Zhou *et al. Plant Methods* (2024) 20:151
https://doi.org/10.1186/s13007-024-01277-1

Plant Methods

METHODOLOGY Open Access

An automated phenotyping method for Chinese Cymbidium seedlings based on 3D point cloud

Yang Zhou[1,2*], Honghao Zhou[3] and Yue Chen[4]

Abstract

Aiming at the problems of low efficiency and high cost in determining the phenotypic parameters of Cymbidium seedlings by artificial approaches, this study proposed a fully automated measurement scheme for some phenotypic parameters based on point cloud. The key point or difficulty is to design a segmentation method for individual tillers according to the morphology-specific structure. After determining the branch points, two rounds of segmentation schemes were designed. The non-overlapping part of each tiller and the overlapping parts of each ramet are separated in the first round based on the edge point cloud-based segmentation, while in the second round, the overlapping part was sliced along the horizontal direction according to the weight ratio of the tillers above, to obtain the complete point cloud of all tillers. The core superiority of the algorithm is that the segmentation fits the tiller growth direction well, and the extracted skeleton points of tillers are close to the actual growth direction, significantly improving the prediction accuracy of the subsequent phenotypic parameters. Five phenotypic parameters, plant height, leaf number, leaf length, leaf width and leaf area, were automatically calculated. Through experiments, the accuracy of the five parameters reached 98.6%, 100%, 92.2%, 89.1%, and 82.3%, respectively, which reach the needs of various phenotypic applications.

Keywords 3D point cloud data, Phenotyping, Chinese Cymbidium, Segmentation

Introduction

Chinese Cymbidium is a popular, traditional flower in China, with more than 1,000 years of cultivation history and a broad market prospect. With the increasing consumer acceptance of Cymbidium in various countries, the Cymbidium industry has developed rapidly [1]. After standardized cultivation, the Cymbidiums have bright flowers and are of high ornamental value. Plant phenotyping technology is a system of technology integration that obtains yield, resistance, and quality-related traits from organogenesis [2]. By analyzing the phenotype of the Cymbidiums, it is possible to perform a high-throughput and automated assessment of seedlings and screen out the qualified seedlings. It can also provide morphological information for the subsequent

*Correspondence:
Yang Zhou
zybuaa@163.com
[1]College of Electronic and Information Engineering, Zhejiang university of science and technology, HangZhou 310023, ZheJiang, China
[2]School of Innovation and Entrepreneurship, Zhejiang university of science and technology, HangZhou 310023, ZheJiang, China
[3]Academy for Advanced Interdisciplinary Studies, Nanjing Agricultural University, NanJing 210095, JiangSu, China
[4]Zhejiang Academy of Agricultural Sciences, HangZhou 310022, ZheJiang, China

establishment of a growth model [3]. At present, the measurement of phenotypic parameters of Cymbidium seedlings is still at the stage of manual measurement. However, manual measurement is prone to errors. A few papers simply extract phenotypic parameters such as ground diameter and plant height based on 2D images [4, 5], which have certain limitations in measurement accuracy and adaptability. Therefore, proposing a method to automatically obtain phenotypic parameters based on the 3D morphological model will greatly improve the efficiency of phenotype, and is of great significance for automated monitoring of growth status, variety selection, genomics and other application.

In plant phenotyping applications, optical sensors and sensor-based technology are the mainstream. The 2D or 3D pixel array acquired by the sensor, combined with the phenotype extraction algorithm, are used to obtain the phenotypic parameters of the plant. However, because the image only has 2D information, it cannot get the full specific traits of plant organs (such as leaves), thus decreasing the accuracy of predicted parameters [6, 7]. By multi-angle acquisition, the 3D pixel array containing the 3D coordinates, color information and spatial position information of the plant is generated. The information obtained is richer and has better applicability than 2D images [8]. Therefore, developing a set of 3D morphological model reconstruction and automated phenotyping algorithms can provide an efficient and convenient way for digitizing the structure of Cymbidium. Optical sensors and related equipment have made certain progress in acquiring 3D models of different plants. Laser scanners have better imaging quality on tall plants and large outdoor scenes [8–13]. As a low-cost and efficient sensor, time-of-flight(TOF) cameras are also widely used in plant applications [14–18], they are also used to build a device and achieve good imaging performance of plant seedlings. Structured light cameras also have some successful cases in predicting the growth of 3D crop models and extracting phenotypic parameters [19–21]. However, due to their imaging principle, structured light cameras have poor imaging performance outdoors, and their accuracy is also easily affected by the shooting distance. Therefore, structured light cameras usage has gradually declined, and they faded away in plant phenotyping. In addition to these commonly used sensors, there are also ways to use multi-view stereo vision to reconstruct a 3D model using multiple 2D images [22, 23]. However, the surface of the plant organs constructed by this approach is smooth and lacks effective features on the surface. According to the demand of our application, the research team built a set of non-destructive 3D pixel array acquisition devices suitable for Cymbidium seedlings based on the method of [18] combined with TOF camera, and carried out research on automated phenotyping methods of Cymbidiums.

In plant applications and biomass analysis, the leaf is a core organ that is related to plant health, growth status, and photosynthesis. In a leaf there are some important phenotypic parameters such as leaf length, leaf width, leaf number, and leaf area. One of the important steps in phenotypic estimation is to separate leaves from the entire plant point cloud. Therefore, the algorithms with high applicability are a key point to study. A ring-corrected stem and leaf segmentation algorithm was used for searching the leaf tip based on the ring structure [24], but not all orchid leaves fit the described structure. A skeletonization method combined with Laplace transformation [25] obtained the skeleton diagram and the stem skeleton points [10] was used to calculate the stem diameter, however, the main structure of the orchid is consist with tillers, and they do not have clear boundary between the stem and leaf. The slicing idea was proposed [14] to slice the plant along the vertical axis. Then a Hough plane [26] was extracted based on the skeleton points, thereby achieving stem and leaf segmentation. For the point cloud of leaf part, two phenotypic parameters, including leaf angle and leaf area were extracted. A similar slicing-based stem and leaf segmentation method was also applied to plant point clouds of large plant target from the radar scanner [11]. The leaves of the orchids connected together near the soil, and the two methods described above cannot distinguish the leaves at this area.

The related research provides a good reference for the Cymbidium phenotype. During the growth, the Cymbidium sends out ramets, and the ramets put forth tillers. The complex morphological structure makes it difficult to separate each tiller from the Cymbidium. To overcome this difficulty, we designed an algorithm to determine the branch point of the plant first. After determining the tiller branch points, our study proposed an algorithm with two round segmentations. In the first round, the point cloud of the overlapping and the non-overlapping part of the tillers of each ramet were separated. In the second round, the overlapping part of each tiller was sliced and combined with the point clouds of the corresponding overlapping part to obtain a complete point cloud of each tiller. The overall goals of this research are: (1) Reconstructing the 3D model of Cymbidium seedlings; (2) Developing an automated algorithm to extract the point clouds of all tillers of each ramet; (3) Figuring out the phenotypic parameters, including plant height of each ramet, and leaf length, leaf width, leaf number, leaf area of each tiller.

Materials and methods

Point cloud image acquiring and preprocessing

A non-destructive point cloud acquisition device was built for plant 3D scanning, similar to the one described in our previous research [18]. During the image collection, images of Cymbidium were captured every 180 degrees, while the original RGB image and depth image of Cymbidium from the depth camera are shown in Fig. 1A and Fig. 1B, and the converted point cloud is shown in Fig. 1C. The result of curtain removal is shown in Fig. 1D. The soil and the part below the soil were removed through pass-through filtering, as shown in Fig. 1E. Because of the imaging principle of TOF itself, points generated by flying noise (FPN) were generated

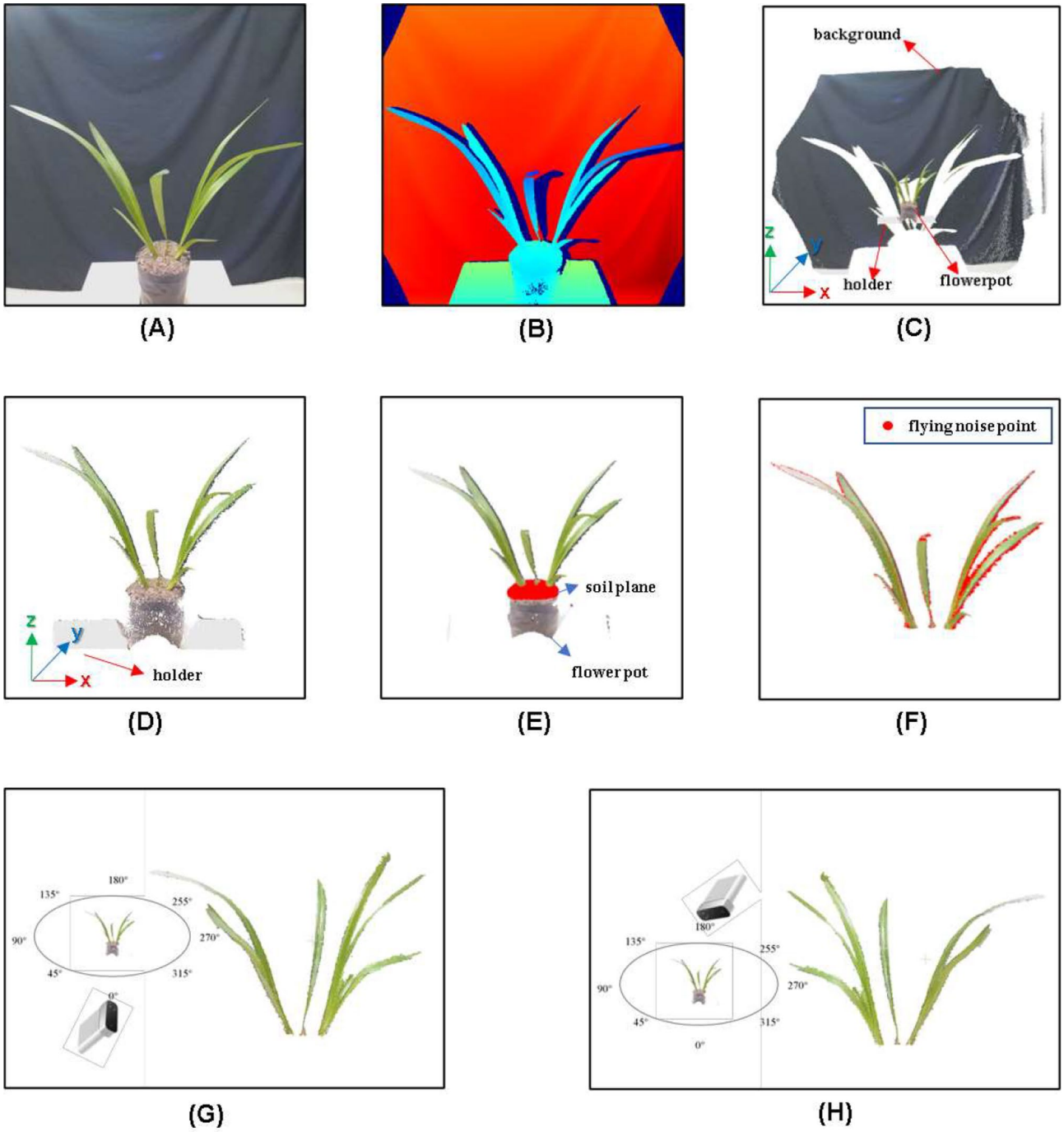

Fig. 1 Point cloud preprocessing. (**A**) RGB image of Cymbidium. (**B**) Depth image of Cymbidium. (**C**) point cloud of *Cymbidium*. (**D**) Point cloud after background removed. (**E**) The soil plane marked in red for removal. (**F**) The voxels of flying noise marked in red. (**G**) 0-degree registered point cloud. (**H**) 180-degree registered point cloud

and affected the subsequent calculation, and an algorithm based on Principal Component Analysis(PCA) [27] was used to remove FPN. For each point in the point cloud after pre-processing, KD-tree accelerated radius search method was used to search for neighborhood points within the setting range, and local surface fitting was performed on all the searched voxels. The vector perpendicular to the local surface was used as the local normal vector. If the angle between the local normal vector and the vector between the voxel and the coordinate origin was less than the specified threshold, then the voxel was regarded as FPN and removed. The FPN defined by the above method is shown in Fig. 1F. Then, a radius-based outlier filter [28] was used to remove discrete noise points. Finally, the Cymbidium point clouds at 0 and 180 degrees were registered. The rotational registration [8] performed a rough registration firstly. Then the Iterative Closest Point (ICP) algorithm [29] was used to minimize the distance between corresponding voxels, thereby finishing fine registration. The front and back views of the point cloud after registration are shown in Fig. 1H and Fig. 1I.

Cymbidium tiller segmentation

The traditional slicing algorithm was not fit for the Cymbidium. Discovering morphological features of Cymbidium, Cymbidium seedlings were often found to have several ramets, and each ramet had multiple tillers. As Fig. 2A illustrated, a single plant included three ramets, and different ramets were boxed with different colors, and these three ramets had 3, 1, and 4 tillers, respectively. Therefore, the key point of our method was to segment the Cymbidium tillers from each ramet. For leaf extraction, the slicing method was commonly applied on each slice layer along the vertical direction, the centroid of each cluster was extracted as a skeleton point, and then skeleton points were connected by minimum spanning tree searching. The sum of the distances between all the skeleton points was taken as leaf length of the leaf [30]. Figure 2B shows horizontal slice layers along the vertical direction, and Fig. 2C shows the connectivity of skeleton points. It could be clearly seen that the traditional slicing method had some errors. The ① and ② in Fig. 2C show an incorrect connection of two different tiller skeleton points. Because the connectivity of the minimum spanning tree was based on the nearest point, two adjacent skeleton points might belong to the two tillers, resulting in an incorrect connection, while there might exist a wide slice in the horizontal direction. For example, at ③ in Fig. 2C, the tillers grow in the horizontal direction and the slice direction is fixed, the extracted slice layer is too wide in the horizontal direction and cannot be used in subsequent calculations. Finally, in part close to the soil, such as ④ in Fig. 2C, because the tillers were concentrated near the soil, traditional horizontal slicing only got one point as the skeleton points of all the tillers, which made it impossible to determine the number of tillers.

For one Cymbidium ramet (the following definition is only used for algorithm implementation, not for Cymbidium anatomy), the tiller closest to the vertical direction (z-axis) is defined as the main tiller, and the other tillers are defined as lateral tillers (the main tiller and lateral tillers is shown in Fig. 3). Based on the morphological structure of the Cymbidium, lateral tiller grows out from the main tiller, also called non-overlapping parts of tillers, and lateral tillers no longer produce new tillers. Some ramets only contained main tillers (no lateral tillers). In this case, there was no point cloud of overlapping tillers of this ramet. Based on the technical difficulties, our research team proposed a set of algorithms for tiller segmentation. When branch points between overlapping and non-overlapping parts were determined, a total of two rounds of segmentation were performed. In the first round, the preprocessed point cloud and branch position set were the input, and the non-overlapping point cloud of each tiller and the point cloud of the overlapping part of all tillers of each ramet were the output. In the second round, the point clouds of the overlapping parts were separated according to the corresponding non-overlapping tillers. The separated parts and the non-overlapping tillers obtained in the first round were merged one by one according to the corresponding relationship, thereby obtaining a complete point cloud of different tillers of each ramet. Each tiller point cloud in the segmentation process was composed of different clusters, and the centroid of each cluster was extracted as the skeleton point of the tiller, which could be used for subsequent phenotypic parameter calculation.

Before segmentation, it was necessary to obtain the amount of ramets. Because the bulb part of each ramet close to the soil would not produce a branch, we horizontally sliced the Cymbidium point cloud under 1 cm thickness at the position of 2–3 cm above the soil, and performed Euclidean clustering on the slice layer to obtain multiple clusters. The number of clusters was used to represent the amount of ramets of the whole Cymbidium. These sliced clusters were defined as initial clusters and stored an initial cluster set (initial cluster set). Figure 3 shows the initial clusters of three ramets. From left to right, the three initial clusters were distinguished by red, green and blue colors. In order to perform two rounds of segmentation, a branch position set was created to store the 3D coordinates of the tiller branch points, and an overlapping set of point clouds of the tillers was used to store point clouds of overlapping parts of tillers of different ramets (overlapping set). Each element represented the point cloud of the overlapping part of one ramet. Another set, called non-overlapping parts of

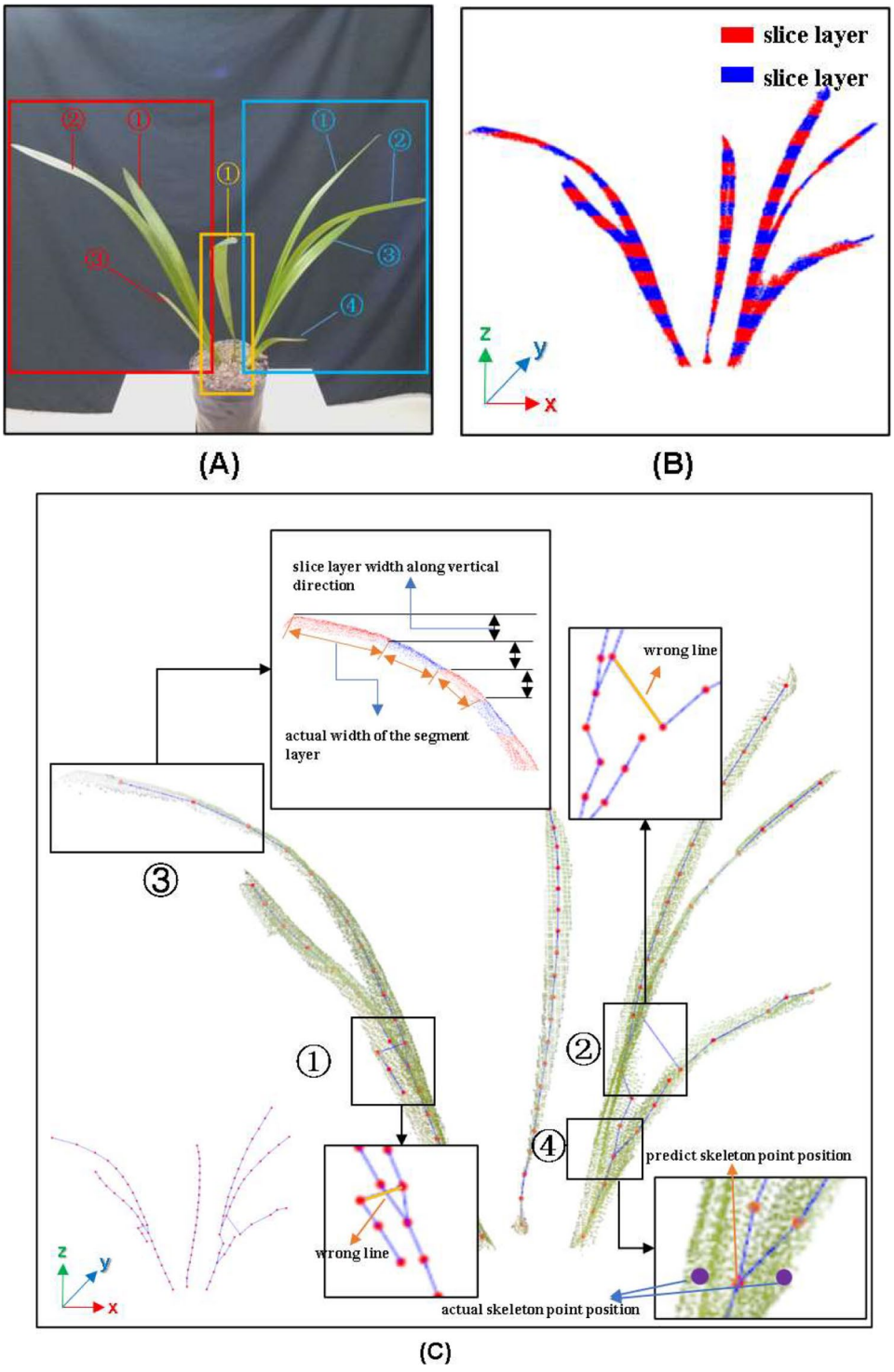

Fig. 2 Slicing skeleton method. (**A**) Distinction between ramets and tillers. Different colors represent different ramets, and the tillers of each ramet are numbered with different numbers. (**B**) Vertical slicing. Each slice layer is distinguished by red or blue colors. (**C**) Skeleton connection through using traditional slicing method. In ① and ②, the wrong connection is marked in orange. In ③, the black slice layer is a slice layer along the vertical direction, and the orange slice layer is the actual slice layer. In ④, the red points represent the skeleton point predicted by the slicing skeleton method, and the purple points represent the actual skeleton point

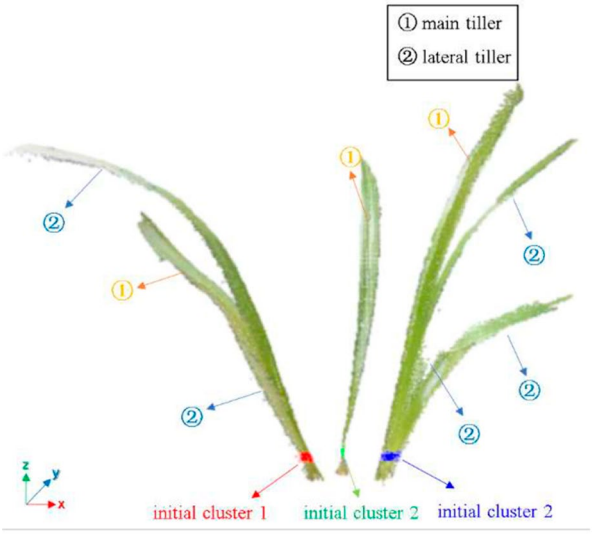

Fig. 3 Classification of main tiller and lateral tillers of Cymbidium. ① the main tiller, ② the lateral tiller. The initial clustering clusters of the three ramets are marked with red, green and blue colors, respectively

tillers (non-overlapping set), was with tillers as units, to store point clouds of non-overlapping parts of different tillers of each ramet.

Determination of branch points (Algorithm 1)

First, the initial cluster of each ramet was taken as the reference cluster, its horizontal width as the slice thickness. The centroid of the reference cluster was the starting point, then the preprocessed point cloud was sliced upward in the vertical direction to obtain the upper slice layer from the reference cluster. A Euclidean clustering method was performed on the upper slice and the layer where the reference cluster was located, the clusters whose centroid distance from the current reference cluster was greater than the slice thickness were removed from the upper slice layer to obtain the actual clusters of upper slice layer. If the number of actual clusters in the upper slice layer was less than or equal to that of clusters in the reference layer, the cluster whose centroid was closest to the centroid of the reference cluster among all actual clusters in the slice layer was directly regarded as the following reference cluster in the next searching loop. Otherwise, the amount difference between clusters was recorded as N. Then the N centroid points of actual clusters closest to the reference cluster (excluding the reference cluster in the next loop) were added to the branch position set as branch points of non-overlapping tiller. Based on the reference cluster, the upward slicing along the vertical direction was continued, and the reference cluster and branch position set were renewed according to the above rules until the actual cluster number searched was 0. At this time, the branch position set contained coordinates of all the branch points. Finally, by traversing all the initial clusters, the lateral branch points of all ramets of the Cymbidium were obtained.

Table 1 Nomenclature in Algorithm 1

Naming	Explanation
rc	reference cluster
Max_z	maximum value in vertical direction
$widt?$	the horizontal width of the reference cluster
$Centroid\,()$	centroid of the point cloud
bps	branch position set
non-overlapping set	point cloud set of non-overlapping part of tillers

The associated nomenclature in Algorithm 1 is shown in Table 1 and the algorithm 1 is shown in Fig. 4. The preprocessed point cloud and the initial cluster set were input in Algorithm 1. One of the initial clusters (line 1) from the initial cluster *set* was the input of the following steps and performing the iterations as the main loop of Algorithm 1 until all initial clusters had been traversed (lines 1–45). In the main loop, the point cloud $cloud_l$ was used to store the point cloud of the tiller overlap part of the ramet (line 2), and the initial cluster was used as the reference cluster (line 3). The variable named *num of clusters searched* was set to store the clusters searched (initial value was 1) (line 4) and start the following iteration according to the sub-loop process of algorithm 1(lines 5–44). The sub-loop process was mainly to determine all the branch points of a ramet of Cymbidium. The main loop process continuously traversed the sub-loop process to determine the branch points of all ramets. The reference cluster (rc) was added to the ramet point cloud $cloud_l$ (line 6) (the initial value of rc was the initial cluster). The centroid of each reference cluster was taken as the starting point, and the horizontal width of

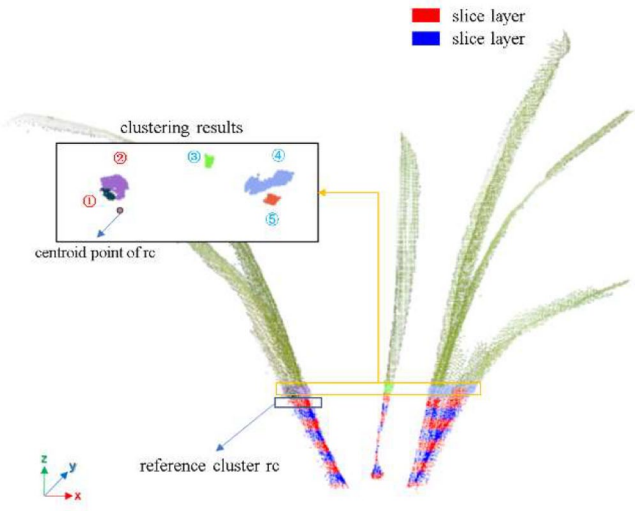

Fig. 4 Algorithm for branch point determination

the reference cluster was taken as the slice thickness. For the point cloud $cloud_p$, upward slicing was performed along the vertical direction from the starting point, and the sliced point cloud was added to the point cloud $cloud_{tmp}$(lines 7–13). The Euclidean clustering on the point cloud $cloud_{tmp}$ (slice layer) was carried out. Each slice layer could obtain one or more clusters, and these clusters were stored in the cluster set (line 14) and a variable *num of clusters* above (the initial value is 0) (line 15) was set accordingly. The horizontal width of the reference cluster was regarded as the distance threshold(*threshold*), the distance between the centroid of the reference cluster and the centroids of all clusters was figured out. If the distance was within the threshold, these clusters were regarded as the above reference cluster and the number of clusters above would be recorded (lines 16–23). Figure 5 shows the image of recording the number of clusters above.

The number of clusters above had three situations as follows:

Case 1: (lines 24–32)
If the number of clusters above was greater than that of clusters searched (line 24), it meant that a branch point of a non-overlapping tiller was located where the non-overlapping part of tillers grew out. The number of branch points (*num of branch points*) was the difference value between the number of clusters above (*num of clusters above*) and the number of clusters (*num of clusters searched*) (line 25). The reference cluster was regarded as the starting point, and all upper clusters from near to far were sorted based on the distance between cluster centroids (line 26). The one cluster closest to the reference cluster was the position where the main tiller was located, and this cluster was used as the reference cluster for the next searching loop (line 27). For the others, the centroid positions of the first number of branch point clusters were recorded in the branch position set (lines 28–30). The number of clusters above was replaced by number of clusters searched (line 31). Figure 6B and Fig. 6C show two situations in which number of clusters above is greater than the number of clusters searched. In Fig. 6B, a branch point exists. After sorting the upper clusters according to the centroid distance, the nearest upper cluster where ① is located is used as the reference cluster, and the centroid of the ② is added to the branch position set. In Fig. 6C, there exist two branch points. The nearest upper cluster where ① is located is used as a reference cluster, and the centroids of upper clusters ② and ③ are added to the branch position set.

Case 2: (lines 33–36)
If the number of clusters above is less than or equal to that of clusters searched (line 33), there was no branch point for non-overlapping tillers here. The reference cluster in the next following loop was one of the upper clusters that had the smallest distance to the reference cluster in current loop where the main tiller was located for continued upward searching (lines 34–35). Figure 6D shows the situation when the number of clusters above is less than or equal to the number of clusters. There was no new branch point. The cluster closest to the top is selected as the reference cluster in the next loop.

Case 3: (lines 37–43)
If the number of clusters above was zero, the highest point of the main tiller had been searched (the point cloud $cloud_{tmp}$ was empty), indicating that the search for this ramet was completed. This time, for this ramet, if the branch position set was empty, there was no starting

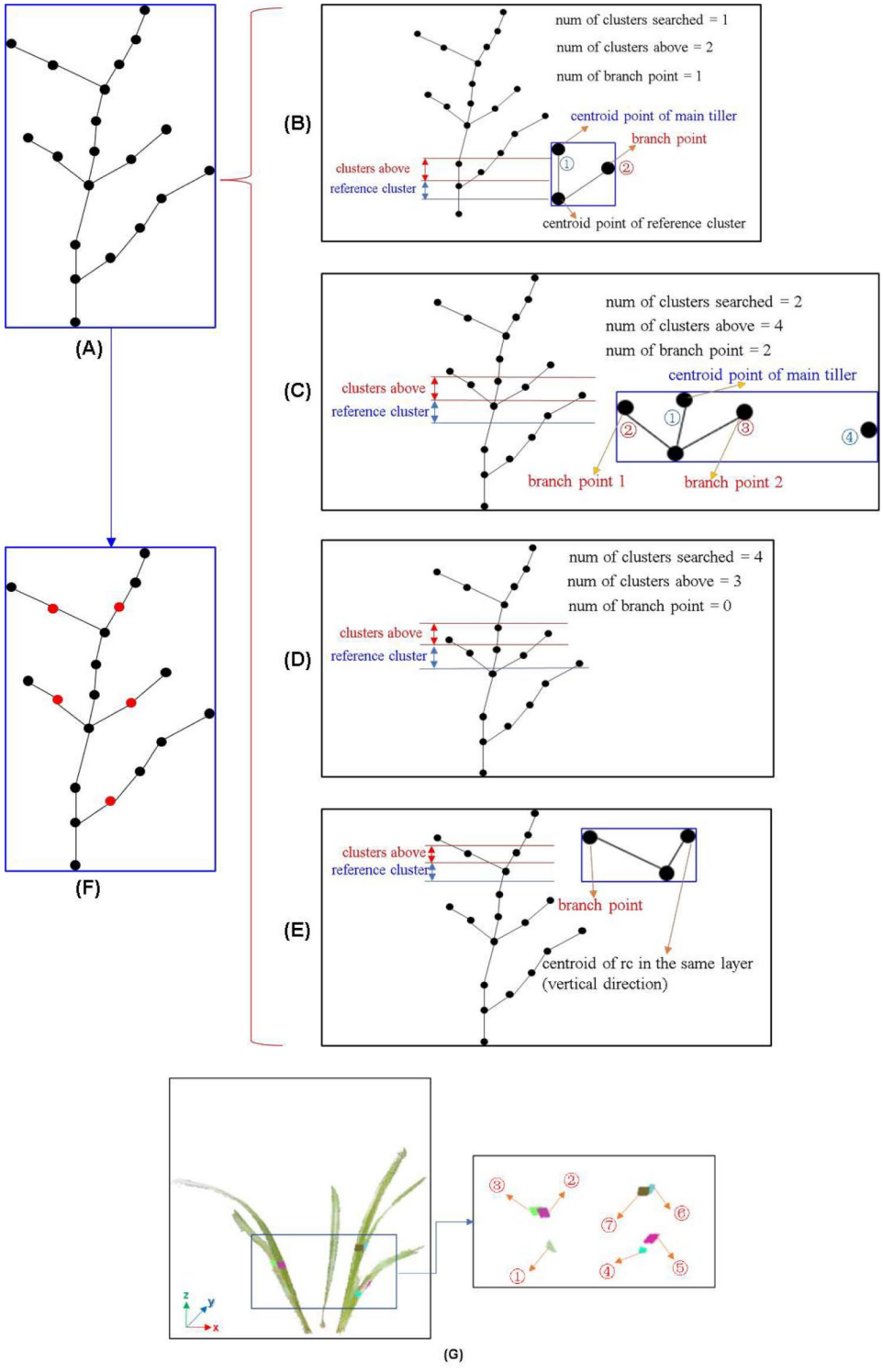

Fig. 5 The above searched clusters. Euclidean cluster on the slice layer point cloud above the reference cluster and five clusters are obtained, the ① and ② are the clusters searched above the reference cluster. ③, ④, and ⑤ are the cluster where other tillers are located

point of the non-overlapping tillers during the upward search process, and the ramet had only one main tiller and no lateral tillers. If the branch position set was not empty (line 38), there was at least a branch point for the non-overlapping part of the lateral tillers during the upward search of the ramet along the main tiller, and the branch points of the non-overlapping tillers for the main tillers not recorded. Here, the cluster corresponding to

Algorithm 1 Branch point determination.

Input: *The point cloud after preprocessing and initial cluster set.*
Output: *overlapping set and branch position set.*

```
for each initial cluster in initial cluster set do
    create cloud_l to store the overlapping portion of the ramet where initial cluster located
    rc = initial cluster
    num of clusters searched = 1
    while true do
        add rc to cloud_l
        #Get the vertical slice layer above the reference cluster
        Generate a temporary cloud cloud_tmp to store points in slice layer of Z-axis
        for each point in cloud_p do
            if point.z >= Max_z(rc) and point.z <= Max_z(rc) + width then
                add point to cloud_tmp
            end if
        end for
        cluster set = euclideanClustering(cloud_tmp)
        num of clusters above = 0
        for each cluster in cluster set do
            if dis(Centroid(cluster), Centroid(rc)) < threshold then
                num of clusters above++
            end if
            else
                 remove this cluster from the cluster set
            end else
        end for
        if num of clusters above > num of clusters searched then
            num of branch point = num of clusters above - num of clusters searched
            sort all clusters in the cluster set from near to far according to centroid distance
            rc = cluster set[0]
            for i in (1, 1 + num of branch point) do
                add Centroid(cluster cluster[i]) to bps
            end for
             num of clusters searched = num of clusters above
        end if
        if num of clusters above <= num of clusters searched then
            sort all clusters in the cluster set according to centroid distance
            rc = cluster set[0]
        end if
        if num of clusters above == 0 then
            if ls.size() ! = 0 then
                add the Centroid(rc) at the same layer as the last position in bps to bps
            end if
            add cloud_l to overlapping set
            break
        end if
    end while
end for
```

Fig. 6 Branch point determination. (**A**) Centroid point connection of ramets. (**B**) Num of clusters above is greater than the num of cluster searched, one branch point exists. (**C**) Num of clusters above is greater than the num of cluster searched, two branch point exist. (**D**) Num of clusters is less than the num of cluster searched, and there is no branch point. (**E**) Num of clusters above is zero. Return to the position of the last branch point. (**F**) Branch points of a ramet. The red points represent branch points, and the coordinates of branch points are recorded in the branch position set. (**G**) The cluster where all the branch points are located. There are seven branch points in the plant

the last location was recorded in the branch position set (the position coordinate corresponding to the maximum value of the branch position set index), and the centroid of the reference cluster (rc) located in the same layer (vertical direction) was added to the branch position set as the starting point of the non-overlapping part of the main tiller (line 39). Figure 6E shows this situation in which the branch point of the non-overlapping part of the main tiller was found. In this case, recording all non-overlapping tiller branch point positions was finished (recorded in the branch position *set*). The point cloud $cloud_l$ was stored in overlapping *set* (line 41), and the sub-loop process of the ramet ended (line 42). Figure 6G shows the clusters corresponding to all branch points of a ramet.

First round segmentation (algorithm 2)

The main idea of this algorithm was to segment the non-overlapping part point cloud of the lateral tillers or the main tillers by the input of the branch position set. Each branch point of the non-overlapping tillers was used as the starting position of the lateral tillers, and the input of the first round was the point cloud by subtracting the overlapping part point cloud from the preprocessed point cloud of each ramet.

For the point cloud to be segmented, the upper and lower segment lines consistent with the growth direction of lateral tillers were designed to locate the cluster of lateral tillers between the upper and lower segment lines. By updating the upper and lower segment lines along the growth direction, the cluster of lateral tillers between the upper and lower segment lines was continuously searched, and the non-overlapping part of lateral tillers starting from the branch point was searched according to its growth direction. Each branch point of non-overlapping tiller (the starting position of lateral tillers) was traversed, and the segmentation of non-overlapping point clouds of all lateral tillers was achieved ramet by ramet.

The aim of the first round was to segment the non-overlapping part of the point cloud of the lateral tiller and the main tiller according to the branch position set. The detailed pseudo-code of this process is shown in Fig. 7 and the associated nomenclature in Algorithm 2 is shown in Table 2. In Algorithm 2, the overlapping set, the branch position set, and the preprocessed point cloud were the input, and one set of the location coordinates (line 1) from the branch position set was the starting point of iteration of main loop until all the location coordinates were traversed (lines 1–46). According to the cluster of each position in the branch position *set* and the point cloud $cloud_p$ for further segmentation, at the XOZ view, the slope of the lower segment line was 0 ($k_{lower} = 0$), and its intercept was the minimum coordinate value in the

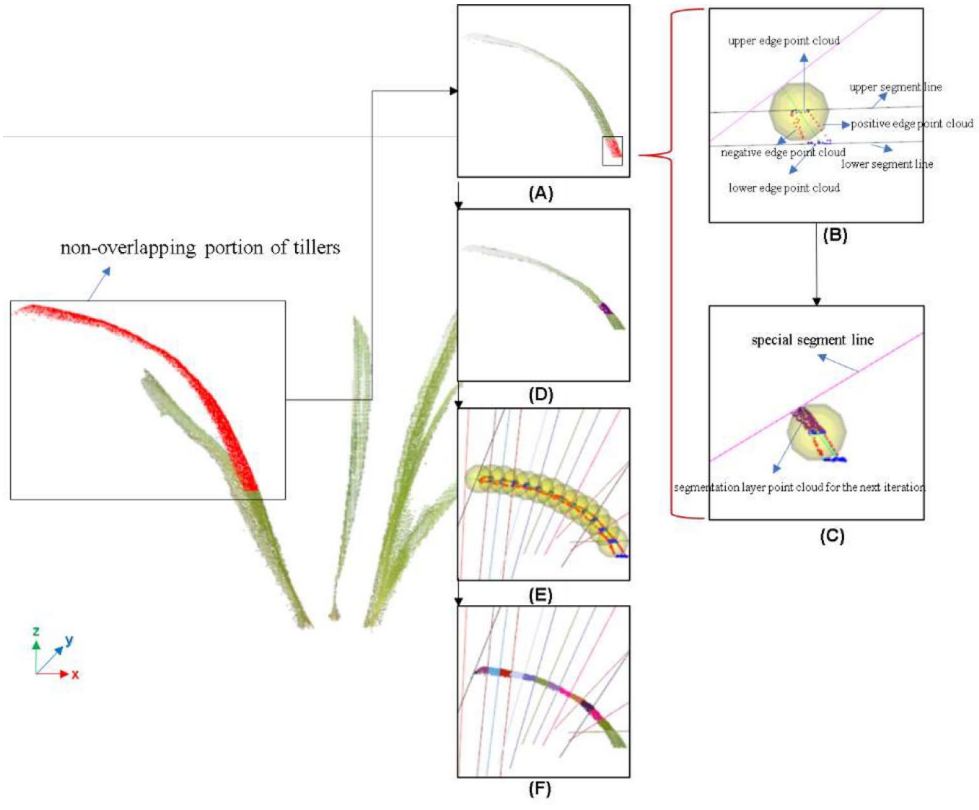

Fig. 7 Algorithm for first round segmentation

Table 2 Nomenclature in Algorithm2

Naming	Explanation
$k_{lower}\, k_{upper}$ $b_{lower}\, b_{upper}$	The slope and intercept of the upper and lower segment line.
$minPt.z\, maxPt.z$	The maximum and minimum values in the vertical direction of the cluster.
$cloud_p$	Preprocessed point cloud.
$cloud_s$	The point cloud to store the non-overlapping part of the tiller.
$line(L_k,\ L_b)$	The line formed by slope L_k and the intercept L_b. Only the XOZ plane is considered.
$dis(point,\ line())$	The distance from the point to the line. Only the XOZ plane is considered.
$cloud_{lower}\, cloud_{upper}$	Upper and lower boundary point cloud.
$point_{lower}\, point_{upper}$	The center point of upper and lower boundary point cloud.
$Centroid\,()$	Center of the point cloud.
$k_{center}\, b_{center}$	The slope and intercept of the line connecting the center points of the upper and lower boundary.
$cloud_{pos}\, cloud_{neg}$	The point cloud where the positive and negative boundary are located.
$k_{pos}\, k_{neg}$	The slope of the line fitted by the point cloud where the positive and negative sides of the boundary are located.
$k_{new}\, b_{new}$	The slope and intercept of the special segment line.

vertical direction of the cluster ($k_{lower} = minPt.z$). The slope of the upper segment line was 0 ($k_{upper} = 0$), and its intercept was the maximum coordinate value in the vertical direction of the cluster ($k_{upper} = maxPt.z$) (line 2). The cluster between the upper and lower segment line (line 3) was obtained, and the method described in step (2.2.2.1 was used to calculate the boundary point cloud of the cluster (line 4) and a point cloud $cloud_{edge}$ to store boundary points was set up (line 5).

For the boundary point cloud $cloud_{edge}$, the sub-loop of algorithm 2 were as follows (lines 6–43): The horizontal and vertical coordinates of each point in the point cloud $cloud_{edge}$, were input to the general equation of the upper and lower segment lines respectively, if the output value was less than the specified threshold, this point was added to the corresponding upper boundary point cloud $cloud_{upper}$ or lower boundary point cloud $cloud_{lower}$, the this point was removed from the boundary point cloud $cloud_{edge}$ (lines 7–16). As shown in Fig. 8B, the black lines are the upper segment line and the lower segment line, and the upper and lower boundary point clouds are marked in blue. The centroid $point_{upper}$ and $point_{lower}$ (lines 17–18) of the upper and lower boundary point clouds were calculated to find a connecting line between the two centroid points (line 19) as shown in Fig. 8B (Green line). The positive and negative directions were defined by the sides of the connecting line. From the XOZ plane, each remaining point in the boundary point cloud $cloud_{edge}$ was classed into two classes, the positive or negative direction of the connecting line by coordinate connect line equation. The points (excluding the points in the upper and lower boundaries (line 10, line 14)) were added to the positive boundary point cloud $cloud_{pos}$ and the negative boundary point cloud $cloud_{neg}$ (lines 20–27) accordingly. The positive and negative boundary point clouds were subjected to Ransac straight line fitting, and the slopes of the two fitting lines k_{pos} and k_{neg} (line 28) were obtained. As shown in Fig. 8B, the red points on the left side of the connecting line (green line) are defined as the point of the positive side boundary, and the red points on the right side of the connecting line (green line) are defined as the point of the negative side boundary.

In order to complete the segmentation task for the tiller where the input cluster was located, a specific segment line was designed to search the object point cloud along the growth direction, and the searched point cloud was used as the point cloud of the non-overlapping part of the tiller which was added to the temporary point cloud $cloud_{tmp}$ (line 30). The slope k_{new} of the specific segment line was obtained according to Eq. (1), which was the reciprocal of the absolute value of the larger slope of the two fitting lines, which meant, the direction of k_{new} was perpendicular to one of the fitting lines with a larger slope value. The distance between the centroids of the upper and lower boundary point clouds $point_{lower}$ and $point_{upper}$ was the radius, a circle whose center was the centroid of upper boundary point cloud was determined. The slope of the specific segment line was k_{new}, and the intercept new_b of the special segment line was figured out by using the distance Eq. (2) from the point to the line. Where the coordinate of the center point of the upper edge was (x0, z0), the radius r represented the distance between the centroid of the upper and lower edges $point_{lower}$ and $point_{upper}$ (line 28). Figure 8C shows the specific segment line (marked in purple), while the distance between the upper and lower segment lines was the radius. The purpose of designing the circle here was to keep the same searching step length in all directions (the set length was the radius of the circle). When the slope k_{new} and the intercept b_{new} were got, the general equation of the specific segment line could be obtained.

$$k_{new} = -1/max(fabs|k_{pos},\ k_{neg}|) \tag{1}$$

$$\frac{\left|-\frac{1}{\max(k_{pos},\, k_{neg})}x0 - z0 + b_{new}\right|}{\sqrt{(k_{new})^2 + 1}} = r \tag{2}$$

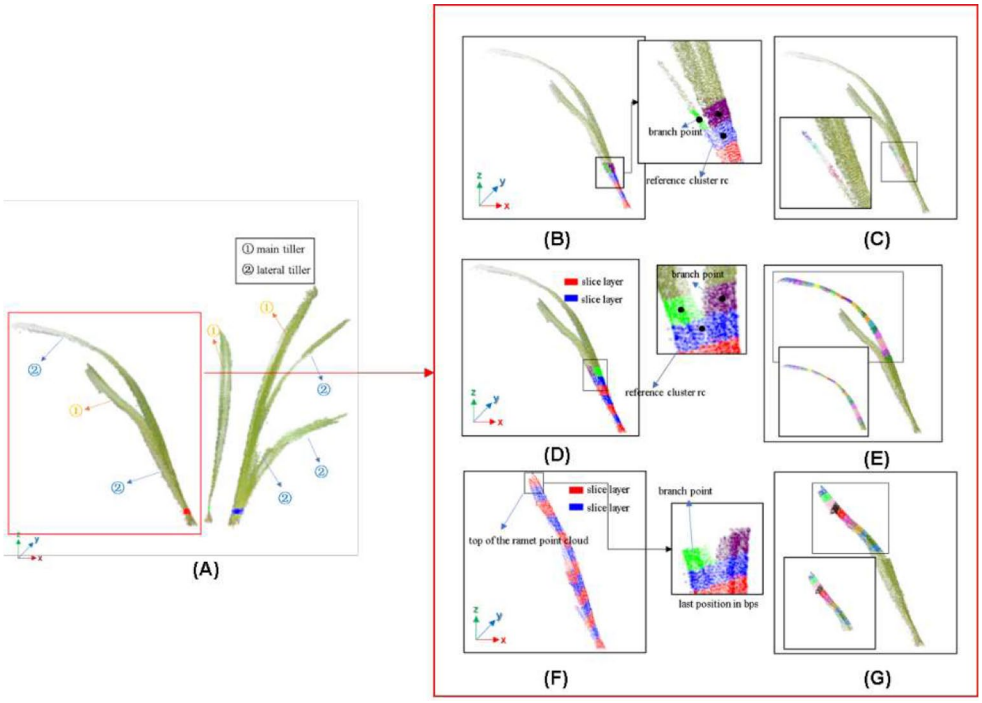

Fig. 8 non-overlapping part segmentation. (**A**) The initial cluster was marked in red. (**B**) The blue point cloud: the upper and lower edges. The red part: the positive and negative edges. The black lines: the upper and lower segment line. The green line: the connecting line. (**C**) Special segment line is marked in purple. (**D**) The cluster of the next round to be segmented is marked in purple. (**E**) Segment lines in each round. (**F**) Segmentation of non-overlapping part. Each searched cluster is distinguished by different colors

In the current round of sub-loop of Algorithm 2, the specific segment line was used as the upper segment line, and the upper segment line obtained by the previous round was used as the lower segment line. The point cloud $cloud_p$ between the upper and lower segment lines was added to the point cloud $cloud_{tmp}$ (lines 30–35). Then, if the $cloud_{tmp}$ was empty, there was no point between the upper and lower segment lines, indicating that the highest point of the tiller had been reached, then Algorithm 2 was ended (lines 36–38), and the cluster between two segment lines was added to the non-overlapping part of the tiller point cloud. Otherwise, if it was not empty, it indicated that the highest point of the non-overlapping part of the tiller had not been reached, and the sub-loop of Algorithm 2 needed to be continued.

In the detail of this procedure, the coordinates of each point in the point cloud $cloud_p$ (only XOZ plane was considered) were the input of the general equation of the upper and lower boundaries respectively, and the product of the two-equation output was calculated. If the product was less than or equal to 0, it indicated that the point was located between the upper and lower boundaries, and the boundary point cloud $cloud_{edge}$ (line 39) was emptied. Using the method described in step 2.2.3, the boundary points of the point cloud $cloud_{tmp}$ was calculated again and stored in the boundary point cloud $cloud_{edge}$, and the non-overlapping part of the point cloud $cloud_s$ was stored in the point cloud $cloud_{tmp}$ (lines 40–41).

The upper and lower segment lines in the sub-loop iteration were updated for the next round of Algorithm 2, as follows: The lower segment line of the next round was updated to the upper segment line of this round, and the upper segment line of the next round was updated to the specific segment line (line 42), for the next round of the sub-loop of Algorithm 2. Figure 8D shows the cluster to be segmented in the next round. Figure 8E shows all the clusters. Figure 8F shows each segmented point cloud (distinguished by different colors). The segmented non-overlapping part point cloud $cloud_s$ was added to the non-overlapping set (line 45). In the branch position set, the last element was the starting position of non-overlapping part of the main tiller. The point cloud of non-overlapping part of the main tiller of the ramet was added to the overlapping part point cloud $cloud_l$ in step 2.2.1. Therefore, the non-overlapping part point cloud of the main and lateral tiller was separated from the overlapping part point cloud $cloud_l$ and added to the non-overlapping set, and proposed segmentation method ensured that the segmentation thickness between each upper and lower lines along the growth direction of the tiller was consistent. Figure 9 shows the segmentation of non-overlapping parts of all tillers.

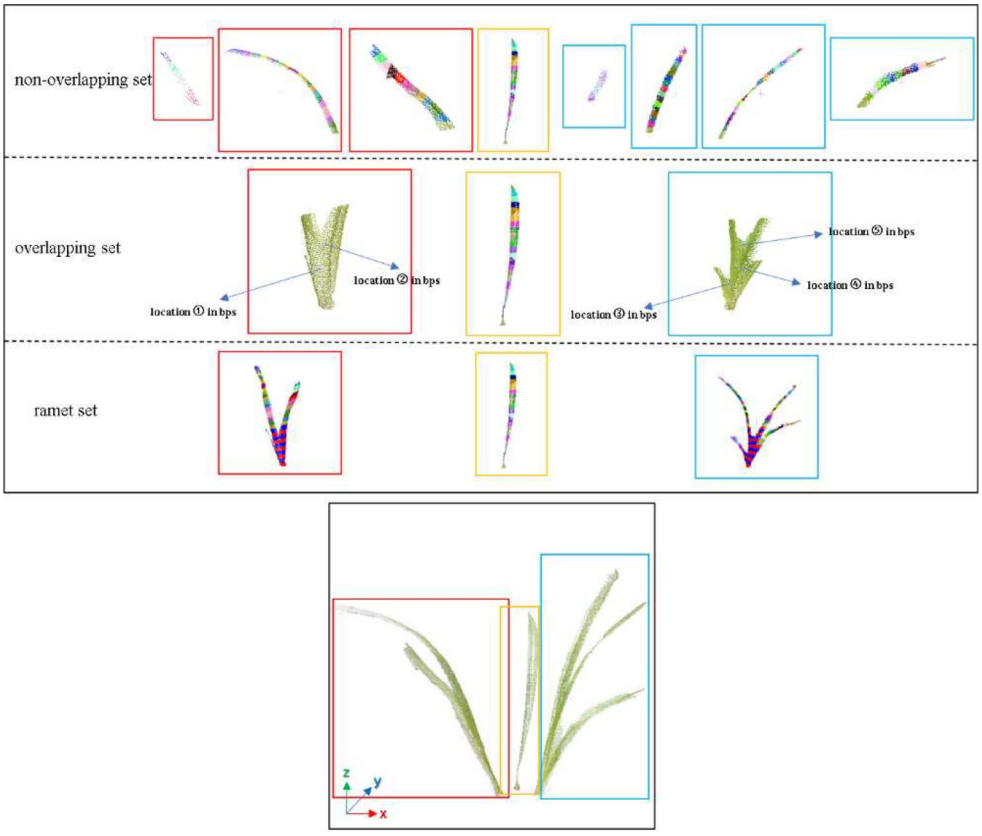

Fig. 9 First round segmentation. (**A**) The ramet where the red box is located is the discussed ramet. (**B**) The cluster marked in blue is the reference cluster rc. The cluster marked in green is the initial cluster of non-overlapping parts to be segmented. (**C**) The non-overlapping part of the first tiller, each segmented cluster is represented by different colors. (**D**) The cluster marked in blue is the reference cluster rc. The cluster marked in purple is the initial cluster of non-overlapping parts to be segmented. (**E**) The non-overlapping part of the second tiller, each segmented cluster is represented by different colors. (**F**) The top red cluster is the cluster where the highest point in Y-axis of the ramet is located. The cluster marked in green is the initial cluster of non-overlapping parts of the last tiller to be segmented. (**G**) The non-overlapping part of the last tiller, each segmented cluster is represented by different colors

The non-overlapping set of all the tillers in one ramet was separated in the overlapping set by iteration of the main loop of Algorithm 2 (lines 1–46). After the first round of segmentation, Fig. 10 shows all the elements in each set, the point cloud of the seven tillers in non-overlapping set, the point cloud of overlapping part of the three tillers in overlapping set (where the second ramet has only one tiller, so the non-overlapping part of the point cloud and the overlapping part of the point cloud are the same), and the branch position set (bps) where a total of 7 branch point locations are recorded).

Edge point cloud of cluster extraction

The conventional point cloud edge extraction was polar coordinate transformation, but the threshold of the angle was undetermined leading to the generation of internal holes and the failure of edge point extraction [31]. From the morphological feature of Orchid, the extraction method based on triangular mesh was an effective method. Through triangular mesh [32], the relationships between points in point could be established, and the edge points cloud of the cluster were marked by searching and discrimination. However, the triangular mesh construction was easily affected by noise, so smoothing [33] was used to reduce the discrete noise. Subsequently, the greedy projection triangulation [34] was carried out to obtain the triangular mesh and the reconstructed surface model, which contained the connection relationships of three points in each triangular mesh. In one triangular mesh, there were three edges, and the edges were divided into two categories, outer or inner edges of the point cloud. When traversing all triangular meshes, if it was an inner edge, an edge formed by two points would become a common edge of two triangular meshes, and the inner edge would be recorded twice for two meshes. For the outer edge, it only belonged to one mesh, and was recorded only once. Based on this principle, when traversing all the edges of all triangular meshes, as long as the edges that were stored only once were regarded as the outer edges, and these outer edges were connected together as the edge point cloud of cluster. Figure 11A show the outer boundary of a local cluster. It could be

Algorithm 2 First round segmentation.

Input: *overlapping set, branch position set and point cloud* cloud_p
Output: *Nonoverlapping set and overlapping set*

for each *position* **in** branch position set **do**
  $k_{lower} = 0$, $k_{upper} = 0$, $b_{lower} = \text{minPt.z}$, $b_{upper} = \text{maxPt.z}$
  obtain the cluster corresponding to the position
  create a temporary cloud cloud_{edge} to store cluster edge points
  calculate cluster edge points and store them into cloud_{edge}
  while true **do**
    for each *point* **in** cloud_{edge} **do**
      if dis(point, line(k_{lower}, b_{lower})) < threshold **then**
        add this point to cloud_{lower}
        remove this point from cloud_{edge}
      end if
      if dis(point, line(k_{upper}, b_{upper})) < threshold **then**
        add this point to cloud_{upper}
        remove this point from cloud_{edge}
      end if
    end for
    $\text{point}_{lower} = \textbf{Centroid}(\text{cloud}_{lower})$
    $\text{point}_{upper} = \textbf{Centroid}(\text{cloud}_{upper})$
    calculate the k_{center} and b_{center} of the straight line connected by point_{lower} and point_{upper}
    for each *point* **in** cloud_{edge} **do**
      if (k_{center} * point.x - point.z + b_{center}) < 0 **then**
        add this point to cloud_{neg}
      end if
      if (k_{center} * point.x - point.z + b_{center}) > 0 **then**
        add this point to cloud_{pos}
      end if
    end for
    fit straight lines to cloud_{pos} and cloud_{neg} and calculate k_{pos} and k_{neg}
    solve for k_{new} and b_{new} according to formula (1) and (2)
    create point cloud cloud_{tmp}
    for each *point* **in** cloud_p **do**
      if (k_{new} * point.x - point.z + b_{new}) * (k_{upper} * point.x - point.z + b_{upper}) < 0 **then**
        add this point to cloud_{tmp}
      end if
    end for
    if cloud_{tmp} is empty **then**
      break
    end if
    clear all points in cloud_{edge}
    calculate edge points of cloud_{tmp} and store them into cloud_{edge}.
    create point cloud cloud_s
    add cloud_{tmp} to cloud_s
    $k_{lower} = k_{upper}$, $b_{lower} = b_{upper}$, $k_{upper} = k_{new}$, $b_{upper} = b_{new}$
  end while
  add cloud_s to non-overlapping set
end for

Fig. 10 Display of all elements in of non-overlapping set, overlapping set and branch position set. Three colors, including red, orange and blue, were used to represent three ramets respectively. ① to ⑤ represent the five positions of the branch position set respectively

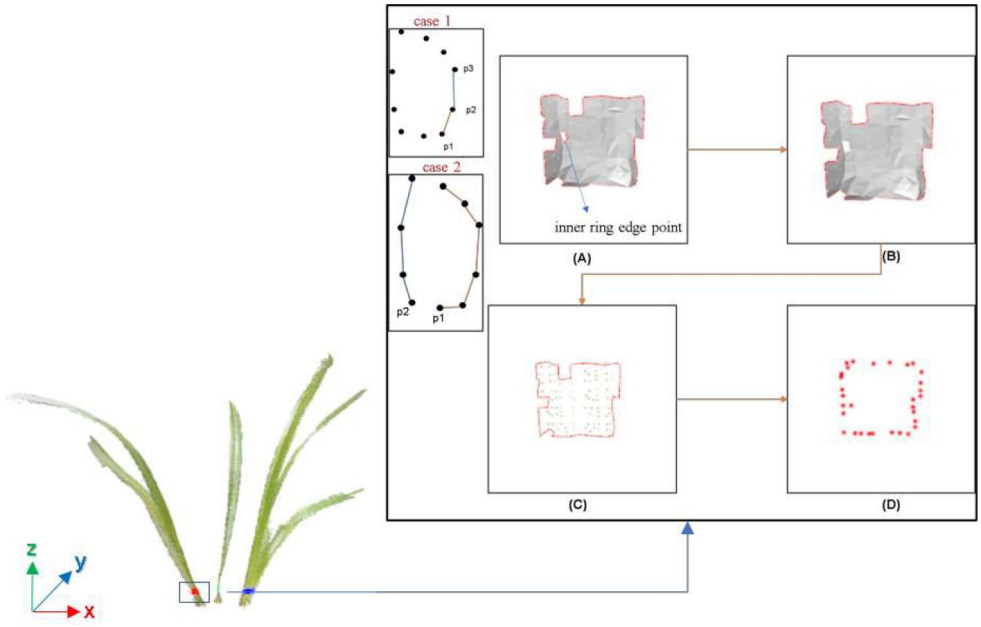

Fig. 11 Boundary point cloud extraction. (**A**) Boundary point extraction. The boundary points are marked in red, but these boundary points include inner ring edge points. (**B**) Triangular mesh of the outer boundary, and the outer boundary points are connected by red lines. (**C**) An outer boundary point cloud, and the outer boundary points are connected by red lines. (**D**) Outer edge points

found in the Figure that holes were generated during the triangular mesh construction. This was because the search radius of mesh was manually set for the distance of the neighborhood points, and if the distance between points was too far, thus exceeding the set search radius, the grid could not be formed between the far points, resulting in hole generation. However, if the search radius was blindly expanded, the hole problem was solved, but the one grid area contained more neighborhood points, thus losing the curvature details of the constructed surface. Therefore, the search radius needed to be controlled within a certain range to avoid the hole generation [35]. When a surface is generated with holes in the search process of the outer edge of a cluster, the edge of the inner hole would only be traversed once during the search, and it would be wrongly classified to the outer boundary point, so it was necessary to patch up these inner holes. It could be found that all edge points of one cluster or holes could form a ring by two adjacent edge points, while the outermost edges, the number of ring-containing points of the whole cluster was the largest. Therefore, it only needed to find the ring whose number of points was largest, the outer edge was located, and a set of ring extraction algorithms was proposed. For all the edges and points that had been traversed once, we stored these edge points in an edge set. Each element in the set was the 3D coordinates of the location of two points, that is, there was an edge between the two points. By analyzing the location coordinates of the two points, the rings were classified according to the following three cases:

(1) The location coordinates of the two points on the edge had not been recorded, indicating that these two points belonged to a new ring. We added these two points to a new ring set (the ring set was stored in the coordinates of the points).

(2) The location coordinates of one of the two points of the edge had been recorded. We added the point whose location coordinates had not been recorded to the ring set of the point whose location coordinates had been recorded, as shown in case 1 of Fig. 11. Assuming that the location coordinates of points p1 and p2 had already been added to a ring set, point p3 belonged to the same ring set of p1 and p2 when determining the points of p2 and p3,

(3) The position coordinates of both two points on the edge were recorded. If the two points were in different ring sets, we merged all the points of two rings together as one new ring. As shown in Fig. 11 case 2, when determining the points of p1 and p2, the position coordinates of points p1 and p2 were added to two different ring sets, all points in the ring set where point p2 was located (all points connected by all blue lines) were added to the ring set where point p1 was located (all points connected by all red lines), thus realizing the combination of sets.

By traversing all the elements of the edge set, we added all the points to different ring sets. The ring set with the largest number of points was used as the set of cluster boundary point cloud. Hence, the extraction of cluster boundary point clouds was completed.

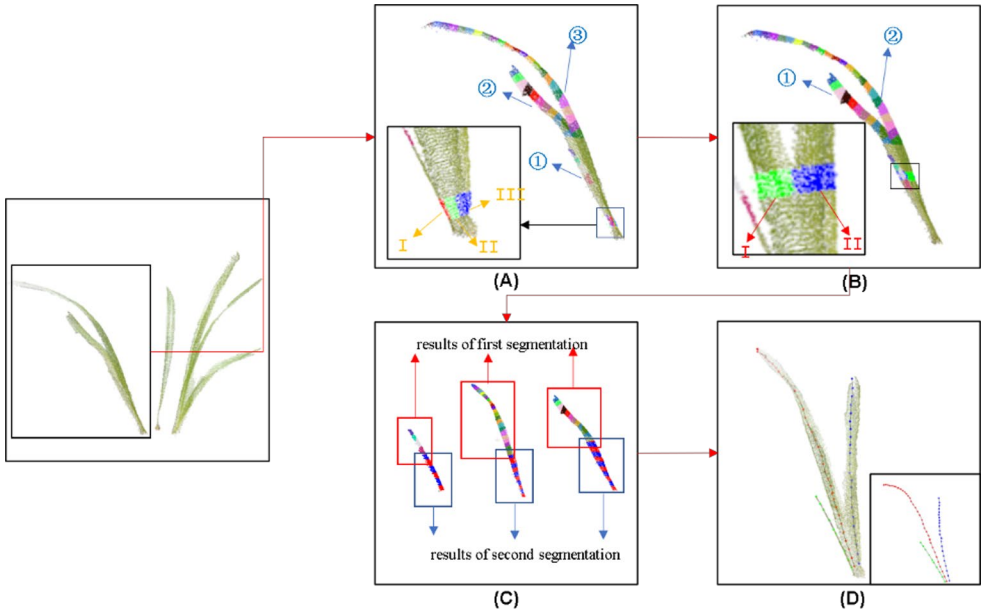

Fig. 12 Algorithm for second round segmentation

Second round segmentation (Algorithm3)

The aim of Algorithm 3 was to traverse each cluster in the point cloud of the overlapping part, and segment each cluster along the horizontal direction according to the weight ratio of each tiller in the point cloud of the non-overlapping part above it. Then, according to the principle of proximity, each part of the segmented cluster was joined with the corresponding non-overlapping parts of the tiller. This process segmented and merged the clusters in overlapping set from bottom to top, so that the point cloud of each tiller extended from non-overlapping set to the soil. And the complete point cloud of each tiller in the current ramet was obtained.

The main loop of the second round separated the point clouds of the overlapping set of all tillers under each ramet according to their corresponding non-overlapping set attribution, and then the separated part was combined with the non-overlapping set at the same tiller obtained by the first round based on nearest distance, so as to obtain the complete point clouds of different tillers under each ramet. The detailed pseudo-code of the second round of segmentation is shown in Fig. 12. In the second round of the segmentation process, overlapping set, non-overlapping set and the branch position set were as input, and each element of the overlapping set of the tillers at different ramets was taken out as the point cloud $cloud_o$. The average distance of each above tillers was Figured out by the average value of centroid of the positive and negative edge point clouds of all clusters in the non-overlapping set of the tiller where the $cloud_o$ was located was calculated, while the positive edge point cloud $cloud_{pos}$ and negative edge point cloud $cloud_{neg}$ were from the Algorithm 2. This average value of distance was used as the weight value of the tiller (each non-overlapping tiller had a weight value), and the weight ratio (weight ratio) corresponding to all tillers was obtained (line 2). The $cloud_o$ from the overlapping part was sliced starting from the lowest point in the vertical direction of the point cloud, along vertical direction (the horizontal slice was obtained), and the slice width was consistent with the slice layer width of the first round of segmentation (line 3). The main loop process of Algorithm 3 was repeated to obtain the complete point cloud of each tiller of all the ramets. The associated nomenclature in Algorithm 3 is shown in Table 3. The main process steps were as follows: The point cloud of each slice layer was stored in a temporary point cloud $cloud_{tmp}$(lines 4–9). Subsequently, the point cloud $cloud_{tmp}$ of each slice layer along the horizontal direction was sliced according to the weight ratio (line 10), and added the segmented point cloud in each horizontal slice layer to the non-overlapping point cloud closest to the slice according to the proximity principle (non-overlapping set is corresponded to the ramet and tiller point cloud) (line 11). (For example: Assuming that the minimum value of the horizontal (x-axis) coordinates of all point clouds in the slice layer along the vertical direction was 1, the maximum value

Table 3 Nomenclature in Algorithm3

Naming	Explanation
$Max_z()\,Min_z()$	The maximum and minimum values of the point cloud in the vertical direction
$widt?$	The horizontal width of the reference cluster

was 7, there were three tillers above and the weight ratio was 1:2:3. Then we classified the points under (x-axis) coordinates range from 1 to 2 as the points of tiller set 1, the points at range from 2 to 4 as the points of tiller set 2, and the points at range from 4 to 7 as the points of tiller set 3). In the process of searching, if the centroid of the slice layer was included in the record of the branch position set, it indicated that the part of the point cloud of non-overlapping tillers of the main tiller and the lateral tiller above has been segmented and recorded (line 13), and the subsequent lateral tiller was no longer involved in the segmentation of the overlapping part of the main tiller obtained by continuing the upward search. Therefore, in the program, the corresponding weight value to the non-overlapping tiller was set to 0 in the weight ratio (line 14). The Algorithm 3 main loop was repeated, and slicing was executed again along the vertical direction until the highest point Max_z in the vertical direction of $cloud_o$ was reached, and the complete point cloud set of each tiller of the ramet was obtained. The main flow of Algorithm 3 was run multiple times until the complete point cloud set of all ramets was located. As shown in Fig. 13A, the slice layer is sliced horizontally according to the weight ratio, and the sliced parts are added to non-overlapping set. Slice layer I, II and III correspond to the non-overlapping parts of ①, ② and ③ respectively. As shown in Fig. 13B, when the leftmost tiller is searched, the centroid of cluster is recorded in the branch position set, indicating that the tiller point cloud corresponding to the tiller has been segmented. The weight of the tiller is set to 0, and the slice layer is sliced horizontally according to the weight ratio and added to the corresponding tiller non-overlapping point cloud according to the corresponding serial number. Slice layers I and II correspond to the non-overlapping parts of ① and ② respectively. Figure 13C shows the complete point cloud of all tillers, which is composed of the segmented point clouds of the first and the second rounds of segmentation. After

Naming	Explanation
$Max_z()$ $M \in ¿_z()¿$	The maximum and minimum values of the point cloud in the vertical direction
$widt\,\hbar$	The horizontal width of the reference cluster

Algorithm 3 second round segmentation.

Input: *overlapping set , non overlapping set and location set.*

Output: *non overlapping set.*

```
for each overlapping point cloud O_c in overlapping set do
    calculate the leaf width weight ratio of all tillers of the ramet
    for z = Min_z(O_c); z<= Max_z(O_c); z+= width do
        create a temporary cloud temp_c to store slicing layer point cloud
        for each point in O_c do
            if point.z >= z and point.z <= z + width then
                add point to temp_c
            end if
        end for
        for each position in location set do
            if the vertical value of this position ∈ [z, z + width] then
                reset corresponding tiller weight of this position in non-overlapping set to 0
                recalculate the leaf width weight ratio
            end if
        end for
        slice temp_c along the horizontal direction based on weight ratio
        add each slice layer of temp_c to the corresponding point cloud in non-overlapping set
    end for
end for
```

Fig. 13 Second round segmentation. (**A**) Slice layers horizontally according to the weight ratio and add them to the corresponding tiller non-overlapping point cloud according to the corresponding sequence number. The red, green and blue clusters correspond to the non-overlapping parts of ①, ② and ③ respectively. (**B**) One tiller has been segmented, only two tillers are considered. The green and blue clusters correspond to the non-overlapping parts of ① and ② respectively. (**C**) The complete point cloud of all tillers is combined by the segmented point clouds of the first round of segmentation and the second round of segmentation. The red frame and the blue frame represent the segmentation results of the first round and the second round respectively. (**D**) The skeleton point connection, with red, blue and green colors represent the skeleton point of each tiller. (**E**) The skeleton point connection of all tillers of each ramet, and the skeleton point connection of all tillers of each ramet is distinguished by different color block diagrams. (**F**) The skeleton point connection of the whole plant

completing the first and second rounds of segmentation, the point cloud set where all the tillers are located can be obtained, and the centroids of each cluster in all the tiller sets are extracted. These centroids are used as the skeleton points of the tiller cluster, and all the skeleton points are connected from bottom to top to obtain the skeleton point connection diagram of all the tillers. The skeleton point connection of the ramet is shown in Fig. 13D, and the skeleton point connection of the whole plant is shown in Fig. 13E and Fig. 13F.

Morphological trait extraction

Figure 14A show the point cloud of typical tillers. For each tiller, five phenotypic parameters were figured out, plant height, leaf number, leaf length, leaf width, and leaf area. The height from the lowest to the highest point of the plant was defined as the plant height. Because Orchid might produce multiple ramets, the two-round segmentation was applied, the point cloud of each complete tiller was stored in the non-overlapping set. In this study, the height of all tillers was calculated (the difference between the vertical Z-direction values of the highest and lowest points of each tiller). And the maximum value of the difference was used as the estimated value of plant height. Figure 14B shows the height of each tiller. For the number of leaves, the number of elements in the non-overlapping part set was the number of leaves of the plant. In Fig. 14, the red, orange and blue block diagrams represent three different ramets. The distance between centroids of the searched clusters of each tiller in two round segmentation was calculated, and leaf length of the tiller was represented by the sum of the distances. As shown in Fig. 14C, we take the sum of the distances of all the centroids of tillers as the estimated value of tiller leaf length. Leaf width was defined as the maximum width of a tiller whose leaf length was the longest of one Cymbidium. In the steps of Algorithm 2 (lines 20–27), different points (excluding the points in the upper and lower edges (line 9 and line 13)) were classified in the positive and negative directions of the connect line. The positive boundary point cloud $cloud_{pos}$ and the negative boundary point cloud $cloud_{neg}$ (lines 20–27) were stored respectively, and the centroids of the positive boundary point cloud $cloud_{pos}$ and the negative boundary point cloud $cloud_{neg}$ were calculated respectively. The maximum distance between the centroid points of the positive and negative edge point clouds of all clusters of tillers with maximum leaf length was used as the estimation value of leaf width. Figure 14D shows the connections of the cluster centroid points on the positive and negative sides of each tiller. Leaf area was the sum of the area of all triangular mesh and the area of triangular holes of all clusters of each segmented tiller.

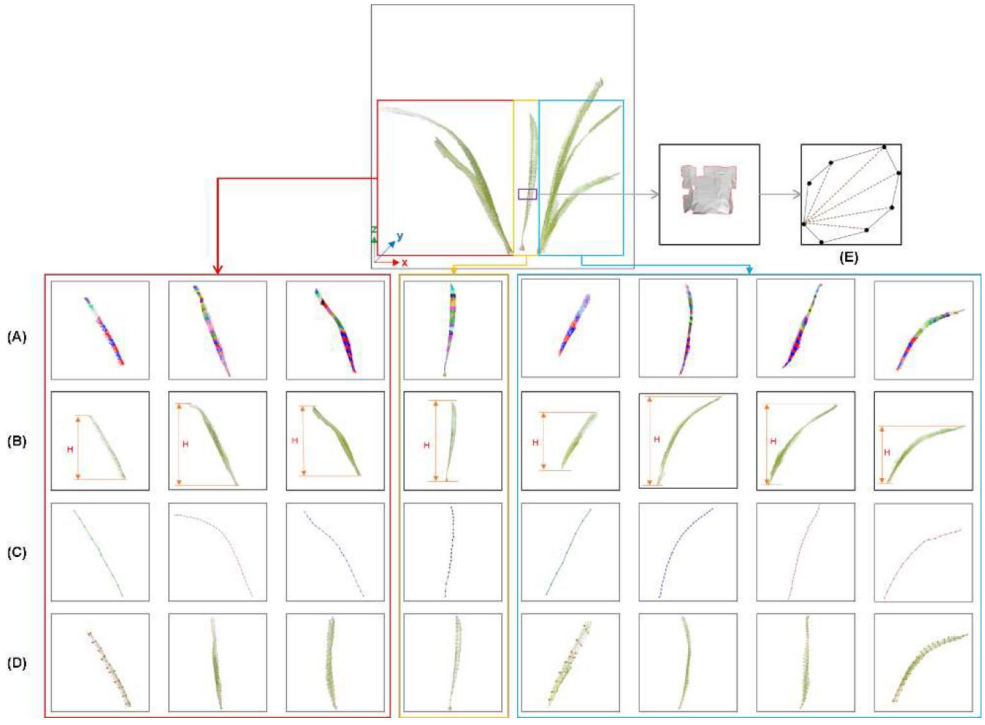

Fig. 14 Morphological trait extraction. (**A**) Point cloud of each tiller. The red, orange, and blue block diagrams represent all tillers of the first, second, and third ramets, respectively. (**B**) Height H of each tiller, and the highest H as the estimated value of plant height. (**C**) The centroid connection of each tiller, and the sum of the distances of all centroid is the estimation value of leaf length. (**D**) The connections of the centroids on the positive and negative sides of each cluster of tillers. (**E**) The calculation method of the area of the internal hole or mesh, based on triangle meshes, the areas of all the meshes or holes are calculated

For the small hole or mesh inside the leaf tiller, as shown in Fig. 14E, we selected one point of the edge as the center and formed triangles with the remaining points in the hole or mesh edge. The triangle was obtained by the coordinates of the three points and the area sum of all the triangles was calculated as the area of the hole. Through the above method, the sum of the triangular meshes and the internal hole area of all clusters was calculated and used as the estimation value of the leaf area of the whole tiller.

Results

In this study, three representative Orchid seedlings, Molan, Qingfeng and Red Beauty, were acquired for verification, and each variety had 15 typical samples with the height range of 15–50 cm. Figure 15 shows the represented seedlings at image processing, Fig. 15A shows the plants after preprocessing, Fig. 15B shows the results of clustering along the vertical slice of 2–3 cm above the soil, where we estimated the number of ramets by the number of clusters. For each Cymbidium, our study performed two rounds of segmentation, the non-overlapping part of tillers and the overlapping part of tillers were determined by the first round of segmentation, as shown in Fig. 15C. Subsequently, the complete point cloud of each tiller was obtained by the second-round segmentation, as shown in Fig. 15D. For the leftmost Cymbidium, in the first round of segmentation process, each ramet had only one tiller, and the complete point cloud of each tiller had been obtained, hence it was not necessary to segment the overlapping point cloud of tillers through the second-round segmentation. After the first and second rounds of segmentation, the centroid of each cluster was extracted as the skeleton point. And the skeleton points were connected from bottom to top, to obtain the point cloud skeleton connections, as shown in Fig. 15E.

Accuracy assessment

A total of 157 tillers of 45 Orchid seedlings of heights 15–50 cm were verified and compared. Five phenotypic parameters of plant height, leaf number, leaf length, leaf width and leaf area were extracted and compared. From the above analysis, the accuracy of leaf number was 100%. Three indicators of correlation coefficient (r), root mean square error (RMSE) and mean absolute error (MAE) [36], were used to evaluate the accuracy of the algorithm. The comparison and charts of phenotypic parameters between manual measurements and system estimates are shown in Fig. 16.

Among them, as shown in Fig. 16A, the correlation coefficient (r=0.99) between the plant height estimated by the system and the actual measured value, RMSE and MAE were 0.46 cm and 0.32 cm, respectively. As shown in Fig. 16B, the median, maximum and minimum values of the predicted values were 23.4 cm, 41.9 cm and 12.8 cm, respectively. The median, maximum and minimum values of the actual measured values were 23.7 cm, 41.4 cm and 13 cm, respectively. As shown in Fig. 16C, the correlation coefficient (r=0.94) between the leaf length estimated by the system and the actual measured value, RMSE and MAE were 2.5 cm and 2.17 cm, respectively. As shown in Fig. 16D, the median, maximum and minimum values of the estimated values were 26.4 cm, 49.9 cm and 12.4 cm, respectively. The median, maximum and minimum values of the actual measured values were 27.3 cm, 49.4 cm and 13.2 cm, respectively. As shown in Fig. 16E, the correlation coefficient (r=0.91) between the leaf width estimated by the system and the actual measured value, RMSE and MAE were 0.37 cm and 0.33 cm, respectively. As shown in Fig. 16F, the median, maximum and minimum values of the estimated values were 2.96 cm, 4.87 cm and 1.49 cm, respectively. The median, maximum and minimum values of the actual measured values were 2.94 cm, 4.95 cm and 1.13 cm, respectively. As shown in Fig. 16G, the correlation coefficient (r=0.92) between the leaf area estimated by the system and the actual measured value, RMSE and MAE were 16.66 cm^2 and 15.38 cm^2, respectively. As shown in Fig. 16H, the median, maximum and minimum values of the estimated values were 60.61 cm^2, 181.4 $\mathrm{cm}^2$and 21.47 cm^2, respectively. The median, maximum and minimum values of the actual measured values were 66.23 cm^2, 204.5 cm^2 and 23.78 cm^2, respectively. And the paired violin plot with four parameters is shown in Fig. 16I to Fig. 16L respectively.

For plant height, the average measured manually was 23.29 cm, and the average absolute error was 1.4% of the average. It was proved that this algorithm was suitable for predicting plant height. The reason for the error was that there were bulbs in the Cymbidium, and some bulbs would be exposed to the soil. In the process of manual measurement, the height of the bulb part was excluded from the plant height. However, the bulb was higher than the soil plane and would not be removed when the point cloud under soil plane was removed. Therefore, the rest of the bulb part would be included in the plant height, leading to an error that could be ignored in real application.

For leaf length, the average leaf length of all leaves measured manually was 27.81 cm, and the accuracy of leaf length was 92.2%. The reason for the error was due to the algorithm itself. The centroid was used as the representative of each cluster in the tiller. However, for some specific parts such as the tip of the tiller, the actual centroid was slightly shifted from the one got by our algorithm. And in the manual measurement, we took the distance from the soil plane to the tip of the leaf as the artificial measurement value of the leaf length. Therefore,

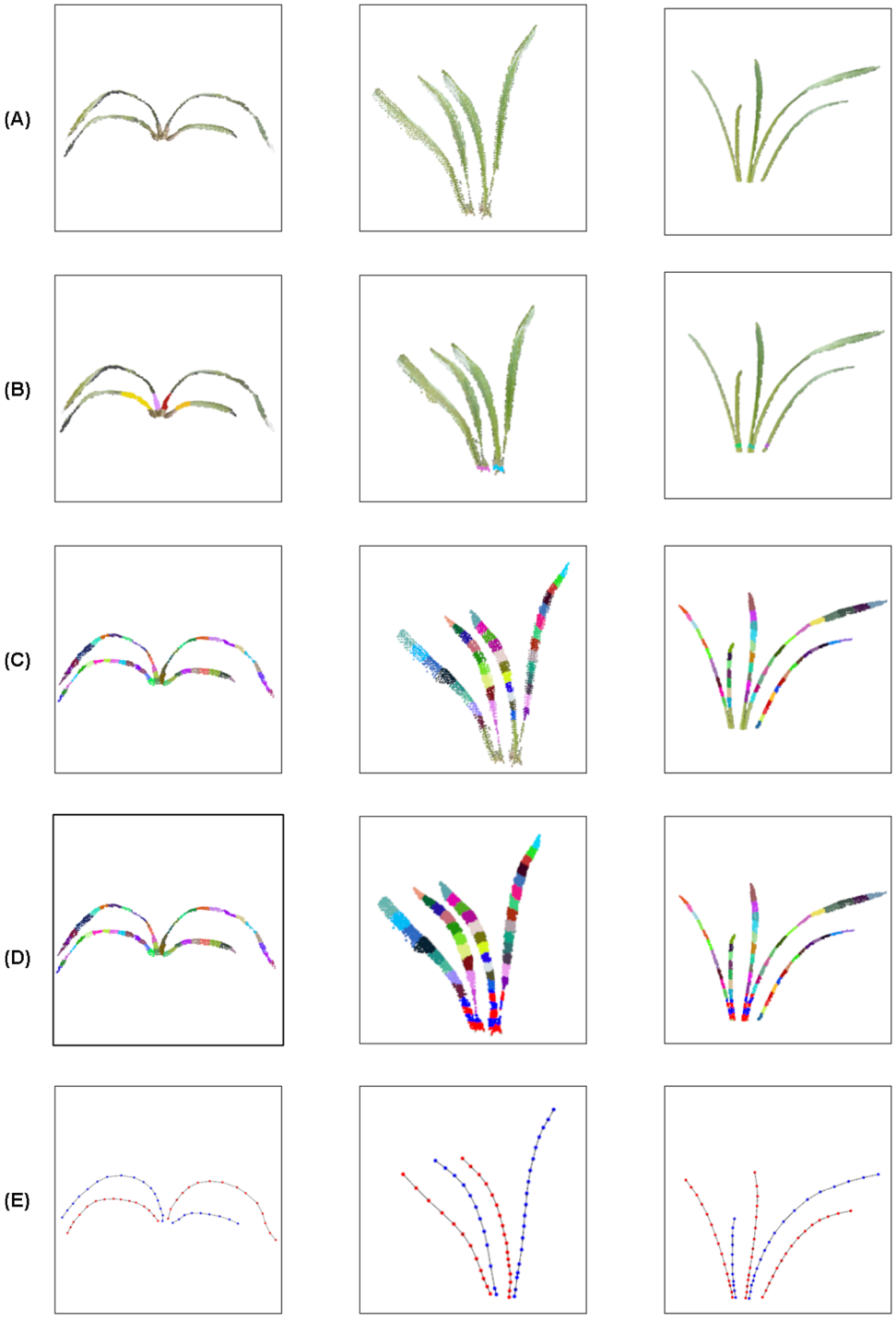

Fig. 15 Procedure of tiller segmentation. (**A**) Preprocessed point cloud. (**B**) The number of ramets (initial cluster) determined. The initial cluster of each ramet was marked with different colors. (**C**) Results of first round of segmentation. The output is marked with different colors. (**D**) Results of second round of segmentation. The output is marked with red and blue colors. (**E**) The skeleton points of different tillers were recorded by red and blue colors

if we calculated the distance from the tiller tip to the centroid of the top cluster, and added this distance value to the estimation value, which would improve the accuracy of the estimation value. However, because the tiller tip did not have obvious characteristics, it was difficult to determine the position of the tiller tip by the algorithm. However, compared with the traditional methods, if the conventional vertical slice method was used, the average absolute error of the test accounted for 24.1% of the

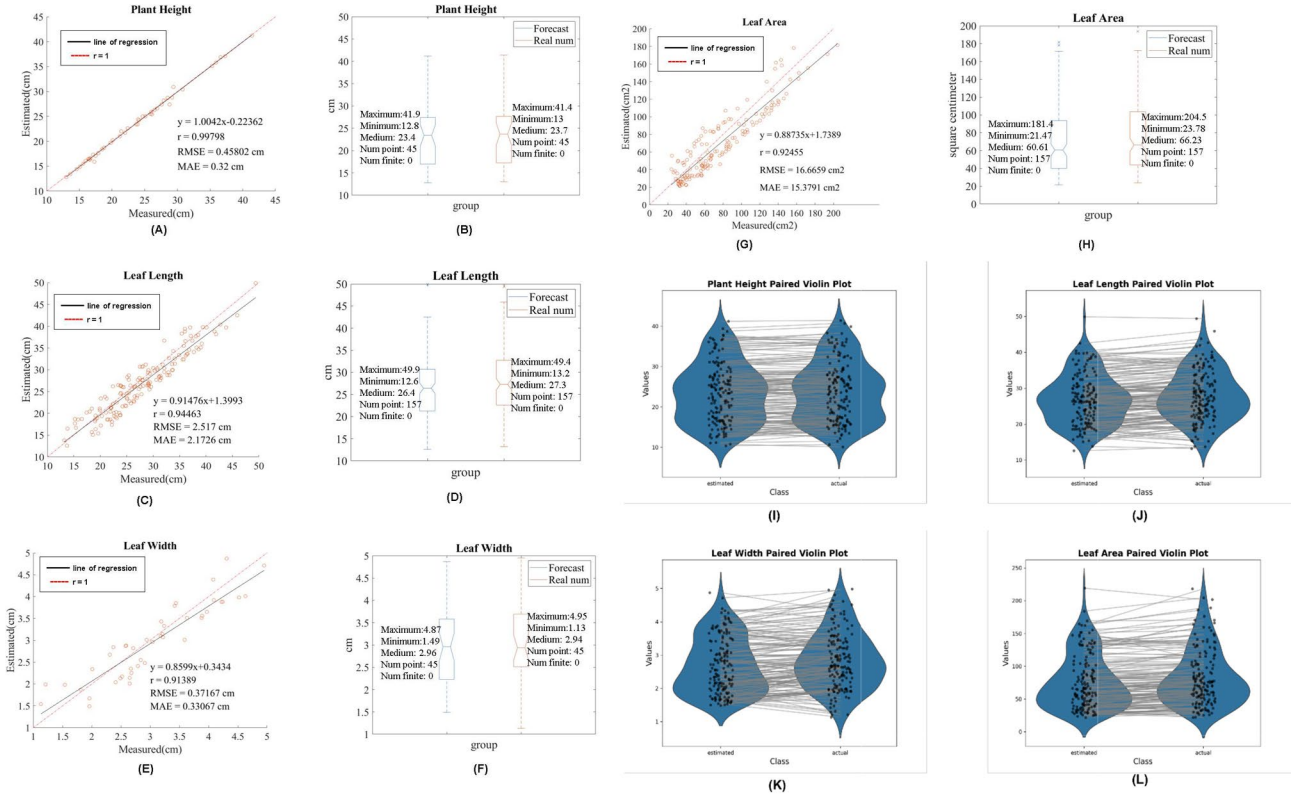

Fig. 16 Comparison of four phenotypic parameters. In Figure. **A**, **C**, **E** and **G**, the red dotted line represents the line whose predicted value is equal to the actual value ($r = 1$), and the black line represents the fitted line. In Figure **B**, **D**, **F**, **H**, the blue group represents the predicted value, the orange group represents the actual value. Figures **I**, **J**, **K**, and **L** show the corresponding violin plot between the estimated and actual values

average value. Therefore, our algorithm would greatly improve the accuracy of leaf length.

For leaf width, the average leaf width of all tillers measured manually was 3.04 cm, and the average absolute error accounted for 10.9% of the average. For each ramet, the part near the soil would overlap at camera view, and it was difficult to distinguish the leaf structure. The second round of segmentation was performed horizontally according to the tiller width obtained by the first-round segmentation. However, the tillers of Orchids were different at different angles, and it was obvious that using a ratio to segment all overlapping parts would generate errors. Moreover, the manual measurement would also produce certain errors.

For leaf area, the average leaf area of all tillers was 86.9 cm^2, and the accuracy was 82.3%. The verification method was different from the other three parameters, and the way we verified the leaf area was to remove the tillers from the plant and use the printer's copy function to copy the tillers onto a piece of A4 paper. The proportion of points with non-zero pixels in A4 paper to all points was multiplied by the area of A4 paper as the artificial value of the leaf areas. In the process of copying, the leaves would be flattened, so the actual value would be a bit larger than the estimated value. Another reason was the tillers overlapping. When the overlapping part of the tillers was sliced, we used a fixed distance along the horizontal direction, but different parts of tillers would have different growth conditions, so the slicing method would produce errors itself. However, it was obviously unable to improve the accuracy of leaf area by destroying the morphological structure of plants on the site. For this low-cost rapid measurement method, the accuracy of leaf area could reach 82.3%, which also showed that the algorithm had certain feasibility and promotion significance.

Discussion

A set of phenotypic parameter measurement method for Orchid seedlings was developed to process the point cloud in this study. There are several ramets of Cymbidium, and each ramet has tillers. Therefore, the proposed method mainly focused on segmenting tiller and ramet, and five parameters including plant height, leaf number, leaf length, leaf width and leaf area for each tiller were Figured out. The tiller segmentation algorithm had two rounds, and in the first round, the non-overlapping parts and overlapping parts of tillers were extracted. In the second round, the overlapping part of tiller was segmented according to the weight proportion of the average leaf width of tillers above, to get whole point cloud of all

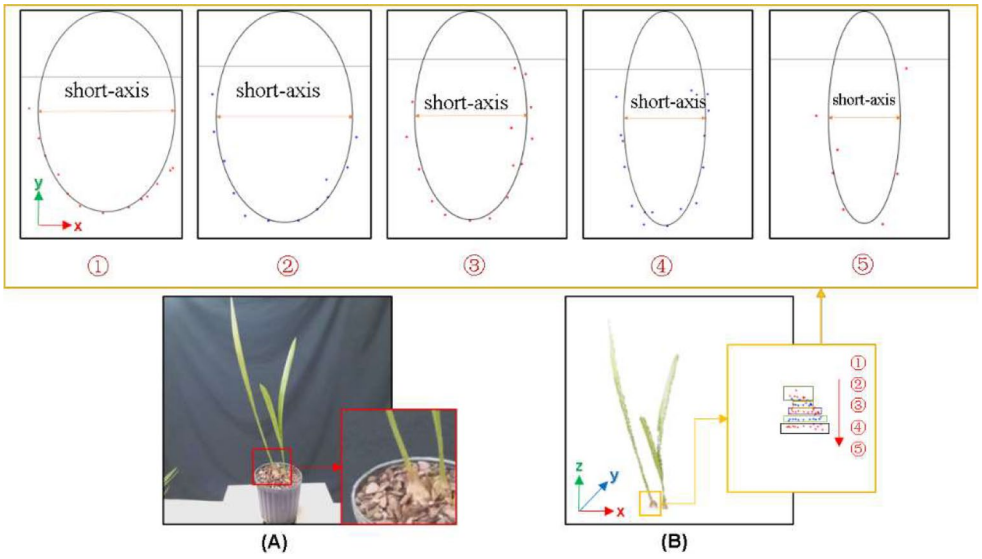

Fig. 17 Bulb interference. (**A**) The bulbs of Cymbidium are marked in red. (**B**) The bulbs were sliced along the vertical direction, and 2D ellipse fitting was performed. From ① to ⑤, the projection of each slice layer is shown. The orange line is the short axis of the fitted ellipse

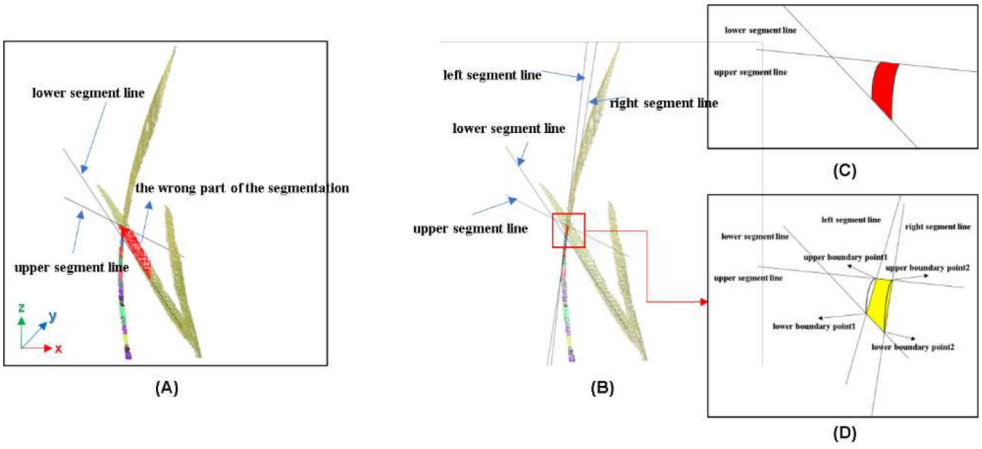

Fig. 18 The tiller overlapping error analysis. (**A**) When the tillers overlap, the rest of the tillers will be mistakenly added to the next round of cluster. The part of the red mark is the wrong segmentation part. (**B**) Left, right, upper and lower segment lines are distinguished. (**C**) The actual cluster of current tillers, marked in red. (**D**) The cluster obtained by our proposed algorithm is marked in yellow

tillers. For phenotypic parameters, the accuracy of plant height, leaf number, leaf length, leaf width and leaf area were 98.6%, 100%, 92.2%, 89.1% and 82.3%, respectively.

For the technical prospect, the following improvements of our proposed method have also been made. The corresponding result is shown in Fig. 17. First, there was the bulb problem, as shown in Fig. 17A. In the growth process of Orchid seedlings, each ramet would spread pseudo-bulbs, and some bulbs broke above the soil, in the preprocessing, the bulb part would not be classified as the soil and removed. When calculating the length of the tiller, the length of the bulb would be included in the leaf length. In the actual manual measurement, the length of the bulb part would not be included, leading to such errors. Because the bulb was close to an ellipsoid and the width of the bulb was much larger than that of the tiller, we sliced the bulb part horizontally in the vertical direction. As shown in Fig. 17B, for each slice layer in the vertical direction, we projected the slice layer onto the XOY plane and performed 2D ellipse fitting. From bottom to top, ①-⑤ represent each vertical slice layer, respectively. It could be found that from ① to ⑤, the length of short axis become smaller. Therefore, when the short axis length of fitting 2D ellipse at the slice layer was close to the average leaf width of tillers, this slice layer was the boundary between tiller and bulb, and the bulb height was got which was subtracted from the leaf length, hence, the accuracy of the leaf length was greatly improved. The second case was the overlapping of tillers, as shown in Fig. 18. In Fig. 18A, the tillers of different ramets were

overlapping. In the process of segmentation of one tiller, when the point between the lower segment line and the upper segment line was obtained as a slice layer(cluster), the point cloud of other tiller was mistakenly added to this layer, causing the large width of the slice layer and an error slicing has occurred. Therefore, the left and right vertices on both sides of the upper and lower boundary point cloud (a total of four points) were selected, and four points were connected in pairs according to the way shown in Fig. 18B. The line connected by the two upper vertices of the upper boundary was defined as the upper segment line, and the line connected by the two lower vertices of the lower boundary was defined as the lower segment line. The line connected by the left and right vertices constituted the left and right segment lines respectively. Within the four segment lines (only consider the XOZ plane), the points were used as the cluster or slice of tiller of current round. According to our improved segmentation method, (as shown in the yellow marker part of Fig. 18D), compared with the actual cluster part (as shown in the red part of Fig. 18C) segmentation by the left and right segmentation lines reduces errors caused by the overlapping. Because the area of the overlapping cluster itself was also small, the error accounted for a small proportion of the entire leaf, which was feasible to further application. In the actual, we rarely met these two special situations. Among the 157 tillers we analyzed, there were only 8 special situations. For these two situations, we solved the problems by the scheme described above.

Conclusion

Based on the low-cost visual phenotype system of Azure Kinect depth camera, this study proposed an automated measurement method for five phenotypic parameters of Cymbidium seedlings at a height of 10–50 cm. After verification, the measurement results of phenotypic parameters met the application real requirements. The system and algorithm of this study provided an efficient and economical solution for plant phenotypic feature extraction, which could provide a technical basis for subsequent growth model establishment, genome research, plant breeding and so on.

Acknowledgements
The authors would like to thank Dr. Zhuyun Wang for its in-kind support.

Author contributions
Y.Z and H.H.Z. wrote the main manuscript text and Y.C. prepared materials. Y.Z and H.H.Z. designed the methods. All authors reviewed the manuscript.

Funding
"Three Institute and Nine Party" Agricultural Science and Technology Cooperation Plan of Zhejiang Province(2023SNJF027), Zhejiang Key Research and Development Program (2021C02038 & 2021C02061), "Pioneer" and "Leading Goose" R&D Program of Zhejiang (2023C02028, 2023C01254),Zhejiang Province Commonweal Projects (LGG21F010003), General Projects of Agricultural and Social Development in Hangzhou (202203B06).

Data availability
No datasets were generated or analysed during the current study.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare no competing interests.

Received: 24 March 2024 / Accepted: 23 September 2024
Published online: 30 September 2024

References

1. Zhou Z et al. *Orchid conservation in China from 2000 to 2020: Achievements and perspectives.* 2021. 43(5): pp. 343–349.
2. Xiao Q et al. *Advanced high-throughput plant phenotyping techniques for genome-wide association studies: A review.* 2022. 35: pp. 215–230.
3. Yasrab R, et al. Predicting plant growth from time-series data using deep learning. Remote Sens. 2021;13:331.
4. Das Choudhury S, et al. Holistic and component plant phenotyping using temporal image sequence. Plant Methods. 2018;14:1–21.
5. Green JM, et al. PhenoPhyte: a flexible affordable method to quantify 2D phenotypes from imagery. Plant Methods. 2012;8:1–12.
6. Gibbs JA, et al. Plant phenotyping: an active vision cell for three-dimensional plant shoot reconstruction. Plant Physiol. 2018;178(2):524–34.
7. Borges LM et al. Schrödinger's phenotypes: Herbarium specimens show two-dimensional images are both good and (not so) bad sources of morphological data. 2020. 11(10): pp. 1296–308.
8. Wang Y, Chen YJP. Non-destructive measurement of three-dimensional Plants based on point cloud. Plants. 2020;9(5):571.
9. Burt A, et al. Extracting individual trees from lidar point clouds using treeseg. Methods Ecol Evol. 2019;10(3):438–45.
10. Wu S, et al. An accurate skeleton extraction approach from 3D point clouds of maize plants. Front Plant Sci. 2019;10:248.
11. Ai M, Yao Y, Hu Q, Wang Y, Wang W. An automatic tree skeleton extraction Approach based on Multi-view Slicing using terrestrial LiDAR Scans Data. Remote Sens. 2020;12(22):3824.
12. Hu C, Pan Z, J.J.o.A T. Leaf and wood separation of poplar seedlings combining locally convex connected patches and K-means + + clustering from terrestrial laser scanning data. J Appl Remote Sens. 2020;14(1):018502.
13. Miao Y, et al. Banana plant counting and morphological parameters measurement based on terrestrial laser scanning. Plant Methods. 2022;18(1):1–16.
14. Xiang L, et al. Automated morphological traits extraction for sorghum plants via 3D point cloud data analysis. Comput Electron Agric. 2019;162:951–61.
15. Teng X, et al. Three-dimensional reconstruction method of rapeseed plants in the whole growth period using RGB-D camera. Comput Electron Agric. 2021;21(14):4628.
16. Yang T, et al. 3D reconstruction method for tree seedlings based on point cloud self-registration. Comput Electron Agric. 2022;200:107210.
17. Liu Y, et al. Fast reconstruction method of three-dimension model based on dual RGB-D cameras for peanut plant. Plant Methods. 2023;19(1):17.
18. Zhou H et al. A fast phenotype approach of 3D point clouds of Pinus massoniana seedlings. Front Plant Sci, 2023. 14.
19. Nguyen TT, et al. Structured light-based 3D reconstruction system for plants. Sensors. 2015;15(8):18587–612.
20. Rosell-Polo JR, et al. Advances in structured light sensors applications in precision agriculture and livestock farming. Adv Agron. 2015;133:71–112.
21. Nguyen TT, et al. *Plant phenotyping using multi-view stereo vision with structured lights*. In *Autonomous air and ground sensing systems for agricultural optimization and phenotyping*. SPIE; 2016.

22. Al Khalil OJOSJ. Structure from motion (SfM) photogrammetry as alternative to laser scanning for 3D modelling of historical monuments. Open Sci J, 2020. 5(2).
23. Wang Y, et al. 3DPhenoMVS: a low-cost 3D tomato phenotyping pipeline using 3D reconstruction point cloud based on multiview images. Agronomy. 2022;12(8):1865.
24. Miao T, et al. Automatic stem-leaf segmentation of maize shoots using three-dimensional point cloud. Comput Electron Agric. 2021;187:106310.
25. Cao J et al. *Point cloud skeletons via laplacian based contraction*. in. 2010 *Shape Modeling International Conference*. 2010. IEEE.
26. Dalitz C, Schramke T, Jeltsch MJIPOL. Iterative Hough transform for line detection in 3D point clouds. Image Process Line. 2017;7:184–96.
27. Ma Z et al. *Optimization of 3D Point Clouds of Oilseed Rape Plants Based on Time-of-Flight Cameras.* 2021. 21(2): p. 664.
28. Dziubich T et al. Depth Images Filter Distrib Streaming 2016(2): pp. 91–8.
29. Zhang Z. J.I.j.o.c.v. Iterative Point Matching Registration free-form Curves Surf. 1994;13(2):119–52.
30. Schrader J et al. Leaf size estimation based on leaf length, width and shape. 2021. 128(4): pp. 395–406.
31. Mineo C, et al. Novel algorithms for 3D surface point cloud boundary detection and edge reconstruction. J Comput Des Eng. 2019;6(1):81–91.
32. Thurmond VA. J.J.o.n.s., *the point of triangulation*. J Nurs Scholarsh. 2001;33(3):253–8.
33. Alexa M, et al. Computing and rendering point set surfaces. IEEE Trans Vis Comput Graph. 2003;9(1):3–15.
34. Borouchaki H. S.J.C.m.i.a.m. Lo, and engineering, *Fast Delaunay triangulation in three dimensions*. Computer Methods in Applied Mechanics and Engineering, 1995. 128(1–2): pp. 153–167.
35. Feng C, et al. A fast hole-filling method for triangular mesh in additive repair. Appl Sci. 2020;10(3):969.
36. Chai T. and R.R.J.G.m.d. Draxler, *Root mean square error (RMSE) or mean absolute error (MAE)?–Arguments against avoiding RMSE in the literature*. 2014. 7(3): pp. 1247–50.

Publisher's note