# Mapping Floral Resources for Bees using Drone Imagery

*By*
Arya Massarat

A thesis presented to
the Biology Department
of Harvey Mudd College
in partial fulfillment of
the degree of Bachelor of Science

HARVEY MUDD COLLEGE
April 27, 2020

## Abstract

Poor nutrition among modern day honey bee colonies is contributing to their decline. Yet understanding how the diversity and abundance of flowering species around a colony affects its health remains difficult because of the manual labor required to analyze these large foraging landscapes. We describe a procedure for automatically mapping the species of flowering plants around a colony from overlapping drone images. We developed a pipeline for stitching the images together, identifying plants within them, and classifying each plant by its species. The resulting map of the flowering species surrounding a colony could be used in future experiments that aim to assess how a colony's health and foraging behavior is influenced by the spatial distribution of the floral species in its vicinity.


## Introduction

The western honey bee (or *Apis mellifera*) may be one of the most important living species on the planet. Pollinators in general are responsible for the continued existence of nearly three fourths of the crop species responsible for 90% of our food worldwide (Kluser et al., 2010). Bees are the predominant group of pollinators in most regions and the honey bee is the most common domesticated bee species (Kluser et al., 2010). Yet the most important living species has been rapidly disappearing. The number of honey bee hives has seen a substantial decline of at least 59% since 1947 (Chisel, 2015).

Many causes have been linked to the decline of the honey bee, including the presence of parasites and pesticides (Kluser et al., 2010). However, ecologists suspect that the decline is due to the rise of agricultural monoculture, where farmers grow only one or a few species of

flowering plants in a large landscape. Monocultures could be problematic because they lack nutritional diversity for honey bee colonies (Garibaldi et al., 2009; Nicholls & Altieri, 2013). Indeed, evidence has shown that the nutrition available to a colony can be an important factor for predicting colony survival (Dolezal & Toth, 2018). However, the subject remains challenging to study. Knowledge of the distribution of the species of flowering plants in a landscape is crucial for understanding the effect, but manually recording the location and species of the available plants is usually untenable, especially since honey bees can travel up to 10 km from their nests (Donaldson-Matasci & Dornhaus, 2012; Hofmann et al., 2017; Von Frisch & Chadwick, 1967). An automated method for identifying flowering plant species in a landscape would better enable efforts to understand which plants contribute to colony survival.

Automated methods for analyzing landscapes in ecology have usually been done with satellite imagery because it can be used to capture large, expansive landscapes very quickly (Singh et al., 2015 and Barnhart et al., 2019). However, satellite imagery is not usually of a sufficiently high resolution to be used for identifying individual parts of a plant (Anderson & Gaston, 2013). More recently, low-flying drones or unmanned aerial vehicles (UAVs) have been used to capture particular attributes (or *features*) of a plant (Anderson & Gaston, 2013). This extra level of information has lent itself to a different type of analysis known as OBIA (object-based image analysis).

OBIA works by first grouping adjacent pixels together into objects using a process known as *segmentation* (Blaschke, 2010). When running OBIA on a drone image, for example, each cluster of adjacent pixels representing a plant would be treated as a distinct object. By contrast, traditional methods relying on lower resolution imagery would treat each pixel as an

object, since a plant rarely spans more than one pixel in these images. After segmentation, objects that appear similar to each other are assigned specific labels, usually by a machine learning classifier (Blaschke, 2010). For example, OBIA can be used to label objects within a forest as tree crowns, objects within a national park as trees, or objects within a farm or grassland as plants (Lu & He, 2017; Pflanz et al., 2018; Singh et al., 2015; Yu et al., 2006). Our method uses OBIA on the drone imagery of a landscape to label plants by their species. By using OBIA rather than the traditional pixel-based method, we can leverage the high resolution of our drone images to calculate local *features* for every plant, which can help increase the accuracy of our classifier. For example, a feature like contrast can be a good indicator of the abundance of flowers in the plant, since the flowers stand out against the green of the vegetation surrounding them. We can calculate the total contrast in a plant in an effort to measure the abundance of these flowers. Indeed, this may be a very useful feature for distinguishing flowering from non-flowering plants. However, we would not be able to do this type of analysis with a traditional pixel-based method because contrast cannot be calculated for a single pixel. UAV and drone generated imagery offers unique advantages over satellite imagery when used with OBIA.

Unfortunately, UAVs can sometimes be too close to the target to capture the entire landscape. Thus, the overlapping images a UAV captures must typically be stitched together before they can represent the entire landscape of interest. The stitched images together create an *orthomosaic* (Gross & Heumann, 2016). In addition, images are not always taken at the same orientation or angle relative to the ground, and thus, a process known as *orthorectification* is required to scale images correctly. Software like Agisoft Metashape is typically used for the stitching and orthorectification steps, although other alternatives also exist (Gross & Heumann,

2016; Lu & He, 2017; *PhotoScan/Metashape Professional*, 2019). One advantage to using Metashape is that, in addition to an interactive GUI, it has a Python package that can be used to automate stitching within a script. Unfortunately, Metashape is known to frequently introduce artifacts into the orthomosaics that it generates (Gross & Heumann, 2016). This could present challenges for methods that attempt to perform OBIA on the flawed orthomosaic.

Prior work in our lab used a more traditional, pixel-based method (rather than OBIA) to map the flowering plant species in a landscape. Unfortunately, the accuracy of this method was sensitive to the presence of artifacts in the orthomosaic. In addition, sections of the pipeline, such as the stitching step, were not fully automated and required occasional user input. Also, the method used a sliding window rather than an object-based analysis method. As we previously discussed, object-based analysis methods can more accurately classify regions in an image by calculating features for the object rather than for a naive group of pixels. In addition, the sliding window approach required a large execution time and a lot of memory, as it was calculating features for every possible window in the orthomosaic, even though many of these windows did not actually contain plants.

We developed a new, completely automated pipeline for using drone imagery to automatically generate a map of the flowering plant species that are available to a honeybee colony. The new pipeline starts by stitching the drone images together and follows with the two main steps in OBIA, segmentation and classification (Fig 1). OBIA is used to identify individual plants and label them by their species. Unlike other methods, however, our pipeline can use two different strategies, hereinafter referred to as the default strategy and the “experimental” one. Both strategies begin by stitching the original drone images together into an orthomosaic (Fig 1).

Then, segmentation is performed on the stitched orthomosaic in order to locate plants within the landscape (Fig 1). After this, the two strategies diverge, each of them implementing a different approach for classifying the plants by their species. The default strategy runs the classification step directly on the segmented plants from the orthomosaic (Fig 1). Unfortunately, this makes the classification accuracy dependent on the quality of the orthomosaic, which is likely to have artifacts. The experimental strategy aims to avoid this pitfall by mapping the plant locations from the orthomosaic back to the original drone images. Then, classification is performed on the plant locations from the original drone images rather than the orthomosaic (Fig 1).

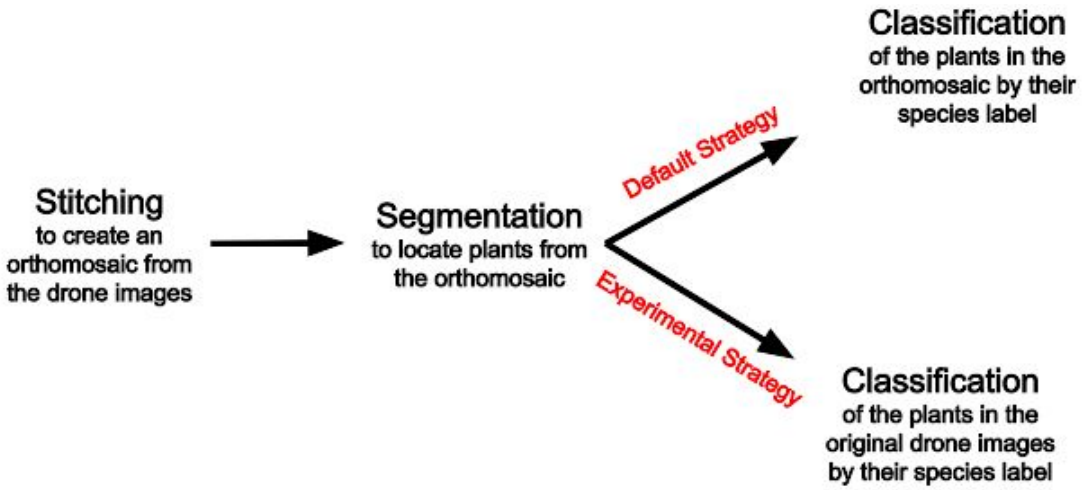



**Fig 1** A skeleton diagram of our pipeline's major steps: stitching, segmentation, and classification. In the default strategy, classification is performed on segments obtained from the orthomosaic. In the experimental strategy, these segments are first mapped back to their locations in the original drone images and classification is performed on these images instead.

In contrast with the default strategy, the experimental strategy runs the classification step on many small, individual drone images at the same time, rather than on a large orthomosaic image. Thus, while the experimental strategy involves more steps, it can execute more quickly than the default strategy if we parallelize the execution of those steps. In addition, the experimental strategy provides us with more control over the total memory usage of the pipeline, since classification on each drone image will require less memory than on the entire orthomosaic

and a limit can be placed on the number of drone images that are classified at the same time. After developing the method, we evaluated the experimental strategy against the default in order to confirm the existence of these improvements in accuracy, speed, and memory usage.

## Methods

We created a computational pipeline for generating a map of flowering plant species over a landscape captured by drone imagery. The pipeline is best characterized by its three major steps: stitching, segmentation, and classification. However, there are many smaller steps that are required between these. In addition, we developed two different strategies for performing the classification step in an effort to improve the accuracy, runtime, and memory usage of the pipeline. We next describe all of the steps in the pipeline, making note of how they are used in each of the two strategies and discussing the advantages one has over the other. The default strategy is described first, and then an explanation is given for how the experimental strategy differs. The source code for the pipeline is available on Github (https://github.com/beelabhmc/flower_map).

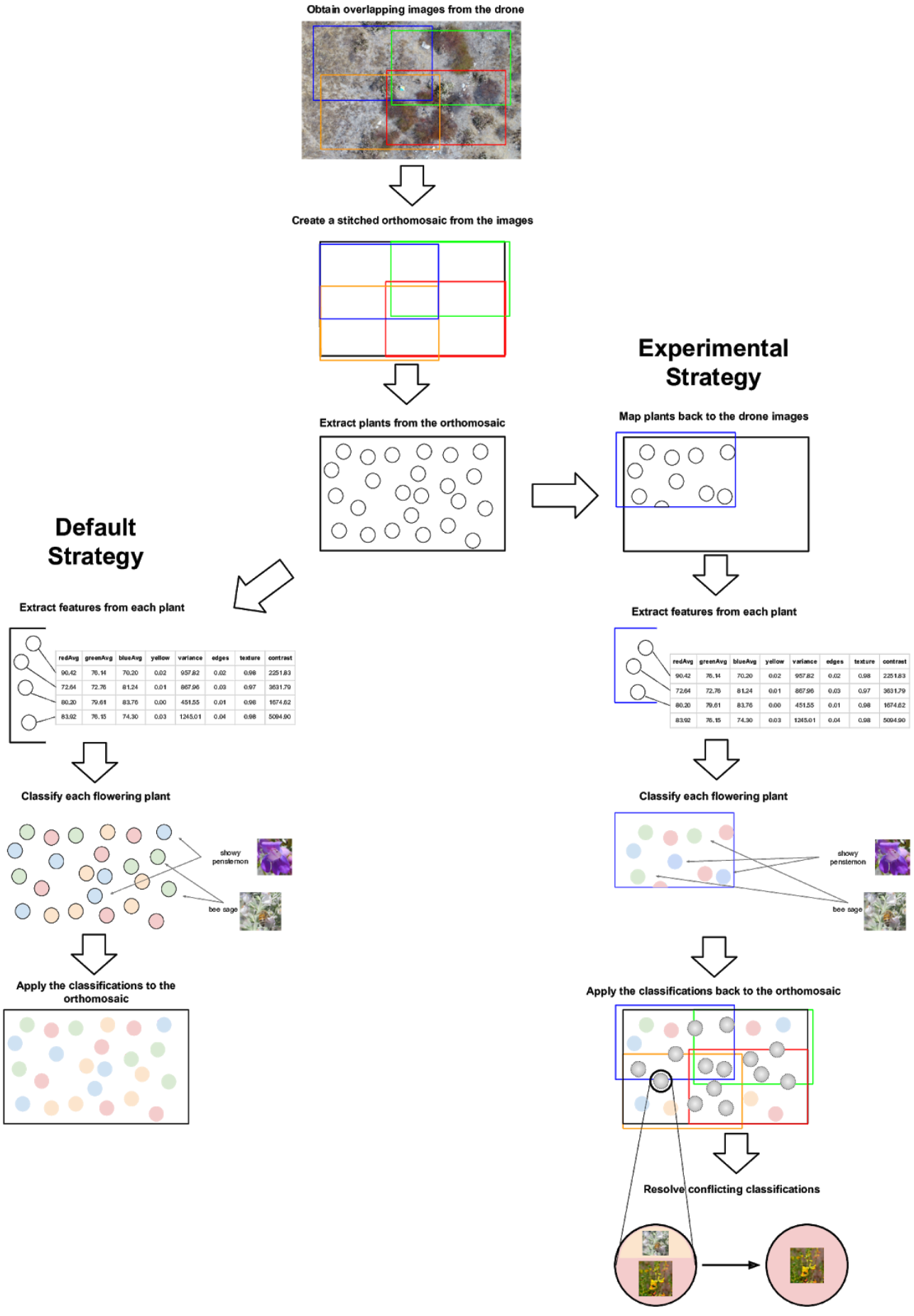

Obtain overlapping images from the drone
Create a stitched orthomosaic from the images
Extract plants from the orthomosaic
Experimental Strategy
Map plants back to the drone images
Default Strategy
Extract features from each plant
redAvg greenAvg blueAvg yellow variance edges texture contrast
90.42 76.14 70.20 0.02 957.82 0.02 0.98 2251.83
72.64 72.76 81.24 0.01 867.96 0.03 0.97 3631.79
80.20 79.61 83.76 0.00 451.55 0.01 0.98 1674.62
83.92 76.15 74.30 0.03 1245.01 0.04 0.98 5094.90
Classify each flowering plant
showy penstemon
bee sage
Apply the classifications to the orthomosaic
Extract features from each plant
redAvg greenAvg blueAvg yellow variance edges texture contrast
90.42 76.14 70.20 0.02 957.82 0.02 0.98 2251.83
72.64 72.76 81.24 0.01 867.96 0.03 0.97 3631.79
80.20 79.61 83.76 0.00 451.55 0.01 0.98 1674.62
83.92 76.15 74.30 0.03 1245.01 0.04 0.98 5094.90
Classify each flowering plant
showy penstemon
bee sage
Apply the classifications back to the orthomosaic
Resolve conflicting classifications

**Fig 2** A schematic of the major steps of both strategies in our pipeline. First, overlapping drone images are taken of the landscape. These images are stitched together into an orthomosaic using the overlapping regions of the images as a guide. In both strategies, segmentation is used to isolate each plant. In the default strategy, features of each plant are calculated and then used to classify the plant by its species. In the experimental strategy, the segments are mapped back to the original drone images first, and then the same steps are performed. Because the experimental strategy might lead to conflicting classifications of the same plant in different images (pictured above in grey), an extra step is required to resolve these differences at the end.

## Stitching

The first step in the pipeline is stitching, in which the drone images are aligned with each other based on their overlapping regions (Fig 2). Many software packages exist that can be used to perform this task, but we use Agisoft Metashape because it is commonly used for this task (Barnhart et al., 2019; Lu & He, 2017), and it has a python-based API that allows us to automate the stitching process within our pipeline (*PhotoScan/Metashape Professional*, 2019). Metashape works by aligning the images based on a series of shared points (a *point cloud*) that it locates within each image. This is where it becomes crucial that images overlap in order to increase the number of shared points in the point cloud. The software outputs a special *project file*, which can be used to recreate the alignments of each image later on in the experimental strategy. The project file also contains a stitched orthomosaic, the result of merging and aligning each individual drone image (Fig 2). This orthomosaic is exported to a full quality TIFF file for processing in the next steps.

## Segmentation

Next, the pipeline runs the segmentation step, which extracts the location of the plants in the orthomosaic. The first step in segmentation is to identify a set of pixels within the orthomosaic that may belong to plants. For every pixel in the orthomosaic, we generate two features that can be used as an indication of this. These are 1) the green from the pixel's RGB value, and 2) the contrast in a radius of two pixels around each pixel (Fig 4). We used these features because they tend to be high in regions of the image where there are plants but low in

regions where there are not any plants. In addition, the two features tend to complement each other, since the amount of green in a plant will be low if it has many white flowers but the contrast will be high because those flowers will stand out against the surrounding green. Because contrast can take a long time to calculate, we only generate this feature for a few pixels in the image spaced 30 pixels apart from each other, and the rest of the pixels are assigned the nearest calculated value.

Next, threshold values are chosen for the green and contrast values. All pixels that have a feature value above the threshold are recognized as being part of a plant, while the other pixels are labeled as ground (Fig 4). The two thresholded versions of the image are then overlaid on top of each other so that pixels labeled as part of a plant under either threshold are labeled as part of a plant in the merged result (Fig 4). This is equivalent to taking the union of the two sets of pixels that meet their respective threshold requirements.

The next steps in segmentation aim to create clusters of thresholded pixels, where each cluster represents a plant. *Morphological operations* like *dilation* and *erosion* are used on the result from the previous step in order to remove noise and isolate separated, clumped regions of pixels (Fig 3). First dilation is performed to further concentrate pixels in regions of the image where they are already highly concentrated (Fig 3). Then erosion is used to further dilute pixels in regions of the image where they are not concentrated (Fig 3). Repeated iterations of erosion and dilation are used if images are particularly noisy. In fact, morphological operations are also applied to the thresholded contrast in the image before it is merged with the green in order to reduce noise in the contrast values (Fig 4). Morphological operations help isolate clusters of pixels corresponding to each plant (Fig 3 and Fig 4).

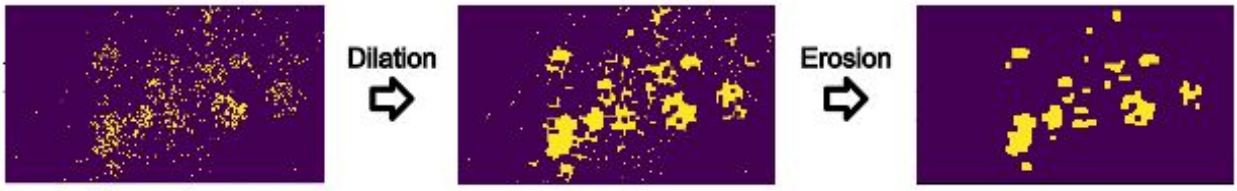



**Fig 3** Demonstration of the morphological operations, dilation and erosion. Dilation colors pixels yellow in regions of the image where the yellow is already highly concentrated. Erosion does the opposite, coloring pixels purple in regions of the image where the yellow is diluted. Morphological operations are used to reduce the noise in an image, yielding distinct, continuous regions of yellow that can be labeled as plants.

While helpful for isolating clumps in an image, morphological operations are a blunt tool. Too many iterations of dilation may merge adjacent plants into a single large plant, while too many iterations of erosion may lead to the disappearance of plants altogether. It can be difficult to balance the use of these two operations. We use a method known as *the watershed algorithm* to more intelligently group (and separate) clusters of pixels into distinct, segmented regions. The watershed algorithm takes as input two images, each of which is generated using morphological operations (Fig 4). The first image contains clumps of pixels known with a high confidence to be contained within a plant, while the second image contains clumps that are known with a lower confidence (Fig 4). The lower confidence regions are generated by performing dilation on the higher confidence regions, so the higher confidence regions are usually contained within their lower confidence counterparts. The watershed algorithm simulates flooding of the lower confident regions from the highly confident regions, which are treated as sources for the water. Boundaries for each plant are defined where the water from one source meets that of another source. This maintains adjacent plants as distinct entities when they appear merged as one large plant in the lower confident image but remain separated in the higher confident image (Fig 4).

In the last step of the segmentation script, each separate cluster of pixels should represent a plant (Fig 2). The contour of each cluster is represented as a list of coordinate points within the image. These contours defining each segmented region are extracted to a special JSON file,

which can be read by a software called LabelMe (Wada, 2016). LabelMe provides us with an interactive GUI for editing the contours and adding new ones if need be (Wada, 2016). The JSON file can also be interpreted by downstream steps in the pipeline.

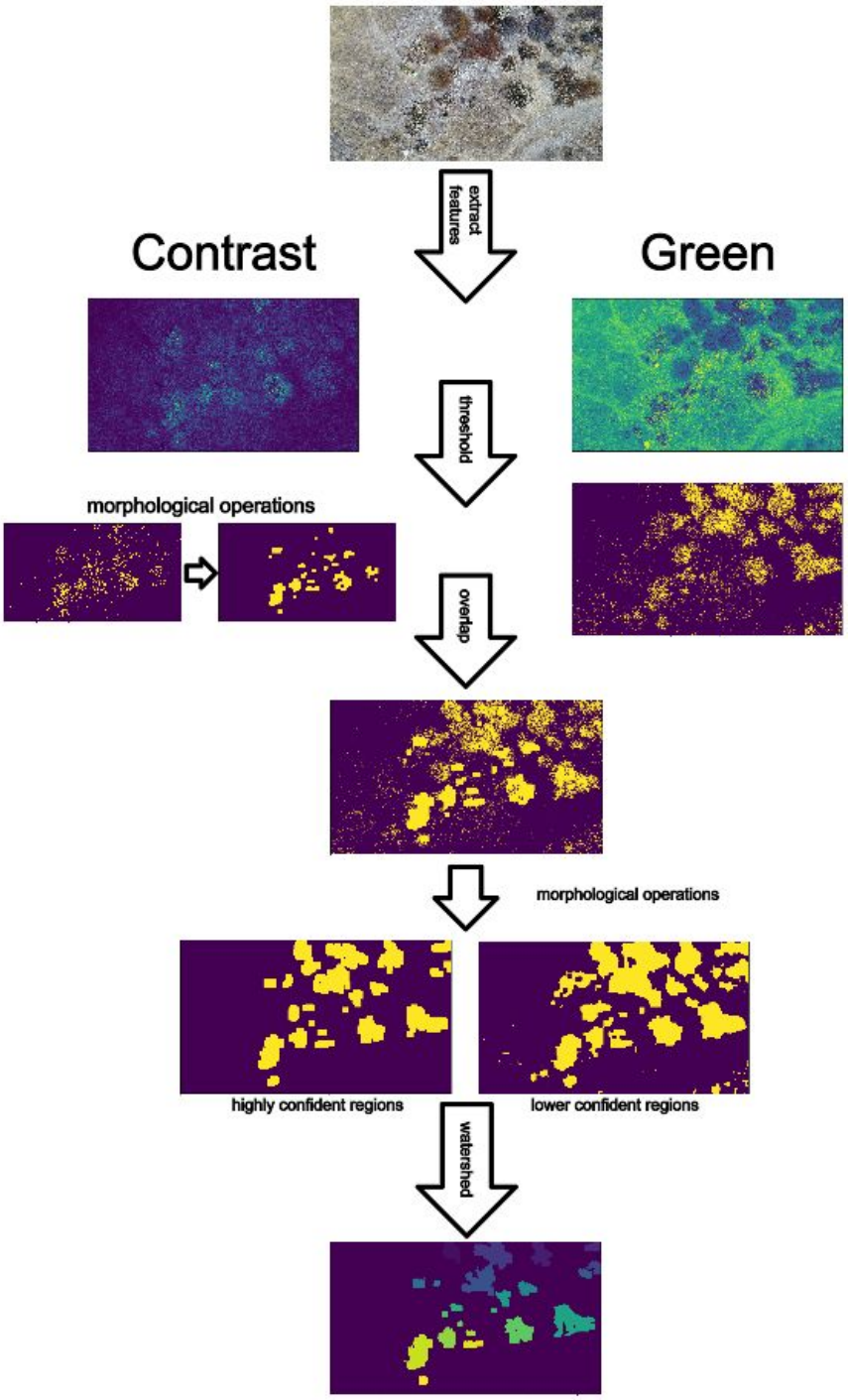



**Fig 4** A schematic of the steps used in segmentation, in which we extract the location of the plants in the orthomosaic. In the first step, contrast and green features are calculated for every pixel in the orthomosaic. Next, pixels with high feature

values are colored yellow, while the rest are colored purple. A union is taken of the sets of yellow pixels from the contrast and green images in order to merge them. Then, morphological operations are used to identify regions with high concentrations of yellow pixels. Finally, the watershed algorithm resolves the differences between a set of regions known to be plants with high confidence and a set of regions known with lower confidence. Each separate plant is colored a different color in the final image.

## Feature Extraction

The next major step in the pipeline is classification, where we label the segmented regions from the previous step by their species (Fig 2). To do this, we must first calculate *features*, which describe unique aspects of each segmented region (Fig 2). The classifier uses these features to decide which species each region matches best. The feature extraction step takes as input 1) the JSON file containing the regions that were segmented and 2) the orthomosaic from which the regions came. Cassie Burgess, a former lab member, identified several features that were useful for classifying flowering plants: 1) variance of color, 2) average amounts of red, green, and blue, 3) the number of pixels that fall within the yellow hue, 4) HSV skew and standard deviation, 5) proportion of edge pixels, and 6) percentage of grids within the image that have high texture.

In addition, we also calculate another set of features which has proven useful in many studies that attempt image analysis: the texture of an image, as measured by the Gray-Level Co-occurrence Matrix (GLCM). The GLCM provides useful characteristics like contrast, dissimilarity, and homogeneity, which tend to have high values in non-background regions of an image (Hofmann et al., 2017; Sebastian V et al., 2012). All of the resulting feature values are exported to a TSV file where each row label is a segmented region and the columns refer to each of the features (Fig 2).

## Classification

Once the pipeline has feature values for each segmented region in the orthomosaic, it can finally assign a species label to each plant (Fig 2). The classification step uses a machine learning classifier to predict the species of each segmented region based on the combination of the values of its features. First, the classifier uses pre-labeled training data to discover which feature values correspond best with each species label. Then, the features of segmented regions without pre-assigned labels are used to predict which species the segmented regions represent.

Recently, the remote sensing community has begun to use a novel non-parametric machine learning classifier known as *the random forest (RF)* (Belgiu & Drăguț, 2016). Among most studies that analyze landscape imagery, the RF has shown the best accuracy of the other classifiers (Belgiu & Drăguț, 2016; Feng et al., 2015; Shafri et al., 2007; Singh et al., 2015). Random forests are also useful because they can provide a metric for evaluating how *important* each feature is for classifying the objects (Belgiu & Drăguț, 2016). The random forest calculates a probability of each segmented region belonging to a species. A segmented region is labeled as a species if it is predicted to be that species with a probability greater than 50%. In order to visualize the results, each segmented region and its assigned species labels are overlaid on top of the orthomosaic to create a map of the flowering plant species across the landscape (Fig 2).

## The Experimental Strategy

One of the challenges with using the Metashape stitching software is that it can be prone to create artifacts within the orthomosaics that it generates. When the software fails at resolving

pixels that are particularly different from each other, these artifacts appear as regions that are either blurry or simply completely black (Fig 5).

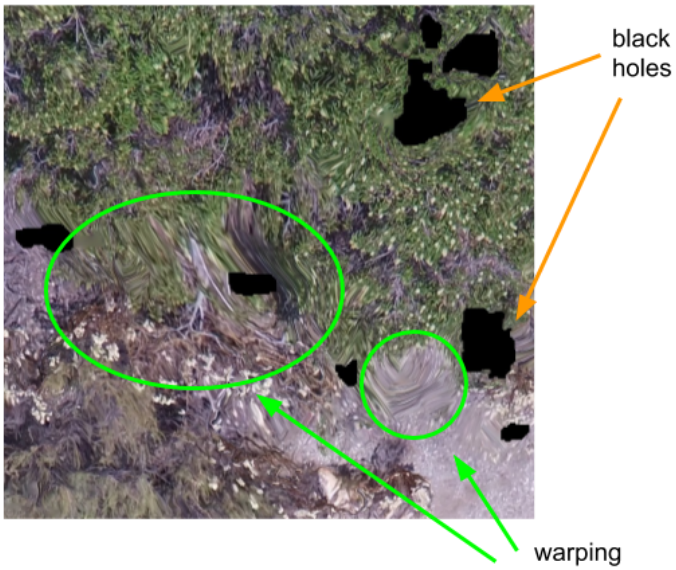



**Fig 5** Black holes and warping are artifacts that can result from the use of the Metashape stitching software.

We surmised that the artifacts in the orthomosaic could affect the quality of our classifications. The experimental strategy was developed to improve the accuracy of the classifier by extracting features from the original drone images rather than the orthomosaic (Fig 2). It also allows us to parallelize the feature extraction and classification steps to reduce the running time and memory usage of the pipeline.

The experimental strategy diverges from the default after segmentation of the orthomosaic is performed (Fig 2). First, it transforms the coordinates of the contours of every segmented region in the orthomosaic to the coordinate systems of each of the original drone images. Since each drone image can be shifted, rotated, and scaled during its inclusion in the orthomosaic, this type of transformation may be complicated (Fig 6). However, it is possible using a series of functions available through the Metashape project file, which the pipeline

created in the stitching step. For every drone image, the pipeline creates a JSON file containing segments from the orthomosaic.

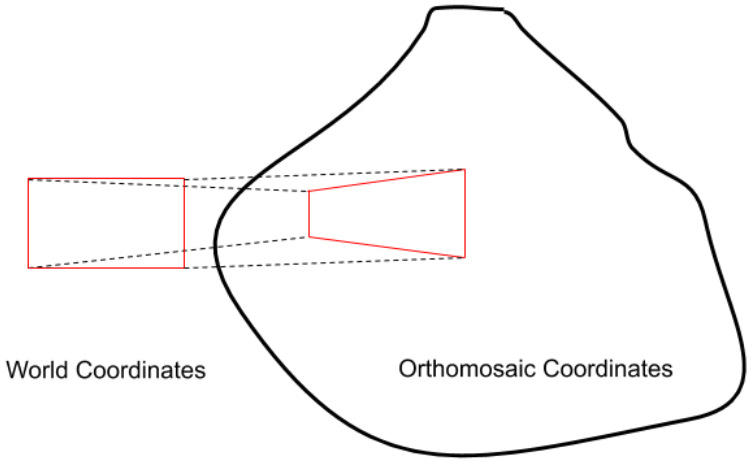



**Fig 6** When stitching, image coordinates can be shifted, rotated, and scaled from their locations in the real world to those in the orthomosaic. It is not immediately obvious, then, how one might reverse the transformation later.

The JSON files (and the drone images to which they belong) are passed through the rest of the pipeline, including the feature extraction and classification steps, just as is done in the default strategy. The experimental strategy therefore ensures that classification is performed on segments lacking artifacts from the original images rather than those from the orthomosaic.

The last step of the experimental strategy is to classify each of the segments in the orthomosaic using the species assigned to the corresponding segments in each drone image. Note that a single segment in the orthomosaic may appear in multiple drone images due to the overlap between the images (Fig 2). Each of these duplicates of the segment may be assigned a different classification (Fig 7). In addition, parts of the segments may be cut off if they straddle the edges of an image (Fig 7). Thus, the last step of the experimental strategy requires that we resolve these conflicting classifications while also taking into account that some of the segments may be only partial (Fig 2). This is achieved by taking an average of the classification probabilities, where

each segment's probability is weighted by the area it encompasses within the original segment (Fig 7).

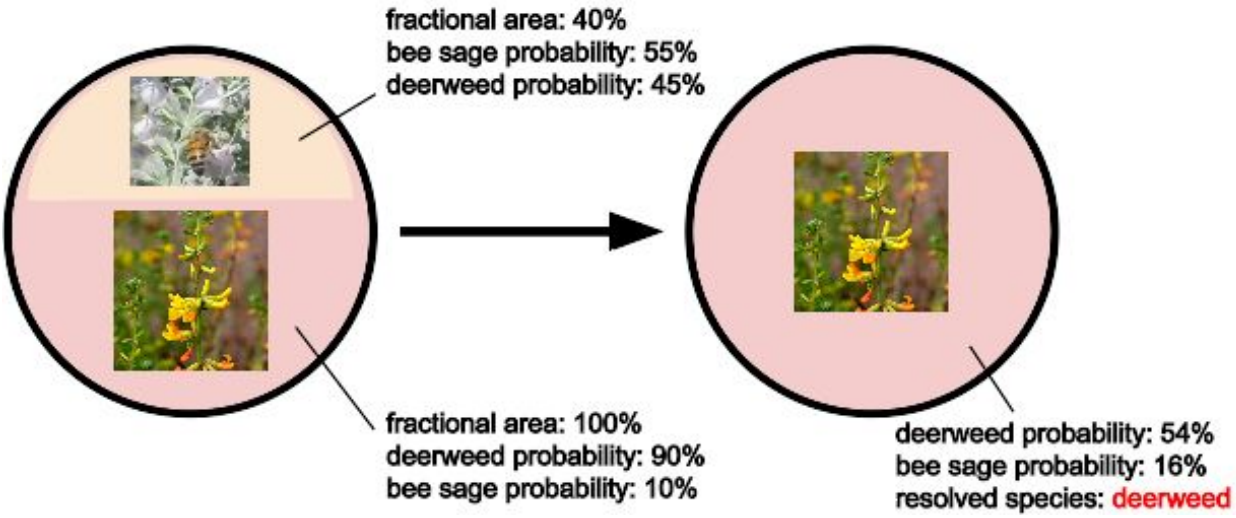



**Fig 7** Resolving conflicting classifications in the experimental strategy. An average is taken of the random forest classification probabilities of each conflicting classification. Since some segmented regions from the orthomosaic may not appear in their entirety within the drone images, this average is weighted by the fraction of the total area that each segmented region encompasses within the original orthomosaic.

## Results

We tested the pipeline on a set of 52 overlapping drone images taken from the Bernard Field Station in Claremont, CA on June 30, 2017 (hereinafter referred to as Dataset A) (Table 1). During this time period, only one species of flowering plant is usually in bloom, *Eriogonum fasciculatum* (the California buckwheat). This makes the task of classifying a segmented region a fairly simple binary choice between either "N/A" (not flowering) or "buckwheat." For small tests, a subset (hereinafter referred to as Subset A1) of 11 overlapping images from Dataset A was used instead (Table 1).

| Dataset Name | A | A1 | W | E |
|---|---|---|---|---|
| Number of Images | 52 | 11 | 46 | 68 |
| Capture Date | 6/30/2017 | 6/30/2017 | 6/22/2017 | 6/26/2017 |

| | | | | |
|---|---|---|---|---|
| **Default Strategy Running Time (hrs)** | 29:41:04 | 3:30:04 | 24:23:30 | 22:39:23 |
| **Classification Running Time - Default Strategy (hrs)** | 0:09:54 | 0:04:01 | 0:07:15 | 0:03:19 |
| **Classification Running Time - Experimental Strategy** | 0:00:31 | 0:00:27 | 0:00:28 | 0:00:24 |
| **Default Strategy Max Memory Usage (MB)** | 57745.9 | 23409.18 | 53407.25 | 62364.52 |
| **Classification Max Memory Usage - Default Strategy (MB)** | 8457.35 | 3698.53 | 4449.26 | 2759.16 |
| **Classification Max Memory Usage - Experimental Strategy (MB)** | 309.03 | 311.41 | 314.56 | 308.39 |

**Table 1** The datasets upon which we tested our pipeline and their size and the associated running time and maximum memory usage. The running time and maximum memory usage is also provided for the feature extraction and classification steps. Parallelization of these steps in the experimental strategy substantially reduced their computational load. However, the feature extraction and classification step do not contribute substantially to the total running time and maximum memory usage of the entire pipeline.

To test the running time and memory usage of the pipeline, we also executed it on two other datasets, Dataset W and Dataset E (Table 1). We saw the memory usage and running time of the feature extraction and classification steps decrease substantially after performing parallelization using the experimental strategy (Table 1). However, the overall runtime and maximum memory usage of the pipeline remained much higher (Table 1). Indeed, the steps we parallelized using the experimental strategy contributed less to the overall resource load of the pipeline than the stitching step, which contributed most of the computational load.

## Stitching

Although it is more common to perform stitching through the Metashape GUI, we opted to run Metashape through its Python package in order to automate the stitching step. In order to evaluate whether this had any impact on the quality of the orthomosaics generated, we compared an orthomosaic from Dataset A generated using the automated stitching script against an orthomosaic generated from the Metashape GUI by our lab in the pas (Fig 8). Both orthomosaics

exhibited warped sections where overlapping regions were not properly aligned (Fig 8). In fact, the orthomosaic generated from the automated stitching script demonstrated slightly fewer warped sections than its non-automated counterpart (Fig 8).

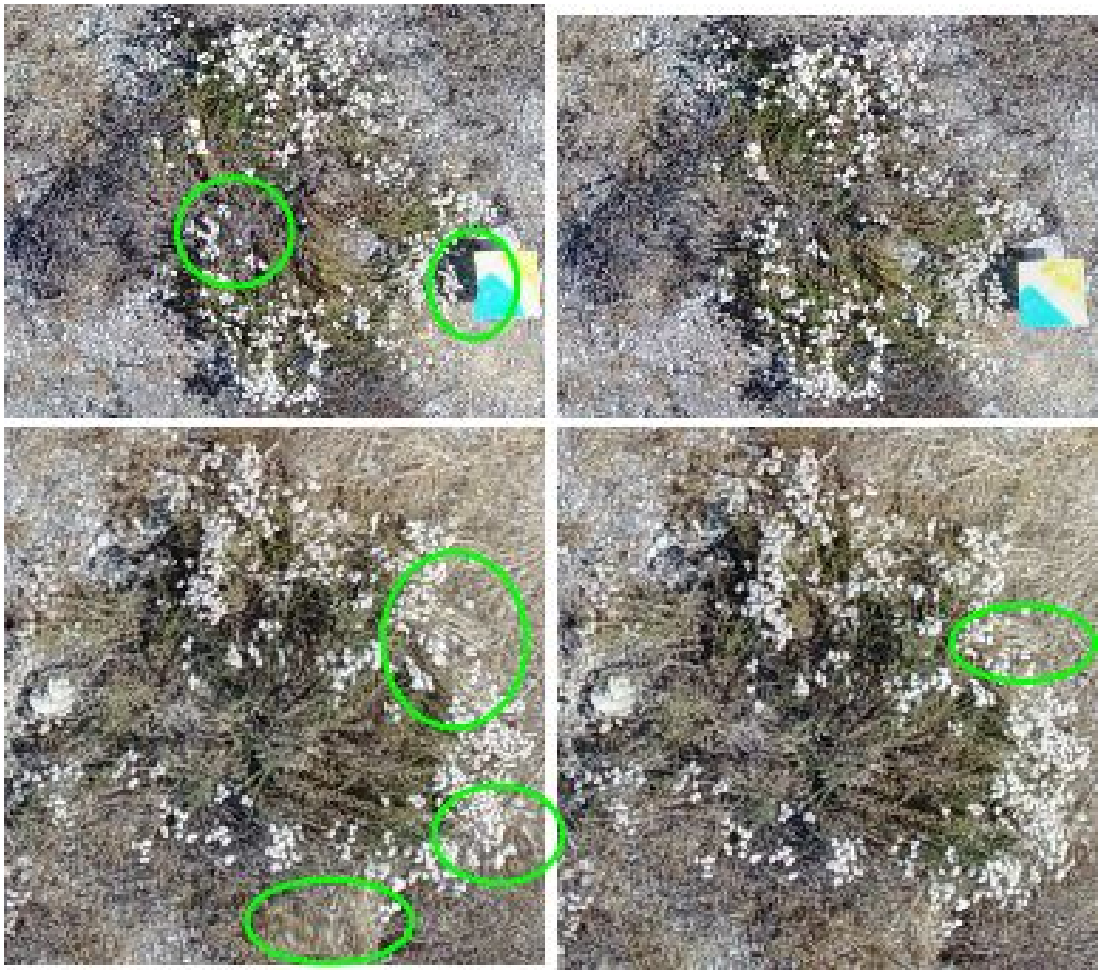
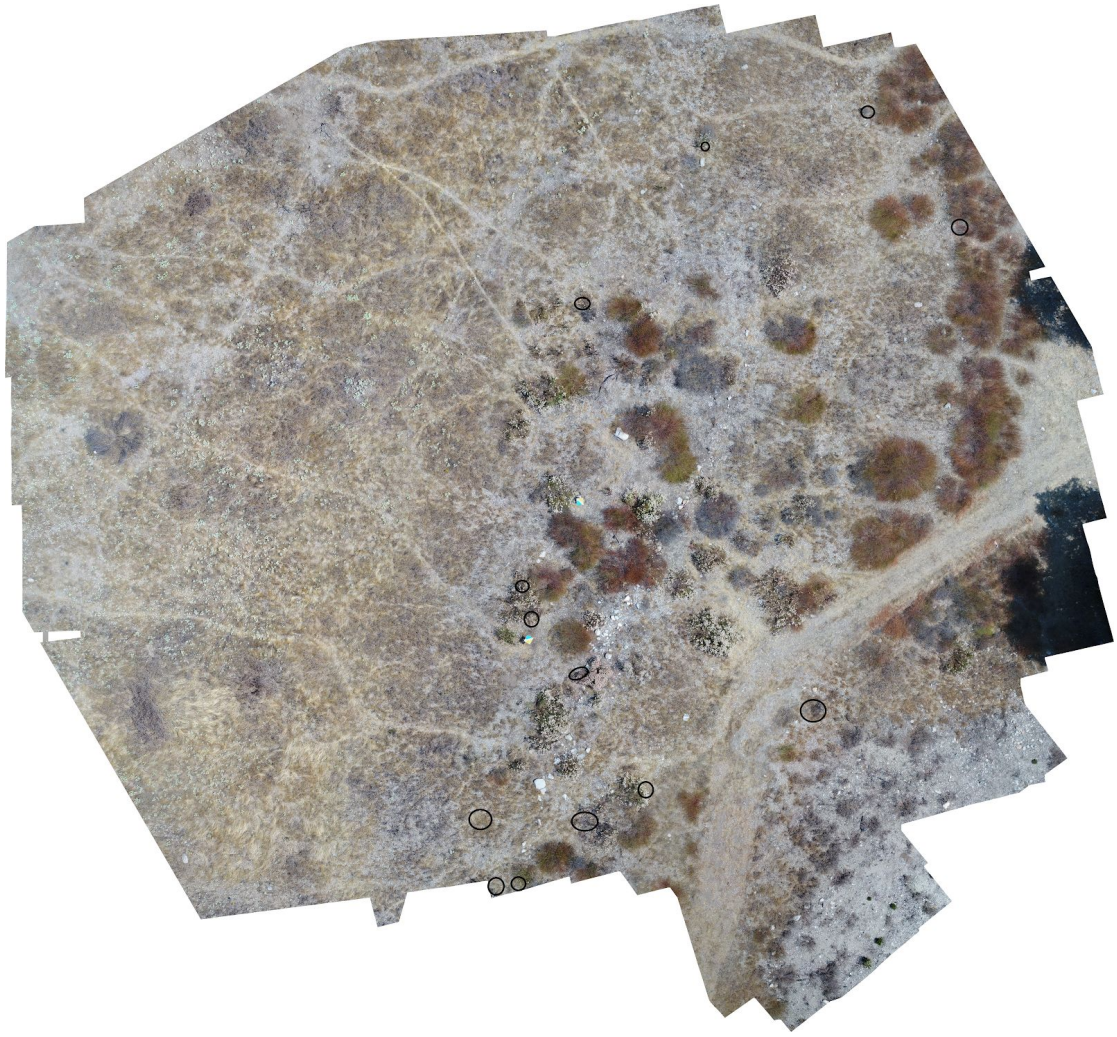


**Fig 8** Two plants as they appear in the orthomosaic created with the GUI (left) vs the automated stitching script (right). The automated stitching script generally led to fewer artifacts.

In general, warped sections are more prominent in sections of the orthomosaic that are covered by fewer overlapping images, such as the fringes of an orthomosaic (as contrasted with its center) (Fig 9). We also tested the automated stitching algorithm on Subset A1, which has fewer overlapping regions owing to being a smaller dataset. The orthomosaic generated from this

smaller subset of the images had warped sections in regions that the larger orthomosaic did not (Fig 9 and Fig 10).



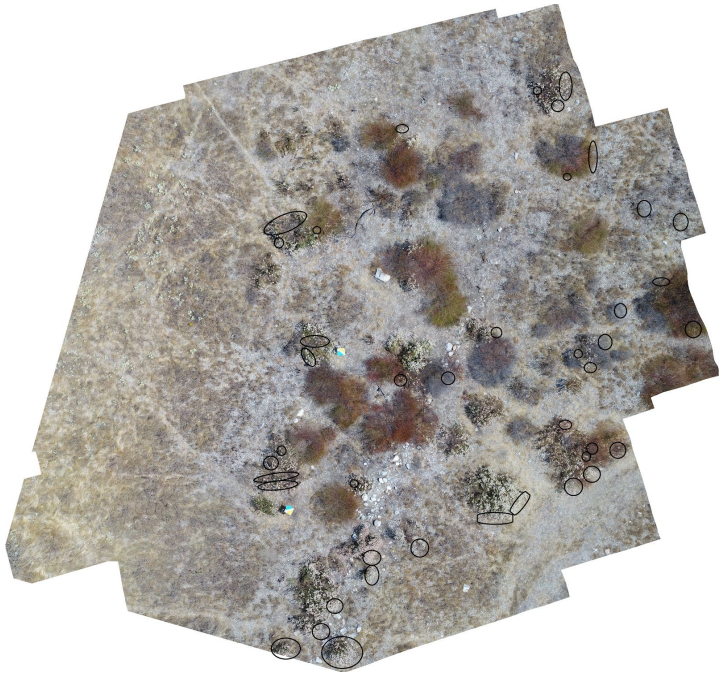


**Fig 9** Orthomosaics from Dataset A (top) and Subset A1 (left), a subset of 11 overlapping images from Dataset A. Warping artifacts (circled in black) are more common in regions with less overlap between images, such as the edges of both orthomosaics or the smaller orthomosaic composed of fewer images.

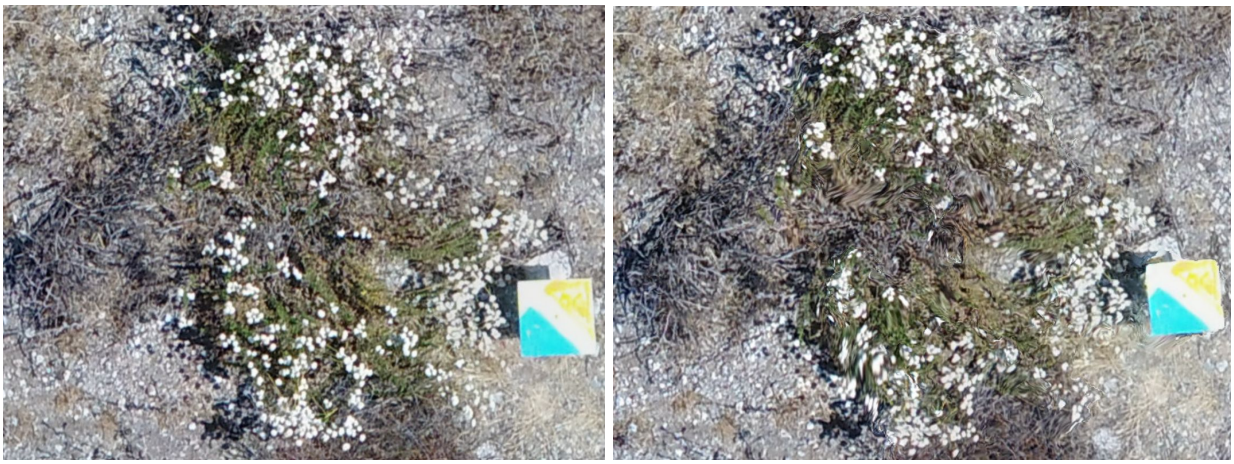


**Fig 10** A plant from Dataset A (left) and Subset A1 (right). Artifacts are more common in the orthomosaic from the smaller subset of images.

## Segmentation

I first tested the segmentation script on a single drone image from Subset A1. Of the 29 plants pictured in the image, six were not recognized by the script and are instead circled manually in blue (Fig 11). Of those six false negatives, four were flowering plants. In addition, there were two segments recognized by the script that did not contain any plants. Both of these

false positives appeared in regions of the image with high contrast due to the presence of white rocks (Fig 11). Among the true positives, many segments only outline part of the plant to which they belong (Fig 11). In general, segments are more likely to contain too little of a plant than too much of the background surrounding the plant (Fig 11).

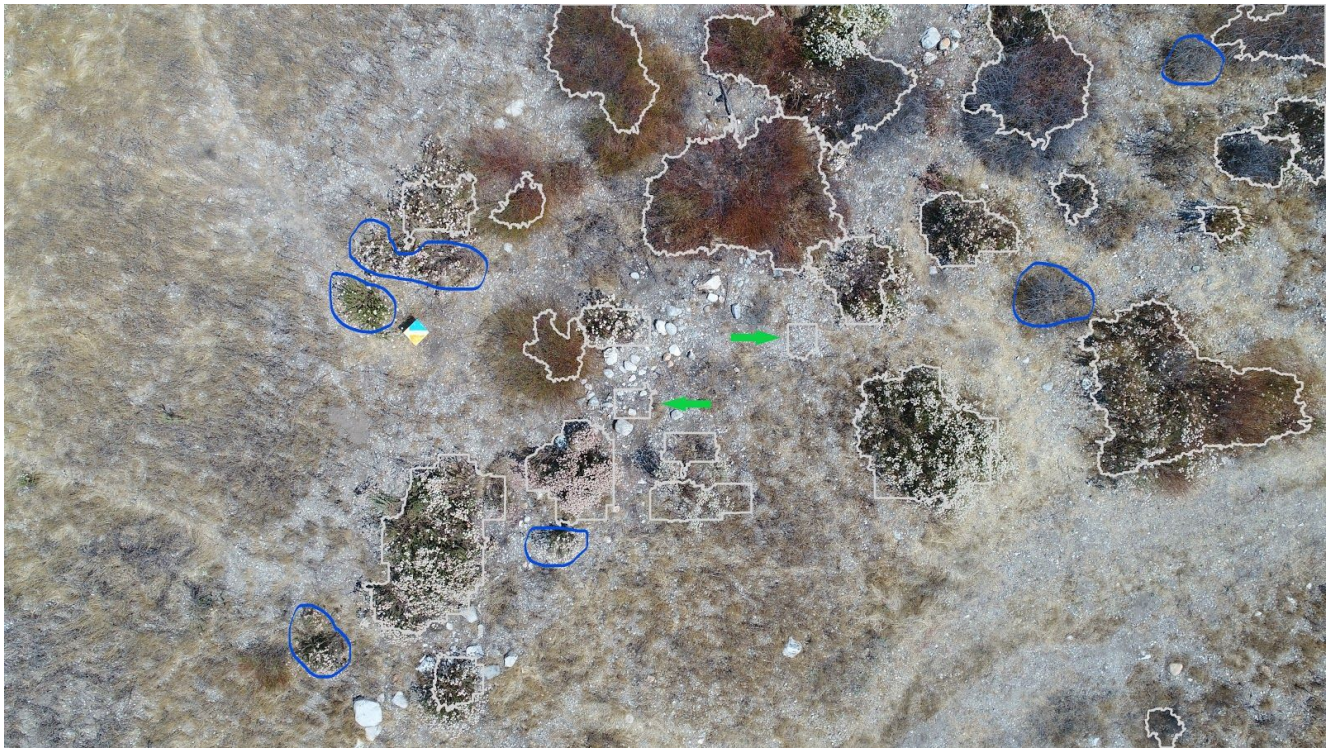


**Fig 11** Segmentation on a single drone image from Subset A1 (in white). Plants that were not segmented are circled in blue. Segmented regions that do not contain plants are indicated by green arrows.

I also tested the segmentation script on the orthomosaics created from Subset A1 and Dataset A (Fig 12). Interestingly, some of the plants were segmented differently in the orthomosaics than in their original images (Fig 11 and Fig 12). For example, there were many plants that were not detected. In addition, the segmented regions from the orthomosaics differed from their counterparts in the original drone image by their shape (Fig 11 and Fig 12). Lastly, segmentation on the orthomosaic from Dataset A seemed to be of roughly equal accuracy as on

the orthomosaic from Subset A1. The segmentation script recognized 26 of the 40 plants in Subset A1 but just 23 of the 40 plants in the same region in Dataset A (Fig 12).

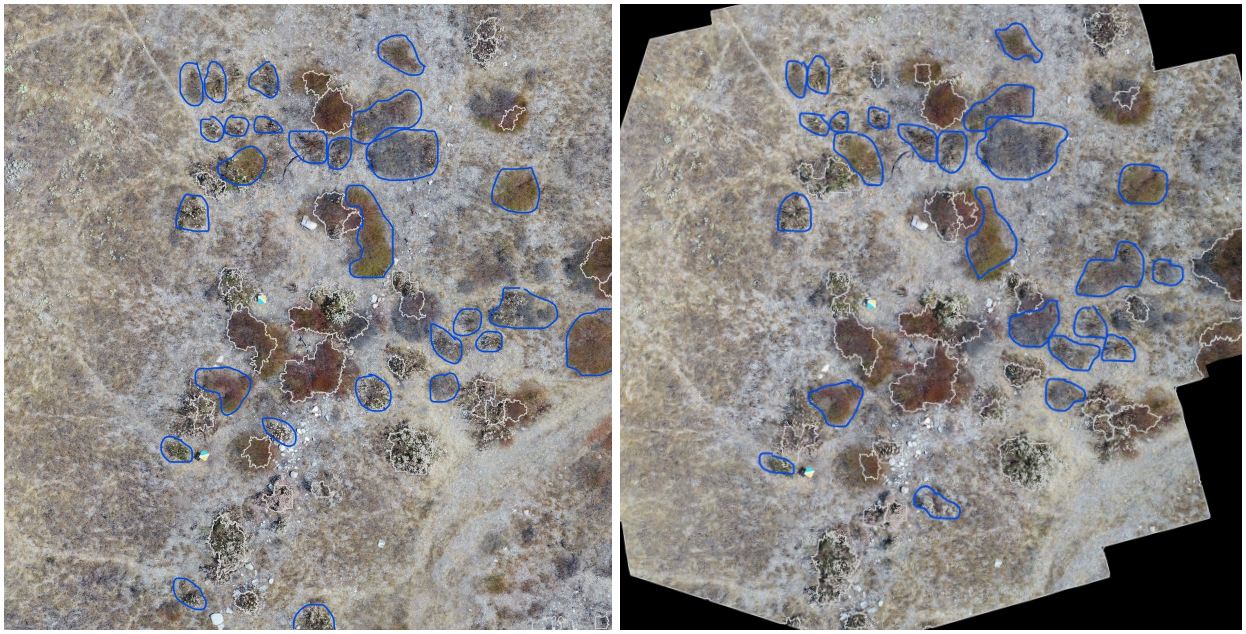


**Fig 12** Segmented regions from Dataset A and Subset A1, a subset of 11 overlapping images in Dataset A. Segmented regions are circled in white. Plants that were not detected are circled in blue. Segmentation was less accurate in the orthomosaics than on the original drone images, but its accuracy only slightly decreased between the orthomosaics from Subset A1 to Dataset A.

## Classification

The classifier was trained and tested on 36 segmented regions from the orthomosaic of Subset A1 (Fig 14). In order to evaluate the accuracy of the classification step irrespective of the quality of the segmented regions from the orthomosaic, these plants were segmented and labeled manually using the LabelMe software tool rather than the automated stitching script (Wada, 2016). We trained the classifier on half of the 36 segmented regions, and tested it on the rest. In order to ensure our evaluation was unbiased, the training and test sets were constructed so that they each contained an equal number of buckwheat and non-flowering plants (Fig 14). This guaranteed that we were not training or unfairly rewarding a classifier that simply tended to assign one label more often than the other.

The classifier was able to label 16 of the 18 segmented regions in the test set correctly (Fig 15 and Table 2). The remaining two regions were false negatives; they were labeled as not flowering when, in fact, they contained buckwheat flowers. Of the two false negatives, one contained very few flowers (Fig 13). The other false negative (hereinafter referred to as Plant X) was a more interesting case, since it was closer to the edge of the orthomosaic and appeared to contain artifacts (Fig 16). We further investigated the effect of these artifacts in the experimental strategy.

**Fig 13** One of only two false negatives that were predicted to be not flowering but had flowers. This plant in particular contains very few buckwheat flowers.

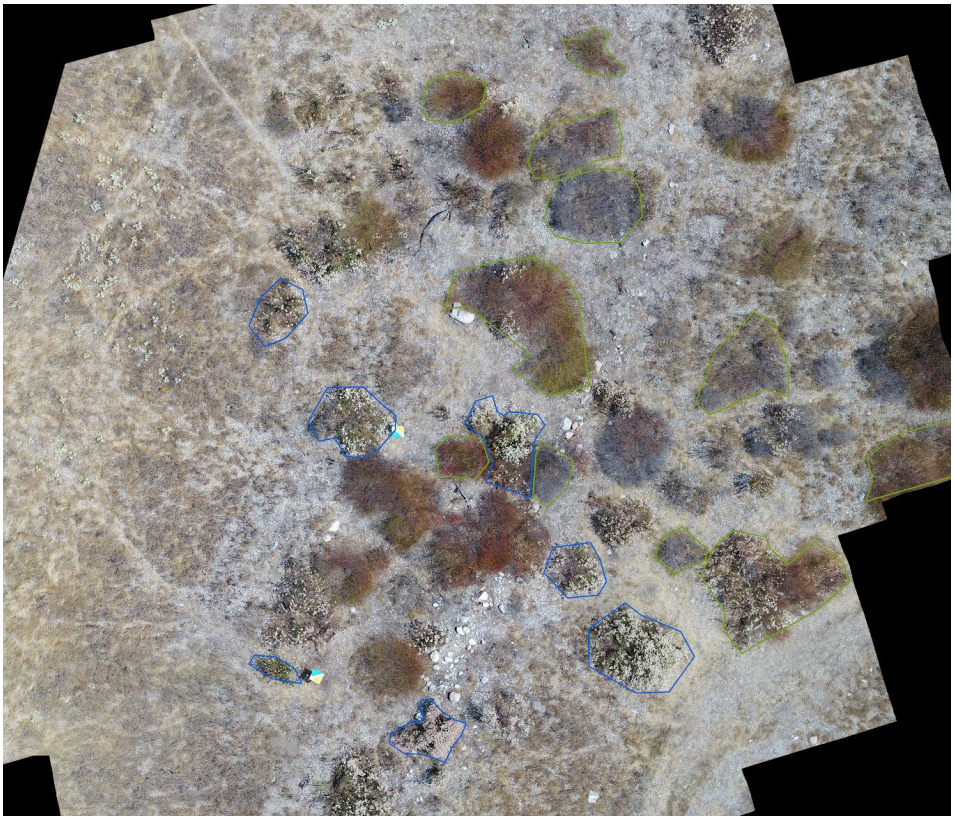


**Fig 14** The predicted species of each plant in the orthomosaic of Subset A1. These segments were manually generated using the LabelMe annotation tool. Blue segments are buckwheat and green segments are non-flowering plants. There were only two false negatives.

## The Experimental Strategy

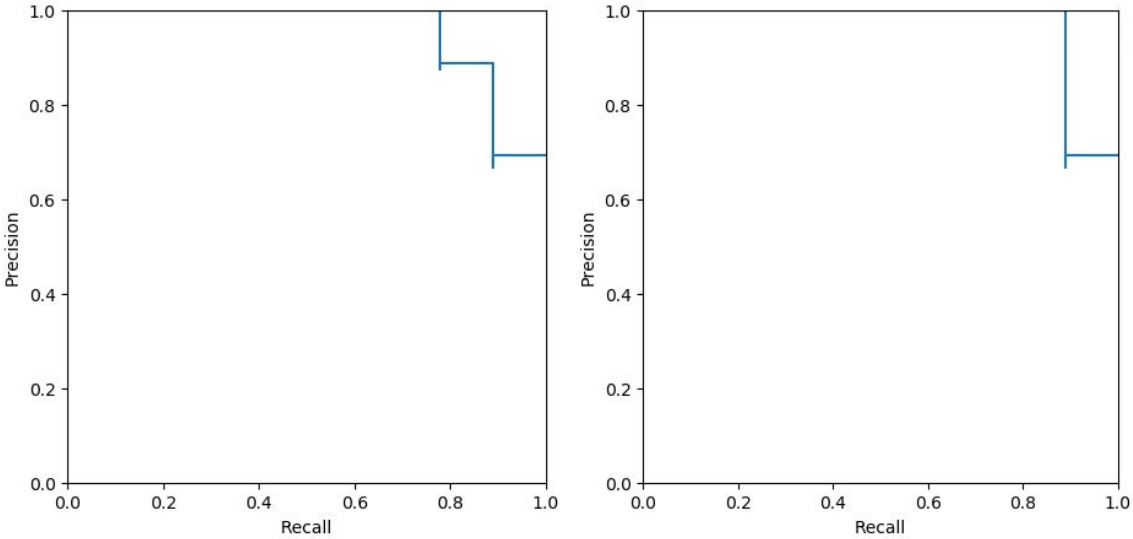


**Fig 15** Precision vs recall curve for 18 manually segmented regions from the orthomosaic for Subset A1 using the default strategy (left) vs the experimental strategy (right).

| | Default Strategy | Experimental Strategy |
|---|---|---|
| **Accuracy** | 0.89 | 0.94 |
| **Recall** | 0.78 | 0.89 |
| **Precision** | 1.00 | 1.00 |
| **F1 Score** | 0.88 | 0.94 |
| **Total Positives** | 7 | 8 |
| **Total Negatives** | 11 | 10 |
| **AUROC** | 0.94 | 0.95 |

**Table 2** Standard metrics for evaluating the performance of our machine learning classifier. The accuracy of our pipeline improved in the experimental strategy, which correctly classified one of the two false negatives from the default strategy. However, the area under the receiver operator characteristic (AUROC) and precision remained relatively unchanged.

We expected that the experimental strategy would be more accurate at classifying the flowering plant species than the default strategy because it runs the classification step on the original drone images rather than the orthomosaic. In order to evaluate the merits of the experimental strategy, we transformed the coordinates of the manually segmented regions from the orthomosaic of Subset A1 back onto the original drone images (Fig 2). These segments were then fed to the classifier, and any conflicts among the resulting labels were resolved according to the methods previously described (Fig 7). The classifications from the experimental strategy were slightly more accurate, with one fewer false negative (Plant X) among the 18 regions than was seen with Strategy 1 (Fig 15 and Table 2). Comparison with the original drone image reveals that Plant X contained warped sections in the orthomosaic (Fig 16). Overall, this increased the accuracy of the pipeline by five percent (Table 2). We also calculated other metrics for assessing the accuracy, including the overall precision and recall (true positive rate), the harmonic mean between precision and recall (known as the F1 score), and the area under the receiver operator

characteristic (AUROC), which is a curve created by plotting the true positive rate over the false positive rate (Table 2).

**Fig 16** Plant X, pictured in the original drone image (left) and the orthomosaic (right). Plant X was a false negative in the default strategy but was correctly classified in the experimental strategy. Warping is prominent in the right half of the orthomosaic segment.

Among the features used by the classifier, contrast proved to be the most important for discerning buckwheat from non-flowering plants (Fig 17). However, the average of each of the red, green, and blue values among all pixels was also ranked with a high importance by the classifier (Fig 17). Color moments, like the HSV skew and standard deviation, which describe the distribution of the colors in the segmented region, were ranked very low and generally weren't interpreted as useful by the random forest (Fig 17).

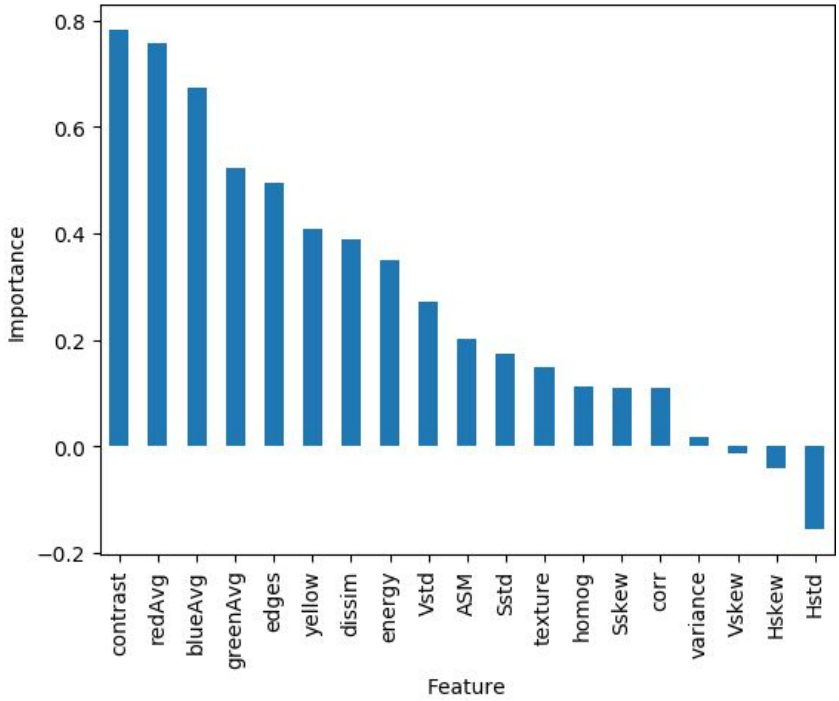



**Fig 17** The importance of each feature used for classifying plants as flowering buckwheat. Importance values were calculated by the random forest classifier. The contrast, a measure of the texture of the plant, proved to be the most important feature.

Interestingly, contrast in Plant X more than doubled (on average among the drone images in which it appeared) in the experimental strategy, whereas many of the other most important features were unchanged (Fig 18).

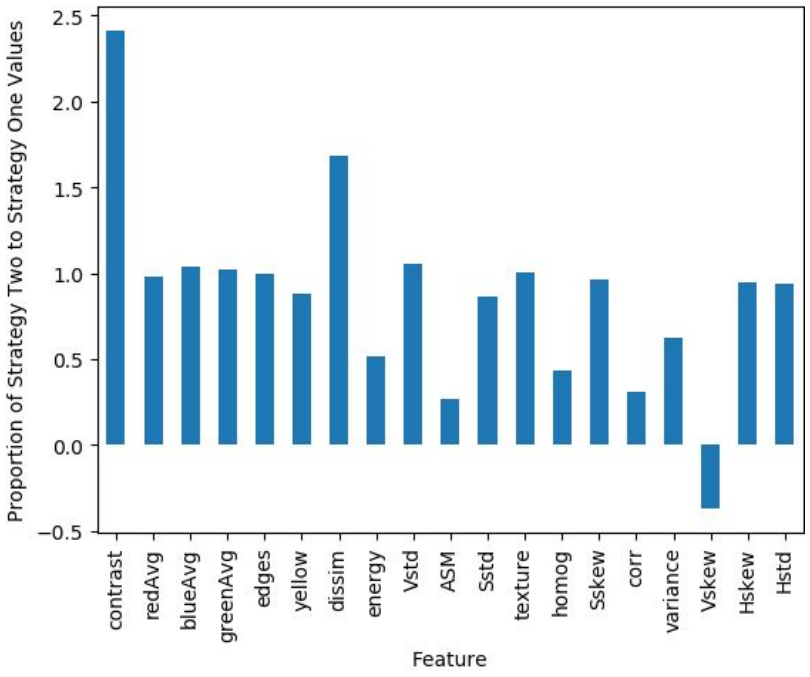


**Fig 18** Comparing the values of each feature in Plant X between the default (Strategy 1) and experimental (Strategy 2) strategies. Averages of the values of each feature were taken among all of the images that contained Plant X in the experimental strategy. The ratio of these averages to their corresponding values in the default strategy is plotted. The contrast of Plant X more than doubled between the default and experimental strategies. This is the largest change among the top five most important features.

## Discussion

Our work demonstrates that it is possible to automatically detect and classify the species of flowering plants in a landscape using drone imagery, despite a range of technical complications. We had reasons to believe, for example, that the default strategy of our pipeline could use an unreasonable amount of memory and require a large execution time. In addition, we suspected that the presence of artifacts in our orthomosaics could decrease the quality of our classifications. For these reasons, we developed a second version of the pipeline, dubbed the "experimental" strategy, that parallelized the classification step and avoided running the classifier on the orthomosaic. Indeed, we saw slight improvements to the number of correctly classified plants in the experimental strategy compared to the default strategy (Fig 15 and Table 2). This was likely due to artifacts in the orthomosaic, which seem to have an effect, albeit an infrequent one. For example, we saw that the single plant that was correctly classified by the experimental strategy but not the default strategy (Plant X) had warped sections that seemed to contribute to its misclassification (Fig 16). It is likely that the decreased classification accuracy is due to a change in contrast from the warping. We have already seen that contrast is one of the most important features for discerning buckwheat from non-flowering plants and it seems likely that warped sections could have much lower contrast. Indeed, the contrast value for Plant X more than doubled in the experimental strategy, the largest change among the top five most important features (Fig 18). While uncommon in all but the fringes of large orthomosaics (Fig 9), the

presence of artifacts *can* have a deleterious effect on the accuracy of the pipeline, and that effect is mainly due to a decrease in the contrast feature value.

Given more time, we would like to further investigate the effect that artifacts can have on the classification step in our method. This would involve testing the pipeline on a larger set of plants. The small size of our test set is a major barrier to being more confident that the two strategies consistently differ in accuracy as much as we observed them to do in Subset A1. Testing on a larger set of plants would also allow us to further confirm the role of artifacts in misclassifying plants in the default strategy.

We would also like to test the classifier on other flowerings plants besides the California buckwheat. At the moment, the pipeline has been quite successful at classifying buckwheat plants, but it is possible that the features that were successful for classifying buckwheat may be less important for other flowering plants. For example, we suspect that contrast was an important feature for classifying segmented regions with buckwheat because the white color of the buckwheat flower has a high contrast with the dark green vegetation from the rest of the plant (Fig 17). However, contrast may not be as important for flowers with differently colored flowers. One way of testing this hypothesis would be to train and test the pipeline on drone imagery from other seasons. June, the season that our data came from, is usually too dry for many flowering plants. However, earlier months in the year tend to have flowers with different colors, like *Penstemon spectabilis* (the showy penstemon), *Eriodictyon trichocalyx* (yerba santa), and *Salvia apiana* (the bee sage). The showy penstemon is dark green with purple flowers. The yerba santa is also dark green but has white flowers. The bee sage is a silver color and has pale, pink flowers.

By testing the classifier on other seasons, we could develop more features that could improve the applicability of the classifier for different plants.

It is also conceivable that the observed artifacts could have an effect on the segmentation step. However, our current evidence for this hypothesis is mixed. We saw, for example, that the segmentation script more accurately identified plants in one of the original drone images than in the orthomosaic. This would seem to imply that artifacts in the orthomosaic negatively affect the performance of the segmentation script. However, if this were true, we would have seen that the number of plants identified by the segmentation script in Subset A1 would increase for the same region in Dataset A, since Dataset A tended to have far fewer artifacts than Subset A1 (Fig 9). Interestingly, this was not the case (Fig 12).

We could more thoroughly test whether artifacts can affect the segmentation script by creating a new version of the experimental strategy which performs segmentation on each original drone image rather than the orthomosaic. However, this would require that we merge overlapping segmented regions from different images that belong to the same plant, just as we had to do with the RF classification probabilities in the experimental strategy. This may be difficult to achieve without also unintentionally merging adjacent (yet distinct) plants. However, it could help us further reduce the memory usage and running time of our pipeline by letting us parallelize the segmentation step. The merits of attempting this approach will need to be weighed against knowledge of the total running time and maximum memory usage of the pipeline, which remains high mostly because of the stitching step rather than the segmentation step.

An alternative might be to create a new version of the segmentation script which incorporates the use of features that are not affected by the artifacts to begin with. These could be

extracted from the Metashape project file, for example. The most promising Metashape feature for improving the segmentation accuracy is a metric called *elevation*, which represents Metashape's estimate for each pixel's height above the ground. We would expect that elevation values would be higher for objects within the orthomosaic, like plants or rocks, than for non-objects like dirt or grass. Thus, a new version of the segmentation script could threshold on the elevation values much like the current segmentation script did for the green and contrast features.

One potential problem with the segmentation script as it is currently implemented is that it may be depending on the contrast feature value more than is appropriate. We saw that this led to the misidentification of two segmented regions that were not actually plants. Thus, one way to improve the segmentation accuracy might be to combine the green and contrast values of each pixel in a different way prior to performing thresholding in the segmentation script. For example, we could take a weighted average of the feature values so that we could control the effect of each feature on the result by setting the weights to favor green values over contrast values.

The species maps created by our pipeline can be used to understand how the landscape around a honey bee colony may affect its foraging behavior and its survival. This could have many applications for future work. One question we are interested in investigating is whether the species of flowering plants around a focal plant might affect the rate at which bees visit the focal plant. Honey bee scouts recruit other individuals in the hive to pollen-rich areas using a special "waggle" dance (Okada et al., 2014). However, recruitment of individuals from the rest of the hive is prone to some error and variation, particularly in the length of the waggle (Okada et al., 2014; Tanner & Visscher, 2010). These errors can actually be beneficial, as they can help bees

find new, pollen-rich areas (Okada et al., 2014). However, to account for this error, we hypothesize that a honey bee scout will tend to recruit the rest of her hive to buckwheat plants that have other buckwheat in close proximity rather than those surrounded by non-flowering plants. We could test this hypothesis by calculating a score for each buckwheat plant which indicates its proximity to other buckwheat plants. The maps output by our pipeline will be useful here because they can give us the locations of other buckwheat plants in a single kilometer radius from the focal plant. The score can be calculated as the proportion of buckwheat plants to other plants in the area around the focal plant. We plan to test the relevance of this score in a generalized linear model (GLM) that predicts observed counts of bee visitation rates. The use of a GLM would allow us to also account for other factors that we know from previous experiments also affect honey bee visitation rates, including the size of the plant and the number of flowers. Honey bees also tend to visit some species of plants more than others (Dobson, 1987; Russell et al., 2016). We would like to also test whether the diversity of the other species of plants surrounding a buckwheat plant may affect its visitation rate by honey bees. To do this, we could calculate a local diversity score for each plant which represents the heterogeneity of the plant species in its vicinity. We hypothesize that plants with high local diversity scores will be less likely to be visited by bees. This hypothesis could also be tested using the same GLM we mentioned in the prior paragraph, so that we can also take into account the abundance of flowers in addition to the diversity when predicting bee visitation rates.

To test whether honey bee colony survival is improved for regions that have a diversity of flowering plant species, we could calculate a similar diversity score across all plants in a 10 km radius from a hive. This would be similar to the diversity score calculated previously, except

that we would use the hive as the focal plant. Given the current running time and memory usage of our pipeline, this would require extensive drone imagery and a very robust computing environment. We could use online databases of beekeeper surveys of colony survival rates (like the USDA's National Honey Bee Survey) and UAV flight imagery. We could also calculate other metrics that are speculated to affect honey bee foraging efficiency and colony survival. For example, we could include per species and overall *patchiness*, a measure of how likely flowers are to appear in small, isolated clumps (Donaldson-Matasci & Dornhaus, 2012), the heterogeneity of patch quality (Beekman & Lew, 2008), or the diversity of plant species within patches (Blaauw & Isaacs, 2014). With enough colonies, we could test these metrics for predicting the time before managed honey bee colonies die using another GLM. We believe our method has the potential to be crucial for studying honey bee survival and finally reversing the disturbing decline in the number of colonies of this incredibly important organism.

## Literature Cited

Barnhart, I., Moro Rosso, L. H., Secchi, M. A., & Ciampitti, I. A. (2019). Evaluating Sorghum Senescence Patterns Using Small Unmanned Aerial Vehicles and Multispectral Imaging. *Kansas Agricultural Experiment Station Research Reports*, *5*(6). https://doi.org/10.4148/2378-5977.7799

Beekman, M., & Lew, J. B. (2008). Foraging in honeybees—When does it pay to dance? *Behavioral Ecology*, *19*(2), 255–261. https://doi.org/10.1093/beheco/arm117

Blaauw, B. R., & Isaacs, R. (2014). Larger patches of diverse floral resources increase insect pollinator density, diversity, and their pollination of native wildflowers. *Basic and Applied Ecology*, *15*(8), 701–711. https://doi.org/10.1016/j.baae.2014.10.001

Blaschke, T. (2010). Object based image analysis for remote sensing. *ISPRS Journal of Photogrammetry and Remote Sensing*, *65*(1), 2–16. https://doi.org/10.1016/j.isprsjprs.2009.06.004

Chisel, J. (2015). Honey Bees' Impact on the U.S. Economy. *Economics Theses*. https://soundideas.pugetsound.edu/economics_theses/98

Dobson, H. E. M. (1987). Role of flower and pollen aromas in host-plant recognition by solitary bees. *Oecologia*, *72*(4), 618–623. https://doi.org/10.1007/BF00378991

Donaldson-Matasci, M. C., & Dornhaus, A. (2012). How habitat affects the benefits of communication in collectively foraging honey bees. *Behavioral Ecology and Sociobiology*, *66*(4), 583–592. https://doi.org/10.1007/s00265-011-1306-z

Garibaldi, L. A., Aizen, M. A., Cunningham, S., & Klein, A. M. (2009). Pollinator shortage and global crop yield. *Communicative & Integrative Biology*, *2*(1), 37–39.

https://doi.org/10.4161/cib.2.1.7425

Gross, J. W., & Heumann, B. W. (2016). A Statistical Examination of Image Stitching Software Packages For Use With Unmanned Aerial Systems. *Photogrammetric Engineering & Remote Sensing*, *82*(6), 419–425. https://doi.org/10.14358/PERS.82.6.419

Hofmann, S., Everaars, J., Schweiger, O., Frenzel, M., Bannehr, L., & Cord, A. F. (2017). Modelling patterns of pollinator species richness and diversity using satellite image texture. *PLOS ONE*, *12*(10), e0185591. https://doi.org/10.1371/journal.pone.0185591

Kluser, S., Neumann, P., Chauzat, M.-P., Pettis, J. S., Peduzzi, P., Witt, R., Fernandez, N., & Theuri, M. (2010). *Global honey bee colony disorders and other threats to insect pollinators*. https://archive-ouverte.unige.ch/unige:32251

Lu, B., & He, Y. (2017). Species classification using Unmanned Aerial Vehicle (UAV)-acquired high spatial resolution imagery in a heterogeneous grassland. *ISPRS Journal of Photogrammetry and Remote Sensing*, *128*, 73–85. https://doi.org/10.1016/j.isprsjprs.2017.03.011

Nicholls, C. I., & Altieri, M. A. (2013). Plant biodiversity enhances bees and other insect pollinators in agroecosystems. A review. *Agronomy for Sustainable Development*, *33*(2), 257–274. https://doi.org/10.1007/s13593-012-0092-y

Okada, R., Ikeno, H., Kimura, T., Ohashi, M., Aonuma, H., & Ito, E. (2014). Error in the Honeybee Waggle Dance Improves Foraging Flexibility. *Scientific Reports*, *4*(1), 1–9. https://doi.org/10.1038/srep04175

Pflanz, M., Schirrmann, M., & Nordmeyer, H. (2018). Drone based weed monitoring with an image feature classifier. *Julius-Kühn-Archiv*, *No.458*, 379–384.

*PhotoScan/Metashape Professional*. (2019). Agisoft. http://www.agisoft.com/downloads/installer/

Russell, A. L., Golden, R. E., Leonard, A. S., & Papaj, D. R. (2016). Bees learn preferences for plant species that offer only pollen as a reward. *Behavioral Ecology*, *27*(3), 731–740. https://doi.org/10.1093/beheco/arv213

Singh, M., Evans, D., Tan, B. S., & Nin, C. S. (2015). Mapping and Characterizing Selected Canopy Tree Species at the Angkor World Heritage Site in Cambodia Using Aerial Data. *PLOS ONE*, *10*(4), 26. https://doi.org/10.1371/journal.pone.0121558

Tanner, D. A., & Visscher, P. K. (2010). Does Imprecision in The Waggle Dance Fit Patterns Predicted by The Tuned-Error Hypothesis? *Journal of Insect Behavior*, *23*(3), 180–188. https://doi.org/10.1007/s10905-010-9204-1

Von Frisch, K., & Chadwick, L. E. (1967). *The dance language and orientation of bees* (Vol. 1). Belknap Press of Harvard University Press Cambridge, MA.

Wada, K. (2016). *labelme: Image Polygonal Annotation with Python*. https://github.com/wkentaro/labelme

Yu, Q., Gong, P., Clinton, N., Biging, G., Kelly, M., & Schirokauer, D. (2006). Object-based Detailed Vegetation Classification with Airborne High Spatial Resolution Remote Sensing Imagery. *Photogrammetric Engineering & Remote Sensing*, *72*(7), 799–811. https://doi.org/10.14358/PERS.72.7.799