Classification of Anthurium flowers using combination of PCA, LDA and support vector machine

Alireza Soleimanipour, Gholam Reza Chegini*, Jafar Massah

(*Department of Agrotechnology, College of Aburaihan, University of Tehran, Tehran, Iran*)

Abstract: Two main steps in object recognition systems involve feature representation and classification. In this study, the combination of principal components analysis (PCA), linear discriminant analysis (LDA) and support vector machine (SVM) approaches were used to develop a cultivar classification system for Anthurium flower, which PCA, LDA, and SVM were applied for data reduction, feature extraction, and classification, respectively. The algorithm was tested on a database of Anthurium flower images, which included the images of 20 cultivars of the flower with different sizes, and little variations in angles of rotation (from $-\pi/12$ to $\pi/12$) that flowers are placed under the camera, and lighting conditions. Results were evaluated from the two points of view of classification accuracy and computation time, and the algorithm had remarkable results when trained using suitable multi-class SVM classifier features, as it is possible to increase the classification accuracy up to 99.5% using an RBF kernel function and Sparse random multi-class codding method. Cultivar recognition of flowers is an important step for subsequent flower real-time grading tasks and such algorithms could be used for these procedures.

Keywords: Anthurium flower, classification accuracy, multi-class SVM, computation time, cultivar recognition

Citation: Soleimanipour, A., G. R., Chegini, J. Massah. 2018. Classification of Anthurium flowers using combination of PCA, LDA and support vector machine. Agricultural Engineering International: CIGR Journal, 20(1): 219–228.

1 Introduction

Recently the machine learning and pattern recognition have become the most popular areas of research in computer vision and demanding in the variety of applications such as optical character recognition, automatic speech recognition, computer vision deals with the recognition of objects as well as the identification and localization of their three-dimensional environments, recognition of objects on earth from the sky (by satellites) or from the air (by aeroplanes and cruise missiles), Archaeology, medical diagnosis tests, personal identification systems, web search engines, data mining, database retrieval, face recognition, identifying fingerprints, bioinformatics, as well as agriculture, etc. (Robert, 1965; Zhao et al., 2003; Dutt et al., 2012; Fernandes et al., 2005; Sharafi et al., 2016; Barajas-Garcia et al., 2016).

The object recognition algorithms are divided into two main categories, model-based or geometric-based object recognition approaches and appearance-based object recognition approaches (Gaidhane et al., 2014). The model-based object recognition methods construct a model of the object, which is able to capture the object variations in the image. For example, in face recognition applications, feature-based method derives distance and relative position features from the placement of internal facial elements (e.g., eyes, etc.). In these approaches, the prior knowledge of the object is highly utilized to design the model. So, the systems based on such methods are semi-automatic and labor consuming (Lu, 2003).

In the appearance-based methods, which are considered as the most successful ones, the two-dimensional images are represented in one-dimensional space. Therefore, many appearance-based approaches use statistical techniques to analyze the distribution of the object image vectors in the vector space and derive an efficient and effective representation (feature space) according to different applications. The similarity

Received date: 2017-07-14 **Accepted date:** 2017-11-21

*** Corresponding author: Gholamreza Chegini,** Associate Professor, Department of Agrotechnology, College of Aburaihan, University of Tehran, Tehran, Iran. Email: chegini@ut.ac.ir. Tel: (+98) 9126356329, Fax: (+98) 2136040614.

between the stored prototypes and a test image is then carried out in the feature space using a comparison between the feature vectors. Principle component analysis (PCA), linear discriminant analysis (LDA), discriminative common vectors (DCV), and their kernel versions are well-known linear and non-linear type appearance-based methods (Lu, 2003; Belhumeur et al., 1997; Yu and Yang, 2001; Cevikalp et al., 2005; Edizkan et al., 2013).

The postharvest operations in greenhouses or fields, such as bunching, packing, and grading are determinant for quality and life of cut flowers and foliage (Celikel and Karaaly, 1995). Nowadays, developing automatic systems to implement the postharvest operations is a necessity for all ornamental cultivars, as well as Anthurium, in order to speed up the processing operations. Grading is one of the main processes in postharvest operations and because of the high vulnerability of flowers, mechanical grading systems should be carefully designed to ensure efficiency and avoid damaging the flowers. Designing a machine for grading various flowers with multiple cultivars require a system equipped with a robust cultivar recognition algorithm.

There are various ways to recognize a plant, like flower, root, leaf, fruit etc. Recently, computer vision and pattern recognition techniques have been applied towards automated process of plant recognition (Pan and He, 2008). Until now, different classification techniques like support vector machines (SVM) (Zhang et al., 2011), probabilistic neural network (PNN) (Kadir et al., 2011; Wu et al., 2007), moving media centres hyperspheres (MMC) (Du et al., 2007; Zhang et al., 2004) and ANN (artificial neural network) with back-propagation (Heymans et al., 1991; Satti et al., 2013) have been used for plant classification.

Zhang et al. (2011) proposed a method to generate the feature space that combines local texture features using wavelet decomposition and co-occurrence matrix statistics and global shape features to describe the collected plant leaves. SVM classifiers were used to classify the different species. Priya et al. (2012) presented a machine learning approach for plant leaf recognition. The algorithm was able to classify 32 kinds of plants using features extracted and processed by PCA, and SVM was used as classification.

The overall objective of our research is to design an automated flower grading machine by a computer vision approach, which is mainly based on image processing and object recognition techniques. In order to utilize the grading machine for various flowers and cultivars, it should be able to recognize the flowers cultivars, before sorting or grading them. Therefore, the grading machine needs to be equipped with a cultivar recognition system, which was mainly developed on machine learning approaches. In this study, an algorithm was developed to identify the Anthurium flower cultivars using a combination of PCA, LDA and SVM (Figure 1).

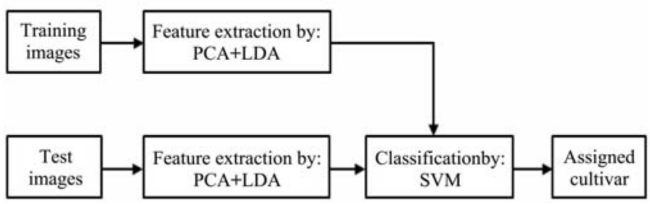


Figure 1 A classification system for cultivars of Anthurium flower

2 Materials and methods

The parts of an object image such as boundary shape, special areas, color distribution and background are regarded as significant information for the object recognition system. Moreover, images are often affected by various factors, such as illumination, image size, angle of rotation that the object is placed under camera, changes in object shape, etc. All above factors would cause difficulties for object recognition and lead to the recognition rate dropped, so image normalization must be carried out previous to feature extraction. The normalization includes geometric normalization, object image correction, gray balance, etc.

A computer vision system was developed for image capturing, image processing as well as cultivar recognition. The computer vision system included a high-resolution IP camera (Grandstream GXV 3601 HD-IP Camera), a proper lighting room and a Laptop. Lighting of imaging room was done using 12VDC white LED lamps with 6500 Kelvin color temperature and illuminance of approximately 5500 lux, which installed symmetrically in the room for uniform light distribution on the sample. Images were randomly rotated from $-\pi/12$

to π/12 under camera. We used the images of 20 cultivars of Anthurium flower (Figure 2). The images were cropped and resized to four resolutions of 128×128, 64×64, 32×32 and 16×16 pixels with a pixel density of almost 8.6, 4.3, 2.15, and 1.07 pixels cm^{-1}, respectively (Figure 3). Geometric normalization and preprocessing operations, such as applying color filter to RGB images to reduce the lighting undesirable effects and contrast improvement, converting the RGB images to gray-scale format, reducing nose through a 3×3 median filter to enhance images quality, were performed on the images.

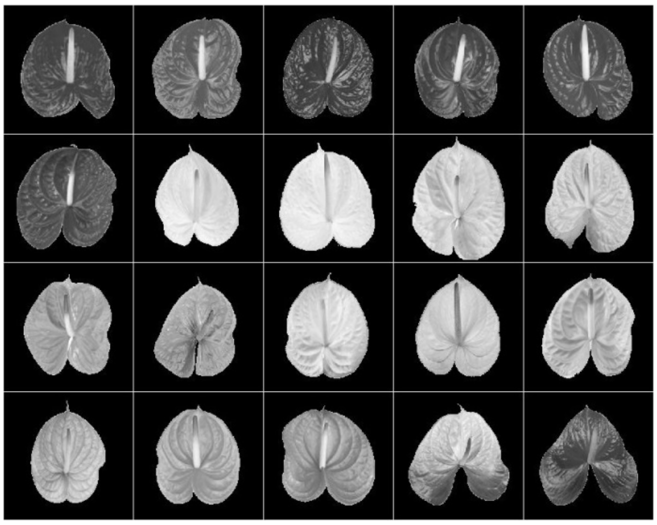
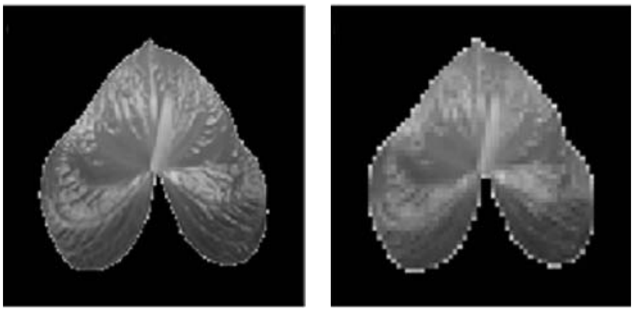
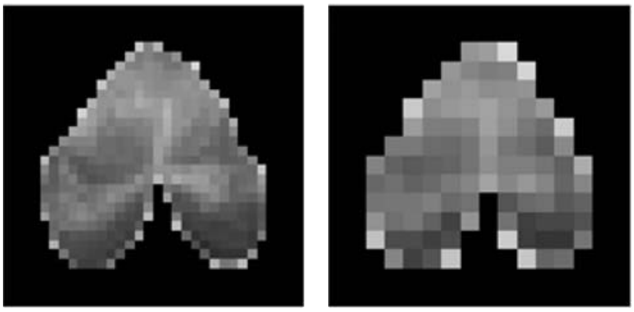


Figure 2 Twenty cultivars of Anthurium flower after geometric normalization and preprocessing



a. Database images with 128×128 pixels b. Database images with 64×64 pixels



c. Database images with 32×32 pixels d. Database images with 16×16 pixels

Figure 3 Database images with different resolutions

2.1 Feature extraction

A grey-scale image can be represented as a vector of pixel values (i.e., a 128×128 pixels gray image can be represented as a vector containing 16384 values among 0 to 255). PCA and LDA are algorithms that transform image vectors into their subspaces (also called “feature spaces”) and serve as a feature extraction stage by which it is possible to find a hyperplane that separates data into classes. Both methods implement linear separation of data. PCA is a standard technique used to approximate the original data with lower dimensional feature vectors. PCA aims to maximize between-class data separation, while LDA tries to maximize between-class data separation and minimize within class data separation (Mazanec et al., 2008; Nallammal and Radha, 2012). In summary, PCA was adopted to reduce dimensions of images before LDA was used for feature extraction in this study.

2.1.1 Principle component analysis

PCA is generally used to reduce the dimensionality of the dataset but it retains most of the original variability in the data. PCA transforms the image vectors into their subspaces or “feature space” and is a way to express the data to highlight their similarities and differences, which facilitate identifying patterns in data. In fact, it is discriminating input images into several classes, by maximizing variances between classes (Nallammal and Radha, 2012). The steps of PCA technique include preparing the data, subtracting the mean, calculating the covariance matrix, calculating the eigenvectors and eigenvalues of the covariance matrix and finally selecting the principal components. So, the basic steps of PCA algorithm in detail are as follows:

a. Determine PCA feature spaces from k training images, which transform a $M×N$ matrix image to a $1×(M×N)$ vector image. For instance, i-th image with resolution of 16×16 pixels, image vector containing (16×16) pixels is as follows;

$$x^i = [x_1^i, x_2^i, ..., x_{256}^i] \quad (1)$$

b. Store all k vectors for training images in one matrix as image matrix, which results in a $k×(M×N)$ matrix. For number of training images equal with 1000 (from 20 classes and each class consists 50 images, which were randomly selected and considered as training dataset), we have a 1000×256 matrix;

$$X = \left\lfloor x^1, x^2, ..., x^{k(=1000)} \right\rfloor \quad (2)$$

c. Compute covariance matrix, which achieved from multiplying the image matrix to its transpose. Covariance matrix is a 1000×1000 square matrix;

$$\emptyset = XX^T \quad (3)$$

d. Compute eigenvalues and eigenvectors (as a 1000×1000 matrix) and order the eigenvectors according to their corresponding eigenvalues in descending order.

$$\emptyset V = \Lambda V \quad (4)$$

where, V and Λ are the eigenvectors and eigenvalues of the covariance matrix, respectively.

e. Eliminate eigenvectors associated with zero eigenvalues and order the eigenvectors in V according to their corresponding eigenvalues in descending order. So, the resulted matrix forms the eigenspace, where each column of that is the eigenvector. Visualized eigenvectors of the covariance matrix are called eigenspace features or eigenfaces, which obtained by multiplying the selected eigenvectors and the training images matrix, **X** in Equation (2). In Figure 4, the 40 first eigenspace features of Anthurium flower are shown. The first eigenspace features are more important. In order to decrease computation time, the most important eigenspaces that belong to higher eigenvalues are chosen to train the classifier. Some of the first important features were used to train the SVM classifiers, after applying LDA on them.

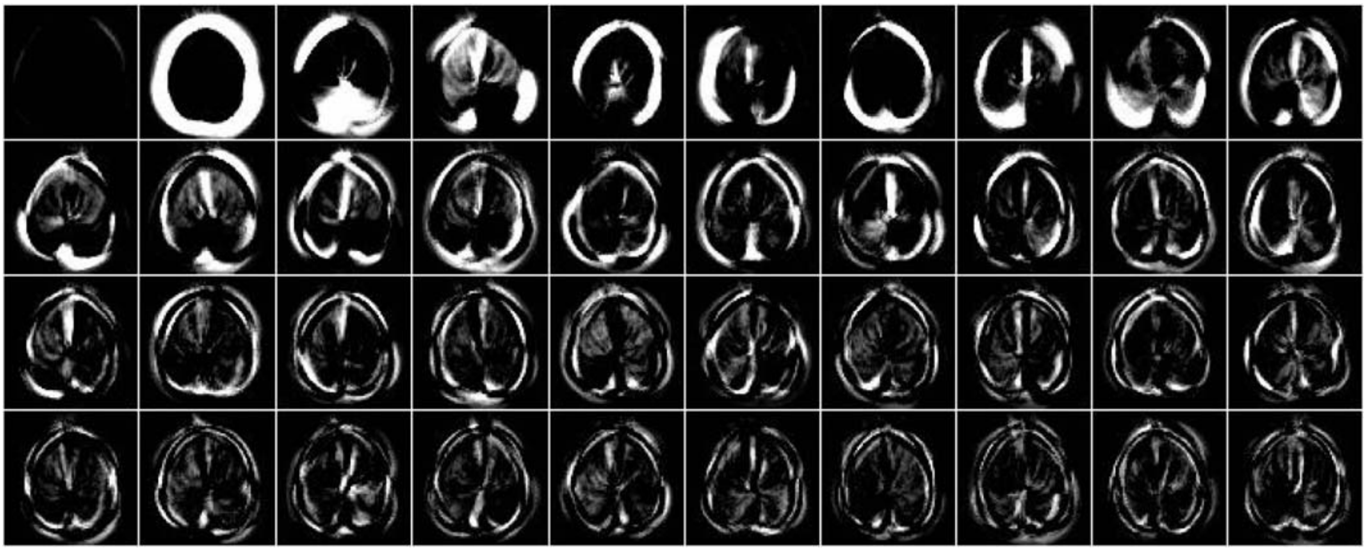


Figure 4 The first 40 eigenspace features or eigenfaces of Anthurium flower belong to higher eigenvalues, which were used to train the SVM classifiers

2.1.2 Linear discriminant analysis

LDA searches for a group of basis vectors, which makes different class samples have the smallest within class scatter and the largest between class scatter. So, in addition to dimensionality reduction, LDA searches the directions for maximum discrimination of classes (Raghavendra et al., 2012). In fact, LDA uses feature subspace obtained from PCA as input data, which cuts the eigenvectors in eigenspace matrix that are not important for object recognition. It is in order to improve computing performance. The basic steps of LDA algorithm are as follows:

a. Determine LDA feature spaces from training data which obtained from PCA algorithm (the first 50 feature spaces produced by applying PCA on training dataset, which is a 50×1000 matrix).

b. Calculate the within-class and the between-class scatter matrixes.

$$\boldsymbol{S}_i = \sum\nolimits_{\boldsymbol{x} \in \boldsymbol{X}_i} (\boldsymbol{x} - \boldsymbol{m}_i)(\boldsymbol{x} - \boldsymbol{m}_i)^{\mathrm{T}} \quad (5)$$

$$\boldsymbol{S}_W = \sum\nolimits_{i=1}^{C} \boldsymbol{S}_i \quad (6)$$

$$\boldsymbol{S}_B = \sum\nolimits_{i=1}^{N} n_i (\boldsymbol{m}_i - \boldsymbol{m})(\boldsymbol{m}_i - \boldsymbol{m})^{\mathrm{T}} \quad (7)$$

where, $\boldsymbol{S}_w$ and $\boldsymbol{S}_B$ are the within class and between class scatter matrixes; n_i is the number of images in the class, $\boldsymbol{m}_i$ is the mean of the images in the class and $\boldsymbol{m}$ is the mean of all the images.

c. Solve the generalized eigenvalue problem to compute eigenvalues and eigenvectors.

$$\boldsymbol{S}_B \mathbf{V} = \mathbf{\Lambda} \boldsymbol{S}_W \mathbf{V} \quad (8)$$

d. Order the eigenvectors according to their corresponding eigenvalues in descending order and keep only the eigenvectors associated with non-zero eigenvalues.

It is worth to note that all training images are projected onto PCA method subspace and all resulted PCA eigenspace data are projected onto LDA subspace. Each test image is also projected to the same subspace and compared and classified using different SVM models. Totally, four SVM compact models were developed with an RBF kernel function and different multi-class coding methods. The used multi-class SVM coding methods include; One-Versus-All, One-Versus-One, Dense Random, and Sparse Random.

2.2 Classification

In pattern recognition and machine learning, classification is the problem of identifying which of a set of class a new observation belongs, on the basis of a training set of data containing observations (or instances) whose class is known (Subramanian, 2014). In this study, the classification of the known cultivars of Anthurium flower from the unknown ones is implemented using multi-class Support Vector Machine.

2.2.1 Support Vector Machine

SVM is a supervised learning algorithm that has been successful in proving itself as an efficient and accurate classification technique for various applications. According to its supervised nature, SVM is implemented in two steps, i.e. training and classification. In the training step, SVM learns a decision boundary in input space from pre-classified training data. In the classification step, classifies input vectors according to the learned decision boundary (Leopold and Kindermann, 2007). A single SVM is a binary classifier, that is, the class labels can only take two values ($y = \pm1$). The data for training are a set of eigenspace features vectors obtained from PCA and LDA algorithms, x_i ($1 \le i \le$ number of training images), along with their categories y_i. The classes are separated by a hyperplane. The equation of a hyperplane is:

$$w \cdot x + b = 0 \tag{9}$$

where, w and b are parameters that are learned in the training step and which determine the class separating hyperplane. Computing this hyperplane is equivalent to solving the following optimization problem. Best separating hyperplane defined as following problem.

$$y_i(w \cdot x_i + b) \ge 1 \tag{10}$$

The SVM aims to find a decision that z_i, as a query sample, belongs to the positive class ($y = +1$) or the negative class ($y = -1$) based on the training samples by the function:

$$class(z_i) = \mathrm{sgn}(\hat{w}zi + \hat{b}) \tag{11}$$

Computing this hyperplane is equivalent to solving the following optimization problem, in which tries to minimize it.

$$V(w,b,\varepsilon) = \frac{1}{2}ww + C\sum_{i=1}^{n}\varepsilon_i \tag{12}$$

$$y_i(wx_i + b) \ge 1 - \varepsilon_i \tag{13}$$

$$\varepsilon_i \ge 0 \tag{14}$$

In machine learning, kernel methods are a class of algorithms for pattern analysis. The SVM classifier belongs to kernel methods and is the most known member of them. The kernel algorithms map data from an original space to a higher dimensional feature space using non-linear mapping (Oravec et al., 2008). Although the high-dimensional space increases the difficulty of the problem, a trick for computing the scalar products in the feature space exists. Computation of the scalar product between two feature space vectors can be done using kernel functions. Using kernel functions, the feature space need not be computed explicitly (Mazanec et al., 2008). In other words, the kernel SVMs allow fitting the maximum-margin hyperplane in a transformed feature space, which the transformation may be nonlinear and the transformed space is a higher dimensional space. Though the classifier is a hyperplane in the higher dimensional feature space, it may be nonlinear in the original input space (Zhang and Wu, 2012). Four common kernels are included: Homogeneous Polynomial (HPOL), Inhomogeneous Polynomial, Radial Basis Function (RBF) or Gaussian kernel, and Hyperbolic Tangent. For each kernel, there should be at least one adjusting parameter so as to make the kernel flexible and tailor itself to practical data. In our tests we use SVM with the RBF (radial basis function) kernel function:

$$k(x_i,x_j) = \exp(-\gamma||x_i - x_j||2),\ \gamma > 0 \tag{15}$$

where, x_i, x_j are data points from the original space. It is important to find optimal parameters γ and C because different parameter setups are suitable for solving different problems.

The SVM method was originally developed as a linear classifier. Several methods have been proposed for multi-class SVMs, and the dominant approach is to reduce the single multi-class problem into multiple binary classification problems. Throughout the methods for multiclass problems, we consider the following four approaches: (i) one-versus-all (OVA) method using winner-takes-all strategy (WTA SVM), (ii) one-versus-one (OVO) method implemented by max-wins voting (MWV SVM), (iii) Dense Random, and (iv) Sparse Random.

The OVA, OVO, Dense Random, and Sparse Random multi-class methods, approximately used K, $K(K\text{-}1)/2$, $10\times\log_2 K$, and $15\times\log_2 K$ binary learners to develop their multi-class SVM classifiers, respectively, which K is the number of classes. In OVA method, for each binary learner, one class is positive and the rest are negative. This design exhausts all combinations of positive class assignments. In OVO method, for each binary learner, one class is positive, another is negative, and the software ignores the rest. This design exhausts all combinations of class pair assignments. In the Dense Random method, for each binary learner, the software randomly assigns classes into positive or negative classes, with at least one of each type. Also, in the Sparse Random method, for each binary learner, the software randomly assigns classes as positive or negative with probability 0.25 for each and ignores classes with probability 0.5 (MathWorks, 2015).

3 Results and discussions

The performance of the system was tested by making real-time recognition experiments on a database of 2000 images from 20 cultivars of Anthurium flower (100 images from each cultivar). Images were prepared under different lighting conditions, sizes and angles of rotation that the flowers are placed under camera (from $-\pi/12$ to $\pi/12$). For instance, Figure 5 shows 40 query images from four cultivars at various conditions, which used to evaluate the proposed recognition algorithm. As explained before, in the training step, the flower images with cropped and resized to 128×128, 64×64, 32×32, and 16×16 pixels; the image features were reduced and extracted by PCA and LDA algorithms; and after eliminating worthless eigenspace features, important ones that belong to higher eigenvalues were used to train the SVM classifier.

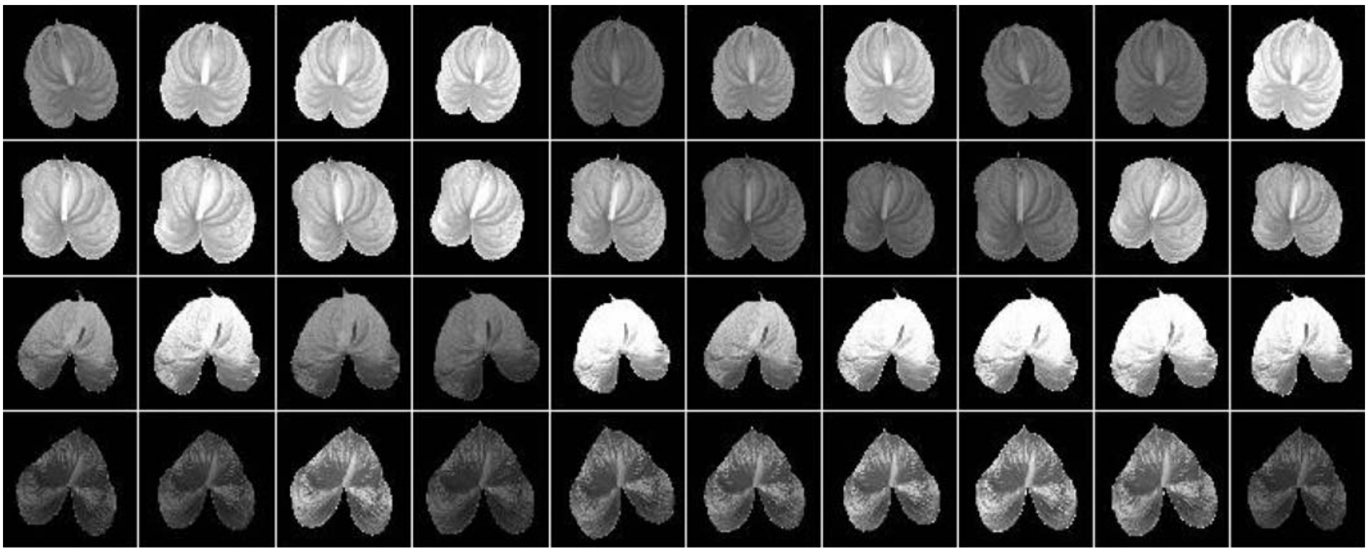


Figure 5 Examples of images used as query images to evaluate cultivar classification algorithm for Anthurium flower

The system was evaluated on the basis of the classification accuracy and recognition speed. The experiments were carried out on a Laptop with Intel B960 2.20 GHz processor and 4.00 GB of RAM running under the Microsoft Windows 7 operating system. The algorithm was entirely developed on the Matlab 2015a (The MathWorks®) computer program. In order to a proper evaluation of the computation time, all other programs were stopped before running the program on the computer. The algorithm can be run or tested on any computer platforms where Matlab is available.

3.1 Feature selection performance

As mentioned above, the proposed algorithm is based on holistic features of different cultivars of Anthurium flower. In this method, pixel values of gray-scale images are considered as features. The pixel value is a single number that represents the brightness of the pixel. Based on image resolution, features were extracted using a combination of PCA and LDA methods. Two MATLAB scripts were written to process the training and testing images separately. The output of training process was the eigenspace features or the flower eigenfaces, which were used to create the projected features for both of training and query images. By initial evaluations and tests and errors, we found that the first 50 projected features are enough, as increasing the features more than 50 had no significant effect on the recognition accuracy but also increased the computation time at the classification stage (results are not presented). The performance of the feature selection algorithm was evaluated using the computation time of training the algorithm with 50 images. Also, the computation time for obtaining the projected features of query images was evaluated (Figure 6). As Figure 6 shows, by increasing the image resolution, the computation time increased, too. The testing execution time was subjected to 100 images, and according to the results presented in Figure 6, the highest

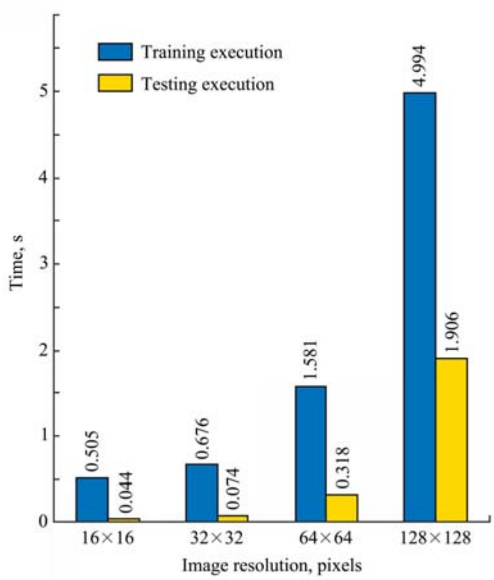


Figure 6 The time required to train and test the algorithm, in feature selection step (applying PCA+LDA on the training and test databases images), with respect to image resolution

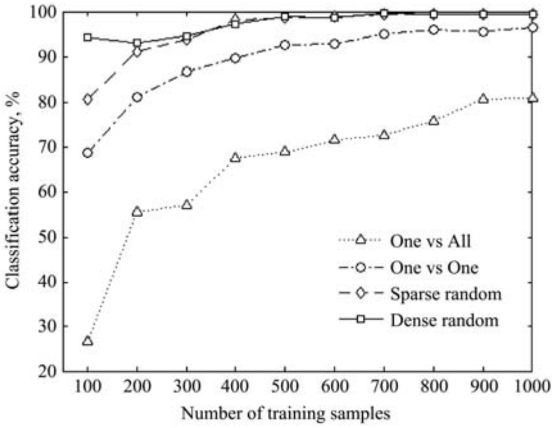
computation time for tests belonged to the image resolution of 128×128 pixels, and equal with 0.019 s for each image.

3.2 Classification performance

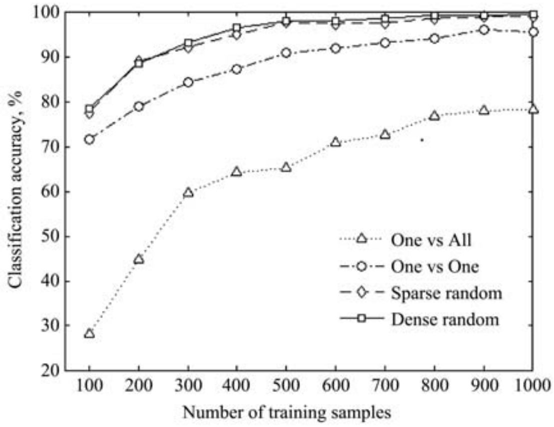
The most important criteria for evaluating the performance of these methods are their accuracy rate of classification and computation time. In classification step, we train all datasets only with a Gaussian or RBF kernel function and the function coefficients (C and γ in Equations (12) and (15)) were the same for all experiments. In order to implement multi-class SVM, we used four methods solving several binary SVMs (one-versus-all, one-versus-one, Dense Random, and Sparse Random). The classification accuracy was calculated using the following equation for different conditions, such as number of training samples, image resolutions, multi-class SVM methods.

$$classification\ accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{16}$$

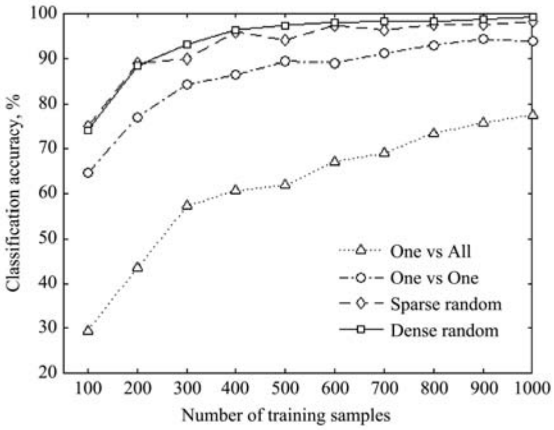
which, TP is true positive; TN is true negative; FP is false positive, and FN is false negative. Figure 7 shows the classification accuracy for different image resolutions and multi-class SVM methods against the number of training samples. The Dense Random and Sparse Random methods had better performance than the OVO as well as OVA methods according to the classification accuracy. Besides, the classification accuracy for image resolution of 16×16 pixels was better than others. As the classification accuracies at the image resolution of 16×16 pixels and for the number of training samples more than 40 for each cultivar and using multi-class SVM coding methods of Sparse Random and Dense Random were more than 99.5%, averagely. The OVO method had better performance than OVA method based on its classification accuracy. We evaluated the results for samples with two other images resolutions (8×8 and 256×256 pixels) and observed that these resolutions did not lead to good results (results are not presented). According to SVM section, in our study, the OVA, OVO, Dense Random, and Sparse Random multi-class SVM methods used, approximately, 20, 190, 43, and 65 binary learners to develop their classifiers. Inappropriate performance obtained for OVA method can be used because of its low binary learners.



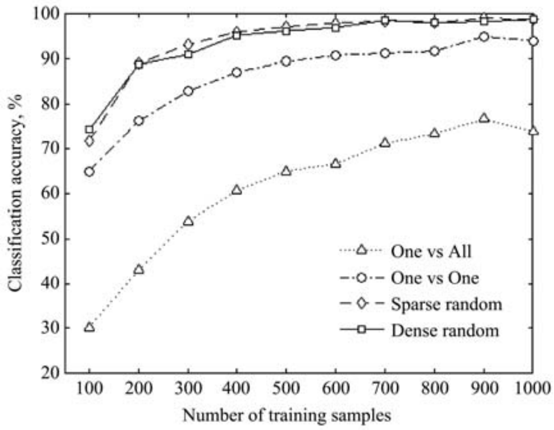
a. Samples with image resolutions of 16×16 pixels



b. Samples with image resolutions of 32×32 pixels



c. Samples with image resolutions of 64×64 pixels



d. Samples with image resolutions of 128×128 pixels

Figure 7 The classification accuracy of different image resolutions and multi-class SVM coding methods against the number of training samples with image resolutions of different pixels

Hsu and Lin (2002) discussed decomposition implementations for two all together methods and compared them with three methods based on several binary classifiers: one versus one, one-against-all, and Directed Acyclic Graph (DAG) methods. Their experiments on large problems showed that one versus one method and DAG may be more suitable for practical use. Generally, it could be concluded that SVM classifiers had proper performance in plant identification systems. In comparison with other machine learning methods, Priya et al. (2012) proposed an algorithm to classify 32 kinds of plants via the leaf images based on 12 Digital Morphological Features (DMFs) obtained from 5 basic features, which extracted and proposed by PCA to form the input vector of SVM. They expressed that the proposed algorithm performance in compression with k-nearest neighbor (KNN) classifier had better accuracy and takes very less time for execution.

Table 1 shows the computation time for training the SVM with different image resolutions and multi-class SVM coding methods. The computation time belongs to classifying 1 query image with the different compact models created in training stage. The values are average of processing 100 images and using 1000 images to train the classifiers. As can be seen from Table 1, the training and testing time for the OVA method was lowest and averagely was equal with 0.76 s for training the compact model, and the classification time of each query image with this compact model was equal 0.31 s. The highest training time was belonged to the Sparse Random method and was averagely equal with 24.86 s and the higher classification time was 0.47 s per image. It is clear that the image resolution has no effect on training execution time of the compact models as well as classification time of query samples (Table 1). Also, the classification accuracy and the computation time have an inverse relationship with each other, as classification with higher accuracy requires taking methods that operate in more time-consuming manner.

Table 1 The time required to train and test SVM models with different multi-class coding methods and image resolutions (the values are corresponded to recognizing cultivar of 1 query image in seconds)

Image resolution	SVM coding method							
	One vs One		One vs All		Sparse Random		Dense Random	
	Train (s)	Test (s)	Train (s)	Test (s)	Train (s)	Test (s)	Train (s)	Test (s)
16×16	3.780	0.763	0.697	0.301	24.672	0.473	20.038	0.407
32×32	3.592	0.721	0.784	0.303	24.922	0.463	20.186	0.420
64×64	3.615	0.713	0.842	0.314	25.047	0.476	20.076	0.437
128×128	3.588	0.776	0.727	0.307	24.794	0.481	19.645	0.451

4 Conclusions

Twenty cultivars of Anthurium flower were classified using a combination of PCA, LDA and SVM methods as a holistic object recognition approach. In feature selection stage, 50 eigenspace features were selected to train SVM classification models with an RBF kernel function and different multi-class coding methods included one vs one, one vs all, Dense Random, and Sparse Random. Based on the results, we conclude that the classification system trained using the dataset with image resolution of 16×16 pixels and by Dense Random multi-class SVM coding method is a remarkable classifier to recognize various cultivars of Anthurium flower, with the classification accuracy of 99.5% and computation time of 0.4 s for each query image. Also, according to the algorithm computation time, it is possible to use a computer vision system equipped with such algorithms for real-time classification of flowers and cultivar recognition purposes.

References

Barajas-Garcia, C., S. Solorza-Calderon, and J. Alvarez-Borrego. 2016. Classification of fragments of objects by the Fourier masks pattern recognition system. *Optics Communications*, 367: 335–345.

Belhumeur, P. N., J. P. Hespanha, and D. J. Kriegman. 1997. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 19(7): 711–720.

Celikel, F. G., and Y. Karaaly. 1995. Effect of preharvest factors on flower quality and longevity of cut carnations (Dianthus caryophyllus L.). *Acta Horticulturae*, 405: 156–163.

Cevikalp, H., M. Neamtu, M. Wilkes, and A. Barkana. 2005. Discriminative common vectors for face recognition. *The IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(1): 4–13.

Du, J., X. Wang, and G. Zhang. 2007. Leaf shape based plant species recognition. *Applied Mathematics and Computation*, 185(2): 883–893.

Dutt, V., V. Chaudhry, and I. Khan. 2012. Pattern recognition: an overview. *American Journal of Intelligent Systems*, 2(1): 23–27.

Edizkan, R., H. Cevikalp, and H. S. Yavuz. 2013. DCV-based face recognition system and its application on the embedded development board. *AWER Procedia Information Technology and Computer Science*, 4: 734–738.

Fernandes, A. P., M. C. Santos, S. G. Lemos, M. M. C. Ferreira, A. R. A. Nogueira, and J. A. Nobrega. 2005. Pattern recognition applied to mineral characterization of Brazilian coffees and sugar-cane spirits. *Spectrochimica Acta Part B*, 60(5): 717–724.

Gaidhane, V. H., Y. V. Hote, and V. Singh. 2014. An efficient approach for face recognition based on common eigenvalues. *Pattern Recognition*, 47(5): 1869–1879.

Heymans, B. C., J. P. Onema, and J. O. Kuti. 1991. A neural network for opuntia leaf-form recognition. In *Proc. International Joint Conference on Neural Network,* 2116–2121. Singapore, Singapore, 18-21 November.

Hsu, C. W., and C. J. Lin. 2002. A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks*, 13(2): 415–425.

Kadir, A., L. E. Nugroho, A. Susanto, and P. I. Santosa. 2011. Leaf classification using shape, color, and texture features. *International Journal of Computer Trends and Technology*, 3(3): 256–263.

Leopold, E., and J. Kindermann. 2007. Content classification of multimedia documents using partitions of low-level features. *JVRB-Journal of Virtual Reality and Broadcasting*, 3(2006): no. 6. URL: http://nbn-resolving.de/urn:nbn:de:0009-6-7607.

Lu, X. 2003. Image analysis for face recognition. In *Proc. Personal Notes*, 1–37. Department of Computer Science and Engineering Michigan State University, May 2003.

MathWorks. 2015. *Statistics and Machine Learning Toolbox: User's Guide (R2015a)*. Natick, Massachusetts, United States: The MathWorks, Inc.

Mazanec, J., M. Melisek, M. Oravec, and J. Pavlovicova. 2008. Support vector machine, PCA and LDA in face recognition.

Journal of Electrical Engineering, 59(4): 203–209.

Nallammal, N., and V. Radha. 2012. Performance evaluation of face recognition based on pca, lda, ica and hidden markov model. In *Data Engineering and Management,* eds. R. Kannan, F. Andres, 96–100. Berlin, Heidelberg: Springer.

Oravec, M., M. Beszeded, and G. Rozinaj. 2008. Detection and recognition of human faces and facial features. In *Speech, Audio, Image and Biomedical Signal Processing Using Neural Networks,* eds. B. Prasad and S. R. M. Prasanna, 283–306. Germany: Springer-Verlag.

Pan, J., and Y. He. 2008. Recognition of plants by leaves digital image and neural network. In *2008 International Conference on Computer Science and Software Engineering,* 906–910. Hubei, China, 12-14 December.

Priya, C. A., T. Balasaravanan, and A. S. Thanamani. 2012. An efficient leaf recognition algorithm for plant classification using support vector machine. In *2012 International Conference on Pattern Recognition, Informatics and Medical Engineering (PRIME),* 428–432. Salem, Tamilnadu, India, 21-23 March.

Raghavendra, U., P. K. Mahesh, and A. Gudigar. 2012. A novel face recognition method using pca, lda and support vector machine. In *Proc. 2nd International Conference on Computer Science and Information Technology*, 241–249. Bangalore, India, 2-4 January.

Robert, L. G. 1965. Machine perception of three-dimensional solids. In *Optical and Electro-Optical Information Processing*, eds. J. T. Tippett et al., 159–197. Cambridge, MA.: MIT Press.

Satti, V., A. Satya, and S. Sharma. 2013. An automatic leaf recognition system for plant identification using machine vision technology. *International Journal of Engineering Science and Technology*, 5(4): 874–879.

Sharafi, S., S. Fouladvand, I. Simpson, and I. A. B. Alvarez. 2016. Application of pattern recognition in detection of buried archaeological sites based on analysing environmental variables, Khorramabad Plain, West Iran. *Journal of Archaeological Science: Reports*, 8(1): 206–215.

Subramanian, K. 2014. Face recognition using eigenface and support vector machine. *International Journal of Innovative Research in Computer and Communication Engineering*, 2(7): 4974–4980.

Wu, S., F. Bao, E. Xu, Y. Wang, Y. Chang, and Q. Xiang. 2007. A leaf recognition algorithm for plant classification using probabilistic neural network. In *Proc. 2007 IEEE International Symposium on Signal Processing and Information Technology*, 11–16. Giza, Egypt, 15-18 December.

Yu, H., and J. Yang. 2001. A direct LDA algorithm for high-dimensional data with applications to face recognition. *Pattern Recognition*, 34(10): 2067–2070.

Zhang, G., X. Wang, D. Huang, Z. Chi, Y. Cheung, and J. Du. 2004. A hypersphere method for plant leaves classification. In *Proc. 2004 International Symposium on Intelligent Multimedia, Video and Speech Processing (ISIMP 2004)*, 165–168. Hong Kong, China, 20-22 October.

Zhang, H., P. Yanne, and S. Liang. 2011. Plant species classification using leaf shape and texture. *American Journal of Engineering and Technology Research*, 11(9): 2025–2028.

Zhang, Y., and L. Wu. 2012. Classification of fruits using computer vision and a multiclass support vector machine. *Sensors*, 12(9): 12489–12505.

Zhao, W., R. Chellappa, P. J. Philips, and A. Rosenfeld. 2003. Face recognition: a literature survey. *ACM Computing Surveys,* 35(4): 399–458.