THE PLANT DOCTOR: A TRICORDER FOR THE GARDEN
San Francisco State University (SFSU)

Team Name: The Plant Doctors
Team Type: University Team
Team Members: Madu Nzerem, Faye Orcales, Berenice Chavez, Maxamillian Schneider, Johana Aleman, Amisha Dhawan, Patra Holmes, Anneke Moeller, Daisy Estrada Reynoso, Lorena Benitez Rivera
Team Leader: Thomas Zimmerman, IBM Research-Almaden
Team Region: North America
Submission Category: Agriculture

Problem Statement

Agriculture is an essential component of California's economy. Six counties (Fresno, Tulare, Monterey, Kern, Merced, and San Joaquin) account for about half of California's total value of agricultural production. In addition to the economic important of commercial agriculture, community gardens popping up around big cities like Los Angeles, San Jose and San Francisco provide city dwellers the opportunity to have fresh produce, get exercise, learn about, and enjoy the beauty of nature. The urban gardener and production farmer share common problems; what and when to plant, at what density, how much water and fertilizer to apply, when to harvest, and how to manage pests, diseases, and wildlife. While the commercial farmer is typically limited to a few cash crops, the urban farmer has many more choices for plants (e.g., vegetables, herbs, tubers, flowers), techniques (e.g., containers, raised beds, vertical gardens, hydroponics, aquaponics) and locations (e.g., backyard, terrace, roof, community). The multitude of choices for urban growers combined with lack of experience can make for a frustrating experience.

Solution Description

Our solution is to design a device that can assist urban gardens in analyzing, selecting, organizing, and maintaining healthy and productive gardens. The device provides the gardener **species identification,** to know what plants are in their garden; **optimal water**, **fertilizer, and nutrition**, to neither starve plants or waste and pollute resources; and **companion plants and placement,** so plants benefit each other by attracting pollinators, repelling pests, providing shade and beneficial nutrients. The device is based on the multi-camera OAK-D device, with modified cameras to extend the spectral response.

We call the device the "The Plant Doctor: A Tricorder for the Garden", named in honor of the science fiction prop designed for *Star Trek* that performed the functions of sensing, recording, and computing. The device is a collection of software modules that performs plant metrics and analysis through a common user interface. This allows modules to be developed simultaneously by the team and enable new modules to be added in the future. To meet the three-month development schedule, each module will be delivered as minimum viable product (MVP), demonstrating the operation of each module. The software we developed is open source, so individual models can be individually improved and extended. The modules and video links are as follows:

- **Enhanced Spectral Imaging** Predict plant health by viewing leaves at multiple wavelengths. https://www.linkedin.com/feed/update/urn:li:activity:6801034906647560192/
- **Plant Identification** Identify plants from their leaves. https://www.powtoon.com/s/cuOGVSqkpYD/1/m
- **Neighbor Analysis** Identify plants and relative location in a plot to predict and advise on companion planting and optimal spacing. https://www.powtoon.com/c/eKaZB3kkxE5/1/m
- **Environmental Science** Analyze soil chemistry measuring nitrogen, phosphorus, potassium, and pH. https://www.powtoon.com/c/eKaZB3kkxE5/1/m

Results Summary

The **Enhanced Spectral Imaging** team compared an iPhone to a modified OAK-D unit, to predict the health of a basil plant that was intentionally let to die from lack of watering over a two-week monitoring period. One of the OAK-D's monochrome cameras was replaced with a monochromatic camera with the IR blocking filter replaced with a visible light-blocking filter, allowing the image sensor to capture just the near IR spectra. Plant health was predicted by calculating a Normalized Difference Vegetation Index (NDVI) using the relative magnitude of the red and green image channels for the iPhone, and the red channel (from the color camera) and infrared image sensor (from the replaced monochromatic camera) for the modified OAK-D. All cameras were tested with their default settings. The iPhone produced a smooth monotonic NDVI curve showing a clear correlation with decaying health due to dehydration while the OAK-D produced a decaying NDVI curve, albeit with significant noise. While the iPhone NDVI calculations were performed on one RGB camera, the OAK-D calculations were performed on two cameras (color and modified monochromatic). We hypothesize the noise observed in the OAK-D NDVI calculations may be due to the two OAK-D cameras having different default settings, image processing dynamics (e.g., white balance and automatic exposure), and images were extracted from video recordings.

The **Plant Identification** team trained and tested 10 classes of plant leaves from a large public leaf image dataset (Kaggle competition) on a VGG-16 model with near perfect results. Then they trained a MobileNetv2-SSD on six classes from the public leaf image dataset and tested it on live leaf samples from a home garden captured with the 4k color OAK-D camera. The classification was perfect except for a major exception in one class where all bell pepper leaves were consistently misidentified as orange leaves. We hypothesize this was caused by their similar silhouette and may be mitigated by giving more weight to the color of the leaf.

The **Neighbor Analysis** team used the OAK-D and Luxonis's DepthAI application to record video, identify and track plants in 3D. They used this data to calculate plant area as an indicator of plant health, productivity and inter-plant spacing.

The **Environmental Science** team used a commercial soil chemical test kit (Rapitest) to measure soil concentrations of nitrogen, phosphorus, potassium, and pH. The chemical indicators were imaged with the OAK-D color camera. Image analysis was performed in the HSV (hue, saturation, value) color space as the chemical indicators change saturation and the pH indicator changes hue. The tests were performed under a variety of lighting conditions. Using the five color references from each testing kit containers for calibrating each measurement, the team was able to predict chemical concentration and pH with a resolution that exceeded the five reference levels.

Project Video Links

- **Enhanced Spectral Imaging** www.linkedin.com/feed/update/urn:li:activity:6801034906647560192/
- **Plant Identification** www.powtoon.com/s/cuOGVSqkpYD/1/m
- **Neighbor Analysis** www.powtoon.com/c/eKaZB3kkxE5/1/m
- **Environmental Science** www.powtoon.com/c/eKaZB3kkxE5/1/m

Final Report PowerPoint Slide Deck

- https://github.com/CCCofficial/PlantDoctor/blob/main/PlantDoctorFinalReport.pptx

Code Repository

- https://github.com/CCCofficial/PlantDoctor

Team Datasets

- https://ibm.box.com/s/kpf3ggy9ayfp78rjbb8m88gkj843wgyi

Team Paper Index

- **Enhanced Spectral Imaging........................**Page 3
- **Plant Identification......................................**Page 6
- **Neighbor Analysis...**Page 11
- **Environmental Science.................................**Page 15

Enhanced Spectral Imaging
Madu Nzerem, Faye Orcales

Abstract

When planting new plants throughout the year, urban farmers face common problems like determining what they can plant, when, and how to take care of that plant. To appease these problems, SFSU students and instructors under the PINC Program collaborated on a project called the Plant Doctor. This project was chosen to compete in OpenCV's AI Competition, and its overall goal was to provide urban farmers with the tools to maintain the health of plants within their garden. This paper details the work made under the Plant Doctor subgroup, the Enhanced Spectral Imaging group. This subgroup's main goal was to use enhanced spectral imaging to determine a plant's overall health. Smartphone cameras and a modified version of OpenCV's OAK-D device were used to gather the image data needed to approximate a plant's health. Images were collected from basil plants deprived of water over the course of two weeks, under indoor and outdoor lighting conditions. To track the plant's overall health over the course of two weeks without water, we calculated its Normalized Difference Vegetation Index (NDVI) value, based on the ratio of near infrared to red light reflecting from plant leaves.

Introduction

Many healthy plants absorb red light to make food while reflecting green and infrared light [2] due to their chlorophyll molecules that help them photosynthesize [1]. In the 1970s scientists began to use this information to monitor vegetation health. The monitoring of vegetation health was quantified using the Normalized Difference Vegetation Index (NDVI) [3] [4], calculated as follows.

$$\text{NDVI} = (NIR - Red) / (NIR + Red)$$

Dying plants tend to reflect less infrared light, reducing the NDVI value.

Methods

Two devices were used to capture images of plants: an iPhone and a modified OAK-D device. To calculate NDVI, the infrared light reflected from plant leaves is compared to the red light reflected. Since the iPhone only detects red, green and blue wavelengths, we also evaluated calculating NDVI with just green and red light. The standard OAK-D device contains three cameras, one 4K RGB camera and two monochromatic cameras. While the image sensor of the monochromatic cameras is sensitive to IR, unfortunately they contain an IR blocking filter. We modified one of the OAK-D cameras by replacing one of the monochromatic cameras with a monochromatic camera with the IR blocking filter removed (Arducam 1MP OV9282 drop-in replacement). To limit the camera's sensitivity to just IR, we added an IR800nm filter (Figure 1).

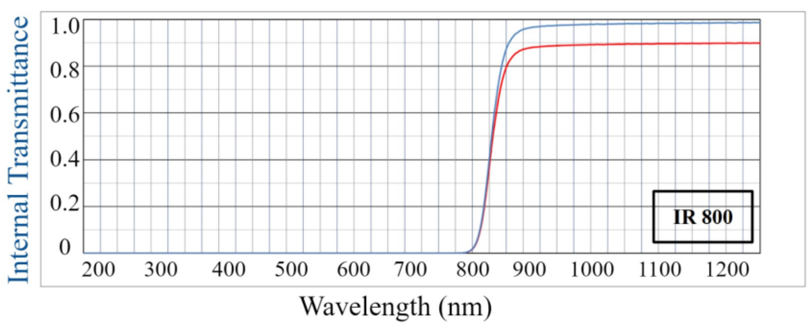


Figure 1. Spectral transmittance curves of the IR800 optical filter used on the modified OAK-D devices to limit detection to near infrared.

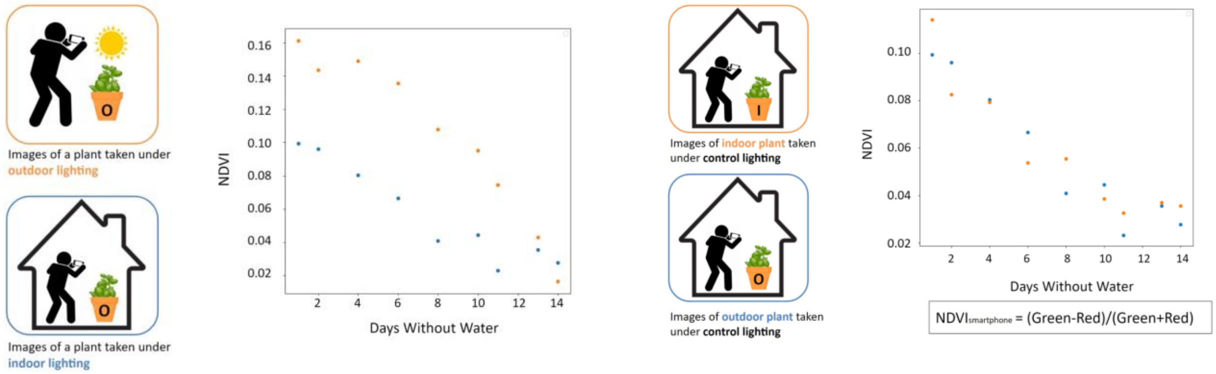


Figure 2. (Left) Both indoor and outdoor grown plants show similar change in NDVI over time. Schematic on the left shows which images were used. Graph on the right shows the change in NDVI over time in an indoor and outdoor plant. **(Right)** Images taken under outdoor lighting were shown to have higher NDVIs. Schematic on the left shows which images were used. Graph on the right shows the change in NDVI over days without water, comparing indoor lighting vs outdoor lighting. Dependent parameter was the NDVI. Independent parameters were the days dehydrated and the lighting conditions the plant was taken in (outdoor or indoor lighting).

Two trials were conducted for the iPhone and OAK-D cameras. One trial used one basil plant grown outdoors with images taken under indoor and outdoor lighting conditions. The second trial used one basil plant grown indoors with images taken under indoor lighting conditions. For both trials over a two-week period, daily images were taken of the whole plant, the leaves, and a color reference chart. During each trial the plants were not watered, causing them to dehydrate, wither, and die. When collecting images with the OAK-D device, each device was used for one conducted trial. The trial using the modified OAK-D took images of a plant grown in indoor lighting conditions while the trial using the standard OAK-D took images of a plant grown in outdoor lighting conditions. While individual images were taken with the iPhone, images were extracted from videos recorded with the OAK-D, as we did not have example code on how to capture images with the OAK-D device.

Results

The iPhone camera provides RGB channels, so we calculated NDVI from the average red and green channel values from a plant region-of-interest for each day, centering on greatest leaf density.

$$\text{NDVIsmartphone} = (Green - Red) \,/\, (Green + Red)$$

Images of plant leaves were captured of an un-watered basil plant each day for two weeks with the iPhone and modified OAK-D cameras. To see how independent NDVI values are to lighting conditions, we captured images of a plant grown outdoors under two lighting conditions, outdoor with uncontrolled control lighting and indoors with controlled lighting (Figure 2, left). To see how independent NDVI values are of growing light conditions, we captured images of plants grown outdoors and indoors under controlled indoor lighting (Figure 2, right).

To compare NDVI measurements made with the single iPhone RBG camera with NDVI measurement made with the RGB and modified infrared cameras of the OAK-D Lastly, we collected images of an indoor plant captured with controlled indoor lighting (Figure 4). Results showed that the OAK-D had higher NDVI values than the NDVIs from the smartphone images. Both devices showed a decrease in NDVI, however the OAK-D NDVI decrease was less consistent than the NDVI decrease seen from the iPhone data (Figure 3).

Discussion

We have demonstrated that plant health can be quantified by calculating the NDVI value of plant leaves. The iPhone correctly predicted the decreasing health of a water-starved plant for every daily sample over two weeks using the red and green channels of images captured with default settings. Using two cameras on the OAK-D to get the red and infrared channels to correctly calculate the NDVI with red and infrared channels was a difficult modification, requiring the purchase of a new camera without the IR block filter, opening the OAK-D case, and swapping the cameras. This also added the complexity of two camera with different lenses, image sensors, and possibly image processing software.

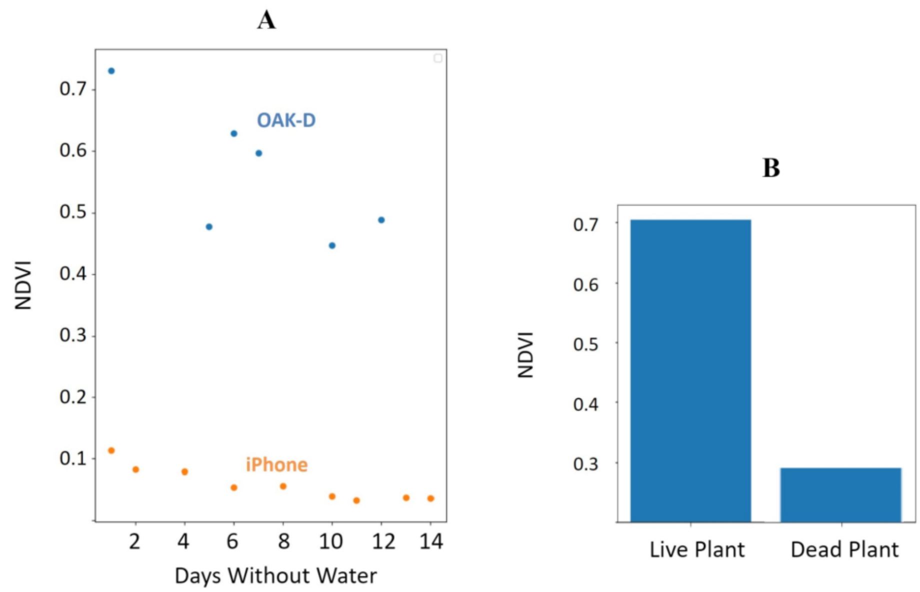


Figure 3. NDVI values calculated from an iPhone RGB camera and dual OAK-D cameras. While both devices show decreasing NDVI values as the plant dies, the iPhone NDVI values are more consistent than the OAK-D produced NDVI values. (**A**) Graph showing NDVI over time of OAK-D images vs iPhone images. To minimize the number of variables, images used were of indoor grown plants taken with indoor lighting. (**B**) Bar graph showing significant decrease in NDVI from live and dead plant images collected using the OAK-D.

This was also complicated by the fact that we had to record and extract images from video, adding another signal processing path to the acquisition. These and other unknown variable may account for the lower sensitivity and large variations in the NDVI values produced by the modified OAK-D device.

Supplemental Information

The images, video data and Python code used for research is available here:
https://sfsu.box.com/s/ftvm3fqmhicnz033t3bvy5tipmumpq4h

References

[1] Wollaeger, H. & Runkle, E., 2021. “Green light: Is it important for plant growth?”, *MSU Extension*, www.canr.msu.edu/news/green_light_is_it_important_for_plant_growth.
[2] "California Agricultural Production Statistics", www.cdfa.ca.gov/Statistics/ (2019)
[3] Weier, J. & Herring, D., 2000, “Measuring Vegetation (NDVI & EVI)”, *NASA Earth Observatory*, earthobservatory.nasa.gov/features/MeasuringVegetation/measuring_vegetation_1.php.
[4] "NDVI from the PLOTS near infrared camera", Public Lab, #1781, revision #35, publiclab.org/wiki/ndvi-plots-ir-kit (2016

Plant Identification

Johana Aleman, Amisha Dhawan, Patra Holmes

Abstract

The development of a tricorder for the garden can automate plant identification and volume estimation for both urban and commercial farmers. This tricorder can help promote polyculture by making it easier to identify a variety of plants and knowing which ones can be planted with specific companion plants. Since we are a part of the OpenCV competition, we implement transfer learning techniques to teach MobileNetV2-SSD how to identify eight plant species based on leaf structure. This model is fed into the OAK-D camera to allow users to scan their garden and automatically inform them the Plant ID and percentage of confidence when shown a single leaf of a plant. Here, we show our preliminary results of our current model as our project transitions into summer.

Introduction

Our module of The Plant Doctor aims to use the OAK-D to automate plant identification and volume estimation. Agriculture is a large part of California's economy, bringing in $14B from almond, grape, and strawberry sales alone in 2019 [1]. There are also many more community gardens popping up in urban areas. Companion planting, also known as polyculture, is the practice of placing different types of plants close to one another so that they can benefit from each other's properties, such as different resistance to pests and disease or varying nutrient consumption. Both production and urban farmers share the common task of determining which species would be beneficial to one another to make the best use of their space. Automating this process would promote sustainable farming by requiring less use of pesticides and fertilizer, which would also save the farmer time by minimizing their application - a situation benefiting both the environment and the user. The identification performed by this module would be used in the Neighbor Analysis portion of the project, which will focus on creating a 3D map of the garden, identifying existing plants, and suggesting neighboring plants for companion planting.

The OAK-D is a device with three cameras and a Myriad X processing chip which has AI capabilities. The camera in the center is a 4K color camera and the two stereo cameras on either side, giving the device depth perception, are monochromatic. All three of these cameras can simultaneously send video to the Myriad X chip, which can instantaneously process the video input using AI and output a video stream with information, such as what objects appear in the video and their distance from the camera. Documentation on the device can be found at: https://docs.luxonis.com/en/latest/pages/products/bw1098obc/. To accomplish the identification needed for the project, the module will utilize the AI capabilities of the OAK-D by implementing transfer learning.

Transfer learning is a method where a pre-trained Convolutional Neural Network is altered to classify the user's objects of interest. Convolutional Neural Networks (CNNs) are the basis of the computer vision needed for AI and classify objects based on their features, such as area and shape. CNNs are named as such since they are modeled after the brain: They are composed of nodes, weights, and biases, making a complex network through which the model can make "decisions" based on the features of the object presented and classify it. There are many different models publicly available to users that have already been trained to classify thousands of images over many hours. What these models have been trained to classify is irrelevant, and it is only important that they have learned the basics of how to perform classification by extracting features. Transfer learning takes advantage of a pre-trained CNN by transferring its ability to classify over to any object(s) of interest that the user wants by training it further with their own image datasets. Training using transfer learning can be done much faster, as only tens or hundreds of images are needed to attain a high accuracy of classification.

There are two types of CNNs: Standard and Single-Shot Detectors (SSDs). Standard CNNs are very large, where each image gets passed through loops and other complex algorithms for classification to occur. Since they are built this way, they require a lot of computing power and thus can normally only be used on computers. They are usually used to classify objects in large image datasets, for example the classification of cells as normal or mutant in images from a microscope. These CNNs can be so accurate that they can correctly classify images that would otherwise be misclassified by a human. SSDs on the other hand are on average 30 times smaller and 10 times faster than their larger counterparts. They send the input image in once and send it straight through the classifier, bypassing the loops and other complexities needed for classification by a large

Figure 1. Bounding box annotated image data of a bell pepper leaf (left) and a grape leaf (right).

CNN. SSDs are meant to perform object detection on small devices that lack the processing capabilities of a computer, such as dash cams, and can differentiate and classify multiple objects in the same image, placing bounding boxes around each object being detected, while simultaneously streaming video. For plant identification, transfer learning for both types of models was accomplished using images of leaves of various plant species.

Methods

To begin, we had to build a dataset to perform image classification. We first attempted to build our own dataset by taking photos of whole plants in gardens accessible to us, as well as finding images of whole plants on the internet. These images were collected at variable distances from the plant and contained different structures of the plant, such as flowers and fruit, as well as leaves. Since the images were so different, the first model we trained, a VGG-16 model [2], was unable to correctly classify them. After realizing that our images were too varied within each class for the classification to be accomplished, we decided to work with images of leaves only, since leaf structure remains consistent throughout a plant's life cycle, and other similar plant identification projects had used this approach. A large dataset from Kaggle was found [3] consisting of leaf images and from there we chose eight plant classes: bell pepper, apple, tomato, grape, strawberry, peach, raspberry, and orange. In Figure 1, we have shown some of our training images, which have a monochromatic background so that the structure and color of the leaf can be easily distinguished as our object of interest. This segmentation technique helps the model focus on only the principal features. We decided to include strawberry and grape since California is a large producer of these crops. We then trained another VGG-16 model using 70 images per class for training, 10 per class for validation, and 20 per class for testing, with a learning rate of 0.0001 over 20 epochs. During the training a loss graph was made (Figure 2, right) to monitor the classification of validation data. While optimizing the model for our eight classes, we discovered that the standard VGG-16 model we had been training could not be easily deployed on the OAK-D and decided to begin training a MobileNetv2-SSD instead, which is meant to run inference on a small device for object detection.

For the training of the MobileNetv2-SSD model we used the same Kaggle dataset but chose 300 images per class and used a Google Co-laboratory notebook (colab.research.google.com) provided by depthAI and RoboFlow [4]. We also had to further curate the data by annotating the images with bounding boxes using SuperAnnotate [5], as the model required the images to be labeled in this manner for training. After annotating, we had to upload the annotated data to RoboFlow [6] to be downloaded in the TFRecord format, as SuperAnnotate did not have the download available in the proper format for training with the depthAI/Roboflow code. During training of the model over 30000 steps, we learned that the metric for evaluating the accuracy of an SSD model is very different from that of the larger VGG-16 model that we were familiar with. The metric for the SSD model is Intersection Over Union (IoU). The IoU calculates the area of overlap over the area of union of the bounding boxes (Figure 2, left). Since this metric was focusing on the bounding box position rather than the class of the object, we decided to evaluate the model by testing it using the camera.

Before it could be used on the OAK-D, the model first had to be converted into a format usable by the device. This required the model to be converted from the ".pb" and ".config" file format into an Intermediate Representation consisting of a ".xml"

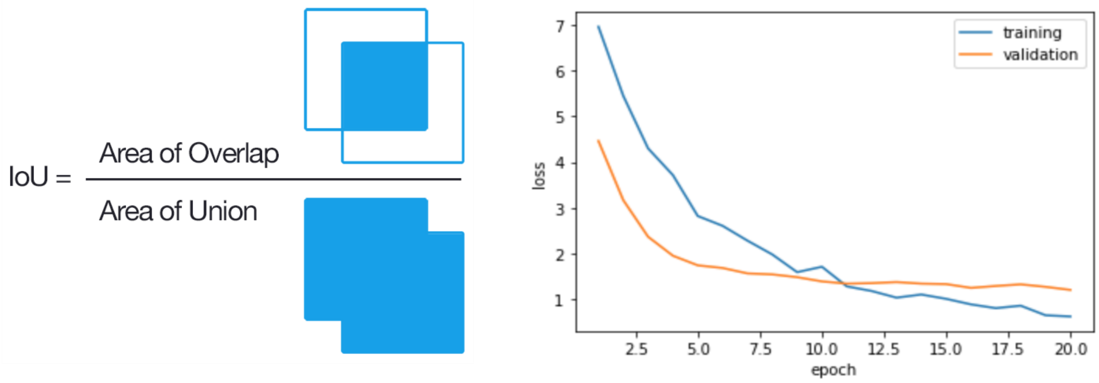


Figure 2. (Left) Image depicting how IoU values are calculated as a measure of accuracy for object detection models. **(Right)** Loss graph from VGG-16 model depicting loss of training and validation images over 20 epochs.

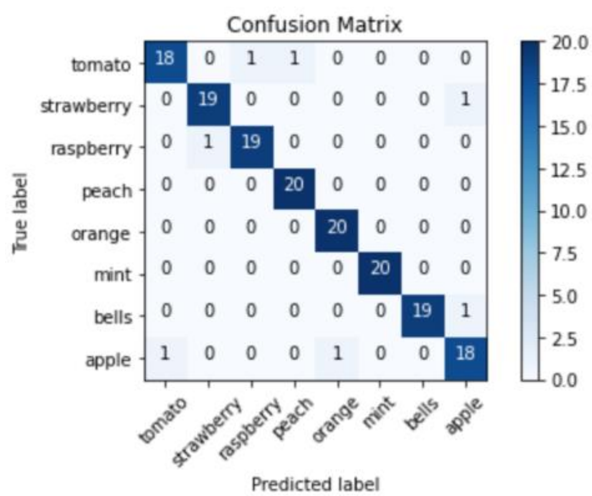


Figure 3. Confusion matrix for VGG-16 model showing high accuracy.

and a ".bin" file using openVINO's DL Workbench on Intel's DevCloud [7]. These then needed further conversion using openVINO in the terminal of a local drive into a ".blob" file. This final file could then be passed to the OAK-D in a Python script and we could see how well the model was performing by testing it with live plants. The OAK-D outputs a bounding box along with the class name and confidence score of the object being classified and using these two outputs we could estimate how well our model was performing. Each leave is picks off the plant and imaged individually for classification.

Results

During the model's training, validation images are shown to it to keep track of how the model is performing and to detect if the model is becoming overtrained on the training data and unable to correctly predict classes on unseen images ("overfitting of the model"). Our model seemed to be overfitting after 11 epochs, where the training line falls below the validation line (Figures 2, right). Ideally, the validation and training line should be closest to 0 and the training line should not fall below the validation line. After the model has been trained, we can evaluate how accurate it is by showing it a new set of test images that it has not seen before. This is visualized with a confusion matrix in Figure 3, with the true class shown on the y axis, and the predicted class on the x axis. For example, 18 out of 20 images were classified correctly as tomato, and 2 were misclassified: 1 as raspberry, and another as peach. A good confusion matrix should show a clear diagonal, meaning that most of the test images were correctly classified. This aids in understanding how the model are classifying images and identifying where it is having trouble. The model's overall accuracy score was about 96%.

```
Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.866
Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 1.000
Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.981
Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = -1.000
Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.866
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.889
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.889
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.889
Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = -1.000
Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.889
INFO:tensorflow:Finished evaluation at 2021-05-08-23:59:11
```

```
Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.153
Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.309
Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.146
Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.048
Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.415
Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = -1.000
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.042
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.202
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.217
Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.080
Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.516
Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = -1.000
```

Figure 4. (Left) IoU values from the first training of the MobileNetv2-SSD, without annotating images with bounding boxes. **(Right)** IoU values from the second training of the MobileNetv2-SSD, after annotating training images with bounding boxes.

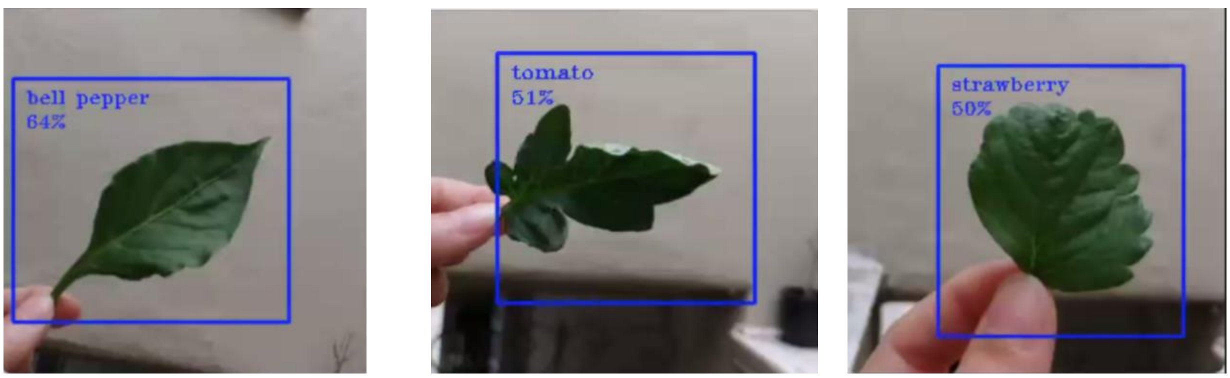


Figure 5. The OAK-D running our trained MobileNetv2-SSD and correctly classifying plant leaves; bell pepper (top left), tomato (top right) and strawberry (bottom left).

This confusion matrix came from the VGG16 model trained on leaf images. We had great accuracy with this model, but we later learned that it could not be converted for use on the OAK-D since it was so large. With this additional challenge of moving from a computer to a remote device, we then had to train one of the smaller SSD models. The smaller models, however, have a more complex way of assessing the accuracy, and it is easier to just test it in the field with the device.

The industry best-practice is to include a minimum IoU requirement for their human annotation tasks, to ensure that the annotations that are delivered have an IoU >= X (where X = 0.95 is typical) with respect to the “perfect” annotation of that object. The IoU values are ranged between 0.5 to 0.95, therefore if a value is closer to 0.95, that means that the predicted box has an area similar to the annotated box.

To test the models in the field, we captured images from six plants in a backyard garden. For our first trial, we gave the model images that had annotations for a few images, while the annotations for other species were missing. We can observe in Figure 4 (left), that the IoU values for the unannotated images were very small, i.e., the model did not train well, and the prediction values are below 0.5.

After observing these results, we further went on to annotate our whole image dataset and got the evaluation results as seen in Figure 4 (right). It is observed that the IoU values are higher i.e., greater than 0.5 and are closer to 0.95, which states that the model has good prediction values, and the model can learn significant features of the object. Therefore, we further went onto testing this model on the actual live stills of the plant.

Figure 5 shows the OAK-D running our trained SSD model and correctly classifying the leaves. The output is a bounding box with the predicted class name and the confidence percentage. For example, on the left, the model is 64% sure that the leaf in the bounding box is from a bell pepper. To improve the confidence score, we collected images the test leaves on a white background, and re-ran the testing, which produced much better results (Figure 6). However, there is a consistent misclassification of bell pepper leaves as orange leaves.

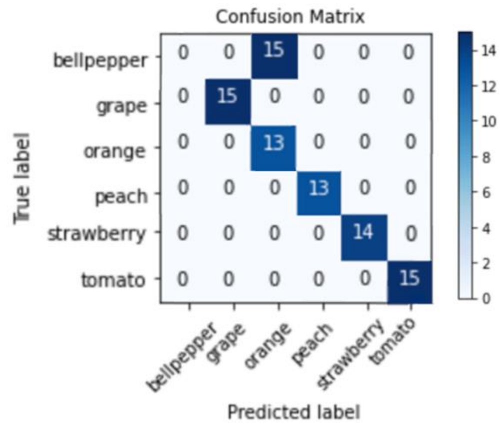


Figure 6. Confusion matrix of six plants trained on Kaggle dataset and tested on leaf samples images collected from a home garden and captured with an OAK-D 4K RGB camera on a white background.

Discussion

Based on our results when testing out the model on the OAK-D, the Mobilenet CNN model can correctly classify the species of six plants when shown an image of single leaves. We discovered that accuracy dramatically improves when leaves are imaged with a white background. By training the model with more images with an updated training set, we expect confidence levels will increase. More training images should include black and white images as well as flipped images.

Currently we image individual leaves. A more robust identification system could make inferences on individual leaves and the whole plant. The OAK-D's depth perception capability can be used to automatically select plant identification modes. Detecting proximity (e.g., between 35 to 50 cm) would cause the system to identify the plant based on leaf structure. A greater distance (>50 cm) would switch to an algorithm that identifies whole plants.

Another future objective is to estimate the volume of the plant to indicate plant biomass productivity. The volume estimation module could be automatically triggered when the camera is at distance sufficient to view the entire plant. An extensive training set that included physical measurements of stem and leaves dimensions and weights would be required.

References

[1] "California Agricultural Production Statistics", www.cdfa.ca.gov/Statistics/ (2019)

[2] VGG-16 training code: https://colab.research.google.com/drive/1yKbT01y5aEnMHGixeD7aLGOdmJ4NTIes

[3] Kaggle dataset: https://www.kaggle.com/abdallahalidev/plantvillage-dataset

[4] depthAI/Roboflow MobileNetv2-SSD training code: https://colab.research.google.com/drive/1EIS2Tz6MxW9dhyfam72qUVzU1nZ5qwJG#scrollTo=mJz_ToJtkufM

[5] SuperAnnotate: www.superannotate.com

[6] Roboflow annotated leaf data download (COCO format): https://app.roboflow.com/ds/IQYY8pQdT5?key=l6UmYlsmdr

[7] Intel's DevCloud: https://devcloud.intel.com/edge/advanced/connect_and_create/

Neighbor Analysis

Berenice Chavez & Maxamillian Schneider

Abstract

Companion planting is beneficial for maintaining healthy and productive gardens. Identifying complementary neighboring plants can benefit gardeners and farmers in maintaining healthy crops. Our goal is to use an OAK-D device and Luxonis's DepthAI library to write a Python program to detect potted plants to measure the distance, area and volume of empty space that exists between two plants. Future directions for this project include, using plant identification to look up preferred inter-plant spacing and suggest complementary plants.

Introduction

Companion planting is used by farmers and gardeners for pollination, pest control, increased crop productivity and space utilization. Through companion planting, the living environment for plants can be maintained so that plants are healthy [1].

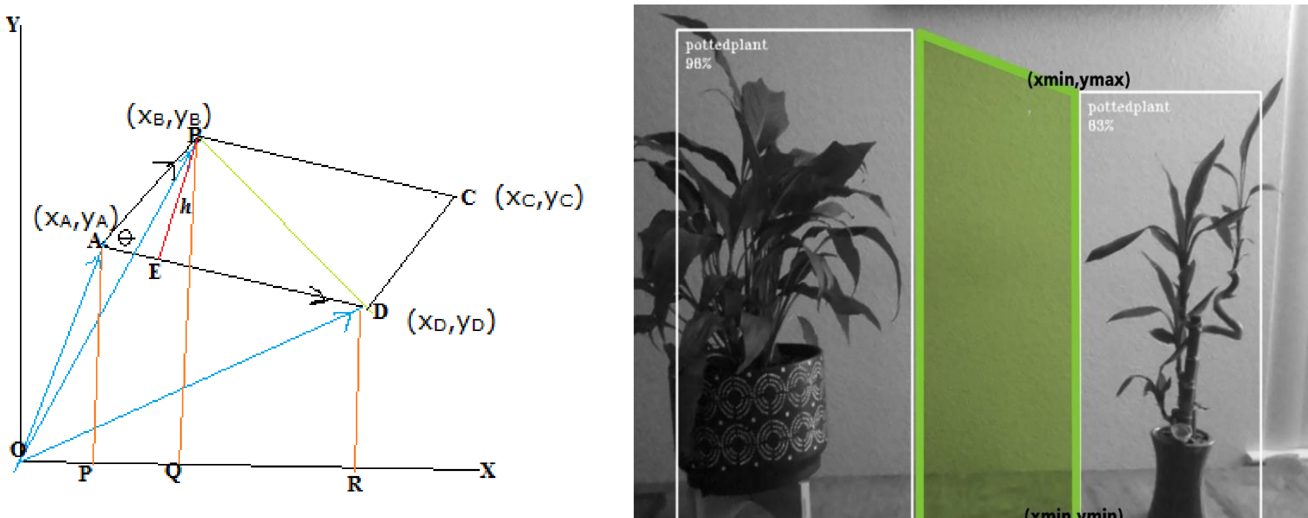
We use the OAK-D device to identify plants and their 3D location in a plot. The OAK-D is a multi-camera device that has neural inferences and depth processing capabilities. The device has two side stereo 1 Mpix mono cameras and a middle 12 Mpix 4K RGB (red, green, blue) camera for color imaging [2]. Luxonis's DepthAI library can be used with OAK-D to help with artificial intelligence, depth vision and the tracking of objects [3]. Data and power are transferred through a USB-C connector, which can be directly connected to a laptop [2].

We modified the DepthAI sample code [4,5] to detect plants and their 3D based on depth and bounding box coordinates. We used this data to calculate the distance and volume between plants.

Methods

Luxonis provides documentation on DepthAI that includes installation directions as well as sample codes to run with the OAK-D camera. One of the sample codes is used for object tracking with the color (RGB) camera (Figure 1, left) [4]. The original sample code detects a potted plant through a trained neural network. Once the detected object is captured by the camera device a bounding box is drawn (Figure 1, right) around the object, the area inside the box is referred to as the region of interest (ROI). Additionally, the DepthAI library helped the OAK-D device's stereo cameras obtain the spatial coordinates and bounding box coordinates of ROI.

Figure 1. (Left) OAK-D device. **(Right)** Potted plant captured by the OAK-D camera with a bounding box drawn around it. The spatial coordinates (X, Y, Z) are printed inside the bounding box.



$$S = |(x_B - x_A) \cdot (y_D - y_A) - (y_B - y_A) \cdot (x_D - x_A)|$$

Figure 2. (Left) Formula and geometric construction to find the area. **(Right)** The green box indicates the space where volume and area need to be found. The vertices used for area calculation are labeled.

To help us accomplish the rest of our goal of measuring distance, volume and area between plants we modified [5] the sample code to:

- Enumerate objects in each frame
- Create a CSV (comma-separated value) file that contains the object-of-interest frame, object ID, object label, bounding box coordinates (xmin, ymin, xmax, ymax), and the spatial coordinates (x, y, z)

Enumerating objects captured in each frame facilitates the process of calculating the distance between objects of interest. The enumerate function in python assigns an index to every item. In this case, every time the OAK-D identified a plant.

The distance calculator helped in calculating the length of empty space between two objects. The calculation was done by using the spatial coordinates x and x of two bounding bonding ROIs. Upon our object detection software making a positive identification of a potted plant, that plant was marked with a unique ID and the x, y, and z locations were saved as reference points. As a second plant enters the field of view the same loop also assigns a unique ID and saves its spatial location. Once both objects have had their special record, the equation of a line is applied to find the distance between the two objects. The difference between the x and y spatial coordinates of both IDs was calculated by using subtraction (x2-x1 & y2-y1). Each result for x and y was then multiplied by two. Lastly, Python has a module called math, that helps with mathematical functions. We used the math module to take the square root of the calculated sum. The resulting distance measured in millimeters is then displayed in the Python console.

Creating a CSV file allowed us to store our data in a more efficient and neater way. This was done using Python's csv module, which is used to read and write data into a format of a spreadsheet. An empty CSV file was created at the beginning of our Python code. Column names (roiID, label, xmin, ymin, xmax, ymax, x, y, z) were added to the empty file. In the CSV file, the bounding box coordinates are represented by the xmin, ymin, xmax, ymax columns. The spatial coordinates are under the x, y, z columns. Creating a CSV file makes it easier for us to access the data in the future. Additionally, we were able to use the data to calculate the volume and area of the space between identified plants. Preliminary calculations for the area between two plants were done in Excel using our CSV data file. The equation and geometric proof used to find the area of a parallelogram with vertices are shown in Figure 2 (left) [6].

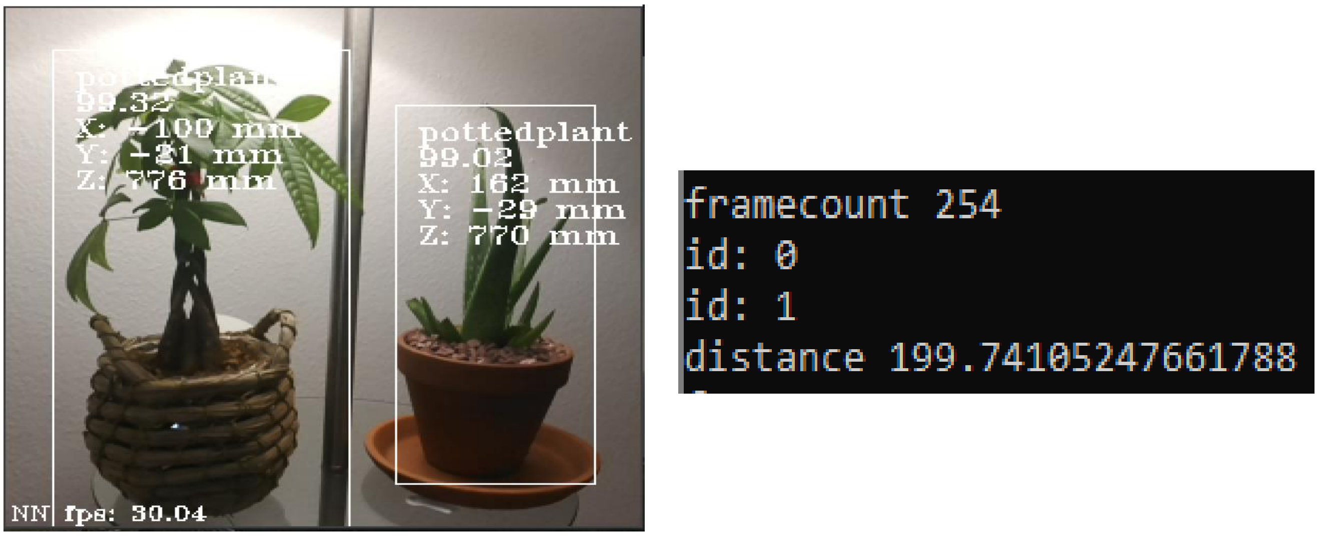


Figure 3. (Left) Two identified plants with their spatial coordinates displayed within the bounding box**. (Right)** Distance is displayed in millimeters through the terminal.

With the CSV data file we obtained, we were also able to create a three-dimensional representation of identified potted plants (Figure 2, right). The 3D visual gave us a visual representation of the volume taken up by a single plant. This depth frame was saved to a CSV with the X and Y pixel coordinates represented by the row and column respectively and the Z value is the number propagating the coordinate. This CSV can then be used to create a 3-dimensional plot using the python library Matplotlib.

Results

The area between two plants is calculated by using the bounding box coordinates of two plants (Figure 3, left). Excel is used for this process and the outcomes serve as preliminary data (Table 1). Two plants have been identified, simultaneously enumerated, and the distance measured. This data gives the user a spatial understanding of how these two plants are related and if there is adequate room for additional plants.

Discussion

We have used the OAK-D to calculate the area and distance of plants in a garden. Using the plant ID (from the Plant ID team), we can look up the preferred plant spacing and companion plants in order and make these suggestions to the user. We can also alert the uses if the existing neighbor distances are outside of the preferred spacing.

Table 1. The bounding box coordinates from two different identified objects.

ID	xmin	ymin	xmax	ymax	area
0	414	158	461	205	21296
1	219	131	327	239	---

Supplemental Information

The Python code and data files for this project can be found in our GitHub repository [5].

References

[1] "Companion Planting Guide", Burpee & Co, www.burpee.com/gardenadvicecenter/areas-of-interest/flower-gardening/companion-planting-guide/article10888.html (2020).
[2] R. Naushad. "Introduction to OpenCV AI Kits (OAK-1 and OAK-D) medium.com/swlh/introduction-to-opencv-ai-kits-oak-1-and-oak-d-6cdf8624517 (2021)
[3] "Welcome to DepthAI Gen2 API Documentation," Luxonis. https://docs.luxonis.com/projects/api/en/latest/
[4] "Spatial object tracker on RGB camera," Luxonis. https://docs.luxonis.com/projects/api/en/latest/samples/29_2_spatial_object_tracker/ (2021).
[5] "Neighborhood, 2021", GitHub Repository, https://github.com/Onejudoka/neighborhood/find/main
[6] Z. Shekhtman. "How do you find the area of a parallelogram with vertices?" (2016). socratic.org/questions/how-do-you-find-the-area-of-a-parallelogram-with-vertices

Environmental Science

Anneke Moeller, Daisy Estrada Reynoso, Lorena Benitez Rivera

Abstract

Soil health is imperative for plants to grow healthy and strong. Currently, inexpensive chemical tests exist to measure soil nutrients and pH but rely on human visual interpretation of color. To improve accuracy and precision, we developed an application to measure and interpret the chemical tests. A Rapitest Soil Test Kit is used to measure chemical content of nitrogen, phosphorus, potassium, and pH present in soil growing basil plants. We use an OAK-D color camera to measure hue and saturation of chemical indicators, perform calibration for each measurement using the kit's color chart, to compensate for ambient lighting, represent the image in HSV (hue, saturation, value) color space, and use interpolation and Beer-Lambert's law to improve resolution.

Introduction

Soil is the foundation of optimal plant growth, the first source for agriculture and food security. We rely on soil for massive food production. Healthy soil is a key to feeding 9 billion people by 2050 (1). Healthy soil is indispensable for healthy food, healthy people, and a healthier planet. We live in one of the most agriculturally exploited states, California. The Mediterranean climate and productive soils of California make the state one of the most productive agricultural regions in the world [2] The Plant Doctor (PD) is a group of SFSU university students that is participating in the global OpenCV Al Competition 2021 [8]. PD is competing in the category of agriculture where our group contributes by evaluating the soil of plants working with the enhanced spectral imaging group to correlate soil data to plant health [3]. The goal of the project is to calculate soil nutrient concentration accurately and reliably by analyzing soil chemistry.

The Rapitest Soil Test Kit is used to test the pH and chemical concentrations of basil plants soil. The Rapitest kit provides the test result information of pH, nitrogen, phosphorus, and potassium (Figure 1). The pH of a soil determines how well plants can utilize nutrients in soil. The Rapitest kit provides the correct level of pH which controls how well plants utilize the nutrients available in the soil. All plants have a pH preference, so it is important to know the pH level of the soil. Knowing the pH level of the soil helps determine the exact condition to get fertilizer and/or adjust the pH more accurately for effective growing. Phosphorus deficiency causes stunted growth and seed sterility. Phosphorus also aids plants maturity, increases vitamin content and aids the plants resistance to disease and winterkill. Potassium strengthens the plant by helping form carbohydrates and promotes protein synthesis. Having sufficient potassium improves the color and flavor of fruit, it also aids early growth, stem strength and cold hardiness. Testing the nitrogen provides information on the plant nutrition, and leaf growth. Too much nitrogen creates overabundant foliage with delayed flowering, disease and poor-quality fruits [9].

Images of the test kit results taken by the OAK-D camera provide the data for image analysis [5]. The test results are interpreted using the colorimeter technique that relies on the Beer-Lambert's Law ($A=\epsilon lc$) which studies the absorbance and concentration of the color chart and sample color values of the Rapitest test results [6].

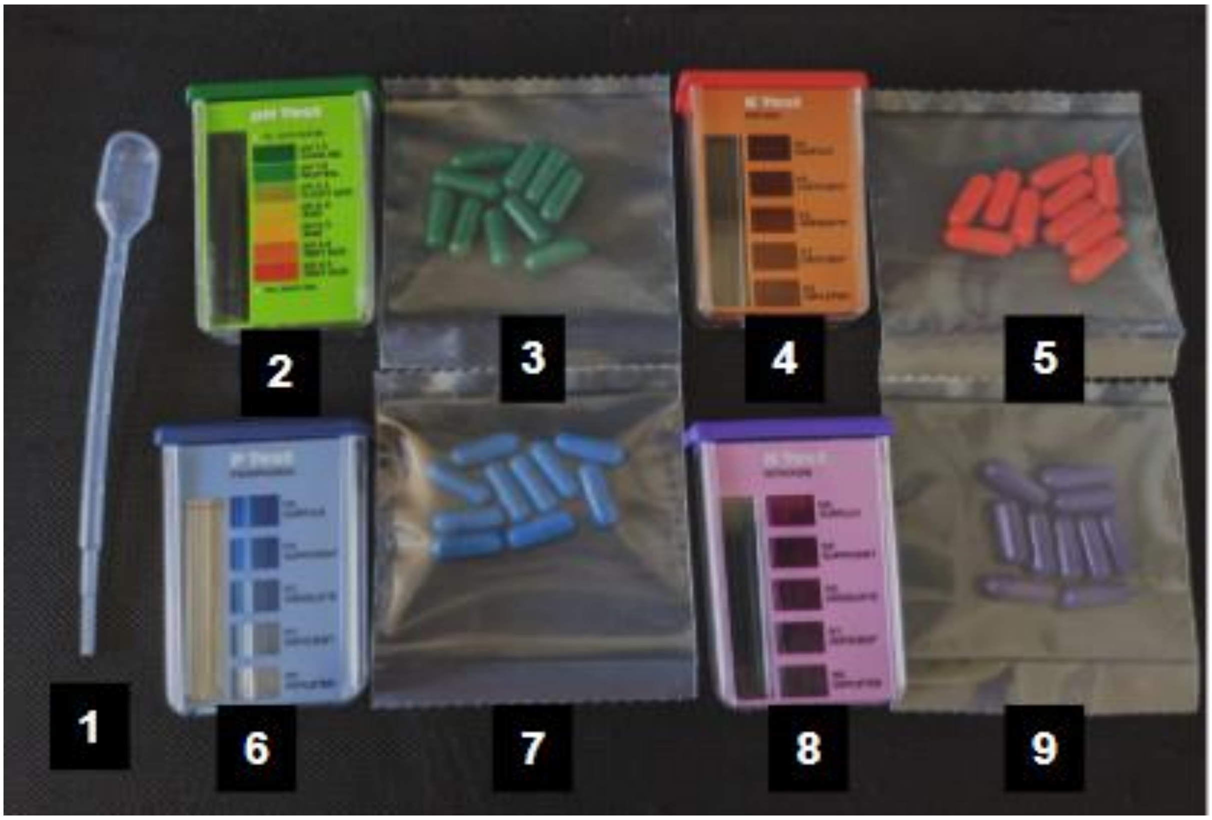


Figure 1. Luster Leaf, Rapitest Soil Test Kit includes: **(1)** Dropper **(2)** pH color reference **(3)** pH capsules **(4)** Nitrogen color reference **(5)** Nitrogen capsules **(6)** Phosphorus color reference **(7)** Phosphorus capsules **(8)** Potassium color reference **(9)** Potassium capsules.

Methods

The Rapitest kit was used to measure soil pH and concentrations of nitrogen, phosphorus, and potassium. Soil from a basil plant was mixed with distilled water and left to settle for up to 24 hours. The solution was placed in the Rapitest kit test chambers [4] and mixed with the respective chemical and pH test capsules. Once the color had developed, images of the test chamber and color reference charges were captured with the RGB OAK-D camera under various lighting conditions at 9 am, 1 pm and 6 pm using either a white background or no background. The images were then processed with an algorithm written in Python using the OpenCV library.

The RGB images were converted to HSV values, a representation of color that aligns with human perception [7]. The HSV color space separates the impact of light brightness (value) from color (hue) and intensity (saturation), providing a more accurate means to measure the Rapitest results.

To collect the HSV values, the desired pixel area in the image had to be selected manually. The pixel coordinate of each area could be used to calculate the average HSV values. The Rapitest soil kit includes a reference and a sample chamber. The HSV values for each color in the reference chart were used to construct a calibration curve. To make this plot the H, S, V values were separated and plotted individually. Then the HSV values of the sample were used to calculate the soil nutrient content based on the constructed calibration curve. The saturation (S) value was used to determine nitrogen, potassium and phosphorus concentrations. The hue (H) was used to test for pH.

Image quality metrics were developed to determine whether an image would be a good candidate for calculating soil chemistry. A good image would have a reference chart (Figure 1) for chemical concentration with a monotonic increase in

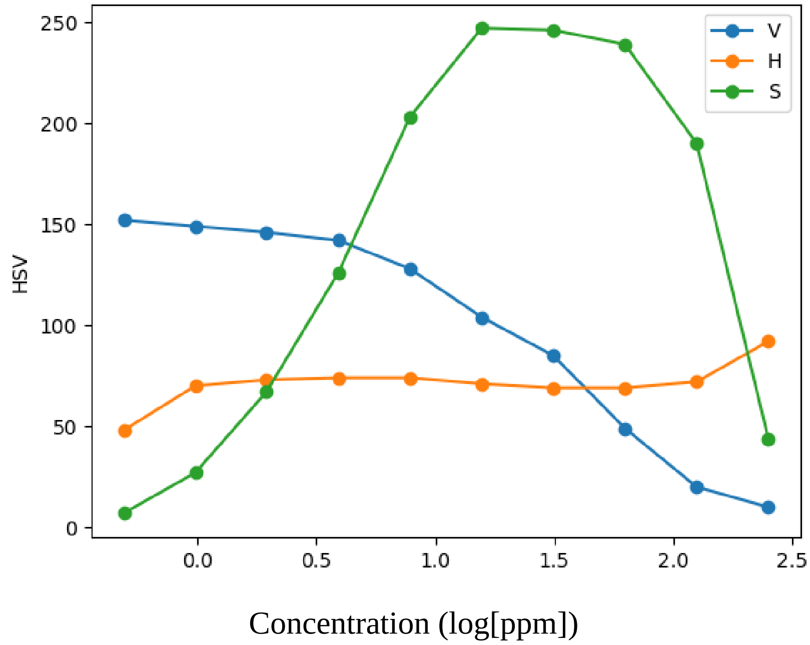


Figure 2. Plot of HSV values (y-axis) vs concentration of green food coloring in log[ppm] (x-axis) where blue=value, orange=hue, and green=saturation.

saturation and low variance in hue. The reference chart for pH would have a monotonic increase in hue and low variance in saturation.

The sensing range and agreement with Beer's law was tested using serial dilution of food coloring. A food coloring standard solution was prepared by mixing five drops of green food coloring with 10 ml of water, which corresponds to 250 ppm. The dilutions were prepared by a factor of ½. The concentration was plotted on a logarithmic scale to accommodate the wide range of concentration. A constant slope would indicate compliance with Beer's law (Figure 2).

Images of the Rapitest color reference charts were captured, with and without a white background, and the HSV values plotted (Figure 3 and 4). Soil samples were collected, mixed 5:1 with distilled water by volume, manually shaken for approximately one minute, then allowed to settle for two hours. Water on top of the settled soil was pipetted into the Rapitest chamber and imaged with the OAK-D RGB camera. Soil from a dying basil plant was used as test soil and store-purchased Sphagnum Peat Moss soil was used as control soil. Several tests were performed on the same soil sample to check consistency of the Rapitest indicator chemicals.

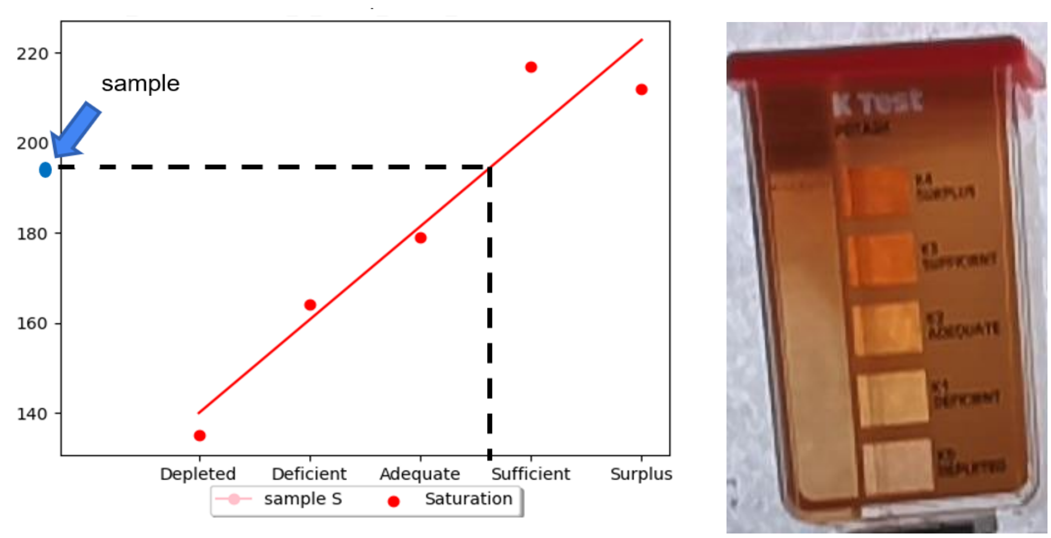


Figure 3. (Left) Calibration curve of color references (x-axis) on a potassium test container and the resulting saturation (y-axis) value produced from an image captured on a white background with indirect lighting. The saturation of the liquid from a soil sample is indicated by the blue arrow. (Right) The liquid from a soil sample mixed with the chemical indicator for potassium is poured into the test chamber on the left section of the container.

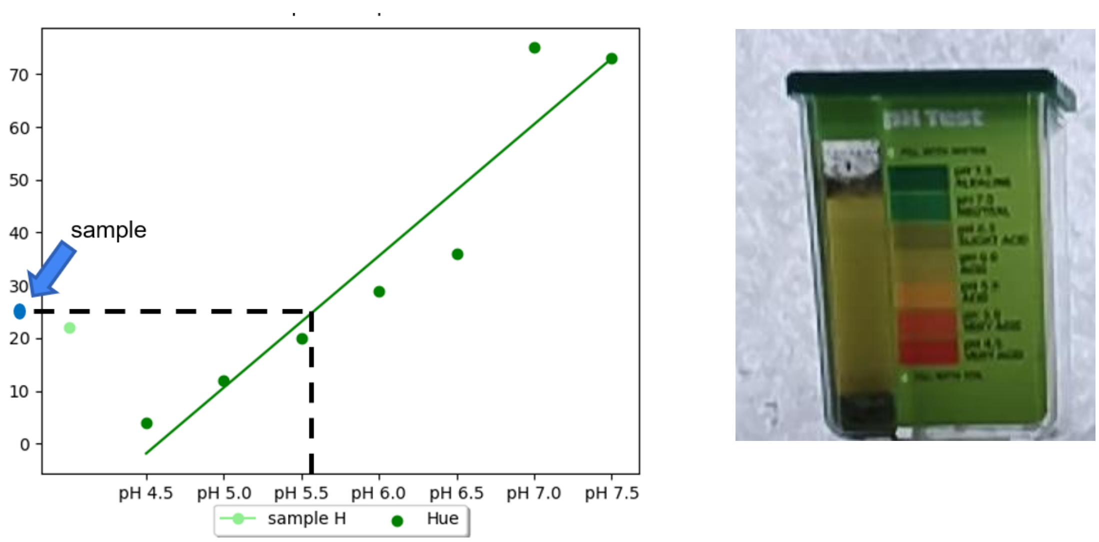


Figure 4. (Left) Calibration curve of color references (x-axis) on a pH test container and the resulting hue (y-axis) values produced from an image captured on a white background with indirect lighting. The hue of the liquid from a soil sample is indicated by the blue arrow. (Right) The liquid from a soil sample mixed with the pH indicator is poured into the test chamber on the left section of the container.

Discussion

Our test with food coloring verified the use of saturation (S) as the absorbance variable to determine chemical concentration according to Beer's law. When the concentration gets very high, the resulting chemical indicator is so dark that little light reaching the image sensor, producing erroneous saturation values. We detect this condition and ignore any saturation reading when the brightness value (V) is less than 100. Similarity hue (H) is a good metric for pH as long as the brightness value (V) is between 25 to 150.

If a sample passes the brightness value (V) criteria, we can use Beer's law to find the chemical concentration. Further, we can use the linear fit of the calibration curve to predict chemical concentration beyond the five levels provided by the color references. In Figure 3 the potassium soil sample chemical indicator produces a saturation (S) value of 188. While a human may estimate is as somewhere between "Adequate" and "Sufficient", performing an interpolation on the linear fit can give a numeral prediction of where it lies between these indication levels. Generating a numeric assessment can be useful to determine if a chemical concentration is increasing or decreasing between readings. Similarly, the pH reading of the soil in Figure 4 can be interpolated to a higher resolution than the seven values provided by the color references.

Supplemental Information

Data files are kept in two sub-folders. Folder 'data' contains files as they're used in the analyses. Sub-folders 'Unhealthy Plants Soil Pictures' and 'Healthy Plants Soil Pictures' contain the images of the nutrients tested that are saved here: https://drive.google.com/drive/folders/1zHRbAvdVvFsvAQUZB4UhRPZC6di3C6_A?ths=true

Code is for Python programming language. Scripts are organized by the data set each one analyzes, and they can be found here: https://github.com/anneke32/Environmental_Science

Rapitest Soil Kit https://www.amazon.com/Luster-Leaf-1601-Rapitest-Phosphorous/dp/B0000DI845/ref=sr_1_1?crid=1MJ577ETIED49&dchild=1&keywords=rapitest+soil+test+kit&qid=1621568974&sprefix=rapitest+soil%2Caps%2C259&sr=8-1

Soil Testing Protocol *https://docs.google.com/document/d/1jhtjPZLJipHrU7QfkKwM4t6SvpOyqJKmt1j2hpw6WiA/edit*

References

[1] Feeding 9 Billion | National Geographic. Accessed May 14, 2021, https://www.nationalgeographic.com/foodfeatures/feeding-9-billion

[2] Soil Management – California Agricultural Water Stewardship Initiative. Accessed May 14, 2021, http://agwaterstewards.org/practices/soil_management

[3] Zimmerman, Thomas. "The Plant Doctor: a Tricorder for the Garden" pdf file.

[4]"Rapittest Soil Test Kit" #1601, Luster Leaf Products, Atlanta, GA. Accessed March 11, 2021, www.lusterleaf.com/nav/soil_test.html

[5] OpenCV AI Kit. Kickstarter. Accessed May 14, 2021, https://www.kickstarter.com/projects/opencv/opencv-ai-kit

[6] The Beer-Lambert Law. Chemistry LibreTexts. Published October 3, 2013. Accessed May 14, 2021, https://chem.libretexts.org/Bookshelves/Physical_and_Theoretical_Chemistry_Textbook_Maps/Supplemental_Modules_(Physical_and_Theoretical_Chemistry)/Spectroscopy/Electronic_Spectroscopy/Electronic_Spectroscopy_Basics/The_Beer-Lambert_Law

[7] HSL and HSV - Wikipedia. Accessed May 14, 2021, https://en.wikipedia.org/wiki/HSL_and_HSV

[8] OpenCV AI Competition, Accessed May 20, 2021, https://opencv.org/opencv-ai-competition-2021/

[9] Rapitest Soil Test Kit. Accessed May 20, 2021, https://www.biconet.com/testing/rapitestSoilKit.html