

# Mobile Plant Classification

Diploma thesis of

**Dragos Constantin**

At the faculty of Computer Science  
Institute for Anthropomatics

Advisors:

Dr.-Ing. Hazım Kemal Ekenel,  
Dipl.-Inform. Tobias Gehrig

Duration: 01. February 2012 – 31. July 2012

---

Computer Vision for Human-Computer Interaction Research Group  
Institute for Anthropomatics  
Karlsruhe Institute of Technology  
Title: Mobile Plant Classification  
Author: Dragos Constantin

Dragos Constantin  
35 Berliner Strasse  
Karlsruhe  
dragos.constantin@outlook.com

# Statement of authorship

I hereby declare that this thesis is my own original work which I created without illegitimate help by others, that I have not used any other sources or resources than the ones indicated and that due acknowledgement is given where reference is made to the work of others

Karlsruhe, 31 July 2012

.....  
(Dragos Constantin)



# Abstract

The purpose of this work is to prototype an automated plant recognition system to help botanists in the task of plant identification. Another aim was to optimise such a system for performance and execution speed for easy portability to mobile devices, while improving the recognition quality of state-of-the-art methods.

The system is based on feature extraction and classification methods applied on leaf images, more specifically Angular Radial Transform (ART), Fourier Descriptors (FD) and Scale Invariant Feature Transform-based Bag-of-Words (SIFT-BoW) as region, contour and local leaf features respectively. The feature vectors are classified by means of Random Forest and Nearest Neighbour classifiers. Tests were conducted on large datasets, with focus on the Pl@ntLeaves dataset from the ImageCLEF 2011 plant identification task. Although FD and SIFT based systems have already been evaluated for this task, in similar but not identical methodologies, we test for the first time the performance of ART and Random Forests on plant identification. Weighted confidence based late fusion was successfully used to classify images based on multiple feature vectors. Optimizations have been proposed for shape-based recognition with focus on the ART implementation, reducing the required computational time by means of important feature selection. Importance is computed either from the intrinsic Random Forest values or from discriminant potential of features.

Results of comparisons with state-of-the-art methods are very favorable for the proposed system when classifying leaf images on uniform backgrounds. Recognition rates on unconstrained natural plant photographs, however, were average in comparison with state-of-the-art methods and generally too low to be practical. Although ART and FD, as shape descriptors alone, perform better than the state-of-the-art, the importance of local features has been highlighted, as it noticeably improves system performance. Of the two classifiers tested, Random Forests was selected as the better one, proving better generalisation, classification times and scalability than Nearest Neighbour. Execution time results indicate the system proposed is indeed fast enough to run in reasonable times on slower devices.



# Kurzzusammenfassung

Durch die rapiden Vortschritte bei mobiler Rechenleistung können Aufgaben der Entscheidungsfindung, die bis jetzt von Menschen übernommen werden mussten, nun mit Hilfe des Computers gelöst werden. Zu diesen Aufgaben gehört auch die Identifikation von Pflanzen. Bisher mussten Botaniker Pflanzen manuell anhand bestimmter Merkmale klassifizieren. Die Blattformen gehören dabei zu den wichtigsten Unterscheidungsmerkmalen, die eine Pflanzenart definieren. Diese Arbeit stellt ein effizientes System vor, das Pflanzen in Bildern ihrer Blätter effizient erkennen kann. Das System umfasst sowohl Methoden zur Identifikation von Blättern in Bildern, die vor einem gleichmäßigen Hintergrund fotografiert wurden, als auch in solchen, die in natürlicher Umgebung aufgenommen wurden. Bei der Identifikation werden Methoden aus den Bereichen des maschinellen Sehens und der Bildverarbeitung zur Gewinnung von Form und lokalen Deskriptoren, die von überwachten Klassifikatoren aus dem Bereich des maschinellen Lernens klassifiziert werden, eingesetzt. Mittels Bildsegmentierung in Bildern mit gleichmäßigem Hintergrund erhält man die Form der Blätter. Merkmale dieser Form werden durch die Angular Radial Transform (ART) und Fourier Deskriptoren (FD) gewonnen. Parallel dazu, werden beide Bildarten - vor gleichmäßigem Hintergrund und in natürlicher Umgebung - mit der Scale Invariant Feature Transform (SIFT) verarbeitet und in einer Bag-of-Words (BoW) Repräsentation kodiert. Die erhaltenen Merkmalsvektoren dieser Klassifizierung werden durch Late Fusion fusioniert, um die abschließende Ausgabe des Systems zu generieren und die Leistung von Random Forests und Nearest Neighbours zu vergleichen. Strategien zur Effizienzsteigerung basieren auf einer Dimensionsreduktion des Merkmalsraums durch Auswahl der geeignetsten Merkmale.

Ausführliche Experimente wurden auf drei Datensätzen, deren wichtigster Pl@ntLeaves aus dem ImageCLEF 2011 Plant Identification Task ist, ausgeführt. Im Vergleich zu state-of-the-art Anwendungen erreicht das hier vorgestellte System sehr gute Ergebnisse bei der Erkennungsleistung - also Genauigkeit und Geschwindigkeit. Die Ergebnisse zeigen die Wichtigkeit von lokalen Merkmalen als vervollständigende Deskriptoren zu den Formmerkmalen - sogar in Bildern mit gleichmäßigem Hintergrund. Die Arbeit wird mit einer Analyse der Effizienz und der Skalierbarkeit des Systems, sowohl auf langsameren Endgeräten als auch mit größeren Datensätzen, abgeschlossen.





## List of abbreviations

<b>1NN</b>	Nearest Neighbour Classifier
<b>ART</b>	Angular Radial Transform
<b>BoW</b>	Bag of Words
<b>CART</b>	Classification and Regression Tree
<b>CEDD</b>	Color and Edge Directivity Descriptor
<b>CDA</b>	Canonical Discriminant Analysis
<b>CCH</b>	Circular Covariance Histograms
<b>DFT</b>	Discrete Fourier Transform
<b>DEG</b>	Maximum/Average Degree Descriptor
<b>DoG</b>	Difference of Gaussians
<b>DT</b>	Decision Tree
<b>EM</b>	Expectation Maximisation
<b>EOH</b>	Edge Orientation Histograms
<b>FD</b>	Fourier Descriptor
<b>FDA</b>	Functional Data Analysis
<b>FFT</b>	Fast Fourier Transform
<b>HOG</b>	Histogram of Oriented Gradients
<b>HSL</b>	Hue Saturation Lightness
<b>IDSC</b>	Inner-Distance Shape Context
<b>KNN</b>	K-Nearest Neighbours Classifier
<b>LDA</b>	Linear Discriminant Analysis
<b>LSH</b>	Locality Sensitive Hashing
<b>MSER</b>	Maximally stable extremal regions
<b>RF</b>	Random Forests Classifier
<b>RGB</b>	Red Green Blue
<b>RIT</b>	Rotation Invariant Points
<b>RMMH</b>	Random Maximum Margin Hashing
<b>SIFT</b>	Scale Invariant Feature Transform
<b>SVM</b>	Support Vector Machine



# Contents

<b>Statement of authorship</b>	<b>iii</b>
<b>Abstract</b>	<b>v</b>
<b>Kurzzusammenfassung</b>	<b>vii</b>
<b>List of abbreviations</b>	<b>ix</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation . . . . .	2
1.2 Goals . . . . .	3
1.3 Thesis overview . . . . .	3
<b>2 Related Work</b>	<b>5</b>
2.1 Leafsnap - Searching the World's Herbaria . . . . .	5
2.1.1 Segmentation . . . . .	6
2.1.2 Inner Distance Shape Context . . . . .	6
2.1.3 Results . . . . .	7
2.2 ImageCLEF . . . . .	9
2.2.1 IFSC/USP - Degree measures on small-world complex networks . . . . .	11
2.2.2 INRIA - Directional Fragment Histogram and Random Maximum Margin Hash . . . . .	13
2.2.3 LIRIS - Active Polygon models . . . . .	14
2.2.4 SABANCI OKAN - Mathematical morphology features . . . . .	16
<b>3 Theoretical Principles</b>	<b>17</b>
3.1 Segmentation . . . . .	18
3.1.1 Derivative based histogram thresholding . . . . .	22
3.1.2 Clustering based histogram thresholding . . . . .	23
3.2 Shape Based Feature Extraction . . . . .	24
3.2.1 Fourier Descriptors . . . . .	24
3.2.2 Angular Radial Transform - ART . . . . .	27
3.2.3 Complex network maximum degree descriptor . . . . .	30
3.3 Local Feature Extraction . . . . .	32
3.3.1 Keypoint detection and SIFT descriptor extraction . . . . .	32
3.3.2 Bag-of-Words model . . . . .	35

---

3.4	Classification . . . . .	36
3.4.1	Closest cluster center classification . . . . .	36
3.4.2	K-Nearest Neighbours classification . . . . .	37
3.4.3	Random Forests . . . . .	39
3.4.4	Attribute selection . . . . .	41
3.4.5	Early / Late feature fusion . . . . .	42
<b>4</b>	<b>Methodology</b>	<b>43</b>
4.1	System design . . . . .	44
4.1.1	Segmentation . . . . .	46
4.1.2	Shape descriptor extraction . . . . .	46
4.1.3	Local feature extraction . . . . .	48
4.1.4	Attribute selection . . . . .	49
4.1.5	Classification . . . . .	50
4.2	Optimisations . . . . .	52
4.2.1	ART optimisation . . . . .	52
4.2.2	Random Forests multi-threading cross-validation . . . . .	53
<b>5</b>	<b>Evaluation</b>	<b>55</b>
5.1	Datasets . . . . .	55
5.1.1	MPEG7 CE-1 . . . . .	55
5.1.2	Plummers Island 2011 . . . . .	56
5.1.3	Swedish leaves . . . . .	57
5.1.4	ImageCLEF Pl@ntLeaves 2011 . . . . .	57
5.2	Comparison with IDSC . . . . .	59
5.2.1	Experiment setup . . . . .	59
5.2.2	Results . . . . .	60
5.3	Performance on ImageCLEF 2011 Pl@ntLeaves dataset . . . . .	61
5.3.1	Scan and Scan-like image training . . . . .	62
5.3.2	Photograph training . . . . .	65
5.3.3	Testing . . . . .	66
5.4	Attribute selection and optimisations . . . . .	71
<b>6</b>	<b>Conclusion</b>	<b>77</b>
	<b>Bibliography</b>	<b>79</b>

# 1. Introduction

Under circumstances of fast species extinction mainly due to human development, botanists face the daunting task of researching as many species of plants as possible before they go extinct. Such research is mainly characterized by recording species distribution and, more importantly, the evolution of this distribution over time. During this process, botanists are faced with one main recurrent task: identifying plant species on the field and recording relevant information such as local growth density and mutations. Such activities are nowadays done manually, with little to no help from computers or automated methods. Increasing the species identification speed would thus greatly benefit botanical surveys, allowing for more efficient and often sampling of vegetation. However, the benefits are not only limited to the recording of endangered species: the increasing rate and affordability of global transportation has led to the introduction of foreign plant species which can harm the local environmental balance. Keeping track of these species is very important for the future of local agriculture and following their evolution is vital for a healthy ecosystem.

Until recently, there was a lack of digital applications to help accomplish the aforementioned surveys, but the boom of mobile computing (laptops, smartphones, tablets, etc) has opened new possibilities in computer aided tools. Plant identification, for instance, has always been a particularly time consuming task, as very subtle features may differentiate members within the same families. Although plants have many features that aid in identifying the species, such as dimension, branch shape and area of development, one of the most defining features is their leaf. While identifying leaves, even experienced botanists often rely on dichotomous feature trees in order to correctly determine the plant species, a process which can be long and painstaking. Leaf features present in such trees vary in shape, color, and vein patterns but shape and edge characteristics are omnipresent. The importance of leaf shape as a defining species feature has been acknowledged by the scientific community and has been the focus of many publications describing automated recognition methods:[BCJ<sup>+</sup>08, WBX<sup>+</sup>07, YAT11, CTM<sup>+</sup>11] and others.

Our purpose is to offer an automated plant recognition system to aid botanists in quickly identifying plant species on the field. We propose a system in which the person working on the field would simply take a photo of a leaf and automatically receive a short list of the

most probable plant species, together with standard images of their leaves for confirmation. This list would be ordered by relevance and would contain about 10 species, all displayed at once, with the possibility of extending it further. The botanist would then visually decide the correct species. This would require considerably less effort than navigating through a feature tree, as humans are able to process visual information much faster than text. To achieve this, we shall focus on leaf shape information, both general shape and edge details. We thus reduce the problem of plant species identification to that of automated shape extraction, recognition and classification. We also test common local feature extraction methods and analyse their potential for complementing shape information.

Rendering the identification process automatic is a challenging task on many levels, as both variances in leaf shapes - such as color and age - and those in photographs - such as lighting, rotation and background - need to be taken into account to provide a robust classification.

## 1.1 Motivation

Motivation for the current thesis comes from two main directions.

Firstly, a collaboration with the Botanical Institute from KIT, that should make the first steps towards simplifying local plant data acquisition. In parallel with the prototyping of an automated species recognition system, work has been done in collecting about 2000 high quality images from 150 species from South-Western Germany.

Secondly, challenges to the computer vision community have been presented in both [BCJ<sup>+</sup>08] and the ImageCLEF 2011 plant identification task [GBJ<sup>+</sup>11], to find the best algorithm suited for plant species identification. The algorithms proposed in both publications represent the current state-of-the art, the system from [BCJ<sup>+</sup>08] having evolved since 2008 into a smartphone application that provides plant identification services to the general public. Throughout this thesis we will judge the suitability of such algorithms not only by their capacity of delivering quality results but also by their execution time and ability to run on slow devices. As it is often the case, there is a trade-off between the two.

## 1.2 Goals

Our main goal is to define an automated plant recognition system, focusing on real-world usability. We aim to find a balance point between precision and speed, preferably increasing both in comparison with the state-of-the-art.

Botanical field work is often done in remote places in which phones barely have GSM signal and an Internet connection is unavailable. Due to algorithm processing requirements, plant identification methods, such as the Leafsnap application from [BCJ<sup>+</sup>08], require a constant Internet connection in order to offload the processing from the portable device to a powerful server. This has its inconveniences on which we wish to improve.

Our main focus will be on the following:

- **Performance** - we wish to improve on the results of current state-of-the-art methods
- **Portability** - the system should be easily portable, with good down-scalability computational-wise, without loss of performance so that it can provide good results on modern portable devices.
- **Robustness** - small variations in leaf shape, photographing style and background should not affect the outcome of the system.
- **OS independence and Speed** - the system will be written as much as possible from scratch in C++ in order to achieve best performance for the given task. Library dependencies will be greatly limited so that the current system would be easily integrated in a multitude of operating systems such as Linux, MS. Windows, iOS or Android.

## 1.3 Thesis overview

In the *Related Work* 2 section we will present the current proposed methods for plant recognition. We present in detail five of the state-of-the-art methods, which achieve very good results with different approaches. Particular attention will be given to the ImageCLEF plant recognition task from 2011, as it represents a recent and solid basis for comparing different methods on a level playing field.

The theoretical and mathematical underlying principles of the current work will be described in *Theoretical Principles* 3, together with other important notions necessary to support the correctness of this work.

The *Methodology* 4 section will connect the mathematical principles into the current system's process and detail the system architecture, its implementation and optimizations. The parameters and reasoning behind implementation choices of the theoretical principles are described in sufficient detail to allow the reproduction of this work's efficient implementation.

Finally, the *Evaluation* 5 section will introduce the datasets used for testing and detail the performance of the proposed system, analysing results on ImageCLEF data, comparing with state-of-the-art and related works, while describing how general system configuration affects performance. We also analyse optimisation methods for fast execution times of the system.





## 2. Related Work

This chapter is dedicated to the review of state-of-the-art methods and presentation of their results. We will focus our attention on one of the most ambitious plant recognition systems, developed in 2008 at the University of Columbia, and the more recent 2011 Image-CLEF plant identification task, which tested 8 different systems under the same conditions. We thus aim to provide a solid overview of the recent related work and simultaneously introduce the context of our work.

### 2.1 Leafsnap - Searching the World's Herbaria

In 2008, a joint effort between the University of Columbia and the Smithsonian Botanical Institute produced a plant recognition system described in [BCJ<sup>+</sup>08]. Upon publication of the system, a challenge was also sent to the computer vision community to improve the methods presented. The research finally took shape in the public LeafSnap smartphone application, which offers plant identification services over the Internet.

The solution presented is a two-device system in which a slow, portable device used in the field takes a photo of the plant leaf to be classified and sends that image over the Internet to a powerful server, which does the computations. The results of the classification are then forwarded back to the device, which displays species information. An assumption is introduced: a user would not only be interested in the first result of this classification; it is useful to display a list of results, ordered by the match probability between the sample image and multiple classes. In [BCJ<sup>+</sup>08], the precision metric that is most often cited is not the probability of obtaining the correct class in the first result, but the probability that the correct class is displayed in the first 10 results, meaning rank 10 correct classification. This is important from a usability point of view because a human can quickly refine results visually and choose the correct class.

From a theoretical point of view, the plant recognition is shape based, meaning that leaf photographs are reduced to binary images composed of foreground - leaf - and background. On the resulting foreground shape, inner-distance shape context descriptors are computed, described in [LJ08]. Because these descriptors are computed on specific points on the leaf

contour, the classifier needs to match them individually in order to determine similarity between shapes. In the following we present the system in more detail.

### 2.1.1 Segmentation

As previously mentioned, the algorithm is shape based and therefore needs a contiguously segmented image. To achieve this, the leaf image is transformed from RGB color space to HSL, from which only the saturation channel is used. This transformation is based on the assumption that the background will generally have less color than the leaf. It is mostly robust to shadows and general lighting changes but may be influenced by false white balance at image acquisition. The resulting image is then thresholded by means of a parametric Expectation-Maximisation - EM - algorithm as follows: two pixel distributions are assumed to exist, one for the background, one for the foreground. These distributions are assumed quasi-Gaussian, the background being close to black levels and the foreground being close to white levels. The EM algorithm iteratively searches for the gray levels which best fit these two distributions, updating the distributions with new values, then repeats. Once converged, a threshold is set at the intersection of these distributions, signifying the best background/foreground separation. Once the threshold is found, each pixel in the image that is smaller than the threshold will be considered background and each pixel greater than the threshold will be considered foreground.

### 2.1.2 Inner Distance Shape Context

Shape Context descriptors have been first introduced in [BMP02] and represent log-polar histograms of contour distribution as shown in 2.1.

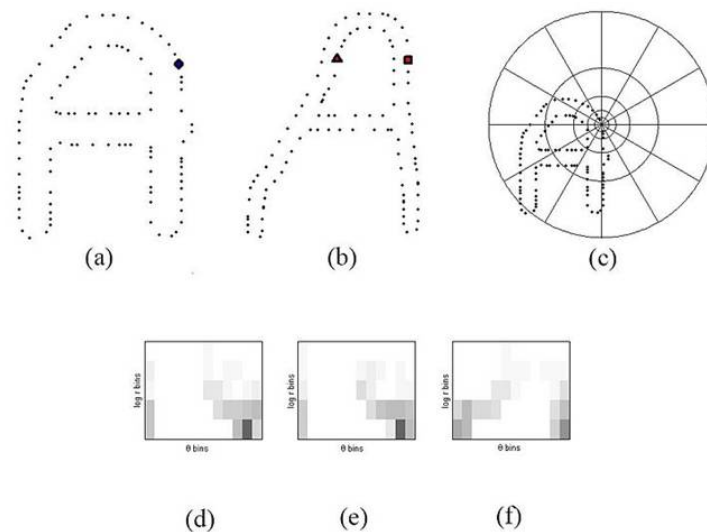


Figure 2.1: Shape Context Descriptor Overview as presented in [BMP02]

A contour is resampled to a fixed number of points. In each of these points, a histogram is computed such that each bin counts the number of sampled contour points that fall into

its space. Because the space distribution of each histogram bin is logarithmic on the radial axis, the descriptor has higher descriptive power in relation to contour points closer to it, effectively being a local shape descriptor. Figure 2.1 presents an overview of the Shape Context descriptor: Two shapes, in this case two  $A$  shapes have their contour sampled into points  $(a), (b)$ . An overlay of the log-polar histogram space is drawn in  $(c)$ . The histograms shown in  $(d), (e)$  and  $(f)$  match the circle, square, and, respectively, triangle shaped contour points, to prove that descriptors computed in similar points on similar shapes will provide close histograms.

The Inner Distance Shape Descriptor [LJ08] is based on essentially the same principle as the Shape Descriptor, but instead of using Euclidean distance and simple polar transformation as bin coordinate spaces, it uses the inner distance and the inner angle as the coordinate space. Both the inner distance and angle are visually exemplified in Figure 2.2.

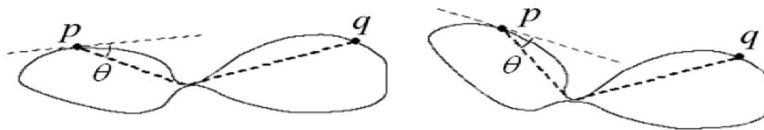


Figure 2.2: Inner Distance and Angle as described in [LJ08]

The inner distance between two points,  $p$  and  $q$  in the image, is the minimum path between the points inside of the shape. The inner angle  $\theta$ , is the angle this path creates with the contour tangent at the descriptor point. As the IDSC descriptor is a histogram, distance between two descriptors has been defined through the  $\chi^2$  statistic. In order to classify two shapes using such descriptors and define their similarity, one would normally have to match all possible pairs of descriptors on the contour, resulting in  $O(n^3)$  complexity, where  $n$  is the number of points on the contour. Through dynamic programming, the complexity has been reduced to  $O(n^2)$ . The classification process is therefore computationally intensive and does not scale well with large datasets.

### 2.1.3 Results

The IDSC descriptor has been tested in both [LJ08] and [BCJ<sup>+</sup>08] with the results we further describe. In [LJ08] the main focus was on general and articulated shape recognition, whilst in [BCJ<sup>+</sup>08] the application was focused only on plant recognition. From the presented results we retain only three, on databases that are still available today and can be considered a comparison basis.

These datasets, which are also presented in detail in Section 5.1, are:

1. MPEG7 CE-1: 70 general object classes with 20 binary images each, 1400 images in total
2. Swedish leaves dataset from Linköping University and the Swedish Museum of Natural History: 15 leaf classes with 75 images each,
3. Plummers Island dataset 2008: 157 leaf classes with an average of 30 images per class, 5013 images in total

The following results express the correct classification accuracy, meaning the number of correct results over the number of tested images.

Dataset	Precision	Evaluation metric	Evaluation method
MPEG7 CE-1	84.5%	Bullseye	leave-one-out cross-validation
Swedish Leaves	94.5%	Bullseye	leave-one-out cross-validation
Plummers Island	90%	Bullseye, rank 10	leave-one-out cross-validation

Table 2.1: Summary of IDSC results

A more detailed view of the Plummers Island dataset results is offered in [BCJ<sup>+</sup>08] under the form of the following precision graphic:

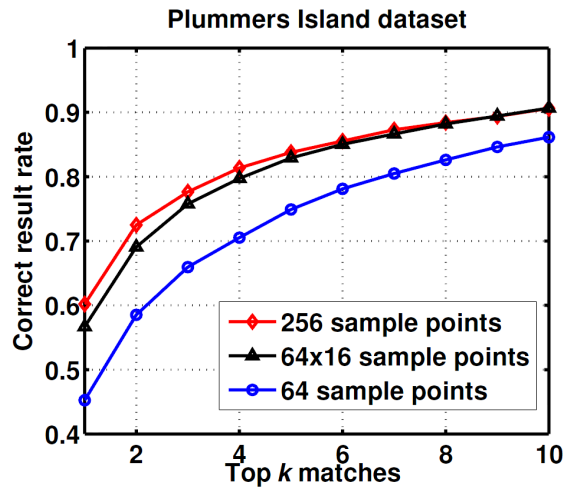


Figure 2.3: Bullseye precision of IDSC on Plummers Island dataset as described in [BCJ<sup>+</sup>08]

The result of a leave-one-out cross-validation evaluation represents the probability of obtaining a correct result in the first  $k$  matches. It means that although there is only a 60% chance of the first match being correct, there is a 90% chance of displaying a correct match in the first 10.

## 2.2 ImageCLEF

The ImageCLEF Plant Identification Task is part of the larger ImageCLEF task which aims to provide a context for image retrieval research exchange. ImageCLEF is itself part of the bigger Cross-Language Evaluation Forum, which focuses on multilingual and multi-modal information access evaluation. Other topics such as Medical Image Classification and Retrieval as well as Robot Vision have their own task in ImageCLEF.

The 2011 ImageCLEF Plant Identification task was approached with great interest, as it presents a solid benchmark basis for our system by means of a large and varied dataset, as well as results from a total of 8 participating groups. The entire training and testing datasets were made available on the ImageCLEF website (also found in [GBJ<sup>+</sup>11]), thus allowing us to test our performance against very recent similar systems.

The Pl@ntLeaves dataset is split into approximately 4000 training and 1400 testing images, each set further divided into three categories: scans, pseudo-scans and photographs, as shown in Figure 2.4. Details of the dataset and its structure can be found in Section 5.1.4.

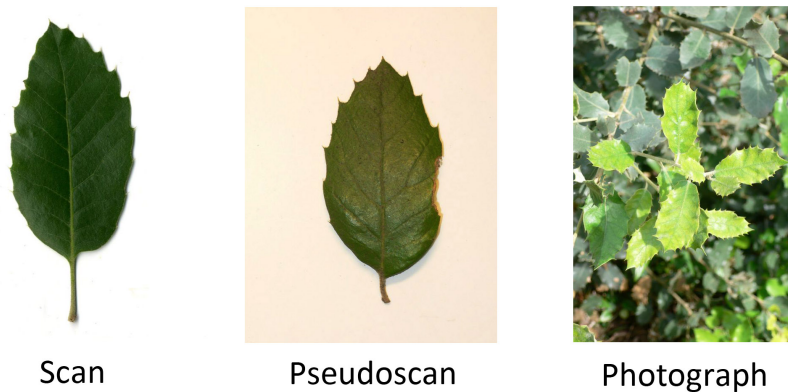


Figure 2.4: Sample of the three different image types for the species *Quercus Ilex*

The following table presents an overview of the methods used by the participants, as presented in [GBJ<sup>+</sup>11] and the corresponding publications:

Group	Best Score	Shape features	Local features	Classification
<b>IFSC/USP</b> [CFB11]	49.6%	Max/Avg degree	-	CDA + Naive Bayes
<b>INRIA</b> [GJY <sup>+</sup> 11]	44.9%	-	Hough, EOH, 2D Fourier	SVM → RMMH
<b>LIRIS</b> [CTM <sup>+</sup> 11]	43.7%	Active Polygons	-	NN
<b>Sabancı-Okan</b> [YAT11]	40.4%	FD	Texture, Color Moments	SVM
Kmimmis	18.4%	-	SIFT	K-NN
UAIC*	15.6%	-	CEDD+SIFT	SVM
RMIT	5.6%	-	GIFT	NN / DT
Daedalus	4.1%	-	SIFT	NN

\*UAIC was the only group trying to make use of image metadata such as GPS coordinates and plant taxon.

Table 2.2: Overview of plant recognition methods from ImageCLEF 2011

The Best Score mentioned above is an average over normalised scores in each image category: scan, pseudoscan and photos. The normalisation procedure used is detailed in [GBJ<sup>+</sup>11], as well as in this work, in Section 5.1.4. It is important to mention that IFSC/USP use manual intervention in the segmentation algorithm, hence allowing them to apply shape features on natural photographs, all other methods being fully automatic. Each group was allowed more than one run, the Table 2.2 describing only the results and configurations of the best runs from each group.

The following table presents the results of all the runs from the 8 groups, together with separate scores for each of the three image categories, in descending order of mean score:

Run id	Participant	Scans	Pseudoscans	Photographs	Mean
IFSC USP run2	IFSC	<b>0,562</b>	0,402	<b>0,523</b>	0,496
inria imedia plantnet run1	INRIA	<b>0,685</b>	0,464	0,197	0,449
IFSC USP run1	IFSC	0,411	0,430	<b>0,503</b>	0,448
LIRIS run3	LIRIS	0,546	0,513	<b>0,251</b>	0,437
LIRIS run1	LIRIS	0,539	<b>0,543</b>	0,208	0,430
Sabancı-okan-run1	SABANCI-OKAN	<b>0,682</b>	0,476	0,053	0,404
LIRIS run2	LIRIS	0,530	0,508	0,169	0,403
LIRIS run4	LIRIS	0,537	<b>0,538</b>	0,121	0,399
inria imedia plantnet run2	INRIA	0,477	<b>0,554</b>	0,090	0,374
IFSC USP run3	IFSC	0,356	0,187	0,116	0,220
kmimmis run4	KMIMMIS	0,384	0,066	0,101	0,184
kmimmis run1	KMIMMIS	0,384	0,066	0,040	0,163
UAIC2011 Run01	UAIC	0,199	0,059	0,209	0,156
kmimmis run3	KMIMMIS	0,284	0,011	0,060	0,118
UAIC2011 Run03	UAIC	0,0927	0,163	0,046	0,100
kmimmis run2	KMIMMIS	0,098	0,028	0,102	0,076
RMIT run1	RMIT	0,071	0,000	0,098	0,056
RMIT run2	RMIT	0,061	0,032	0,043	0,045
daedalus run1	DAEDALUS	0,043	0,025	0,055	70,041
UAIC2011 Run02	UAIC	0,000	0,000	0,042	0,014

Table 2.3: ImageCLEF 2011 normalized classification scores for each run and each image type. Best three scores in each category are highlighted

In the following we will take a closer look at the plant identification framework used by the first four groups.

### 2.2.1 IFSC/USP - Degree measures on small-world complex networks

The plant recognition procedure described by IFSC in [CFB11] relies on the novel idea of representing the leaf contour as a complex network, meaning a graph with non-trivial topological layout. As pre-processing, both scans and pseudoscans are segmented with the default Otsu method [Ots79], natural photographs of the plant being manually segmented. On the segmented images, contour detection is used and leaf contours are extracted.

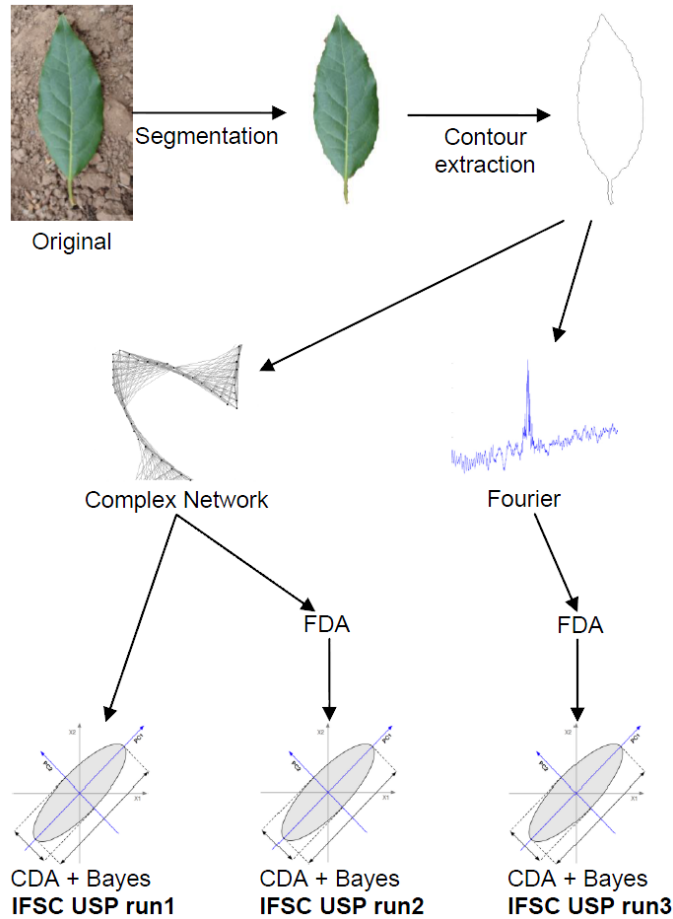


Figure 2.5: Overview of the IFSC/USP method as presented in [CFB11]

The contour is then represented as a graph: each point of the contour being a *node* and the Euclidean distance between two points representing the respective *edge* cost. This representation results in a distance matrix between nodes. The network is then iteratively thresholded with  $t_j$  for various distance values and the maximum degree and average degree of the graph nodes  $s_i$  for each respective threshold will be concatenated into a feature vector.  $t_j$  typically ranges between 0.25% and 92.5% of the maximum distance between any two nodes. The computed feature vector is composed of pairs of maximum and average degrees for each  $t_i$  thus having a dimension equal to twice the number of thresholds. Exact details of the descriptor are given in Section 3.2.3 as the feature has also been implemented and tested in this work. In one of the runs sent to ImageCLEF, Fourier Descriptors were used as a comparison with the Complex Network features.

In the second and best run, the feature vector has been transformed prior to classification by means of Functional Data Analysis - FDA. In FDA, the feature vector is considered to be a discrete representation of a continuous function. The purpose is to transform the feature vector from the feature space into a function space through means of interpolation. More specifically, the feature vector is interpolated using B-splines bases, the resulting basis coefficients constituting the new feature vector on which classification will be performed.

Lastly, classification is achieved by a Naive Bayesian classifier on canonical variables resulting from Canonical Discriminant Analysis - CDA. CDA, a specialization of Linear Discriminant Analysis, searches for canonical functions that maximise the ratio between inter-class and intra-class feature variability. CDA will create a new canonical space of dimensions equaling the number of classes minus one. Although the ImageCLEF dataset contains 71 classes, it has been noticed that only 10 of the 70 canonical variables account for 99.99% of feature variance. These 10 main components were used as features for the parametric Gaussian Naive Bayes classifier.



### 2.2.2 INRIA - Directional Fragment Histogram and Random Maximum Margin Hash

The INRIA group had two runs in ImageCLEF 2011, the first run based on local features and the second based on shape features, both using the same classification method. The best score was obtained by the first run, being 12 percentage points higher than the shape feature based method. Although both methods are described in [GJY<sup>+</sup>11], we will focus on describing the first one based on its performance and the rarity of high performance local feature based plant recognition methods in the literature.

The large scale local feature method consists of the following steps:

- Computation of color Harris keypoints
- Local feature extraction in a small vicinity around keypoints: histograms of the 2D Fourier transform, Hough transform and Edge Orientation respectively
- Local feature matching through Random Maximum Margin Hashing
- Spatial consistent matches filtering with a Random Sample Consensus algorithm based on a rigid transform model
- Classification through top-K rule

The local features are designed to complement each other: the Hough transform is expected to offer shape information, the Fourier transform would offer texture information and the Edge Orientation Histogram represents edge information such as leaf edge characteristics. The Harris keypoints are limited to 500 per image and the concatenation of features results in a 280 dimensional feature vector for each of these keypoints.

Random Maximum Margin Hashing - RMMH - is a recent data dependent hashing method described in [JB11]. Similar in a way to the more renowned Locality Sensitive Hashing - LSH, it is based on projecting feature vectors onto randomised vectors and binning them. If features fall often in the same bins, it is assumed they are close together in the feature space, therefore matching each other. The most important concept is that a high dimensional feature space is reduced to a much lower dimensional one, whose dimensions are represented by the number of random hash vectors. However, in contrast to LSH which randomly chooses the projection vector, the RMMH selects each projection vector based on the training dataset. Each vector is obtained by randomly partitioning the features in two classes, irrespective of their labels and then training an SVM on this binary partitioning. The resulting vector which splits these two random partitions with maximum margin is one of the RMMH vectors.

Using RMMH on local features gives a similarity measure between keypoints but not necessarily shapes, hence further filtering of the RMMH output with a rigid model is applied. A geometric model accounting for size, translation and rotation of the shape is used for filtering; the parameters are computed by means of the Random Sample Consensus algorithm, randomly selecting two matched keypoints and checking if their distances correspond to the model and updating the model accordingly.

Finally a query feature vector is labeled by voting on the top 10 returned training images ranked by geometrical consistency score.

### 2.2.3 LIRIS - Active Polygon models

The LIRIS group presented a model based leaf matching method with firm roots in botanical knowledge about leaf shape variations. [CTM<sup>+</sup>11] presents an identification system which actively tries to match standard shape polygons on images.

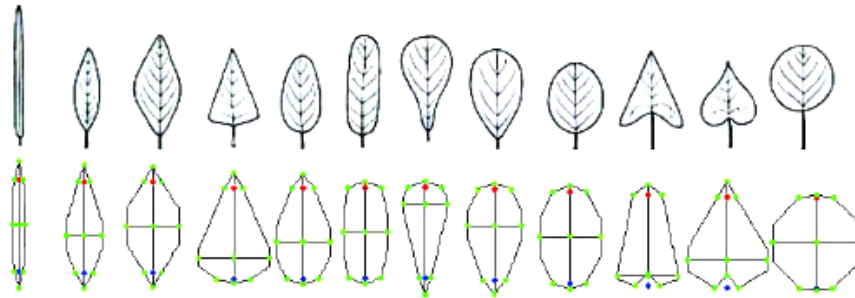


Figure 2.6: Main leaf shapes and their corresponding hand-tuned models as presented in [CTM<sup>+</sup>11]

All the above shapes are described by a single polygon model determined by the following parameters:

- $\alpha_B$ , opening angle at the base of the leaf
- $\alpha_T$ , opening angle at the tip
- $\omega$ , the relative maximal width
- $p$ , the relative position where this width is reached

The approach for leaf matching is very similar to that of active contours, but in this case the number of points is fixed and the possible transformations are limited by the main leaf shape, as shown in Figure 2.6. As with active contours, the polygon points climb the gradient slopes from the image, in search of stable equilibrium between model and gradient energies. Due to the existence of leaves with multiple lobes, the model simultaneously converges multiple polygons and then removes those with too much overlapping. The number of lobes is then reduced to 3, as experiments show this is sufficient.

Preprocessing of the images is applied under the assumption that leaves are generally centered and mostly vertical. A distance map is then computed in  $L^*a^*b^*$  color space in which each map pixel at a given position represents the color distance between central pixels and the image pixel at that position. The distance measure is a sum of normalised differences for each channel.

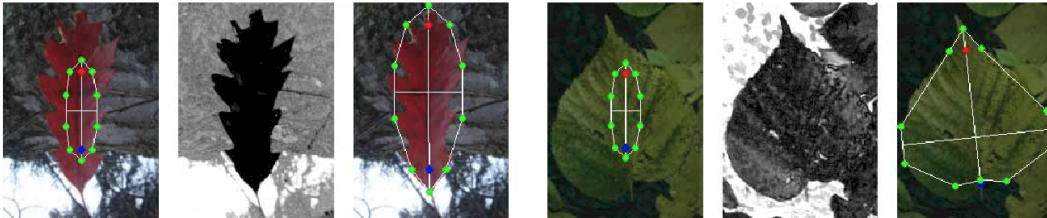


Figure 2.7: Two sample images, their respective distance maps and converged polygon models as presented in [CTM<sup>+</sup>11]

The Active Polygon Model is then applied on the difference map, convergence of the model (as shown above) resulting in knowledge about the general shape, size and position of the leaf in the image. From this polygon a new active contour is formed that quickly converges on the leaf edge gradient, offering a detailed leaf contour and information which can also be used to segment the image.

The final features that are extracted are the 4x3 polygon model parameters previously described, together with curvature variance and mean sampled at 4 different osculating circle dimensions. The mixed descriptor is very compact, being composed of only 20 features in total: 12 for general shape description and 8 for contour.

Classification is done using a nearest neighbour method on a weighted distance metric: each attribute of the feature vector is scaled separately for optimal performance.

The main advantage of this method is the extremely low dimensionality of the feature vector as well as the possibility of automatically segmenting natural plant photographs due to the strong *a priori* of the model.

### 2.2.4 SABANCI OKAN - Mathematical morphology features

The system presented by the SABANCI OKAN group in [YAT11] was of particular interest due to its high precision on scan type images. Similar to the INRIA method, this system has a score of over 68% on scans, which is 12 percentage points higher than any other system proposed in ImageCLEF 2011. However, the very low score of 0.5% on photograph types negatively affects their average score, placing them in fourth place.

The proposed system uses a total of 8 descriptors split into two main categories:

- Texture descriptors: Circular Covariance Histograms - CCH and Rotation Invariant Points - RIT
- Color moments: statistical moments generalised to a three channel image
- Shape descriptors: Fourier Descriptors, Width length/volume factor, Convexity measures, Basic Shape Statistics and Border Covariance

The exact details of the theoretical principles of all the above descriptors can be found in [YAT11, FHST05, AL09].

Descriptors were then fused into two separate feature vectors to train SVM classifiers:

- Shape feature vector: Composed of all the shape descriptors, including Fourier, but none of the others
- General feature vector: Composed of all the descriptors except Fourier

Kernelised SVMs with Radial Basis Functions were used as classifiers for each feature vector resulting in one class distance vector per feature vectors. The class distance vectors are then fed into another SVM with the purpose of finding the best late fusion parameters. The resulting 5 best classes from this classification are then reweighted by classification through a multi-class SVM. The cross validation results at each step of the classification process are described in the following table:

Stage	Accuracy (%)
Classifier using only shape features	71.46
Classifier using all features except FD	89.69
Classifier combination (late fusion)	90.10
After resolving ambiguities (re-classification)	93.64

Table 2.4: Cross-validation accuracies on scans and pseudoscans at different classification steps as described in [YAT11]

The cross-validation results indicate the importance of having both shape as well as texture and color descriptor information. The 18 percentage points jump in precision when using such information complementary to shape features is in itself more important than the 5 percentage points improvement through late fusion and re-classification. We note that classification processing time is likely high, as for each sample to be classified, an SVM is trained to avoid ambiguities.

### 3. Theoretical Principles

The system we describe in this work, as well as related works, is part of the general framework of visual pattern recognition problems and as such, follows the following procedure:

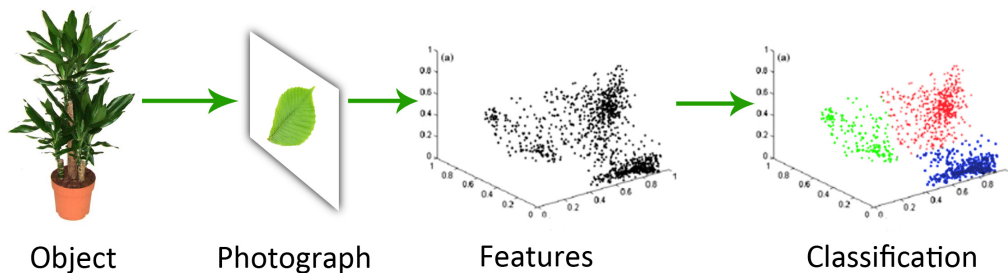


Figure 3.1: Process Overview

*Objects* with classes from the real world - plants and their species - are considered to be somehow similar inside a given class. They are recorded through a *pattern* - photograph - producing a non-ideal representation in pattern space. This representation is too high dimensional and variant to a multitude of factors thus requiring that lower dimensional *features* are extracted to better represent the object. The main assumption is that if objects are similar in the real world, relevant features will be close to one-another in feature space.

Once the features are extracted, a *classifier* has the task of attributing a class label to the object based on its features. This class will represent in our case the plant species from which the image and the respective features originated.

This framework has the advantage of reducing the dimensionality of the classification problem as well as providing invariance to image-related factors such as translation, resizing and lighting conditions if the respective invariant features are extracted.

Specializing the aforementioned framework to our problem, the following steps of the proposed process will be explained:

1. Segmentation
2. Shape based feature extraction
3. Local feature extraction
4. Classification

We assign a sub-section for each of these elements in which we describe the theoretical basis of the specific methods being used.

### 3.1 Segmentation

In order to use certain features, more specifically shape based descriptors, it is necessary to extract the leaf shape from the image. We are hence looking for a function which takes a three channel color image as input and outputs a binary image in which each pixel belonging to the leaf would have the maximum value and each pixel belonging to the background will have the minimum value, as shown in the following example:

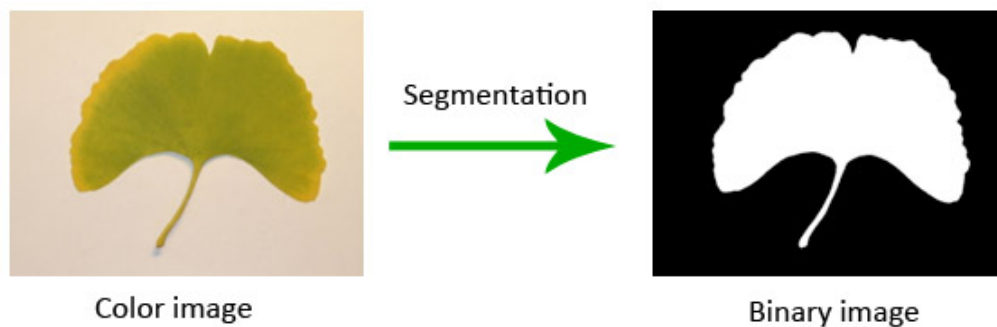


Figure 3.2: Segmentation example

In this work we will be looking at different ways of obtaining the segmentation functions based on thresholding, that is for a color pixel  $p(x, y)$  belonging to color image  $I$ , defined by each channel value  $p_1(x, y)$ ,  $p_2(x, y)$  and  $p_3(x, y)$  respectively, the segmentation function defining the segmented image  $S$  is as follows:

$$Seg : I \rightarrow S$$

$$Seg(x, y) = \begin{cases} 1, & \text{if } a \cdot p_1(x, y) + b \cdot p_2(x, y) + c \cdot p_3(x, y) > threshold \\ 0, & \text{else} \end{cases}$$

Where the  $(x, y)$  are pixel coordinates varying from  $(1, 1)$  to the image dimensions  $(nW, nH)$  and the three linear coefficients  $a, b$  and  $c$  represent weights for each respective image channel. We are then faced with the problem of finding discriminative color channels and the threshold value in order to define a good segmentation function.

The first assumption we make in order to find such a function is that the background of the leaf image is generally uniform and contains little color information. This assumption holds if we look at leaf image datasets such as those from ImageClef 2011, Plummers Island and the Swedish Leaf dataset presented in Section 5.1. Qualitatively analysing different color channels from color spaces such as RGB, L\*a\*b, HSL on these datasets, we have reached the conclusion that the Blue channel from RGB and the Saturation channel from HSL offer strong discriminative potential, as shown below:

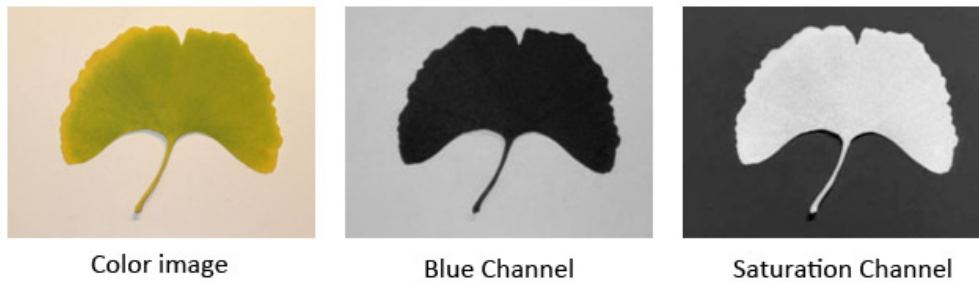


Figure 3.3: Leaf image together with discriminative color channels

The reason we did not only consider the saturation channel as it is done in [BCJ<sup>+</sup>08, CFB11] is that very often, complex leaves are held together by branches with low saturation levels resulting in non connected leaf areas. The Blue channel helps in this regard although it also introduces the risk of falsely segmenting leaf shadows. Considering the above, we specialise the segmentation function to

$$Seg : I \rightarrow S$$

$$Seg(x, y) = \begin{cases} 1, & \text{if } a \cdot (1 - Blue(x, y)) + b \cdot Sat(x, y) > threshold \\ 0, & \text{else} \end{cases}$$

Where both channels *Blue* and *Sat* are considered to have pixel values between 0 and 1. The linear combination of these channels has shown to offer good contrast in practice for  $(a, b)$  coefficient values between  $(0.5, 0.5)$  and  $(0.2, 0.8)$  depending on the balance we wish to achieve between sensitivity to shadows and correct segmentation of complex leaves with branches.

In order to achieve good segmentation performance, the *threshold* variable has to be meaningfully defined for each image. In this work we propose two efficient algorithms for threshold determination, both making the assumption that the composite gray level

channel defined by  $K(x, y) = a \cdot (1 - Blue(x, y)) + b \cdot Sat(x, y)$  is discriminative enough to separate the leaf and background into two, non overlapping gray level distributions. Considering  $K$  to be discretely represented on  $nG$  gray levels, both algorithms work on the histogram of  $K$ , defined as follows:

$$h(i) = \sum_{(x,y)=(1,1)}^{(nW,nH)} \mathbf{1}_{K(x,y)=i} \quad \text{with} \quad \mathbf{1}_{K(x,y)=i} = \begin{cases} 1, & \text{if } K(x, y) = i \\ 0, & \text{else} \end{cases}$$

Where  $i$  is a discrete gray level variable ranging from 0 - black to  $nG$  - white. In the case of most images, it is an 8 bit representation with values between 0 and 255. Based on the histogram, a small percentage of both the lowest and highest gray levels is clipped in order to improve general contrast and foreground/background gray level distance. The image is then transformed as described in the following algorithm:

---

**Algorithm 1** Black / White clipping
 

---

```

1:  $nTotalPixels \leftarrow imageWidth \cdot imageHeight$ 
2:  $nBlackPixels \leftarrow nTotalPixels \cdot blackClipProcentage$ 
3:  $nWhitePixel \leftarrow nTotalPixels \cdot whiteClipProcentage$ 
4:  $pixelCount \leftarrow 0$ ,  $grayLevel \leftarrow 0$ 
5: while  $pixelCount < nBlackPixels$  do
6:    $pixelCount \leftarrow pixelCount + h(grayLevel)$ 
7:    $grayLevel \leftarrow grayLevel + 1$ 
8: end while
9:  $newBlackLevel \leftarrow grayLevel$ ,  $pixelCount \leftarrow 0$ ,  $grayLevel \leftarrow nG$ 
10: while  $pixelCount < nWhitePixels$  do
11:    $pixelCount \leftarrow pixelCount + h(grayLevel)$ 
12:    $grayLevel \leftarrow grayLevel - 1$ 
13: end while
14:  $newWhiteLevel \leftarrow grayLevel$ ;
15: for all image pixels  $p \in K$  do
16:   if  $p > newWhiteLevel$  then
17:      $p \leftarrow nG$ 
18:   else
19:     if  $p < newBlackLevel$  then
20:        $p \leftarrow 0$ 
21:     else
22:        $p \leftarrow (p - newBlackLevel) / (newWhiteLevel - newBlackLevel)$ 
23:     end if
24:   end if
25: end for

```

---

With default White and Black clipping percentages being 5% and 10% respectively. The effect of the algorithm is shown on a sample histogram in Figure 3.4.



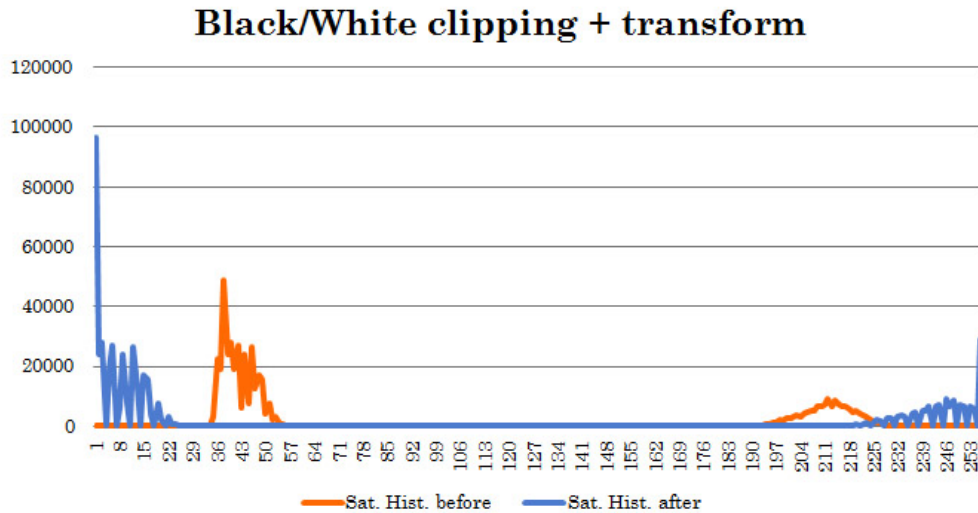


Figure 3.4: Effects of black/white clipping on the histogram of the  $K$  channel of a leaf image. Orange shows the histogram before clipping, while blue shows it after.

The main effect of the aforementioned transformation is that general image contrast is higher, providing better separation between foreground and background pixels. The darkest 10% pixels now become black, while the lightest 5% pixels become white. Values between the newly identified black and white graylevels are then linearly stretched between 0 and  $nG$ .

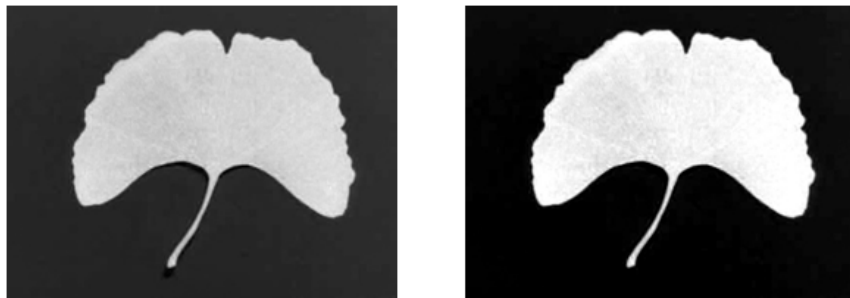


Figure 3.5: An example of the effects of clipping on the  $K$  channel of a leaf image.

Figure 3.5 shows the result of our clipping algorithm on the extracted  $K$  channel of a leaf image. We notice how the light-gray leaf against a dark-gray background is transformed into a nearly perfect white leaf against a black background, thus increasing the contrast and providing a better image for thresholding segmentation.

### 3.1.1 Derivative based histogram thresholding

Derivative based histogram thresholding makes the assumption that the image is already clipped as described before and starting from the lowest gray level, searches for the point where the variation of  $h(i)$  is negative and close to null, meaning the first minimum found after descending a slope. This minimum generally matches values just above the highest gray level of background pixels.

As derivative methods are very prone to noise and the clipping transformation also tends to produce jagged histograms,  $h(i)$  is firstly smoothed by means of a sliding window. It is on this smooth histogram that the derivative method is applied, resulting in a stable threshold.

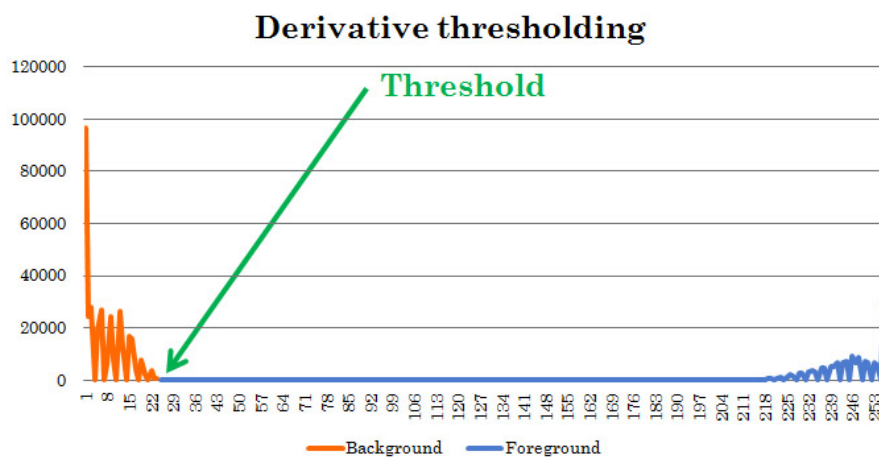


Figure 3.6: Threshold as defined by derivative method

Although this method is both fast and stable, it deteriorates rapidly when there is strong overlapping between foreground and background pixel distributions. All segmentation methods are sensitive to such imperfections but the presented derivative method may especially converge at bad gray levels, such as perfect white, when there is no distinguishable minimum. However, a solution to this problem would be to limit the minimum search to a default value, as a last resort if convergence is not reached.

### 3.1.2 Clustering based histogram thresholding

Another way of finding a significant threshold is by applying a clustering algorithm on the histogram values. In [BCJ<sup>+</sup>08, BCGM98] a parametric Expectation Maximisation algorithm was used to define two clusters: one for the background and one for the foreground. The threshold is then defined as the mid-distance between the central points of those clusters. In a similar fashion we apply a 1 dimensional version of the K-means algorithm, initialising it with two cluster centers, one at 0 and one at  $nG$  and letting it converge on the optimal distribution. The K-means algorithm is one of the simplest non-parametric clustering methods and represents a specialisation of EM algorithms. Our initial threshold will be at mid distance between converged cluster centers as shown in Figure 3.7.

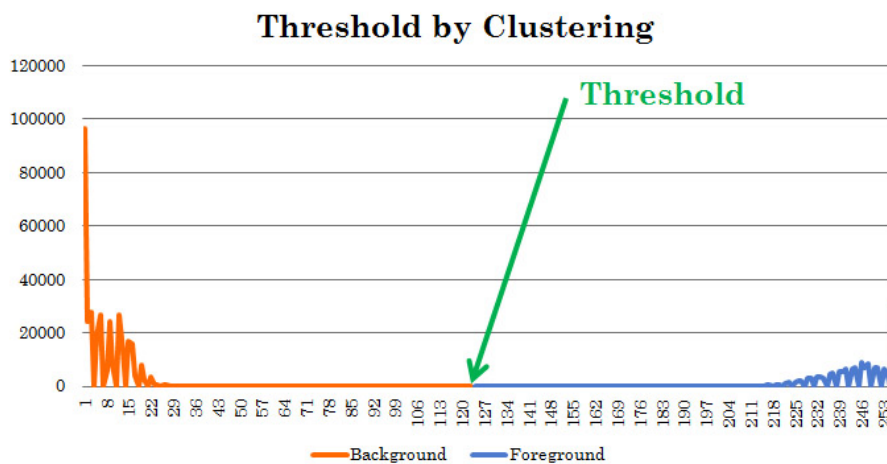


Figure 3.7: Threshold as defined by the clustering method

If segmentation results are not satisfying and a bias is observed in the output, either towards segmenting background pixels as foreground or vice-versa, the threshold can be modified by computing a weighted mean of the two cluster centers:

$$threshold = \frac{ClusterCenter_1 + biasCorrection * ClusterCenter_2}{1 + biasCorrection}$$

This method will pull the threshold towards foreground or background clusters as necessary.

## 3.2 Shape Based Feature Extraction

In this section we will present the shape descriptors used in this work. Shape features represent a class of descriptors whose sole purpose is to describe the shape of an object, with no other information about color or texture. They usually require the image to be segmented ([BCJ<sup>+</sup>08, CFB11, WBX<sup>+</sup>07, BB08, KNSS11]), meaning to be reduced only to its shape, although active methods, such as the Active Polygon Method described in [CTM<sup>+</sup>11], converge on the shape of an object while avoiding the need for segmentation. Shape descriptors can be split into two categories: region based and contour based. While contour based descriptors only take into account pixels from the edges of the shape, region based descriptors take into account all the pixels of the shape. In this work we present two shape descriptors aimed to complement each other:

- Fourier Descriptors: features containing the frequencial information of the contour
- Angular Radial Transform Descriptors: features encoding the shape regions in polar coordinate frequencial bases

The reasoning behind this choice is that leaves have particular shape variances and that shape features described in botanical dichotomous trees generally fall into two classes: the general leaf shape (rounded, teardrop, lobed, etc.) and the edge type (smooth, jagged, irregular, etc.). These two classes match well with the definitions of regional and contour descriptors and while ART offers region based information, contour based FDs have the potential of expressing the high frequency detail of leaf edges.

For comparison reasons and due to the simplicity of the Maximum / Average Degree Descriptor, introduced at ImageCLEF 2011 in [CFB11], it is also presented in detail and implemented as part of this work, although it is not part of the proposed plant recognition system.

### 3.2.1 Fourier Descriptors

Frequencial analysis of shapes through Fourier transform is one of the oldest and most often quoted methods of shape analysis and recognition. We can cite few of many publications centered around shape contour Fourier transform together with their applications: shape analysis [ZR72], character recognition [TR94], shape coding [CB84], shape classification [KSP95] and shape retrieval [ZL01]. Most modern shape classification methods benchmark their novel shape descriptors against FDs in order to prove their effectiveness [LJ08, CFB11, BPK01, YAT11].

In the cited literature we find many different versions of FDs, with variations coming mostly from different contour representations, such as complex coordinates, centroid distance, etc. In [ZL01] a comparison of the most common representations is made, concluding that transforming the contour into a 1D centroid distance signal has the best performance on general shape recognition. The tests were run on the MPEG7 database described in Section 5.1.1. Hence, we will describe this method in detail.

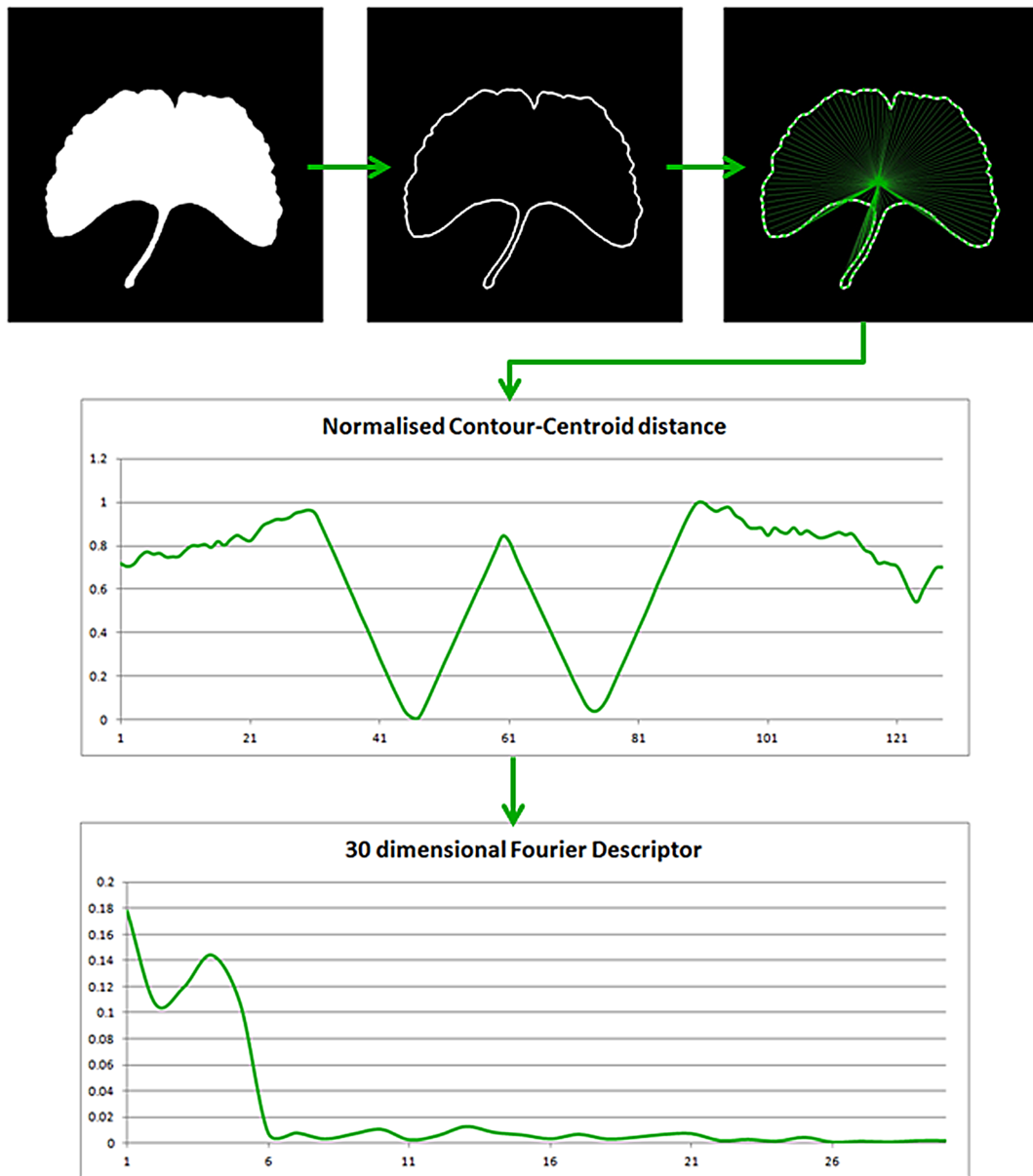


Figure 3.8: Overview of the Fourier Descriptor computation with toy example

Computation of the contour from the segmented image is made by means of morphological operators. The segmented image  $S$  is eroded, producing  $S_e$ . The contour is then obtained from  $S_{contour} = S - S_e$ , the coordinates of all the remaining white pixels representing the final ordered contour set  $C = \{(x_1, y_1), \dots, (x_m, y_m)\}$ . The centroid is defined as the average coordinate of the contour pixels:

$$Centroid(x_c, y_c) = \frac{1}{m} \cdot \sum_{i=1}^m (x_i, y_i)$$

A one dimensional contour signal  $C_s$  is therefore obtained by computing centroid distances for each point of the contour:

$$Cs(i)_{i=1}^m = \sqrt{(x_c - x_i)^2 + (y_c - y_i)^2}$$

The advantage of this representation is that  $C_s$  is already a translation invariant feature of the original shape. Normalising  $C_s$  between 0 and 1 also achieves scale invariance:

$$Csn(i)_{i=1}^m = \frac{Cs(i) - \min_{j=1}^m (Cs(j))}{\max_{j=1}^m (Cs(j)) - \min_{j=1}^m (Cs(j))}$$

The resulting scale and translation invariant vector is then linearly resampled to a fixed number of points -  $n_{contour}$  - in order to achieve feature consistency between shapes and also to provide the Fast Fourier Transform with a power-of-two size signal. Resampling is done at equidistant perimeter points, meaning the distance between two points in the final signal will be equal to the perimeter of the contour divided by  $n_{contour}$ . After resampling we obtain the final spatial contour signal:  $C_{sig} = (t_1, t_2, \dots, t_{n_{contour}})$ .

The Discrete Fourier Transform - DFT - is a frequencial domain transformation and is defined by its coefficients  $c_n$  as follows:

$$c_n = \frac{1}{N} \cdot \sum_{t=0}^{n_{contour}} C_{sig}(t) \cdot \exp\left(\frac{-j2\pi nt}{n_{contour}}\right), n = 0, 1, \dots, n_{contour} - 1$$

Each fourier coefficient  $c_n$  is a complex number containing the magnitude and phase of the  $n^{th}$  frequency in the original signal:

$$Magnitude(c_n) = \sqrt{Re(c_n)^2 + Im(c_n)^2} \quad Phase(c_n) = atan\left(\frac{Im(c_n)}{Re(c_n)}\right)$$

The phase component is dependent on the point on the shape at which we started to sample the contour, the magnitude however is not. Using the Fourier magnitude from each coefficient - also noted  $|c_n|$  - will thus produce a shape descriptor which is translation, scale and rotation invariant. The final feature vector - the Fourier Descriptor - for a given shape is finally computed as:

$$FD = \left( \frac{|c_1|}{|c_0|}, \frac{|c_2|}{|c_0|}, \dots, \frac{|c_{n_{contour}-1}|}{|c_0|} \right)$$

Where all the frequency magnitudes are normalised by the first one, providing a more robust scale invariance than the initial centroid distance normalisation. In practice, the Fast Fourier Transform algorithm is used to compute DFT. FFT requires that the input vector  $C_{sig}$  has a length which can be expressed as a power of two, such as 64, 128, 256, etc. Although the FFT requires specific contour sizes, its main advantage is that its complexity is  $O(n_{contour} \cdot \log(n_{contour}))$ .

#### 3.2.2 Angular Radial Transform - ART

The Angular Radial Transform was first introduced in [KK99] as a new scale and rotation invariant region shape descriptor and shortly afterwards became adopted in the MPEG-7 standard [BPK01]. It has been successfully tested as a descriptor on logo matching in [WN11] and on face recognition in [FQ03], but due to its origins in the signal processing community, it does not benefit from the same amount of exposure in computer vision as Fourier descriptors, for instance. A generalisation of ART is also presented in [RCB04] which introduces linear deformation invariance in a more general descriptor.

The ART is a moment basis transformation, part of the larger family of Zernike moments, whose moment bases are defined on a unit disk in polar coordinates  $(\rho, \theta)$  as following:

$$V_{nm}(\rho, \theta) = A_m(\theta) \cdot R_n(\rho)$$

$$A_m(\theta) = \frac{1}{2\pi} \exp(jm\theta)$$

$$R_n(\rho) = \begin{cases} 1 & n = 0 \\ 2 \cos(n\pi\rho) & n \neq 0 \end{cases}$$

Where  $A_m(\theta)$ ,  $R_n(\rho)$ ,  $m$  and  $n$  indicate the angular and radial moments and their respective degrees.

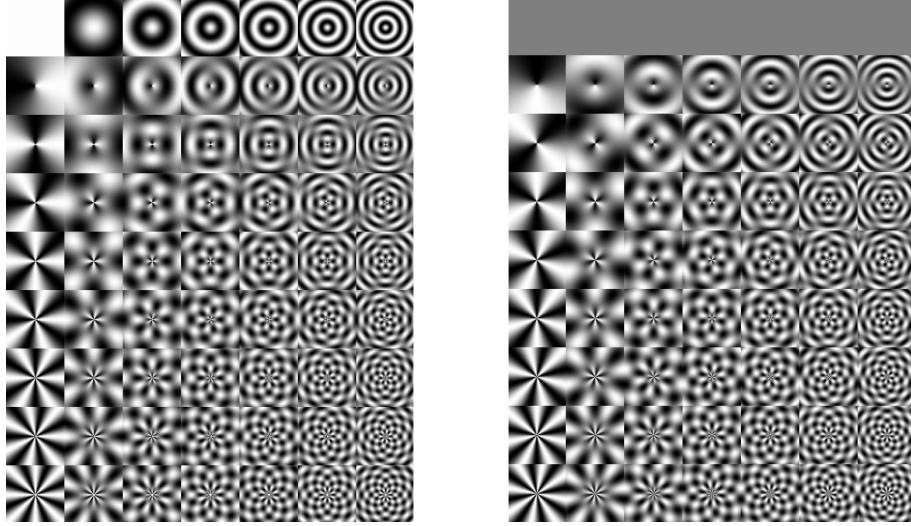


Figure 3.9: Real (left) and Imaginary (right) components of  $V_{nm}$  with  $M = 9$  and  $N = 7$  angular and radial moments respectively

The above figure is a visual representation of  $7 \times 9$  ART moments bases with constant angular moment degrees for each line and constant radial moment degrees for each column. Null values are represented as 50% gray, white values being positive, black values being negative.

The ART transform on an image signal  $I(\rho, \theta)$  is then defined as the inner product of the bases and the image signal on the unit disk. The resulting coefficient for each basis is:

$$C_{nm} = \int_{\rho=0}^1 \int_{\theta=0}^{2\pi} V_{nm}(\rho, \theta) I(\rho, \theta) d\theta d\rho$$

Due to  $V_{nm}$  being complex, the ART coefficients are inherently complex as well. As was the case with Fourier descriptors, the phase of these coefficients is rotational dependent, their magnitude, however, is not. Therefore, we only use the magnitude information from the ART transform for our descriptor. Scale invariance is achieved in a similar fashion with FDs, by dividing all  $|C_{nm}|$  coefficient magnitudes by the magnitude of the first one,  $|C_{00}|$ . Once we establish the number of radial and angular components of the descriptor as  $M$  and  $N$  respectively, the ART feature vector is defined by:

$$ARTD_{MN} = \left( \frac{|C_{01}|}{|C_{00}|}, \frac{|C_{02}|}{|C_{00}|}, \dots, \frac{|C_{N0}|}{|C_{00}|}, \frac{|C_{10}|}{|C_{00}|}, \dots, \frac{|C_{NM}|}{|C_{00}|} \right)$$



The final feature vector is  $M \cdot N - 1$  dimensional and is scale and rotational invariant, however not translation invariant. In order to achieve translation invariance, we resize and recenter the segmented leaf images  $I$  in the center of gravity of the foreground, as depicted in Figure 3.10.

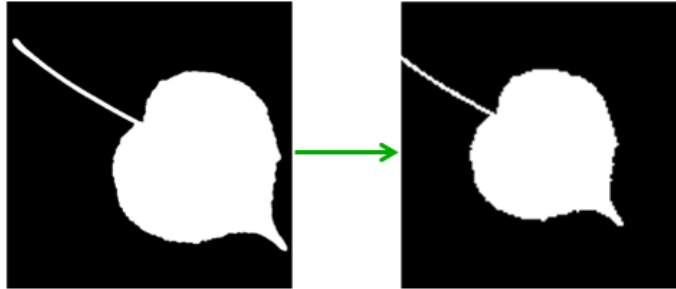


Figure 3.10: Resizing and recentering of a segmented image

Applying this procedure before ART basis decomposition will guarantee that the final descriptor presents the three relevant invariances for our use case: rotation, translation and size invariance.

The inverse ART transform is rarely mentioned in the literature as ART is most often used as a shape descriptor and not an information encoder. We found it interesting however to qualitatively analyse the output of the inverse transform in order to better establish ART parameters and actually see the information maintained by the transform. The inverse image  $\hat{I}(\rho, \theta)$  is defined as:

$$\hat{I}(\rho, \theta) = \sum_{n=0}^N \sum_{m=0}^M C_{nm} \cdot V_{nm}(\rho, \theta)$$

In Figure 3.11 the inverse transform is visualised through 4 reconstructions of a segmented leaf image with varying moment degrees.

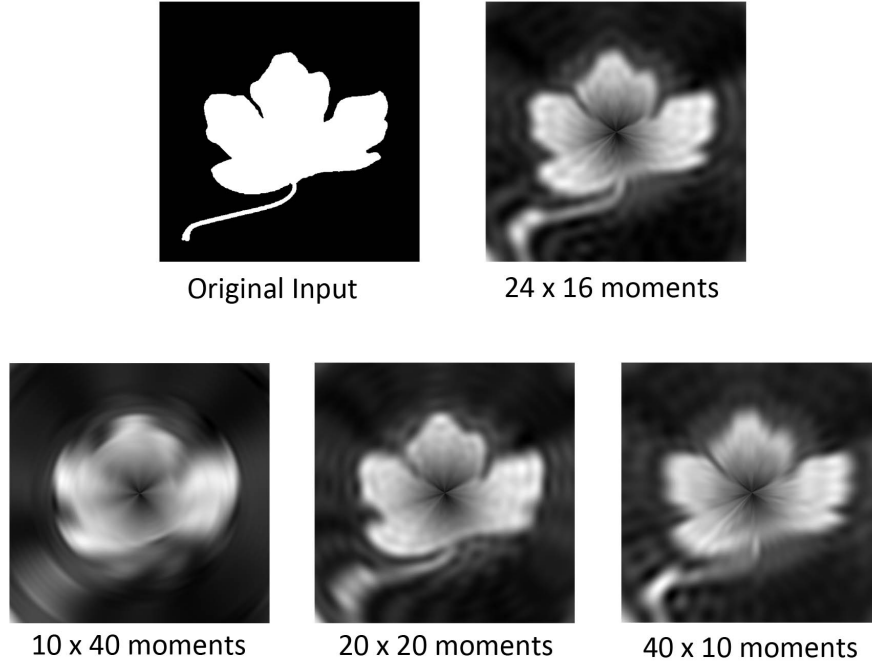


Figure 3.11: Original input image  $I$  compared to different ART inverse reconstruction based on  $24 \times 16$ ,  $10 \times 40$ ,  $20 \times 20$  and  $40 \times 10$  angular and radial moments respectively

### 3.2.3 Complex network maximum degree descriptor

The maximum degree descriptor introduced in [BCB09] and also used by the IFSC/USP group at the ImageCLEF 2011 task [CFB11], is a novel shape descriptor based on modeling the shape contour as a graph and extracting a dynamic evolution signature from this graph.

In a similar fashion to the Fourier descriptor method previously mentioned, the leaf contour is extracted and resampled to a fixed number of 2D contour points:

$$S = \{s_1(x_1, y_1), s_2(x_2, y_2), \dots, s_n(x_n, y_n)\}$$

The obtained resampled contour  $S$  is then considered to be the set of graph nodes for a small world complex network  $G = \langle S, W \rangle$  where  $W$  is the normalised edge set defined as follows:

$$d(s_i, s_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \quad E = \{w_{ij} = d(s_i, s_j) | \forall s_i, s_j \in S\}$$

$$W = \frac{E}{\max_{w_{ij} \in E}}$$

The graph  $G$  thus obtained will be defined by nodes on the shape contour and edges representing the normalised distance between each of these nodes.

The dynamic evolution signature of  $G$  is further defined for a distance threshold  $T_l \in [0..1]$  as:

$$A_{T_l} = \{a_{ij} | \forall w_{ij} \in W; \begin{cases} a_{ij} = 0 & w_{ij} \geq T_l \\ a_{ij} = 1 & w_{ij} < T_l \end{cases}\}$$

The dynamic evolution signature at  $T_l$  will consequently have a value of 1 for all edges smaller than  $T_l$ . An example of signature evolution for three values of  $T_l$  is given below.

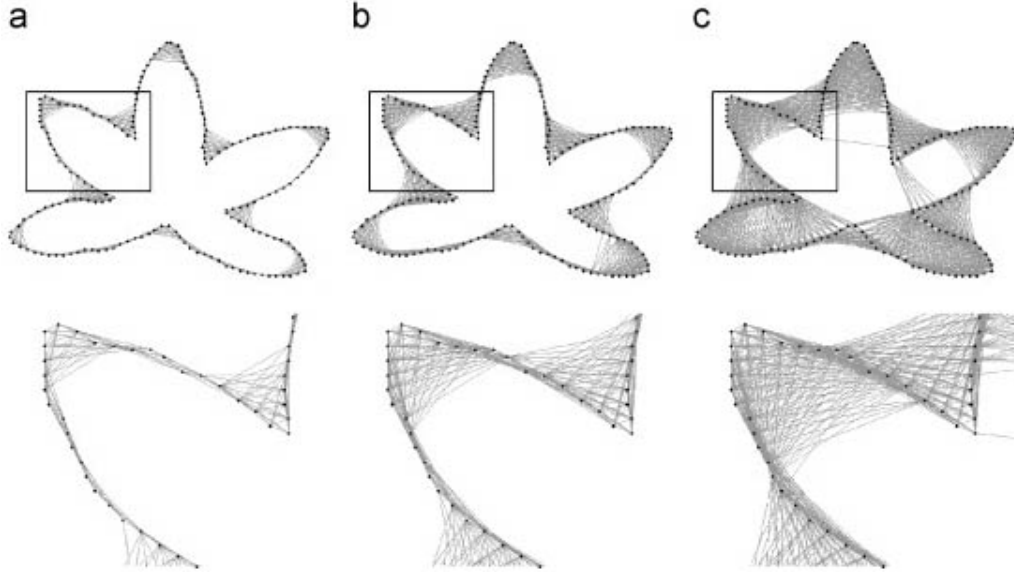


Figure 3.12: Network dynamic evolution for (a) $T_l = 0.1$ , (b) $T_l = 0.15$  and (c) $T_l = 0.2$  as described in [BCB09]

Considering the degree  $k_i$  of an undirected graph node is defined as the number of edges leaving or entering a given node, we can compute two degree values representative for a given dynamic evolution signature: the maximum and average degree.

$$k_i = \sum_{j=1}^n a_{ij}, \quad k_{avg} = \frac{1}{n} \sum_{i=1}^n k_i, \quad k_{max} = \max_i k_i$$

Normalisation for both  $k_{avg}$  and  $k_{max}$  can be made by dividing them by the total number of contour points  $n$ , in our case this is however unnecessary, as the input contour  $S$  always has the same size. We note  $k_{avg} < T_l >$  and  $k_{max} < T_l >$  the average and maximum degrees of an evolution signature  $A_{T_l}$  produced by the threshold  $T_l$ .

The maximum/average degree descriptor is then defined as the concatenation of  $(k_{max} < T_l >, k_{avg} < T_l >)$  pairs for all  $L$  defined  $T_l$ 's resulting in a  $2 \times L$ -dimensional feature vector:

$$DEG = (k_{max} < T_1 >, k_{avg} < T_1 >, k_{max} < T_2 >, k_{avg} < T_2 >, \dots, k_{max} < T_L >, k_{avg} < T_L >)$$

Due to the definition of the maximum and average degrees, their values are inherently invariant to the translation and rotation of a given shape contour. They are, however, dependent on the dynamic evolution threshold values. Nevertheless, using the  $W$  normalised edge set instead of  $E$  assures that all edge weights will be the same between identical shapes, irrespective of size.

### 3.3 Local Feature Extraction

In opposition to general shape features, local features represent highly localised information from small areas of an image, defined around interest points. It is assumed that interest points detected through the same method on similar images will produce similar local features. Although the main focus of this work is efficient and discriminative shape descriptors, the use of basic local features was added for two reasons:

- Texture information: as shape descriptors only use a binary image of the leaf shape, all potentially useful texture and color information such as leaf veins, is ignored. Local features have the potential of representing such characteristics well.
- Natural photograph recognition: taking the first steps toward recognising leaves from natural plant photographs in which the background is unconstrained.

The local feature extraction method we propose is based on the Scale Invariant Feature Transform, grouped into a Bag-of-Words model. The combination of local features and Bag-of-Words models is extensively used in the computer vision community for content-based image retrieval and general object recognition [Low99, CDF<sup>+</sup>04]. The BoW feature extraction process' overview is as follows:

- For training data:
  - Find points of interest in each image - keypoints
  - Compute SIFT descriptors at each keypoint
  - Obtain Bag-of-Words model by clustering all SIFT descriptors from all images, irrespective of class labels, into  $n_{words}$
  - Extract final descriptor for each image as a word histogram of the image's SIFT descriptors
- For testing data:
  - Extract test image keypoints and SIFT descriptors
  - Compute word histogram of these descriptors using the trained BoW model
  - Compare and classify word histograms as feature vectors

#### 3.3.1 Keypoint detection and SIFT descriptor extraction

Most keypoint detection methods such as Laplacian of Gaussians, Hessian and Harris are based on the gradient of the image, or similar light variation representations. They are all based on the basic assumption that points of interest are found at local maximal gradient locations, points which humans usually perceive as detail.

We derive our current methods from qualitative analysis of leaf images, scan-like as well as natural photographs from the field, and visual comparison of multiple keypoint detectors (dense sampling, DoG[Low99], SURF[BTG06], MSER[MCUP02], Harris[FG87, HS88]): Harris corner detection on opponent color channel and Difference-of-Gaussians - DoG - on composite or intensity channel. Qualitative analysis of keypoint distributions show that DoG is too sensitive to noise to be used on color based channels, such as saturation or color opponent channels. DoG seems to better converge on leaf features when coupled with intensity based channels which do not suffer as much from JPEG compression as does color information. Harris corner detection, on the other hand, requires strongly contrasted edges in order to find keypoints. It was thus coupled with color opponent channels.

The first keypoint detection method applies the Harris corner detection algorithm described in [HS88] on a channel containing exclusively color information -  $O_1$ . This opponent channel is inspired by the  $b$  channel from the  $L^*a^*b^*$  color space and is computed as:

$$O_1 = \frac{R - 2G + B}{4} + 0.5 \quad O_2 = \frac{R - 2B + G}{4} + 0.5 \quad I = \frac{R + G + B}{3}$$

Where  $R$ ,  $G$  and  $B$  are the respective RGB channels with values between  $[0..1]$ . The resulting output consists of interest points mainly placed on corners or edges of the leaf's image. The reason we prefer using  $O_1$  rather than  $I$  is that it offers a degree of lightness invariance and it is mostly unaffected by noisy backgrounds such as dirt, pavement and other color uniform backgrounds as shown in Figure 3.13.

The second keypoint detection method employed is the Difference-of-Gaussians, which is also the default SIFT keypoint detection process described in [Low03]. We use a composite channel  $C_s$  for this method as color opponent channels can be noisy from compression artifacts and DoG is more susceptible to noise than Harris corner detectors.

$$C_s = \frac{R + G}{2} \sqrt{S} \quad S = \frac{\max(R, G, B) - \min(R, G, B)}{\max(R, G, B)}$$

Where  $S$  represents the saturation channel of the image. The reasoning behind the construction of this composite channel is based on leaf color characteristics: leaves are saturated elements in an image, ranging in color from red to yellow to green. The composite channel thus offers better contrast for such regions than the intensity channel.

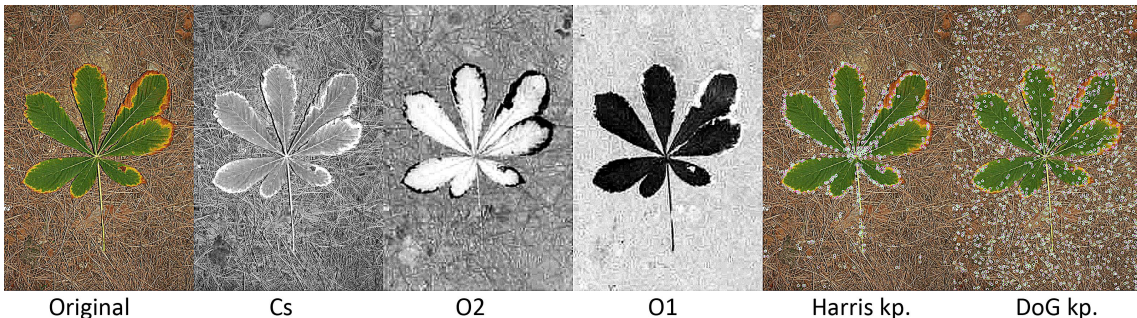


Figure 3.13: Example of a leaf photograph taken against noisy background. The discussed channels are shown and keypoint detection results are highlighted in light green.

Once the keypoints are extracted, we compute the SIFT described in [Low99] centered on each keypoint. SIFT is based on image gradient value and orientation in a  $16 \times 16$  pixel grid around the keypoint.

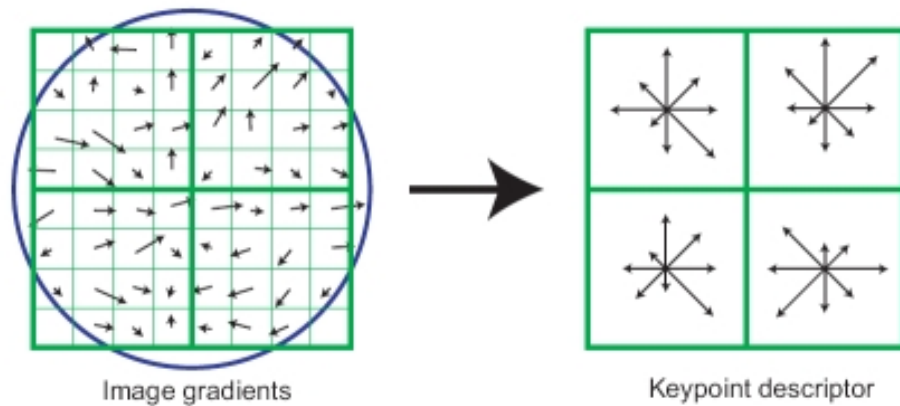


Figure 3.14: SIFT descriptor as presented in [Low03]. Dimensions of the window and histograms are reduced from the actual implementation for readability.

The gradient intensities are weighted by a Gaussian windows as indicated by the overlaid circle and are afterwards accumulated in  $4 \times 4$  8-bin orientation histograms, depending on their orientation. The resulting 128 dimensional feature vector is a robust representation of gradient variation around a keypoint.

The channel used for keypoint detection and the one used for SIFT computation are not necessarily the same. We also explore the possibility of extracting the SIFT features from the intensity channel, while using the opponent and composite channels for keypoint detection only.

### 3.3.2 Bag-of-Words model

The Bag-of-Words model originated in natural language processing and information retrieval fields [Har54]. It is used to reduce the dimensionality of document classification by defining sets, or bags, of words and measuring frequency of appearance of words from such bags in documents. The resulting histograms are used to compare document similarity instead of comparing each individual word.

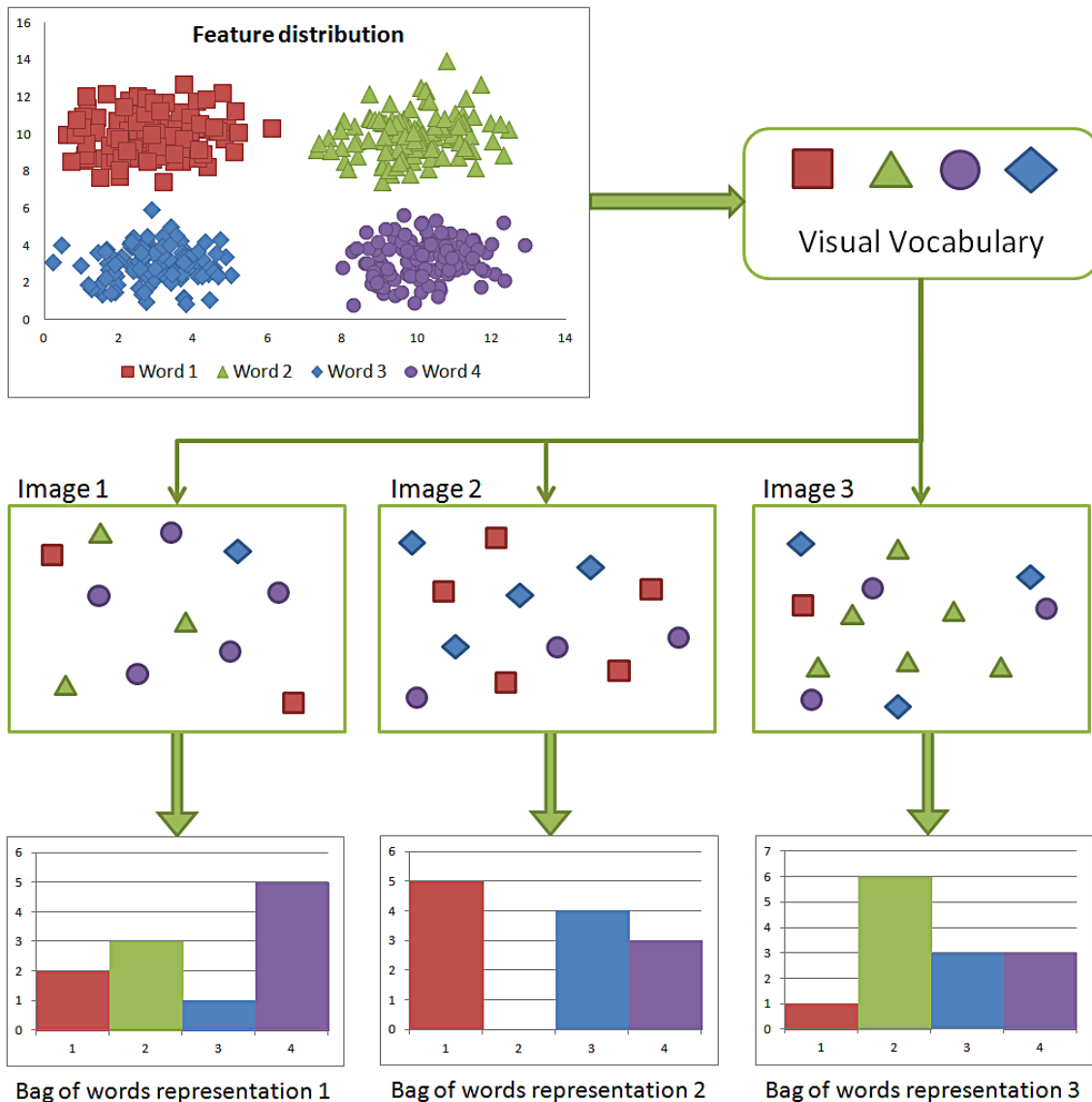


Figure 3.15: Toy example overviewing the bag of visual words approach for image content retrieval

In a similar fashion, it is used in computer vision to avoid the great computational expense of matching local features. The Bag-of-Words model employed in this work is defined by a dictionary, or word vector, in which each word represents a cluster of SIFT features

defined by unsupervised learning over all the features in the training set, irrespective of class or image membership. More specifically, we compute the dictionary by k-means clustering, initialising the algorithm with  $n_{words}$  clusters. The resulting converged cluster centers represent the average SIFT descriptors, or visual words. The BoW descriptor is then computed as a histogram, where each bin counts the number of SIFT descriptors closest to the respective visual word from the dictionary, as shown in Figure 3.15.

### 3.4 Classification

General statistical classification is the process of identifying a set of categories, or classes, to which a new observation belongs, on the basis of prior knowledge such as a training dataset. More specifically, classification in this work will be the process used to assign a certain plant species to an image, based on its feature set. It is also a subset of the more general classification problem in statistics and machine learning, namely supervised learning. We formalise the classification elements as follows

$$C = \{c_i | i = 1..n_{classes}\} \quad f = (a_0, a_1, \dots, a_m), \quad f \in F \quad T = \{(f_{ti}, l_i) | i = 1..n_{train}\}$$

Where  $C$  represents the class set,  $f$  a feature vector in the corresponding  $m$ -dimensional feature space  $F$ ,  $a_i$  is a feature attribute and  $T$  the training set of feature vectors and their respective class labels. Hence, we are looking for a function  $Class(f) : F \rightarrow C$  that assigns a class label from  $C$  to a given feature vector  $f$  based on the data from  $T$ . Throughout this work we will be looking for a classification method not only capable of attributing a class label to a feature vector  $f_s$  but also a confidence vector describing the probability that a given sample belongs to a certain class:

$$Conf(f_s) = \{P_i | i = 1..n_{classes}, P_i = P(c_i | f_s)\}$$

#### 3.4.1 Closest cluster center classification

As an introductory specific example of classification we take a look at classifying an unknown sample  $f_s$  by attributing it to the closest class cluster center from feature space. For each class we compute the average feature vector from  $T$ , representing the respective class cluster center  $f_{ci}$ :

$$CC = \{f_{ci} | i = 1..n_{classes}\} \quad \text{where} \quad f_{ci} = \text{avg}\{f_{tj} | l_j = c_i, \forall j = 1..n_{train}\}$$

$$Class(f_s) = c_j | j = \underset{i}{\min}(dist(f_s, f_{ci}))$$

The confidence vector can be computed by normalising  $dist(f_s, f_{ci})^{-1}$  with the sum over all the classes:

$$Conf(f_s) = \{P_i | i = 1..n_{classes}, P_i = dist(f_s, f_{ci})^{-1} / \sum_{i=1}^{n_{classes}} dist(f_s, f_{ci})^{-1}\}$$

The normalised distances can then be viewed as probabilities of class membership through the reasoning that the closer a given class cluster center is to  $f_s$ , the higher the probability of it being the correct class for  $f_s$ . The distance measure function in practice can be any meaningful distance function between  $m$ -dimensional vectors, such as L1, L2 or Mahalanobis.



#### 3.4.2 K-Nearest Neighbours classification

Special interest was given to KNN due to the following reasons:

- **Simplicity:** it is one of the simplest classifiers with characteristics fitting our requirements.
- **Efficiency:** it requires no training computations and is easily handled by weak processors. Its testing time, however, grows linearly with the size of the training set, limiting the scalability of the classifier.
- **Nearest Neighbour methods** have been popular and shown promising results at the ImageCLEF 2011 plant recognition task [GBJ<sup>+</sup>11, CTM<sup>+</sup>11, GJY<sup>+</sup>11]

The fact that KNN requires no training may also be particularly useful for in-the-field plant recognition work as the database of known species can easily be updated on-the-fly so that correctly identified new samples are further used for classification.

Analogously to closest cluster center classification, KNN is also based on distance measures in feature space but instead of comparing  $f_s$  to a class representative value, it compares it to all samples of the training set  $f_i$ , selecting the first  $k$  closest ones. We call the subset of  $k$ -closest training samples  $K$ .

In the classical KNN approach, a voting scheme is applied through which  $f_s$  is attributed the class label  $l$  most often observed in  $K$ . The class score will therefore be a histogram of label values. In our approach, each class vote is weighted by the inverse of the distance between the respective feature and  $f_s$ . The reasoning behind this weighting scheme comes from observing images from plant species: a label may refer to very different shapes, defining the leaf at different stages of its growth process for instance. This will be reflected in feature space as high intra-class variance of the features and overlap of classes. Hence, weighting the vote with the distance further localises the classification process.

We describe the classification process as well as the computation of the confidence vector  $Conf$  through the following algorithm. A notable specialisation of the KNN algorithm is 1NN or Nearest Neighbour classification in which  $f_s$  is simply given the class label of the closest feature vector from  $T$ .

---

**Algorithm 2** Weighted KNN with ranked classification

---

```
1:  $n_{classified} \leftarrow 0$ 
2:  $confSum \leftarrow 0$ 
3: Sort  $T$  ascending by  $dist(f_s, f_{ti})$ 
4: while  $n_{classified} < n_{classes}$  do
5:   Zero( $ClassScores$ )
6:   for  $i = 1$  to  $K$  do
7:      $ClassScores(l_i) \leftarrow ClassScores(l_i) + 1/dist(f_s, f_{ti})$ 
8:   end for
9:    $(bestClassLabel, maxClassScore) \leftarrow max(ClassScore)$ 
10:   $Conf(bestClassLabel) \leftarrow maxClassScore$ 
11:   $confSum \leftarrow confSum + maxClassScore$ 
12:  for all  $(f_t, l) \in T$  do
13:    if  $l = bestClassLabel$  then
14:       $T \leftarrow T - (f_t, l)$ 
15:    end if
16:  end for
17:   $n_{classified} \leftarrow n_{classified} + 1$ 
18: end while
19: for all  $c \in Conf$  do
20:   $c \leftarrow c/confSum$ 
21: end for
```

---

### 3.4.3 Random Forests

Random Forests - RF - are an ensemble classifier based on decision trees. They were first introduced in [Bre01] and further analysed in [A. 11]. The method we describe and use is based on bagging (Bootstrap aggregating) introduced in [Bre96] and random feature attribute selection for decision tree training introduced in [Ho95].

The basic predictors used in the random forest classifier are decision trees. Decision tree models - DT - predate machine learning techniques and have been used as a support tool for decision making and analysis in fields such as finance and economics. In our specific case, the DT model is used to classify features by applying threshold functions on their attributes at each node as shown in Figure 3.16, with leaf nodes representing class labels.

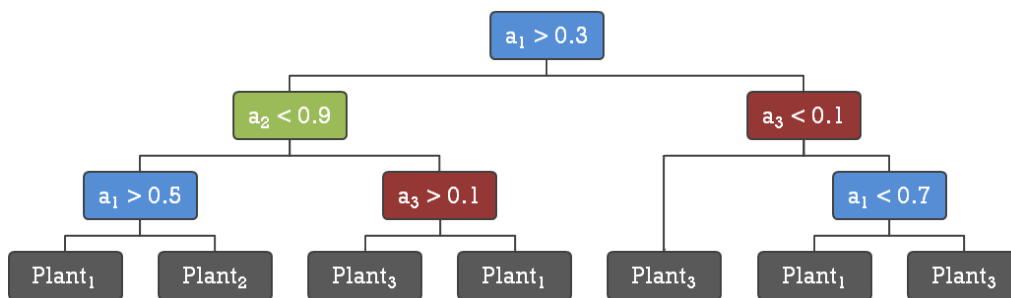


Figure 3.16: Toy example of a decision tree for a 3-attribute feature vector  $f = (a_1, a_2, a_3)$  and three plant classes  $C = \{Plant_1, Plant_2, Plant_3\}$

The function  $Class(f_s)$  is then defined by the evaluation of the DT with the attribute instances of  $f_s$ , starting from the root, until a leaf node is reached.

However, in order to obtain such a classifier the DT must be firstly trained in order to find the attributes and their threshold values for each node, a procedure called *Decision Tree Learning*. Notable examples of algorithms for DT learning are the ID3, C4.5 information gain based methods introduced in [Qui86] and the Gini impurity based CART algorithm introduced in [Bre84]. In both cases, the node attribute and its threshold are selected by a greedy method searching for the best split of the training data with respect to class labels. For each partition the best sub-split is computed and so forth, hence obtaining the complete decision tree. Most notable problems with stand-alone decision trees are the tendency to overfit the training data, if growth is not limited, and the greedy method employed which does not necessarily find the optimal values. Another issue which makes DT's unsuitable for our use-case is that they only produce one class label per evaluated feature vector, with no way of finding which other classes might represent the feature vector. In other words, we are unable to compute the *Conf* set.

The random forests we use in this work contain an ensemble of small DT's that have been trained on random sub-samples of the data. The basic principle is that of predictor bagging: *an ensemble of many classifiers, aggregated to produce a strong one, may function better than using one highly complex classifier*. In our case, instead of training a single DT on the complete training set and attribute set, a large number of small DT's are

trained on random subsets of both samples from the training set and attributes from the feature space. By varying the size of these two subsets we can effectively influence the ratio between randomisation and correlation in the forest: large subsets will mean that the DT's are trained on more correlated information, resulting in smaller variance or randomisation between them.

The particular method described in [Bre01] also presents the idea of out-of-bag estimates used to avoid overfitting, compute attribute importance and estimate internal error. The out-of-bag principle is as follows: because each DT from the RF has been trained on a subset of samples, we can determine, for each training sample  $(f_{tk}, l_k)$ , a subset of classifiers from the RF which have not been trained on the respective sample, called the *out-of-bag classifiers*. By evaluating  $f_{tk}$  with the out-of-bag classifiers we can obtain cross-validation errors for instance. Attribute importance can also be computed by permuting an attribute  $a_m$  in all training samples from  $T$  and computing the out-of-bag error at each permutation. If  $a_m$  is important, permuting it between samples will result in an increase of the error. If not, it means the attribute has no discriminative value for the RF classifier and the error is unaffected.

The classification produced by RF has the form of a voting scheme in which each DT's output label is counted in a class histogram which represents the forest's confidence. The histogram can be further normalised by the number of trees in the forest, producing values which match our definition of *Conf*.

The RF classifier thus solves the problems we noted with stand-alone DT's:

- it inherently reduces overfitting
- by varying the randomness to correlation ratio we can escape the local maxima tendencies of the greedy method and obtain better classification
- a class confidence vector can be computed allowing for ranked classification

### 3.4.4 Attribute selection

Attribute selection is a feature dimensionality reduction method we propose for the following reasons:

- Efficiency: reducing the number of feature dimensions - attributes - improves required computational power
- Noise reduction: not all features extracted are relevant and some often add noise and lower the classification performance

Motivated by the above, we propose to extract specific attributes at the training and classification steps, typically the first  $N_d$  in descending order of their importance. We propose two methods for measuring the importance of an attribute on  $T$ :

- Random Forest attribute importance, described in 3.4.3
- Attribute discriminance, described below

The attribute discriminance function  $D(a_i)$  is computed as the inter-class variance divided by the intra-class variance of an attribute:

$$\begin{aligned} \mu_c &= \text{avg}(f_{tj} | l_j = c, \forall j); & \mu &= \text{avg}(\mu_c | \forall c); & \sigma_{intra}^2(a_i, c) &= \frac{1}{n_c - 1} \sum_{j=1}^{n_c} (f_{tj}[i] - \mu_c)^2 \\ \sigma_{intra}^2(a_i) &= \frac{1}{n_{classes}} \sum_{j=1}^{n_{classes}} \sigma_{intra}^2(a_i, j); & \sigma_{inter}^2(a_i) &= \frac{1}{n_{classes} - 1} \sum_{j=1}^{n_{classes}} (\mu_c - \mu)^2 \\ D(a_i) &= \frac{\sigma_{inter}^2(a_i)}{\sigma_{intra}^2(a_i)} \end{aligned}$$

As a result, the  $D(a_i)$  indicator will be higher if an attribute varies much between classes and little inside them, thus estimating its discriminative power.

### 3.4.5 Early / Late feature fusion

When using multiple descriptors or feature extraction methods, joining heterogeneous features can be problematic as the values of each feature attribute do not have the same significance or scale value. The problem of classifying with multiple descriptors can be approached from two main directions:

- Early fusion: define a new feature vector composed of all the descriptors and classify it as one entity

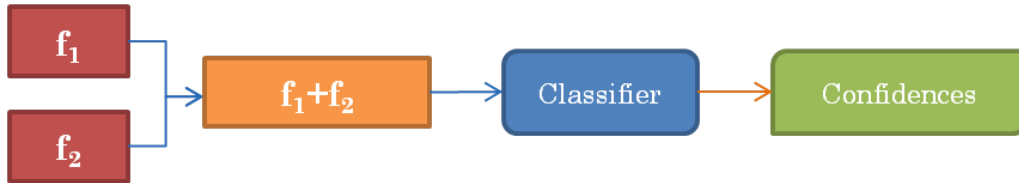


Figure 3.17: Early fusion diagram

- Late fusion: classify each descriptor separately and define a new *Conf* vector based on each of the confidences from each separate classification.

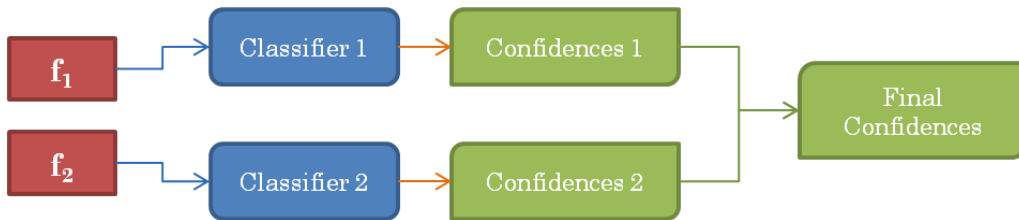


Figure 3.18: Late fusion diagram

In the first case, a simple fusion technique would be to simply concatenate weighted feature vectors into a larger one. In the second case, the *Conf* vector can be defined as a weighted average of confidence vectors from each classification. The main advantage of early fusion is that correlation between attributes from different descriptors can still be taken into account by the classifier. In late fusion, however, such information is lost. The advantages of late fusion are efficiency and parametrisation. Firstly late fusion classification time grows linearly with the number of descriptors, whereas early fusion increases the dimensionality of the feature space. Secondly, we can learn late fusion ratios or manually set them to reflect separate importance for each descriptor.

## 4. Methodology

In this chapter we will present the application and implementation of the aforementioned theoretical principles. Firstly, the system design will be explained, followed by details about the implementation, highlighting original work and external libraries. Secondly optimisation of the system is discussed due of its importance in the transition of the system to real-world portable applications.

## 4.1 System design

The basic system architecture can be split into a layer layout, in which the output from each layer is the input for the next:

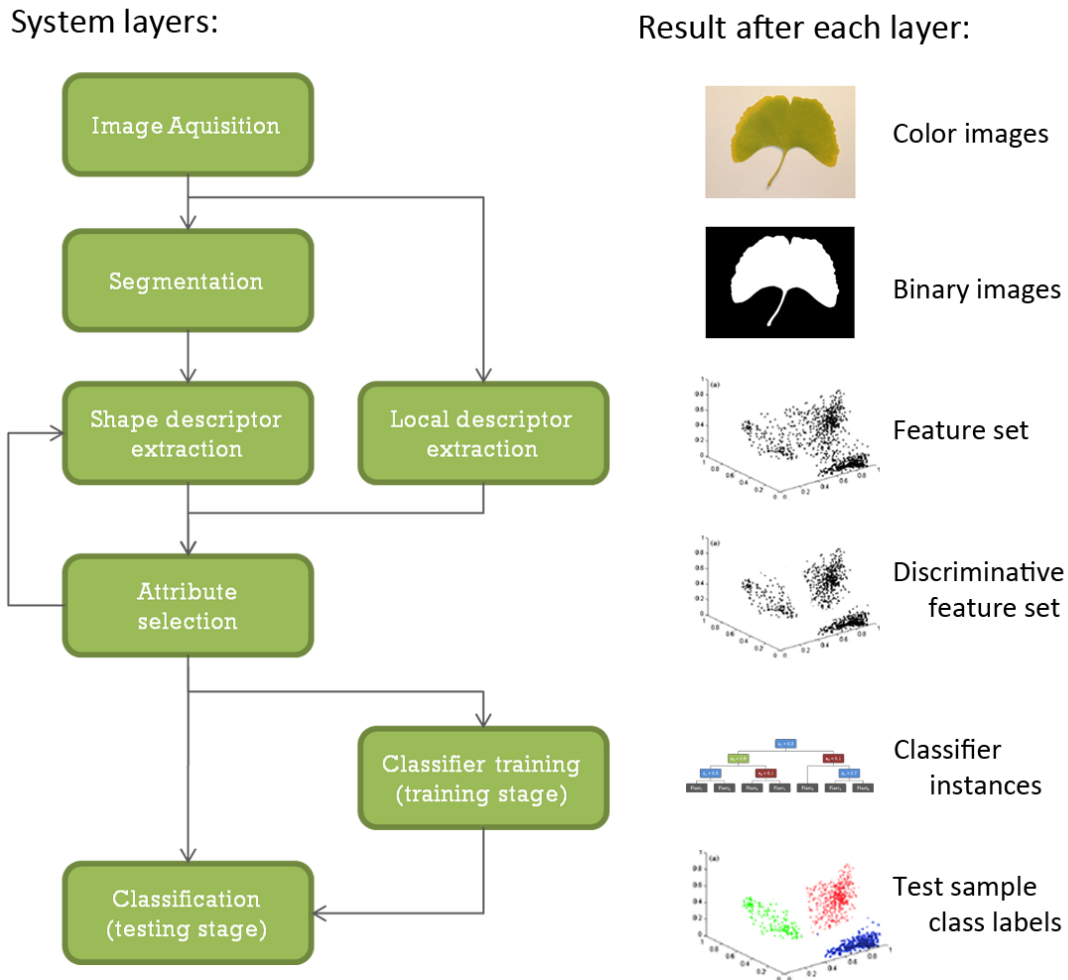


Figure 4.1: Overview of the automated plant recognition system proposed in this work

While discussing implementation details, we note where external libraries from OpenCV have been used. If no mention is given, the respective elements have been implemented from scratch. As much as possible, independent implementation was preferred in this work in the hope of making it easier to port to other operating systems. OpenCV has been chosen as source for external libraries and basic image manipulation (reading, writing, display, etc) due to its open source code and availability on numerous operating systems (Linux, Windows, Android and others). Also, to improve the prototyping process, each layer can



save its output to disk and read its input from disk so the system can resume from any point in the computation process.

In Figure 4.2 we offer an overview of all the possible configurations of our system, based on the layered architecture, as well as the methods introduced in Section 3.



Figure 4.2: Overview of the possible configurations of the system. Numbered cases represent steps, unnumbered cases represent choices. The grayed links represent implemented and tested elements which are however not part of the best and final system configurations.

Although more configurations are discussed and tested in this work, we note two of the most important:

- Shape-based recognition only: Images are segmented with own method, based on k-means clustering, presented in Section 3.1. The only descriptors used are ART and FD, presented in Section 3.2, applied to segmented images and fused late or early. Attribute selection is discriminance based and the classifier used is Random Forests.

- Shape and local features based recognition: As before, images are segmented with k-means clustering. Both shape descriptors are computed, as well as the SIFT+BoW features introduced in Section 3.3. For keypoint selection we retain the slower but better performing DoG, applied to the intensity channel. Attribute selection remains discriminance based, while descriptors are classified individually through Random Forests. Late fusion is applied to the resulting confidence vectors to obtain the final result.

We further discuss the implementation of each separate layer, noting configurations and default parameter values, where applicable.

#### 4.1.1 Segmentation

The segmentation layer receives a color image as input and provides a binary image as output at the same resolution as the input image. It implements the following segmentation methods:

- Derivative based, described in Section 3.1.1, with parameters: smoothing window size, derivative epsilon, starting black point, maximum threshold
- K-means clustering, described in Section 3.1.2, with parameters: starting black point, starting white point, bias
- Otsu segmentation (OpenCV) [Ots79], only used with default parameters

While the Otsu method can be applied directly on a color image, we firstly need to extract the mixed Blue-Saturation channel and perform clipping on it, as described in Section 3.1, before running derivative or k-means based thresholding. Before training on a new dataset, the output of the three methods is qualitatively analysed. Parameters are adjusted by visually inspecting the errors and their characteristics such as falsely including background in foreground, non contiguous areas, etc. The k-means segmentation method was finally preferred on all datasets as it seemed to be the most flexible and robust. The  $a$  and  $b$  parameters of the composite channel  $a \cdot (1 - Blue(x, y)) + b \cdot Sat(x, y)$  used for thresholding were chosen to allow for correct segmentation of complex leaves without adding too much shadow noise. The default values of  $a = 0.8$  and  $b = 0.2$  worked well on all datasets. Default starting black point and white point are set at 5 and 100 respectively, for a 8-bit composite channel image. After visual inspection of the segmented output and counting of the falsely segmented images, we found that the method has 99% success rate on all the leaf shape datasets in which the leaf is photographed against an almost white, uniform background. We obtain this figure by looking at individual images and counting the cases in which segmentation failed to produce a good shape of the image (falsely segmented backgrounds, shadows, etc). However, in many cases, small shadows are present around the leaves, which, if falsely segmented, do not significantly alter its shape. We consider these cases as successful segmentation as well.

#### 4.1.2 Shape descriptor extraction

Shape descriptors receive a binary image in input and produce a descriptive feature vector of the shape it contains.

We have implemented the three shape descriptors presented in Section 3.2 :

- ART descriptor, with parameters:
  - resolution - the working resolution at which the descriptor is computed
  - N - the number of radial bases
  - M - the number of angular bases
  - ART base indexes - vector of base pairs (n,m) indicating which bases to compute, all  $N \times M$  bases are computed if vector is empty
  - number of features - the number of ART coefficients to be computed and returned, must correspond to size of the ART base index vector
- FD, with parameters:
  - contour size - the number of points at which the contour will be resampled (both up- or down-sampling)
  - number of features - the number of FD coefficients to be computed and returned
- Maximum/Average Degree Descriptor - DEG, with parameters:
  - number of network nodes - the number of points at which the contour will be resampled (both up- or down-sampling)
  - number of thresholds - indicates how many shape signatures will be computed
  - (min threshold, max threshold) - defines the thresholding interval

The ART descriptor has been tested at various resolutions, radial and angular degrees, both quantitatively, by error analysis, as well as qualitatively, through observations of the inverse ART transform. Qualitative analysis of bases importance is also done by applying the attribute selection methods from Section 3.4.4. We have found that the angular moments are much more important for leaf description than the radial ones and that resolution values over  $100 \times 100$ px do not show significant improvement on any of the datasets used in this work. We discover that as a rule of thumb, a ratio varying from  $1/2$  to  $1/4$  between the radial and angular degrees produces coefficients with good discriminative properties. These conclusions are also confirmed by [FQ03] where a set of  $3 \times 12$  frequencies is used for face recognition, as well as our own attribute importance analysis in Section 5.4. However, our system reduces the optimal parameters to choose to only one: the number of ART coefficients to be extracted, through the implementation of the attribute selection method described in Section 4.1.4.

The implementation of Fourier descriptors is based on the Fast Fourier Transform algorithm which reduces the computational complexity of the DFT from  $O(n^2)$  to  $O(n \cdot \log(n))$ , where  $n$  is the number of points on the contour. The only requirement is that  $n$  has to be a power of two. Only the largest contour detected in the image is used and it is resampled to a stable 4096 points. The resulting 1D centroid distance signal is given to the FFT, obtaining 2048 useful contour frequency coefficients. From these 2048 coefficients, the first "number of features" coefficients are returned as dictated by the parameters. Attribute selection is not implemented as it brings no computational benefits from a feature extraction point of view because FFT needs to compute all frequency coefficients to work properly.

The descriptor is however fast enough to not need any type of further speed optimisation as concluded in Section 5.4.

The DEG descriptor has been implemented only for comparison purposes as it was successfully used in combination with statistical classifiers in [CFB11]. The potential of DEG coupled with KNN and Random Forests has been investigated but due to low performance the default parameters were not further optimised. Tests with a number of thresholds anywhere between 10 and 100 from the interval  $[0 \cdots 0.5]$  showed no particular improvement in performance when used with the aforementioned classifiers.

For a given binary image, the system can provide each shape descriptor separately or concatenated. ART and FD respectively are scaled before concatenation, either by multiplying each descriptor with a given coefficient or by choosing a ratio as to have the same maximum variance in both ART and FD. This process effectively performs early fusion as described in Section 3.4.5.

### 4.1.3 Local feature extraction

The SIFT + BoW model described in Section 3.3 is used in this work by taking full advantage of the implemented OpenCV methods. It receives a color image as input, derives a single-channel image and uses it for extraction of the visual word representation feature vector. Most of the necessary steps for local feature extraction are offered by pre-implemented functions:

- Keypoint extraction - both Harris and SIFT-DoG methods are implemented, however, with no possible parametrisation
- Descriptor extraction - SIFT descriptor extraction with no parametrisation
- BoW training - is achieved by the K-means++ algorithm, taking the vocabulary size as the K parameter for clustering
- BoW feature extraction - computes the word histogram without any parametrisation

Through "no parametrisation" in the above, we mean the respective methods offered by OpenCV have no parameters to tune their behaviour. Because the keypoint extractors have no parameter for the maximum amount of keypoints detected in an image, we implemented a random sub-sampling procedure which randomly selects a number of keypoints from the keypoint vector if its size is greater than the maximum number permitted. The only variables which affect the performance of the algorithm are thus

- Single-channel input image definition - selection of intensity channel, color opponent channel, saturation channel, etc
- Maximum number of keypoints
- BoW vocabulary size

For natural photographs, the definition of the input image is done by analyzing different channels and color space transformations of the plant images and the keypoint positions. We are looking for a channel which presents rich gradient information on the leaf surfaces

without introducing noisy keypoints. Through this analysis we defined two separate methods described in Section 3.3.1 which are compared in Section 5.3.2. For scan-like images however, the default light intensity channel is used.

The maximum number of keypoints per image used for BoW training has been limited to 2000 due to memory limitations and the fact that in most cases, when an image produced more than 1500 keypoints, they were on noisy backgrounds such as tarmac or dirt. In comparison, scan-like images produce an average of 300 DoG or 100 Harris keypoints.

The default BoW vocabulary size has been set at 1000 words to insure that enough words exist to separate noise generated keypoints from relevant ones. The vocabulary size usually varies between 1000 and 5000 in the content-based image retrieval literature by [Low99, CDF<sup>+</sup>04] and our tests indicate that vocabulary sizes of more than 500 words do not alter the method's performance on leaf classification. Visualising the attribute importance measures shows that the actual number of highly discriminative words is in reality much lower than 1000.

#### 4.1.4 Attribute selection

Attribute selection accepts as input a feature set and a number  $N_d$  of attributes to be selected. The attribute selection layer then plays a double role, being used to find the importance of attributes on training data and extract the important attributes from both training and testing data before classification or training. These roles can be formalised in two methods:

1. Generating indexes of the most discriminative features, ordered by their importance as defined in Section 3.4.4
2. Extracting a new feature set from a given one, considering a vector of attribute indexes and  $N_d$

Attribute selection is performed by using the output of the first method as the index vector for the second. However, separating the two is necessary for feature extraction optimisation: The ART descriptor is especially computational intensive, requiring  $N \times M \times (\text{Image resolution})^2$  computations to obtain a  $N \times M$  feature vector, as it needs to multiply each pixel of the image with each corresponding pixel from all the bases. This requirement can be greatly improved by means of attribute selection through the following procedure:

- Firstly, we compute ART at  $700 \times 700$ px, with  $15 \times 30$  bases, on the entire training set. We chose these values to be at least the double of other ART parameters found in the literature, giving us a very high detail ART feature vector.
- Secondly, we use our attribute selection method to generate the indexes of the most discriminative ART bases, ordered by their importance.
- Finally, when testing, ART will only compute the most important coefficients at feature extraction, requiring no further selection before classification.

The above procedure can be used either to reduce the feature dimension by removing unuseful attributes, or to improve the feature vector quality for a given feature size. It trains, in essence, the feature extractor to produce only the more discriminative features.

We chose a maximum of  $15 \times 30$  bases, to insure that all of the potentially important bases are computed.

In the case of FD, DEG and BoW, the output indexes from the feature selection are not used in descriptor computation as it provides no performance benefits. However, attribute selection can still be performed before the classification layer to produce lower dimensional, more discriminative feature vectors, improving the classification or the training time.

#### 4.1.5 Classification

The classification layer receives a feature set as input and, similar to the attribute selection, has two functions depending on the type of data used - training or testing:

- Training: Based on  $T$ , produce classifier models  $M_c$
- Testing: Based on  $M_c$ , attribute a class label to testing sample  $f_s$

In this layer, the only use of OpenCV was for Random Forests as they provide an accessible implementation of the algorithms described in [Bre01].

Depending on the classifier type used, different parameters can be set:

- Closest Cluster: no parametrisation
- KNN: the number of closest neighbours used for score calculations,  $K$
- Random Forests<sup>1</sup>:
  - Compute variable importance - tells the RF to evaluate the attribute importance vector described in Sections 3.4.3 and 3.4.4 during training.
  - Maximum tree depth - sets the individual DT depth at which the growth is stopped
  - Number of active variables - establishes how many attributes are randomly selected for the training of each DT
  - Minimum sample count - represents the minimum number of samples requesting a split in the tree at a certain node in order to actually perform the split
  - Number of DTs in the forest

The  $K$  parameter will be chosen by cross-validating training sets and choosing the value which produces best performance. Interestingly enough, all tests indicate that for any  $K > 1$ , the general performance of the plant recognition task decreases so the default value for  $K$  is 1.

Default random forest parameters are those proposed in [Bre01]: minimum sample count of 1-2, a maximum tree depth of 20-30, and the number of active variables equal to the square root of the total attribute number. The number of DTs in the forest is set high in comparison to the suggested value of 100. It is set to 1000 if no memory constraints are existent, but it can be set lower if required by the machine. The reasoning behind such a

<sup>1</sup>Detailed description for all OpenCV RF parameters can be found on <http://www.iib-chemnitz.de/cvwrapper/onlinehelp/html/55ec6b4b-21a7-4eb1-1d6d-76316fe31974.htm>

high number of trees is that the classification result  $Conf(f_s)$  will be obtained through a voting scheme and requires a large vote count to distinguish between each class.

In this work, the Random Forest implementation from OpenCV has been slightly modified to allow for exportation of the tree voting vector. By default, the RF implementation - *CvRTrees* - only allows rank 1 classification. A variant of the *CvRTrees.predict* function was made to output the voting results normalised by the number of trees in the forest, called *CvRTrees.predictAllProbs*.

We present, in the following table a short comparative of these classifiers by complexity:

Classifier	Training complexity	Testing complexity for a sample
Closest Cluster	$O(n)$	$O(m)$
KNN	-	$O(n \cdot m)$
Random Forests	$\hat{O}(k \cdot n \cdot m)$	$O(k \cdot d_{max})$

Table 4.1: Computational complexity overview of classifiers, where  $n$  is the size of the training set,  $m$  is the number of classes,  $k$  is the number of DTs and  $d_{max}$  is the maximum depth of DTs in the RF

Notable remarks from the computational complexity analysis of these classifiers:

- Closest Cluster classification has constant testing complexity irrespective of the number of training samples
- KNN requires no training but testing complexity increases linearly with the number of training samples
- RF testing complexity is also independent of training data size and furthermore remains constant relative to the number of attributes of the features set. This is translated in excellent scaling on high dimensional feature vectors. The exact training complexity of Random Forests is difficult to determine mathematically because of the randomness introduced by the different parameters. However, measurements of execution times show that for a given parameter set, training time grows linearly with the number of samples in the training set and attributes, establishing an estimated complexity of  $\hat{O}(k \cdot n \cdot m)$ :

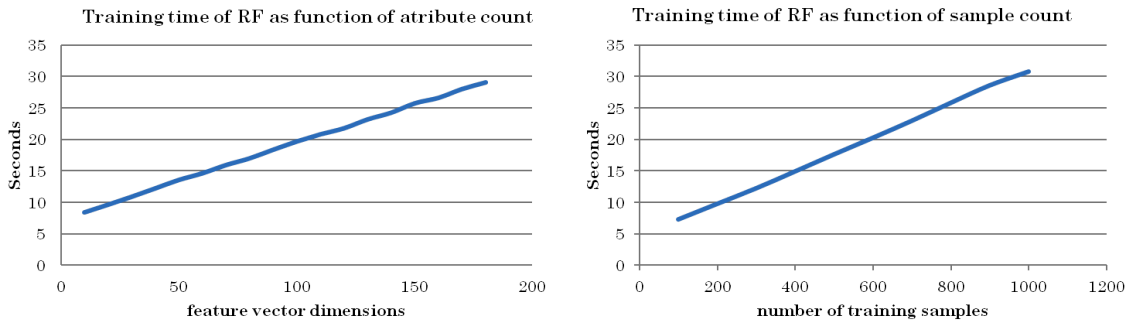


Figure 4.3: Training times on a  $2000 \times 200$  sample training dataset in function of feature dimension and sample count, with default parameters

## 4.2 Optimisations

In this section we discuss a few technical details relevant to the speed optimisation of our system, mainly, speeding up the ART feature extraction as well as the random trees cross-validation through multi-threading.

### 4.2.1 ART optimisation

From an algorithmic point of view, general ART is computed as follows:

---

#### Algorithm 3 ART coefficient extraction

---

Base computation:

```

1: for  $n = 1$  to  $N$  do
2:   for  $m = 1$  to  $M$  do
3:      $ARTbases(n, m) \leftarrow computeBase(n, m)$ 
4:   end for
5: end for

```

Coefficient extraction:

```

1: for  $n =$  to  $N$  do
2:   for  $m = 1$  to  $M$  do
3:     for  $x = 1$  to  $imgResolution$  do
4:       for  $y = 1$  to  $imgResolution$  do
5:          $ARTcoeffs(n, m) \leftarrow ARTcoeffs(n, m) + ARTbases(n, m, x, y) \cdot img(x, y)$ 
6:       end for
7:     end for
8:   end for
9: end for

```

---

We propose two notable improvements:

- Firstly, we notice that  $img$ , being a binary image, has values of either 0 or 1 for background or leaf pixels respectively. Considering that the average total surface of a leaf in an image is about 30% of the total image surface, 70% of computation cycles are wasted multiplying base pixels with 0. Therefore, we use a new coefficient computation algorithm which computes ART coefficients only for non-null pixels.
- Secondly, by computing only bases selected by the attribute selection method presented in Section 4.1.4, we reduce the dimensionality of the  $ARTbases$  structure, allowing for better index optimisation by the compiler.



The efficient algorithm is hence defined:

---

**Algorithm 4** Efficient ART coefficient extraction

---

Base computation:

- 1: Read base index vector  $Biv$  of size  $N_d$
- 2: **for**  $i = 1$  **to**  $N_d$  **do**
- 3:    $ARTbases(i) \leftarrow computeBase(Biv(i).n, Biv(i).m)$
- 4: **end for**

Coefficient extraction:

- 1: **for**  $x = 1$  **to**  $imgResolution$  **do**
  - 2:   **for**  $y = 1$  **to**  $imgResolution$  **do**
  - 3:     **if**  $img(x, y) > 0$  **then**
  - 4:       **for**  $i = 1$  **to**  $N_b$  **do**
  - 5:           $ARTcoeffs(i) = ARTcoeffs(i) + ARTbases(i, x, y)$
  - 6:       **end for**
  - 7:     **end if**
  - 8:   **end for**
  - 9: **end for**
- 

The efficient algorithm represents a 3-fold computational time improvement over the general one for the same number of computed bases. It can be made equivalent with no base selection by generating  $Biv$  with all  $N \times M$  indexes. Furthermore, it produces features ordered by importance.

### 4.2.2 Random Forests multi-threading cross-validation

Due to the training time requirements of Random Forests, leave-one-out cross-validation becomes highly unpractical for parameter testing. However, most modern machines have dual- or quad-core CPUs with simultaneous multi-threading technologies, allowing for efficient execution of up to 8 simultaneous threads. As a result, we have implemented a random forest multi-threaded cross-validation method which delivers leave-one-out results from a 200 dimensional, 7000 sample dataset in about 45 minutes, using 8 threads on a Intel Core i7-2600K Processor with a clock frequency of 3.5Ghz. The method splits the task of testing each sample of the training set  $T$ , of size  $n_{train}$ , into  $n_t$  subsets of size  $n_{train}/n_t$ , retrieving classification errors from each thread and computing the total error. Although random forest have internal estimators for training and cross-validation error based on the out-of-bag classifiers described in Section 3.4.3, we have found that actually retraining the classifier for each test produces more stable error estimates.



## 5. Evaluation

In this section we will look at system performance, comparisons against the state-of-the-art, as well as quantitative and qualitative analysis of how different system configurations and parameters affect performance and execution time.

### 5.1 Datasets

The structure of the data used to test our method is further described. We have collected four different datasets, used in general shape recognition and plant recognition in the state-of-the-art methods in order to have a level-playing-field basis for performance comparison.

#### 5.1.1 MPEG7 CE-1

MPEG-7, formally named "Multimedia Content Description Interface", is a standard for describing multimedia content in a way easily accessible or analysed by computers. It contains both the raw multimedia data as well as descriptors such as ART. The descriptor choices are validated by testing them on general shape datasets. The MPEG7 CE-1 dataset used in this work, as well as in [LJ08, ZL01, LJ05] is a shape dataset, meaning it only contains binary images of various shapes, as shown in Figure 5.1

The dataset has the following properties:

- Size: 1400 images in total
- Class distribution: 70 classes, 20 images each
- Resolution: varying from  $150 \times 150$ px to  $800 \times 800$ px



Figure 5.1: Example of images from the MPEG7 CE-1 dataset

### 5.1.2 Plummers Island 2011

The Plummers island dataset was collected and maintained by the botanists from the Smithsonian Institution National Museum of Natural History. It was assembled as part of a joint effort described in [BCJ<sup>+</sup>08] to create the first working prototype of an automated plant identification system. It is defined as both leaf images and their segmented shapes. The leaf images are of particularly high quality and consistency, being previously flattened and then photographed in a controlled photographic studio environment.

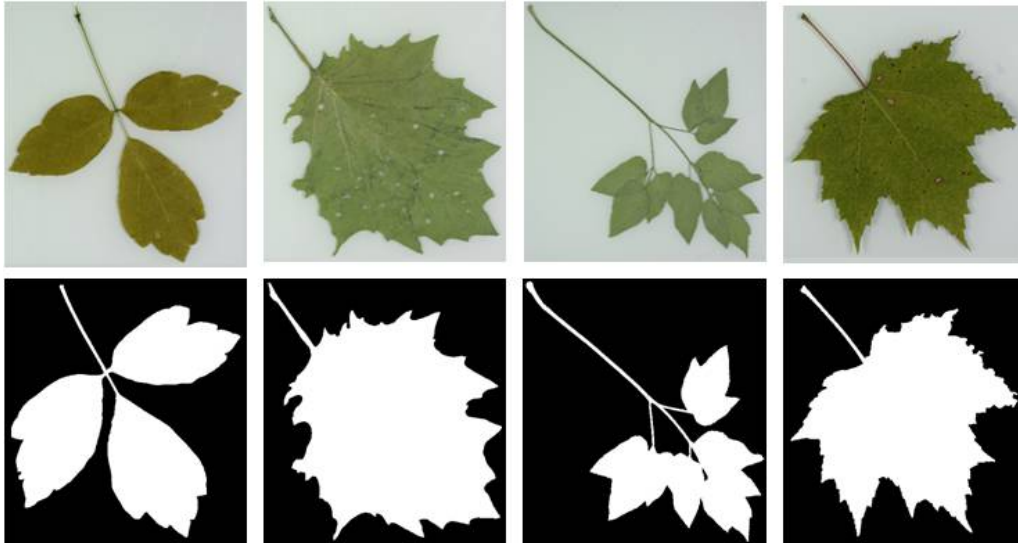


Figure 5.2: Example of images from the Plummers island dataset

It is notable that there is very little lighting and white balance variance between images. The advantage is that the dataset offers as close to perfect samples as possible for training. However, this is not necessarily a positive influence for an automated system as in-the-field images will often exhibit high variances in lighting. Training a system on noisy data may be important for it to learn to discriminate noise from relevant features. It depends, of course, on the descriptors and classifier characteristics, but it is possible for a system trained on perfect samples to behave badly in real-world applications.

The dataset has grown throughout the years, from the 5000 images distributed in 153 classes in 2008 to the current size in 2011 that we use in this work:

- Size<sup>1</sup> : 7152 images in total
- Class distribution: 203 classes, 5-90 images per class, with an average of 30
- Resolution: fixed at 512×512px

<sup>1</sup>actual size is about 14304 images as each leaf is represented in photographs and segmented images

### 5.1.3 Swedish leaves

The Swedish leaf dataset comes from a leaf classification project at Linköping University and the Swedish Museum of Natural History [Sod01]. The dataset was used for the sole purpose of comparing our system against the IDSC descriptor, namely the results specified in [LJ08]. We used the already segmented shapes that came in the IDSC experiment package<sup>2</sup>. The dataset characteristics are as follows:

- Size: 1125 binary images in total
- Class distribution: 15 classes, 75 images per class
- Resolution: fixed at  $256 \times 256$ px

### 5.1.4 ImageCLEF Pl@ntLeaves 2011













Genus Species	Scans	Scan-like photos	Photographs
<i>Acer Negundo</i> L.			
<i>Celtis Australis</i> L.			
<i>Cercis siliquastrum</i> L.			
<i>Nerium Oleander</i> L.			

Figure 5.3: Illustration of the 3 types of images for 4 species from the PL@ntLeaves dataset as shown in [GBJ<sup>+</sup>11]

<sup>2</sup>The IDSC code and datasets can be downloaded at: [http://www.dabi.temple.edu/hbling/code\\_data.htm](http://www.dabi.temple.edu/hbling/code_data.htm)

The dataset used for the ImageCLEF 2011 plant identification task results from a collaborative effort from voluntary contributors mainly from the Metropolitan French territory. It encompasses images from 71 species, 269 individual plants, taken by 17 contributors, producing a dataset of 4500 images.

The plant images are split into three classes:

- Scans: images of flattened leaves against an almost perfect white background
- Pseudoscans: photographs of leaves taken against a uniform background
- Photographs: images of entire plants in nature or of leaves against non uniform backgrounds.

The main statistics of the Pl@ntLeaves datasets are overviewed in the following table:

		Nb of pictures	Nb of individual plants	Nb of contributors
Scan	Train	2325	151	17
	Test	721	55	13
Scan-like	Train	716	51	2
	Test	180	13	1
Photograph	Train	930	72	2
	Test	539	33	3
All	Train	3971	269	17
	Test	1440	99	14

Figure 5.4: Table overviewing important Pl@ntLeaves statistics as shown in [GBJ<sup>+</sup>11]. Corrections were made to the table, as indicated figures were not identical to the dataset made available on-line.

Each image from the dataset is accompanied by a *.xml* file containing information about the content such as plant species and taxon, plant ID, photographer, GPS coordinates. Because the dataset is created by more contributors under varying lighting conditions and styles of photography, it represents more accurately real-world data. Variances for each leaf species are - but not limited to - background, white balance, lighting source (sunny, flash, cloudy, etc.), leaf color, leaf age.

Due to the collaborative nature of the dataset, a bias correction method has been introduced in [GBJ<sup>+</sup>11], its purpose is to distribute the evaluation scores equally amongst contributors and individual photographed plants. The score metric is defined on each testing set as:

$$S = \frac{1}{U} \sum_{u=1}^U \frac{1}{P_u} \sum_{p=1}^{P_u} \frac{1}{N_{u,p}} \sum_{n=1}^{N_{u,p}} s_{u,p,n}$$

Where  $U$  is the number of contributors for the set;  $P_u$  is the number of individual plants photographed by the  $u$ -th user;  $N_{u,p}$  is the number of pictures of the  $p$ -th plant in the set and  $s_{u,p,n}$  is the classification score (0 or 1) for the  $n$ -th picture of the  $p$ -th plant taken by the  $u$ -th contributor.

## 5.2 Comparison with IDSC

IDSC results presented in this section are computed using the original code and data<sup>3</sup> presented in [LJ08]. The Matlab code obtained is however only capable of leave-one-out cross-validation on datasets of 1400 images split into 70 classes with 20 images per class, the same structure as the MPEG-7 CE-1 dataset. In order to correctly compare our system performance with IDSC and because the Plummers Island dataset has grown considerably in the 4 years between the publication of [BCJ<sup>+</sup>08] and this work we have performed a selection from the dataset as follows:

1. select the 70 plant species which have the most images per class from the 203 in the database
2. for each species selected, randomly select 20 images representing it

The dataset constructed in this manner is further referred to as Plummers Island 70.

### 5.2.1 Experiment setup

*Datasets:* MPEG7 CE-1, Plummers Island 70, Plummers Island and Swedish Leaves.

*Descriptors:* ART and FD

*Fusion method:* Early - simple concatenation

*Attribute selection:* none

*Classifier:* Random Forests

*Evaluation method:* Leave-one-out cross-validation

*Parameters:*

- IDSC: 128 contour sample points (as in [LJ08])
- ART: 14 angular  $\times$  7 radial basis, no basis selection
- FD: 32 coefficients
- Early fusion result: 129 dimensional feature vector
- Random Forests: minimum sample count = 10, maximum tree depth = 25, number of active variables =  $\sqrt{129}$ , number of DTs = 600

*Remarks:*

Due to the high sample count of Plummers Island dataset and the need to train 8 decision trees simultaneously for sensible cross-validation times, we were limited by memory in our parameter choices. The default maximum values of  $30 \times 15$  ART bases, 128 FD coefficients and 1000 DTs were reduced accordingly. For the same reasons, the minimum sample count required to split a node has been increased to 10. If the minimum sample count is too low the trees will have too many splits and require large amounts of memory. Whilst these limitations were not necessary for the smaller datasets, we kept these values for consistency.

---

<sup>3</sup>Code and data can be downloaded from: "[http://www.dabi.temple.edu/hbling/code\\_data.htm](http://www.dabi.temple.edu/hbling/code_data.htm)"

### 5.2.2 Results

Method	MPEG7 CE-1	Plummers Island 70	Swedish leaves
IDSC	84.5%	52.8%	94.1%
ART+FD	<b>92.9%</b>	<b>62.1%</b>	<b>96.5%</b>

Table 5.1: Rank 1 results comparison with IDSC

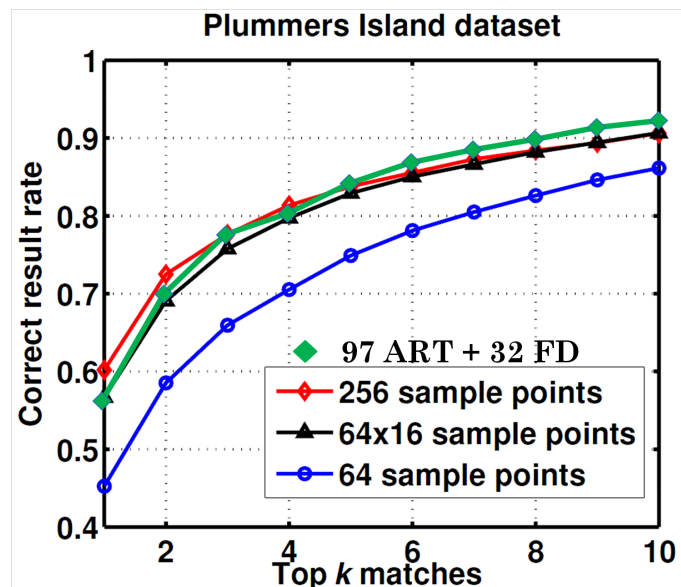
Having a much better precision than IDSC on MPEG7 CE-1 comes as no surprise because ART was chosen as an MPEG7 descriptor due to its performance on this dataset, we notice considerably better performance on the Plummers Island 70 dataset and slightly better on the Swedish Leaves dataset. It is notable that it took our system under 45 minutes to compute the leave-one-out cross validation on the MPEG7 CE-1 or Plummers Island 70 datasets while IDSC required over 20 hours.

We also perform leave-one-out cross-validation on the entire Plummers Island dataset with the following result:

Rank	1	3	7	10
ART+FD	56.2%	77.58%	88.49%	92.23%

Table 5.2: Cross-validation results on the entire Plummers Island dataset

Plotting the ranked precision over the IDSC results image provided in [BCJ<sup>+</sup>08] shows the relative performance:

Figure 5.5: IDSC results from [BCJ<sup>+</sup>08] with overlaid ART+FD results



We notice that we obtain very similar performance measures as IDSC, slightly outperforming it, on a much more complex version of the dataset. The original performance plot, taken from [BCJ<sup>+</sup>08] is obtained by running leave-one-out cross-validation on the 2008 version of the Plummers Island dataset, composed of 157 classes and approximately 5000 images in all. Our algorithm was tested with the same validation scheme, however, on a version of the dataset composed of 203 classes and approximately 7100 images. Because our results are based on a larger dataset, containing 46 more classes, as well as the fact that classification error generally increases with the number of classes, we expect the difference between our method and IDSC to be actually larger than this direct comparison. By comparing our cross-validation run time of under 45 minutes with the 20 hours necessary for IDSC to cross-validate, we note the significant efficiency difference between the two methods.

### 5.3 Performance on ImageCLEF 2011 Pl@ntLeaves dataset

In the following section we look at our system performance on the Pl@ntLeaves dataset, discussing parameter choices and comparing final results with those from other groups participating at ImageCLEF 2011.

In the first step we only use the scans and psudoscans from the training part of the dataset, 3041 images in all. We split the data by randomly sampling approximately 20% of the training images from each class, thus constructing a test dataset we use for parameter validation. When resampling, we respect the original proportions of the number of images per class, thus achieving a balanced testing set of 580 images, containing all 71 classes in the same proportion as the training set. As we want to train our system for optimal performance indifferent of the plant or contributor, we apply the bias correction method described in Section 5.1.4 when validating our results.

In the second step, we apply a similar split on natural photographs from the dataset and analyse SIFT+BoW performance.

In the third step, we run our system on the test part of the dataset, comparing final results with those from the ImageCLEF 2011 plant identification task. The bias correction method is once again employed.

Although not detailed in this section, the DEG descriptor was also tested on the Pl@ntLeaves dataset with both Random Forests and KNN classifiers. No matter the parameter choices it did not exceed a rank 1 precision of 35% on any of the datasets. Due to its much lower score than the other descriptors, we decided not to include a detailed comparison, although it is included in the implementation so that it can be tested in future versions of this system, preferably with other classifiers.

### 5.3.1 Scan and Scan-like image training

*Datasets:* Pl@ntLeaves 2011 training set, split into 580 test images and 2461 training images containing 71 classes each

*Descriptors:* ART, FD and SIFT-BoW for scans and pseudoscans

*Fusion method:* Late - weighted confidence mean

*Attribute selection:* none

*Classifier:* Random Forests and KNN

*Evaluation method:* same as in [GBJ<sup>+</sup>11] on selected training subset

*Parameters:*

- ART: 30 angular  $\times$  15 radial basis, 200 $\times$ 200px resolutions, no basis selection
- FD: 128 coefficients
- SIFT-BoW: DoG keypoints on light intensity channel, max keypoints = 2000, vocabulary size = 500
- Random Forests: minimum sample count and number of active variables are optimised; maximum tree depth = 25, number of DTs = 1000 remain constant
- KNN has been used as nearest neighbour classifier, with  $K = 1$

*Remarks:*

We aimed to have the highest possible values for feature size while still remaining computationally feasible. We allow this reasoning as the architecture of our descriptors means that adding more features has little chance of being destructive for classification. Late fusion was used in which the confidence vectors from each descriptor were weighted. The reasoning applied was that shape descriptors and local descriptors do not perceive the data in a similar way, some classifying certain species better than others. This can lead to contradictory best split computation in the Random Forest and decrease its performance. In a similar fashion 1NN is sensitive to scaling of each attribute and early fusion would have counter-productive effects. The optimal late fusion weights are found by brute force.

Random Forest parameters are optimised, searching for values between 1 and 100 for both minimum sample count and the number of active variables by training each tree 10 times, averaging the final result and selecting parameter values which achieved the best rank 1 bias corrected precision. We first searched for the optimal number of active variables with a fixed minimum sample count of 1, then for the best minimum sample count with the number of active variables fixed to its optimal value. It is notable that generally a minimum sample count of between 1-2 produces optimal results but also very large forests because of the high number of splits. We have found that the parameter can usually be increased to 10 without loss of precision, sometimes actually improving performance. After our tests we find the following optimal random forest parameter values for each descriptor:

- ART: minimum sample count = 15, number of active variables = 30
- FD: minimum sample count = 13, number of active variables = 20
- SIFT+BoW: minimum sample count = 5, number of active variables = 30

Firstly, we look at the performance of individual descriptors, classified with both 1NN and Random Forests and without any bias correction, in order to highlight the bias effects in later comparisons.

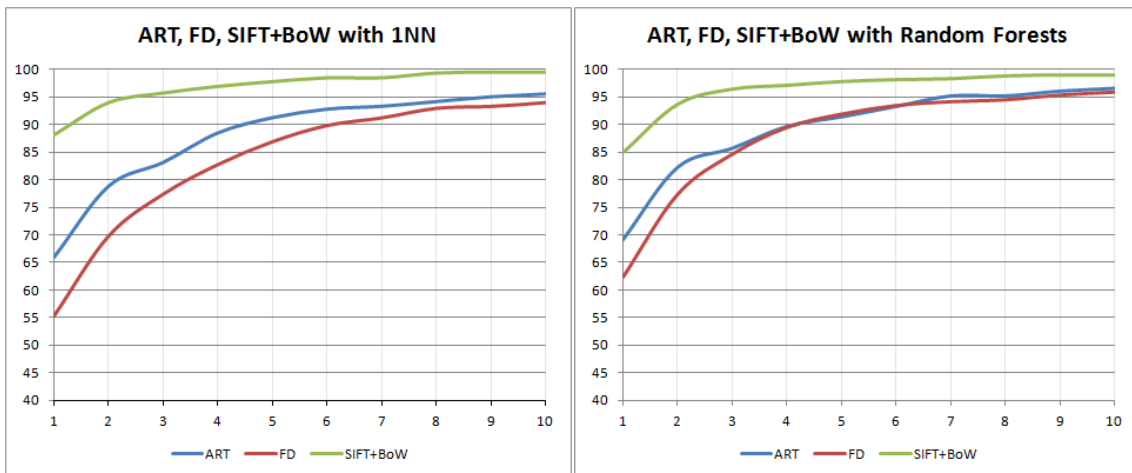


Figure 5.6: Ranked results of individual descriptors on extracted test set, without bias correction. Ranks are on the horizontal axis, vertical shows precision in %

The SIFT+BoW has surprisingly good results on scans-like images, outperforming both ART and FD, showing over 95% chance of providing a correct result in the first 3 returned matches. The difference between descriptors and classifiers used, however, becomes smaller when looking at higher rank results: no matter the descriptor or classifier, there is a greater than 90% chance to obtain the correct class in the first 6 proposed matches. We also note that Random Forests produces better results than 1NN while classifying our shape descriptors, ART and FD. If we consider the overall classifier's performance as an average on all three descriptors, Random Forests has an edge over 1NN, however not by much: average RF rank 1 classification score being 73% versus 69% for 1NN.

We will now analyse the influence of bias correction:

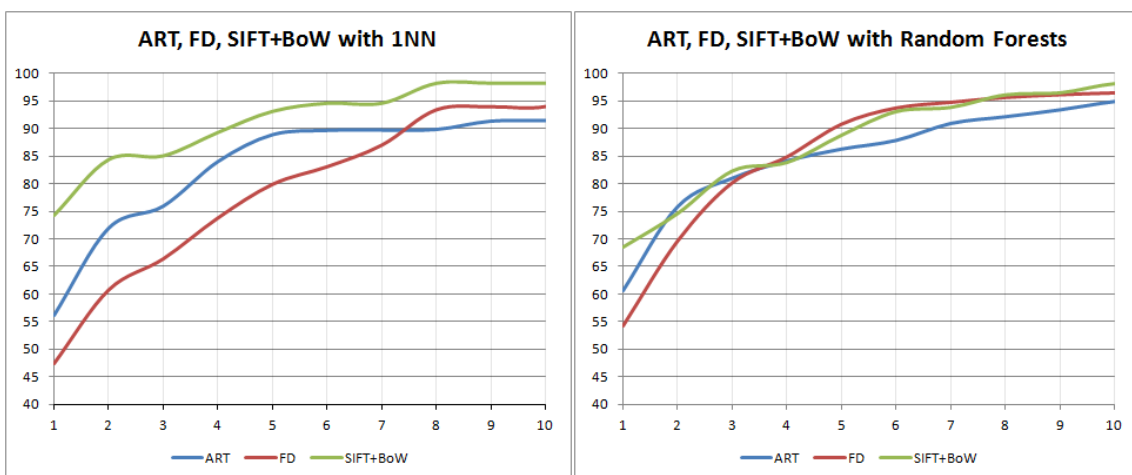


Figure 5.7: Ranked results of individual descriptors on extracted test set, with bias correction. Ranks are on the horizontal axis, vertical shows precision in %

Having applied the correction, the plots become less smooth because of high value classifications: images in the training set vary from 1 to 13 per plant, some being more valuable than others. When applying this correction all plants and contributors get the same importance and some images belonging to plants that are not well represented in the dataset gain more weight. It is important to look at corrected results to avoid optimising the algorithm only for plants that are well represented in the dataset. The same remarks as before apply, SIFT+BoW having the best performance with 1NN. General score has deteriorated, especially in the first 3 ranks, where it is over 10 percentage points lower. The strongest negative influence of the bias correction was on SIFT+BoW and RF, where the rank 1 score plummeted by 20 percentage points, meaning that it achieved its previous high score by correctly classifying images from well represented plants. We notice ART and FD generalise better when the sample count per class is smaller and especially the good performance of simple FDs classified with Random Forests.

Once individual descriptors and classifiers have been optimised, we will search for the best late fusion ratios used to obtain the final confidence vector:

$$Conf(f_s) = Ratio_{ART} \cdot Conf_{ART}(f_s) + Ratio_{FD} \cdot Conf_{FD}(f_s) + Ratio_{SIFT} \cdot Conf_{SIFT}(f_s);$$

Due to the small dimensionality of the optimisation problem, we search for the best  $Conf(f_s)$  by looking at all possible ratio combinations in  $[0..1]$  in steps of 0.1. This process is done on both Random Forests and 1NN, resulting in the following best ratios:

- Random Forests:  $Ratio_{ART} = 0.4$ ,  $Ratio_{FD} = 0.2$ ,  $Ratio_{SIFT} = 0.4$
- 1NN:  $Ratio_{ART} = 0.1$ ,  $Ratio_{FD} = 0.1$ ,  $Ratio_{SIFT} = 0.8$

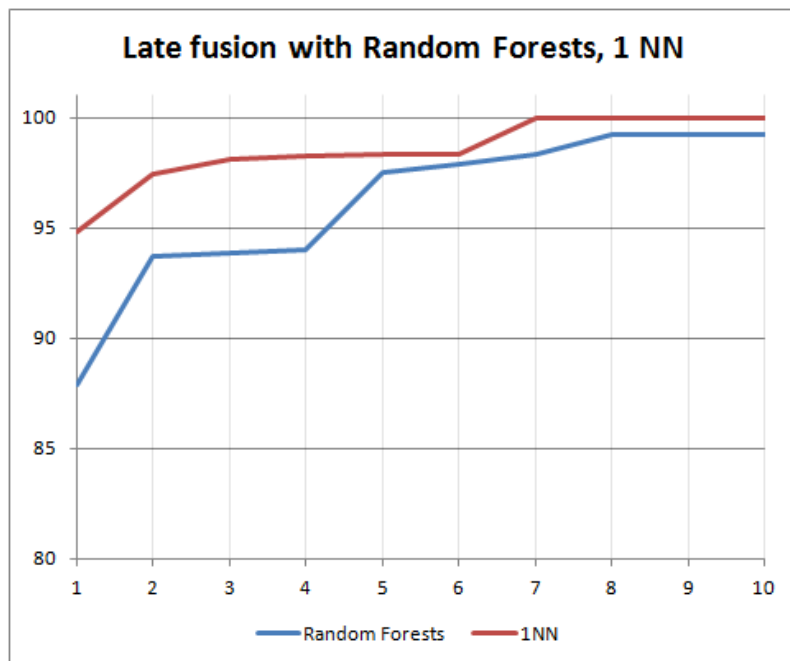


Figure 5.8: Ranked results of late fusion with optimal parameters and bias correction. Ranks are on the horizontal axis, vertical shows precision in %

The late fusion results show a considerable improvement over the individual descriptor results. 1NN outperforms RF, mainly because of the high classification rate of SIFT features. Indeed, the optimal ratios found show that the fusion achieves this high result by using mostly SIFT based confidences. Fusion on the Random Forests shows much more balanced results, with ART and SIFT receiving equal importance in the final result. We should interpret these high precision rates only as validation on the training set and it comes naturally that optimising parameters on this set produces excellent results. Therefore we must judge the true performance of the system as well as the generalisation of our parameter choices on the testing set.

### 5.3.2 Photograph training

In order to train SIFT-BoW on photographs we used the same parameters and method as for scan-like images with the exception of the keypoint detection method. The 930 image training dataset was split into 200 testing images and 730 training images. Because the natural backgrounds contain much more noise than the scan-like images, many intensity channel-based keypoints focus on those areas instead of the leaves themselves. We look to improve on this by using other channels for keypoint detection as described in Section 3.3.1. Descriptor information however was still based on the intensity channel information.

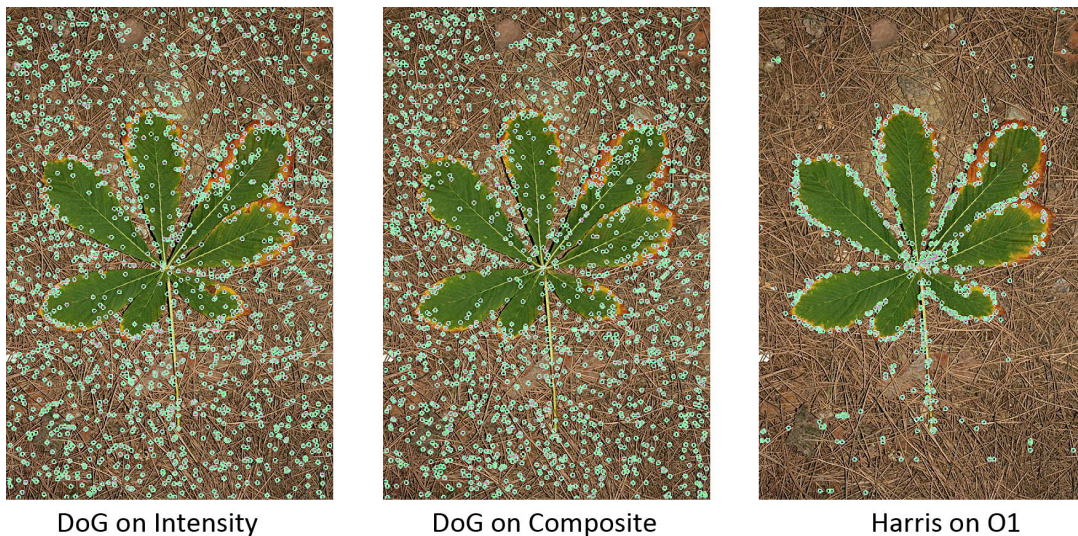


Figure 5.9: Keypoints detected by Difference of Gaussians on Intensity and Composite channels and by Harris corner detection on color opponent channel detailed in Section 3.3.1

The choices of keypoint detectors and color channels were based entirely on qualitative observations of their behaviour on photograph images, their general performance being compared in Table 5.3.

Although it seems that the Harris method is superior in identifying interesting gradient points on and around a leaf, the test results indicate other-wise:

Rank	1	10
DoG on Intensity	16.3%	55.2%
DoG on Composite	13.9%	53.6%
Harris on opponent channel	1.3%	21.2%

Table 5.3: SIFT+BoW results with different keypoint extraction methods

To explain this unexpected behaviour we can offer two main reasons. Firstly, the Harris corner detector tends to stay on leaf edges, not offering any gradient information from the leaf surface. Secondly, the correlation between the  $O1$  channel and the intensity channel may be low, therefore keypoints detected on  $O1$  will not correspond often to similar gradient variations on the intensity channel, producing very different SIFT descriptors. For testing on photographs we therefore use simple SIFT+BoW with DoG keypoints on the intensity channel.

### 5.3.3 Testing

*Datasets:* Pl@ntLeaves 2011 training and testing sets,

*Descriptors:* ART, FD and SIFT-BoW

*Fusion method:* Late - weighted confidence mean

*Attribute selection:* none

*Classifier:* Random Forests and KNN

*Evaluation method:* same as in [GBJ<sup>+</sup>11]

*Parameters:*

- ART: 30 angular  $\times$  15 radial basis, 200 $\times$ 200px resolutions, no basis selection
- FD: 128 coefficients
- SIFT-BoW: DoG keypoints on light intensity channel, max keypoints = 2000, vocabulary size = 500
- Random Forests: optimal parameters found in training, as detailed in Section 5.3.1
- KNN has been used as nearest neighbour classifier, with  $K = 1$

*Remarks:*

As described in the ImageCLEF plant image classification task report, we test our system on scan, pseudoscan and photograph type images separately and apply the bias correction described in Section 5.1.4 for each test sample classification:

### 5.3. Performance on ImageCLEF 2011 Pl@ntLeaves dataset

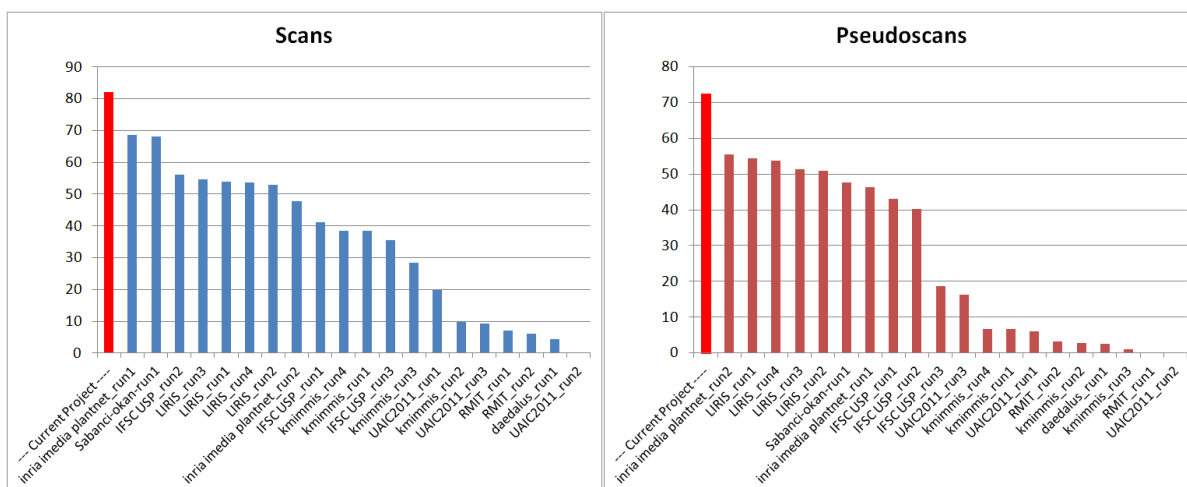


Figure 5.10: Results of the current system on scan and scan-like images compared to participants from the ImageCLEF 2011 plant recognition task

General system performance on scan and scan-like images is outstanding, around 12 percentage points higher than other methods, reaching scores of 81% and 71% on scans and pseudoscans respectively.

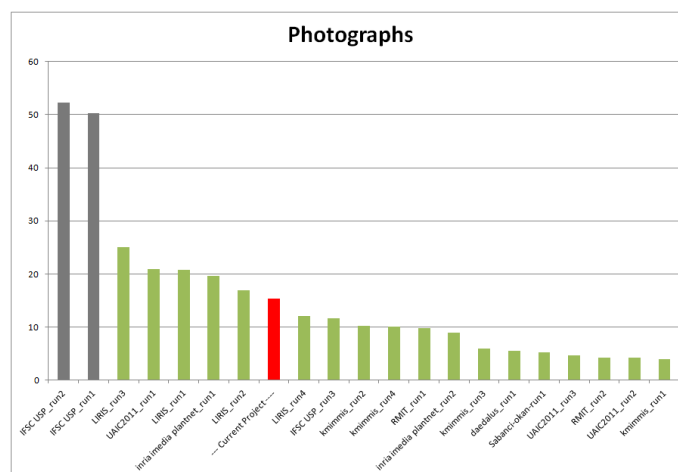


Figure 5.11: Results of the current system on natural photographs compared to participants from the ImageCLEF 2011 plant recognition task

On photographs, however, the SIFT+BoW descriptor alone does not excel, with a score of 15.5% but still outperforming most local feature based methods. The highest automated classification score to outperform was 25.1%, achieved by the LIRIS group and their model-based method, the IFSC/USP photographs having been manually segmented.

In spite of the average results on photographs, looking at the general score of our system and comparing it against the other ImageCLEF methods still shows a non negligible advantage of 7 percentage points over the best submission with manual intervention - IFSC/USP - and 16 percentage points over the best fully automatic submission - INRIA.

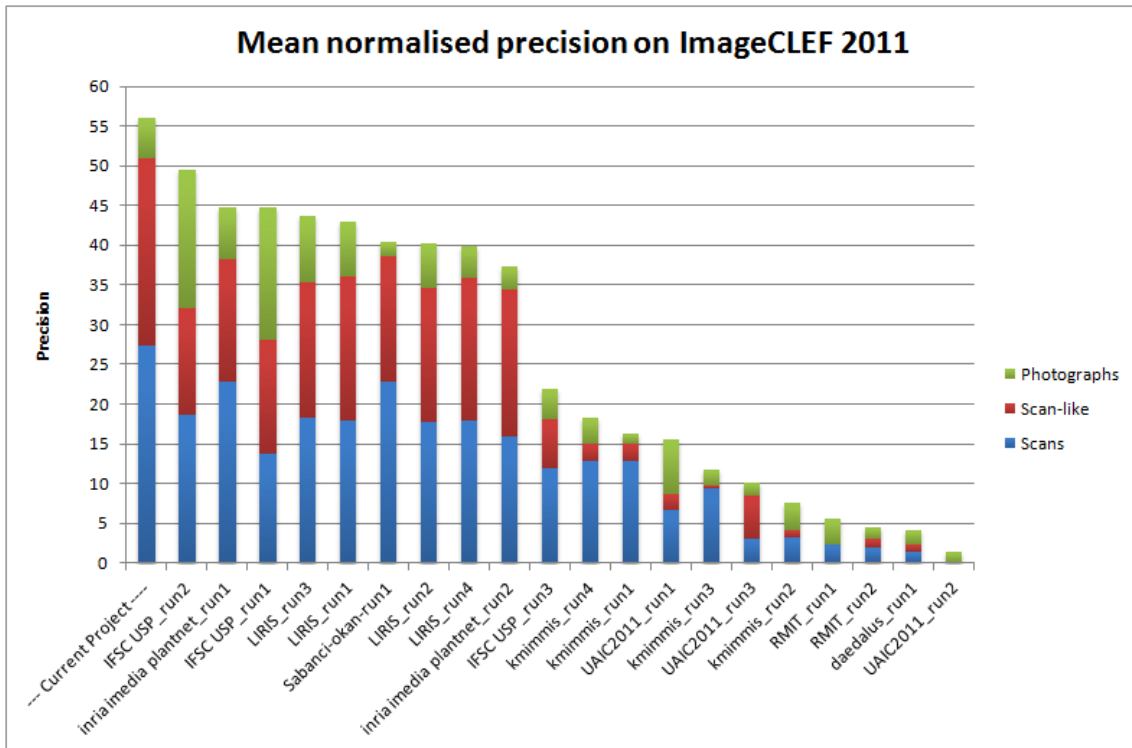


Figure 5.12: General score overview on the Pl@ntLeaves dataset

The general score plot has been recreated using the scores from the table shown in Section 2.2. We have also inserted our results for comparison, under the name of "— Current Project —". We notice that these results are mainly based on very good scan-like image classification, where our system clearly stands out from the rest. For final results and in contrast to the training experiment, the bias correction actually improved our score by approximately 5 percentage points. Analysing the output, we indeed find a few plants that have many images with very high leaf shape variance at which classification fails. These plants are the same as those noted in [GBJ<sup>+</sup>11] for having overall lowest recognition rates.

To better analyse the contribution of each descriptor and more specifically the importance of local features for our final result, we look at the performance achieved by ART, FD, SIFT individually, then fusing only shape features, then fusing with local features as well, classified by Random Forests as well as 1NN. Late fusion ratios between ART and FD were computed in an identical manner as in Section 5.3.1, on the training and validation subsets of the training set. We thus find the optimal shape descriptor ratios for ART and FD only,  $Ratio_{ART}$  and  $Ratio_{FD}$ , to be (0.6,0.4) and (0.7,0.3) for RF and 1NN respectively.



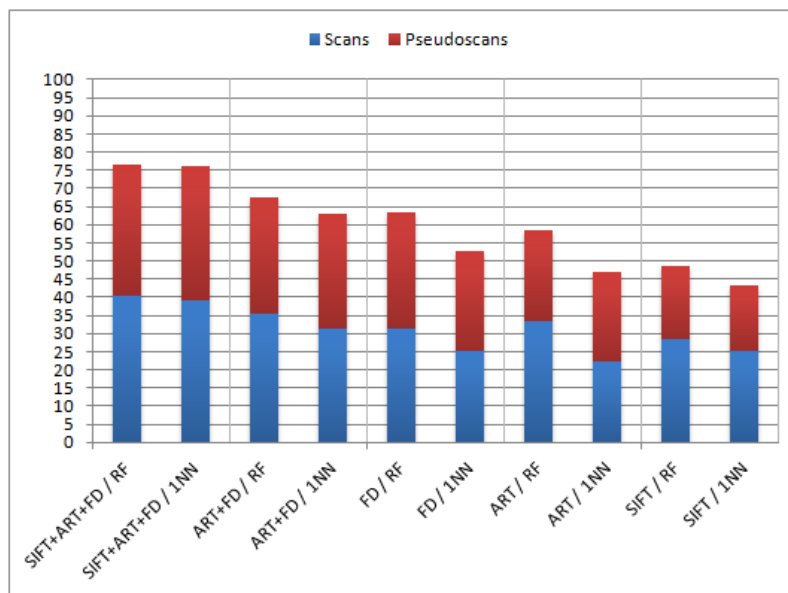


Figure 5.13: Overview of average rank 1 descriptor and classifier performance on the Pl@ntLeaves scan and pseudoscan test sets

The most noticeable difference between the test results and those on the validation set is that RF outperforms 1NN, showing better generalisation. The reason for this difference in performance may come from the way we split the training data into training and validation subsets. We did not take into account the user or the plant from which the images were selected, having images from the same plant in both training and validation subsets. This can introduce unwanted correlation between the two sets, which is not present between the training and the testing set. This would also explain the fact that SIFT-only performed very well on the validation test but poorly on the test set. When testing with the actual test set, where some plants are restricted only to the test set, we have a better real-world performance estimation. Late fusion also performs well as each new fused feature improves precision by around 10 percentage points. Fusion of shape descriptors only, although not as good as fusion of local and shape features, still provides better results than the best scores on scans and pseudoscans from other methods, albeit by a smaller margin.

FD, the simplest descriptor we used shows surprisingly good performance by itself when classified with RF. When comparing its performance to other systems from the ImageCLEF 2011, we notice indeed that it outperforms all of them in the average score on scan and pseudoscans. This shows that although it is one of the oldest shape descriptors, when combined with a strong classifier, it can offer very good results.

RF	SIFT+ART+FD	ART+FD	FD	ART	SIFT
Scans	80.9	71.06	62.79	66.69	56.77
Pseudoscans	72.24	63.89	63.93	50.34	40.04
1NN	SIFT+ART+FD	ART+FD	FD	ART	SIFT
Scans	78.23	62.71	50	44.22	50.30
Pseudoscans	73.82	63.34	55.74	49.64	35.85

Table 5.4: Average rank 1 descriptor and classifier performance on the Pl@ntLeaves scan and pseudoscan test sets

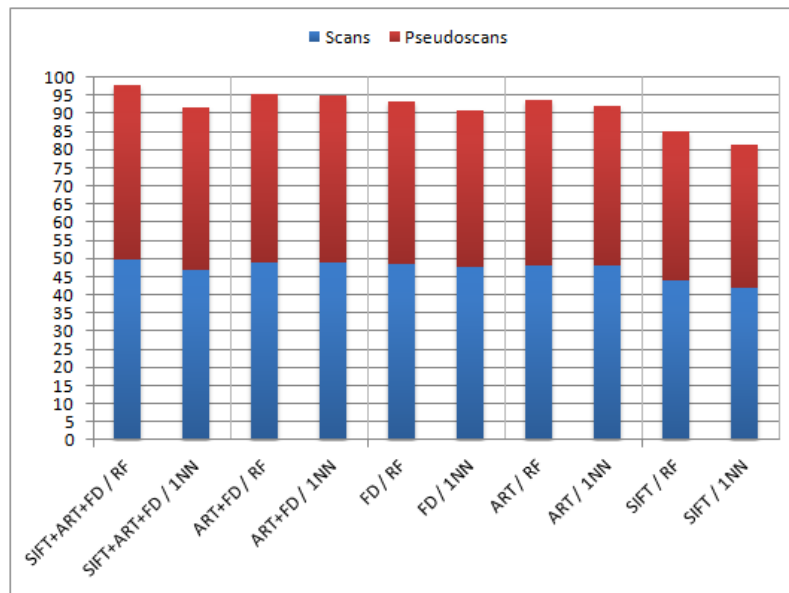


Figure 5.14: Overview of average rank 10 descriptor and classifier performance on the Pl@ntLeaves scan and pseudoscan test sets

We also look at the rank 10 classification results, which show us that RF performs better at finding relevant matches early. All shape descriptors and fused results have a 93% or higher probability of correctly matching a leaf in the first 10 results. It is also interesting to note that in the case of 1NN, fusion of shape descriptors with local features is actually detrimental for rank 10 performance, most probably because fusion parameters were optimized on rank 1 results.

RF	SIFT+ART+FD	ART+FD	FD	ART	SIFT
Scans	99.66	98.04	96.95	95.87	87.94
Pseudoscans	95.86	92.70	89.24	91.39	82.29
1NN	SIFT+ART+FD	ART+FD	FD	ART	SIFT
Scans	93.73	97.31	95.12	96.01	83.96
Pseudoscans	89.31	92.13	86.63	88.08	79.07

Table 5.5: Average rank 10 descriptor and classifier performance on the Pl@ntLeaves scan and pseudoscan test sets

## 5.4 Attribute selection and optimisations

Having analysed our system’s performance with large feature dimensionality and determined its advantage over the state-of-the-art, we now focus on its scalability. An efficient system is not only important for portability to small devices, it also ensures that it is a veritable solution for large databases. If our system were to be useful in a real-world application, the datasets will become much larger than a few thousand images and a hundred classes. In such context, it is important to optimise speed as much as possible, without sacrificing recognition quality. Considering the algorithm for image segmentation requires little processing power, we focus our optimisation task on speeding up feature extraction and classification times. As mentioned in Sections 4.1.5 and 4.1.3, Random Forests training time, KNN testing time and ART’s feature extraction time, each grow linearly with feature dimensionality. We thus apply the attribute selection methods described in Section 3.4.4 and analyse how much we can reduce the dimensions of the feature spaces without sacrificing precision.

Using the exact setup we used in Section 5.3.3, we compare attribute importance as perceived by RF and our defined discriminant function  $D(a_i)$ . All attribute importance computations are done on the training set only.

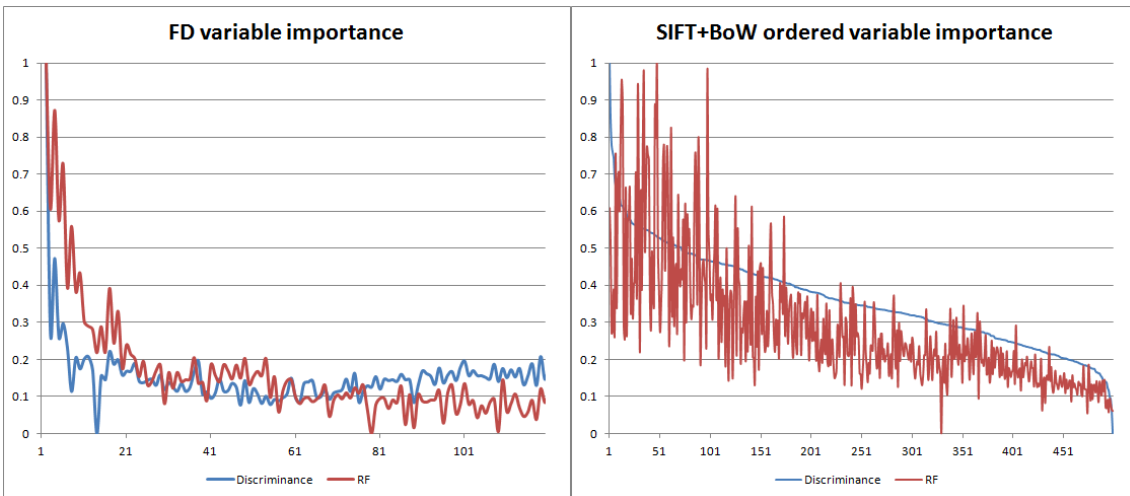


Figure 5.15: Variable importance of FD and SIFT+BoW descriptors, as computed by RF and  $D$ . SIFT+BoW was ordered by discriminance to better show correlation with RF output. FD graph plots the square root of the variable importance for better readability. Importance is on the vertical, attributes are on the horizontal axis.

We notice that the two importance measures show strong correlation, in the sense that they often produce peaks on the same attributes, following the same trend. Because of the noisy importance measures on SIFT+BoW, the attributes were ordered by descending discriminance value for plotting, showing that this sorting also produces a descending trend in the importance measured by RF. Interestingly, we observe a rise in importance of higher frequency FD coefficients, that we attribute to the importance of edge detail in a leaf (jagged, smooth, etc). We also note that many authors ([LJ08, AL09, YAT11, CFB11]), when comparing their systems against FDs, only take the first 20-40 FD coefficients. While

these are indeed the most important, the above analysis shows that higher frequencies also contain discriminative information for leaf classification.

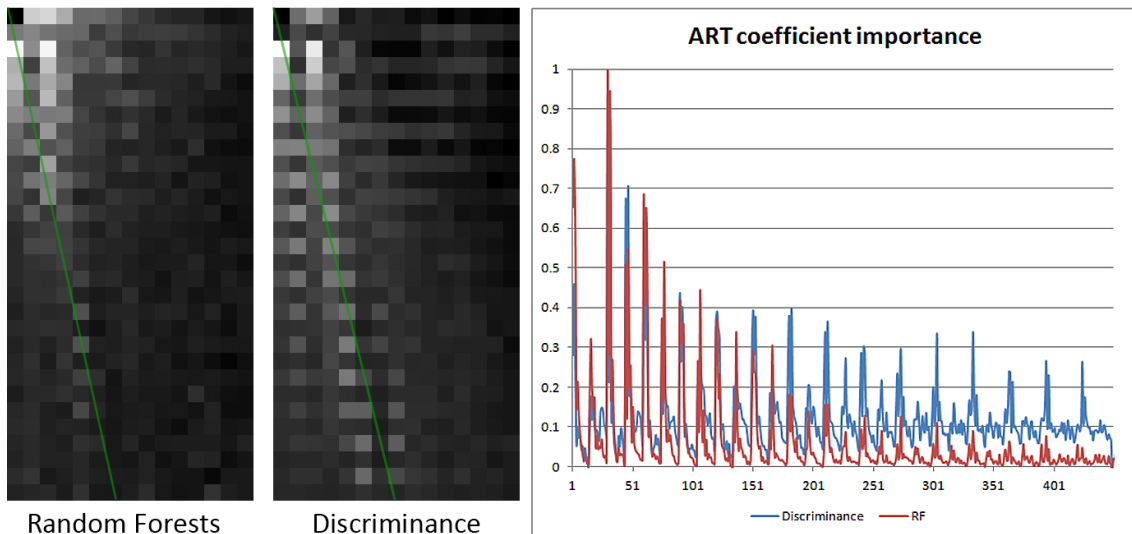


Figure 5.16: ART coefficient importance as computed by RF and  $D$  plotted as grayscale map of  $15 \times 30$  radial and angular degrees as well as a function of descriptor attribute, with importance on the vertical and attributes on the horizontal axis.

As was the case before, we notice strong correlation between the RF variable importance computation and  $D$ . Analysing the grayscale maps, we also notice a trend line - shown in green - in the important frequencies, which confirms the empirical rule of choosing radial and angular degrees with a ratio of  $1/2-1/4$ . The distribution of important bases in the radial  $\times$  angular maps also shows that when selecting a fixed number of degrees to compute, many of the bases are unimportant, the majority being concentrated around the trend line defined by a  $1/4$  ratio of radial to angular frequencies. Considering the strong correlation between importance functions and the faster computation of  $D$ , we use  $D$  as our main importance measure and analyse the effects of selecting features based on its value.

We further look at how rank 1 and rank 10 precision varies in function of the number of attributes selected. For good comparative plotting, FDs were recomputed with 512 coefficients and all the attribute selection count is stopped at 450 for all descriptors. The score showed is bias corrected on the scan-type images from the test set. We chose only scans because they are more statistically significant than pseudoscans (720 test images versus only 180).

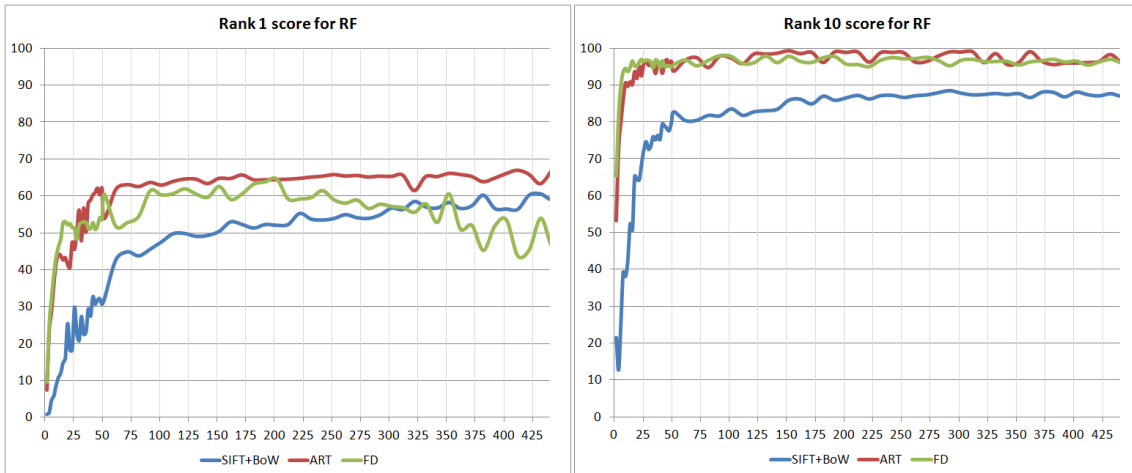


Figure 5.17: Score of descriptors with RF as a function of number of attributes selected.

Although the plots manifest noise from bias corrections and randomness of the classifier, we analyse the general trendline. We note the excellent scalability of ART and FD features, 50-100 discriminative coefficients being enough to obtain almost maximum precision. We also notice that attribute selection is favorable for FD, where the non-important coefficients add noise to the classification, lowering scores. SIFT+BoW however, does not present the same amount of scalability, constantly improving on precision until the last attribute was added. Selecting 100 SIFT+BoW attributes from 500 leads to 10 percentage points score loss. Rank 10 results show that even 25 attributes for ART and FD each is enough to obtain a score of 95%.

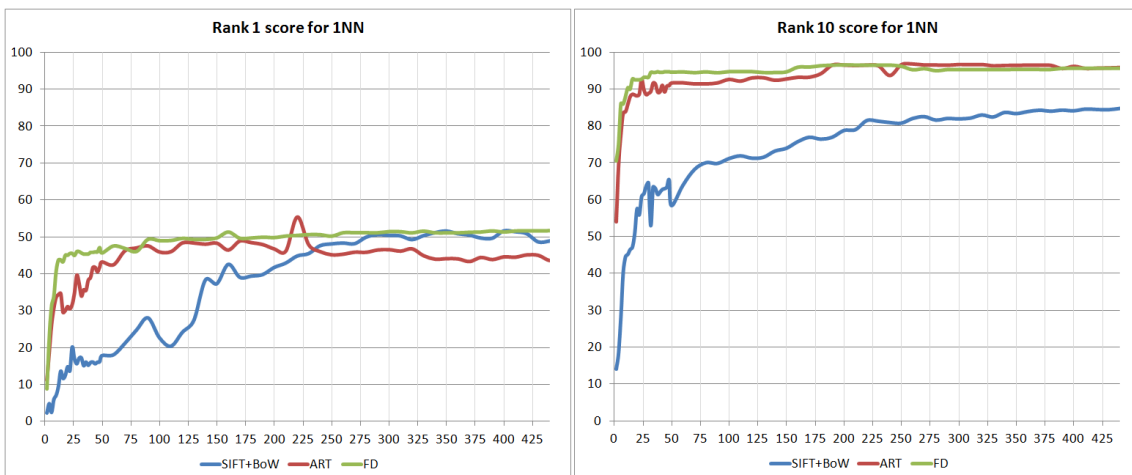


Figure 5.18: Score of descriptors with 1NN as a function of number of attributes selected.

Plots from the 1NN classifier show similar results, albeit at lower scores. We notice 1NN is worse than RF at handling correct classification with a lower number of attributes. Rank 10 results however are somewhat closer to RF but still lower.

Execution times are now analysed and due to the fact that SIFT+BoW takes 1.2 seconds in average to obtain features from an image and its bad scalability, we focus our attention on optimising our shape descriptors: ART and FD. Comparing the precision obtained on scan-like images using ART+FD with 50 selected attributes each and late fusion with our previous best scores, we see little difference between the two:

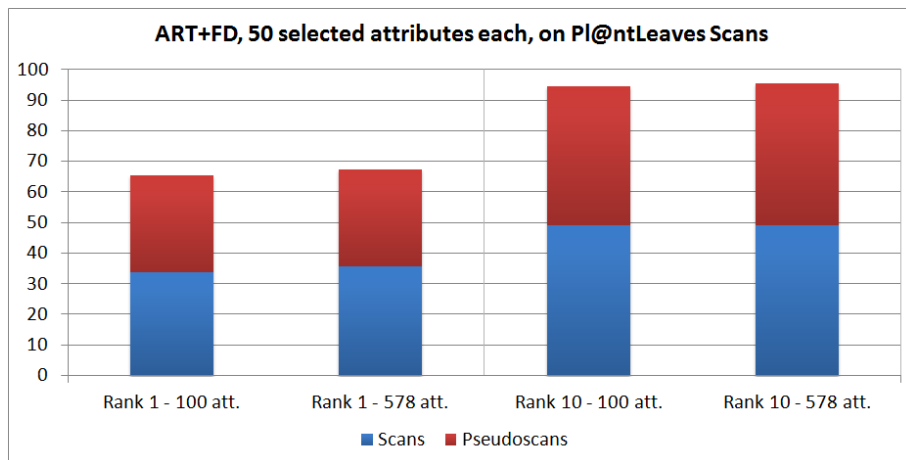


Figure 5.19: Comparative score of system performance with selection of attributes and without

At least concerning shape descriptors, we are able to reduce the dimensionality from 578 attributes to 100 with little, if any, precision loss. We now look at the actual computation times of our system on a Intel Core i7-2600K Processor with a clock frequency of 3.5Ghz:

Attribute count:	50	150	450	50/N	150/N	450/N
Random Forest testing	0.7s	0.7s	0.7s	0.97ms	← <i>idem</i>	← <i>idem</i>
Random Forest training	30.7s	65.6s	202.9s	10ms	21.6ms	66.6ms
KNN testing	5.1s	7.2s	14.3s	7ms	10ms	19.8ms
FD extraction	37.7s	← <i>idem</i>	← <i>idem</i>	112ms	← <i>idem</i>	← <i>idem</i>
ART extraction	41.6s	106.8s	326s	126ms	325ms	1s
Method:	K-means/N		Derivative/N	Otsu/N		
Segmentation time per image:	110ms		50ms	63ms		

Table 5.6: Execution times for different elements of our system on the Pl@antLeaves dataset. Training has been done on the 3041 scans-like images, testing has been done on the 721 scan images. Normalised scores represent processing time per image.

Results are obtained from single-core computation and can still be improved by parallelisation of ART feature extraction, which is the most computational intensive task for shape description. The run time of FDs can also be decreased by reducing the contour resampling size. We notice that we can classify about 3 images per second with 50 ART and 50 FD selected attributes. It is reasonable to assume that classification times will be of the order of seconds on slower devices, still acceptable for our task. However, there are

no efficiency benefits from reducing FD coefficients. The logarithmic and constant testing complexity of RF is made obvious by the extremely low classification times. Training times are also strongly reduced through attribute selection: if we were to consider that training times of 5 hours or more are acceptable for real world applications, we could learn on a database of millions of specimens, making RF a sustainable classifier for the plant identification problem. It is notable that shape descriptors, while offering good performance, are 3 times as fast to compute as SIFT+BoW, which takes 1.2 seconds per image in average. On a more powerful machine however, both shape and local features can be used, with average recognition times of about 2 seconds. Nevertheless, most of the computation time requirements for SIFT+BoW come from the DoG keypoint detector. If used with Harris or MSER detectors, the feature extraction times improve ten-fold, taking 180ms per image, much closer to the ART and FD extraction times. Unfortunately, using such keypoint detection methods drastically decreases precision. A better keypoint detector or local feature altogether would further improve the ability of this system to scale, while offering high precision results, as SIFT+BoW improves our matching scores by a non negligible 10 percentage points.





## 6. Conclusion

In this work, an efficient image based automated plant recognition system is described as a prototype for real-world applications. The plant identification task was modeled through the use of shape and local features, with emphasis on shape descriptors, and classification of these descriptors into plant species. Building upon already tested methods for plant classification such as Fourier descriptors, Nearest Neighbour classifier and SIFT features, we introduce the Angular Radial Transform as a new leaf shape feature as well as Random Forests as a leaf feature classifier with good precision and generalisation properties. Local features were used to complement shape information, proving their importance in the plant identification problem. An optimisation scheme is proposed for efficient implementation with minimal loss of classification quality, the resulting scalability of the system to both slow devices and very large datasets being highlighted.

Starting from the most relevant leaf features used by botanists to identify leaves - general shape, contour detail and texture, we have designed our system to use three general descriptors to encode each respectively: ART, FD and SIFT+BoW. While designing our system, we have compared individual descriptor performance, as well as descriptor fusion results, and analysed the importance of each one. We have tested two classification methods: Nearest Neighbour, which is proven to be effective on a number of descriptors in related works, as well as Random Forests which has not been mentioned in any previous plant identification methods, while noting their computational complexities and real-world usage advantages and disadvantages and declaring Random Forests as the better classifier. The ranked classification scores were taken into account due to their usefulness for the users of our system. Tests were done on the Pl@ntLeaves dataset in order to validate system configuration choices on data originating from our target users. This was desired, as images taken in highly controlled environments, such as the Plummers Island dataset are not as representative for real-world use cases.

The results of comparisons with state-of-the-art methods are favorable for the system presented in this work. We firstly compared our shape descriptors and Random Forest classifier against the IDSC+DP method which was implemented in the first working prototype of a real-world plant identification system. Our method outperformed IDSC on all

datasets. The classification times we had were considerably better, as Random Forests classifies in constant time while IDSC+DP does so in quadratic complexity. Results from comparisons with methods from the ImageCLEF 2011 plant identification task also show favorable scores for our shape features. Late fusion of ART and FD descriptors alone outperforms the best results on both scan and pseudoscan images from other methods, however by a small margin. The introduction of SIFT+BoW features and fusion with shape features significantly improves the score on these images, boosting results to a 13 percentage point margin between our system and the next best on scan-like images. Tests on unconstrained photographs of leaves and entire plants from Pl@ntLeaves were performed using SIFT+BoW only, as segmentation of leaves on such diverse backgrounds proved unfeasible. Results were averaged, achieving a score of 15.5%, 10 percentage points lower than the best fully automated method from the LIRIS group. Ranked classification was computed, showing that on scan-like images we produce rank 10 classification scores over 90% on all datasets, including the 203-class Plummers Island. It was noted that on ImageCLEF scan-like datasets, rank 10 scores were over 90% with shape descriptors, regardless of classifier or descriptor, however Random Forests proved to provide higher scores starting from lower classification ranks. If speed is of the essence and a user would not be bothered by finding the correct match in the first few results and not necessarily the first, even FDs with RF only offer good enough performance for useful plant identification.

Optimisation of the system based on feature dimensionality reduction through important attribute selection has been described. We noted ART to be the main computational intensive shape descriptor and proposed a method of reducing its computational time without noticeable loss of system precision. After optimisation, ART extracts features at a rate of 126ms per image, very close to the 112ms for FD. By reducing feature vector dimensions, the system is able to segment, extract features and classify samples at an average rate of 3 frames per second. We have equally analysed classifier performance and highlighted the excellent scalability of the Random Forests classifier on very large datasets, which are probable to develop when automated plant identification will become a standard tool for botanists or even a crowdsourced application.

Notable directions of improvement for the current work are in unconstrained photograph classification where our method did not excel. However, even other methods presented at ImageCLEF 2011 achieve a maximum score of 25.1%, which is too low to be practical for real-world use. Improving on local features for plant recognition will represent a major breakthrough, making the technology maturer and more usable by a larger target public. Speeding up the local feature extraction process is also another important research direction as it is currently the slowest of all the descriptors used in this work, making it unusable on slow portable devices. A system capable of recognising plants from the video-feed of a cell phone's camera and overlaying plant species around detected leaves would represent an interesting augmented reality system, possible with faster and more precise local feature extraction.

Reviewing our work, we have presented an efficient automated plant recognition system, based on both shape and local features and the Random Forests classifier. We have used the ART descriptor as a new leaf shape feature and optimised its implementation. The proposed system outperforms state-of-the-art methods for scan-like leaf image based plant recognition while showing fast computational times and high scalability potential.

# Bibliography

- [A. 11] A. Criminisi, J. Shotton and E. Konukoglu, “Decision forests for classification, regression, density estimation, manifold learning and semi-supervised learning,” in *Microsoft Technical Report*, 2011. [Online]. Available: [http://research.microsoft.com/pubs/155552/decisionForests\\_MSR\\_TR\\_2011\\_114.pdf](http://research.microsoft.com/pubs/155552/decisionForests_MSR_TR_2011_114.pdf)
- [AL09] E. Aptoula and S. Lefèvre, “Morphological description of color images for content-based image retrieval,” in *Trans. Img. Proc.*, vol. 18, no. 11. Piscataway, NJ, USA: IEEE Press, Nov. 2009, pp. 2505–2517. [Online]. Available: <http://dx.doi.org/10.1109/TIP.2009.2027363>
- [BB08] A. R. Backes and O. M. Bruno, “Fractal and multi-scale fractal dimension analysis: a comparative study of Bouligand-Minkowski method,” in *Journal of Computer Science ISSN 1807-4545*, 2008. [Online]. Available: <http://arxiv.org/pdf/1201.3153.pdf>
- [BCB09] A. R. Backes, D. Casanova, and O. M. Bruno, “A complex network-based approach for boundary shape analysis,” in *Pattern Recognition*, vol. 42, no. 1. New York, NY, USA: Elsevier Science Inc., Jan. 2009, pp. 54–67. [Online]. Available: <http://dx.doi.org/10.1016/j.patcog.2008.07.006>
- [BCGM98] S. Belongie, C. Carson, H. Greenspan, and J. Malik, “Color- and texture-based image segmentation using EM and its application to content-based image retrieval,” in *Proceedings of the Sixth International Conference on Computer Vision*, 1998. [Online]. Available: <http://vision.ucsd.edu/sites/default/files/00710790.pdf>
- [BCJ<sup>+</sup>08] P. N. Belhumeur, D. Chen, D. Jacobs, W. J. Kress, R. Ramamoorthi, S. Sheorey, and L. Zhang, “Searching the world’s herbaria: a system for visual identification of plant species,” in *ECCV*, 2008, pp. 116–129.
- [BMP02] S. Belongie, J. Malik, and J. Puzicha, “Shape matching and object recognition using shape contexts,” in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2002. [Online]. Available: <http://www.cs.berkeley.edu/~malik/papers/BMP-shape.pdf>
- [BPK01] M. Bober, F. Preteux, and Y. Kim, “MPEG7 visual shape descriptors,” in *IEEE Transactions Circuits Systems*, 2001. [Online]. Available: <http://www.uk.mitsubishielectric-rce.eu/pubdocs/VIL01-D112A.pdf>

- [Bre84] L. Breiman, *Classification and regression trees*, ser. Wadsworth statistics/probability series. Wadsworth International Group, 1984. [Online]. Available: <http://books.google.de/books?id=uxPvAAAAMAAJ>
- [Bre96] —, “Bagging predictors,” in *Machine Learning*, 1996, pp. 123–140.
- [Bre01] —, “Random forests,” in *Machine Learning*, 2001, pp. 5–32.
- [BTG06] H. Bay, T. Tuytelaars, and L. V. Gool, “SURF: Speeded up robust features,” in *In ECCV*, 2006, pp. 404–417.
- [CB84] R. Chellappa and R. Bagdazian, “Fourier coding of image boundaries,” in *IEE Trans. PAMI-6(1)*, 1984, pp. 102–105.
- [CDF<sup>+</sup>04] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray, “Visual categorization with bags of keypoints,” in *In Workshop on Statistical Learning in Computer Vision, ECCV*, 2004, pp. 1–22.
- [CFB11] D. Casanova, J. B. Florindo, and O. M. Bruno, “IFSC/USP at ImageCLEF 2011: plant identification task,” in *CLEF (Notebook Papers/Labs/Workshop)*, V. Petras, P. Forner, and P. D. Clough, Eds., 2011.
- [CTM<sup>+</sup>11] G. Cerutti, L. Tougne, J. Mille, A. Vacavant, and D. Coquin, “Guiding active contours for tree leaf segmentation and identification,” in *Cross-language Evaluation Forum*, Sep. 2011. [Online]. Available: <http://liris.cnrs.fr/publis/?id=5374>
- [FG87] M. A. Föstner and E. Gülch, “A fast operator for detection and precise location of distinct points, corners and centers of circular features,” in *ISPRS Intercommission Workshop*, Interlaken, Switzerland, 1987.
- [FHST05] G. Finlayson, S. Hordley, G. Schaefer, and G. Y. Tian, “Illuminant and device invariant colour using histogram equalisation,” in *Pattern Recognition*, vol. 38, no. 2, 2005, pp. 179 – 190. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0031320304001852>
- [FQ03] J. Fang and G. Qiu, “Human face detection using angular radial transform and support vector machines,” in *ICIP (1)'03*, 2003, pp. 669–672.
- [GBJ<sup>+</sup>11] H. Goeau, P. Bonnet, A. Joly, N. Boujemaa, D. Barthelemy, J.-F. Molino, P. Birnbaum, E. Mouysset, and M. Picard, “The CLEF 2011 plant images classification task,” in *ImageCLEF 2011*, 2011. [Online]. Available: <http://hal.inria.fr/docs/00/64/21/97/PDF/plant-retrieval-overview-.pdf>
- [GJY<sup>+</sup>11] H. Goeau, A. Joly, I. Yahiaoui, P. Bonnet, and E. Mouysset, “Participation of INRIA & Pl@ntNet to ImageCLEF 2011 plant images classification task,” in *Working notes of CLEF 2011 conference*, 2011.
- [Har54] Z. Harris, “Distributional structure,” in *Word*, vol. 10, no. 23, 1954, pp. 146–162.
- [Ho95] T. K. Ho, “Random decision forests,” in *Proceedings of the Third International Conference on Document Analysis and Recognition - Volume 1*, ser. ICDAR '95. Washington, DC, USA: IEEE Computer Society, 1995, pp. 278–. [Online]. Available: <http://dl.acm.org/citation.cfm?id=844379.844681>

- [HS88] C. Harris and M. Stephens, “A combined corner and edge detector,” in *In Proc. of Fourth Alvey Vision Conference*, 1988, pp. 147–151.
- [JB11] A. Joly and O. Buisson, “Random maximum margin hashing,” in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. Los Alamitos, CA, USA: IEEE Computer Society, 2011, pp. 873–880.
- [KK99] W. Y. Kim and Y. S. Kim, “A new region-based shape descriptor,” in *ISO/IEC MPEG99/M5472*, 1999.
- [KNSS11] A. Kadir, L. E. Nugroho, A. Susanto, and P. I. Santosa, “A comparative experiment of several shape methods in recognizing plants,” in *International Journal of Computer Science & Information Technology*, 2011. [Online]. Available: <http://arxiv.org/ftp/arxiv/papers/1110/1110.1509.pdf>
- [KSP95] H. Kauppinen, T. Seppänen, and M. Pietikäinen, “An experimental comparison of autoregressive and Fourier-based descriptors in 2D shape classification,” no. 17, pp. 201–207, 1995.
- [LJ05] H. Ling and D. W. Jacobs, “Using the inner-distance for classification of articulated shapes,” in *Conference on Computer Vision and Pattern Recognition.*, 2005. [Online]. Available: <http://www.wisdom.weizmann.ac.il/~bagon/CVspring07/files/cvpr05-innerdist-6.pdf>
- [LJ08] —, “Shape classification using the inner-distance,” in *ECCV 2008, Part IV, LNCS 5305*, 2008. [Online]. Available: [http://www.cs.umd.edu/~djacobs/pubs\\_files/ID-pami-8.pdf](http://www.cs.umd.edu/~djacobs/pubs_files/ID-pami-8.pdf)
- [Low99] D. Lowe, “Object recognition from local scale-invariant features,” in *International Conference on Computer Vision*, 1999, pp. 1150–1157.
- [Low03] —, “Distinctive image features from scale-invariant keypoints,” in *International Journal of Computer Vision*, 2003, pp. 91–110.
- [MCUP02] J. Matas, O. Chum, M. Urban, and T. Pajdla, “Robust wide baseline stereo from maximally stable extremal regions,” in *In British Machine Vision Conference*, 2002, pp. 384–393.
- [Ots79] N. Otsu, “A threshold selection method from gray-level histograms,” in *IEEE Trans. Systems, Man and Cybernetics*, 1979.
- [Qui86] J. R. Quinlan, “Induction of decision trees,” pp. 81–106, 1986.
- [RCB04] J. Ricard, D. Coeurjolly, and A. Baskurt, “Generalizations of angular radial transform for 2D and 3D shape retrieval,” no. RR-LIRIS-2004-023, 2004. [Online]. Available: <http://liris.cnrs.fr/publis/?id=1932>
- [Sod01] O. Soderkvist, “Computer vision classification of leaves from Swedish trees,” 2001.
- [TR94] D. Thomas and W. Rauber, “Two-dimensional shape description,” in *Technical Report: GR UNINOVA-RT-10-94*, 1994.

- [WBX<sup>+</sup>07] S. G. Wu, F. S. Bao, E. Y. Xu, Y.-X. Wang, Y.-F. Chang, and Q.-L. Xiang, “A leaf recognition algorithm for plant classification using probabilistic neural network,” in *IEEE International Symposium on Signal Processing and Information Technology*, 2007. [Online]. Available: <http://arxiv.org/pdf/0707.4289.pdf>
- [WN11] K. O. Wahdan, O.M. and M. Nasrudin, “Logo recognition system using angular radial transform descriptors,” in *Journal of Computer Science*, 2011, pp. 1416–1422. [Online]. Available: <http://thescipub.com/abstract/10.3844/jcssp.2011.1416.1422>
- [YAT11] B. A. Yanikoglu, E. Aptoula, and C. Tirkaz, “Sabanci-Okan system at ImageClef 2011: plant identification task,” in *CLEF (Notebook Papers/Labs/Workshop)*, V. Petras, P. Forner, and P. D. Clough, Eds., 2011. [Online]. Available: <http://dblp.uni-trier.de/db/conf/clef/clef2011w.html#YanikogluAT11>
- [ZL01] D. Zhang and G. Lu, “A comparative study on shape retrieval using Fourier descriptors with different shape signatures,” in *\**, 2001. [Online]. Available: [http://knight.temple.edu/~lakamper/courses/cis601\\_2008/etc/fourierShape.pdf](http://knight.temple.edu/~lakamper/courses/cis601_2008/etc/fourierShape.pdf)
- [ZR72] C. T. Zahn and R. Z. Roskies, “Fourier descriptors for plane closed curves,” in *IEEE Transactions on Computers*, no. 3, 1972, pp. 269–281. [Online]. Available: <http://www.cs.jhu.edu/~misha/Papers/Zahn72.pdf>