[23] New harvest robot introduced [Website]

[24] Belgian Robot Can Pick Strawberries [Website]

[25] Berry-Picking Robots [Website]

[26] The Deleafing Robot | Priva [Website]

[27] Cucumber deleaf robot [Website]

[28] PlantScreen Robotic XYZ System [Website]

doi:10.6342/NTU202303769

[29] MU has developed robots that help analyze soybean and corn crops [Website]

[30 Wikipedia 2022] Melon - Wikipedia [Website]

[31] High-Wire Cucumber growing compared to the Traditional Umbrella System [Website]

[32 Choi., et al. 2015] "Morphology-based guidance line extraction for an autonomous weeding robot in paddy fields." Computers and Electronics in Agriculture 113: 266-274.

[33 McCool, et al. 2017] "Mixtures of Lightweight Deep Convolutional Neural Networks: Applied to Agricultural Robotics." IEEE Robotics and Automation Letters 2(3): 1344-1351.

[34 Marx, et al. 2012] "Design and application of a weed damage model for laser-based weed control." Biosystems Engineering 113(2): 148-157.

[35 Haug, et al. 2014] Plant classification system for crop /weed discrimination without segmentation. IEEE Winter Conference on Applications of Computer Vision.

[36 Van Evert, et al. 2011] "A robot to detect and control broad-leaved dock (Rumex obtusifolius L.) in grassland." Journal of Field Robotics 28(2): 264-277.

[37 Kumar, et al. 2021] "Design and fabrication of smart seed sowing robot." Materials Today: Proceedings 39: 354-358.

[38 Hassan, et al. 2016] Towards autonomy in agriculture: Design and prototyping of a robotic vehicle with seed selector, IEEE.

[39 Katupitiya, et al. 2007] Systems Engineering Approach to Agricultural Automation: New Developments, IEEE.

doi:10.6342/NTU202303769

[40 David, et al. 2012] Irrigation Control Methods for Wireless Sensor Network, American Society of Agricultural and Biological Engineers.

[41 Better Home Gardens 2022] Coconut palm [Website]

[42 Faiçal, et al. 2014] "The use of unmanned aerial vehicles and wireless sensor networks for spraying pesticides." Journal of Systems Architecture 60(4): 393-404.

[43 MAO, et al. 2020] A Agricultural Spraying and Fertilization Robot based on Visual Navigation. 2020 15th IEEE Conference on Industrial Electronics and Applications (ICIEA).

[44 Ohi, et al. 2018] Design of an Autonomous Precision Pollination Robot, IEEE.

[45 Szegedy, et al. 2016] "Rethinking the inception architecture for computer vision", Computer Vision and Pattern Recognition IEEE Conference on, pp. 2818-2826, 2016.

[46 Gavhale, et al. 2014] "An overview of the research on plant leaves disease detection using image processing techniques." IOSR Journal of Computer Engineering (IOSR-JCE) 16(1): 10-16.

[47 Durmus, et al. 2017] Disease detection on the leaves of the tomato plants by using deep learning, IEEE.

[48 Selvaraj., et al. 2013] "Detection of unhealthy region of plant leaves and classification of plant leaf diseases using texture features." Agricultural Engineering International: CIGR Journal 15: 211-217.

doi:10.6342/NTU202303769

[49 Van Henten, et al. 2006] "An Autonomous Robot for De-leafing Cucumber Plants grown in a High-wire Cultivation System." Biosystems Engineering 94(3): 317-323.

[50 Baidu_HK 2022] 高通量植物表型平台_百度百科 (baidu.hk)

[51 Yandun Narvaez, et al. 2017] "A Survey of Ranging and Imaging Techniques for Precision Agriculture Phenotyping." IEEE/ASME Transactions on Mechatronics 22(6): 2428-2439.

[52 Tisné, et al. 2013] "Phenoscope: an automated large-scale phenotyping platform offering high spatial homogeneity." The Plant Journal 74(3): 534-544.

[53 Chen, et al. 2015] Advances in Phenotyping of Functional Traits, Springer India: 163-180.

[54 Virlet, et al. 2017] "Field Scanalyzer: An automated robotic field phenotyping platform for detailed crop monitoring." Functional Plant Biology 44(1): 143.

[55 Shafiekhani, et al. 2017] "Vinobot and Vinoculer: Two Robotic Platforms for High-Throughput Field Phenotyping." Sensors 17(12): 214.

[56 Bahman, et al. 2019] Height Measurement of Basil Crops for Smart Irrigation Applications in Greenhouses using Commercial Sensors. Graduate Program in Electrical and Computer Engineering The University of Western Ontario. MsC.

[57 Constantino, et al. 2018] Towards an Automated Plant Height Measurement and Tiller Segmentation of Rice Crops using Image Processing, Springer International Publishing: 155-168.

doi:10.6342/NTU202303769

[58 Wang and Chen 2020] "Non-Destructive Measurement of Three-Dimensional Plants Based on Point Cloud." Plants 9(5): 571.

[59 Sa, et al. 2017] "Peduncle Detection of Sweet Pepper for Autonomous Crop Harvesting—Combined Color and 3-D Information." IEEE Robotics and Automation Letters 2(2): 765-772.

[60 Hemming, et al. 2014] "A robot for harvesting sweet-pepper in greenhouses."

[61 Mehta, et al. 2014] "Vision-based control of robotic manipulator for citrus harvesting." Computers and Electronics in Agriculture 102: 146-158.

[62 Han., et al. 2012] "Strawberry Harvesting Robot for Bench-type Cultivation." Journal of Biosystems Engineering 37(1): 65-74.

[63 Xiong, et al. 2020] "An autonomous strawberry-harvesting robot: Design, development, integration, and field evaluation." Journal of Field Robotics 37(2): 202-224.

[64 Luo, et al. 2018] "A vision methodology for harvesting robot to detect cutting points on peduncles of double overlapping grape clusters in a vineyard." Computers in Industry **99**: 130-139.

[65 Andrade-Sanchez, et al. 2014] "Development and evaluation of a field-based high-throughput phenotyping platform." Functional Plant Biology 41(1): 68.

[66 Barker, et al. 2016] "Development of a field-based high-throughput mobile phenotyping platform." Computers and Electronics in Agriculture 122: 74-85.

doi:10.6342/NTU202303769

[67 Busemeyer, et al. 2013] "BreedVision — A Multi-Sensor Platform for Non-Destructive Field-Based Phenotyping in Plant Breeding." Sensors 13(3): 2830-2847.

[68 Hamza, et al. 2005] "Soil compaction in cropping systems." Soil and Tillage Research **82**(2): 121-145.

[69 Wu, et al. 2019] "Plant Phenotyping by Deep-Learning-Based Planner for Multi-Robots." IEEE Robotics and Automation Letters 4(4): 3113-3120.

[70 Sugiura, et al. 2005] "Remote-sensing Technology for Vegetation Monitoring using an Unmanned Helicopter." Biosystems Engineering 90(4): 369-379.

[71 Göktoğan, et al. 2010] "A Rotary-wing Unmanned Air Vehicle for Aquatic Weed Surveillance and Management." Journal of Intelligent and Robotic Systems 57(1-4): 467-484.

[72 De-An, et al. 2011] "Design and control of an apple harvesting robot." Biosystems Engineering 110(2): 112-122.

[73 Kang, et al. 2008] An Efficient Rectification Algorithm for Multi-View Images in Parallel Camera Array.

[74 Vit and Shani 2018] "Comparing RGB-D Sensors for Close Range Outdoor Agricultural Phenotyping." Sensors 18(12): 4413.

[75 Okamoto. and Lee 2009] "Green citrus detection using hyperspectral imaging." Computers and Electronics in Agriculture 66(2): 201-208.

doi:10.6342/NTU202303769

[76 Allouis, et al. 2013] "Stem Volume and Above-Ground Biomass Estimation of Individual Pine Trees From LiDAR Data: Contribution of Full-Waveform Signals." IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing 6(2): 924-934.

[77 Wei, et al. 2012] "Multi-wavelength canopy LiDAR for remote sensing of vegetation: Design and system performance." ISPRS Journal of Photogrammetry and Remote Sensing 69: 1-9.

[78 Kapach, et al. 2012] "Computer vision for fruit harvesting robots–state of the art and challenges ahead." International Journal of Computational Vision and Robotics 3(1-2): 4-34.

[79 Hannan, et al. 2007] "A Real-time Machine Vision Algorithm for Robotic Citrus Harvesting." 2007 ASABE Annual International Meeting, Technical Papers 8.

[80 Baluja, et al. 2012] "Assessment of vineyard water status variability by thermal and multispectral imagery using an unmanned aerial vehicle (UAV]" Irrigation Science 30(6): 511-522.

[81 Zhao, et al. 2005] On-tree fruit recognition using texture properties and color data, IEEE.

[82 Wachs., et al. 2010] "Low and high-level visual feature-based apple detection from multi-modal images." Precision Agriculture 11(6): 717-735.

[83 Besl, et.al 1992] A method for registration of 3-D shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence **14**(2): 239-256.

doi:10.6342/NTU202303769

[84 Hartley et al. 2003] Multiple view geometry in computer vision. Cambridge university press.

[85 Tsai et al. 1989] A new technique for fully autonomous and efficient 3D robotics hand/eye calibration, in IEEE Transactions on Robotics and Automation, vol. 5, no. 3, pp. 345-358, June 1989, doi: 10.1109/70.34770.

[86 Daniilidis 1999] Hand-Eye Calibration Using Dual Quaternions. *The International Journal of Robotics Research*. 1999;18(3):286-298. doi:10.1177/02783649922066213

[87 Stack Overflow 2016] HSV Circle [Website]

[88 Rusu 2010] Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments. Künstl Intell 24, 345–348 (2010). https://doi.org/10.1007/s13218-010-0059-6

[89 Perpendicular Distance] Lesson Explainer: The Perpendicular Distance between Points and Straight Lines in Space | Nagwa

[90] 3d - Averaging quaternions - Mathematics Stack Exchange

[91 Lumistrips] Lumistrips The Impact of Color Temperature: A Comprehensive Guide to LED Lighting

[92 LumiGrow] Full Spectrum LED Grow Lights: The Truth You Need to Know • LumiGrow

[93] Hand-Eye Calibration — moveit_tutorials Noetic documentation (ros-planning.github.io)

[94] Japanese 70-year-old Japanese

doi:10.6342/NTU202303769

[95 William 2017] “Leaves Segmentation in 3D Point Cloud”. In: Advanced Concepts for Intelligent Vision Systems. Ed. by Jacques Blanc-Talon et al. Cham: Springer International Publishing, 2017, pp. 664–674. ISBN: 978-3-319-70353-4.

doi:10.6342/NTU202303769