Learning to Count Leaves in Rosette Plants

Mario Valerio Giuffrida[1]
valerio.giuffrida@imtlucca.it
Massimo Minervini[1]
massimo.minervini@imtlucca.it
Sotirios A. Tsaftaris[1,2]
stsaft@gmail.com

[1] Pattern Recognition and Image Analysis (PRIAn)
IMT Institute for Advanced Studies
Lucca, Italy
http://prian.imtlucca.it/

[2] Institute for Digital Communications
School of Engineering
University of Edinburgh
Edinburgh, UK

Abstract

Counting the number of leaves in plants is important for plant phenotyping, since it can be used to assess plant growth stages. We propose a learning-based approach for counting leaves in rosette (model) plants. We relate image-based descriptors learned in an unsupervised fashion to leaf counts using a supervised regression model. To take advantage of the circular and coplanar arrangement of leaves and also to introduce scale and rotation invariance, we learn features in a log-polar representation. Image patches extracted in this log-polar domain are provided to K-means, which builds a codebook in a unsupervised manner. Feature codes are obtained by projecting patches on the codebook using the triangle encoding, introducing both sparsity and specifically designed representation. A global, per-plant image descriptor is obtained by pooling local features in specific regions of the image. Finally, we provide the global descriptors to a support vector regression framework to estimate the number of leaves in a plant. We evaluate our method on datasets of the *Leaf Counting Challenge* (LCC), containing images of Arabidopsis and tobacco plants. Experimental results show that on average we reduce absolute counting error by 40% w.r.t. the winner of the 2014 edition of the challenge –a counting via segmentation method. When compared to state-of-the-art density-based approaches to counting, on Arabidopsis image data $\sim$75% less counting errors are observed. Our findings suggest that it is possible to treat leaf counting as a regression problem, requiring as input only the total leaf count per training image.

1 Introduction

Morphological plant traits such as size, number of leaves, biomass, and shape are influenced not only by the genes, but also by external environmental factors. However, the interaction between genes and environmental (and growth) conditions leads to a combinatorial explosion of possible phenotypes, making the understanding of the link between genotype and phenotype a complex task [18]. Measuring a plant's visual traits manually is costly and requires specialized investigations to carry on the analysis. High throughput plant phenotyping allows large-scale plant analysis [13, 17, 39], in the hope of reducing the bottleneck

Pages 1.1-1.13
https://dx.doi.org/10.5244/C.29.CVPPP.1

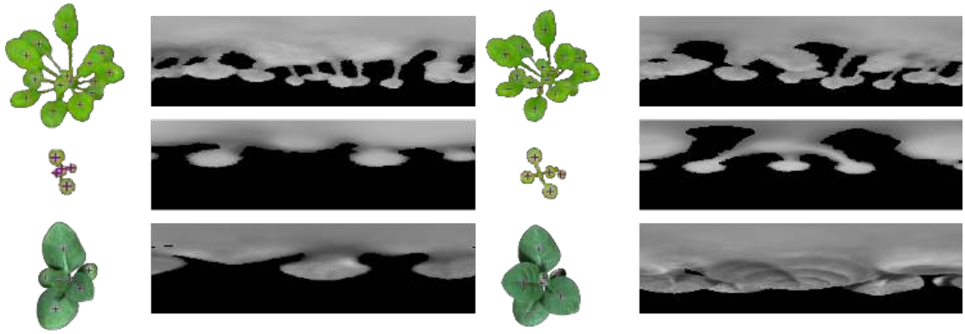


Figure 1: Example images (background was removed) of Arabidopsis taken from the A1 (top), and A2 (middle), datasets respectively, and tobacco taken from A3 (bottom). First and third columns show the same plant (with leaf center annotations in purple) few days after. Second and fourth columns show the corresponding log-polar representations.

in matching phenotype to genotype [16]. Automated systems have to incorporate reliable computer vision techniques to analyze the tremendous amount of data, coming from many plant specimens involved in typical phenotyping experiments [31, 35].

From a phenotyping point of view, the number of leaves in a plant is related to e.g. developmental stage [36], growth regulation [6, 38], flowering time [21], and yield potential. However, counting leaves automatically is a known challenging task [27], due to a plant's rapid exponential growth and complexity. Figure 1 shows example images of rosette plants, i.e., Arabidopsis (top two rows) and young tobacco (bottom row), shown in two different time points of development. It is readily evident how changes in scale, rotation, and appearance may challenge state-of-the-art vision-based counting approaches. Even within a plant (as evident also in the images of Figure 1), leaves vary in size and shape and move around the plant's center, thus appearing rotated when imaged over time. Furthermore, leaves may overlap each other, resulting in major occlusions, which render the counting task challenging even for a human expert.

In this paper, we aim to count the number of leaves in rosette plants (e.g., Arabidopsis and young tobacco, see Figure 1) based on top-view images. We adopt a counting by regression approach through Support Vector Regression (SVR) [11]. Patches extracted from the log-polar domain [2] are used to learn a dictionary in an unsupervised fashion. Local features are pooled together in specific regions of the image to build a global descriptor. At test time given an input image we extract features (projecting on the dictionary), pool responses, and use the learned regressor to estimate the number of leaves.

We test our approach on image data of Arabidopsis (A1, A2), a known model plant [28], and tobacco (A3) [32], in the context of the *Leaf Counting Challenge* (LCC), held in conjunction with the *Computer Vision Problems in Plant Phenotyping* (CVPPP 2015) workshop.[1] Image data provided to challenge participants are accompanied by leaf center annotations for training images and plant segmentation masks for both training and testing images. Experimental results show that our method outperforms the counting-via-segmentation approach in [30] for datasets A1 and A3, where more training data is available. On testing data, we predict the correct number of leaves in 25% of cases and in 57% of the cases the error is at most ±1 leaf. We also compare with two methods from the broad computer vision literature that aim to count objects by learning densities.

[1] http://www.plant-phenotyping.org/CVPPP2015

The contributions of this paper are multi-fold. First, it is the first paper to tackle leaf counting in a learning framework. Second, operating in the log-polar coordinate system (see Figure 1) permits us to learn a rotation and scale invariant dictionary in an unsupervised fashion. Third, to learn better features we do not use all possible image patches, but we identify regions of interest based on a texture heuristic. Finally, we selectively create pooling regions to obtain descriptors invariant to small local transformation, aiming to learn a regressor with better generalization capabilities.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 presents the proposed approach, while Section 4 discusses experimental results. Section 5 offers concluding remarks.

2 Related Work

The literature of automated methods for counting leaves is limited to counting via leaf segmentation approaches [19, 30]. Specifically, *Pape and Klukas* [30] faced the problem of leaf segmentation for the 2014 edition of the CVPPP workshop. After a coarse leaf segmentation, lines separating overlapping leaves are determined based on split points. In [42], leaf segmentation and tracking is performed in a fluorescence video sequence of growing Arabidopsis. A set of leaf candidates is generated in a frame based on Chamfer matching, and each leaf is tracked in the following frames assuming temporal coherence.

In the broad computer vision literature several approaches have been proposed to address the problem of counting objects within a scene. A first class of approaches is the counting-by-detection methods [41], which formulate the problem as a detection task. Typical solutions rely on local features, such as histogram of oriented gradients (HOG) [9, 12], local binary patters [8], or shape [22]. Nevertheless, leaf detection is a challenging task, since leaf surface is almost featureless and shape information is unreliable under heavy occlusion, as it can be seen in Figure 1.

Recently, several methods aiming to estimate the density of objects within a scene have been proposed to address counting applications. *Lempitsky and Zisserman* [23] minimize a loss function based on the Maximum Excess over SubArrays (MESA) distance. Similarly, in [4] density is predicted by per-pixel ridge regression. In [15] random forest regression is used to estimate density. However, density estimation approaches are challenged by objects appearing at different scale or overlapping (occlusions). The counting of overlapping objects is addressed in [3], even though varying object size remains an open issue.

On the other hand, global regression approaches aim to learn a global image representation to relate it via regression to total object count within a scene. For example, *Wang et al.* [40] adopt support vector regression to predict the number of pedestrians in a video frame. After a coarse foreground segmentation, HOG features are extracted and provided to SVR. While such approaches could address varying object size and occlusion, spatial information on object layout cannot be retrieved (which is available with the methods discussed before).

3 Proposed Method

Here we describe a global regression method to count leaves in rosette plants, hereafter referred to as *General Leaf Counting* (GLC). We use as input greyscale images I_j, $\forall j = 1,\ldots,N$, showing a top-view on individual rosette plants. Following the design and require-

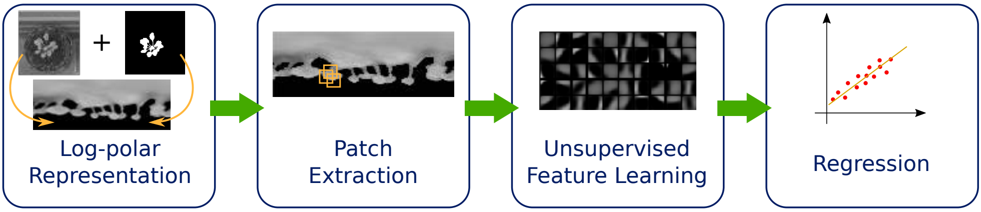
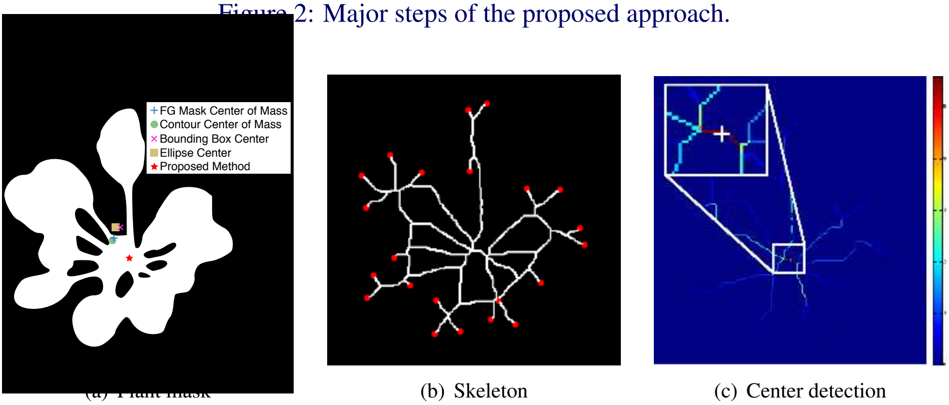


Figure 2: Major steps of the proposed approach.



(a) Plant mask (b) Skeleton (c) Center detection

Figure 3: Center detection in a complex object. Shown are: (a) plant mask available from expert annotation (together with classical calculations of a center and proposed); (b) skeleton obtained from (a) and endpoints as red dots; and (c) most traversed segment, with detected center marked with a white cross. Part (a) shows that finding the center of mass, as done in [1] for plants, the result is unreliable. Other approaches suffer the same shortcomings too. Our approach performed the best since it takes into account a plant's complex structure.

ments of the challenge and of the CVPPP 2015 dataset (cf. Section 1), we assume as given expert annotations per each training image: the (i) total number of leaves, and (ii) foreground segmentation mask providing the location of plant pixels (see Figure 3(a)). For a testing image a foreground mask is also given, so in this work we do not address the problem of plant segmentation from background.

As Figure 2 illustrates our first step exploits the circular arrangement of leaves by converting the image into the log-polar domain. We then learn a suitable feature representation from the data by training a dictionary in an unsupervised fashion on image patches extracted from informative regions. A local descriptor for each patch is computed using the learned dictionary, employing the *triangle encoding* [10]. By max-pooling we combine such feature vectors to obtain a global image descriptor, which we use in a regression framework to predict the number of leaves. Each step is detailed in the following.

3.1 Log-polar Representation

Rosette plants are characterized by a radial arrangement of leaves around the center of the plant (i.e., the stem). In order to exploit this structure, we convert an input image I (the subscript j is omitted for brevity) into the log-polar domain, obtaining a new image denoted as $\tilde{I}$. This conversion not only orients leaves w.r.t. the plant center to appear parallel, but also ensures the same sampling and final dimensions of $\tilde{I}$ for any plant size, accounting for the problem of extracting good descriptors in the presence of large size variability within a

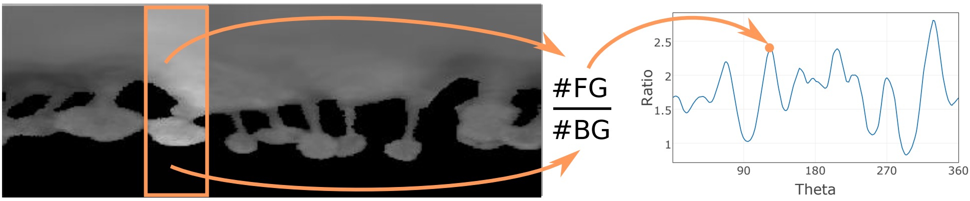


Figure 4: FG/BG ratio: a sliding window moves rightwards to compute the ratio between the number of foreground and background (black) pixels within it. Observe that we have local maxima where leaves are represent even when leaves are overlapping.

training set. The log-polar transformation maps points from the Cartesian (x, y) coordinate system to the log-polar (ρ, θ) coordinate system [2].

Prior to the transformation, we move the origin to the center of the plant. Since finding the center of a mass is unreliable in a complex object (see also Figure 3(a)), here we estimate the position of a plant's center based on the skeleton obtained from the segmentation mask (Figure 3(a)) of I given as input. From the skeleton, we detect the endpoints (plotted in red in Figure 3(b)) and we compute shortest paths along the skeleton connecting each endpoint to all other ones. Aggregating all shortest paths, we identify the segment that is traversed more frequently. We select the center of the region containing this segment as the new origin (x_0, y_0) (Figure 3(c)).

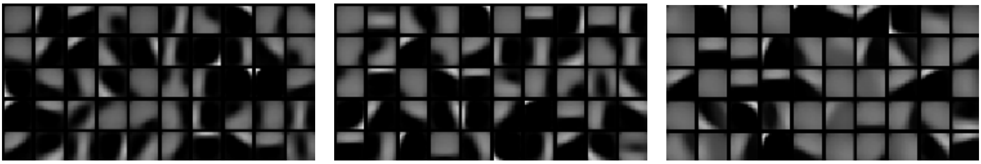
Coordinates in the log-polar domain are then calculated for each point (x, y) as the logarithm of the radius $\rho = \log\sqrt{(x-x_0)^2+(y-y_0)^2}$ and azimuth $\theta = \text{atan2}(y-y_0, x-x_0)$. We sample with increments of 1° in the angular coordinate θ, thus the transformed image $\tilde{I}$ is 360 pixels wide, while the radius is adaptively chosen by computing the distance between a plant's center and the farthest point in the segmentation mask. (Fixed zero padding is added in the lower part of the log-polar image to facilitate the patch extraction step.)

3.2 Patch Extraction

To learn a dictionary, instead of extracting densely all possible patches from $\tilde{I}$, we focus on regions that are most informative from a leaf counting perspective. We identify such regions based on the *FG/BG ratio curve*, i.e., the ratio between the number of foreground (FG) pixels and the number of background (BG) pixels. Using a sliding window as high as $\tilde{I}$ and of fixed width W, we scan $\tilde{I}$ to compute the FG/BG ratio (Figure 4). The ratio between foreground and background pixels will have high value wherever plant pixels are dominant, even when leaves are overlapping. We detect local maxima in the so-obtained curve, and use the corresponding (column) locations to define in $\tilde{I}$ regions of interest of width W' centered on the maxima. From these regions (which may also overlap), we extract $S \times S$ sized patches densely, discarding duplicated patches or patches falling entirely within background. The patches are then normalized by the L_2 norm to reduce photometric variability. We denote the vectorized patches extracted from a log-polar representation $\tilde{I}$ as $\mathbf{p}_i$ of dimension $S^2 \times 1$, where $i = 1, \ldots, P$.

3.3 Unsupervised Feature Learning

We learn from the data, features tailored to our application in an unsupervised learning fashion, using the patches extracted at the previous step. The patches extracted from available



(a) Arabidopsis, A1 (b) Arabidopsis, A2 (c) Tobacco, A3

Figure 5: Features learned with $K = 50$ from patches obtained within the plant(s).

training images are clustered via K-means [14, 37] to learn a representative dictionary (codebook). K-means is an unsupervised learning algorithm that is able to partition the feature space into K clusters, providing also a set of K cluster representatives $\mathbf{c}_k$, so-called *centroids*, examples of which are shown in Figure 5.

All patches $\mathbf{p}_i$ in $\tilde{I}$ are represented by a new vector $\mathbf{z}_i$ using the *triangle encoding* [10]. We determine the distance $\delta_i^{(k)} = \|\mathbf{p}_i - \mathbf{c}_k\|_2$ to the k-th centroid $\mathbf{c}_k$. Let $\bar{\delta}_i$ be the average distance between $\mathbf{p}_i$ and each of the K centroids. The triangle encoding is computed as:

$$z_i^{(k)} = \max\left\{0, \bar{\delta}_i - \delta_i^{(k)}\right\}, \tag{1}$$

where the new vector $\mathbf{z}_i$ has K dimensions. According to a recent study on unsupervised single-layer feature learning this encoding outperforms classical one-hot encoding [10].

3.4 Regression

When all vectors $\mathbf{z}_i$ are determined in an image, we use max-pooling to compute a global descriptor which reduces the size of the descriptor and also adds invariance to small local transformations [5, 20]. We partition the log-polar image $\tilde{I}$ into T non-overlapping equally sized regions ω_t, $t = 1, \ldots, T$. Each pooling region ω_t has the same height as $\tilde{I}$ and is $D = 360/T$ pixels wide. For a region ω_t we build the max-pooling vector $\boldsymbol{\zeta}_t$, whose k-th element is obtained as $\zeta_t^{(k)} = \max_{\mathbf{z}_i \in \omega_t} z_i^{(k)}$. Finally, we obtain the global descriptor for I_j by concatenating all the corresponding $\boldsymbol{\zeta}_t$ in a new vector $\mathbf{x}_j$.

Based on the observations $\mathbf{x}_j$, $j = 1, \ldots, N$, computed from the N training images, and y_j leaf counts, we solve a regression problem to learn from the data a function f that estimates the number of leaves in an image. Here we use SVR to learn a regressor although other nonlinear regression frameworks, such as random forests [7], might be used. (Tests showed no difference between the two.)

SVR shares the same principle of support vector machine [11], but instead of finding the best separation line maximizing the margin between two classes, SVR finds the best fitting line that approximates the data, within a tolerance term ε. SVR minimizes the amount of error outside the $\pm\varepsilon$ threshold (the so-called SVR tube) [34]. To model the nonlinear relationship between image descriptors and number of leaves, we adopt a nonlinear SVR formulation and the *radial basis function* (RBF) kernel $\phi(\mathbf{x}, \mathbf{y}) = \exp(-\gamma\|\mathbf{x} - \mathbf{y}\|^2)$, where $\gamma > 0$ is a model parameter, to map the data into a high-dimensional feature space [33].

To train the SVR we use the vectors $\mathbf{x}_j$ as training samples and the corresponding number of leaves y_j in I_j as target value. The final estimation provided by the regression is a real number, which is rounded to the nearest integer.

Once codebooks and regressors are learned, at test time given an image (and its plant foreground mask) we convert to log-polar domain and patches are extracted, as in Section 3.2. Triangle encodings are computed via the learned dictionary and resulting features are pooled together, to obtain a global descriptor to provide to the regressor.

4 Results and Discussion

In this section, we evaluate our leaf counting approach on image data showing rosette plants. First, we discuss experimental settings and evaluation criteria. Next, we present results obtained on training and testing datasets, comparing also to a variant of the proposed method aimed to learn better representations for the central part of a plant. We compare with a counting via segmentation method [30] and recent density based methods [4, 23].

Image data: We use three datasets, namely A1, A2, and A3, consisting of images showing top views on individual plants provided by the LCC CVPPP 2015 challenge organizers [25, 32]. Images in A1 and A2 (approximately 500×500 pixels) are from *Arabidopsis thaliana* plant subjects, but in A1 are only from wild types (Col-0), while in A2 are also from four different mutant lines (plant identity is unknown in the images). A3 (2448×2048 pixels) shows young tobacco plants (*Nicotiana tabacum*). Each image in the training dataset is provided with a foreground segmentation mask (i.e., plant vs. background), leaf center annotations, number of leaves. Training sets include 128, 31, and 27 images for A1, A2, and A3, respectively. Testing sets include 33, 9, and 56 images for A1, A2, and A3, respectively, and corresponding plant foreground masks, but number of leaves are unknown. Testing results are evaluated by the organizers.

Choice of parameters: We use only the green channel of the original RGB images for computational simplicity. Alternatively we could opt for an illumination invariant transform such as the HSV (Hue, Saturation, Value), or color transforms with class separation properties to obtain one or more channels [26]. For each dataset we repeat the learning process separately. Parameters are found via cross-validation on the training set, and are the same for all datasets. We set $W = 20^\circ$ (see Section 3.2), since smaller values would result in a noisy FG/BG ratio curve, while larger ones would provide coarse results. In the patch extraction phase, we use $S = 15$ and $W' = 40^\circ$. K-means learns $K = 50$ centroids, using the K-means++ initialization criterion [29]. We observe that large values of K lead to a coherent codebook with redundant clusters. Max-pooling is performed using $T = 5$ non-overlapping regions in the log-polar image. Prior to the regression we normalize the global descriptors by subtracting the mean and dividing by the standard deviation (computed on all $\mathbf{x}_j$ vectors). For SVR, we use $\gamma = 1/(TK)$, where TK is the dimension of $\mathbf{x}_j$, and loss parameter $\varepsilon = 0.001$.

Learning separately inner and outer areas, the *IOLC* variant: In rosette plants young leaves tend to grow from the center out and as such less mature leaves are closer to the center. Such leaves are small, they heavily overlap, and due to low resolution are usually missed by many algorithms. To evaluate how the proposed method (GLC), outlined in Section 3, performs in this case, and to show that we can still learn appropriate codebooks, we compare it also with a variant of the proposed method, which learns separate dictionaries according to leaf location. This variant, termed here *Inner-Outer Leaf Counting* (IOLC), relies on leaf center's coordinates to learn separately the inner part of the plant, namely the top-most in log-polar representation, and the lower part, namely the bottom-most in $\tilde{I}$. To separate the upper part from the lower one in a deterministic fashion, the log-polar image is scanned horizontally from the top downward (i.e., from the center outwards). The separation line between the two parts is found at the vertical position where the first background pixel (from the plant mask) is found. The IOLC learns two different codebooks for the two parts

	CountDiff [mean(SD)]		AbsCountDiff [mean(SD)]		Percent Agreement [%]		MSE	
	IOLC	*GLC*	*IOLC*	*GLC*	*IOLC*	*GLC*	*IOLC*	*GLC*
A1	-0.11(1.04)	-0.13(0.88)	0.73(0.75)	0.48(0.74)	40.6	77.3	1.09	0.78
A2	-0.35(2.18)	-0.48(2.20)	1.45(1.65)	1.39(1.76)	41.9	74.2	4.74	4.94
A3	-0.30(1.10)	0.19(0.92)	0.67(0.92)	0.48(0.80)	51.9	92.6	1.26	0.85
All	-0.18(1.31)	-0.14(1.21)	0.84(1.01)	0.63(1.04)	42.5	79.0	1.73	1.48

Table 1: Training results of our proposed method (IOLC and GLC versions).

	CountDiff [mean(SD)]		AbsCountDiff [mean(SD)]		Percent Agreement [%]		MSE	
	IOLC	*GLC*	*IOLC*	*GLC*	*IOLC*	*GLC*	*IOLC*	*GLC*
A1+	0.00(0.72)	-0.02(0.76)	0.39(0.60)	0.41(0.65)	66.4	82.8	0.52	0.58
A2+	-0.16(1.42)	-0.29(1.32)	0.87(1.12)	0.74(1.12)	48.4	77.4	1.97	1.77
A3+	0.07(0.83)	0.07(0.62)	0.52(0.64)	0.30(0.54)	55.6	88.8	0.66	0.37
All+	-0.01(0.89)	-0.05(0.87)	0.49(0.74)	0.45(0.74)	61.8	82.8	0.78	0.75

Table 2: Training results of our proposed method (IOLC and GLC versions) using the *augmented* dataset.

respectively. In this case, max-pooling regions are $T = 2$ in the upper part and $T = 5$ in the lower one. Finally, two separate SVRs are trained, where the target values y_j are chosen according to the number of annotations (leaves) inside the respective areas. The results of the two SVRs are added and then rounded.

Evaluation metrics: We evaluate leaf count accuracy using metrics provided in the LCC: (i) *CountDiff*, average difference between algorithmic estimation of the count and ground truth, reported as mean and standard deviation (SD), (ii) *AbsCountDiff*, average of absolute count errors, and reported as mean (SD) (iii) *MSE*, mean squared error, and (iv) *PercentAgreement*, indicating in how many cases the algorithmic estimation agrees with ground truth. For all metrics, except PercentAgreement, values close to 0 are better. We also measure goodness of fit of the regression using the R^2 coefficient of determination (with $R^2 = 1$ being the best).

Implementation details: We implement our algorithm in Matlab. For training, due to the large size of the datasets, we run the experiments on a CentOS 6.6 server with 4 CPUs Intel Xeon E7540 (6 cores with hyper-threading) and 64 GB of RAM. Although not necessary for testing (since the process is simpler), we use the same computational setup. Overall, we find that it takes approximately 20 secs per image for training, out of which 80% is spent to learn the features, and less than 0.5 secs to train the regressor. On the other hand testing (i.e., predicting the number of leaves in an unseen image) takes less than 3 secs per image, since at test time we only need to extract the patches, obtain the encoding on the learned features, and apply the regressor to estimate the count. At test time memory use is significantly lower, since as we extract a patch its encoding (on the codebook) can be obtained directly.

4.1 Experimental Results

Training Results

Comparing GLC and IOLC: In Table 1 we report the training error for GLC and compare it to the IOLC variant of the proposed method. Overall, GLC obtains better performance,

	CountDiff [mean(SD)]		AbsCountDiff [mean(SD)]		Percent Agreement [%]		MSE	
	GLC	*Ref. [30]*	*GLC*	*Ref. [30]*	*GLC*	*Ref. [30]*	*GLC*	*Ref. [30]*
A1	-0.79(1.54)	-1.8(1.8)	1.27(1.15)	2.2(1.3)	27.3	-	2.91	-
A2	-2.44(2.88)	-1.0(1.5)	2.44(2.88)	1.2(1.3)	44.4	-	13.33	-
A3	-0.04(1.93)	-2.0(3.2)	1.36(1.37)	2.8(2.5)	19.6	-	3.68	-
All	-0.51(2.02)	-1.9(2.7)	1.43(1.51)	2.4(2.1)	24.5	-	4.31	-

Table 3: Results for the testing set of our proposed GLC method with regressor(s) and features learned on the *augmented* dataset. For comparison the findings of *Pape and Klukas* [30] on the same testing set are shown (values for only two metrics were available).

reaching almost 80% agreement with the ground truth (PercentAgreement), indicating that features collected in the entire log-polar representation give satisfactory information to predict even leaves at the center of the plant. Also, with GLC we observe a better fit to the training data (R^2 is 0.83, 0.77, and 0.86 for A1, A2, and A3, respectively) w.r.t. IOLC (R^2 is 0.70, 0.78, and 0.75 for A1, A2, and A3, respectively). Thus, GLC a method that requires only the number of leaves to train (an easier annotation problem) w.r.t. IOLC which needs the leaf centers, shows preferable behavior.

Augmenting the training set: The datasets used here provide a limited amount of training images, which could penalize learning-based approaches. To explore this we train our algorithm by varying the size of training data, whereas the remaining training part is used as a validation set. We find that the MSE in the training set reaches a plateau when we learn using 32 to 64 images, whereas the MSE in the validation set improves by $\sim$20%. This motivated us to augment the dataset by shifting the log-polar image, performing the full learning procedure on the *augmented* dataset. We apply 3 rightward circular shifts for every training image, obtaining a 4-fold increase of each training set. The shift displacement is $D/4$, where D is the pooling region size (see Section 3.4). In Table 2 we report the training error using the augmented datasets. Comparing Tables 1 to 2 we observe that training with the augmented datasets leads to a considerable improvement in all cases, both for GLC and IOLC. Since GLC is simpler and more robust in the following only GLC is reported.

Comparison with density-based counting methods: Our global regression GLC does not use leaf center annotations. To compare our performance with methods that do use such topological information, we adapt also two density-based methods to our application [4, 23]. The approach of *Lempitsky and Zisserman* [23], is used to learn a density function based on leaf center annotations on the A1 training dataset (similar arguments for A2 and A3 hold but are omitted for brevity). We extract from the green color channel, dense SIFT [24] descriptors with bin size of 15, and use K-means to create a codebook of $K = 800$ to represent the data. We learn the pixel-level density function using L_1 regularization in the optimization objective. We find lower performance compared to GLC, obtaining CountDiff = 0.82(1.97) and AbsCountDiff = 1.59(1.42) (cf. Tables 1 to 2). We test also *Arteta et al.* [4], extracting again dense patches. The best result we obtain on the A1 dataset is CountDiff = -0.5(10.5) and AbsCountDiff = 7.3(7.4), confirming that our approach is outperforming state-of-the-art density-based object counting methods, and reaffirming conclusions of ours and others [4] that such methods are unable to accommodate object size variability within the same scene.

Testing Results

To estimate leaf counts on the images in the testing set we use dictionaries and SVR models learned on the *augmented* training sets. We submitted estimated counts to the organizers only for GLC. We report in Table 3 the testing results of the proposed GLC, together with the counting-via-segmentation method proposed by *Pape and Klukas* [30], the winners of the previous leaf segmentation challenge. Our method outperforms the approach in [30]. In particular, we improve significantly the accuracy on A1 and A3 datasets. Overall, number of leaves predicted by our method is off by at most ±1 leaf in 57% of the cases.

The A2 dataset contains several mutants and some subjects exhibit dwarfism, appearing very small in the images. When such images, or images with many small young leaves in the center, are transformed into the log-polar domain, the effect of interpolation introduces artifacts causing performance loss. Albeit the A3 dataset includes very young (and relatively small) plants, the effects discussed before are compensated by increased image resolution. In fact, training and testing error in A1 and A3 are similar, even if the A3 dataset contains the least amount of training images. This motivates future investigations of specialized features for regions close to the center.

5 Conclusions

In this paper, we aim to count leaves in images of rosette plants –a challenging vision problem due to variability in terms of size, appearance, and rotation of leaves. We proposed a machine learning-based approach to estimate the number of leaves from top-view images. We compute global features for each image, using local patches extracted from the log-polar domain, which accounts for rotation and scale variability. We relate global features to leaf count with supervised regression.

Using standardized datasets in the context of the Leaf Counting Challenge, of the CVPPP 2015 workshop, our method outperforms previous state-of-the-art methods [30] on the same data. We also compared with state-of-the-art methods for counting via density estimation, showing that our learning framework outperforms the methods in [4, 23] in dataset we tested (A1). We also found that augmenting the training set, by circularly shifting the log-polar representations, increases performance.

Our approach is simple to train. It requires input images and a foreground/background segmentation (which for plants is easier to obtain than other applications). In terms of annotation it requires only a total leaf (object) count per image. This is much easier than centers or bounding boxes required for density or detection based methods. Our experiments show that with adequate training data, at testing time for an unseen image satisfactory accuracy in counting is obtained within a few seconds (per image), opening the road to automated and reliable leaf count estimation in high throughput phenotyping applications. Integrating such learning-based approaches to centralized cloud based analysis frameworks such as the one available at http://www.phenotiki.com would increase even more the reach of automated high throughput phenotyping [27].

Acknowledgements

This work is partially supported by a Marie Curie Action: “Reintegration Grant” (grant number 256534) of the EU’s Seventh Framework Programme (FP7).

References

[1] E. E. Aksoy, A. Abramov, F. Wörgötter, H. Scharr, A. Fischbach, and B. Dellen. Modeling leaf growth of rosette plants using infrared stereo image sequences. *Computers and Electronics in Agriculture*, 110:78–90, January 2015.

[2] H. Araujo and J. M. Dias. An introduction to the log-polar mapping. In *2nd Workshop on Cybernetic Vision*, pages 139–144, 1996.

[3] C. Arteta, V. Lempitsky, J. A. Noble, and A. Zisserman. Learning to detect partially overlapping instances. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3230–3237, 2013.

[4] C. Arteta, V. Lempitsky, J. A. Noble, and A. Zisserman. Interactive object counting. In *Computer Vision – ECCV 2014*, volume 8691 of *Lecture Notes in Computer Science*, pages 504–518. Springer, 2014.

[5] Y. Boureau, J. Ponce, and Y. Lecun. A theoretical analysis of feature pooling in visual recognition. In *International Conference on Machine Learning*, pages 111–118, 2010.

[6] D. Bradley, O. Ratcliffe, C. Vincent, R. Carpenter, and E. Coen. Inflorescence commitment and architecture in Arabidopsis. *Science*, 275(5296):80–83, 1997.

[7] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[8] W. Chang and S. Lee. Description of shape patterns using circular arcs for object detection. *IET Computer Vision*, 7(2):90–104, 2013.

[9] A. Chayeb, N. Ouadah, Z. Tobal, M. Lakrouf, and O. Azouaoui. HOG based multi-object detection for urban navigation. In *International Conference on Intelligent Transportation Systems*, pages 2962–2967, 2014.

[10] A. Coates, A. Arbor, and A. Y. Ng. An analysis of single-layer networks in unsupervised feature learning. *International Conference on Artificial Intelligence and Statistics*, pages 215–223, 2011.

[11] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[12] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 886–893, 2005.

[13] S. Dhondt, N. Wuyts, and D. Inzé. Cell to whole-plant phenotyping: the best is yet to come. *Trends in Plant Science*, 18(8):433–444, 2013.

[14] C. Elkan. Using the triangle inequality to accelerate k-means. In *International Conference on Machine Learning*, pages 147–153, 2003.

[15] L. Fiaschi, R. Nair, U. Koethe, and F. A. Hamprecht. Learning to count with regression forest and structured labels. In *International Conference on Pattern Recognition*, pages 2685–2688, 2012.

[16] F. Fiorani and U. Schurr. Future scenarios for plant phenotyping. *Annual Review of Plant Biology*, 64:267–291, 2013.

[17] A. Hartmann, T. Czauderna, R. Hoffmann, N. Stein, and F. Schreiber. HTPheno: an image analysis pipeline for high-throughput plant phenotyping. *BMC Bioinformatics*, 12(1):148, 2011.

[18] D. Houle, D. R. Govindaraju, and S. Omholt. Phenomics: the next challenge. *Nature Reviews Genetics*, 11(12):855–866, 2010.

[19] O. Janssens, J. De Vylder, J. Aelterman, S. Verstockt, W. Philips, D. Van Der Straeten, S. Van Hoecke, and R. Van de Walle. Leaf segmentation and parallel phenotyping for the analysis of gene networks in plants. In *European Signal Processing Conference*, pages 1–5, 2013.

[20] Y. Jia, C. Huang, and T. Darrell. Beyond spatial pyramids: Receptive field learning for pooled image features. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3370–3377, 2012.

[21] M. Koornneef, C. Hanhart, P. van Loenen-Martinet, and H. Blankestijn de Vries. The effect of daylength on the transition to flowering in phytochrome-deficient, late-flowering and double mutants of Arabidopsis thaliana. *Physiologia Plantarum*, 95(2): 260–266, 1995.

[22] K. Lee, C. Lee, S. Kim, and Y. Kim. Fast object detection based on color histograms and local binary patterns. In *TENCON 2012 - IEEE Region 10 Conference*, pages 1–4, 2012.

[23] V. Lempitsky and A. Zisserman. Learning to count objects in images. In *Advances in Neural Information Processing Systems 23*, pages 1324–1332. 2010.

[24] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[25] M. Minervini, M. M. Abdelsamea, and S. A. Tsaftaris. Image-based plant phenotyping with incremental learning and active contours. *Ecological Informatics*, 23:35–48, 2014. Special Issue on Multimedia in Ecology and Environment.

[26] M. Minervini, C. Rusu, and S. A. Tsaftaris. Computationally efficient data and application driven color transforms for the compression and enhancement of images and video. In *Color Image and Video Enhancement*, chapter 13. Springer, 2015.

[27] M. Minervini, H. Scharr, and S. A. Tsaftaris. Image analysis: The new bottleneck in plant phenotyping. *IEEE Signal Processing Magazine*, 32(4):126–131, 2015.

[28] R. C. O'Malley and J. R. Ecker. Linking genotype to phenotype using the Arabidopsis unimutant collection. *The Plant Journal*, 61(6):928–940, 2010.

[29] R. Ostrovsky, Y. Rabani, L. J. Schulman, and C. Swamy. The effectiveness of Lloyd-type methods for the k-means problem. *Journal of the ACM*, 59(6):28:1–28:22, 2013.

[30] J. Pape and C. Klukas. 3-D histogram-based segmentation and leaf detection for rosette plants. In *Computer Vision - ECCV 2014 Workshops*, volume 8928 of *Lecture Notes in Computer Science*, pages 61–74. Springer, 2015.

[31] T. P. Pridmore, A. P. French, and M. P. Pound. What lies beneath: underlying assumptions in bioimage analysis. *Trends in Plant Science*, 17(12):688–692, 2012.

[32] H. Scharr, M. Minervini, A. Fischbach, and S. A. Tsaftaris. Annotated image datasets of rosette plants. Technical Report FZJ-2014-03837, Forschungszentrum Jülich GmbH, July 2014. URL http://hdl.handle.net/2128/5848.

[33] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, New York, NY, USA, 2004.

[34] A. J. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 14(3):199–222, 2004.

[35] E. P. Spalding and N. D. Miller. Image analysis is driving a renaissance in growth measurement. *Current Opinion in Plant Biology*, 16(1):100–104, 2013.

[36] A. Telfer, K. M. Bollman, and R. S. Poethig. Phase change and the regulation of trichome distribution in Arabidopsis thaliana. *Development*, 124:645–654, 1997.

[37] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms, 2008. URL http://www.vlfeat.org/.

[38] A. Walter and U. Schurr. The modular character of growth in Nicotiana tabacum plants under steady-state nutrition. *Journal of Experimental Botany*, 50(336):1169–1177, 1999.

[39] A. Walter, F. Liebisch, and A. Hund. Plant phenotyping: from bean weighing to image analysis. *Plant Methods*, 11(1):1–11, 2015.

[40] Y. Wang, H. Lian, P. Chen, and Z. Lu. Counting people with support vector regression. In *International Conference on Natural Computation*, pages 139–143, 2014.

[41] A. Yao, J. Gall, C. Leistner, and L. Van Gool. Interactive object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3242–3249, 2012.

[42] X. Yin, X. Liu, J. Chen, and D. M. Kramer. Multi-leaf tracking from fluorescence plant videos. In *International Conference on Image Processing*, pages 408–412, 2014.