Plant Phenotyping by Deep-Learning-Based Planner for Multi-Robots

Chenming Wu, Rui Zeng, Jia Pan, Charlie C. L. Wang, and Yong-Jin Liu

***Abstract*—Manual plant phenotyping is slow, error prone, and labor intensive. In this letter, we present an automated robotic system for fast, precise, and noninvasive measurements using a new deep-learning-based next-best view planning pipeline. Specifically, we first use a deep neural network to estimate a set of candidate voxels for the next scanning. Next, we cast rays from these voxels to determine the optimal viewpoints. We empirically evaluate our method in simulations and real-world robotic experiments with up to three robotic arms to demonstrate its efficiency and effectiveness. One advantage of our new pipeline is that it can be easily extended to a multi-robot system where multiple robots move simultaneously according to the planned motions. Our system significantly outperforms the single robot in flexibility and planning time. High-throughput phenotyping can be made practically.**

***Index Terms*—Agricultural automation, multi-robot systems, computer vision for automation.**

I. Introduction

PLANT phenotyping is an active research area that bridges genotypes and phenotypes. Nowadays, genomics research can yield a lot of information. Unfortunately, the data generated by sequencing technology far exceeds the current capacity of plant phenotyping [1]. Traditional plant phenotyping heavily relies on laborious and expensive manual operations [2]. It is essential to facilitate phenotyping in efficiency and effectiveness so that researchers working on plant genomes can easily realize the agricultural promise of plant genomics. As an example, breeders usually use phenomic data to study the problem of improving crop yields [3]. Plant phenotyping assesses a variety of plant traits such as growth, development, tolerance, resistance, architecture, physiology, ecology, height, leaf shape, etc. [4], most of which must be measured on a complete 3D model [5].

However, the clutter and occlusion problems caused by naturally grow of plants make phenotyping challenging. It is also difficult to develop a general prototype for a variety of plants. A few works (e.g., [6], [7]) specify fixed viewpoints for the robots to solve the phenotyping problem, apparently requiring repeated fine-tuning of almost every plant. With the development of intelligent robots and computer vision, developing automated systems to improve traditional phenotyping becomes possible, and some automated phenotyping systems have been proposed in the past years. These results have greatly accelerated the breeding process and genetic analysis of precision agriculture.

Next-best view (NBV) planning is a general method in robotics that tries to collect information of an unknown object. The goal of NBV planning is very similar to the one of plant phenotyping. Typically, NBV is achieved by searching for the best viewpoint of the sensors mounted on robots over a set of candidate viewpoints, to maximize the expected gain of information (i.e., minimize expected entropy). However, conventional NBV algorithms are designed for general 3D exploration tasks, which only consider the enclosed volume as a solid. Conversely, plant phenotyping tasks have stronger prior knowledge – the structure of the plant. In this letter, we make an effort to improve the effectiveness of the NBV algorithms in plant phenotyping by incorporating the prior knowledge of plant structure.

In this letter, we propose to use deep learning technique to learn the underlying structure of plants. After having the priors of learned structure, we design a new method to compute next-best viewpoints accordingly. A promising advantage of this method is that it can be easily extended to multi-robot systems. Specifically, we develop such a system equipped with multiple robotic arms that can be manipulated simultaneously (see Fig. 1). To demonstrate the functionality of a multi-robot system in plant phenotyping, we apply the proposed NBV planner to achieve better performance compared to conventional NBV methods. Our system solves phenotyping problem generally without making any assumption about the type of phenotypic data. In summary, we make the following contributions:

- An efficient NBV planning method based on deep learning, which takes advantage of the structural prior of plants and provides a way for computing the information gain of candidate viewpoints.

Manuscript received February 25, 2019; accepted June 11, 2019. Date of publication June 21, 2019; date of current version July 12, 2019. This letter was recommended for publication by Associate Editor A. Kim and Editor Y. Choi upon evaluation of the reviewers' comments. This work was supported in part by the Natural Science Foundation of China under Grants 61725204, 61521002, and 61628211; in part by the Royal Society-Newton Advanced Fellowship under Grant NA150431; in part by the MOE-Key Laboratory of Pervasive Computing, in part by the CUHK Direct Research Grant (4055094); and in part by the grant from Science and Technology Department of Jiangsu Province, China. *(Corresponding author: Yong-Jin Liu.)*

C. Wu and R. Zeng are with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China (e-mail: wcm15@mails.tsinghua.edu.cn; zengr17@mails.tsinghua.edu.cn).

J. Pan is with the Department of Computer Science, The University of Hong Kong, Hong Kong (e-mail: jpan@cs.hku.hk).

C. C. L. Wang is with the Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, Hong Kong (e-mail: cwang@mae.cuhk.edu.hk).

Y.-J. Liu is with the Beijing National Research Center of Information Science and Technology, Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China (e-mail: liuyongjin@tsinghua.edu.cn).

This letter has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors. The material contains the video to demonstrate the experiments conducted by our proposed system. The video is given in MP4 file format. The size of the video is 19.7 MB.

Digital Object Identifier 10.1109/LRA.2019.2924125

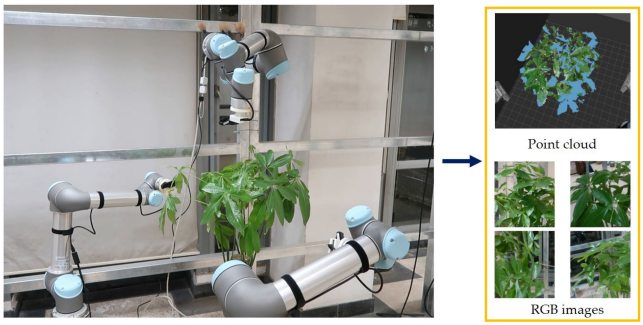


Fig. 1. An overview of our multi-robot system for efficiently completing the plant phenotyping tasks. The picture on the left shows the system setup, which consists of three UR-5 robot arms with a fixed base. The figure on the right shows the results obtained by positioning the sensors to different viewpoints by our deep network based NBV planner.

- A practical solution that can easily extend the proposed learning-based NBV planning method from a single robot to a multi-robot system that does not sacrifice too much speed of planning.

Experimental tests have demonstrated that our approach can obtain results with better completion. The rest of this letter is organized as follows – Section II reviews literature, Section III presents an overview of our robotic phenotyping system; Section IV explains the deep NBV planner; Section V describes all experiments conducted on our system, including both the simulation and the real environment results; future works and conclusion are given in Section VI.

II. Related Work

This section briefly reviews the relevant research of plant phenotyping from three perspectives – robot-assisted phenotyping, deep-learning-based phenotyping, and NBV planning.

A. Robot-Assisted Phenotyping

Compared to laborious phenotyping, the rapid development of robotics has significantly facilitated plant phenotyping in many ways. In the early stage of robot-assisted phenotyping, plants are grown on top of the conveyor system and are automatically transferred to an X-Y scanning cabinet on a regular basis [8]. Lemnatec [9] has built automated phenotyping platforms for laboratory, greenhouse, and outdoor scenes. The platforms comprise various sensors for phenotyping. Ruckelshausen *et al.* [10] propose an autonomous on-site robot platform, namely *BoniRob*, which can be used in single-plant phenotyping. Alenyà *et al.* [11] develop a single-robot system for probing plant leaves from different viewpoints, followed with an NBV-based planning method for actively exploration [12]. Muller-Sim *et al.* [13] invent a field-based mobile system called as *Robotanist* that navigates under the canopy of row crops for outdoor high-throughput phenotyping. Sa *et al.* [14] design an autonomous crop harvesting system consisting of a robotic arm and equipped cutting tools, harvesting the pepper.

One major problem in single robot systems is the limited flexibility (i.e., the reachability of a robotic arm). To solve this problem, a new idea is emerged to manipulate multiple robots simultaneously or asynchronously. As a result, effectiveness or efficiency or both can be improved. To the best of our knowledge, only Gao *et al.* [15] attempt to incorporate multi-robot systems into phenotyping. They design a team of AGV robots to collect several soybean canopy images in outdoor fields. In contrast, we target at phenotyping in laboratory environments.

B. Deep-Learning Based Phenotyping

Autonomous robots significantly reduce human intervention when collecting phenotypic data. Post-processing is crucial to translating it into an interpretable knowledge for agricultural experts. Many efforts have been devoted to the interpretations. We refer the interested readers to a comprehensive review [16] because of space constraint. However, the use of deep-learning techniques to guide robot manipulations in-situ has yet to receive widespread acceptance for robot-assisted phenotyping tasks. McCool *et al.* [17] design a lightweight deep network for real-time weed segmentation for robotic weeding-decision making. The system proposed in [14] uses a deep neural network to detect peduncles of sweet peppers for harvesting. Milioto *et al.* [18] propose a *Convolutional Neural Network* (CNN) network for a similar weeding purpose by leveraging background knowledge. Parhar *et al.* [19] use a variant of *Generative Adversarial Network* (GAN) for in-situ sorghum stalk detection and grasping.

C. Planning for Next-Best View

An intelligent and complete phenotyping system requires an algorithm to drive. However, most existing phenotyping robots lack this. The underlying problem of plant phenotyping is conceptually similar to a particle application of NBV. In the line of NBV planning (note that we only discuss non-model based NBVs in this letter), existing methods mainly rely on frontier-based exploration, where the sensor is iteratively moved to a viewpoint that maximizes information gain [20]. The measurement of the information gain is calculated in the domain of either volumetric-space (ref. [12], [21]–[24]) or surfel-space (ref. [25]). The idea behind volumetric-NBV is simple yet effective: the target 3D object is enclosed with an occupancy grid and act space carving progressively until reaching a specific termination criterion. Our work is established on top of it but additionally uses well-defined plant structural information.

It is worth noting that the deployment of multi-robot systems for phenotypic analysis, including mechanical design and algorithms, is still in its infancy. We address this challenging problem and propose a scalable NBV planner for single-robot or multi-robot phenotyping system.

III. Multi-Robot System

We develop a multi-robot plant phenotyping system having three robotic arms, where each is equipped with an RGB-D sensor [26]. RGB-D sensors not only provide images with a reasonably high resolution, but we can reconstruct the global information of plants to which images can be registered. The collection of global and local data is useful for us to extract phenotypic

characteristics further. The robots can move simultaneously by following the planned collision-free trajectories. Our system is governed by the deep-learning based NBV planner, which supervise the operation of the motion planning module and the trajectory execution module.

A. Hardware

Our system consists of three robotic arms (UR-5), where each arm is equipped with a depth camera (Intel RealSense SR-300) in its eye-in-hand configuration. UR5 controllers connect to an adaptive switchboard with 100 Mbps Ethernet interface, which ensures low latency. The SR-300 depth cameras connect to a USB 3.0 hub with an external power source. The USB hub keeps communications with a PC running our software. All robots move simultaneously in each round of data acquisition for efficiency.

B. Software

The core part of the software in our system is a deep-learning based NBV planner, which is responsible for evaluating and selecting the most promising viewpoint for each round of data acquisition. Viewpoints in Cartesian space are planned as trajectories that can be executed simultaneously by the system. The low-level software components used to control the operation of our system are based on the Robot Operating System (ROS). Integrated information of plant leaves can be progressively updated after a post-processing step, including noise filtering and registration. As the major technical contribution of this letter, the deep-learning based NBV planner consists of two parts. First, a deep network that can predict the 'full' shape of a scanned model (in the form of a point cloud Y_o) from the partially scanned point cloud X. The predicted shape Y_o provides additional clues to the second part of our planner to update the probability map for supervising the planning algorithm to score the candidate viewing points.

IV. Deep NBV Planner

The next-best viewpoint is defined as the one that can provide the maximum amount of information for the plant phenotyping. An NBV planner selects the next-best viewpoint v^* from a set of candidate viewpoints $\mathcal{V}$ by maximizing information gain. Previous NBV approaches usually cast a set of rays $\{r\}$ from each viewpoint v within $\mathcal{V}$, where the quantity and silhouette of $\{r\}$ depend on sensors. Each ray is intersecting with the surrounding environment to determine the information gain required for making NBV decision. The surrounding environment is often described as an enclosed bounding volume (e.g., box, sphere) around a point-of-interest (POI) in a workspace $\mathcal{W}$, which encodes the prior knowledge about the plant phenotyping task, called the *workspace prior* in our letter. The enclosed bounding volume $\mathcal{M}$ is usually represented by octomap [27], which is a widely-used grid structure in which each voxel $x \in \mathcal{M}$ has a probability to encode its status as *occupied*, *free* or *unknown*. Each ray r traverses all voxels and finally ends up by either reaching an occupied voxel or exceeding the maximal distance itself. After casting all rays from v and traversing $\mathcal{M}$, each ray r will give a set of intersected voxels as $\mathcal{X}_r \subset \mathcal{M}$. An entropy-based evaluation process is applied to $\{v\}$ through an objective function E_v. In particular, we use the *Occlusion Aware VI* model [28] and the definition of the information gain of a single ray E_r is:

$$E_r = \sum_{x \in \mathcal{X}} P_v(x) \cdot \big(-P(x) \ln P(x) - \bar{P}(x) \ln \bar{P}(x)\big) \quad (1)$$

where $P(x)$ is the estimated probability of a voxel to be occupied given all measurements and $P_v(x)$ is the pi-productions of $\bar{P}(\cdot)$ of all intersected voxels traversed before x and $\bar{P}(x) = 1 - P(x)$. By summing up the information gains of all relevant rays, we obtain the following entropy model:

$$E_v = \sum_r E_r \quad (2)$$

Among all the sampled viewpoints, the next-best-view v^* can be selected:

$$v^* = \arg\max E_v. \quad (3)$$

After obtaining the best viewpoint v^*, the robot positions the sensor at v^* and updates the occupancy map $\mathcal{M}$ using the captured information. Specifically, the probability value $P(x)$ of a voxel $x \in \mathcal{M}$ is updated by the log-odds rule [27]. The above steps are repeated until the termination criterion is met.

Traditional NBV planning methods generalize well in real-world scenarios, but they lack the task-specific information. For example, although plants grow naturally, high-level spatial structures, such as leaf size and distribution, should follow specific probabilistic rules. Such plant-specific knowledge should be combined with $\mathcal{M}$ in voxel-space to improve the efficiency of NBV planning. Our solution is to train a deep neural network to predict plant-specific information to help with NBV planning. The predicted result is also in the form of a set of voxels. Unlike the ray-casting strategy in traditional NBV planning, our approach casts rays from voxels to viewpoints to achieve efficient planning.

A. Deep Network for NBV

As mentioned, we design a deep network $\mathcal{N}(\cdot)$ to supplement task-specific information for NBV planning. In this section, we describe the architecture of our proposed network model and the way to incorporate the output of the model with the NBV planner's probabilistic map $\mathcal{M}$.

1) Network Architecture: Our network is based on the recently proposed *Point Completion Network* (PCN) [29] but with the capability of predicting the confidences of completed points. However, point cloud completion and confidence inference cannot be trained as a unified task because the value is measuring how close a predicted point to its ground-truth is variable. We separate this task into two different branches and design a new network inspired by multi-task learning (MTL) [30]. The first branch of our network is similar to the original PCN, which uses an encoder-decoder structure to complete a partial point cloud X, with no requirement for any structural assumptions or annotations. The predicted result of this branch is $Y_o \in \mathbb{R}^{m \times 3}$,

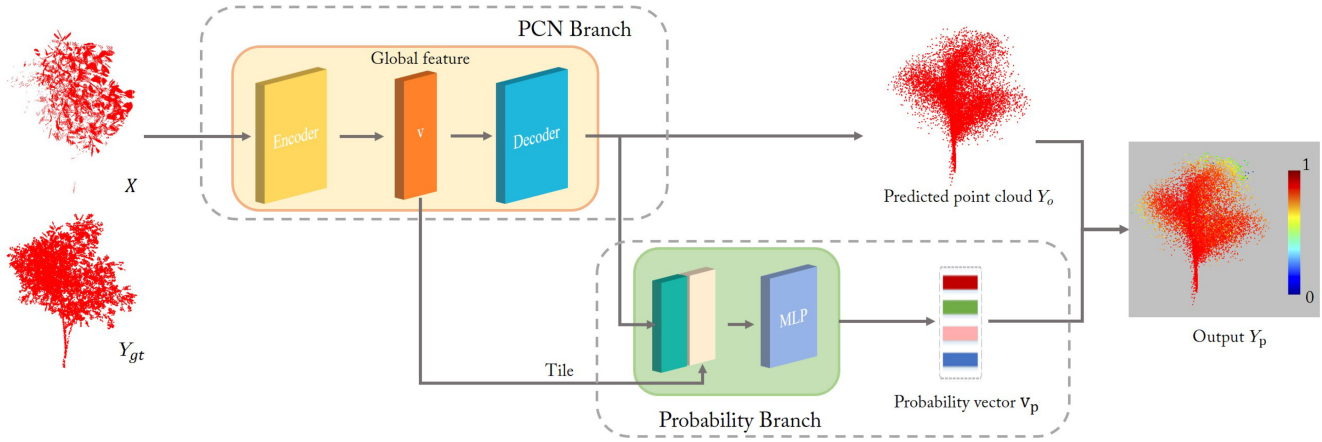


Fig. 2. The network we use for generating point clouds with confidences. An encoder-decoder network – PCN first takes a partial point cloud X obtained by the sensors as input. The encoder module is responsible for generating a global feature v for the decoder and our proposed *probability branch*. The decoded result is also concatenated with the global feature to learn the confidence of the output predicted point. Y_o is the point cloud produced by the part of original PCN, Y_p is Y_o combined with confidences produced by the probability branch and Y_{gt} is the ground truth point cloud.

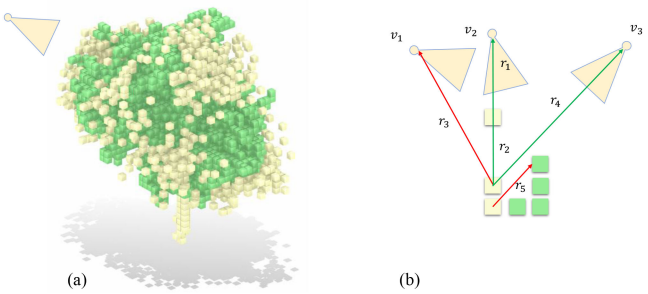


Fig. 3. An illustration of computing the next-best viewpoint using our proposed method. (a) The occupied voxels of $\mathcal{M}$ are shown in green, and the occupied voxels of $\bar{\mathcal{M}}$ is shown in yellow. (b) Our method casts rays from every voxel in $\bar{\mathcal{M}}$ to the candidate viewpoints, accumulating the information gain of all traversed unknown voxels using Eq. 1. The red rays either fall outside of FOV or be blocked by occupied voxels (shown in green) in $\mathcal{M}$.

where m is the number of points. The predicted point cloud Y_o can capture low-level geometric details and thus can encode the plant-specific knowledge in the voxels. The second branch of our network is the *probability branch* that outputs the confidence or probability values of each predicted point in Y_o. In particular, we first concatenate the global features $\mathrm{v} \in \mathbb{R}^k$ $(k = 1024)$ (i.e., the output of the encoder) in the first branch and the decoded point cloud Y_o. The stacked vectors are then sent through a multi-layer perceptron (MLP) that has three linear layers with the sigmoid activations. The MLP will generate a probability vector of $\mathrm{v_p}$ as the output of the second branch to describe the confidence of the prediction Y_o. The final output Y_p of the entire architecture is the concatenation of Y_o and $\mathrm{v_p}$: $Y_p = [Y_o, \mathrm{v_p}] \in \mathbb{R}^{\mathrm{m}\times 4}$. The entire pipeline for the network described above is shown in Fig. 2.

2) Loss Function: For the first branch, we use the same loss function L_o as proposed in the original PCN, which measures the distance between the predicted point cloud Y_o and the underlying ground truth Y_{gt}. For the second branch, the loss function L_p measures the difference between the confidence of the predicted point cloud provided by the second branch and a desired confidence about the point cloud, which is computed according to the nearest point-to-point distance between the predicted point cloud Y_o and the ground truth point cloud Y_{gt}. In particular, given a point $x \in Y_o$, the desired confidence is computed as $\mathrm{v_p^*(x)} = \min_{\mathrm{y}\in \mathrm{Y_{gt}}} \mathrm{e}^{-\lambda\|\mathrm{x}-\mathrm{y}\|_2}$. This formulation gives us real value in the range of $[0, 1]$, which can be regarded as a probability value as well. The confidence loss function L_p is then defined as:

$$L_p = \frac{1}{|Y_o|} \sum_{x\in Y_o} |\mathrm{v_p^*(x)} - \mathrm{v_p(x)}|^2, \tag{4}$$

where λ is a scaling parameter for fitting different scales of plants, and it is set as the diagonal length of the occupancy map $\mathcal{M}$.

3) Training Strategy: As we mentioned before, the ground-truth of a predicted point is variable during the training of the PCN branch. In the design of our MTL-based network, we adopt *hard parameter sharing strategy* [31] to train the probability branch. In hard parameter sharing, a portion of the parameters is shared among different tasks, while the other parameters are task-specific. In our case, the trained hidden layers of the PCN branch are sharing to the probability branch, whose layers are completely task-specific. Therefore, the two different tasks must be trained in a fixed order. More specifically, we have to train the PCN branch before the probability branch.

4) Oracle Generation: The output Y_p of the deep network is a point cloud with confidence values. It can be used to derive a distribution about the possible occupancy status of the voxels, which should be accessed by the sensor's rays. In this letter, we name this useful information for NBV planning as *oracle*. Voxels encoded with free status in $\mathcal{M}$ should have a low probability of being accumulated by a set of measurements of sensors. Our deep network may mispredict these voxels. Considering the map $\mathcal{M}$ that has been built, the conservative strategy of oracle generation should remove these voxels. As a result, it should provide instructions to the NBV planner where the possible voxels that need to be accessed are. Given the combination of scanned

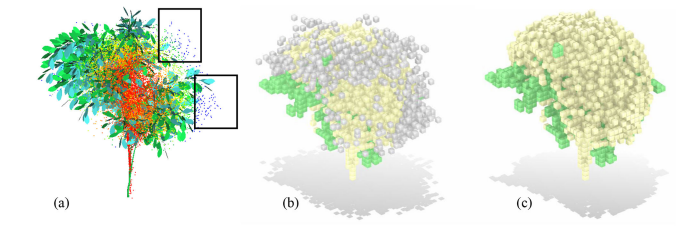


Fig. 4. A comparison of $\bar{\mathcal{M}}$ generated with/without the probability branch. Some predicted points are not as precise as expected, so there are incorrect points in black rectangles. (a) The predicted point cloud with probability (from low to high: blue - red) (b) $\bar{\mathcal{M}}$ generated by point cloud only (c) $\bar{\mathcal{M}}$ generated by using probability – incorrect predictions are shown in gray.

point cloud X, we first use the proposed network to generate a full point cloud with confidences as Y_p. To accommodate the input size of the network, we resize X to X' through adaptive downsampling to feed the network. We use Y_p to update $\mathcal{M}$ and get a new probabilistic map G_p using the log-odds rule [27]. Since there is no explicit correspondence between the input and output of the trained model, the overlap may exist so we must draw a clear line to make oracle more especially effective. To this end, we compute the difference between all occupied voxels in G_p and all occupied voxels in $\mathcal{M}$, and finally the oracle $\bar{\mathcal{M}}$ is built, which is defined as a 0-1 binary grid, a voxel in $\bar{\mathcal{M}}$ is either occupied or free. Adding this probability branch is to combine long-term measurements and network predictions – i.e., a voxel has been measured as empty for many times should not be regarded as occupied even if the network predicts it is. This branch not only provides the confidence distribution of a predicted point cloud but also seamlessly bridges the deep-network to NBV planners who use a log-odds rule as the updating strategy. Our probability branch provides the capability of weighting the predicted point cloud that we use to generate $\bar{\mathcal{M}}$ – see the comparison in Fig. 4.

B. Oracle-Informative Planning

The oracle generated by the above steps is used to guide NBV planning. Ideally, the optimal next-best viewpoint v^* can fully observe all voxels in $\bar{\mathcal{M}}$ because $\bar{\mathcal{M}}$ gives a sufficient estimation of the locations for next scanning. In other words, a 'good' viewpoint should cast more rays to cover voxels in $\bar{\mathcal{M}}$. However, we refer to this situation as 'optimal' because it is practically impossible due to realistic limitations, such as self-occlusions and the limited field-of-view (FOV) of the sensors. Instead of looking for an 'optimal' solution, our planning scheme chooses to find a practical solution that is close to the 'optimal' case.

1) Planning Algorithm: Our proposed planning algorithm tends to choose the viewpoint v which can be seen by more voxels in $\bar{\mathcal{M}}$ with '1' attribute. Traditional NBV approaches usually cast rays from all candidate viewpoints to $\mathcal{M}$. By contrast, our search algorithm traces the casting rays from voxels with '1' attribute in $\bar{\mathcal{M}}$. We connect each voxel in $\bar{\mathcal{M}}$ with each candidate viewpoint to constitute a set of back-tracing rays $\{r'\}$. Specifically, for a voxel x and a viewpoint v, a ray $r' := (x \rightarrow v)$ is feasible if it falls into the FOV region $FOV(v)$ of v, which is described by standard pinhole models. The information gain

Algorithm 1: Oracle-informative NBV planning

Input: A set of point clouds captured by the sensor $\{X_i\}(i = 1, ..., M)$; a set of candidate viewpoints $\{v_j\}(j = 1, ..., K)$.
Output: An entropy-minimized viewpoint v^*.

```
Update the global occupancy map M by newly captured point cloud X_M;
Combine input point clouds as X ← X_1 ∪ ··· ∪ X_N;
Downsample X to X' with the input size of N(·);
Predict the completion with probability Y_p ← N(X);
Build an oracle M̄;
forall x ∈ M̄ do
    forall v_j do
        r' := (x → v_j)
        if r' ⊂ FOV(v_j) then
            E_{v_j} := E_{v_j} + E_r (ref. Eq. 1 and Eq. 2)
        end
    end
end
return the viewpoint v* that gives maximized E_{v*};
```

of E_v contributed by x is computed in the domain of $\mathcal{M} \cap \bar{\mathcal{M}}$ using Eq. 2. We also provide a pseudo code in Algorithm 1.

2) Planning Space: The state-of-art NBV methods such as [22], [23] predefine a bounding geometry and assume that it is able to cover the target object completely. We also follow this assumption to define the search space and choose to sample candidate viewpoints on a sphere around POI with a fixed radius R. The feasibility of a view can be determined by the kinematic model of the robot, together with $\mathcal{W}$. The analysis of feasibility can eliminate unnecessary views in advance.

C. Termination Criterion

It is important to define a termination criterion in NBV planning because there is no explicit expression that can be used to terminate the search. Similar to [23], [24], we define a simple termination criterion to complete the NBV planning, as shown below.

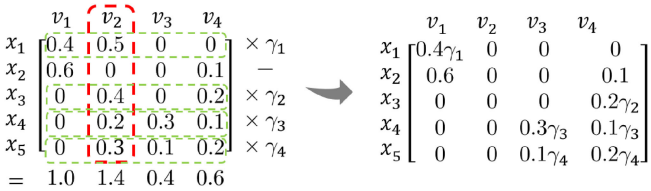
$$E_{v*} < q_{thres} \tag{5}$$

where q_{thres} is a user-defined threshold of the lowest information gain that can be accepted.

D. Extend to Multi-Robot Systems

Mobile robots with eye-in-hand sensors, such as the on used in [22], can use uncertainties to reconstruct complex objects flexibly. However, robotic arms with fixed bases are limited by flexibility (i.e., reachability). If people want to use robotic arms in plant phenotyping, a better solution would be adopting more robotic arms in different setups of bases. For instance, we use three robotic arms at the same around the target plant, as described in Sec. III. The problem of extending single robots to multi-robot systems contains a *set cover problem* [32], which is NP-hard [33]. Due to the difficulty, we decided to propose a practical solution to this problem. It is worth noting that the

most challenging part of this problem is to design an efficient algorithm to avoid overlapped viewpoints as the exact overlaps can only be identified by updating the occupancy map. To this end, we propose a heuristic algorithm as follows.



We design the algorithm based on the preference for paying less attention to a voxel of $\bar{\mathcal{M}}$ if it has been used to select other viewpoints. Let N be the number of robotic arms. We first generate a search space by sampling the candidate viewpoints for all N sub-robots (see Fig. 6 for an $N = 3$ example). Then we apply the proposed NBV planning algorithm on all candidate viewpoints to obtain a matrix $A \in \mathbb{R}^{p \times q}$ (see above). The rows of A are oracle voxels $\{x_i\}$, the columns of A are candidate viewpoints $\{v_i\}$, and $A(i, j)$ is the information gain contributed from voxel i to viewpoint j. The sum of each column of A can be used to select next-best viewpoints. We repeat the following steps until N next-best viewpoints are generated.

- Take the t_{th} viewpoint with the largest col-wise sum value;
- Set all columns whose corresponding viewpoints are belonging to the same robot of t_{th} viewpoint to zero;
- Multiply an adaptive decay $\gamma = \ln 2 - A(s, t)$ to every row s whose $A(s, t) \neq 0$ to decrease the risk of overlapping.

V. Experimental Results

We now present the experimental results of our system. Since the ray-castings are embarrassingly parallel, all the proposed algorithms are implemented on GPU architecture with the help of GPU-Voxels library [34], including the method we implemented for comparison. All tests are run on a PC with Intel(R) Xeon(R) Gold 6146 CPU @ 3.20 GHz, 128 GB RAM and NVIDIA Titan V. We test the deep-learning based NBV planner on simulations and provide a real-world demo with a multi-robot setup.

A. Dataset and Training

An obstacle to adopting deep networks in plant phenotyping is the lack of data. It is an onerous task to collect point clouds of plants as ground truth manually. In this letter, we use *ngPlant* [35] to synthesize data. *ngPlant* is a parametric synthesis tool for plants. We randomly assign parameters such as branching angles and offsets to synthesize different plants. Instead of using the original 2D texture to represent leaves, we apply Delaunay triangulation on the texture to convert them to meshes. The synthesized meshes are then converted to point clouds by uniformly sampling. We reserve 150 models for validation and 100 models for testing. The rest 610 models are used for training. The resolution of occupancy map $\mathcal{M}$ is 0.04 m. We render the plants at different viewpoints using a standard pinhole camera model (Intel(R) Realsense(TM) SR300 @ 640 × 480). The partial point clouds can be obtained by extracting values from the depth-buffer and applying intrinsic and extrinsic parameters. The network is trained by using the Adam optimizer. The first training pass for the PCN branch takes original training settings from [29] for 53 epochs. Then we train the probability branch using 10^{-4} learning rate, and it is converged after 263 epochs (around 15 hours).

TABLE I
Computational Statistic of Each Round

Method	Robots # (N)	Inference time	NBV time
Traditional NBV	1	\	4.030 sec.
	2	\	7.154 sec.
	3	\	11.382 sec.
Our method	1	0.333 sec.	3.077 sec.
	2	0.360 sec.	5.980 sec.
	3	0.401 sec.	10.140 sec.

B. Simulation Experiments

In our simulations, we assume robots can completely cover the search space (i.e., plants) though this is not achievable because of robots' reachability in reality. We compare our method with the traditional NBV planning approach using the same entropy model (i.e., *Occlusion Aware VI*). The evaluations are conducted on different choices of the number of robots N, from 1 to 3, for ten trails. To clarify the experimental results, we define a metric to calculate the precision (i.e., surface coverage rate) between the ground truth point cloud S_1 and the point cloud S_2 measured by sensors as follows.

$$P(S_1, S_2) = \frac{1}{|S_1|} \sum_{x \in S_1} \mathrm{U}\left(\min_{y \in S_2} ||x - y||_2 - \epsilon \right) \tag{6}$$

where $\mathrm{U}(\cdot)$ is the Heaviside step function, and ϵ is the distance threshold used to determine if a point in S_2 has already been captured. All simulations use $\epsilon = 5 \times 10^{-5}$.

Though our method and the traditional method using the same entropy model, the termination criteria are hard to be the same due to different evaluation processes. To fairly compare our method with the traditional method, we compute the average entropy value when both algorithms achieve 85% coverage rates on the same plants and use them as the termination criteria. A comparison of computational statistics between our proposed method and the traditional NBV approach (the basic pipeline in [22] with the *Occlusion Aware VI* entropy model) is shown in Table I, and our proposed method shows a promising speed, although it has an additional inference step taken by the deep neural network. Experimental results show that our method yields better results than the traditional NBV approach. Even if the precision of initialization step (randomly initialized in simulations) is lower than the traditional NBV method, the final result will be superior to the traditional NBV method, which proves the effectiveness and efficiency of our proposed method. An example of comparisons w.r.t. precision $P(S_1, S_2)$ is shown in Fig. 7.

The average accuracy of point cloud predicted by the network can be measured by Chamfer Distance (CD), and Earth Mover Distance (EMD) [29], and they are 0.02617 and 0.28637 respectively. We also provide the results measured by the same metrics

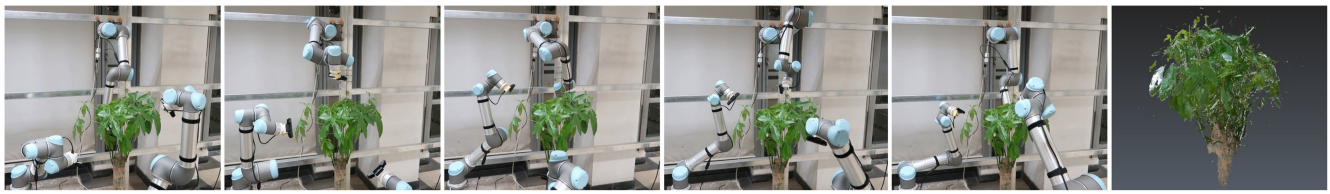


Fig. 5. The progressive views of plant phenotyping by our multi-robot. These pictures are taken when our multi-arm system is running the proposed deep NBV planner. The last column depicts the resultant point cloud of the plant. One can watch the accompanying video for a full illustration.

(CD and EMD) as a function of how much the progress of plant phenotyping in Fig. 8.

C. Real-World System Deployment

1) Motion Planning: The low-level component of our software adopts the *RRT-Connect* algorithm [36] to plan motions, which interchangeably builds two trees to connect the start and goal configurations. Considering the high-dimensional problem introduced by multi-robot, it is easy to plan viewpoints but fails to succeed in most cases. In our system, we ease the problem with the following strategies.

1) It is not necessary to sample all candidate viewpoints in situ, and we presample these viewpoints using an analytically inverse-kinematic solver. We add the maximal enclosed volume (the *workspace prior* used in NBV planning) to the environment to avoid collisions.
2) The initial configurations of our robots in each round of phenotyping are fixed. We compute the initial configurations by finding the minimal one whose corresponding configurations have minimal L_2 distances over all others.
3) We use the planner [36] to generate trajectories between the target and the initial configurations for all robotic arms simultaneously and remove the viewpoints whose corresponding configurations cannot be planned within $t_{max} = 40$ seconds.

By using these rules, the successful rate of motion planning can significantly increase from less than 10% (i.e., randomly generating initial viewpoints without using any rules mentioned above) to 94.2%.

We pre-sampled 100 viewpoints for each of the three robots. It takes around 24 hours to generate motions on four PCs. The motion planner takes 0.238 second for each trajectory on average. We build a database to store the planned trajectories, and they can be queried in real-time during phenotyping.

2) Camera Calibration: The static transformation between the mounted sensor and the end-effector should be well calibrated; otherwise, the point cloud cannot be deployed to the correct location. Moreover, the relative positions of robots should also be calibrated for collision avoidance. We use the classic hand/eye calibration method [37] to randomly acquire a set of poses for each robot to track an AR marker for calibration. The calibrated results are fine-tuning multiple times until a satisfactory result is obtained.

3) Results: The progressive views of our real-robot experiment are shown in Fig. 5 , the scanned result is also shown in the right-most column of the same figure. We also provide a video in the supplementary material to explain the proposed planner and demonstrate the physical experiments.

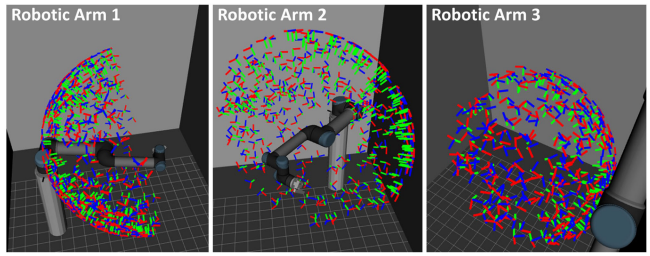


Fig. 6. In a multi-robot system with $N = 3$ sub-robots, we sample candidate viewpoints on a sphere using an analytically inverse-kinematic solver.

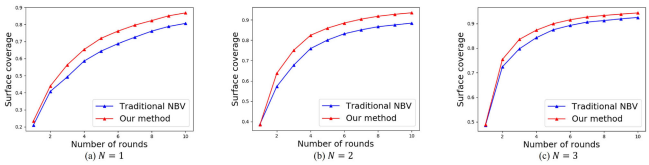


Fig. 7. Comparisons between our proposed NBV method and a traditional NBV method in phenotyping a test plant ($N = 1, 2, 3$). The results show that although the initial precision of our approach is lower than the conventional NBV, our proposed method rapidly outperforms traditional method and shows a larger AUC value. The results are averaged over ten trials each for making the comparison fair.

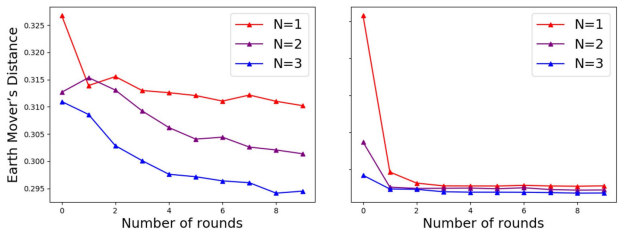


Fig. 8. The progressively measured Chamfer Distance (CD) and Earth Mover Distance (EMD) [29] on the generated results of our network-based planner.

VI. Conclusion and Future Work

In this letter, we present an automated robotic system for fast, precise, and noninvasive plant phenotyping. We propose a deep-learning based NBV planner to compute next-best viewpoints. The planner first uses a deep neural network to estimate a set of candidate voxels for next scanning, and then cast rays from these

voxels to determine the viewpoints to be positioned with sensors. Our proposed learning-based NBV planning method can be easily extended from a single-robot to a multi-robot system without sacrificing too much speed of planning. The results of our experimental tests are encouraging and prove that the system can advance the high-throughput phenotyping research.

Our system has a disadvantage that it heavily relies on the trained deep networks; if the networks cannot produce compelling predictions, the system may not be able to give next-best viewpoints. We plan to solve this problem by incorporating the deep NBV planner with other planners by scheduling them at different stages [38]. We plan to use this system to build a complete dataset in terms of plant phenotyping and conduct quantitative analysis.

REFERENCES

[1] W. Yang *et al.*, "Combining high-throughput phenotyping and genome-wide association studies to reveal natural genetic variation in rice," *Nature Commun.*, vol. 5, 2014, Art. no. 5087.

[2] R. T. Furbank and M. Tester, "Phenomics—Technologies to relieve the phenotyping bottleneck," *Trends Plant Sci.*, vol. 16, no. 12, pp. 635–644, 2011.

[3] F. Tardieu, L. Cabrera-Bosquet, T. Pridmore, and M. Bennett, "Plant phenomics, from sensors to knowledge," *Current Biol.*, vol. 27, no. 15, pp. R770–R783, 2017.

[4] F. Fiorani and U. Schurr, "Future scenarios for plant phenotyping," *Annu. Rev. Plant Biol.*, vol. 64, no. 1, pp. 267–291, 2013.

[5] D. Zermas, V. Morellas, D. Mulla, and N. Papanikolopoulos, "Extracting phenotypic characteristics of corn crops through the use of reconstructed 3D models," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2018, pp. 8247–8254.

[6] T. Duan, S. Chapman, E. Holland, G. Rebetzke, Y. Guo, and B. Zheng, "Dynamic quantification of canopy structure to characterize early plant vigour in wheat genotypes," *J. Exp. Botany*, vol. 67, no. 15, pp. 4523–4534, 2016.

[7] A. Junker *et al.*, "Optimizing experimental procedures for quantitative evaluation of crop plant performance in high throughput phenotyping systems," *Frontiers Plant Sci.*, vol. 5, pp. 770-1–770-12, 2015.

[8] C. Reuzeau *et al.*, "Traitmill: A functional genomics platform for the phenotypic analysis of cereals," *Plant Genetic Resour.*, vol. 4, no. 1, pp. 20–24, 2006.

[9] LemnaTec GmbH, 2019. [Online]. Available: https://www.lemnatec.org/

[10] A. Ruckelshausen *et al.*, "BoniRob: An autonomous field robot platform for individual plant phenotyping," *Precis. Agriculture*, vol. 9, no. 841, pp. 841–847, 2009.

[11] G. Alenyà, B. Dellen, S. Foix, and C. Torras, "Robotized plant probing: Leaf segmentation utilizing time-of-flight data," *IEEE Robot. Autom. Mag.*, vol. 20, no. 3, pp. 50–59, Sep. 2013.

[12] S. Foix, G. Alenyà, and C. Torras, "3D sensor planning framework for leaf probing," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2015, pp. 6501–6506.

[13] T. Mueller-Sim, M. Jenkins, J. Abel, and G. Kantor, "The Robotanist: A ground-based agricultural robot for high-throughput crop phenotyping," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2017, pp. 3634–3639.

[14] I. Sa *et al.*, "Peduncle detection of sweet pepper for autonomous crop harvesting-combined color and 3-D information," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 765–772, Apr. 2017.

[15] T. Gao *et al.*, "A novel multirobot system for plant phenotyping," *Robotics*, vol. 7, no. 4, 2018, Art. no. 61.

[16] A. Kamilaris and F. X. Prenafeta-Boldú, "Deep learning in agriculture: A survey," *Comput. Electron. Agriculture*, vol. 147, pp. 70–90, 2018.

[17] C. McCool, T. Perez, and B. Upcroft, "Mixtures of lightweight deep convolutional neural networks: Applied to agricultural robotics," *IEEE Robot. Autom. Lett.*, vol. 2, no. 3, pp. 1344–1351, Jul. 2017.

[18] A. Milioto, P. Lottes, and C. Stachniss, "Real-time semantic segmentation of crop and weed for precision agriculture robots leveraging background knowledge in CNNs," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2018, pp. 2229–2235.

[19] T. Parhar, H. Baweja, M. Jenkins, and G. Kantor, "A deep learning-based stalk grasping pipeline," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2018, pp. 1–5.

[20] H. H. González-Baños and J.-C. Latombe, "Navigation strategies for exploring indoor environments," *Int. J. Robot. Res.*, vol. 21, no. 10/11, pp. 829–848, 2002.

[21] S. Kriegel, C. Rink, T. Bodenmüller, A. Narr, M. Suppa, and G. Hirzinger, "Next-best-scan planning for autonomous 3D modeling," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2012, pp. 2850–2856.

[22] J. I. Vasquez-Gomez, L. E. Sucar, and R. Murrieta-Cid, "View/state planning for three-dimensional object reconstruction under uncertainty," *Auton. Robots*, vol. 41, no. 1, pp. 89–109, Jan. 2017.

[23] S. Isler, R. Sabzevari, J. Delmerico, and D. Scaramuzza, "An information gain formulation for active volumetric 3D reconstruction," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2016, pp. 3477–3484.

[24] J. Daudelin and M. Campbell, "An adaptable, probabilistic, next-best view algorithm for reconstruction of unknown 3-D objects," *IEEE Robot. Autom. Lett.*, vol. 2, no. 3, pp. 1540–1547, Jul. 2017.

[25] R. Monica and J. Aleotti, "Surfel-based next best view planning," *IEEE Robot. Autom. Lett.*, vol. 3, no. 4, pp. 3324–3331, Oct. 2018.

[26] C. Wu, R. Zeng, and Y.-J. Liu, "A multi-robot system for high-throughput plant phenotyping," in *Cognitive Systems and Signal Processing*, F. Sun, H. Liu, and D. Hu, Eds. Singapore: Springer, 2019, pp. 524–533.

[27] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Auton. Robots*, vol. 34, no. 3, pp. 189–206, 2013.

[28] J. Delmerico, S. Isler, R. Sabzevari, and D. Scaramuzza, "A comparison of volumetric information gain metrics for active 3D object reconstruction," *Auton. Robots*, vol. 42, no. 2, pp. 197–208, Feb. 2018.

[29] W. Yuan, T. Khot, D. Held, C. Mertz, and M. Hebert, "PCN: Point completion network," in *Proc. Int. Conf. 3D Vision*, Sep. 2018, pp. 728–737.

[30] S. Ruder, "An overview of multi-task learning in deep neural networks," 2017, arXiv:1706.05098.

[31] R. Caruana, "Multitask learning: A knowledge-based source of inductive bias," in *Proc. Int. Conf. Mach. Learn.*, 1993, pp. 41–48.

[32] C. Dornhege, A. Kleiner, A. Hertle, and A. Kolling, "Multirobot coverage search in three dimensions," *J. Field Robot.*, vol. 33, no. 4, pp. 537–558, 2015.

[33] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*. New York, NY, USA: Springer, 1972, pp. 85–103.

[34] A. Hermann, F. Drews, J. Bauer, S. Klemm, A. Roennau, and R. Dillmann, "Unified GPU voxel collision detection for mobile manipulation planning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2014, pp. 4154–4160.

[35] S. Prokhorchuk, "ngPlant—Open source plant modeling," 2016. [Online]. Available: http://ngplant.org/

[36] J. J. Kuffner and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2000, vol. 2, pp. 995–1001.

[37] R. Y. Tsai and R. K. Lenz, "A new technique for fully autonomous and efficient 3D robotics hand/eye calibration," *IEEE Trans. Robot. Autom.*, vol. 5, no. 3, pp. 345–358, Jun. 1989.

[38] J. E. Banta, L. R. Wong, C. Dumont, and M. A. Abidi, "A next-best-view system for autonomous 3-D object reconstruction," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 30, no. 5, pp. 589–598, Sep. 2000.